Neuromorphic Computing at Lawrence Livermore Lab

Salishan

Brian Van Essen
Center for Applied Scientific Computing

April 26, 2017

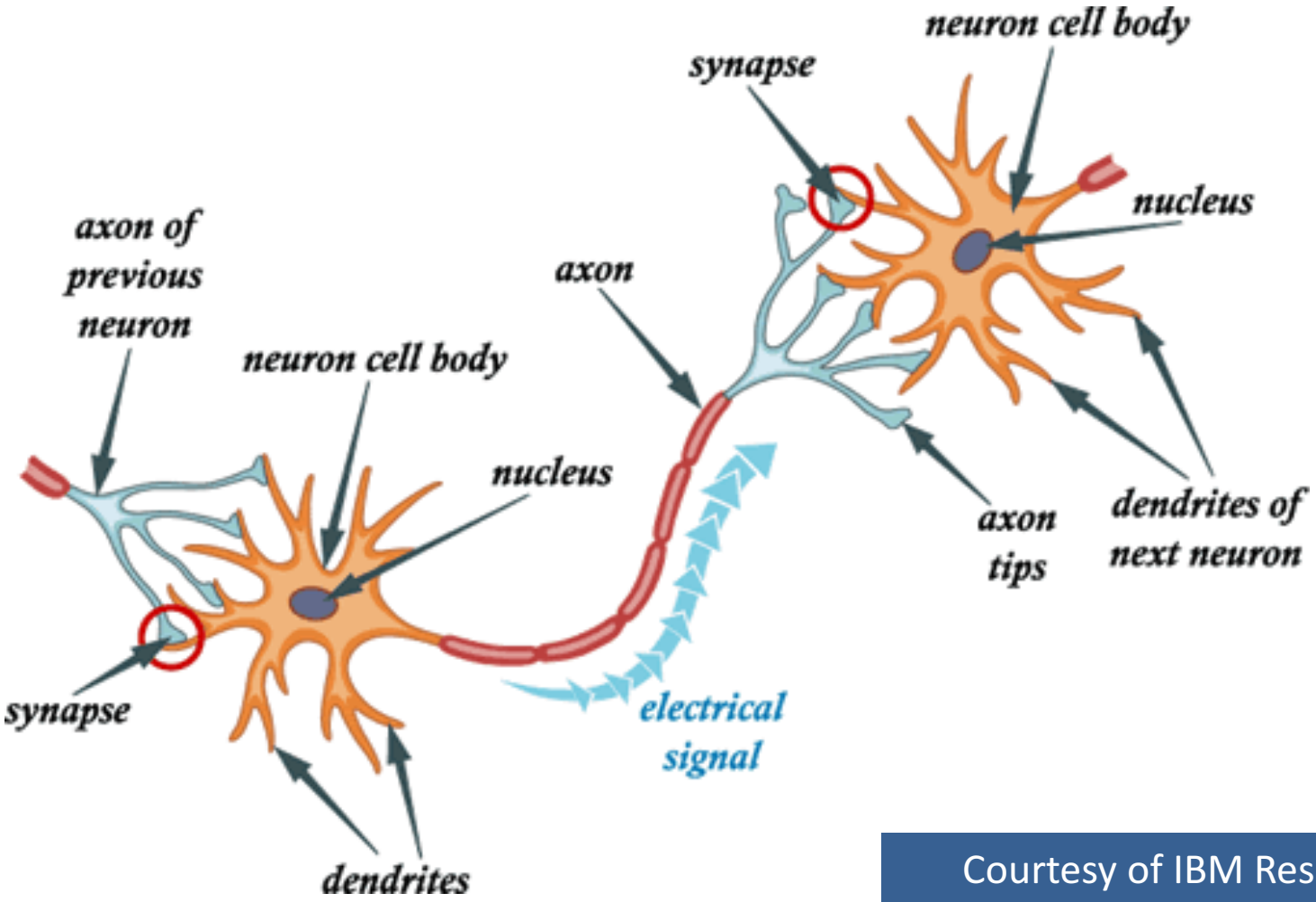
LLNL-PRES-726576
This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

Lawrence Livermore National Laboratory

NNSA ASC has a program for Beyond Moore's Law computing architectures

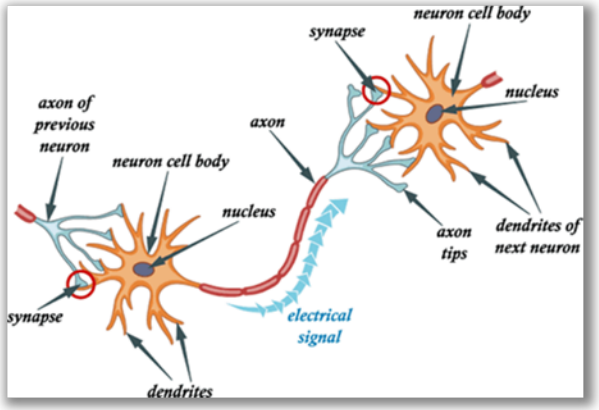
- Explore applicability of new computing architectures to ASC application domain
  - Low power
  - Apply machine learning at scale to ASC simulations
  - Support other national security needs and programs
- Quantum Computing
  - D-Wave system at LANL
  - LLNL is collaborating on algorithms development
- Neuromorphic Computing
  - LLNL is hosting the IBM TrueNorth testbed system
  - LLNL is leading application exploration and development of base capability
  - LLNL is providing collaboration nexus across DOE labs

Biological inspiration

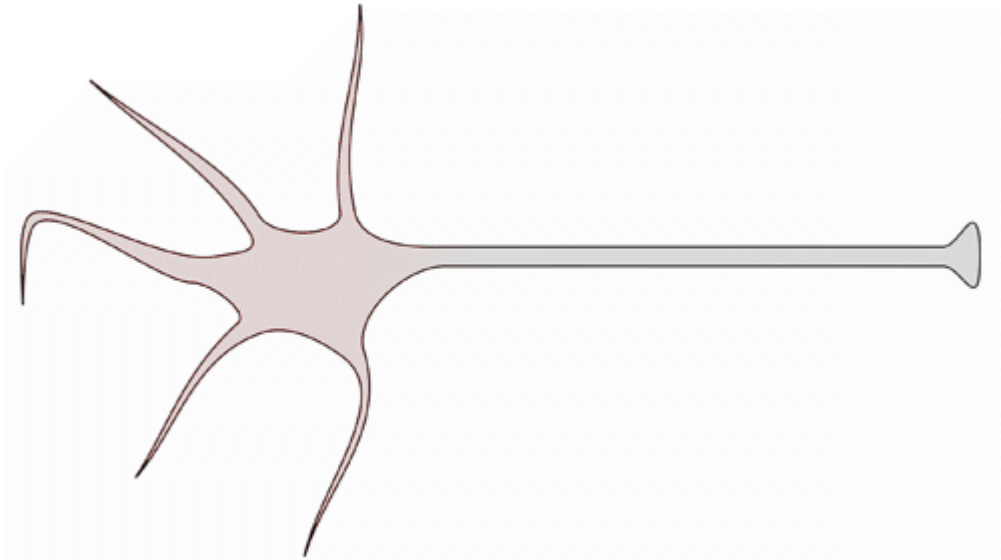


Courtesy of IBM Research – TrueNorth Ambassador Slide Deck

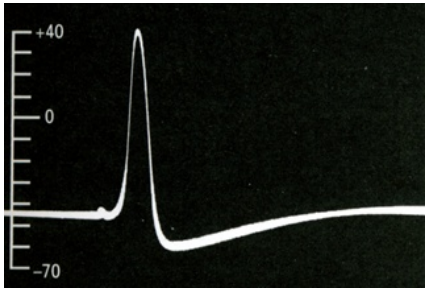
What Neurons Do: Simplistic Description



- Neuron integrates inputs received on dendrites
- Launches an electrical pulse—"action potential" or "spike"—down axon when a threshold is reached

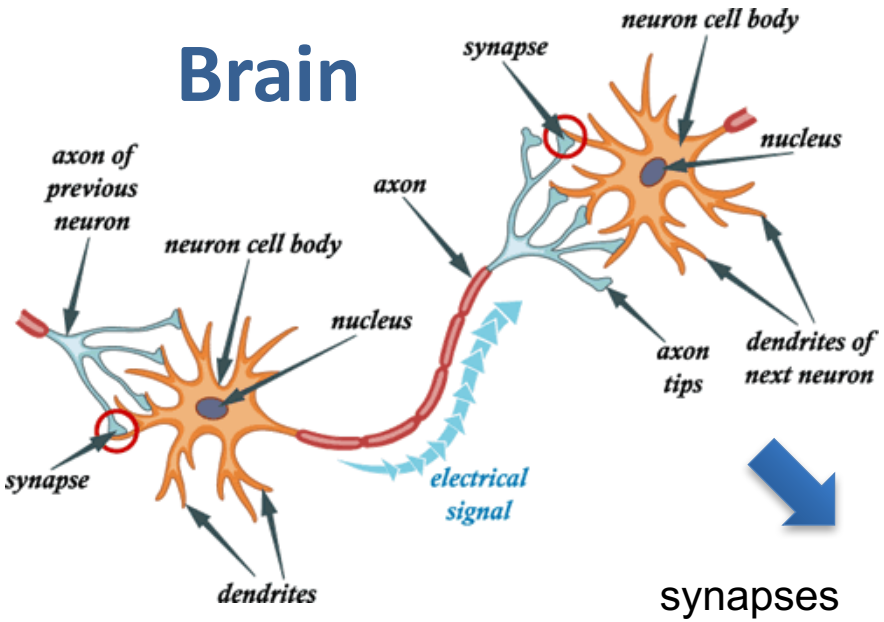


Essentially a binary communications pulse

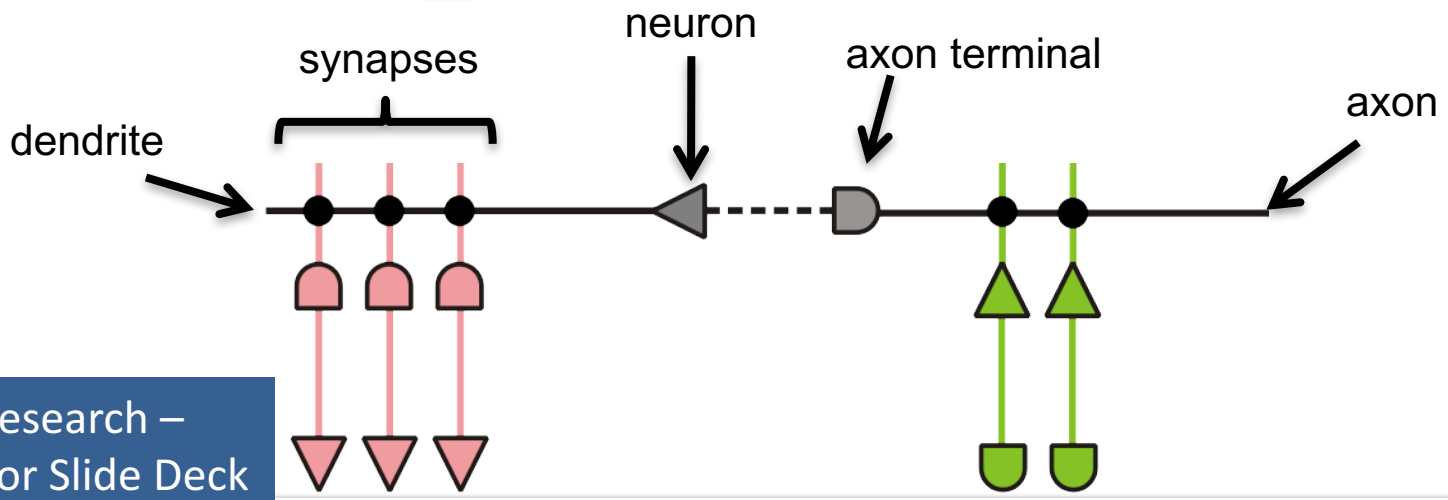


Courtesy of IBM Research – TrueNorth Ambassador Slide Deck

Brain-inspired computing



- Amazing recognition tasks
- Very low power
- TrueNorth Neuromorphic architecture



Courtesy of IBM Research – TrueNorth Ambassador Slide Deck

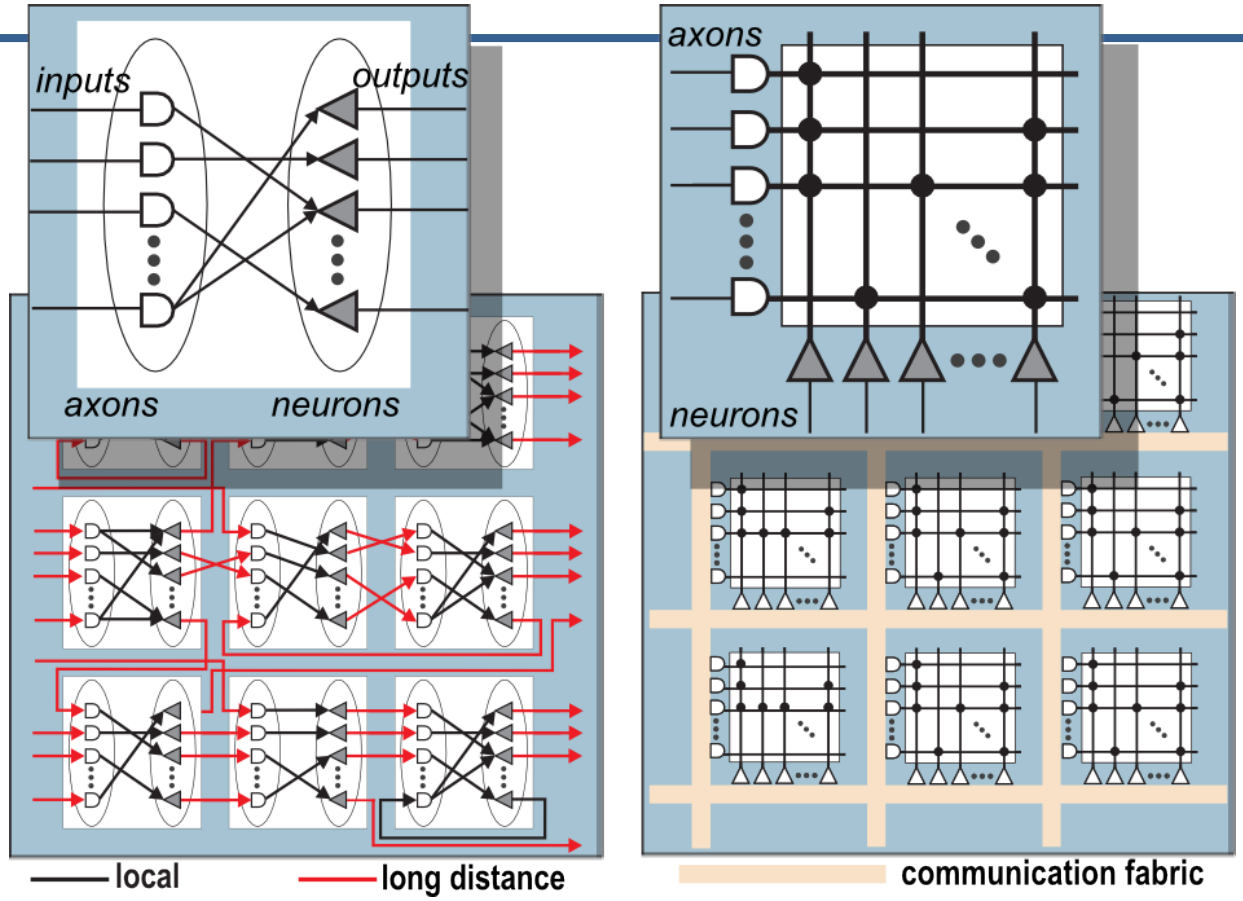
Neuromorphic Computing is a Brain Inspired Computing method

Compute more efficiently per unit area

- Human Brain
  - 20 Watts
  - Biological time scale – Hertz not gigaHertz
  - High connectivity – approximately 10K connections per neuron
  - Spiking compute units (neurons)
- IBM's TrueNorth Neurosynaptic architecture
  - Distributed memory, coupled to neurons
  - Asynchronous circuits
  - Spiking signaling (low resolution data encoding)
  - "Real-time" – kiloHertz clock tick

This is a potentially a fundamental shift in how we solve certain classes of problems.

TrueNorth is a Neurosynaptic Architecture



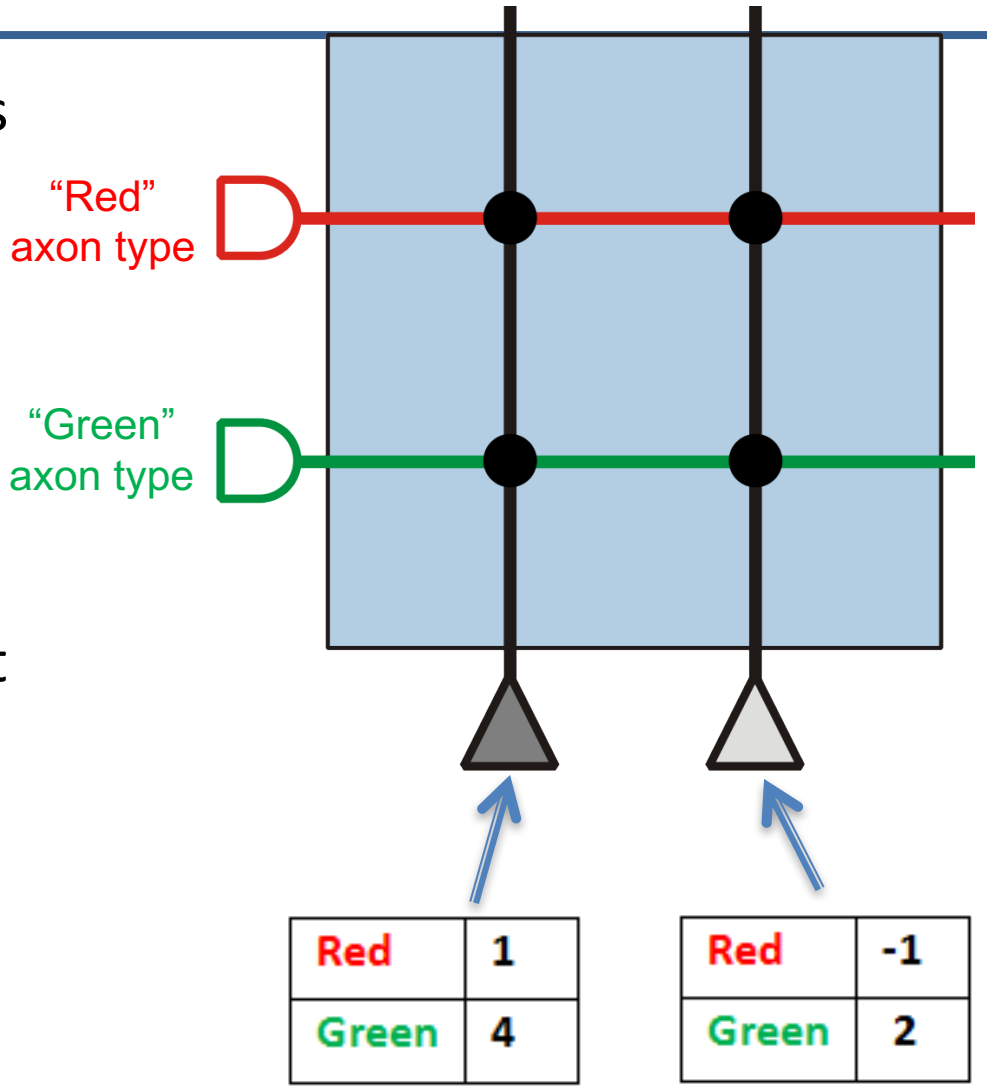
- Programmable by configuring
  - synaptic matrices
  - neuron parameters
  - Network
- Each core has:
  - 256 Spiking Neurons
  - 256 Inputs (Axons)
  - 256x256 Low-Precision Synapses

- Specialized Hardware to emulate spiking neural net. (no typical Instruction-Set Architecture).

Courtesy of IBM Research – TrueNorth Ambassador Slide Deck

Working with architectural constraints

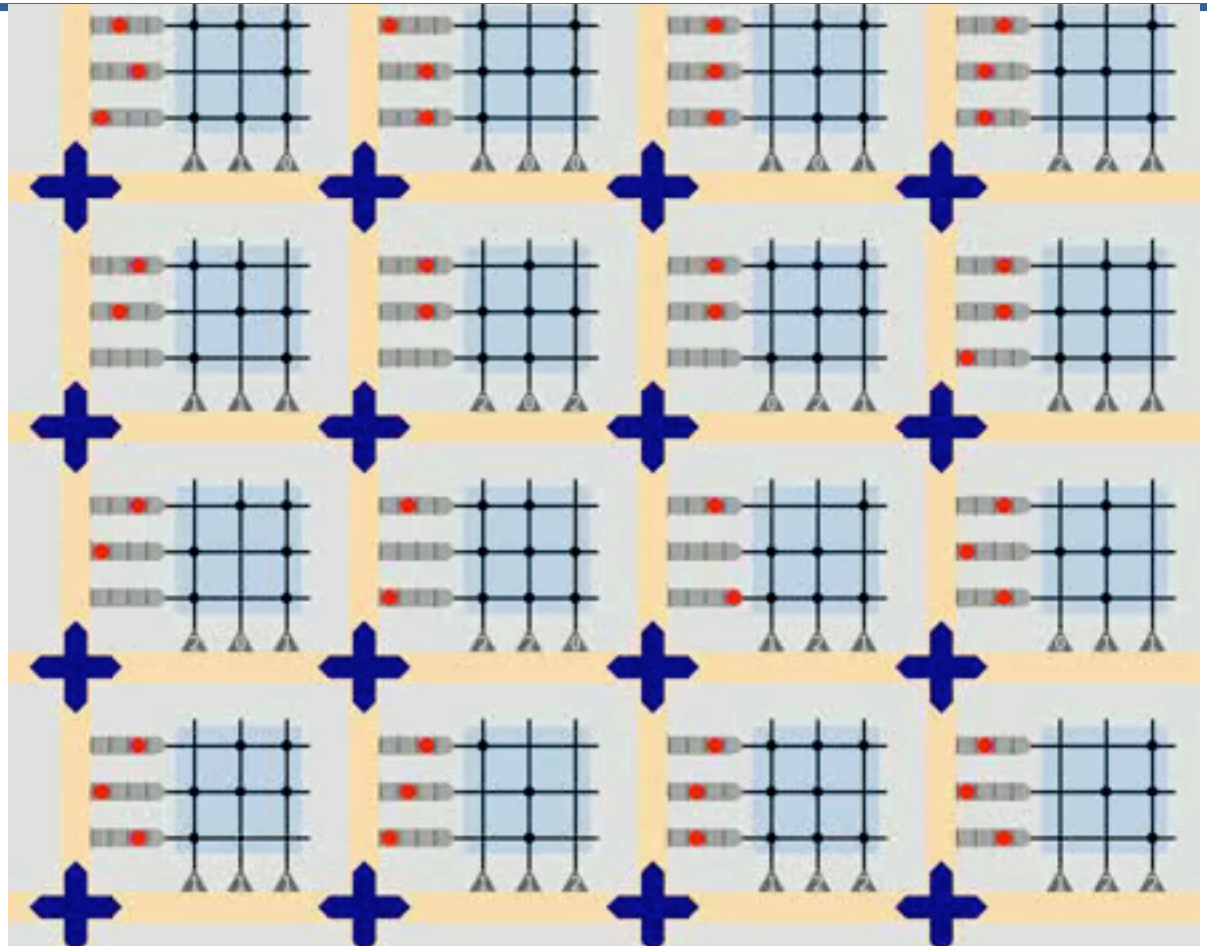
- Limited connectivity patterns between Axons and Neurons
  - 4 unique axon types
  - Each neuron has unique response to each axon type



- Cannot have arbitrary weight patterns for input signals

Courtesy of IBM Research – TrueNorth Ambassador Slide Deck

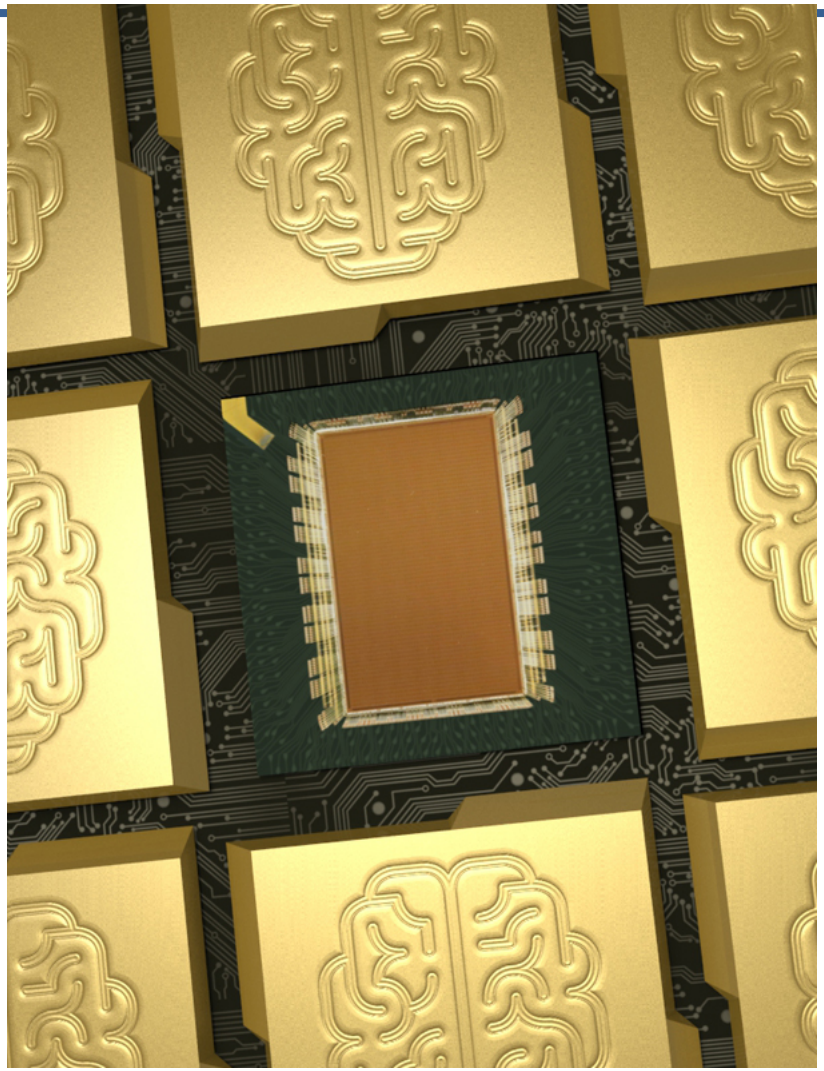
Network of Neurosynaptic Cores



Courtesy of IBM Research – TrueNorth Ambassador Slide Deck

TrueNorth Chip

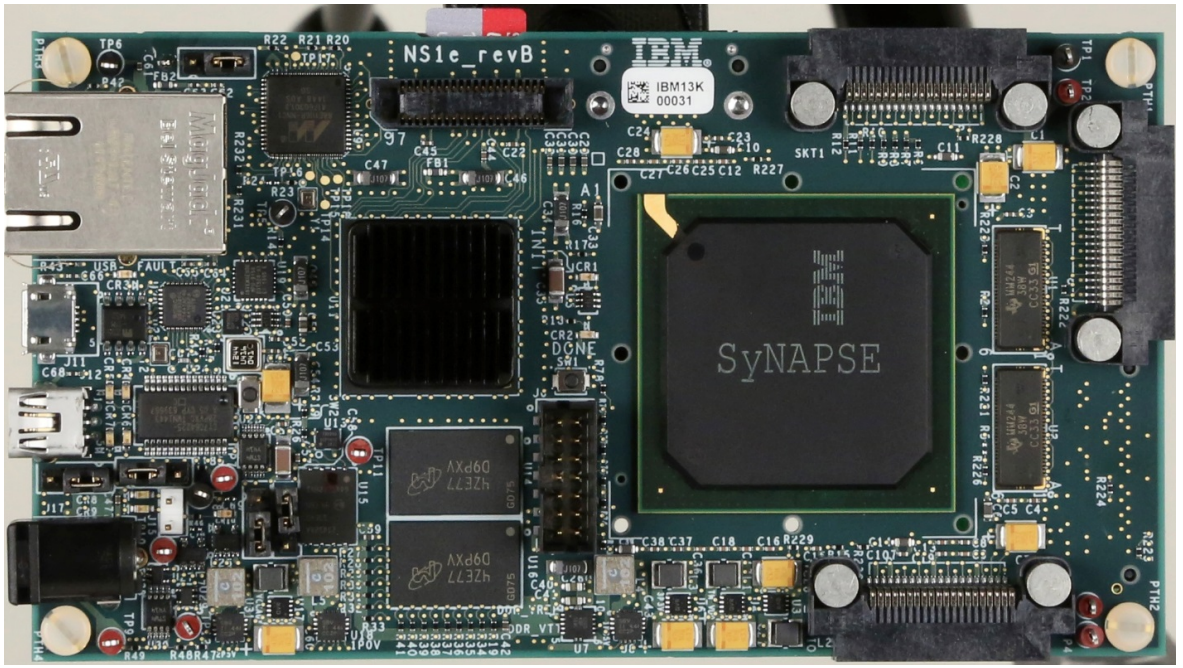
- 4,096 neurosynaptic cores
- 1 million neurons
- 256 million synapses
- Extremely low power
- Readily tiled with other chips



Courtesy of IBM Research –
TrueNorth Ambassador Slide Deck

NS1e – Single-chip system

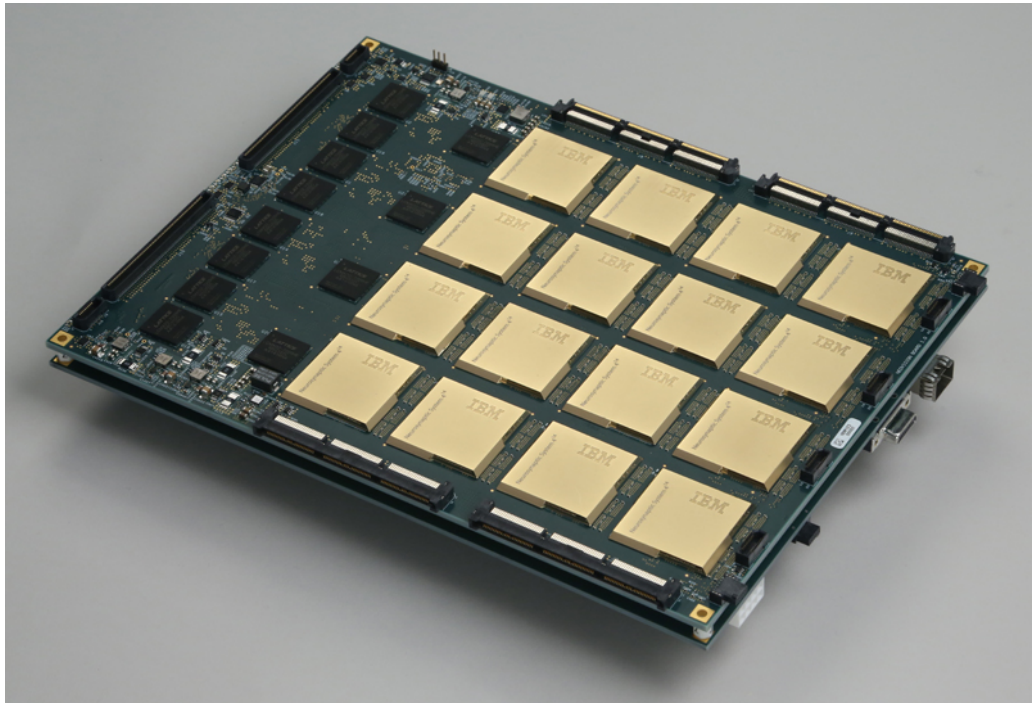
- Designed for mobile neurosynaptic applications.
- Low-Power and many functionalities
- Used for IBM NeuroSynaptic Boot Camp training class



Courtesy of IBM Research – TrueNorth Ambassador Slide Deck

NS16e – Multi-chip System

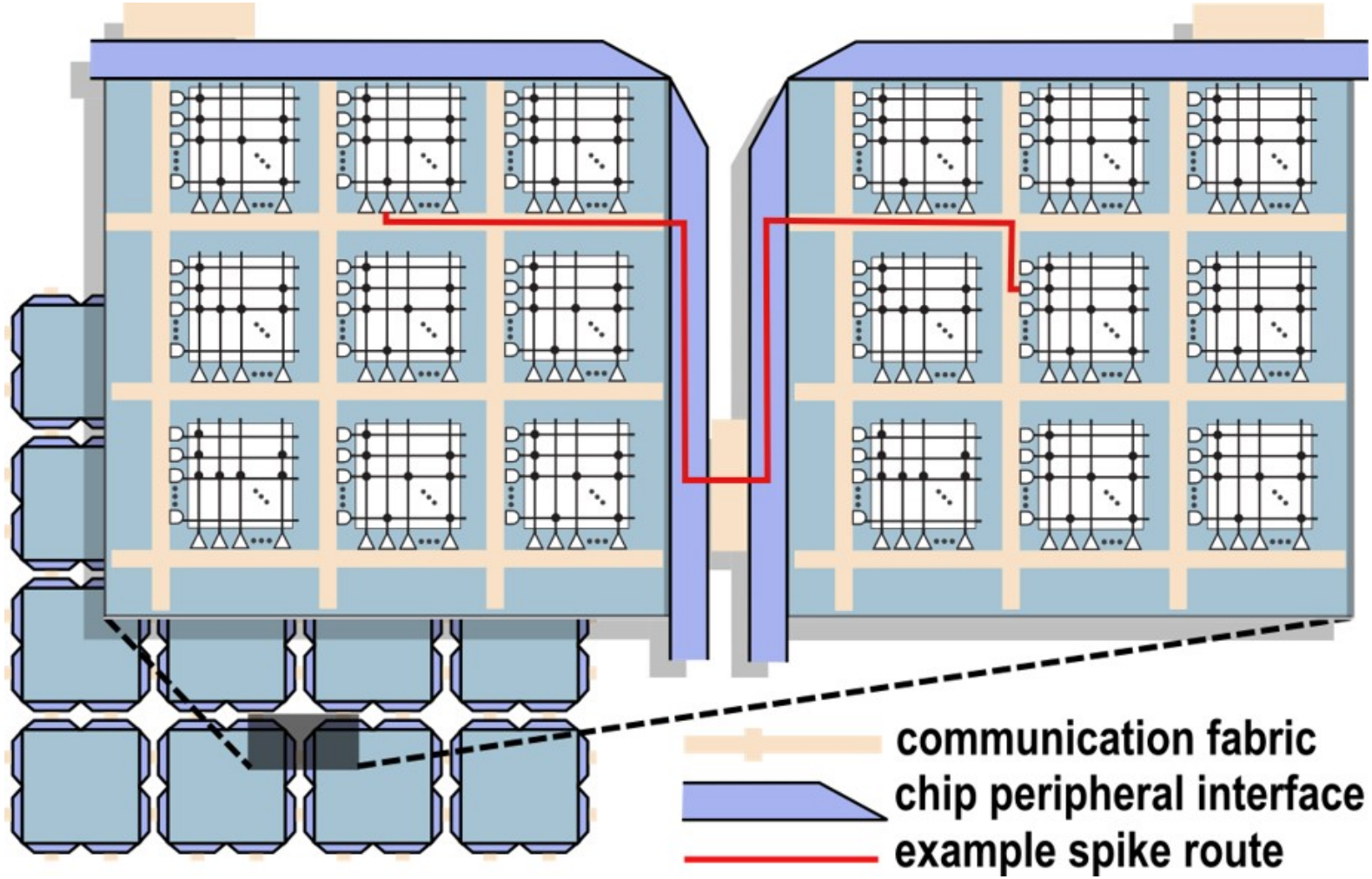
- Chips are designed to talk to each other with no additional circuitry (other than direct connection)
- Designer need not worry about chip boundaries
- Software ("placement") maps model neurons to chip neurons so as to conserve energy, bandwidth



Courtesy of IBM Research – TrueNorth Ambassador Slide Deck

Courtesy of IBM Research – TrueNorth Ambassador Slide Deck

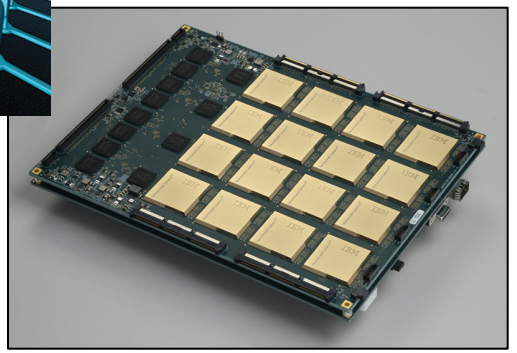
NS16e System Diagram



Livermore is working with IBM on Neuromorphic computing

- Explore how TrueNorth can be applied to problems in the national interest
  - Provide testbed for national labs
- Three thrusts of research:
  - Stand-alone:
    - Embedded, low-power classification & recognition
  - HPC Machine-Learning co-processor:
    - On-line, low-power classification & recognition
  - Neuromorphic co-processor
    - Accelerate new classes of algorithms

Testbed (Mar. 2016)
16M neurons, 4B synapses, ~2-3Watts



TrueNorth operates as a spike coded architecture

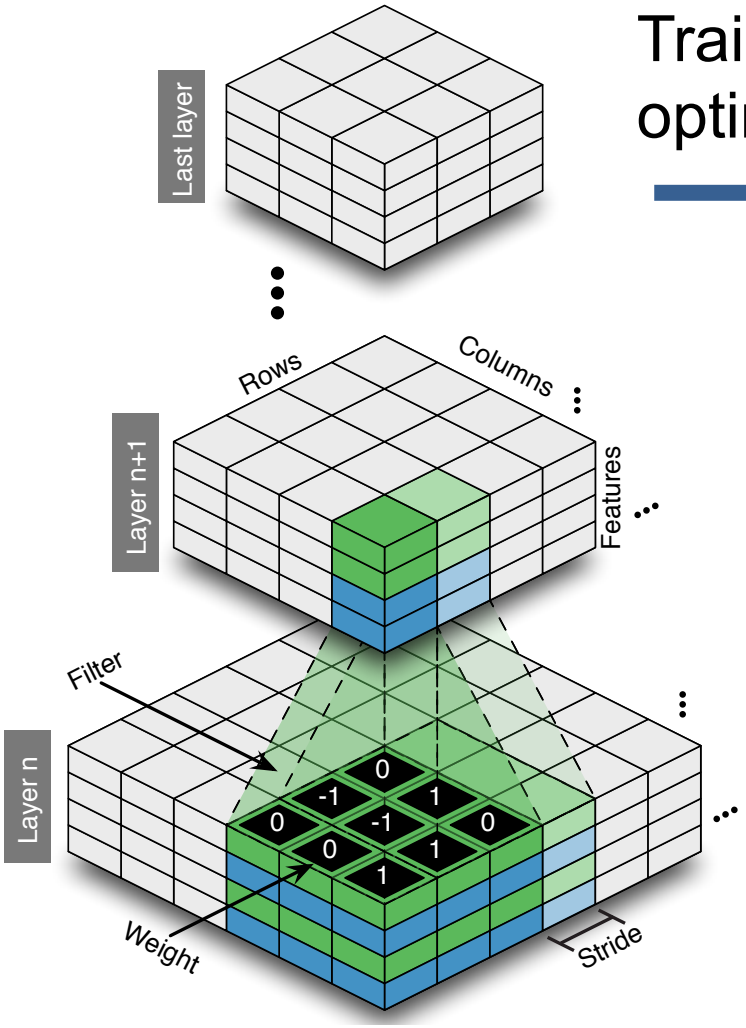
Encoding
(representation)

Decoding
(interpretation)

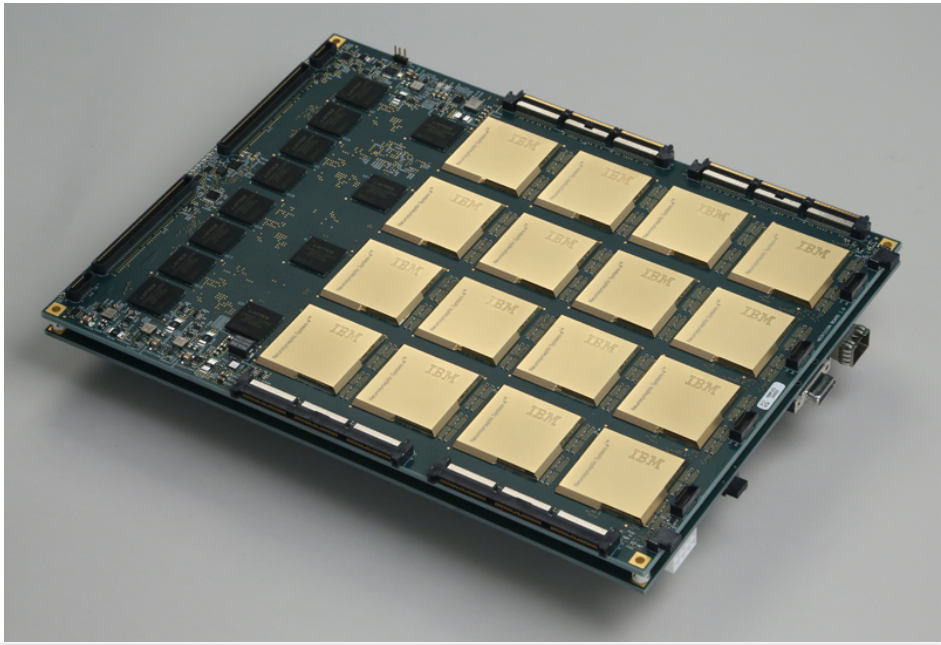
- We must transduce the information into TrueNorth spike representation
  - Spatial encoding (multiple axons / neurons per logical value)
  - Temporal encoding (multiple ticks per logical value)

- **You cannot just drop your matrix into TrueNorth**

Mapping Neural Networks to TrueNorth requires custom toolchain



Train with neural network toolkits that are optimized for TrueNorth architecture



Using TrueNorth to Detect Cars in Overhead Imagery with Context, Clutter, and Occlusion

- LLNL hand-labeled overhead imagery dataset
  - 6 different geographic regions
  - 15cm per pixel at ground level
- Labeled train and test set
  - 32,000 unique cars
  - 58,000 negative targets (look like cars)
  - Amplified data set with 15 degree rotations
- Q: How do TrueNorth Convolutional Neural Network restrictions affect AlexNet?
  - Reduce image size from 256x256 to 56x56
  - Replace first 11x11 filter with 5x7 and 7x5
  - Replace max pooling with average pooling
- A: Performance was not degraded during embedding (e.g. 97.62%)

Using NS16e plus Eedn toolkit to improve car detection DNN

- Original implementation used IBM TNCN toolkit
  - Interim implementation between Caffe+Tea and Eedn
  - Used temporal rate encoding
  - Supports conv-nets

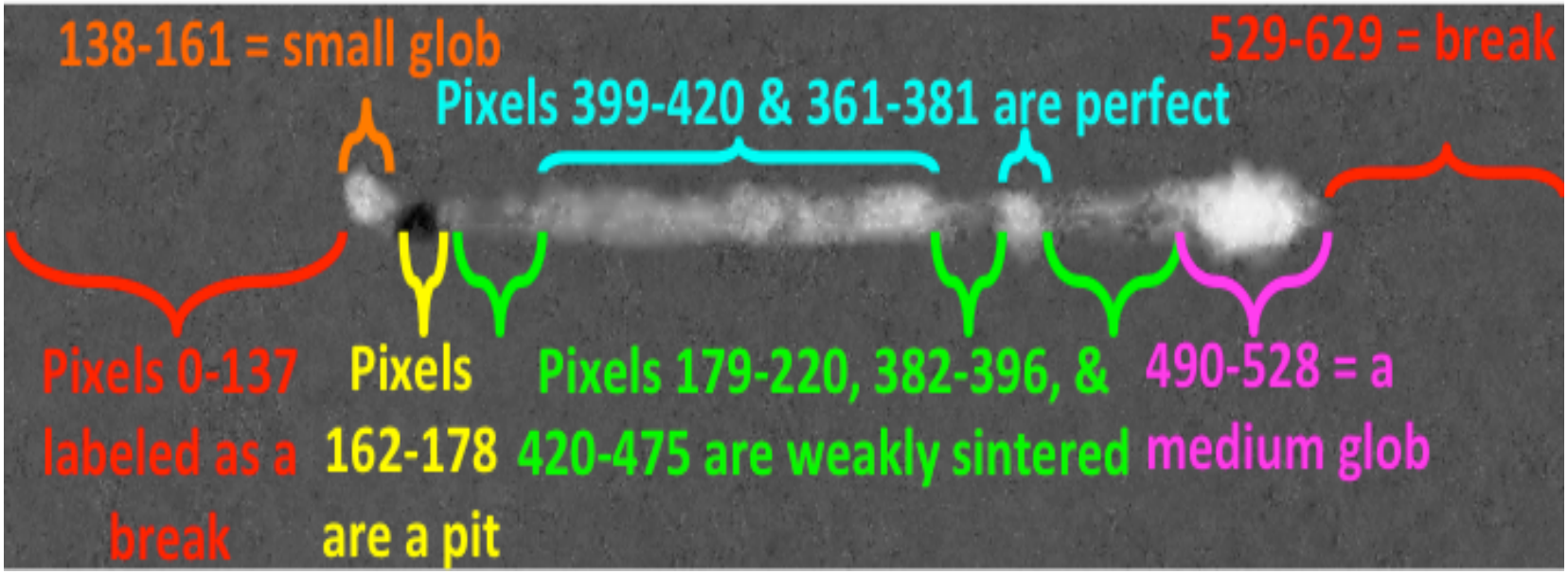
Cores (Chips)	Patch & Stride	Training iters	Code & reset	Train	Test	TN Test
54780 (14)	4x4 / 2	30k, 3k, 9k (x4)	128 / 32	94.53	96.35	97.50

- Switch to new Eedn toolkit
  - Used multi-channel Canny edge filter to encode input into set of binary images
  - Process images in single tick – using 12 channel encoding
  - Supports conv-nets

Cores (Chips)	Patch & Stride	Training iters	Tick Period (ms)	Train	Test	TN Test
12896 (4)	4x4 / 2	80K	1	98.94	59.84	98.64

Defect Detection in Additive Manufacturing with TrueNorth

Stainless steel powder sintered in Argon by a laser moving at 1.2 m/s with 325 Watts
Image: SS_Ar_018_1.2_325.png

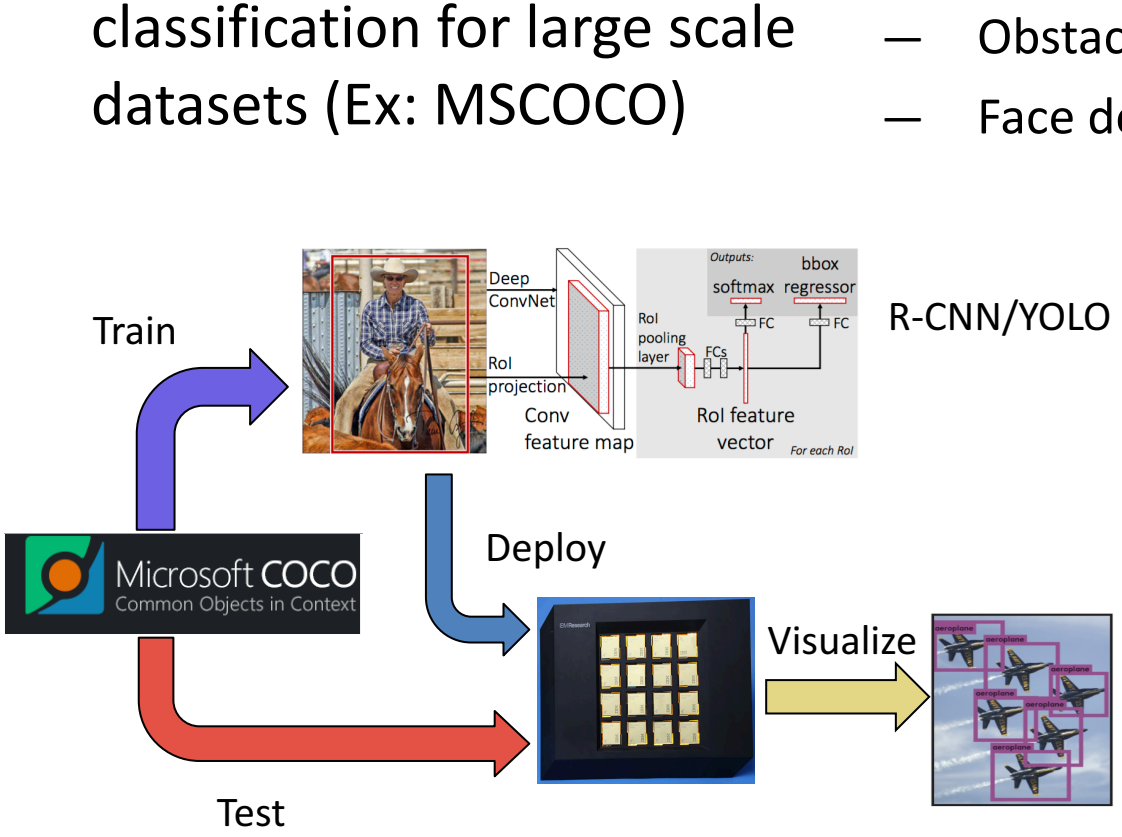


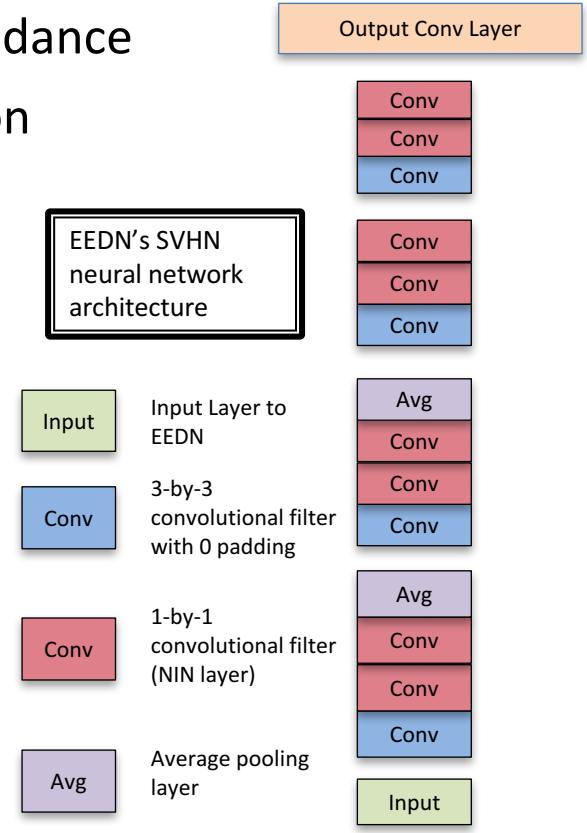
- Classes: 1 =break, 2 =big glob, 3 =medium glob, 4 =small glob, 5 =pit, 6 =weak sinter and 7 =perfect
- Preliminary model accuracy of 81% (based on SVHN model)

Real time image segmentation and object classification

- Perform real time image segmentation and object classification for large scale datasets (Ex: MSCOCO)

- Applications:
  - Video surveillance
  - Obstacle avoidance
  - Face detection





Using NS16e to scaling up R-CNN/YOLO DNN

- Trained using artificial data set (modeled after COCO)
- Scaling up granularity of region proposal:
  - 144 classes (4 Horizontal, 4 vertical and 3 Scaling along x and y)
  - 784 classes (7 Horizontal, 7 vertical and 4 Scaling along x and y)
  - 1600 classes (10 Horizontal, 10 vertical and 4 Scaling along x and y)

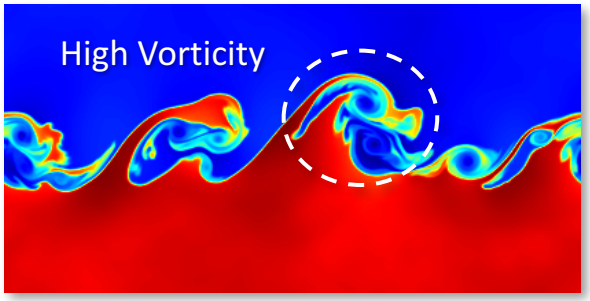
Image Size (in pixels)	# Classes	# Cores	#Chips	Tick Period (ms)	Accuracy in % (NSCS)	Accuracy in % (TN System)
32-by-32	144	4096	1 (1x1)	1	98.0%	99.0%
56-by-56	784	15560	4 (1x4)	100	94.6%	92.0%
72-by-72	1600	24384	7 (2x4)	100	75.0%	

Scaling can be improved with composition of neural networks

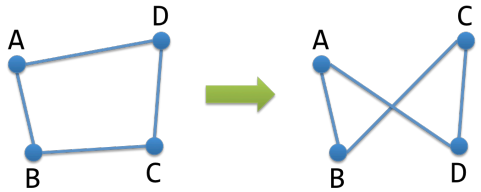
Learning systems will supervise complex simulations in future simulation codes

ALE simulations use dynamic meshes to simulate complex dynamics

- They fail frequently
- Mesh geometry: *mesh zone tangling*
- Physical quantities: *anomalous hot spots*



Goal: Apply machine learning to predict simulation failures and proactively avoid them

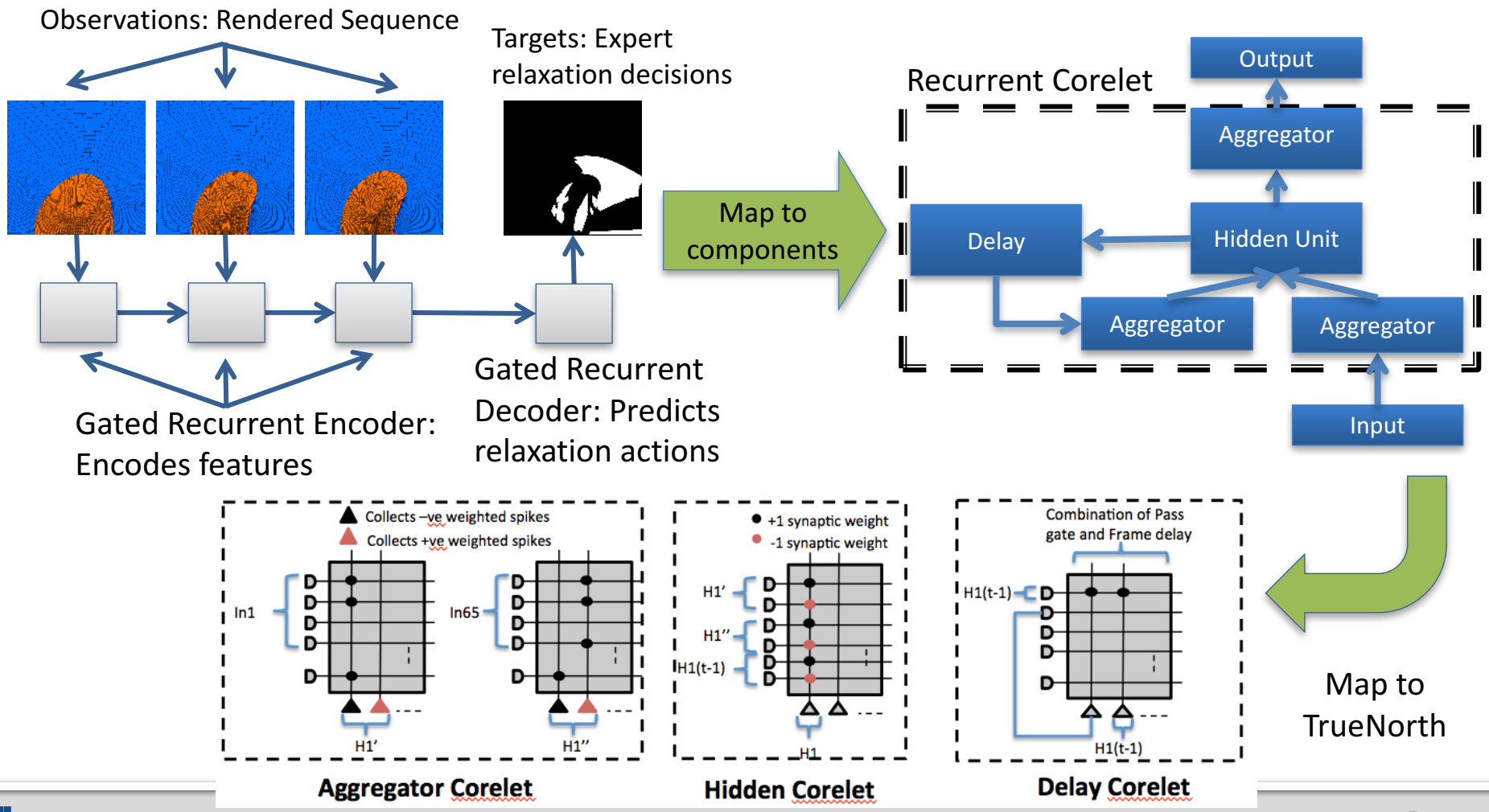


Mesh zone tangling

Feasibility demonstration: *Successfully predicted and automatically avoided* different mesh tangling conditions using 3 test cases – Helium bubble, shock tube, simple hohlraum

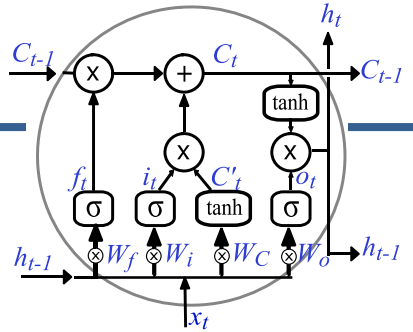
Developing a Versatile Mapping of Recurrent Networks on TrueNorth

- Recurrent model for cloning expert mesh relaxation policies.



Implementing LSTM on TrueNorth

- Temporal behavior uses multi-phase implementation
- Increase precision via:
  - Time-to-spike or rate encoding
  - Spatial encoding

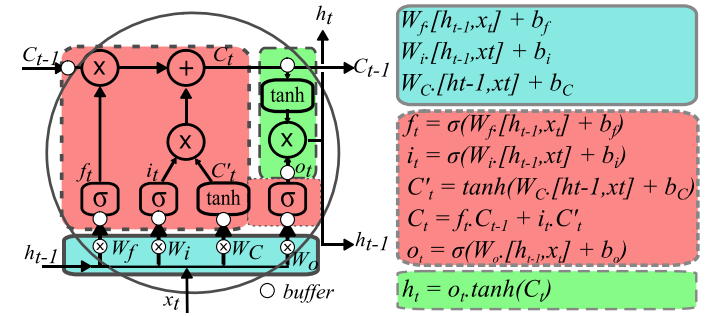


(a)

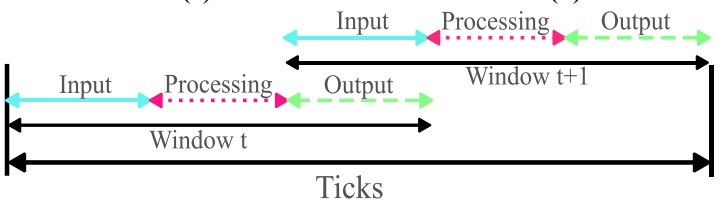
$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f) \qquad i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i)$$
$$o_t = \sigma(W_o.[h_{t-1}, x_t] + b_o) \qquad C'_t = tanh(W_C.[h_{t-1}, x_t] + b_C)$$
$$C_t = f_t * C_{t-1} + i_t * C'_t \qquad h_t = o_f * \tanh(C_t)$$

(b)

Figure 1. (a) Standard LSTM (b) LSTM equations

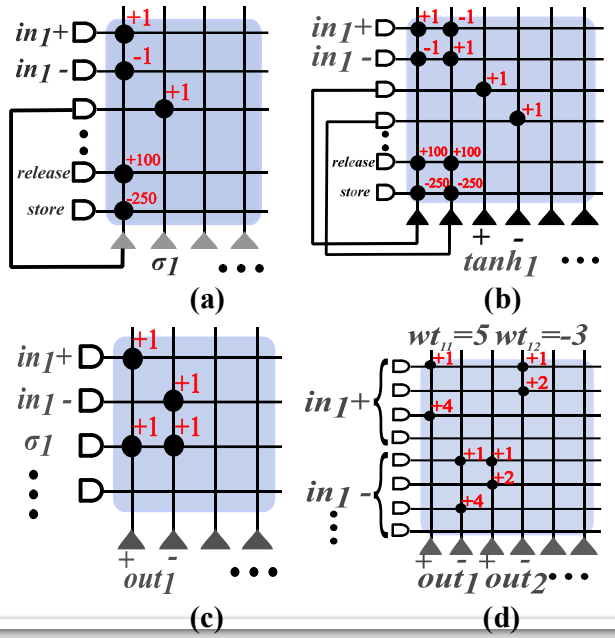


(a) (b)



(c)

Figure 2. (a) LSTM color coded based on operation phase (b) LSTM equations color coded to represent operations in specific phases (c) 3 phases and partial pipelining



(a) (b) (c) (d)

Figure 5. (a) Sigmoid module (b) Tanh module (c) Dot product module (d) IC module

Convolutional Sparse Coding (CSC) on TrueNorth

- Convolutional kernels define a Dictionary and Sparse Feature Maps (SFMs) are generated through a training process using CSC algorithm shown Figure 1.
- TrueNorth implementation flow diagram for image reconstruction is shown in Figure 2.
- Figure 3 and 4 shown experimental results on MNIST and CIFAR-10 dataset respectively.

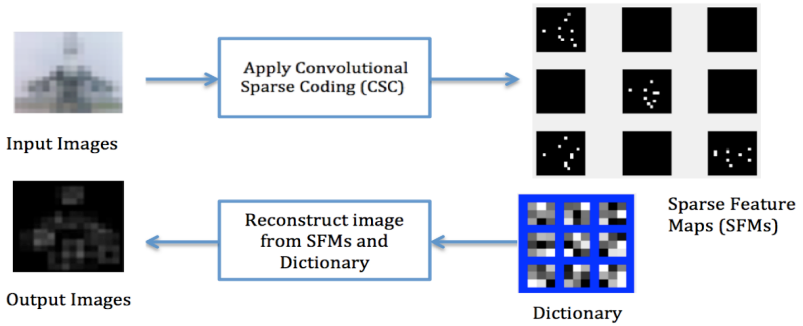


Figure 1. Training and reconstruction on CPU

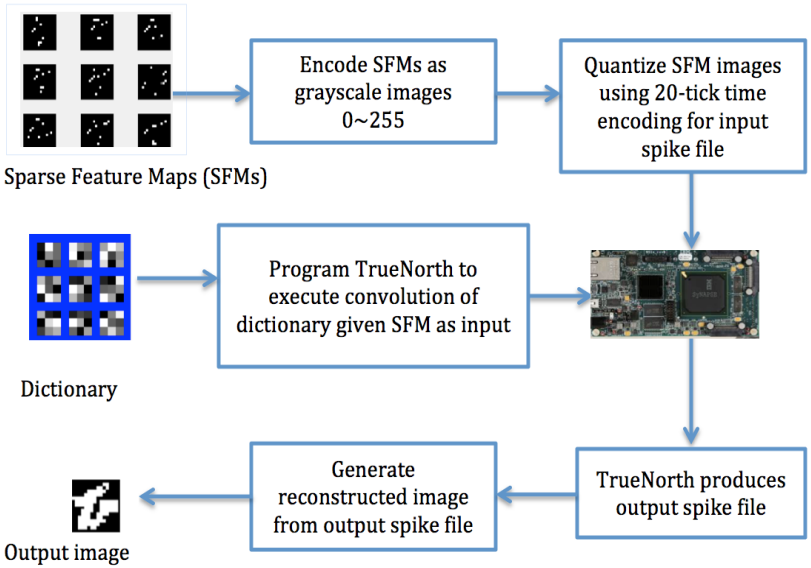


Figure 2. TrueNorth implementation flow diagram

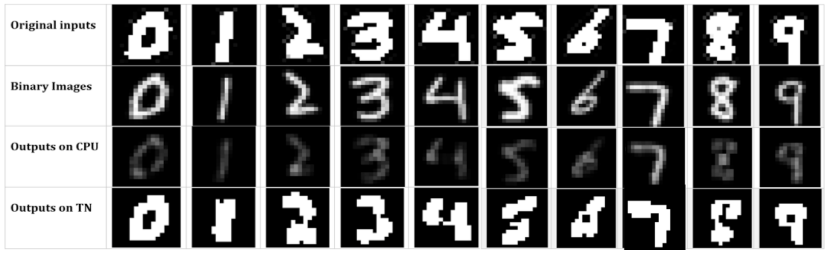


Figure 3. Experimental outputs on MNIST dataset

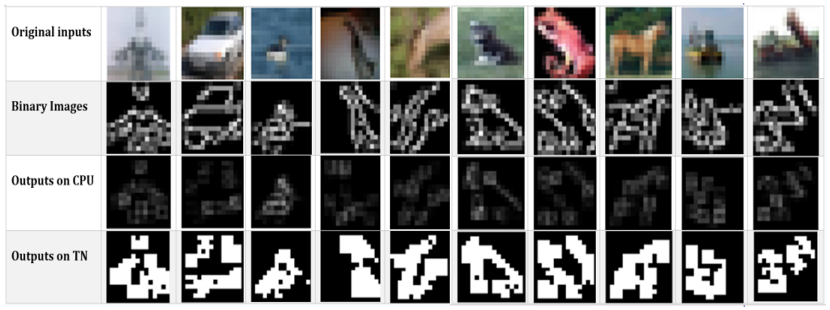
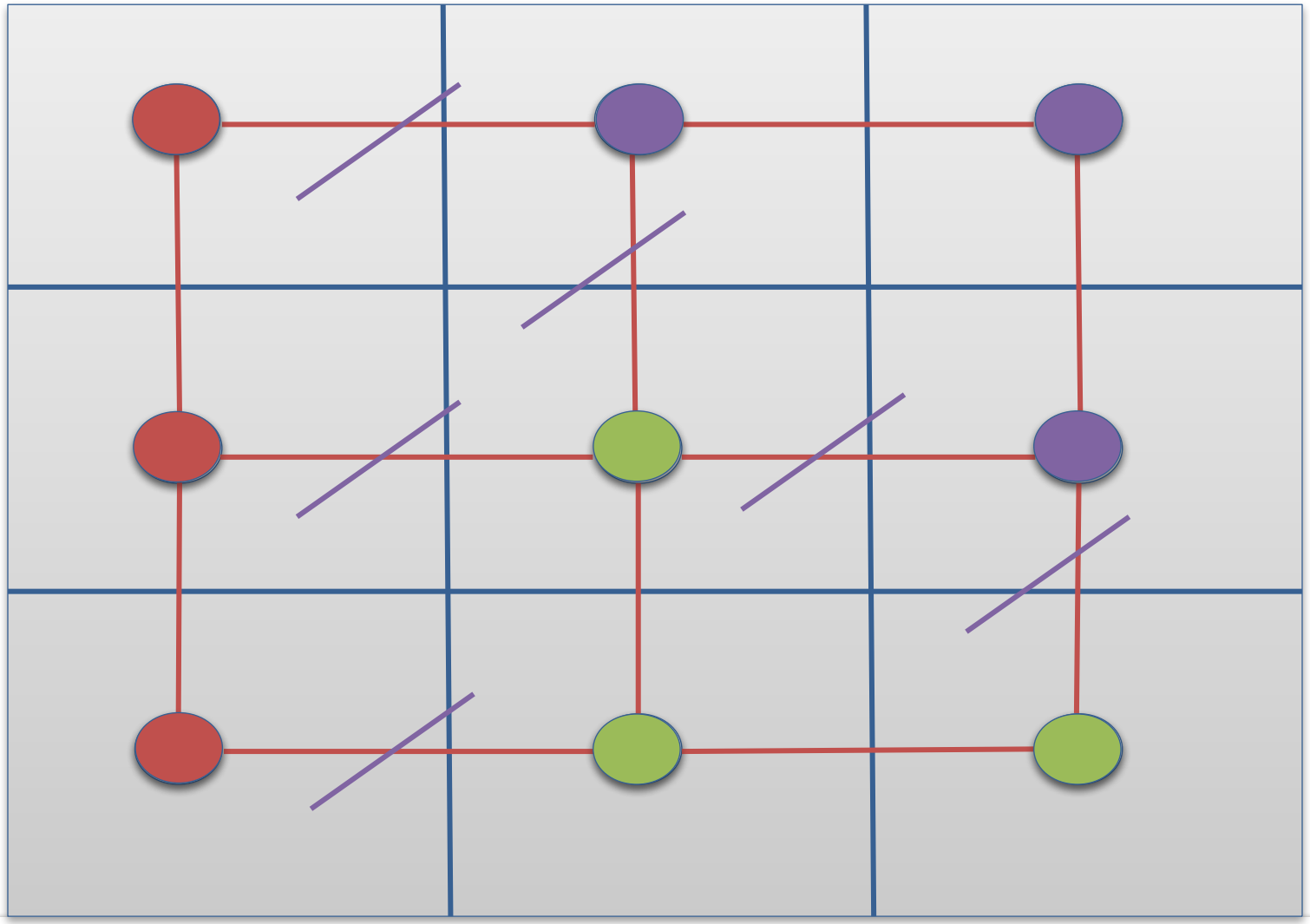
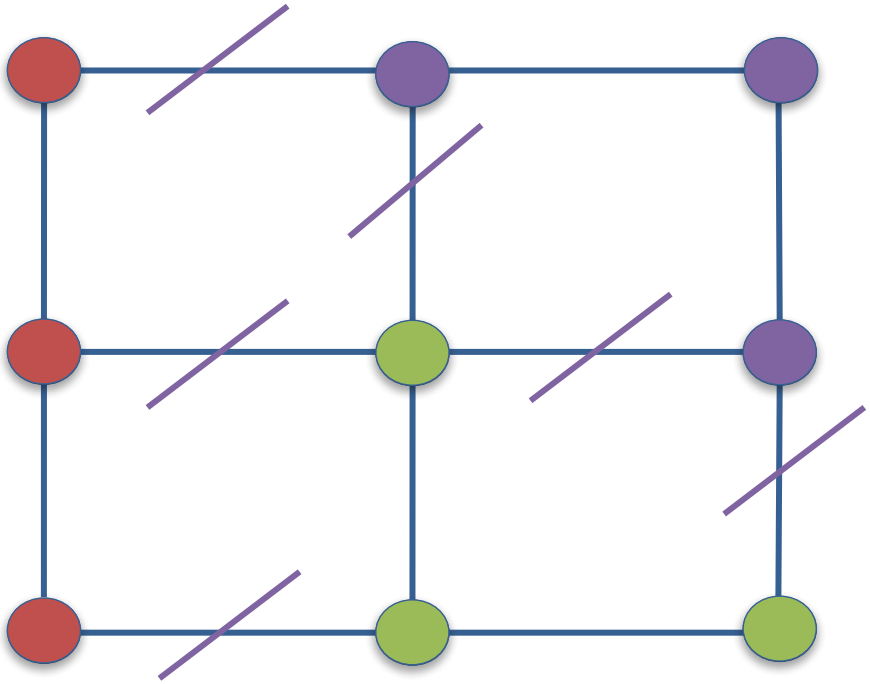


Figure 4. Experimental outputs on CIFAR-10 dataset

Spiking Algorithms Applied to Graphs, Partitioning for Domain Decomposition



Graph Partitioning



Goal: Minimize the maximum partition node weight sum **and** minimize the global sum of cut edge weights.

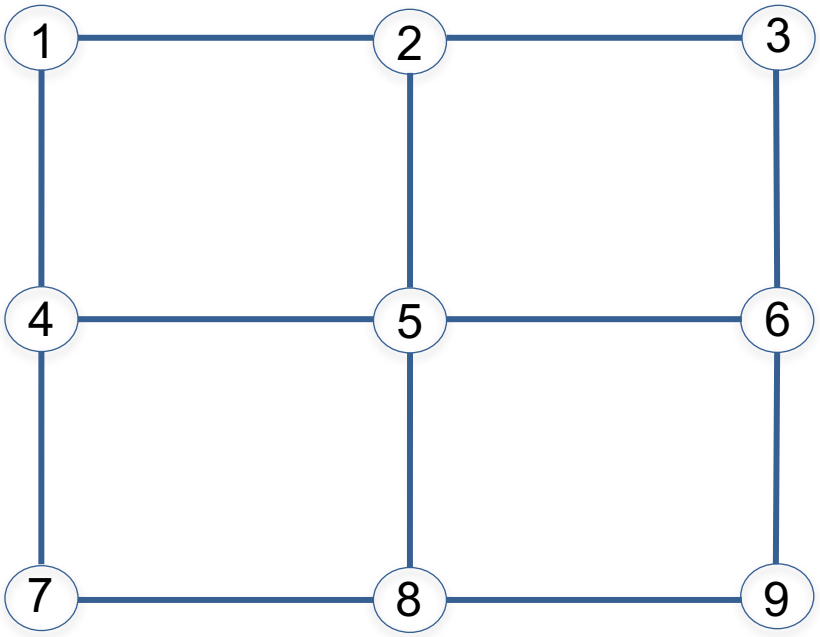
Both nodes and edges can be weighted (not shown).

Graph partitioning is NP-hard.

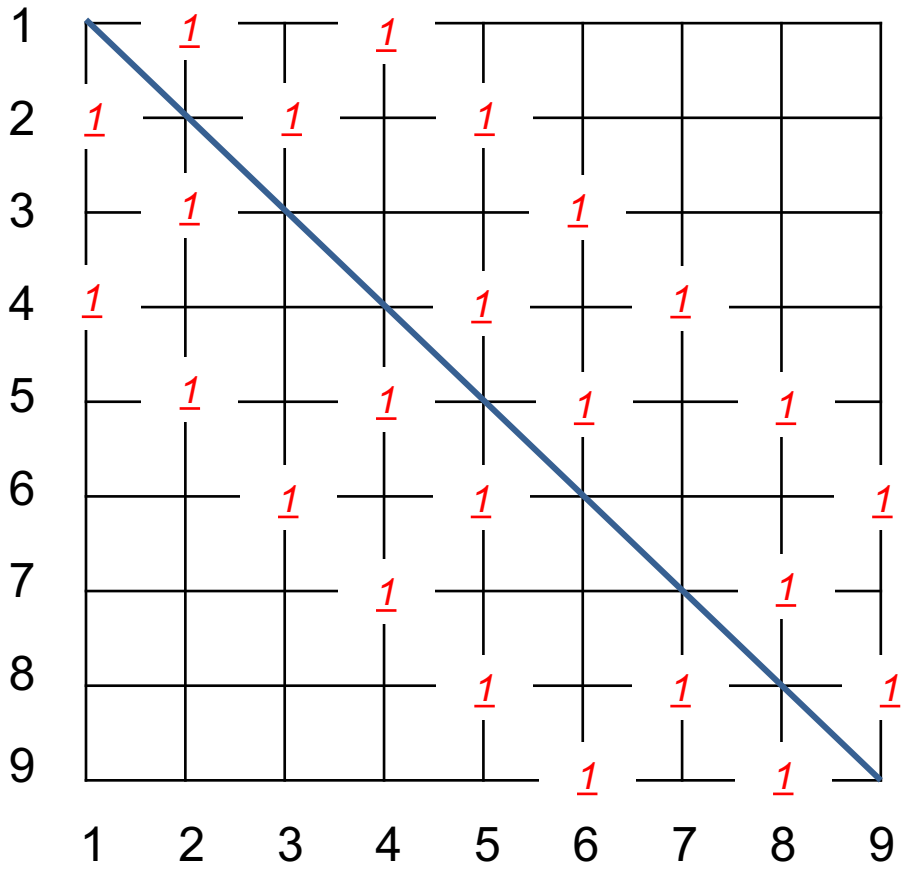
Minimum partition node weight sum = 3
Maximum sum of cut edge weights = 6

The brute force solution is to try all possibilities

Graph Partitioning

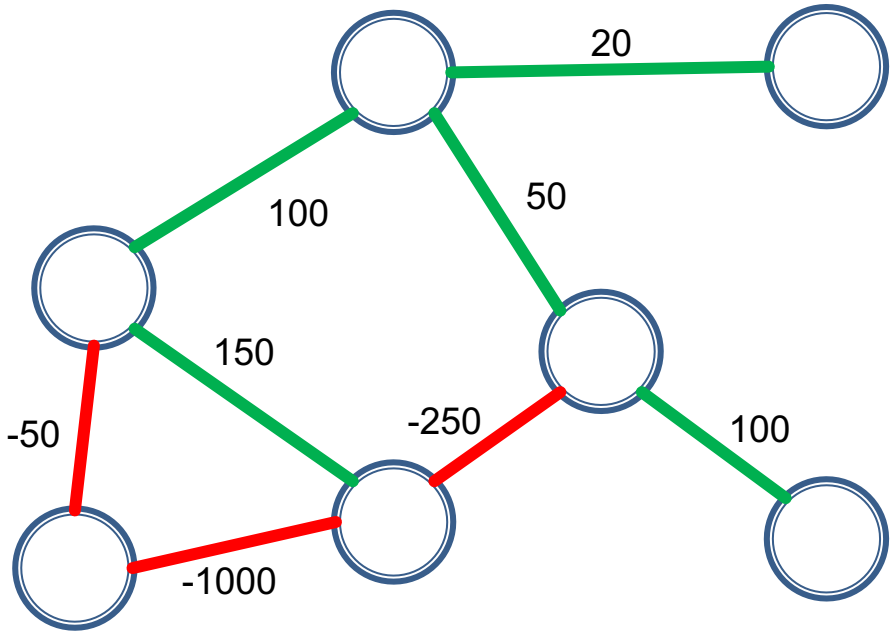


Exploring methods of mapping graph partitioning into a QUBO problem

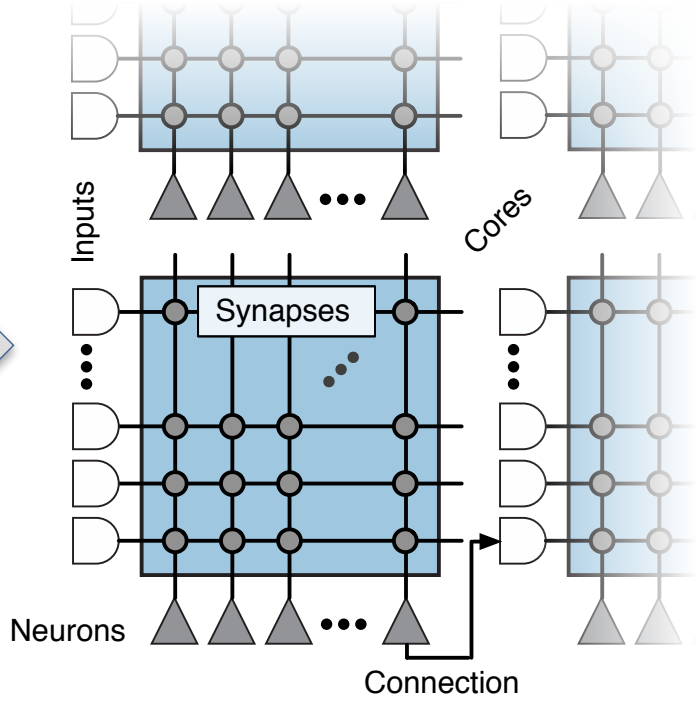


Mapping QUBO (Quadratic Unconstrained Binary Optimization) onto the TrueNorth array

- **Input:** An undirected, weighted graph
- **Goal:** Activate some subset of the nodes to maximize a *score function*
- **Score Function:** Add up the edge weights between any pairs of nodes that are both activated.

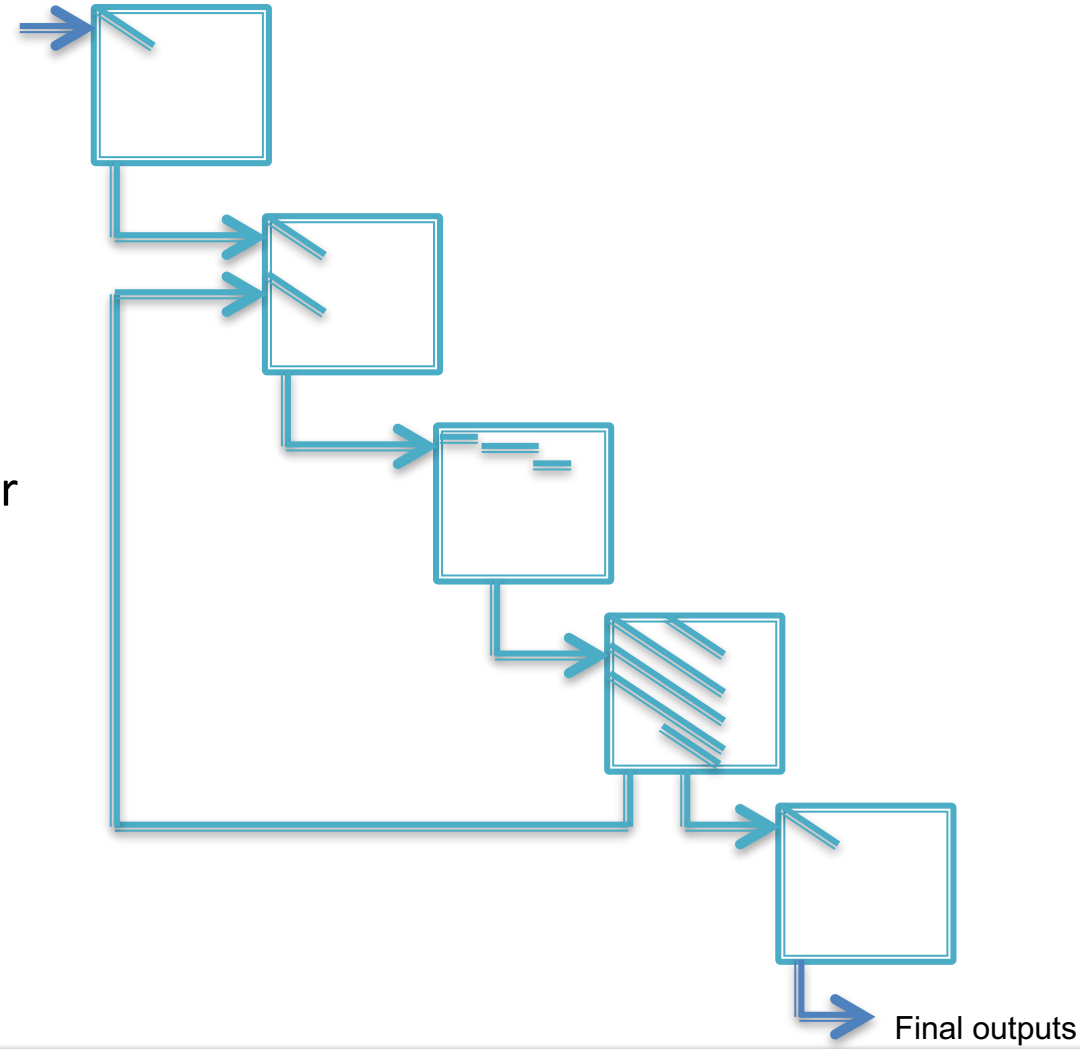


Map



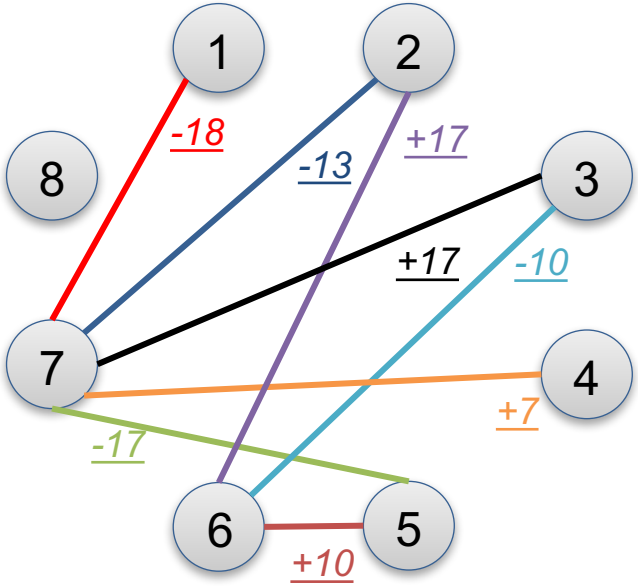
Block diagram of (one of our) QUBO corelets

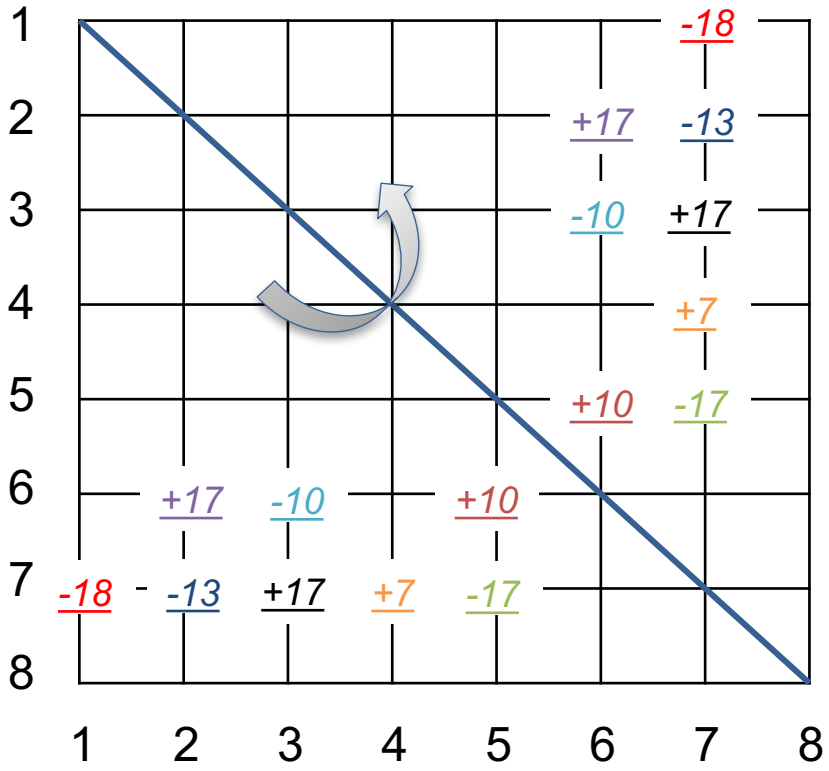
- Random input generator
- Combine random input with recurrent feedback
- Integrate input from neighbor nodes
- Split output for recurrent feedback
- Produce final output



Challenges of mapping k-color graphs

Mapping QUBO maps to IBM's TrueNorth core

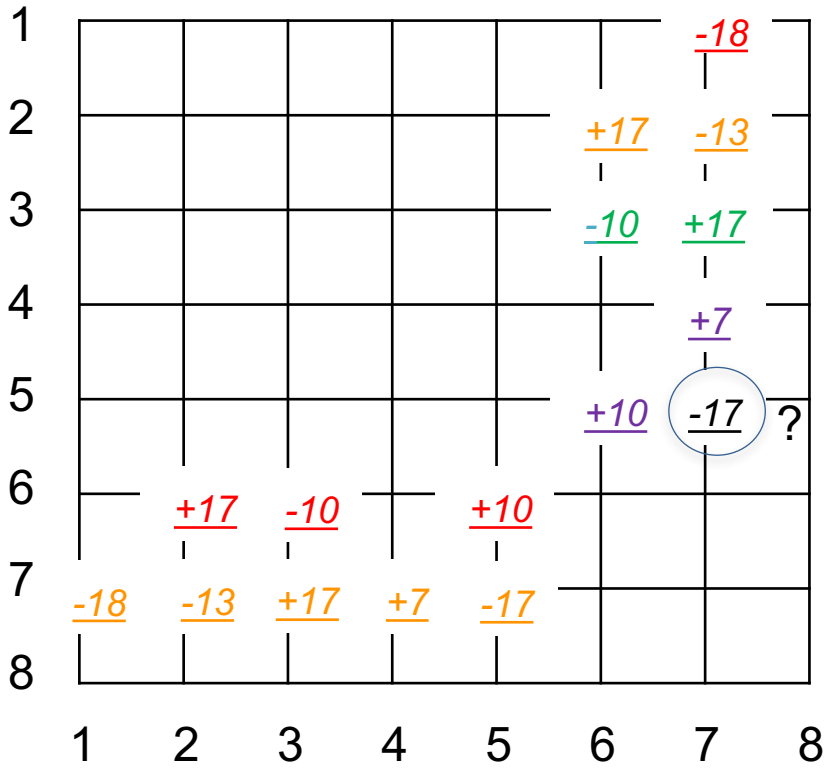




There are restriction due to efficiency

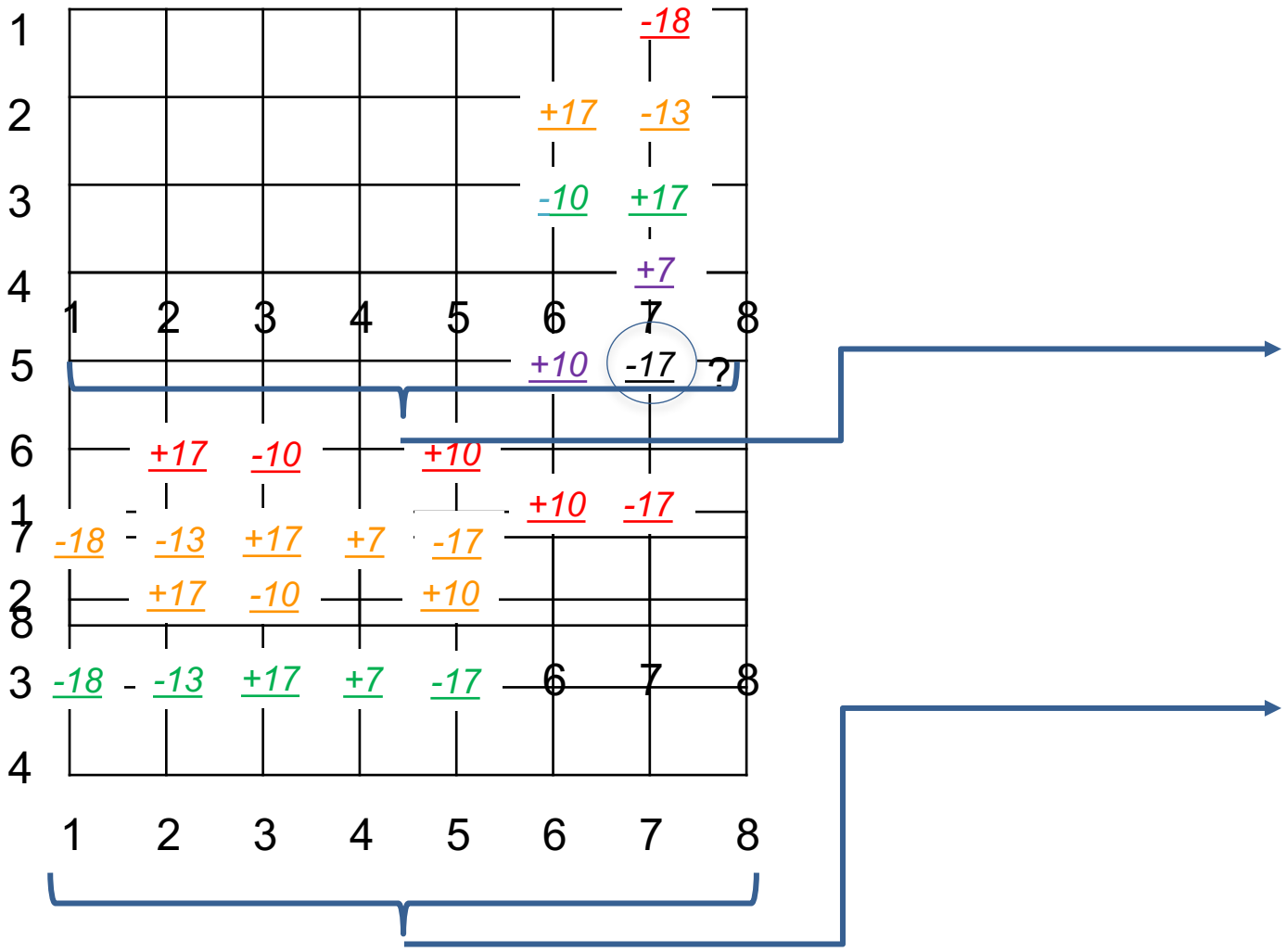
TrueNorth only allows 4 "axon types"

- Each neuron defines up to 4 weights as `S0`, `S1`, `S2`, and `S3`
- Each axon can only map to one value of `S0`, `S1`, `S2`, and `S3`
- If you go beyond `S3` you need to get creative

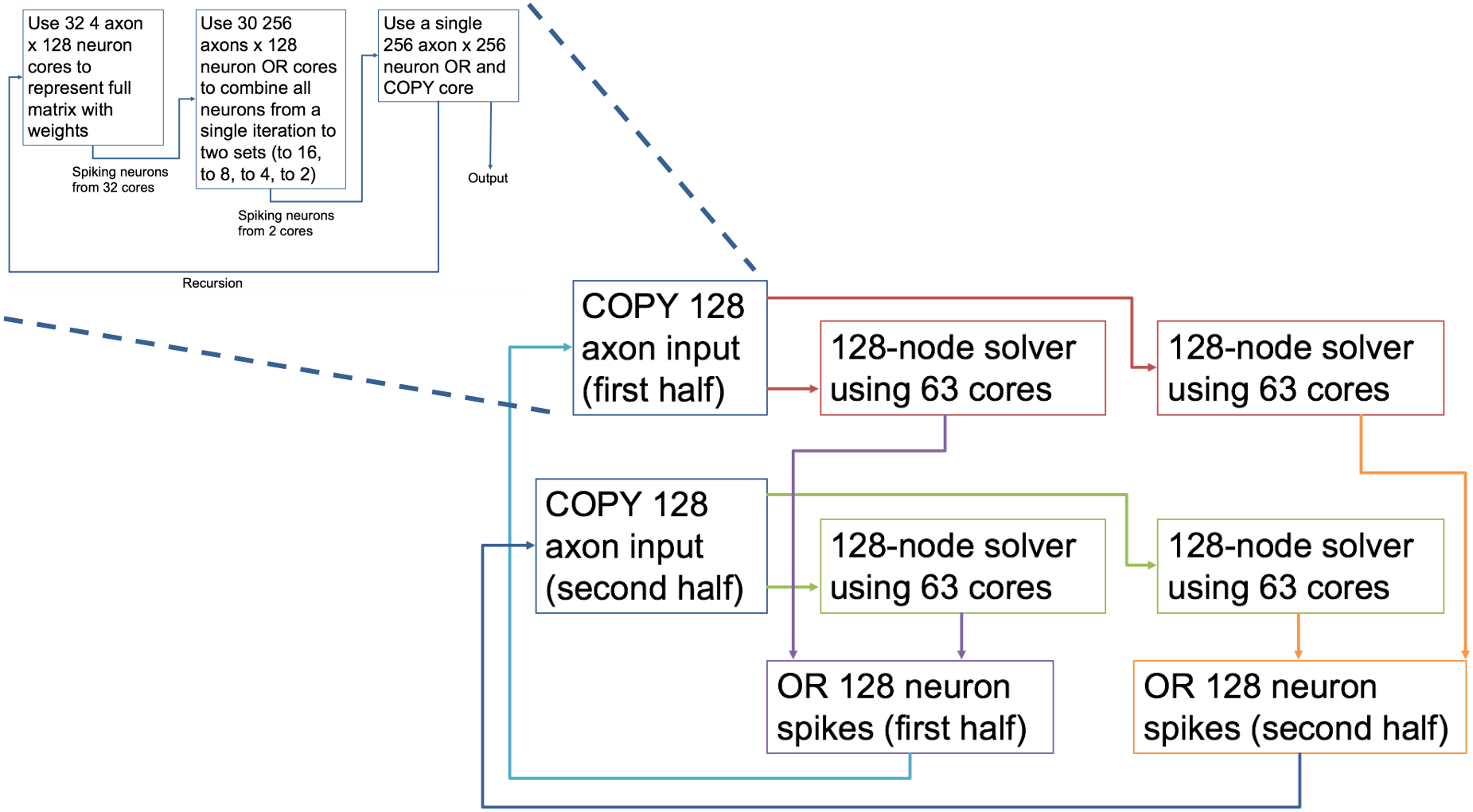


QUBO **for a ~(4-color) graph** maps well to a TrueNorth core

But, what if I have more than degree >4?

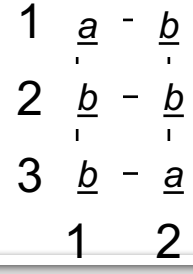


Compose hierarchical structures that grow quadratically in the size of the graph



Improving QUBO embedding efficiency

- Use TN hardware sparsely (e.g. 4 axons x 128 neurons)
  - Allows for “arbitrary” synapse weights, but unnecessary in many applications
- Create k-colorable graph (approximate solution)
  - Quantize graph weights
  - Collapse unique graph weights to ~4 buckets
  - Reducing to **n** weights does not guarantee that you only need **n** S values
- Be clever - evaluating splitting different S values to different cores (after quantization)



Neuron 1: S = [a, b]
Neuron 2: S = [b, b]
Axon 3 maps to S[?]

Developing new TN capabilities

- We are working with LANL to compare the performance of D-Wave with TrueNorth on image reconstruction problem
  - LANL: mapping sparse coding problem to QUBO for execution on D-Wave
  - LLNL: implement sparse coding directly on TrueNorth
- Scaling up QUBO graph on TrueNorth to large graphs
  - Aggregate signals across cores

Exploring potential TN-networks for:

- Processing radiographic imagery
- Executing transport sweeps
- Hyperspectral image analysis

Develop TN-compliant open source deep learning toolkit

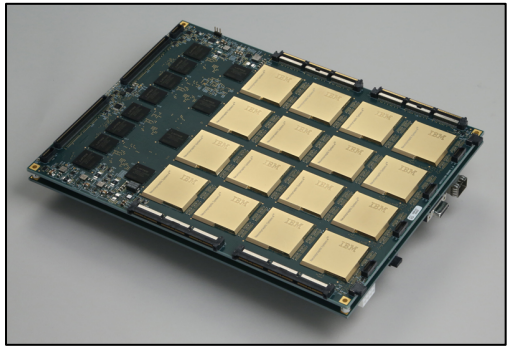
- Scalable training for large models and data sets
- Extensible for new learning algorithms and layers
- Utilize Livermore Big Artificial Neural Network (LBANN) toolkit
  - Implement binary connect learning algorithms
  - Provide output bindings for mapping neurons to logical cores
  - Output logical description of neural network to IBM intermediate representation
  - Use NetworkImporter patch to ingest logical network description into TN-backend toolchain
    - Reuse IBM's place & route, corelet parameterization, TN configuration generation

We are looking at other Neuromorphic Computing architectures – Beyond TrueNorth

- Memristor-based technologies
- Other neuromorphic architectures
  - Intel
  - HRL Laboratories (formerly Hughes Research Laboratories)
  - Academic & Government Research (U. Dayton, ORNL)

We are exploring how TrueNorth can fit within the DOE's mission space

- Provide a testbed and support for national labs
- Focusing research efforts on three thrusts that are increasingly speculative
  - Stand-alone (Embedded)
  - HPC Machine-Learning co-processor
  - Neuromorphic co-processor
- Push the limits of the toolchain and architecture
- Part of the DOE Beyond Moore's law initiative



LLNL Neuromorphic Research Team

- Neuromorphic Researchers:
  - Brian Van Essen
  - Katie Lewis
  - Adam Moody
  - Dave Widemann
  - Keith Henderson
  - Braden Soper
- PhD Summer Interns
  - Rohit Shukla (U. of Wisconsin-Madison)
  - Amar Shrestha (Syracuse University)
  - Zahangir Alom (U. of Dayton)
- LLNL Collaborators
  - Ming Jiang
  - Brian Gallagher
  - Aaron Wilson
  - Sachin Talathi
- LANL Collaborators
  - Garrett Kenyon
  - Nga Nguyen


Lawrence Livermore
National Laboratory