Brain-inspired and energy efficient solutions to hard optimization problems

Gabriel Andres Fonseca Guerra

Research Scientist at Intel's Neuromorphic Computing Lab

April 17, 2023

High Performance Computing for Sustainable Development in Government, Academia and Industry' - opening day talks, Stellenbosch - South Africa

Agenda

1. Application relevance and motivation.
2. Brain-inspired computing: Loihi architecture and systems.
3. Lava open-source development framework
4. Graph algorithms are well suited for neuromorphic computing .
5. Lava-optimization architecture
6. Optimization problems taxonomy.
7. How does the search for solutions work?
8. Increased performance efficiency for solving hard optimization problems
9. INRC community

Hard constraint optimization problems are ubiquitous in real world

Industries

- Finance
- Electricity
- Communi-cation
- Robotics & AI
- Agriculture
- Science
- Insurance
- Logistics
- Advertise-ment
- ...

Example applications

Trajectory optimization
Coordinating mobile assembly robots
Model predictive control for robots
Decision optimization

Drug discovery
Prototype design
Material design
Particle jet reconstruction

Route optimization
Supply chain design
Inventory management
Flight gate assignment
Train scheduling

Inherent constraint optimization problems

- Perceive and infer state of the environment
- Objective: Find shortest path around obstacles, …
- Constraints: No collisions, comfortable acceleration, …

- Objective: Largest number train route assignments
- Constraints: Railroad network capacity

Brain-inspired computing | Brains remain unrivaled computing devices

AUTONOMOUS DRONE

CPU/GPU controller
Power: 18,000 mW
Mass: ~40 grams

Pre-trained to fly between known gates at walking pace

Can't learn anything online

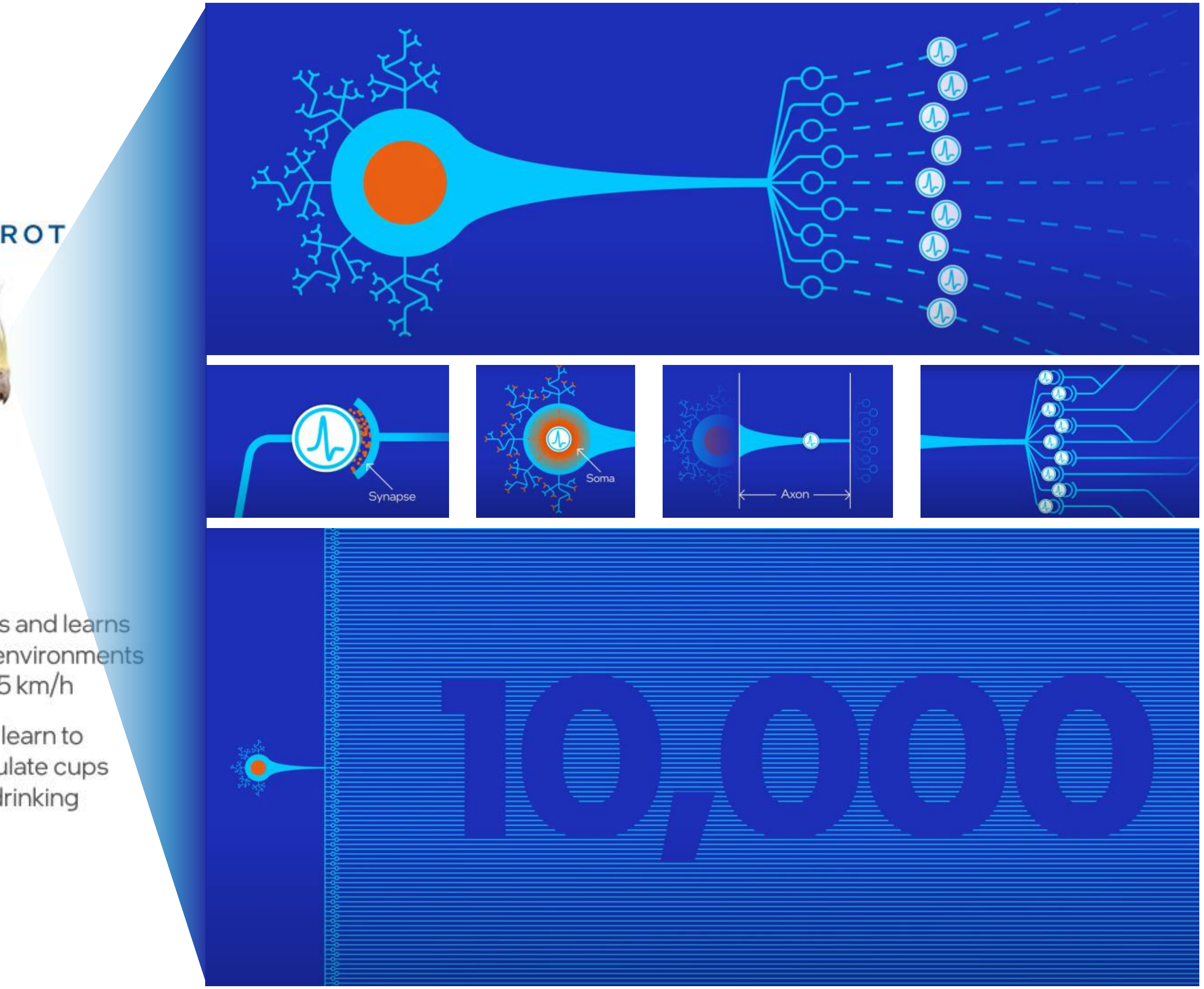
COCKATIEL PARROT

Brain
Power: 50 mW
Mass: 2.2 grams

Navigates and learns unknown environments at 35 km/h

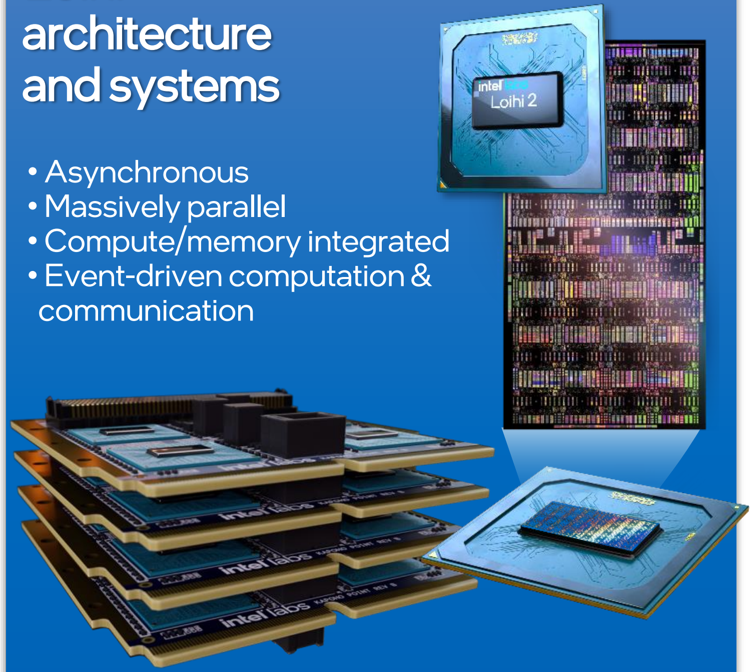
Can learn to speak English words

Can learn to manipulate cups for drinking



Pioneering a new class of computer architecture

Loihi architecture and systems



- Asynchronous
- Massively parallel
- Compute/memory integrated
- Event-driven computation & communication

← Scalable →

Edge

Cloud

SW Products

github.com/lava-nc

Applications, Products, Services

lava-optim library

DL	Optim	VSA	...

Algorithm libraries

LAVA Software Framework

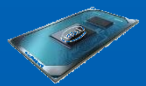
- Event-based
- Multi-Paradigm
- Multi-Abstraction
- Multi-Platform
- Open-source

API

Compiler

Runtime

Heterogenous hardware interface



CPU

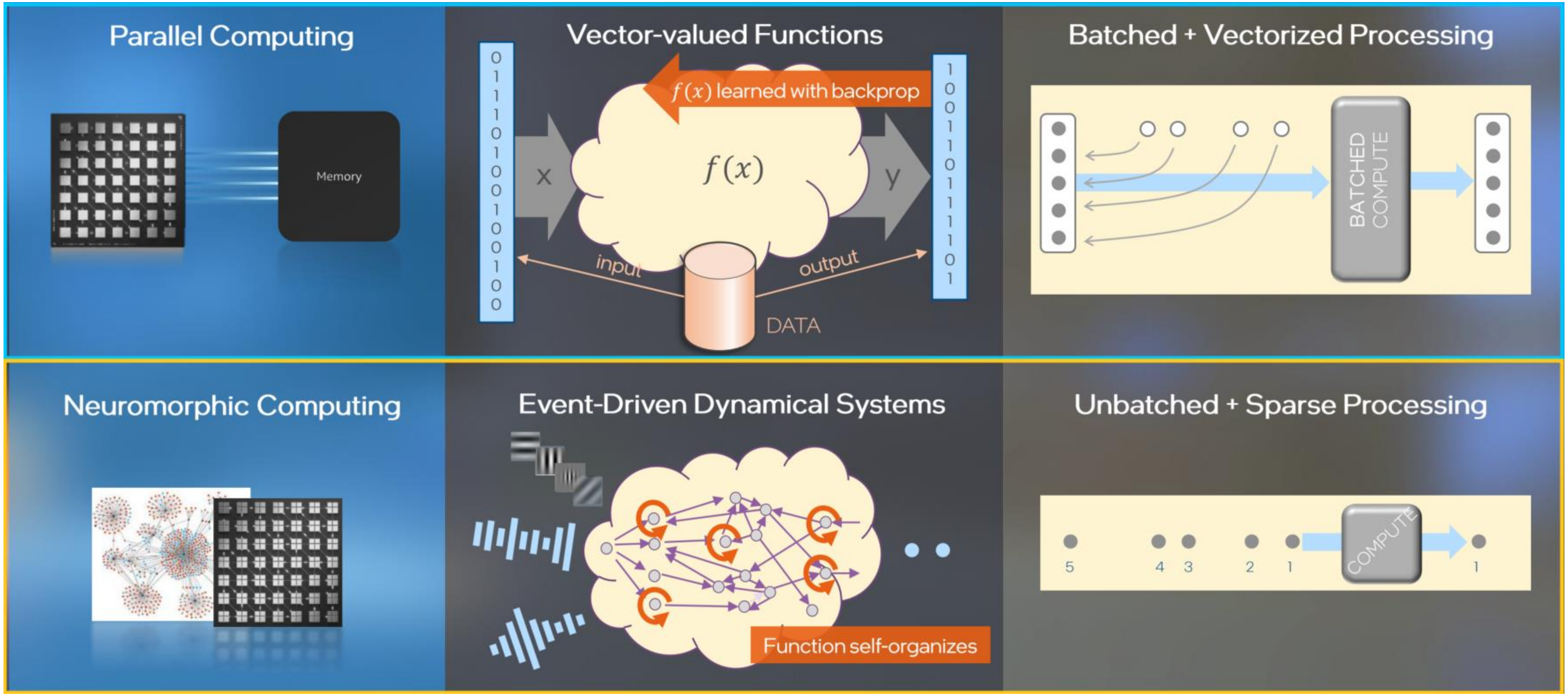
Loihi

GPU

Diverse Neuromorphic Research Community

- Growing community
- Access to Intel neuromorphic systems
- Community-driven workshops
- 20+ publications/ year

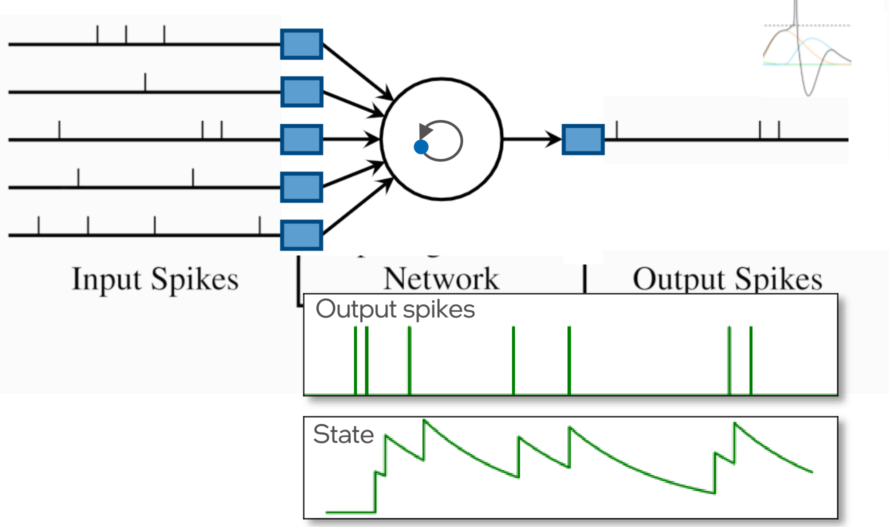
Motivates a fundamentally different kind of computing



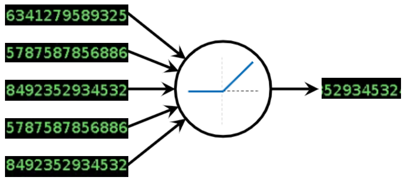
Exploiting dynamics at the neuron level

Maximize computation by minimizing data movement

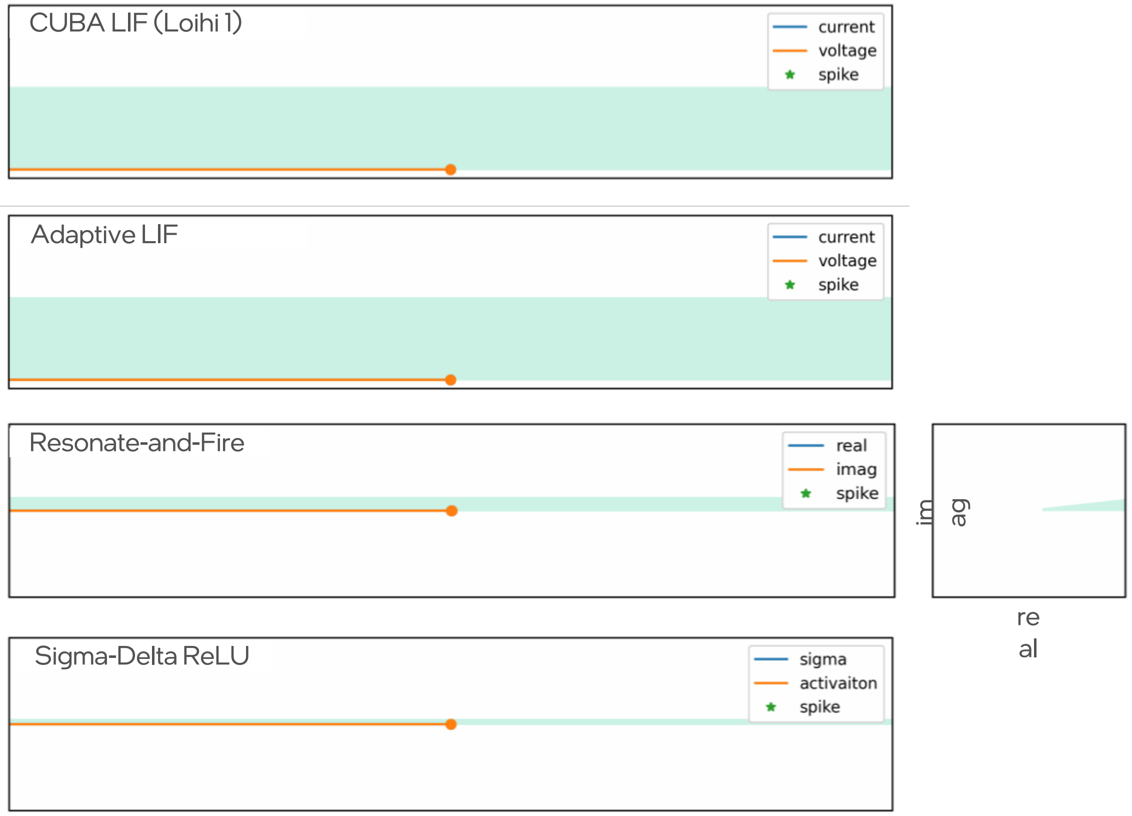
Spiking Neuron (Nonlinear Filter)



Artificial Neuron (Stateless)



Programable spiking dynamical systems



Realized in Loihi

improved in Loihi 2

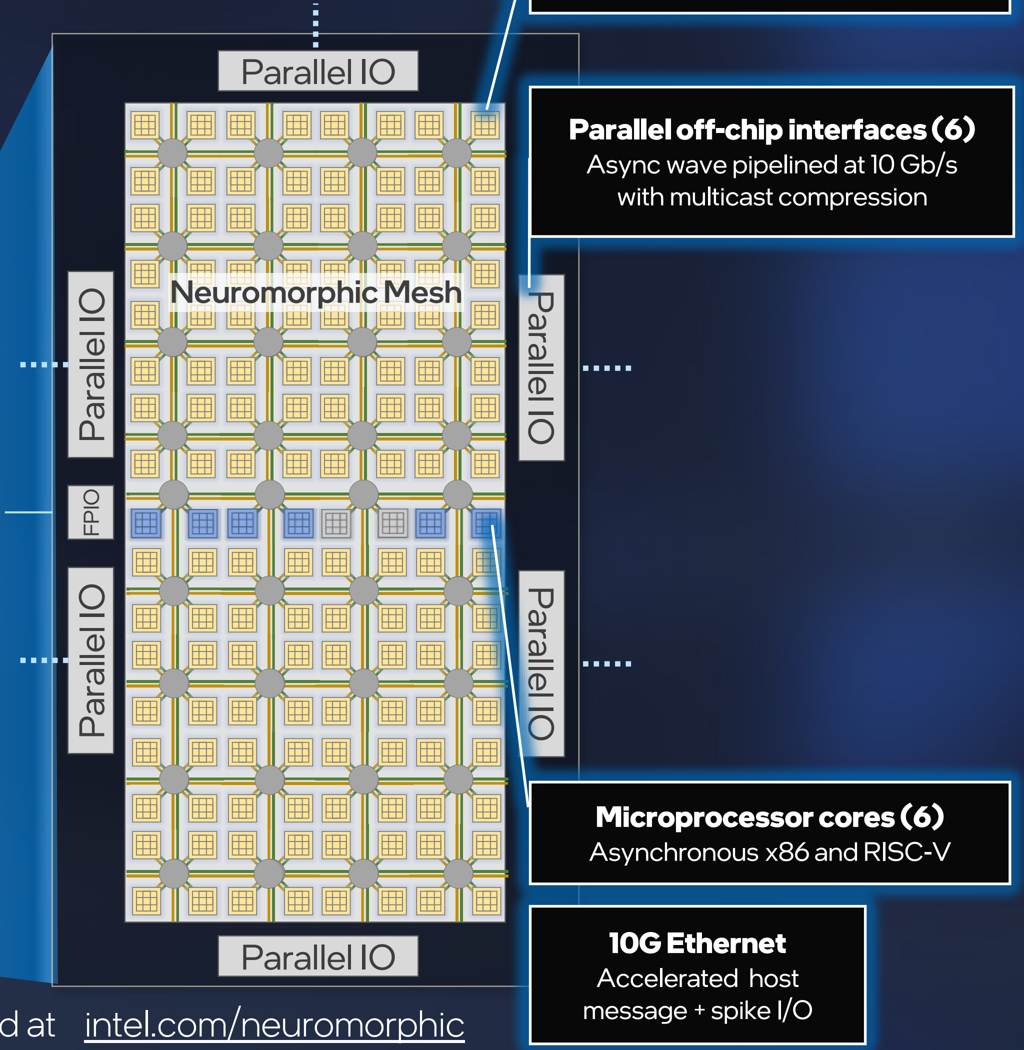
Compute and memory integrated to spatially embody programmed networks
Temporal neuron models (LIF) to exploit temporal correlation
Spike-based communication to exploit temporal sparsity
Sparse connectivity for efficient dataflow and scalability
On-chip learning without weight movement or data storage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

Yet...

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

- Up to 10x faster processing capability*
- Up to 60x more inter-chip bandwidth*
- Up to 1 million neurons with 15x greater resource density*
- Programmable neurons
- Graded spikes
- 3-Factor learning
- 10G Ethernet I/F to host

Loihi 2



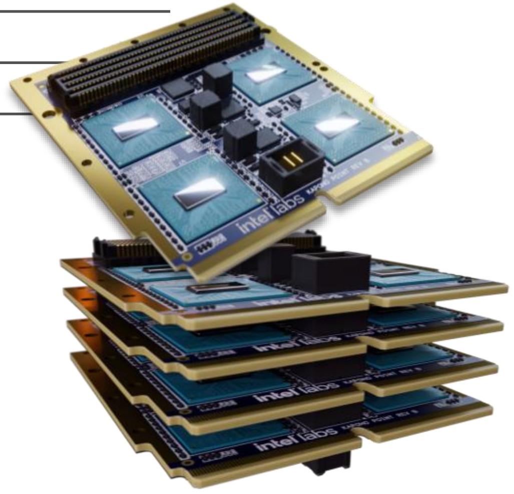
* specs and configuration details can be found at intel.com/neuromorphic

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

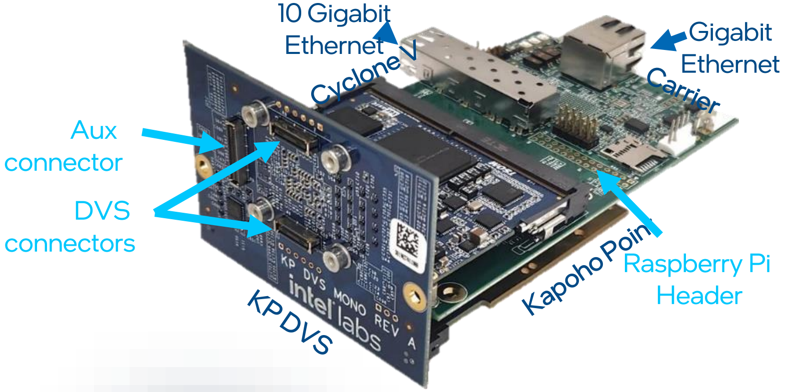
Loihi Systems

Kapoho Point

	Key Properties
Number of chips	8
Max neurons	8.1 M
Max synapses	960 M
Interfaces	GbE via host board 10 GbE direct to Loihi MIPI, GPIO, AER, SHS via interface board
Dimensions	79 mm x 69 mm x 15 mm
Weight	108g
Power supply	12 V
Typical Power	2.4 - 3.6W at 1ms timesteps*



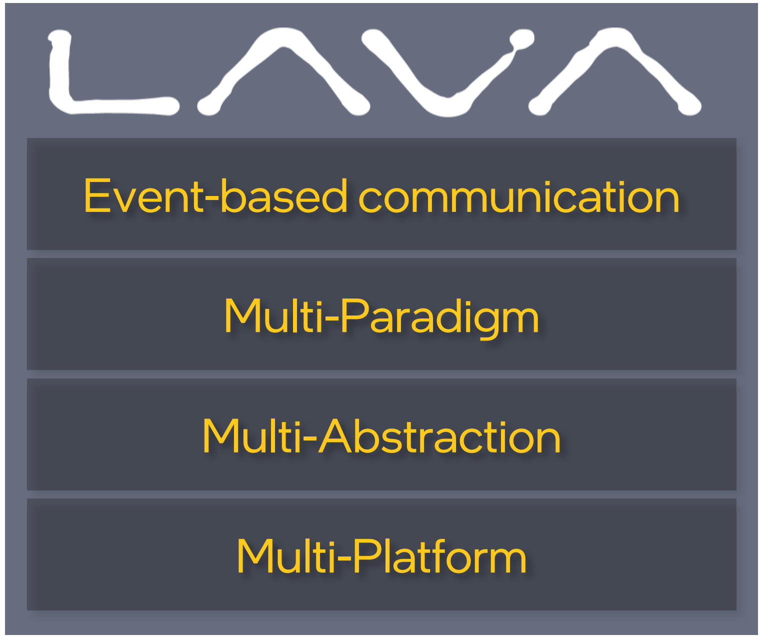
Kapoho Point System

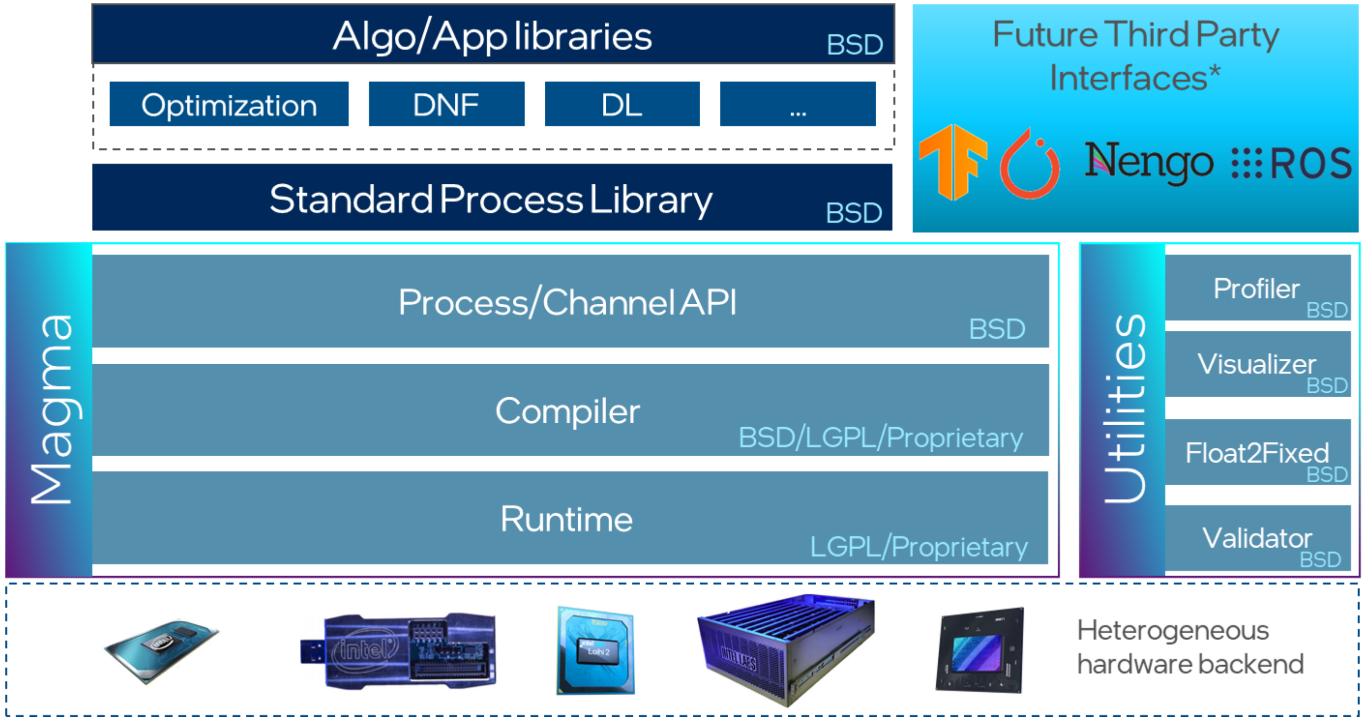


Pohoiki Springs

Lava open-source development framework

- Seeded by Intel but open-source and increasingly community-driven
- Full SW stack from runtime, to compiler, to powerful algorithm/application libraries
- Brain-inspired programming model for heterogeneous HW
  - Parallel & Asynchronous
  - Event-based computation/communication





* specs and configuration details can be found at intel.com/neuromorphic

Mission

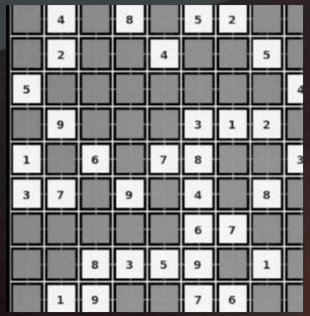
- Converge neuromorphic SW development towards open standard
- Make *exotic* neuromorphic systems accessible to non-expert developers
- Accelerate adoption of neuromorphic technologies
- Enable orders of magnitude gains in compute efficiency

Objective

- Facilitate real-world application development for neuromorphic systems

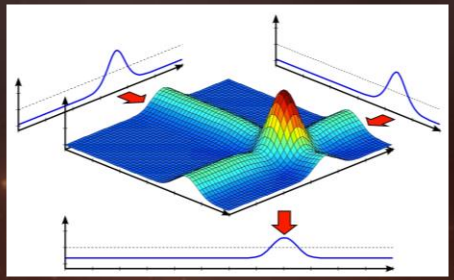
LAVA application libraries

Optimization
lava-optimization



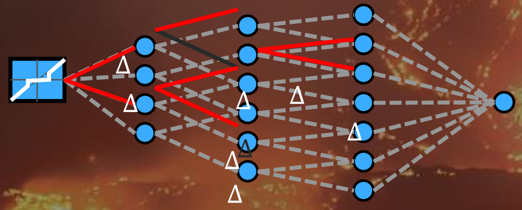
Family of constraint optimization solvers
Today: QP, QUBO
Future: MPC, LCA, ILP, …
Standalone use or as part of AI applications

Neural Attractors
lava-dnf



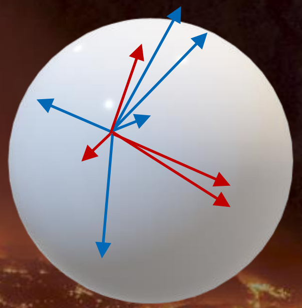
Design models with attractor dynamics
Stabilize temporal data
Selective data processing
Dynamic working memories

Deep Learning
lava-dl



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

Vector Symbolic
lava-vsa

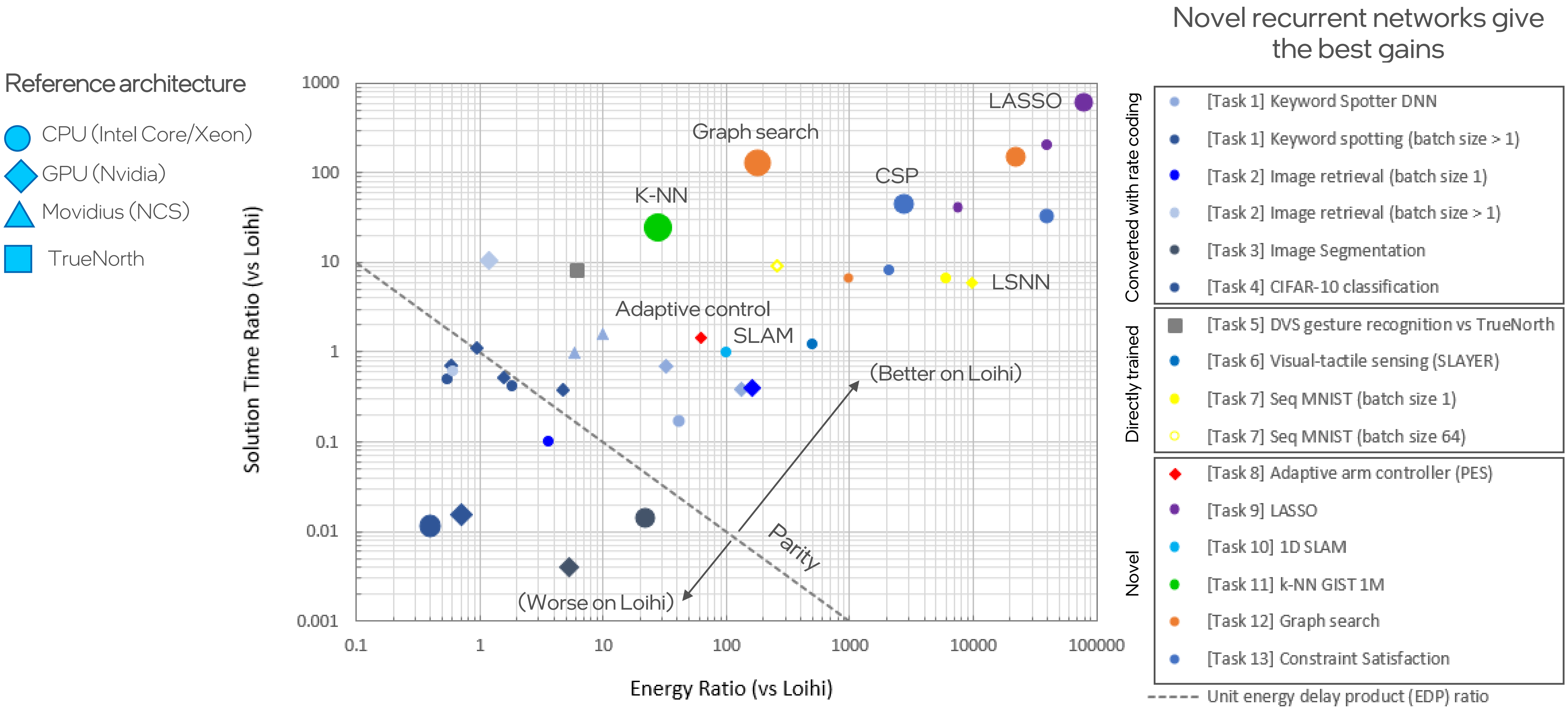


API for algebraic model description for VSAs
Library of data types and operations (composition, binding, factorization, …)

Future directions

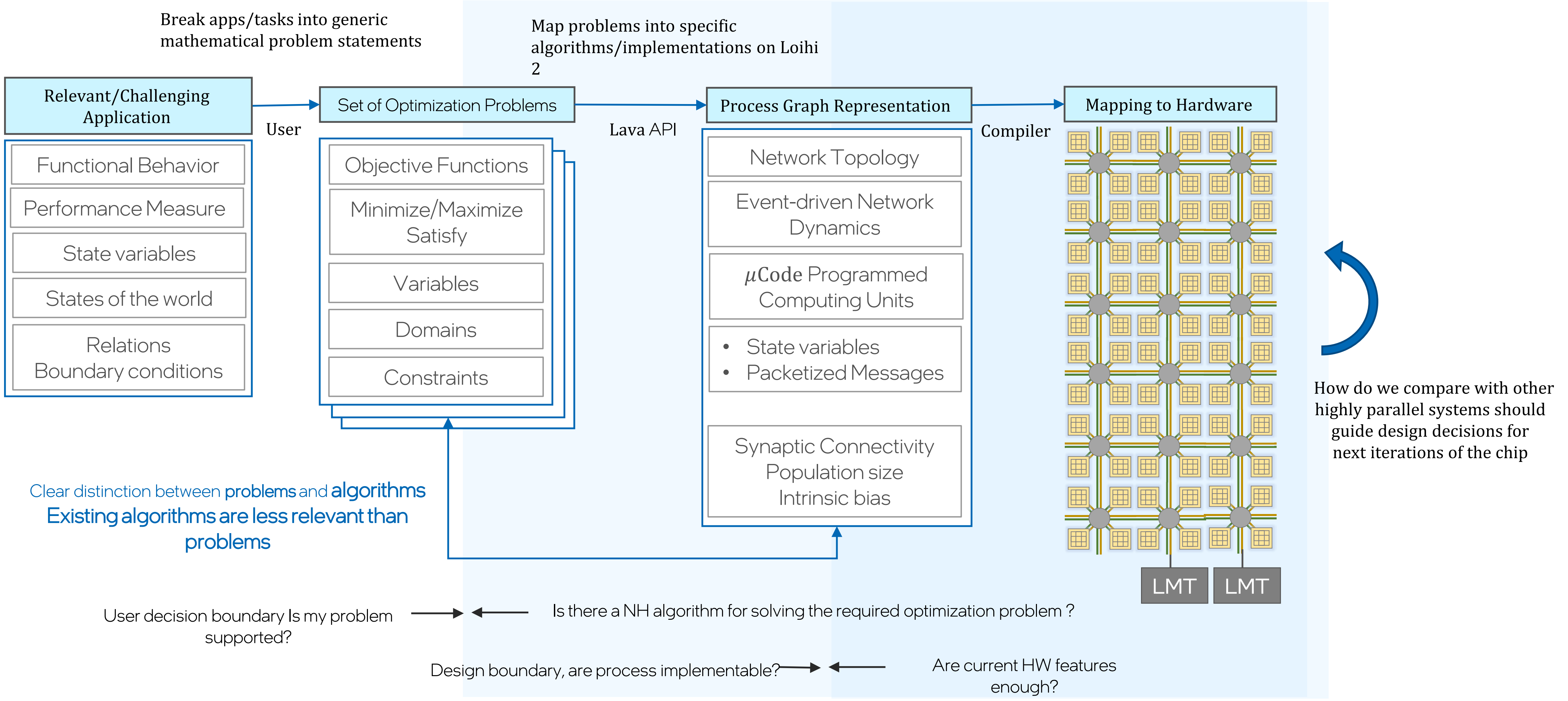
- lava-io (sensor/actuator interfaces)
- lava-robotics (control, planning, physical simulator interfaces)
- lava-evolve (evolutionary training methods)
- lava-ui (graphical network creation, visualization, debugging)
- Signal processing
- Off-the-shelf apps (segmentation, tracking, keyword detection, …)
- Neural simulators (Brian2Lava, …)

For the right workloads, orders of magnitude gains in latency and energy efficiency are achievable



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

Lava-optimization, from application to hardware



Taxonomy of optimization problems we envision to support

Consider arbitrary functions $f: X \mapsto \mathbb{R}^n$, $g: Y \mapsto \mathbb{R}^m$ and $h: Z \mapsto \mathbb{R}^q$.

Optimization : $\min_{\mathbf{x}}\{f(\mathbf{x})\}$

Constrained Optimization: $\min_{x}\{f(\mathbf{x}) |\, g_i(\mathbf{x}) \leq \mathbf{b}, h_i(\mathbf{x}) = \mathbf{c}\}$ ⟵ Include equality and inequality constraints:

Linear (LP)

$f(\mathbf{x}) = \mathbf{c}^{\mathrm{T}}\mathbf{x} + \mathbf{b}$

$g_i(\mathbf{x}) = \mathbf{Ax}$

$h(\mathbf{x}) = \mathbf{Bx}$

ILP: Integer Linear Programming

$\mathbf{x} \in \mathbb{Z}^n$

MILP : Mixed-Integer Linear Programming

$f(\mathbf{x}) = \mathbf{c}_1^{\mathrm{T}}\mathbf{x}_1 + \mathbf{c}_2^{\mathrm{T}}\mathbf{x}_2$

$g_i(\mathbf{x}) = \mathbf{A}_2\mathbf{x}_1 + \mathbf{A}_2\mathbf{x}_2$

$\mathbf{x}_2 \in \mathbb{Z}^n$

$\mathbf{x}_2 \in \mathbb{R}^n$

Combinatorial Optimization:

$dom(f) = \{X_i | 1 \leq i \leq k\}$

CSP $f(\mathbf{x}) = \mathrm{K}$

- f: cost function of the state of a system.
- h and g:
  - constraint both the values of state variables and mutual values of state variables

Nonlinear Programming

$f(x) \neq \mathbf{c}^{\mathrm{T}}\mathbf{x} + \mathbf{b}$

QCQP : quadratically constrained quadratic program

$\min_{x}\{\mathbf{x}^{\mathrm{T}}\mathbf{Qx} + \mathbf{c}^{\mathrm{T}}\mathbf{x} |\, \mathbf{x}^{\mathrm{T}}\mathbf{Ax} + \mathbf{Bx} \leq \mathbf{b}, \mathbf{Cx} = \mathbf{d}\}$

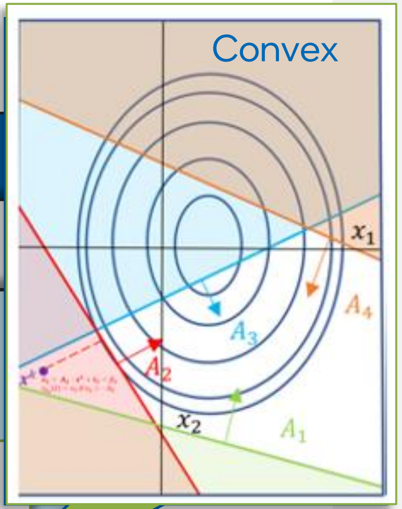
QP: Quadratic Programming

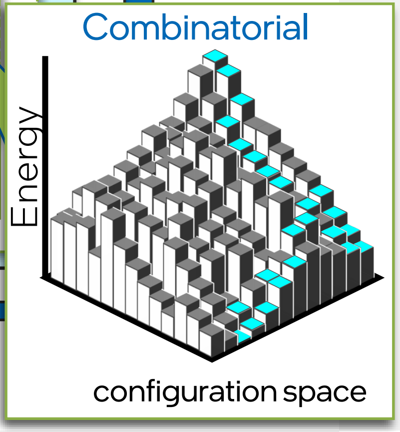
$\min_{x}\{\mathbf{x}^{\mathrm{T}}\mathbf{Qx} + \mathbf{c}^{\mathrm{T}}\mathbf{x} |\, \mathbf{Ax} \leq \mathbf{b}, \mathbf{Bx} = \mathbf{c}\}$

QUBO : Quadraticc Unconstrained Binary Optimization $\min_{x}\{\mathbf{x}^{\mathrm{T}}\mathbf{Qx} | \mathbf{x} \in \{0,1\}^n\}$

Mixed-Integer Quadratic Programming (MIQP)

$\min_{x}\{\mathbf{x}^{\mathrm{T}}\mathbf{Qx} + \mathbf{c}^{\mathrm{T}}\mathbf{x} |\, \mathbf{Ax} \leq \mathbf{b}, \mathbf{Bx} = \mathbf{c}, \mathbf{x} \in \mathbb{Z}^n\}$

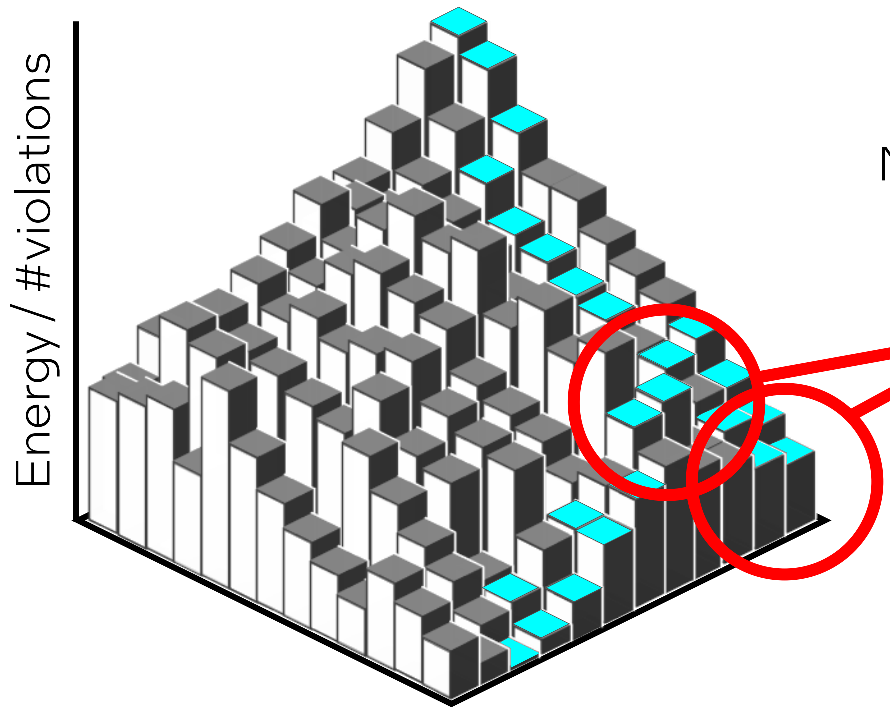




$K \mapsto$ Constant; $\mathbf{c} \mapsto$ First order coefficients vector; $\mathbf{Q} \mapsto$ second order coefficients matrix

The Loihi 2 QUBO solver

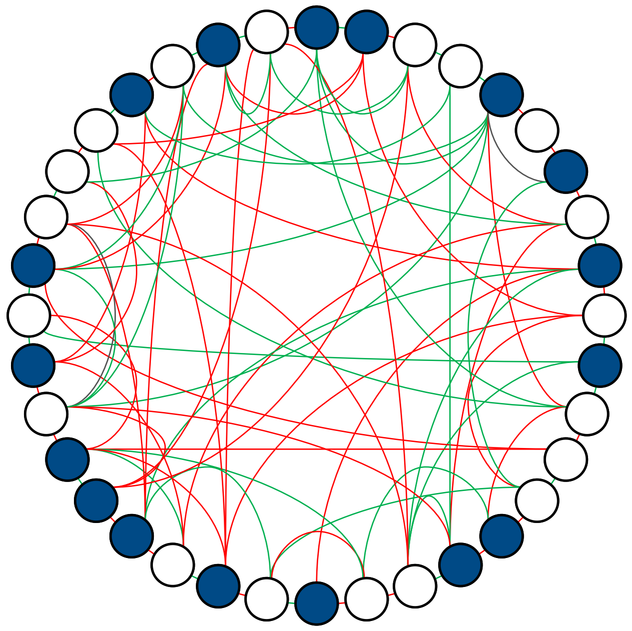
Accelerated by asynchronous parallelism



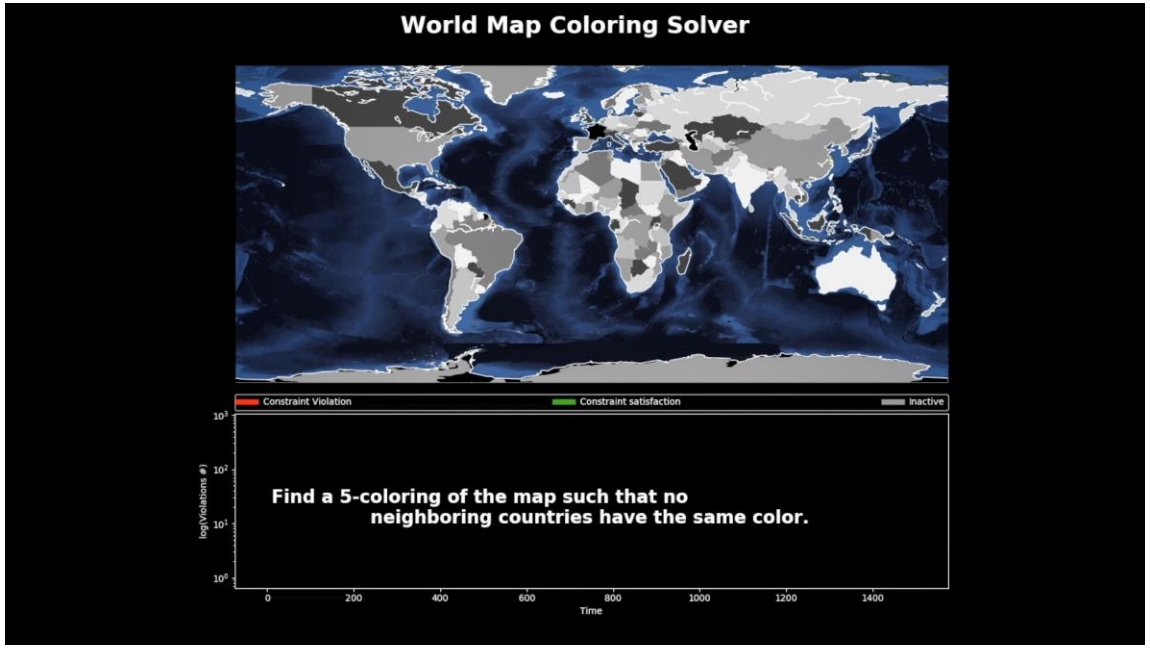
Neural dynamics descend the gradient

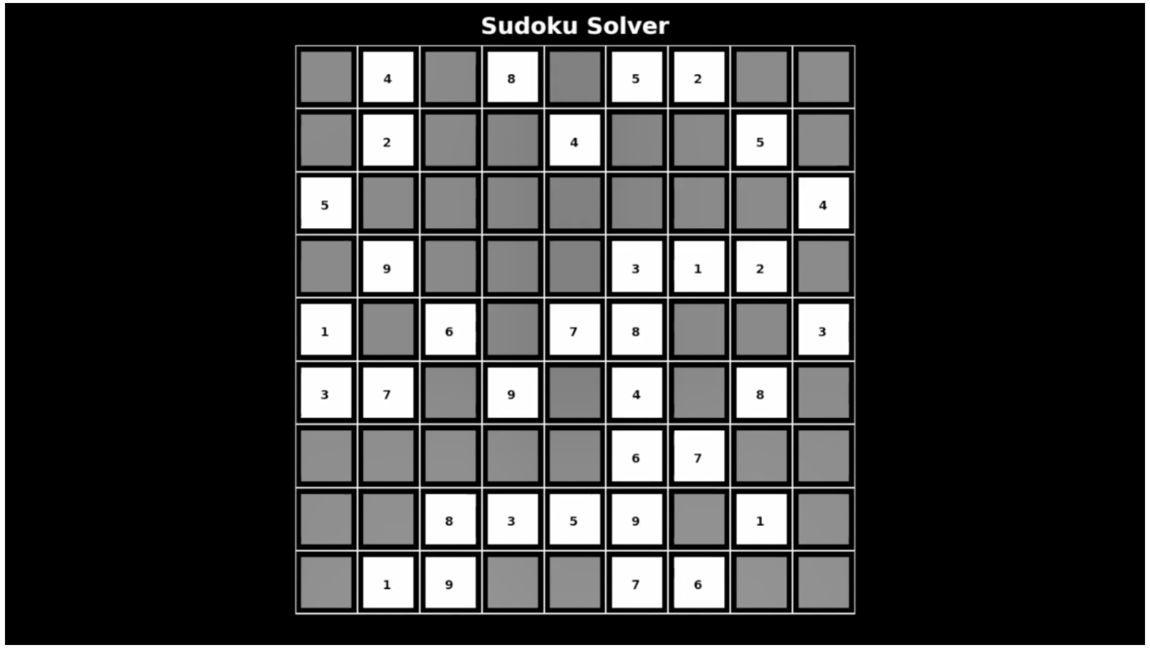
Local minimum escaped by stochastic spiking dynamics

Efficient descent due to massively parallel, asynchronous neuromorphic computing architecture



Examples: map coloring and sudoku





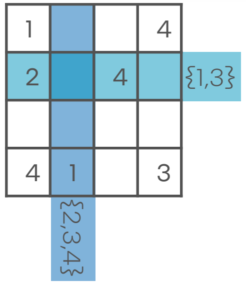
Loihi outperforms standard optimization solvers by orders of magnitude

CSP (Latin Square)

Workload:
Each number occurs exactly once per column/row

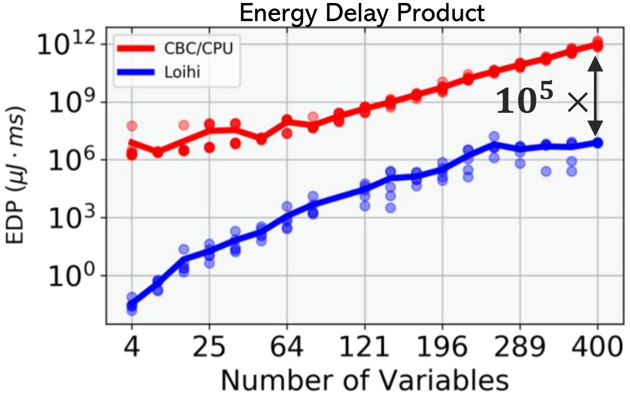
Relevance:

- Proof-of-principle
- NP complete



Solving a 20x20 Latin Square problem:

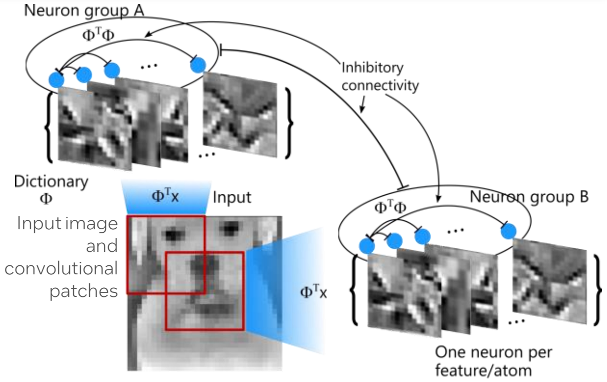
```
#Define the problem
size = 20
latin_squares = translateLatinSquareToCsp(size=size)

#Instantiate a solver for the optimization workload.
csp_solver = CspSolver(problem=latin_squares)

# Solve the problem.
solution = csp_solver.solve()
```

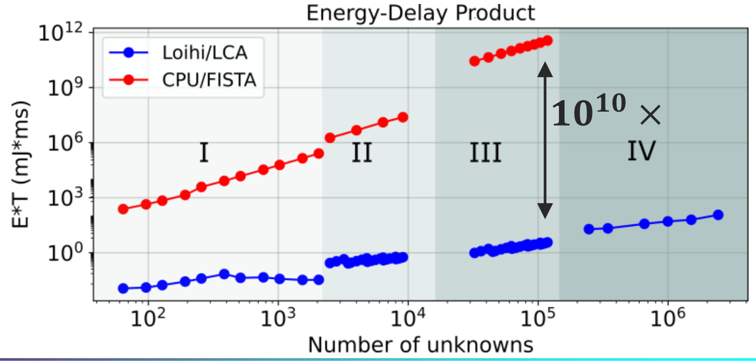


Quadratic Programming (Convolutional Sparse Coding)

Workload:
Find sparse representation of data given overcomplete features dictionary

Relevance:
Compression of information required to store a dataset





For details see: Davies, Mike, et al. "Advancing neuromorphic computing with Loihi: A survey of results and outlook." Proceedings of the IEEE 109.5 (2021): 911-934

Loihi: Nahuku board running NxSDK 0.95 with an Intel Core i7-9700K host with 128GB RAM, running Ubuntu 16.04.6 LTS
CBC/CPU: Intel Core i7-9700K; and RAM: 128 GB, running Ubuntu 16.04.6 LTS running CBC (www.coin-or.org/projects)

Loihi: Wolf Mountain board with NxSDK v0.75
FISTA/CPU: Intel Core i7-4790 3.6-GHz w/ 32-GB RAM, BIOS: AMI F5. OS: Ubuntu 16.04 with Hyper Threading disabled, running SPAMS solver for FISTA (http://spams-devel.gforge.inria.fr)

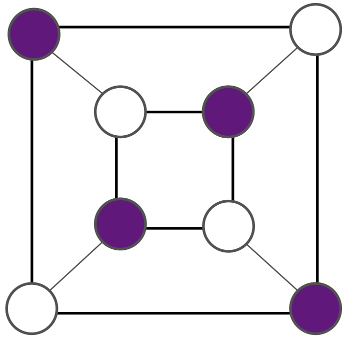
Performance results are based on testing as of September 2021 and may not reflect all publicly available security updates. Results may vary.

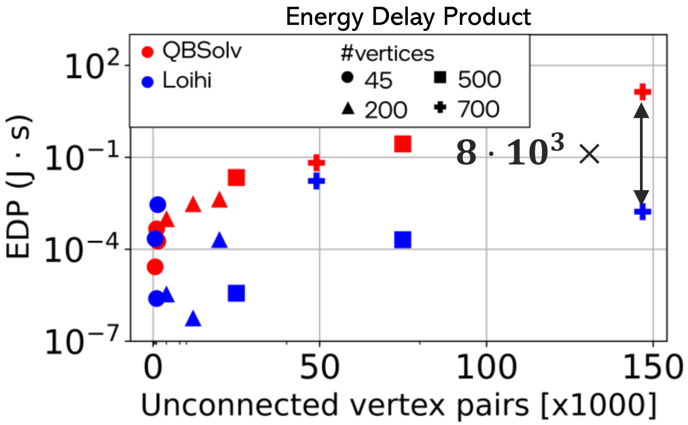
Loihi outperforms standard optimization solvers by orders of magnitude

QUBO (Maximum Independent Set)

Workload:
Find largest set of unconnected vertices

Relevance:
- Target of SOTA quantum annealing approaches
- NP hard



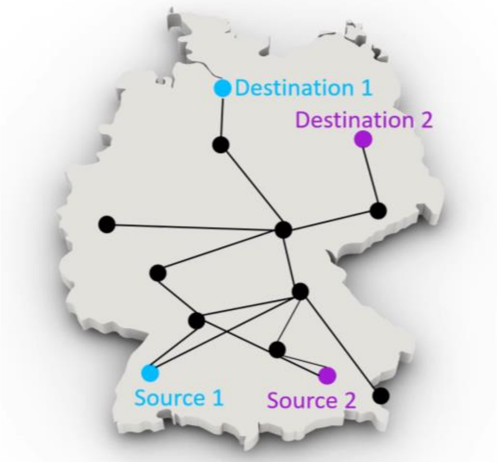


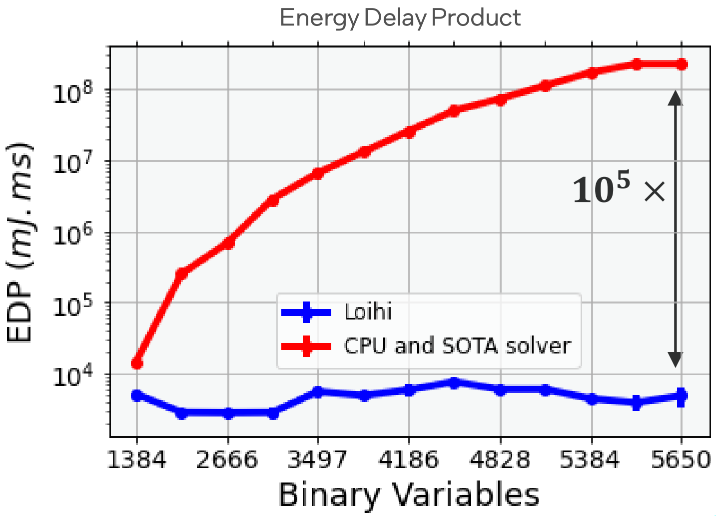
Integer Linear Programing (Train Scheduling)

In collaboration with: DB

Workload:
Find the largest possible set of route assignments, given customer requests and railway, time and train constraints.

Relevance:
- Large-scale, real-world use case
- Applicable to resource allocation in warehouses and production lines.





Loihi: Nahuku board running NxSDK 0.95 with an Intel Core i7-9700K host with 128GB RAM, running Ubuntu 16.04.6 LTS
QUBO-QBSolv/CPU: benchmarks ran on an Intel Xeon CPU E5-2699 v3 @ 2.30GHz with 32GB DRAM (https://github.com/dwavesystems/qbsolv)
ILP-CPU: Commercial solver running on Linux64 with 16 processor cores.

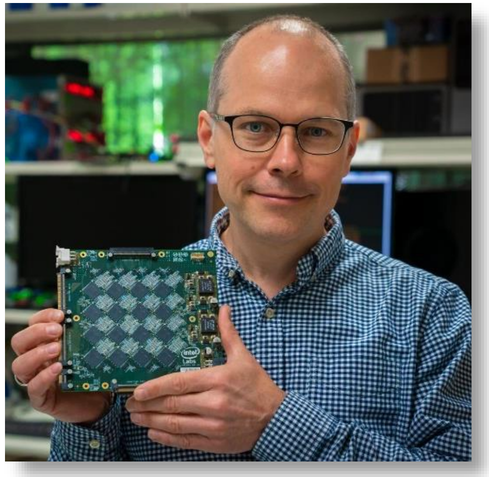
Performance results are based on testing as of September 2021 and may not reflect all publicly available security updates. Results may vary.

INRC community

Launched five years ago by Intel Labs

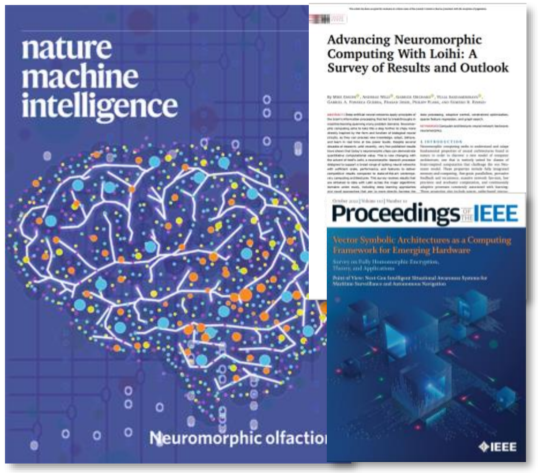
Our mission: Pioneer a new programmable computing technology inspired by a modern understanding of the brain

Access to Intel neuromorphic systems and tools



Community-driven workshops and training events

Producing 20+ publications per year



Growing community with 180+ member groups

Image: intel.com/content/www/us/en/research/neuromorphic-community.html

Legal Information

Performance varies by use, configuration and other factors. Learn more at www.Intel.com/PerformanceIndex.

Performance results are based on testing as of dates shown in configurations and may not reflect all publicly available updates. See backup for configuration details. No product or component can be absolutely secure.

Your costs and results may vary.

Results have been estimated or simulated.

Intel technologies may require enabled hardware, software or service activation.

Intel does not control or audit third-party data. You should consult other sources to evaluate accuracy.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

Backup

Additional information for Q/A

Summary

Optimization problems are ubiquitous in a wide range of technological and industrial applications, from efficient scheduling of package delivery to robotic control to drug discovery. Order of magnitude improvements in the energy efficiency of solving optimization problems have thus the potential to make computing systems more sustainable and environmentally friendly.

In this talk, I will present optimization algorithms and solvers developed for Intel's Loihi research chip, an event-driven, massively and fine-grained parallel hardware inspired by the efficiency of the neural dynamics of biological brains (neuromorphic). Benchmarking results on those solvers show how Loihi enables a competitively fast solution to constraint satisfaction, quadratic unconstrained binary optimization and quadratic programming problems, with orders-of-magnitude advantage in energy consumption when compared with conventional solvers running on standard CPU architectures.

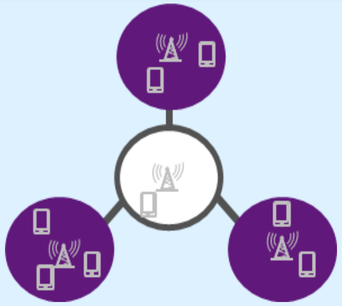
These neuromorphic algorithms take the form of a network of distributed discrete dynamical systems that communicate asynchronously through small binary or integer messages. The global dynamics of such event-driven algorithms explore the state space defined by the input problem simultaneously minimizing a quantity defined by the problem's objective function and maximizing the satisfaction of requirements defined by the problem constraints. The spatiotemporal sparsity resulting from such a computing paradigm is what enables the remarkable energy efficiency of neuromorphic optimization

Application relevance

Motivation

Applications of MIS

Resource allocation



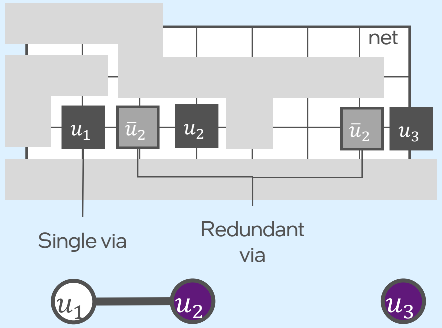
In wireless communication:

Access points and sensors are grouped into small networks.

MIS divides the networks into interference-free iso-frequency groups.

Zhou, J., et al., Q. Efficient graph-based resource allocation scheme using maximal independent set for randomly-deployed small star networks. Sensors 17, 2553 (2017).

Intra-chip communication



In chip design:

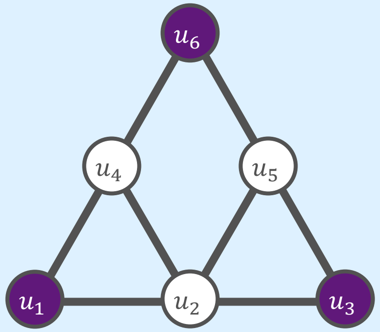
A via connects two adjacent net segments.

Redundant vias are inserted to tolerate single via failures.

The MIS determines the most reliable chip design with the maximum number of redundant vias.

Lee, K. Y. & Wang, T. C. Post-routing redundant via insertion for yield/reliability improvement. In Proceedings of the 2006 Asia and South Pacific Design Automation Conference, 303–308, 2006.

Error-correcting code



In coding theory:

Binary vector(vertices) are joined by an edge if both vectors may end up in the same state due to errors e.g., deletion or transposition of bits.

The MIS describes the largest code that is robust to such errors.

Butenko, S., et al., Finding maximum independent sets in graphs arising from coding theory. Proceedings of the 17th ACM Symposium on Applied Computing, 542–546, 2002.

Hardware accelerators

Hardware accelerators for MIS are subject to substantial research efforts due to the large industrial need

Ebadi, S., et al., Quantum optimization of maximum independent set using Rydberg atom arrays. Science, 376, 6598, 2022.

Mallick, A., et al., Using synchronized oscillators to compute the maximum independent set. Nature Communications, 11, 2020.

Further commercial applications

- Portfolio optimization
- Network immunization
- Ad-hoc networks
- Telecommunication loss networks
- Incremental store placement
- Task scheduling

Wurtz, J., et al., Industry applications of neutral-atom quantum computing solving independent set problems. arXiv:2205.08500v1, 2022.

Butenko, S. Maximum Independent Set and related problems with applications., 2003.

Taxonomy of optimization problems

Problem space

Loihi architecture and systems

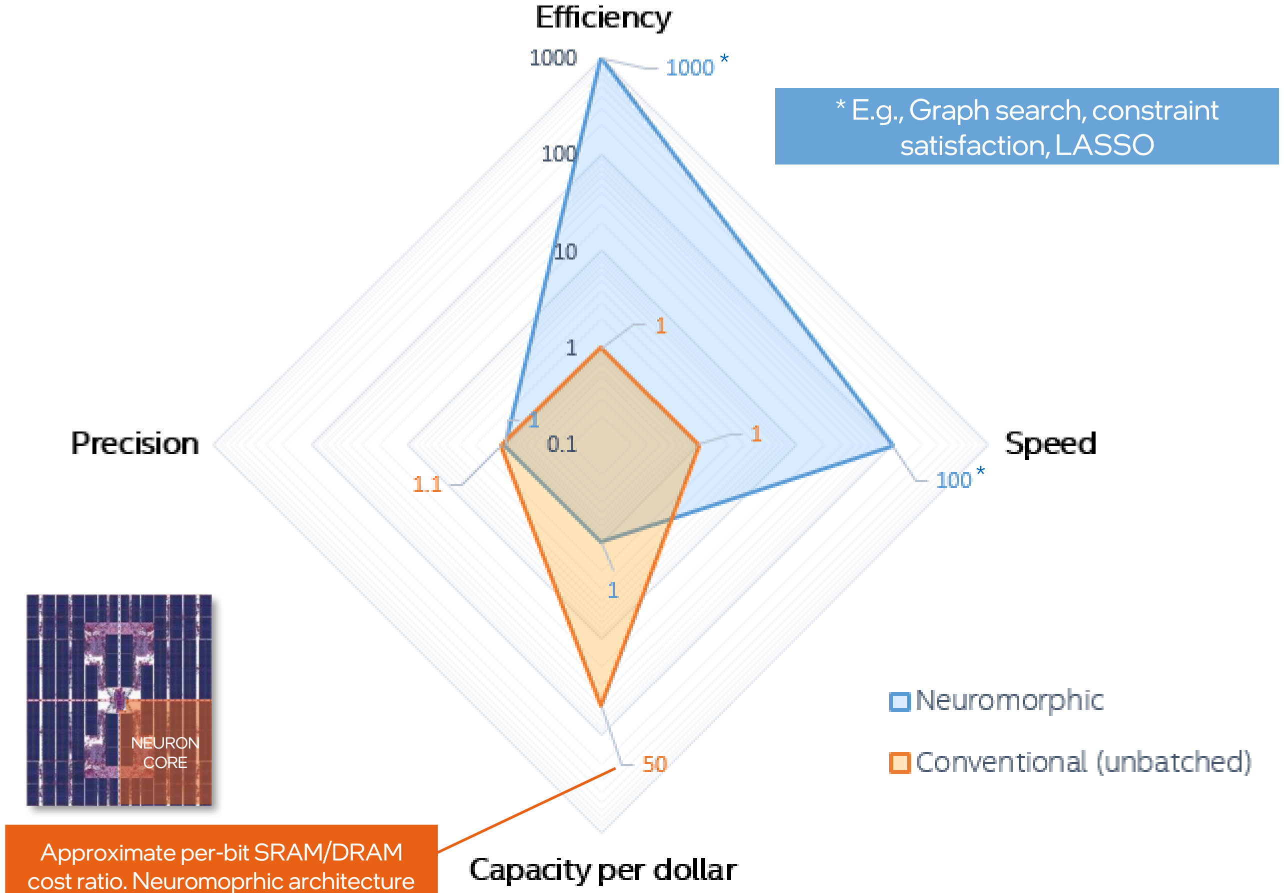
Brain-inspired computing

Orders of magnitude gains are in reach

- In **energy efficiency**
- In **speed of processing data** – especially signals arriving in real time
- In the **data efficiency** of learning and adaptation
- With **programmability** to span a **wide range of workloads** and scales

---

- Near term progress depends on **algorithmic and programming maturity**
- Long term, we will need to **reduce silicon cost with process technology innovations**

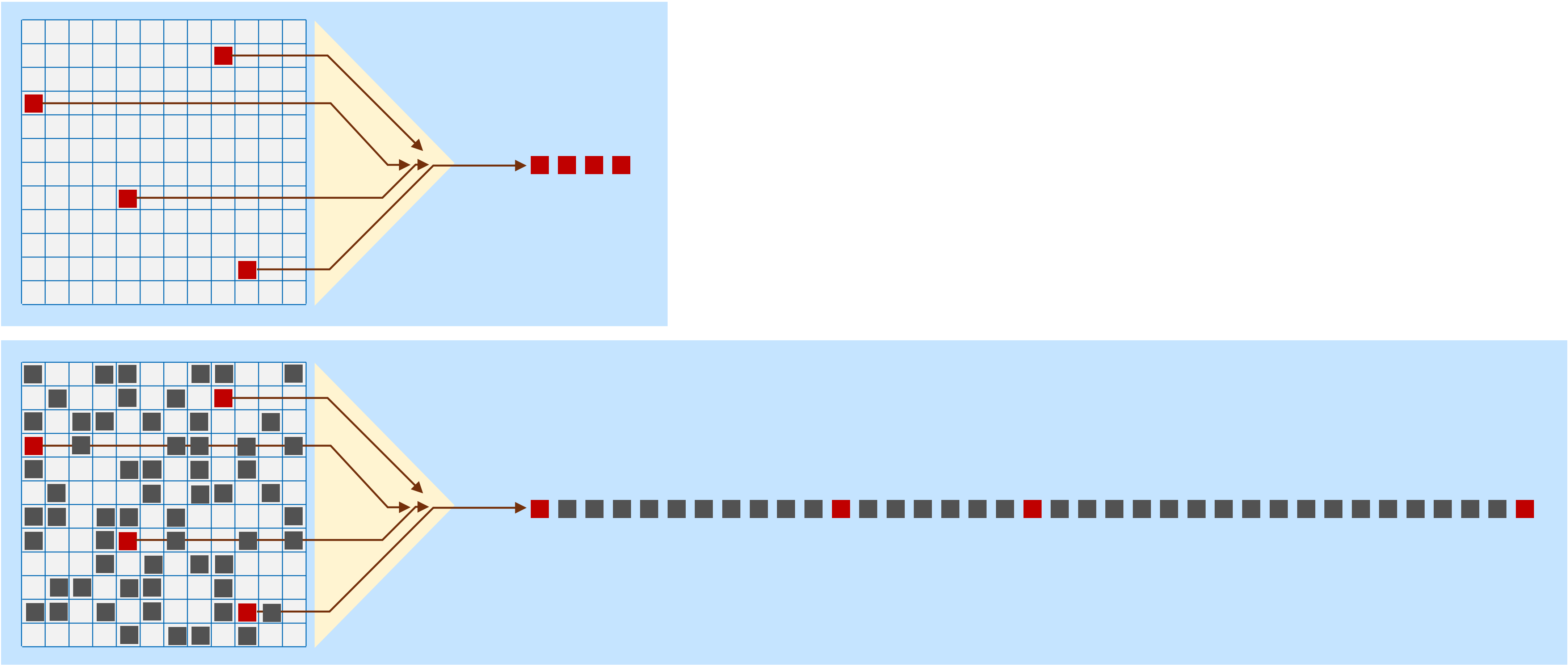


Approximate per-bit SRAM/DRAM cost ratio. Neuromoprhic architecture tightly integrates **memory** with **compute** so large workloads require large quantities of SRAM

* Davies et al, Proc. IEEE 2021. Results may vary.

Exploiting sparse, asynchronous communication

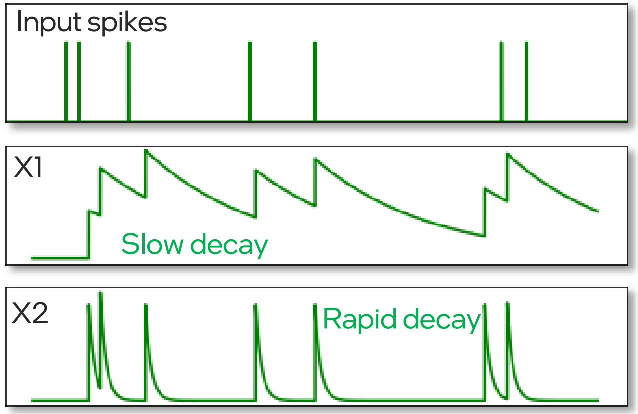
Fast and efficient, whether in brains or in computers



Loihi 2 enhances synaptic plasticity for advanced online learning

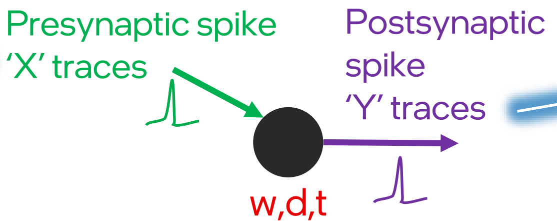
Pre-synaptic Traces (X)

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



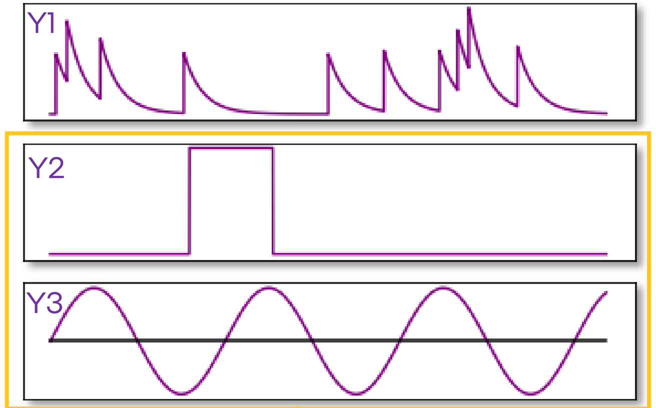
$$w' = w + \sum_{i=1}^{N_P} S_i \prod_{j=1}^{n_i} (V_{i,j} + C_{i,j})$$

<u>Synaptic Variables</u>
Wgt, Delay, Tag
(variable precision)

<u>Variable Dependencies</u>
X0, Y0, X1, Y1, X2, Y2,
Wgt, Delay, Tag, etc.

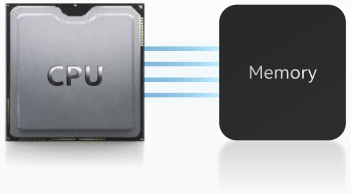
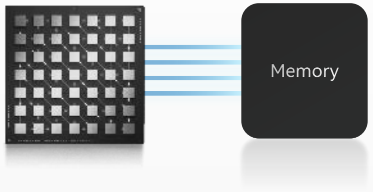
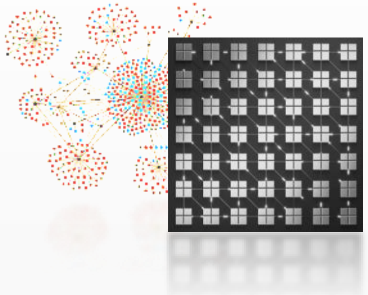
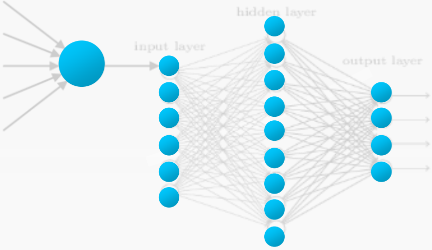
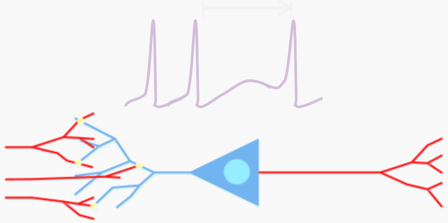
Post-Synaptic Traces (Y)

Loihi 1: LIF filters output spikes to generate traces



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

Neuromorphic hardware, a new class of computer architecture

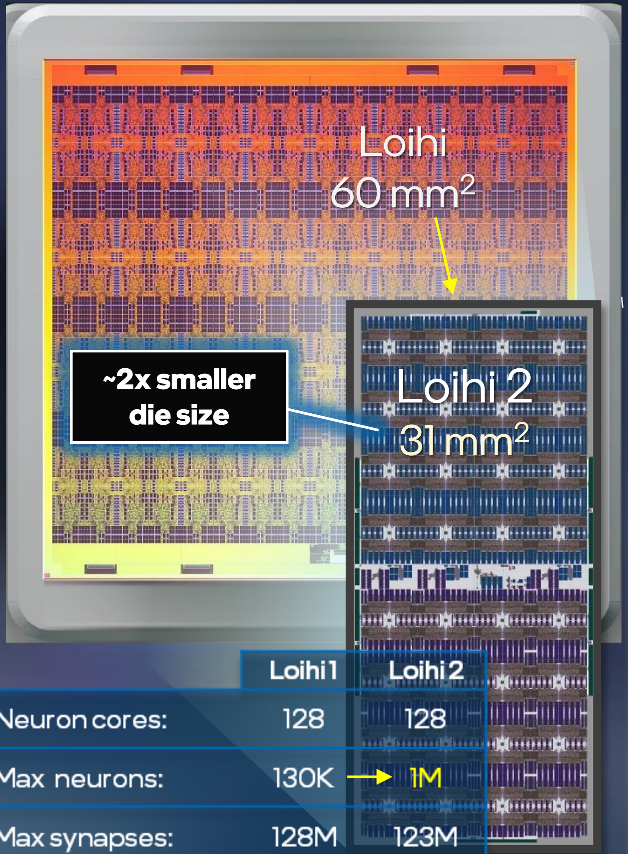
Standard Computing	Parallel Computing	Neuromorphic Computing
		
PROGRAMMING BY ENCODING ALGORITHMS	OFFLINE TRAINING USING LABELED DATASETS	LEARN ON THE FLY THROUGH NEURON FIRING RULES
SYNCHRONOUS CLOCKING	SYNCHRONOUS CLOCKING	ASYNCHRONOUS EVENT-BASED SPIKES
SEQUENTIAL THREADS OF CONTROL	PARALLEL DENSE COMPUTE	PARALLEL SPARSE COMPUTE
		

The Latest Loihi chip: Loihi 2

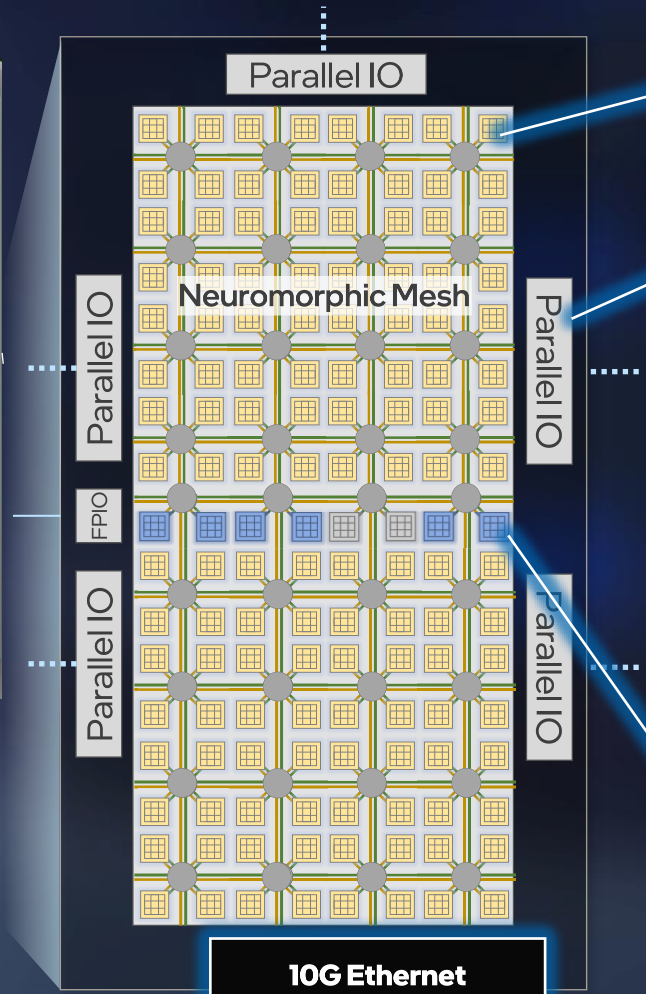
Generalized Spikes Spikes carry int8 magnitudes for greater workload precision

Programmable Neurons Neuron models described by microcode instructions

Programmable Neurons Neuron models described by microcode instructions



	Loihi 1	Loihi 2
Neuron cores:	128	128
Max neurons:	130K →	1M
Max synapses:	128M	123M
Max μP cores:	3 →	6



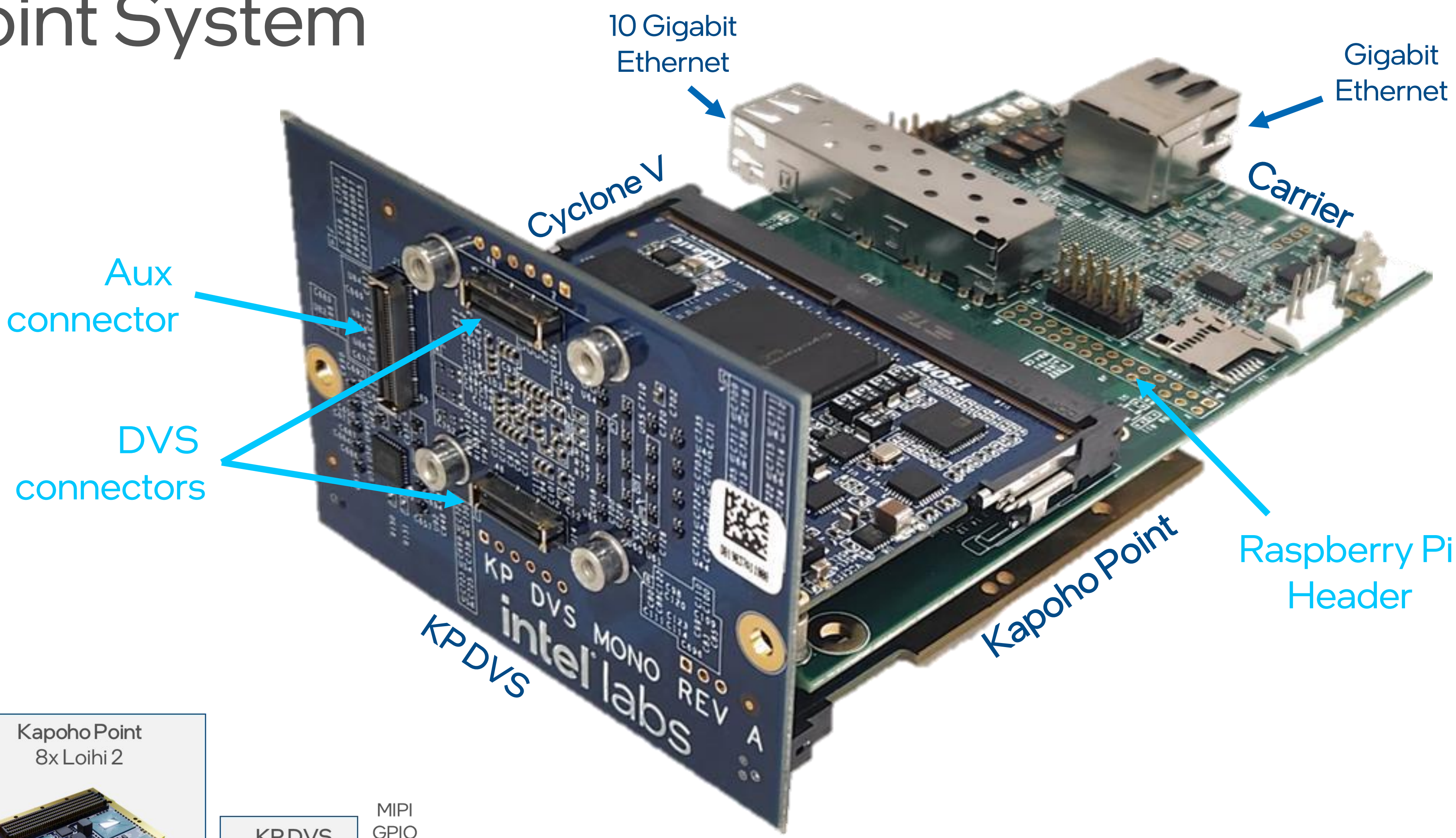
Neuromorphic cores (128)
Fully redesigned with up to 8192 neurons

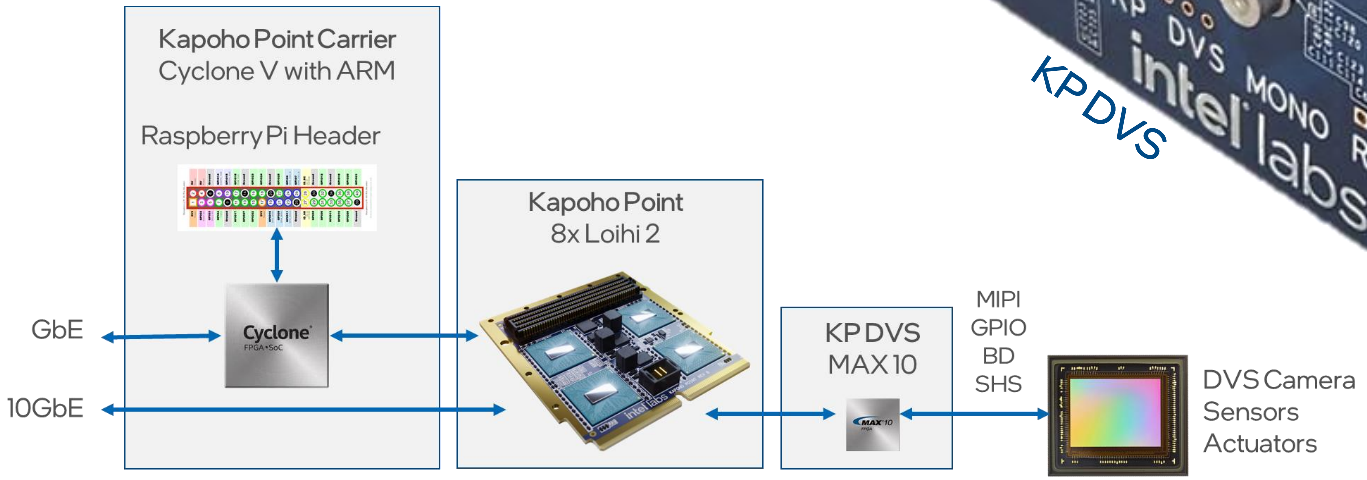
Parallel off-chip interfaces (6)
Async wave pipelined at 10 Gb/s with multicast compression

Microprocessor cores (6)
Asynchronous x86 and RISC-V

10G Ethernet
Accelerated host message + spike I/O

Kapoho Point System





Graph algorithms are well suited for neuromorphic computing

With large energy-delay product advantage

How does the search for solutions work?

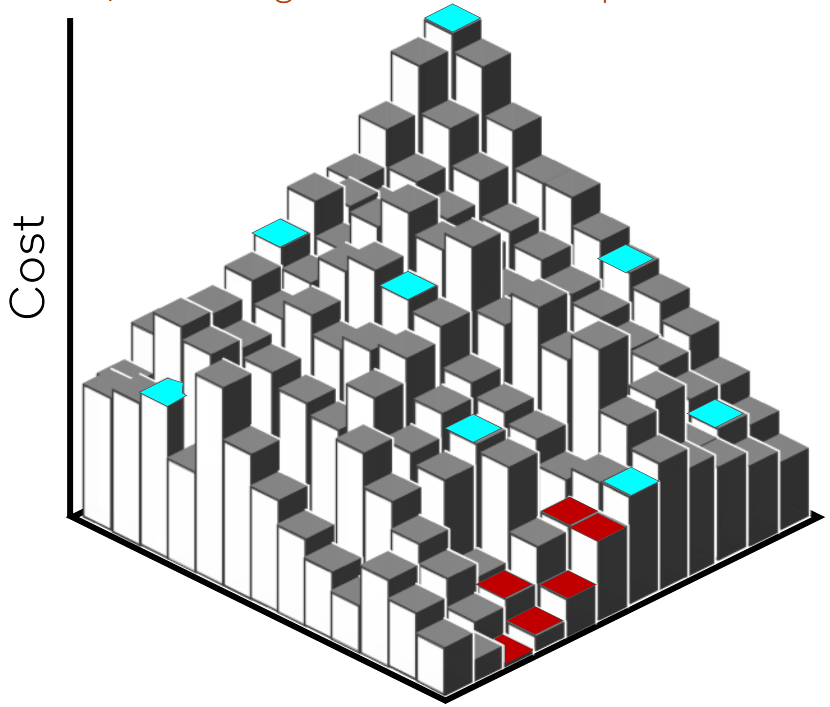
Efficient state space exploration

The Loihi 2 QUBO solver
Advancing by additional algorithmic features

Simulated Annealing

1) explore overall solution space

2) narrowing down the search space



SCIF neuron models

Parallelization

The Loihi 2 QUBO solver

Advancing by additional algorithmic features

Simulated Annealing

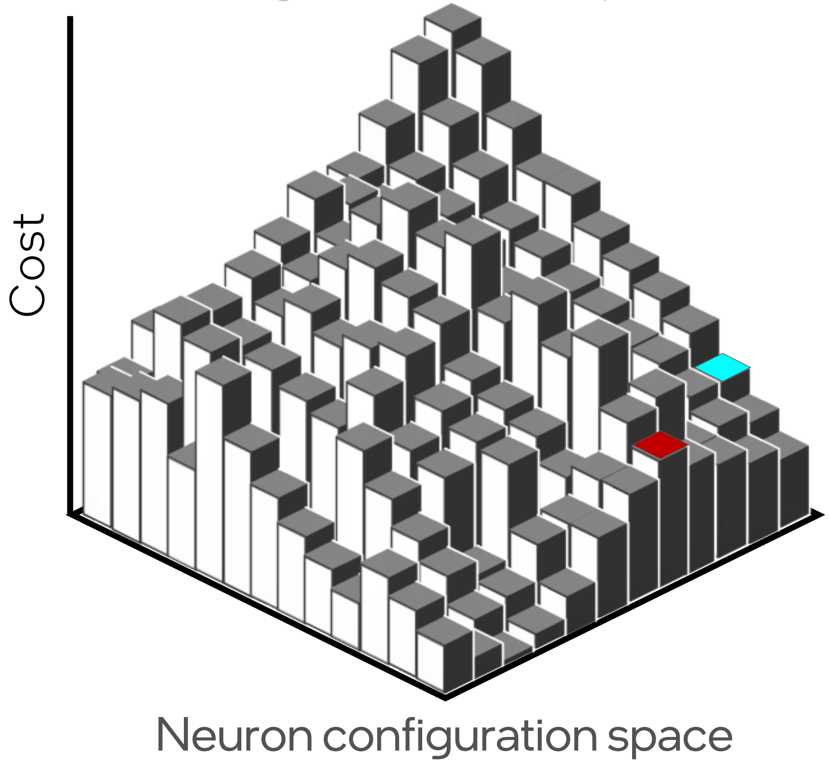
1) explore overall solution space

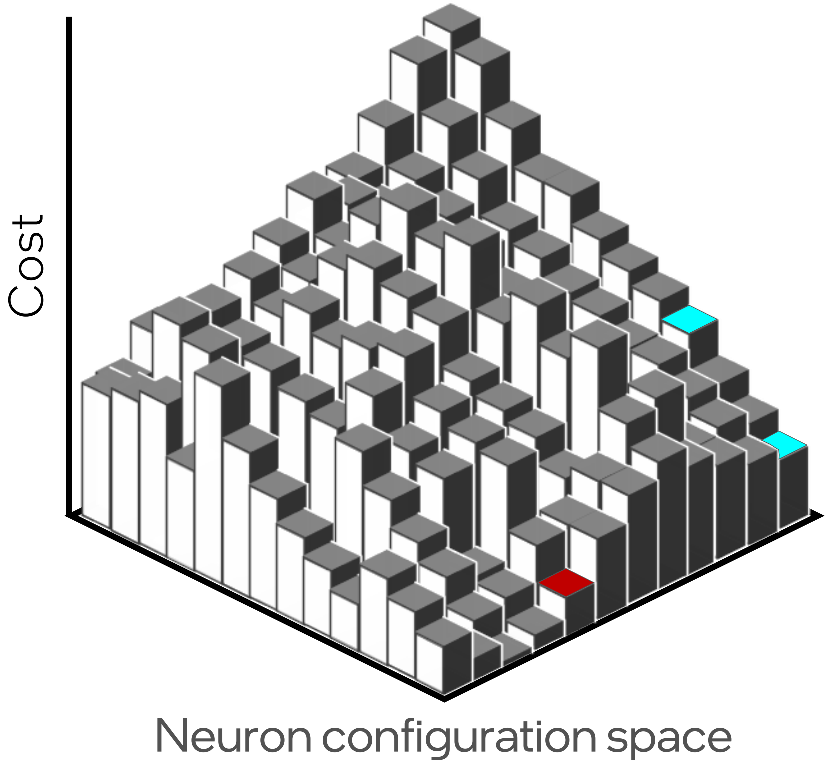
2) narrowing down the search space

SCIF neuron models

higher chance to escape local minima

Parallelization



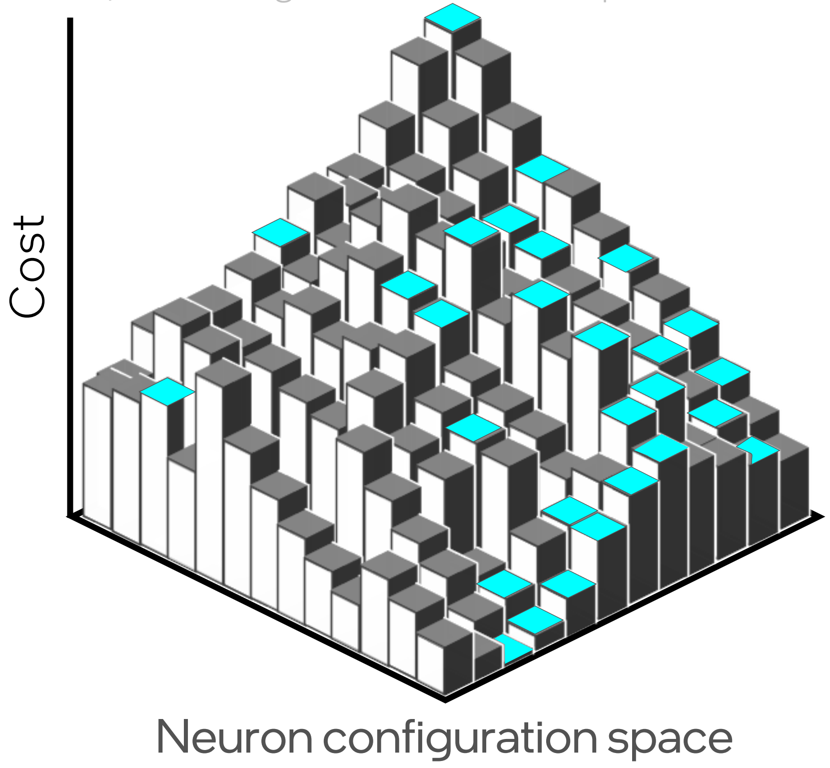


The Loihi 2 QUBO solver

Advancing by additional algorithmic features

Simulated Annealing

1) explore overall solution space
2) narrowing down the search space



SCIF neuron models

higher chance to escape local minima

Parallelization of …

… multiple workloads

… different hyperparameters

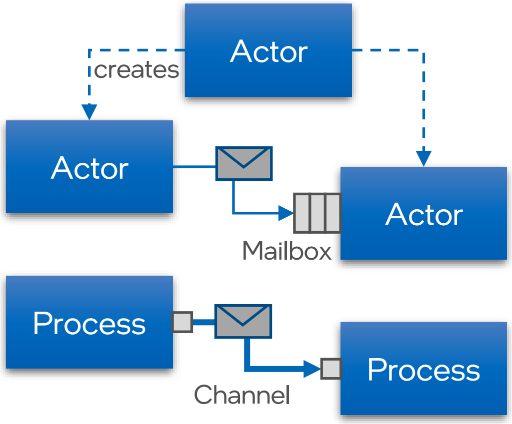
Lava development framework

Open source multi-paradigm software framework for neuromorphic computing

Comparison to other parallel programming models

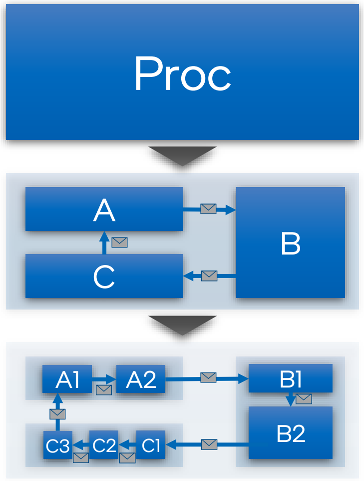
Lava inspired by…

Actor Model[1]
Communicating Sequential Processes[2]



- Asynchronous Processes/Actors
- Private state, no shared memory (safety)
- Direct point-to-point communication
- Message-based communication:
  - Actors:
    - Known destination actors
    - Non-blocking messaging via mailboxes
    - Message handlers
  - CSP:
    - Channels/unknown destinations
    - Blocking messaging semantics

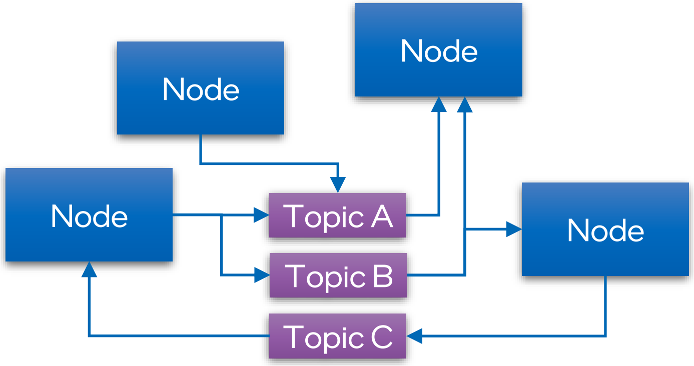
Caltech Asynchronous Synthesis Tools (CAST[3])



- Multi-abstraction:
  - Hierarchical design process
  - Iterative refinement of high-level to lower-level behavioral models
- Multi-platform: Execution on different backends

… but unlike…

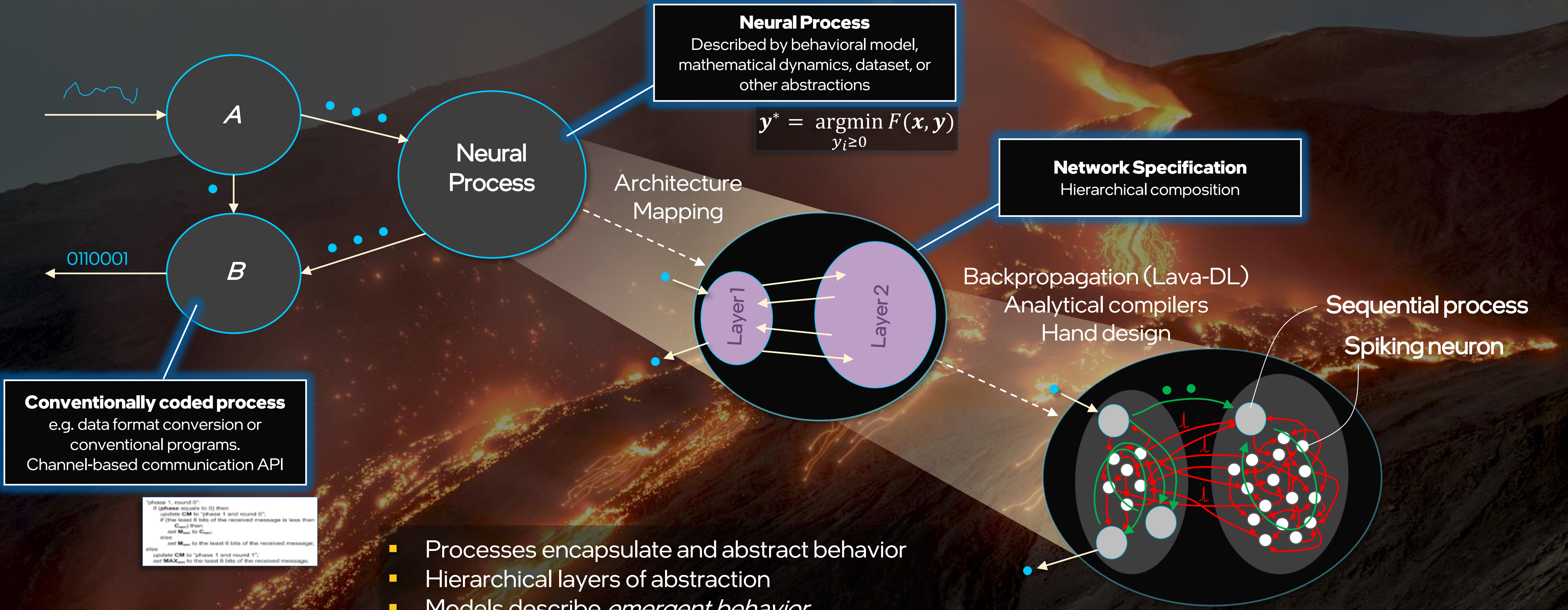
Publish/Subscribe[4]



- No point-to-point connections:
  - Publisher sends messages to named topic
  - Subscriber receives messages from certain topics
- Topic semantics implementable with Actor/CSP model
- Not suitable for neuromorphic systems:
  - More complex
  - Shared-memory overhead
  - Not matched to fine-granular parallelism
- (But Lava soon to offer interface to industry standard Data Distribution System (DDS) → ROS2)

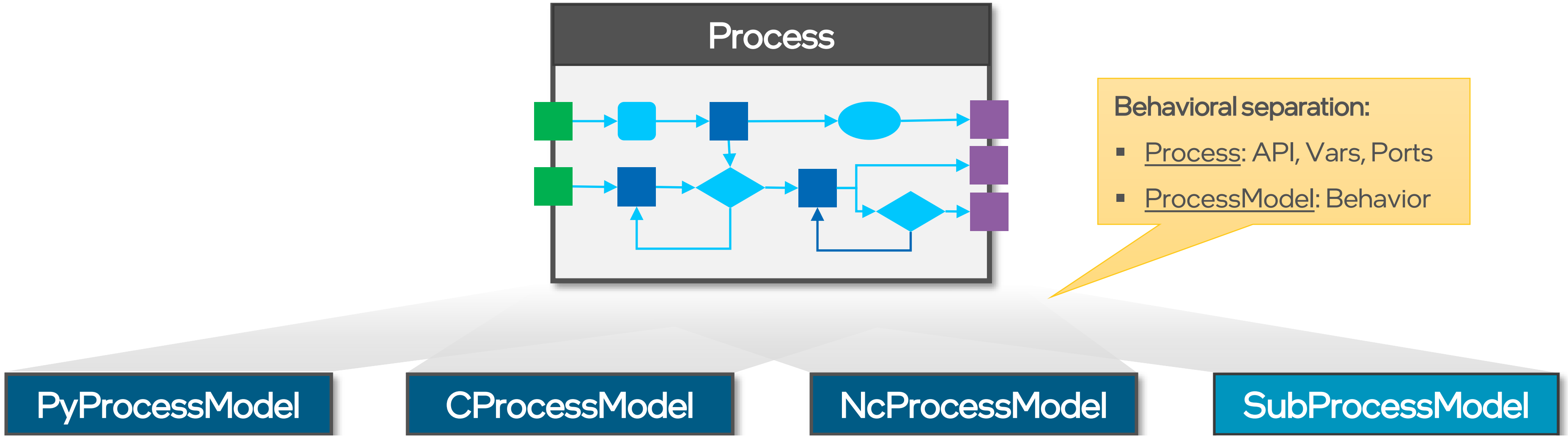
[1]Hewitt (1973) A Universal Modular Actor Formalism for Artificial Intelligence
[2]Hoare (1978) Communicating sequential processes
[3]NCL's proprietary async Hardware Description Language (Unpublished)
[4]Birman (1987) Exploiting virtual synchrony in distributed systems

Behavioral abstractions for productive programming



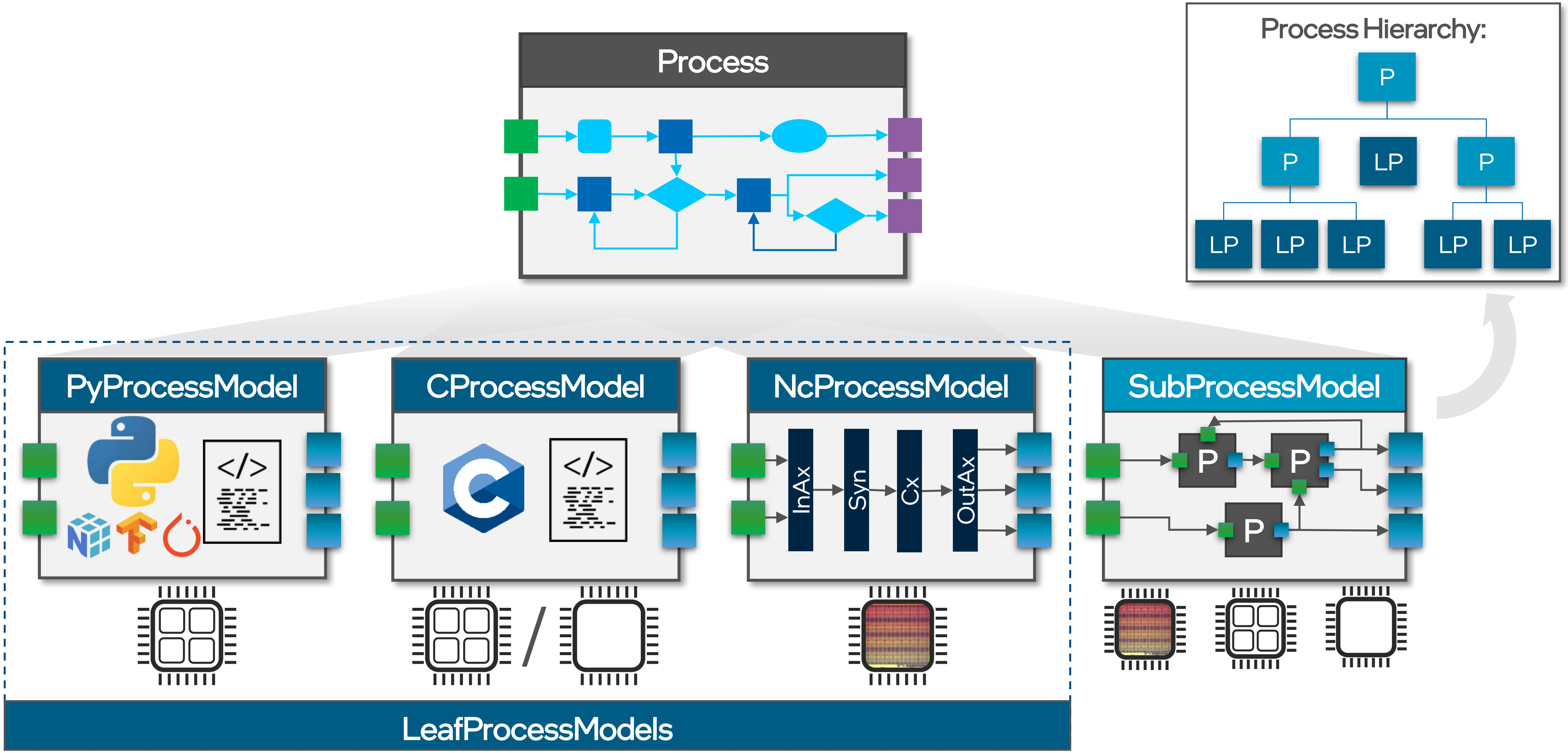
- Processes encapsulate and abstract behavior
- Hierarchical layers of abstraction
- Models describe *emergent behavior*

Behavioral models → ProcessModel



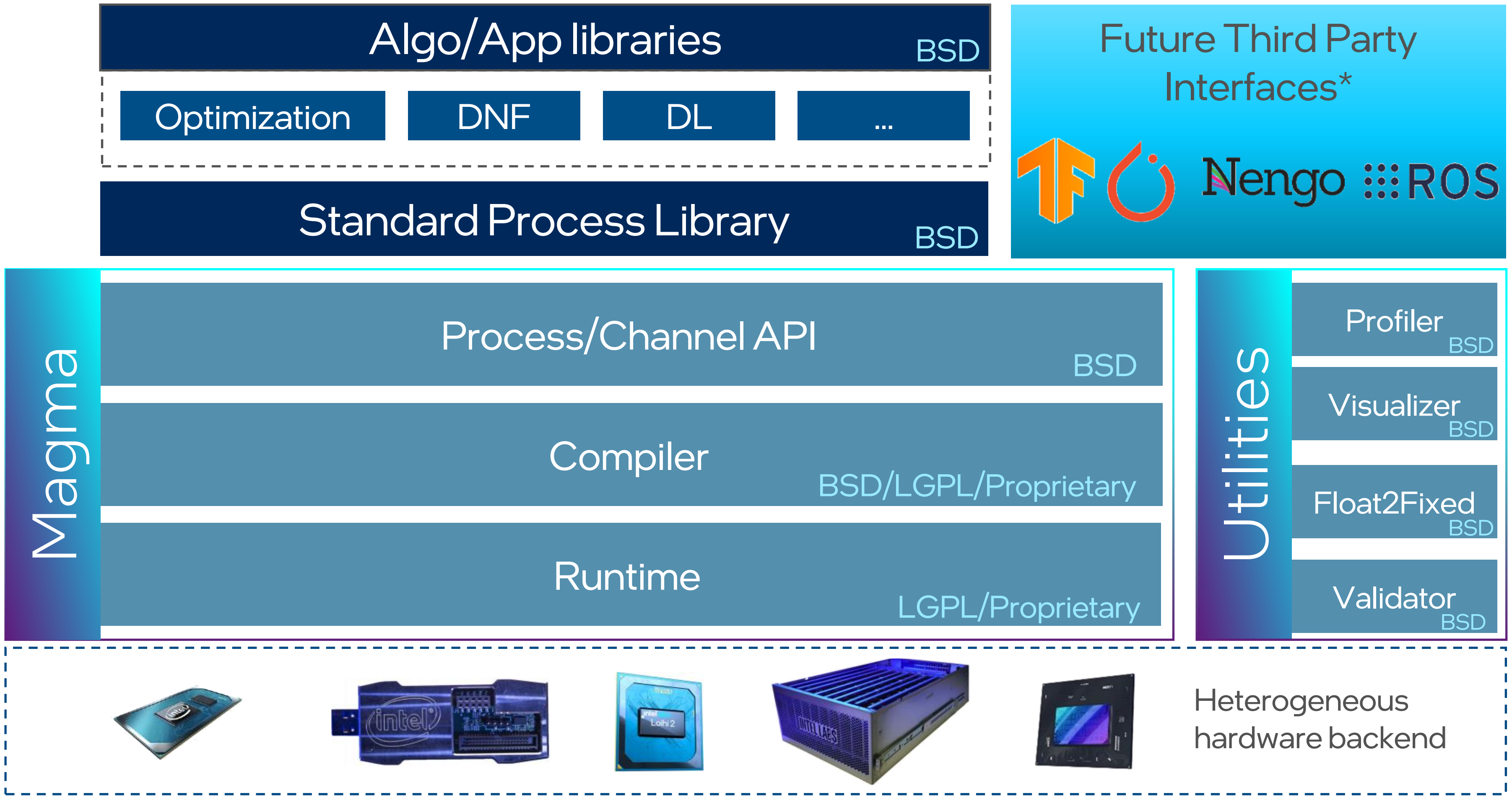
Multiple ProcModels per Proc
Refinement vs. inheritance → Stable application development

Behavioral models → ProcessModel



Two classes of ProcModels: LeafProcModels vs. hierarchical SubProcModels

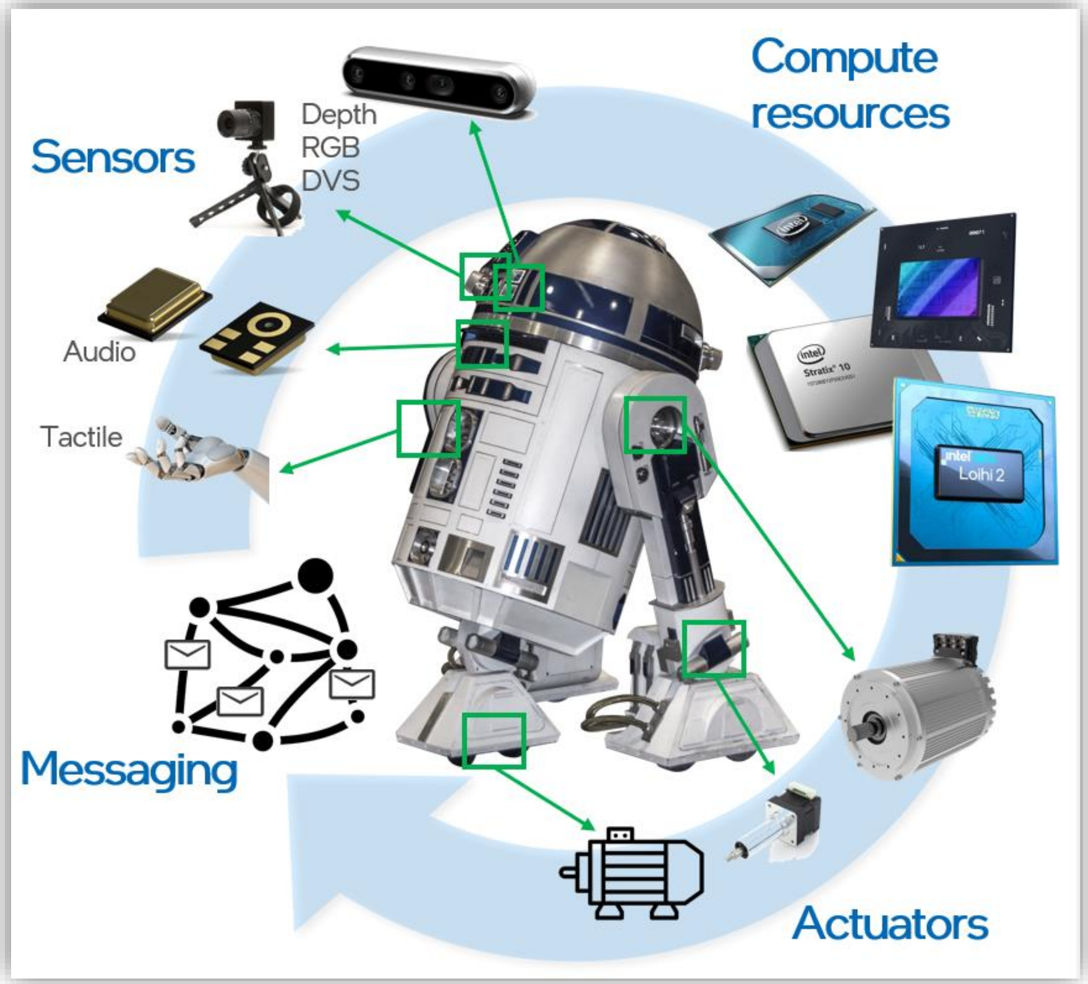
Lava open-source development framework



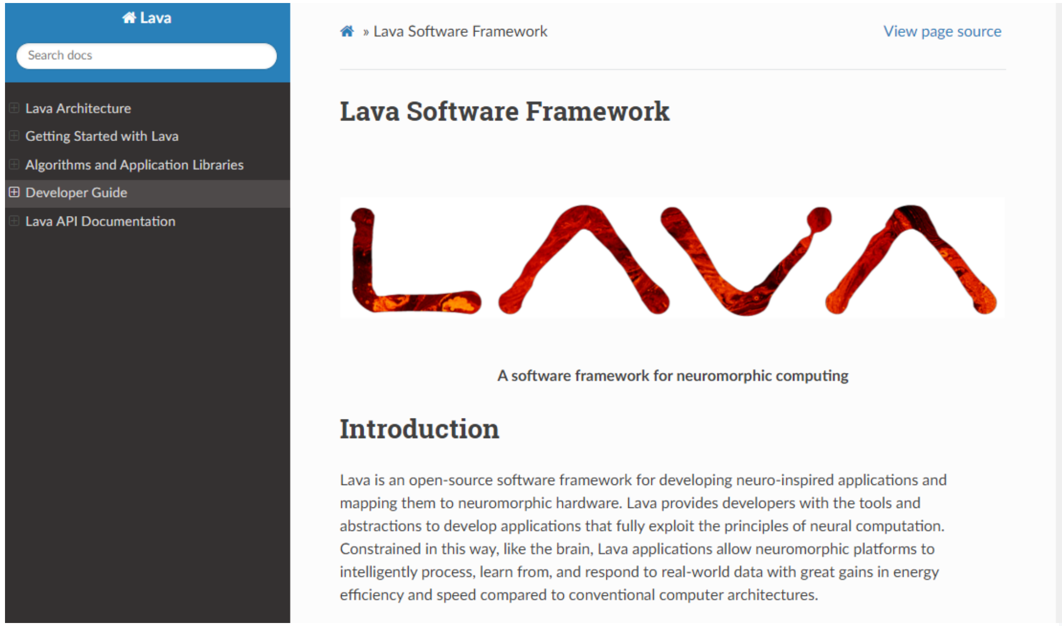
Objective

Facilitate real-world application development for neuromorphic systems

- Heterogeneous compute units:
  - CPUs, GPUs, other accelerators
  - Sensors, actuators (robotics)
- System characteristics:
  - Analog vs. digital
  - Run at different speeds
  - Synchronous vs. asynchronous
  - Event-based
  - Variable precision, stochastic, noisy
  - Highly parallel
  - Distributed
- Lava's programming model:
  - Processes with channel-based communication
  - Multi-paradigm
  - Multi-platform
  - Multi-abstraction

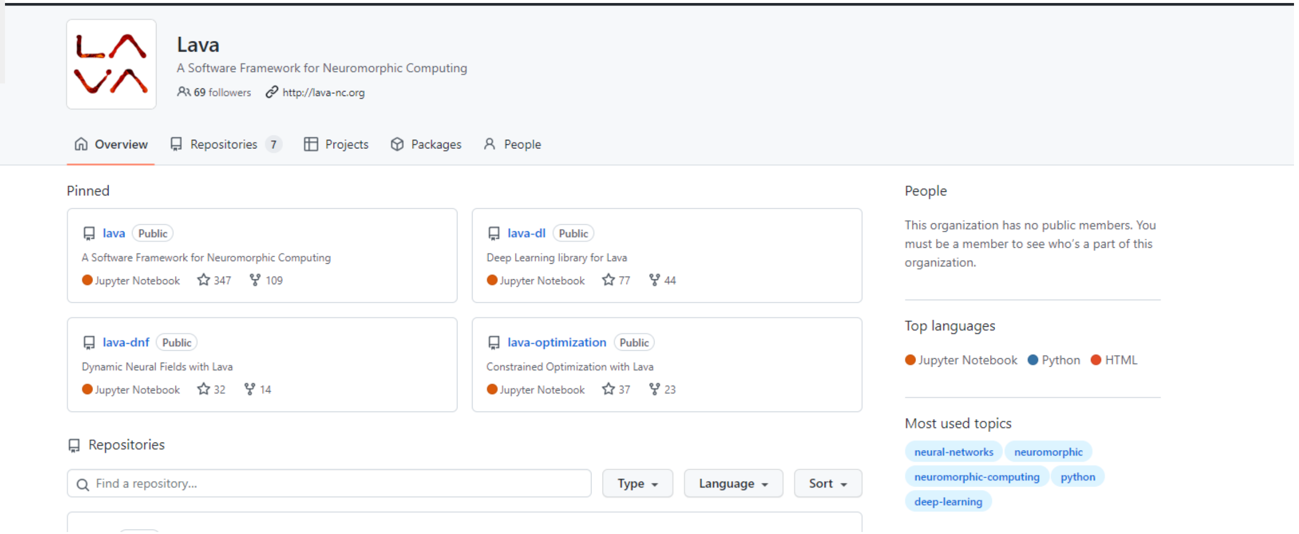


Learn more about Lava and follow (star/fork) us on Github



https://lava-nc.org/

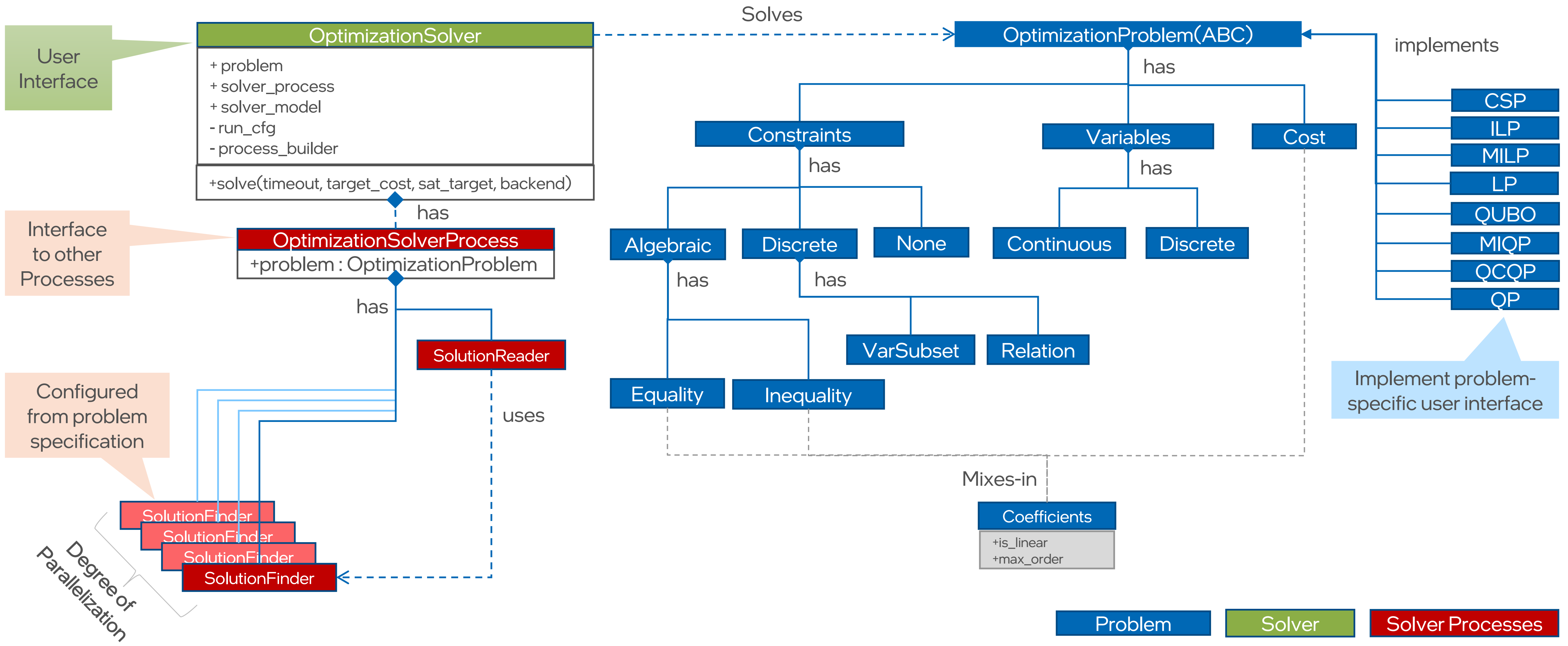
https://github.com/lava-nc



Lava-optimization architecture

Towards neuromorphic-agnostic user experience

Lava-optimization library high-level architecture



Frontend to backend pipeline

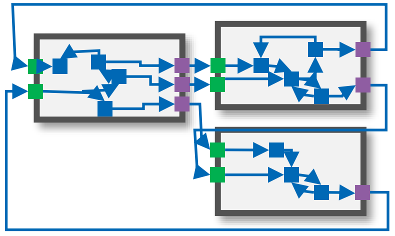
K-SAT

$$(-l_1 \vee l_2 \vee l_3) \wedge$$
$$(l_1 \vee l_4 \vee -l_3) \wedge$$
$$(l_1 \vee l_5 \vee l_6)$$

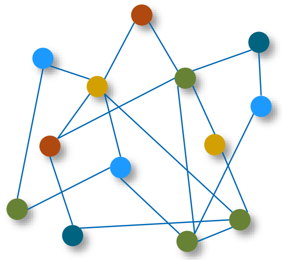
CSPSolver

```
ksat=CSP(variables,
         constraints)

solver=
OptimizationSolver(
ksat)

solver.solve(ksat)
```

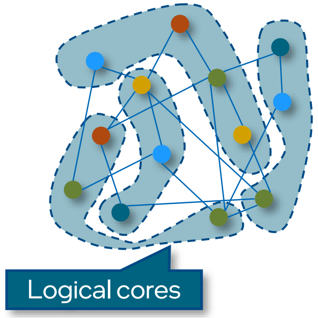
Lava Processes Network



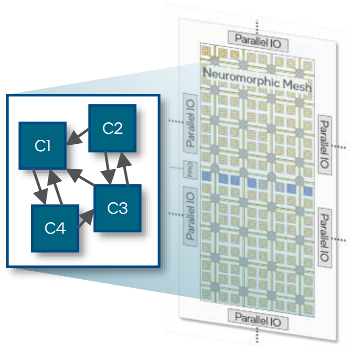
Preprocessing



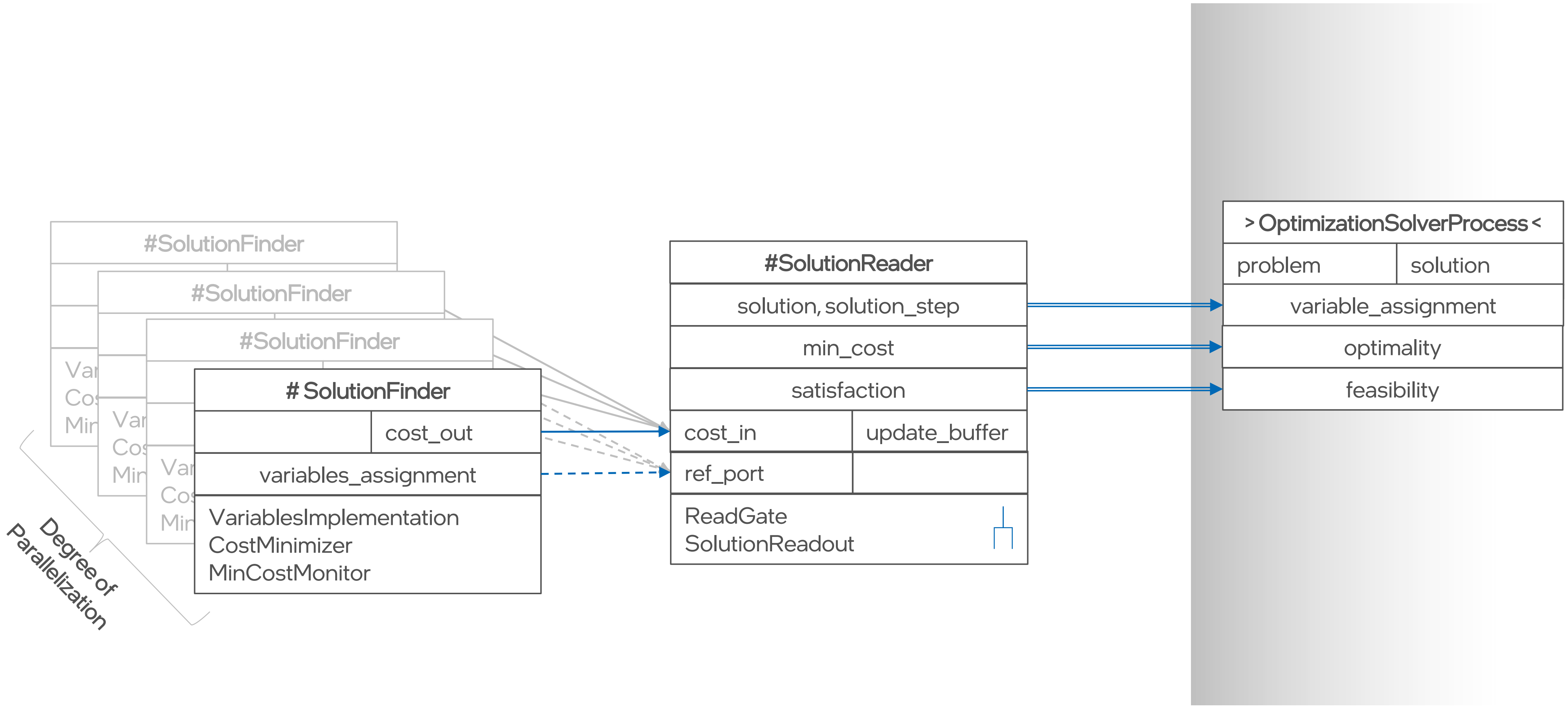
Graph partitioning



Core mapping



SolverProcess Architecture



Increased performance efficiency

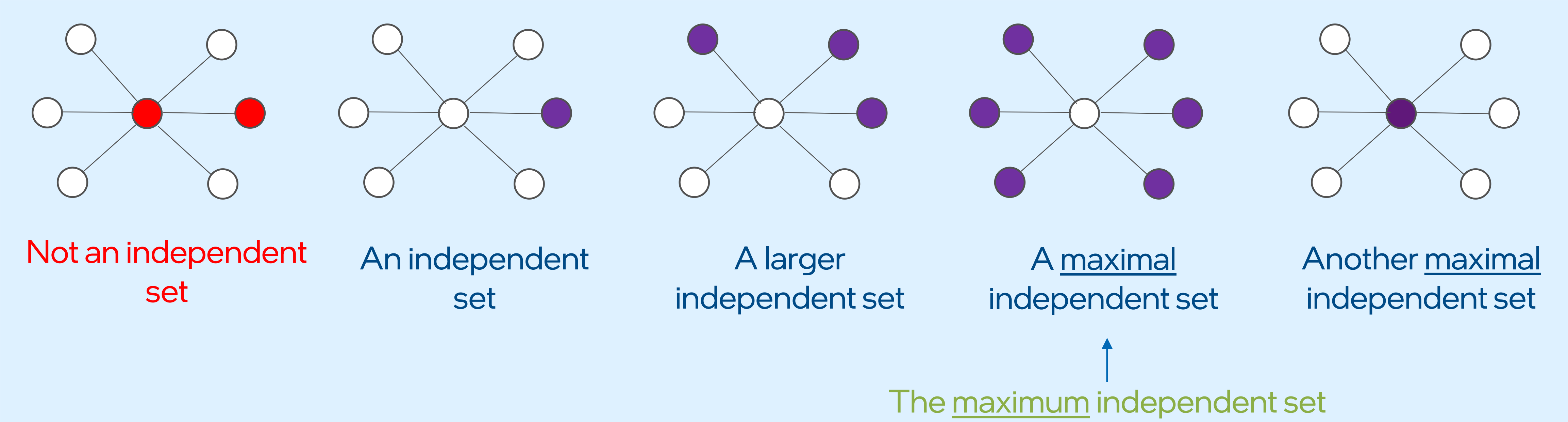
Solving hard optimization problems

Zoom-in maximum-independent set as QUBO

Energy and time to solution

Maximum Independent Set

Problem: Given an undirected graph $G = (V, E)$, find largest subset of disconnected vertices $U \subseteq V$.



Finding the maximum independent set is NP-Hard!

Example applications:

- Resource allocation in wireless communication networks
- Chip reliability via redundant circuitry

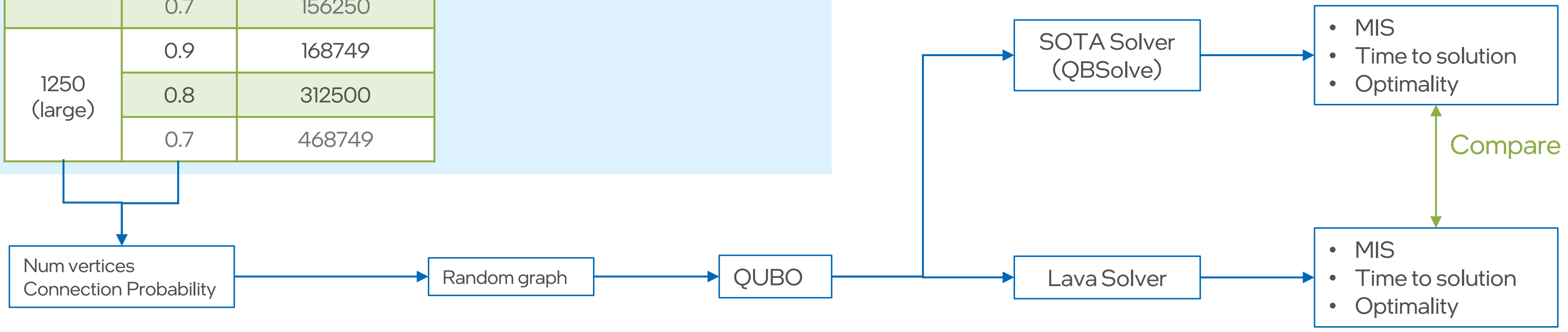
Benchmarking MIS as QUBO on Loihi 2 Vs QBSolve

Dataset

Nodes V	Connection Probability p	Complexity $(V^2 - V) \times (1 - p)$
250 (small)	0.9	6250
	0.8	12500
	0.7	18749
750 (medium)	0.9	56250
	0.8	112500
	0.7	156250
1250 (large)	0.9	168749
	0.8	312500
	0.7	468749

Search space grows with number of unconnected pairs

✖ 20 Problems per complexity ✖ 20 trials per problem



Intel Neuromorphic Research Community

INRC

Available to engaged INRC members

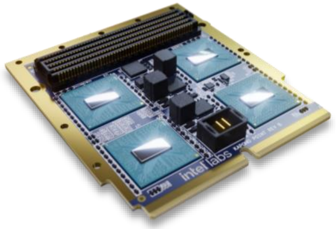
Available to engaged INRC members

Neuromorphic Research Cloud Access

Lava framework

Future Loihi 2
Large-scale systems

Kapoho Point



Our goals for the next three years

- Demonstrate commercially relevant applications
- Enable productive and accessible programming models and software frameworks for neuromorphic computing
- Help converge the field with benchmarking and SW standardization
- Exponentially expand neuromorphic research & developer community
- Demonstrate impactful state-of-the-art technology breakthroughs