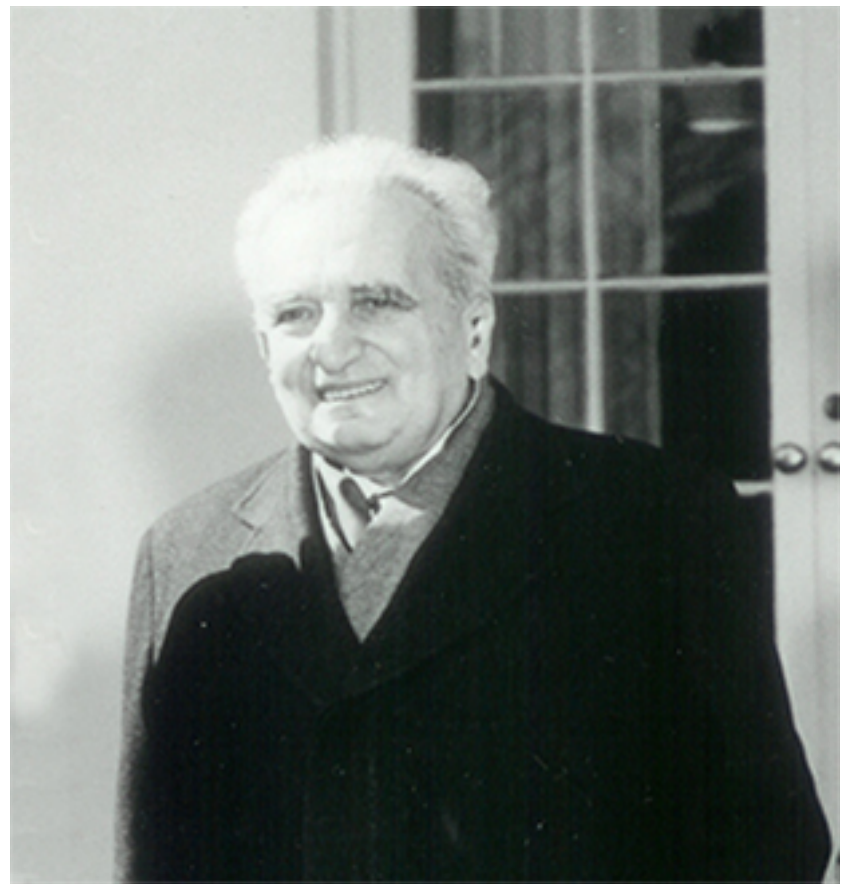
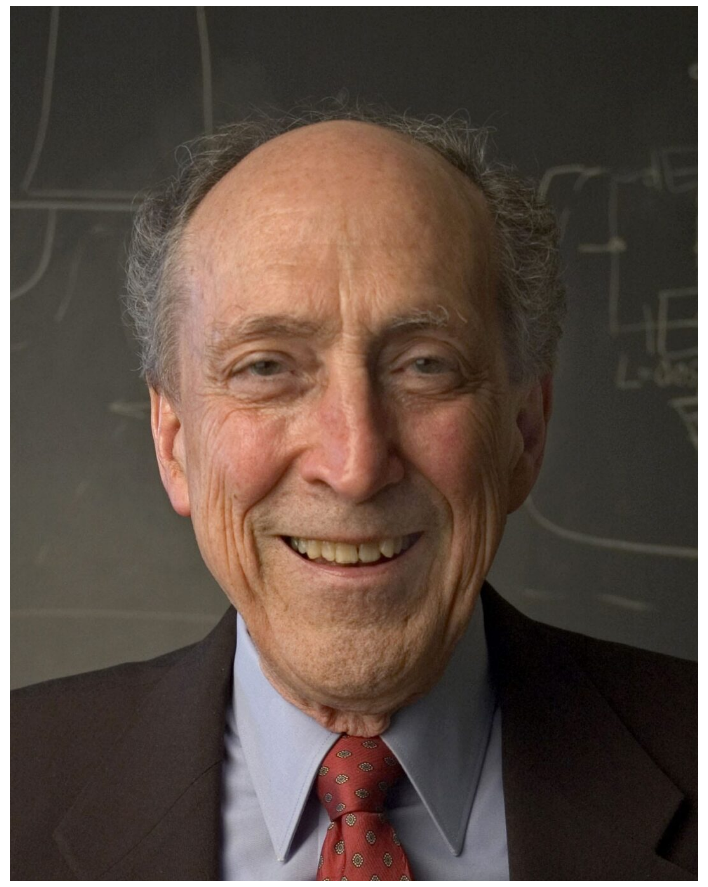
  

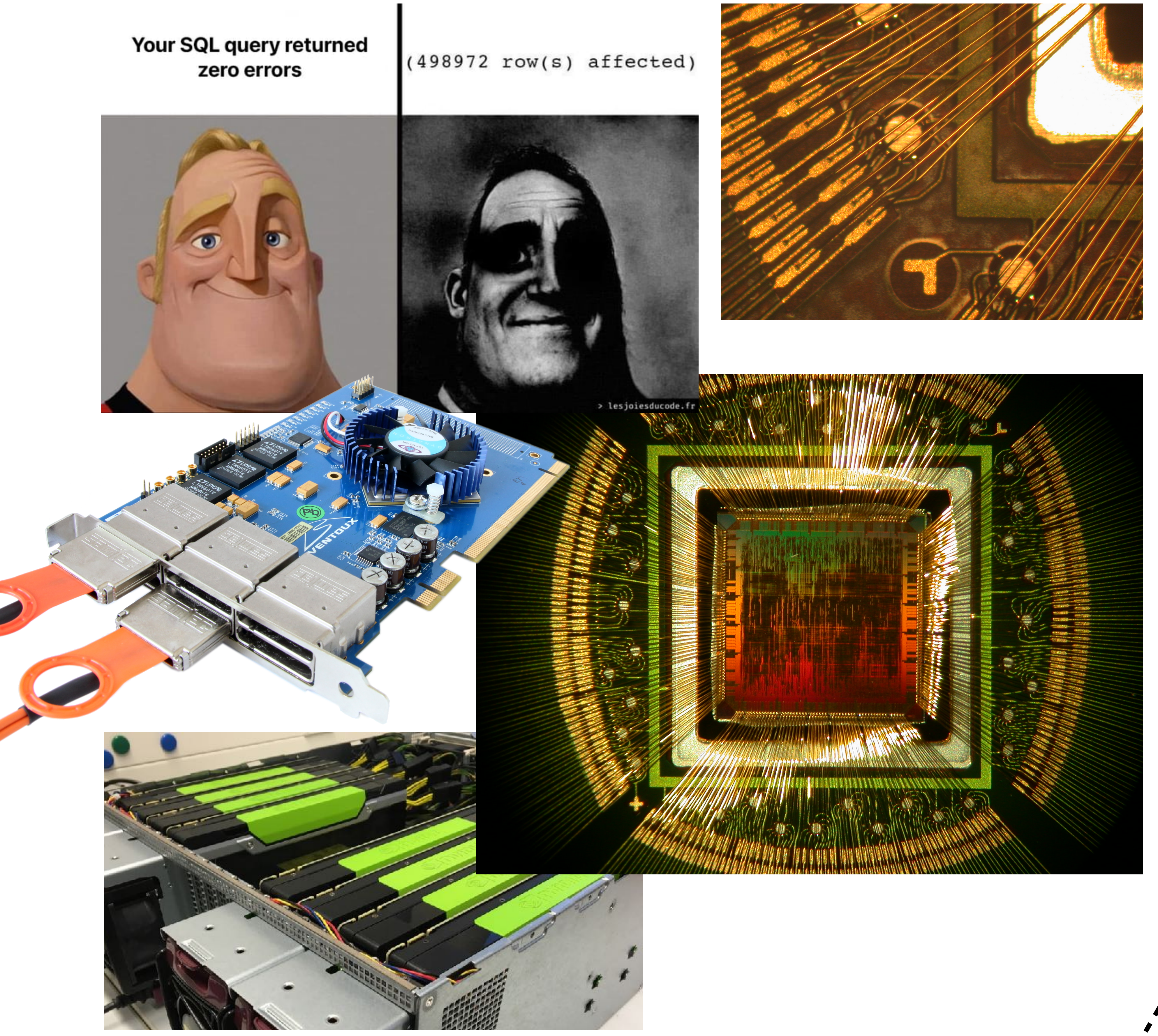
POST-DENNARD PERFORMANCE SCALING

HOLGER FRÖNING
HOLGER.FROENING@ZITI.UNI-HEIDELBERG.DE
COMPUTING SYSTEMS GROUP, INSTITUTE OF COMPUTER ENGINEERING (ZITI)
HEIDELBERG UNIVERSITY

SINO-GERMAN WORKSHOP, XI’AN, OCT 10-16, 2024

RESEARCH BACKGROUND

From: database engineer, HW designer (ASICS, FPGA), HPC



$$\mathbf{x}^l = \Phi(\mathbf{W} \oplus \mathbf{x}^{l-1} + \mathbf{b}^l)$$

Neural Architectures

Compiler

Plethora of HW

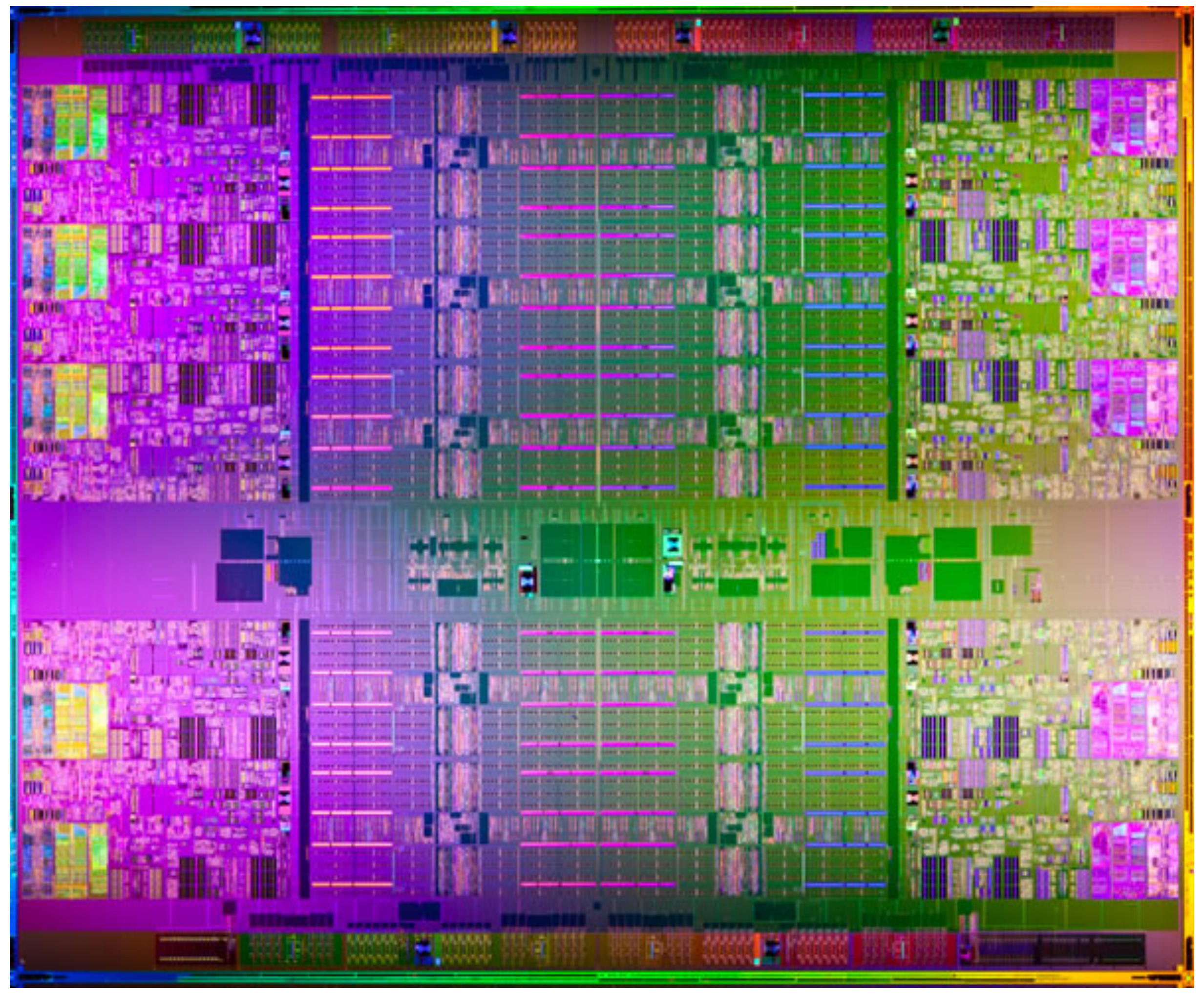
$$perf[\frac{\text{ops}}{\text{s}}] = p[Watt] \cdot e[\frac{\text{ops}}{\text{J}}]$$

$$P = afCV^2 + VI_{leakage}$$

To: vertically integrated approach to efficient ML => HW systems for AI

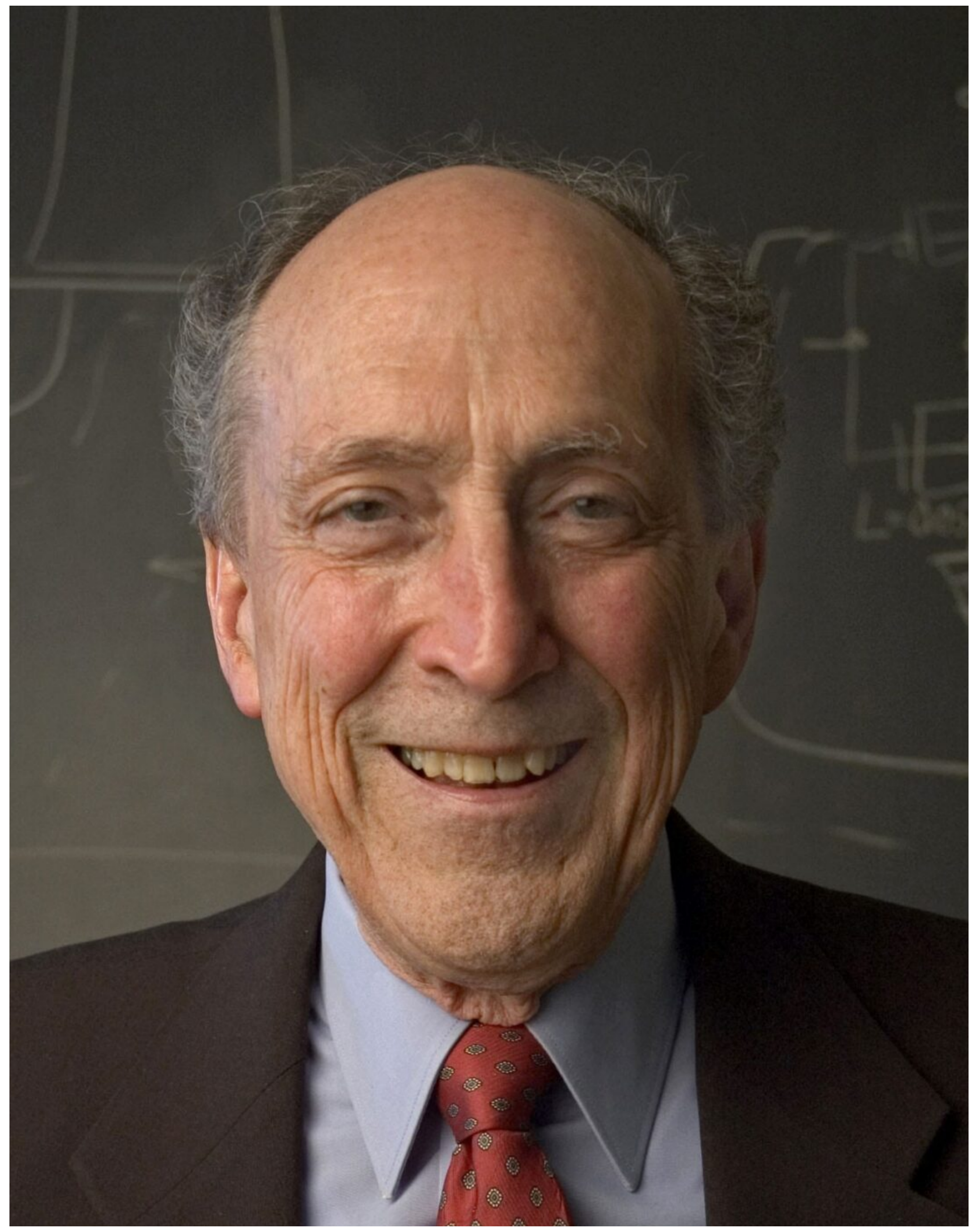
$$a_y = a_{y=0} \cdot 2^{y/2}$$

$$s.t.\ \text{argmin}\frac{\$}{t}$$



$$P = afCV^2$$

CMOS TECHNOLOGY TRENDS & IMPLICATIONS



Governed by Moore & Dennard

POST-DENNARD PERFORMANCE SCALING

$$perf\left[\frac{\text{ops}}{\text{s}}\right] = P[\text{W}] \cdot e\left[\frac{\text{ops}}{\text{J}}\right]$$

$$= P \cdot (e_{op} + e_{mem})$$

Energy is additive

Operation

$$e_{op} = f_{proc}(t, b)$$

GPU CPU FPGA

CMOS-BASED, ARCH DEP.

Memory access

$$e_{mem} = f_{mem}(d, b)$$

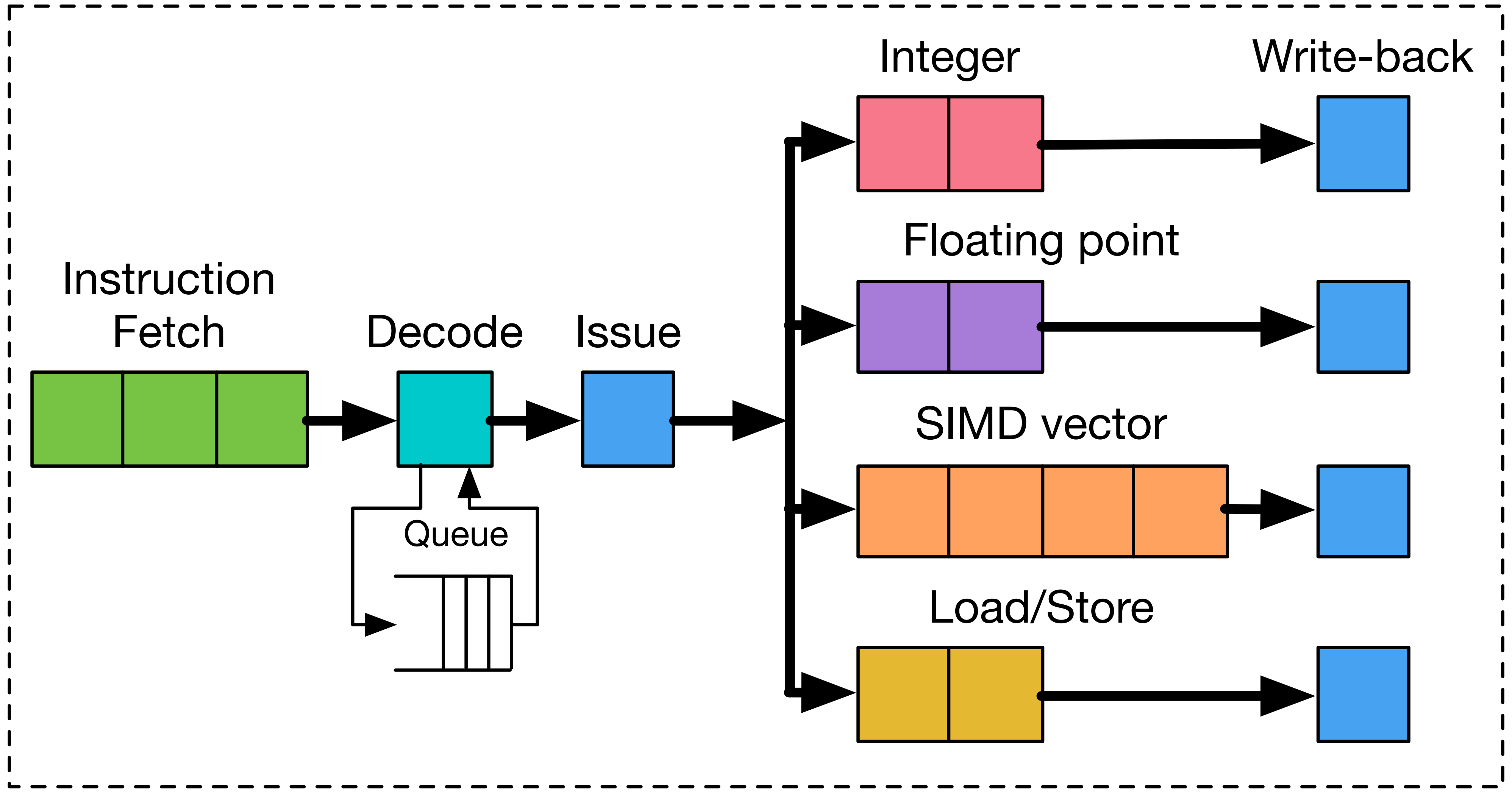
SRAM HBM DRAM

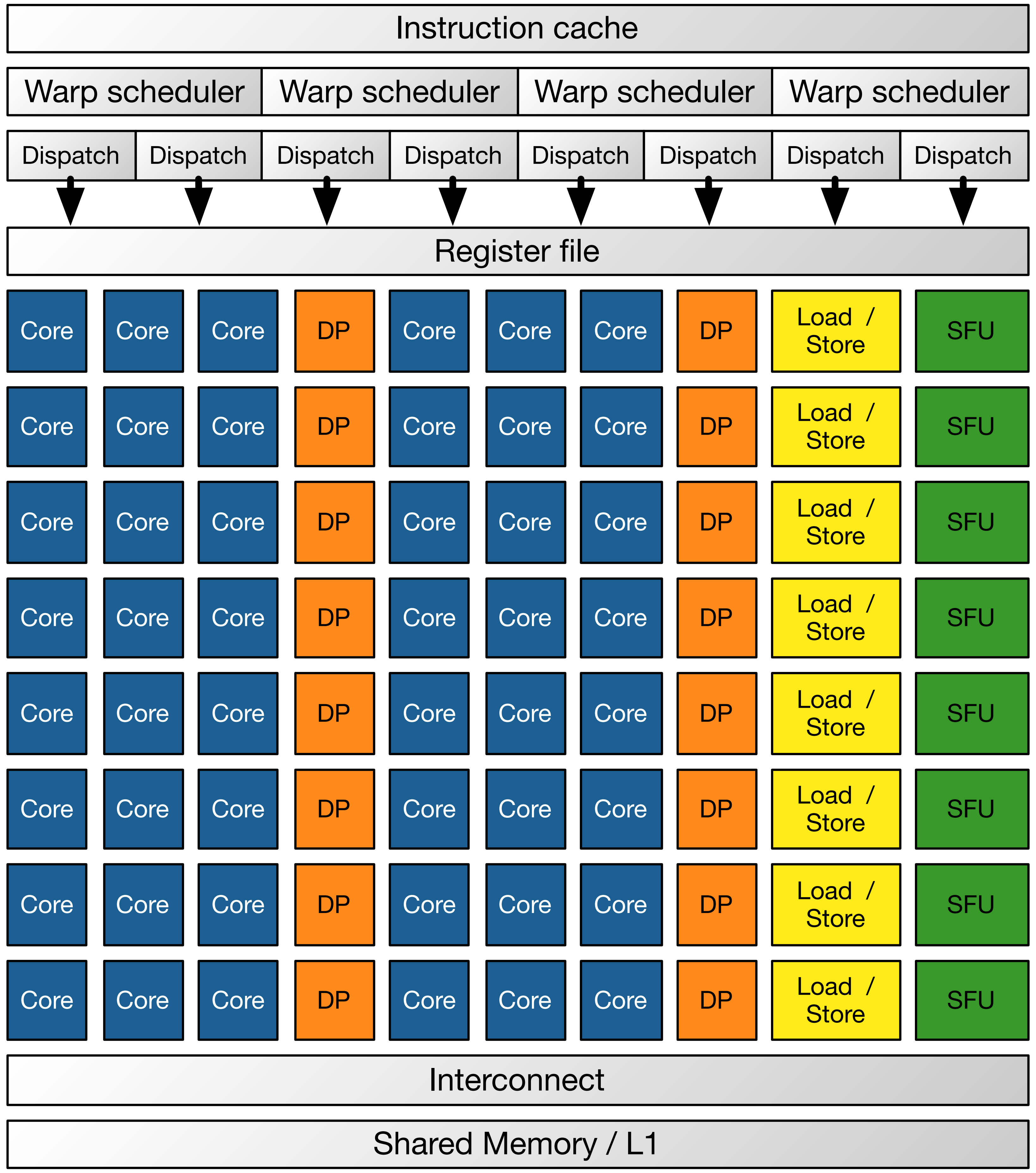
TECHNOLOGY DEP.

Power p, energy e, data type $t[\{float, int\}]$, bit width b, distance d[mm]

PARALLELISM, LOCALITY, STRUCTURE AND PREDICTABILITY

$$P = afCV^2 + VI_{leakage} \propto f^3$$





Frequency reduction
In-order pipelines

Replication

Massively parallel
Energy efficient

PARALLELISM, LOCALITY, STRUCTURE AND PREDICTABILITY

45nm, 2014

Integer	pJ
Add	
8 bit	0.03
32 bit	0.1
Mult	
8 bit	0.2
32 bit	3.1

FP	pJ
FAdd	
16 bit	0.4
32 bit	0.9
FMult	
16 bit	1.1
32 bit	3.7

Memory	pJ
SRAM	(64 bit)
8kB	10
32kB	20
1MB	100
DDR4	1300 - 2600

Computations are of little importance in comparison to memory accesses

M. Horowitz, "1.1 Computing's energy problem (and what we can do about it)," 2014 IEEE International Solid-State Circuits Conference Digest of Technical Papers (ISSCC). doi: 10.1109/ISSCC.2014.675732

PARALLELISM, LOCALITY, STRUCTURE AND PREDICTABILITY

7nm, 2021

👀 @ Bernhard, Hendrik, Kazem, Gregor, Lena

Integer	pJ
Add	
8 bit	0.007
32 bit	0.03
Mult	
8 bit	0.07
32 bit	1.48

FP	pJ
FAdd	
16 bit	0.16
32 bit	0.38
FMult	
16 bit	0.34
32 bit	1.31

Memory	pJ
SRAM	(64 bit)
8kB	7.5
32kB	8.5
1MB	14
DDR4	1300 -
HBM2	250 - 450

FPGA?

Photonic?

CNTCMOS?

MRAM?

RRAM?

Ratios got more extreme over time, HBM came to a rescue

Norman P. Jouppi, et al. 2021. Ten lessons from three generations shaped Google's TPUv4i. ISCA. https://doi.org/10.1109/ISCA52012.2021.00010

PARALLELISM, LOCALITY, STRUCTURE AND PREDICTABILITY

Vector instructions are

Compact: single instruction defines N operations

Amortizes the cost of instruction fetch/decode/issue

Also reduces the frequency of branches

Parallel: N operations are (data) parallel

No dependencies

No need for complex hardware to detect parallelism (similar to VLIW)

Can execute in parallel assuming N parallel data paths

Expressive: memory operations describe patterns

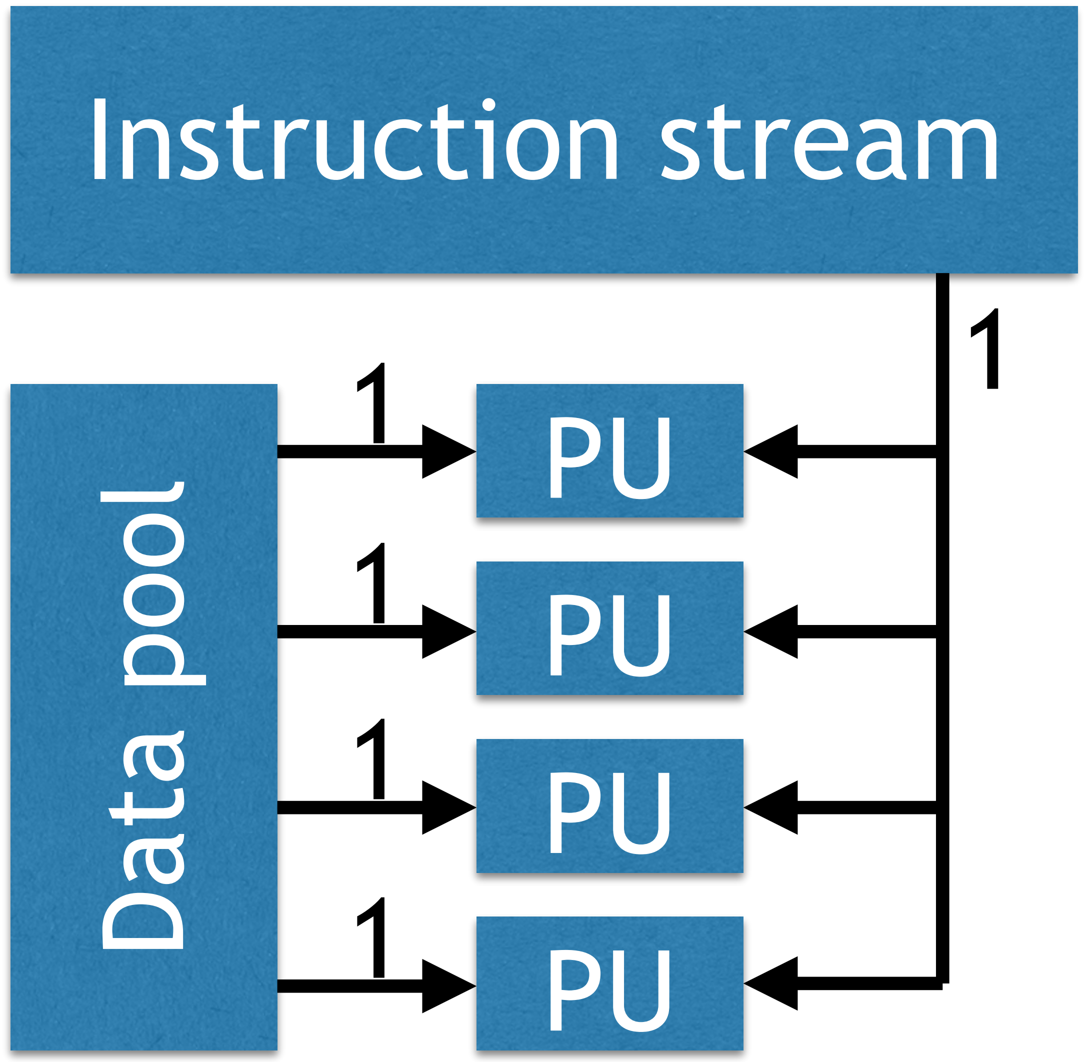
Continuous or regular memory access pattern

Can prefetch or accelerate using wide/multi-banked memory

Can amortize high latency for 1st element over large sequential pattern

1x 20-1300pJ (FETCH)

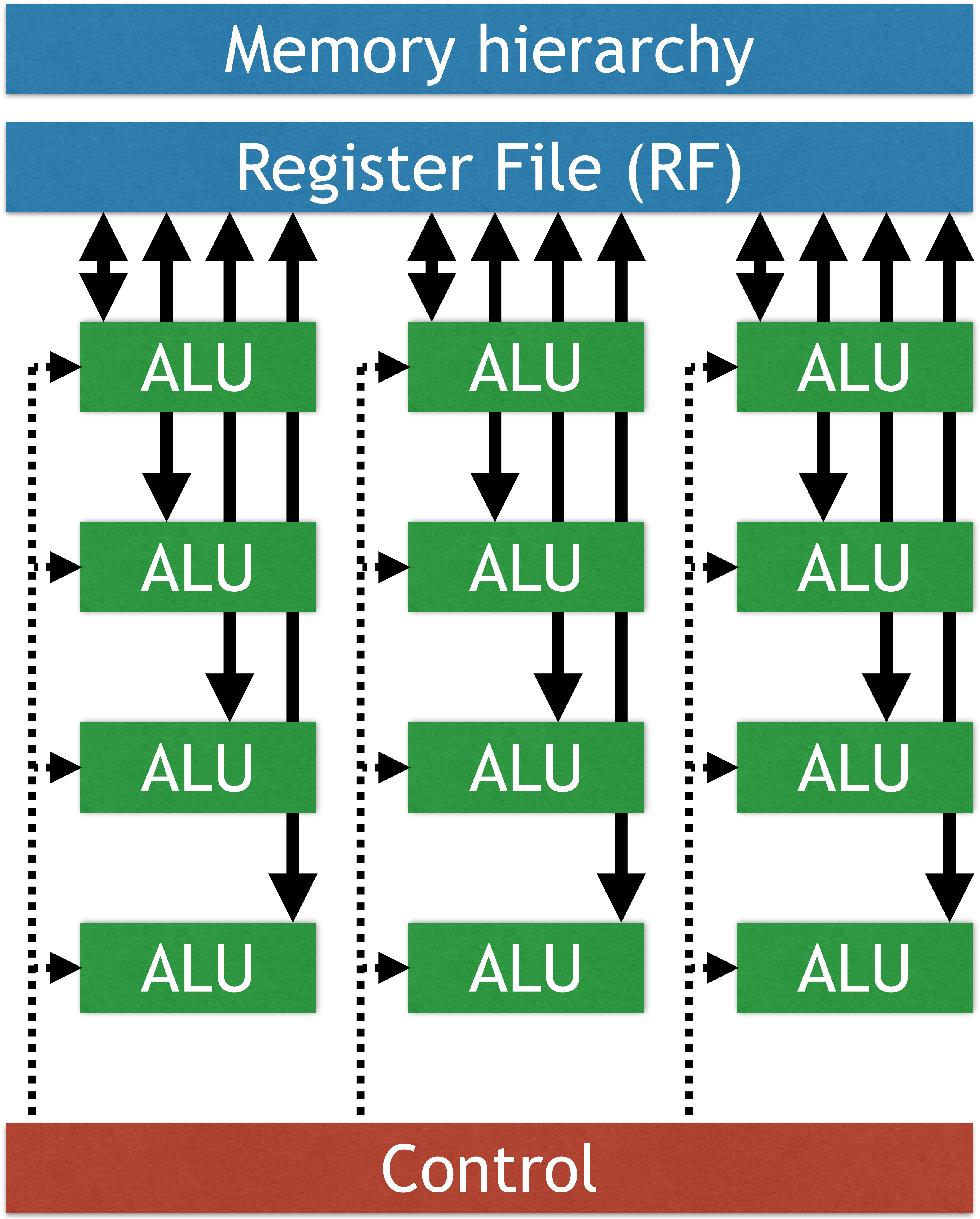
4x SIMD example



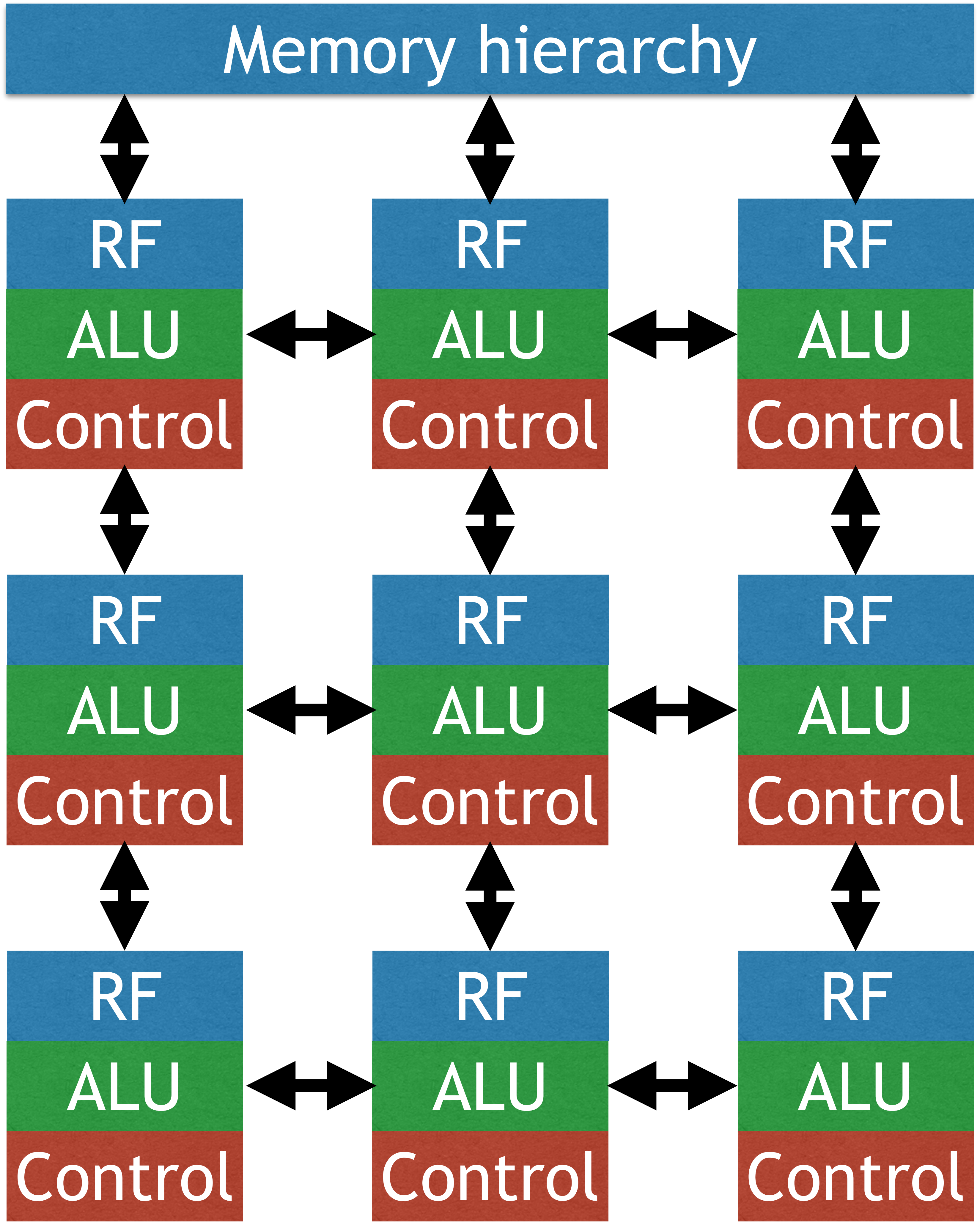
1x 20-1300pJ (CACHE MISS)

N-1x 20pJ (CACHE HIT)

PARALLELISM, LOCALITY, STRUCTURE AND PREDICTABILITY

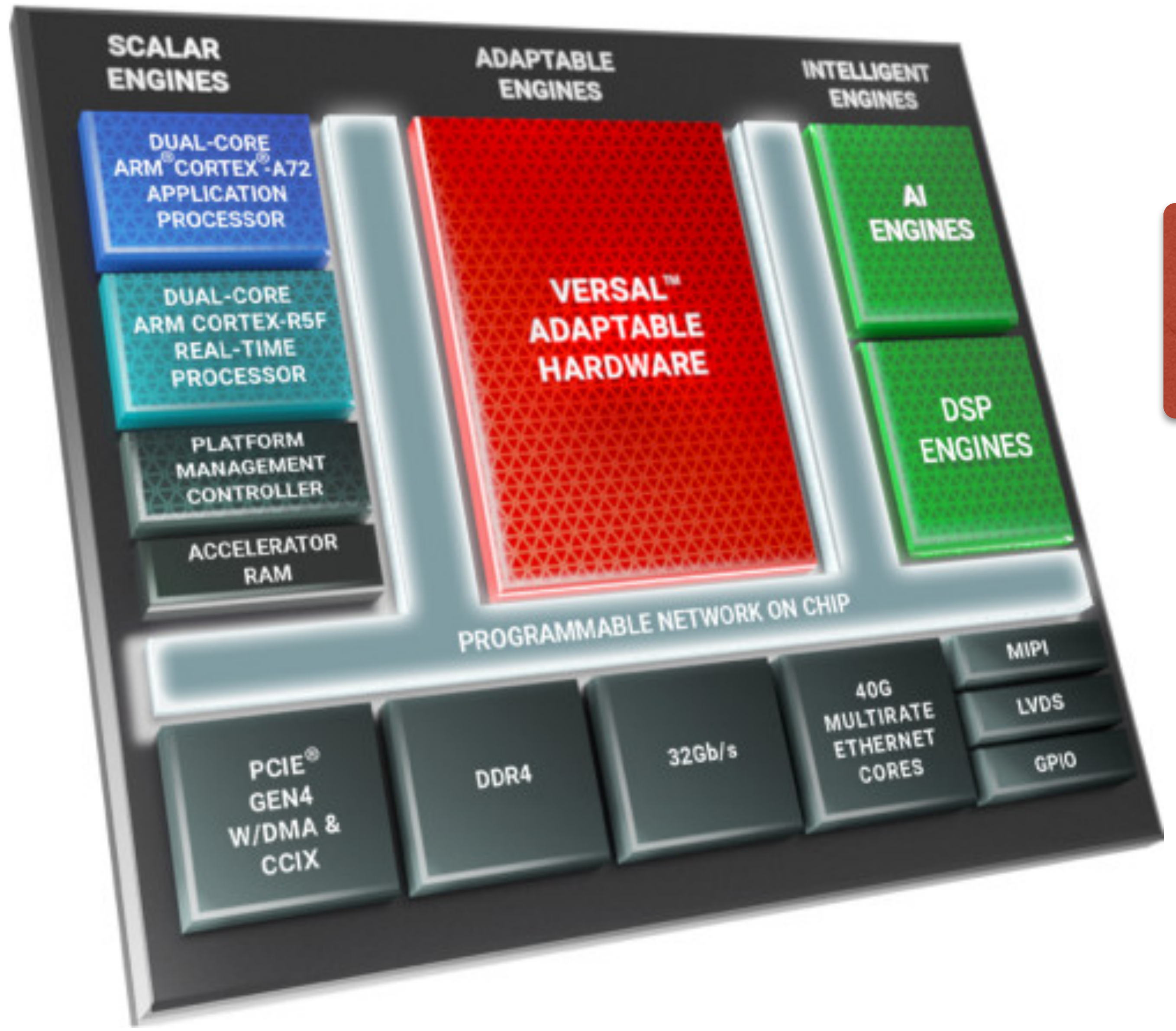


Temporal architecture:
BSP-based multi-core vector processor



Spatial architecture: Systolic array

ARRAY PROCESSOR EXAMPLES



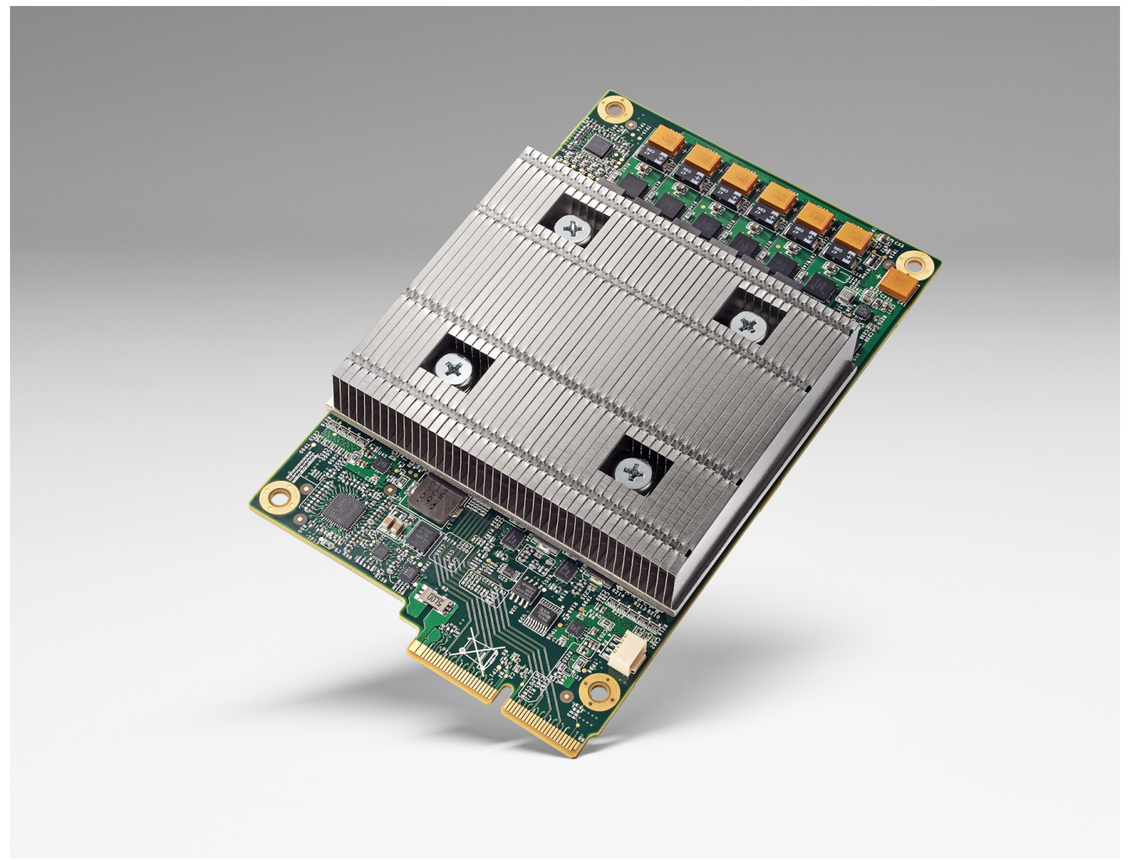
XILINX Versal AI

Kazem, Monday

GraphCore IPU

Sunway SW26010

NVIDIA TensorCore

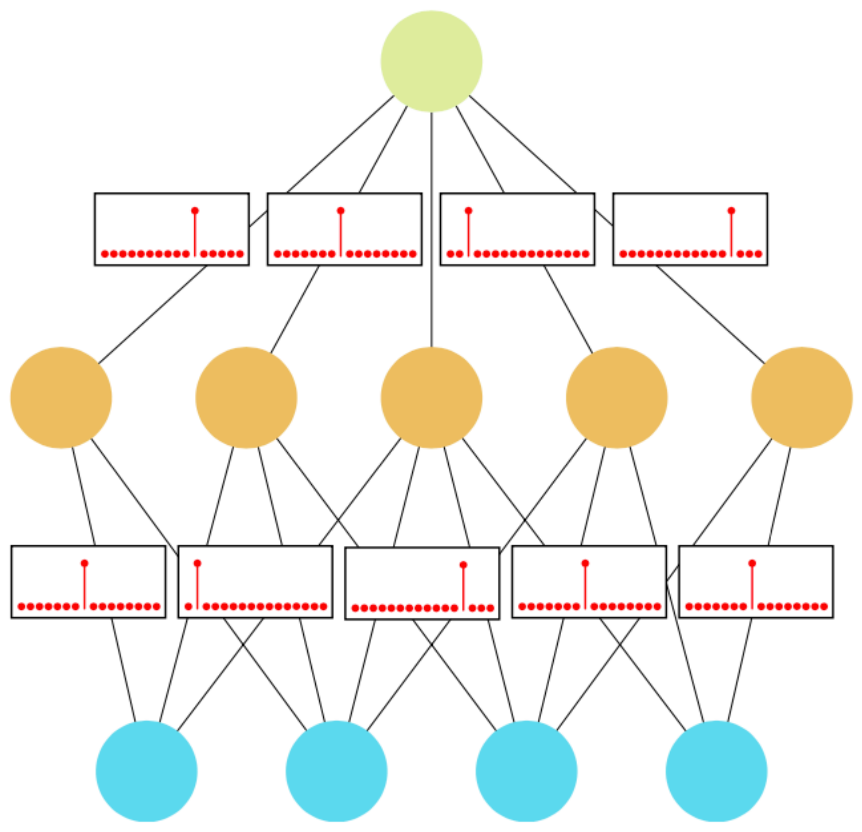


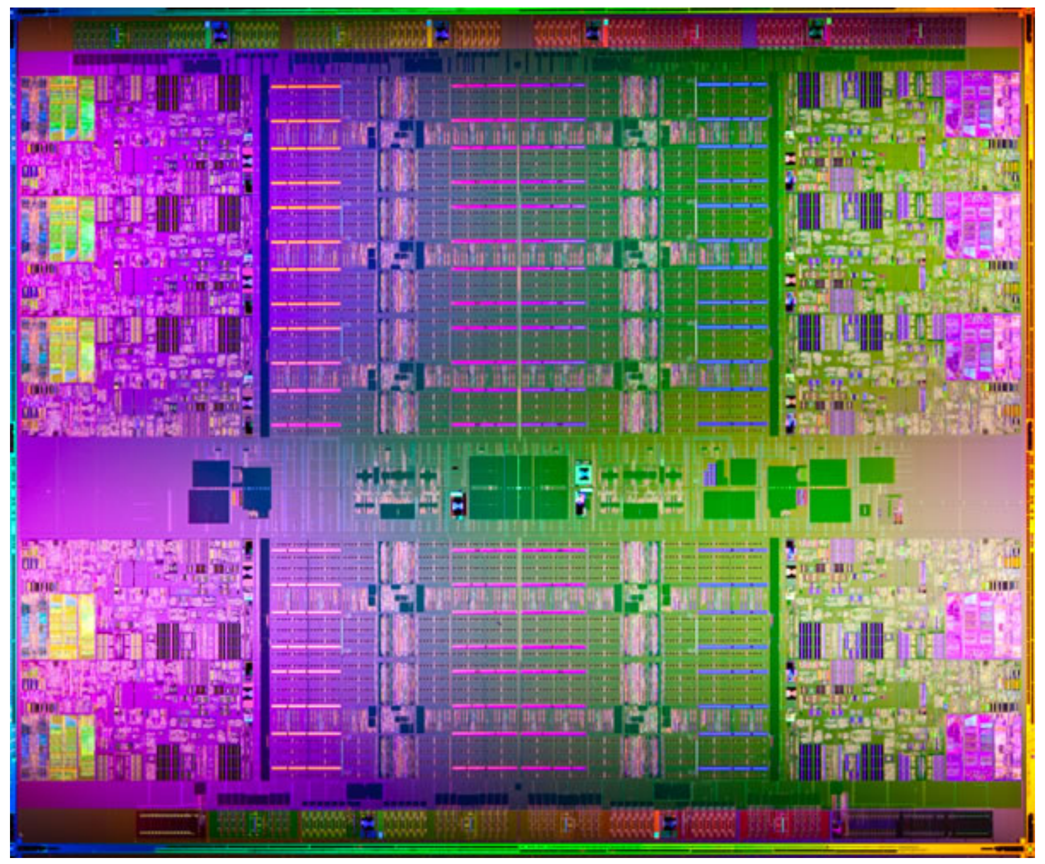
Google TPU

PLSP EXAMPLES

Need for …	Parallelism	Locality	Structure	Predictability
CPU	**Low** (core count)	**Medium** (caching)	**Medium** (v=512, cache block size)	**Low** (speculation, OOO, caching)
GPU	**Extreme** (CUDA core count)	**Medium** (shared mem)	**High** (v=1024 - warp concept), memory coalescing	**Low** (multi-threading)
FPGA/CGRA	**High** (array size)	**High** (blocking NOC)	Depends	**High** (spatial processing)
TPU	**High** (array size)	**Extreme** (neighbor only)	**Extreme** (v=512k - 256×256 array of 8-bit mult.)	**Extreme** (systolic array)

NEURAL ARCHITECTURES
&
PARALLELISM, LOCALITY, STRUCTURE AND PREDICTABILITY (**<u>PLSP</u>**)





DEEP NEURAL NETWORKS ARE VERY INLINE WITH PLSP - SWEET FREEDOM

Reduce-and-Scale [1] -> embedded CPUs - PL~~SP~~

Quantization

- Maximizing sparsity, tenary quantization
- Huffman coding and RLE for compact data structures => more cache hits

$$c = \sum_{i=1}^{N} w_i \cdot a_i, \quad w_i, a_i \in \mathbb{R}$$

$$w_i \in \{W_P, 0, W_N\}$$

$$c = W_l^p \cdot \sum_{i \in \mathbf{i}_l^p} a_i + W_l^n \cdot \sum_{i \in \mathbf{i}_l^n} a_i$$

[1] Günther Schindler, Matthias Zöhrer, Franz Pernkopf and Holger Fröning, Towards Efficient Forward Propagation on Resource-Constrained Systems, ECML 2018, https://doi.org/10.1007/978-3-030-10925-7_26

[2] Günther Schindler, Wolfgang Roth, Franz Pernkopf and Holger Fröning, Parameterized Structured Pruning for Deep Neural Networks, LOD 2020, http://arxiv.org/abs/1906.05180

[3] Torben Krieger, Bernhard Klein and Holger Fröning, Towards Hardware-Specific Automatic Compression of Neural Networks, PracticalDL Workshop @ AAAI 2023, https://arxiv.org/abs/2212.07818

DEEP NEURAL NETWORKS ARE VERY INLINE WITH PLSP - SWEET FREEDOM

Quantization

Reduce-and-Scale [1] -> embedded CPUs - PL~~SP~~

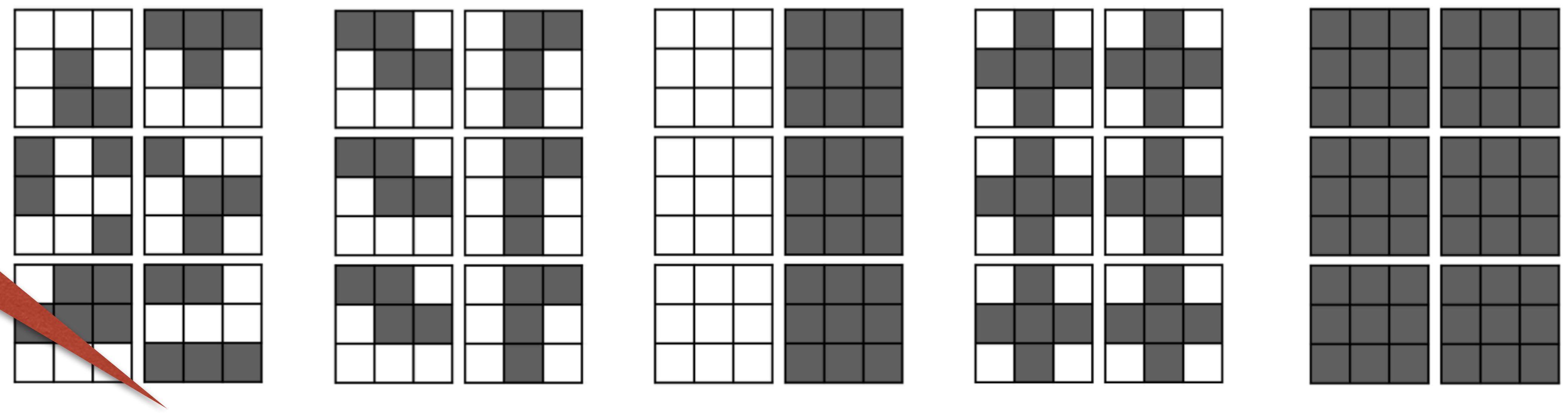
- Maximizing sparsity, tenary quantization
- Huffman coding and RLE for compact data structures => more cache hits

Parametrized Structured Pruning [2] -> GPUs - PLS~~P~~ / PLSP

Pruning

- Pruning towards block sparsity, with block size being inline with GPU architecture
- Thread warp size, memory coalescing, …

Unstructured



(a) Weights (b) Columns (c) Channels (d) Shapes (e) Layers

[1] Günther Schindler, Matthias Zöhrer, Franz Pernkopf and Holger Fröning, Towards Efficient Forward Propagation on Resource-Constrained Systems, ECML 2018, https://doi.org/10.1007/978-3-030-10925-7_26

[2] Günther Schindler, Wolfgang Roth, Franz Pernkopf and Holger Fröning, Parameterized Structured Pruning for Deep Neural Networks, LOD 2020, http://arxiv.org/abs/1906.05180

[3] Torben Krieger, Bernhard Klein and Holger Fröning, Towards Hardware-Specific Automatic Compression of Neural Networks, PracticalDL Workshop @ AAAI 2023, https://arxiv.org/abs/2212.07818

DEEP NEURAL NETWORKS ARE VERY INLINE WITH PLSP - SWEET FREEDOM

Reduce-and-Scale [1] -> embedded CPUs - PL~~SP~~

Quantization

- Maximizing sparsity, tenary quantization
- Huffman coding and RLE for compact data structures => more cache hits

Parametrized Structured Pruning [2] -> GPUs - PLS~~P~~ / PLSP

Pruning

- Pruning towards block sparsity, with block size being inline with GPU architecture
- Thread warp size, memory coalescing, …

Galen (NAS) [3] -> generalization, but up to now only on ARM CPUs

- Combining fine-grained quantization with channel pruning
- Layer-dependent decisions
- Latency test on real HW targets for reinforcement learning

Bernhard, Saturday

Quant./ Prune

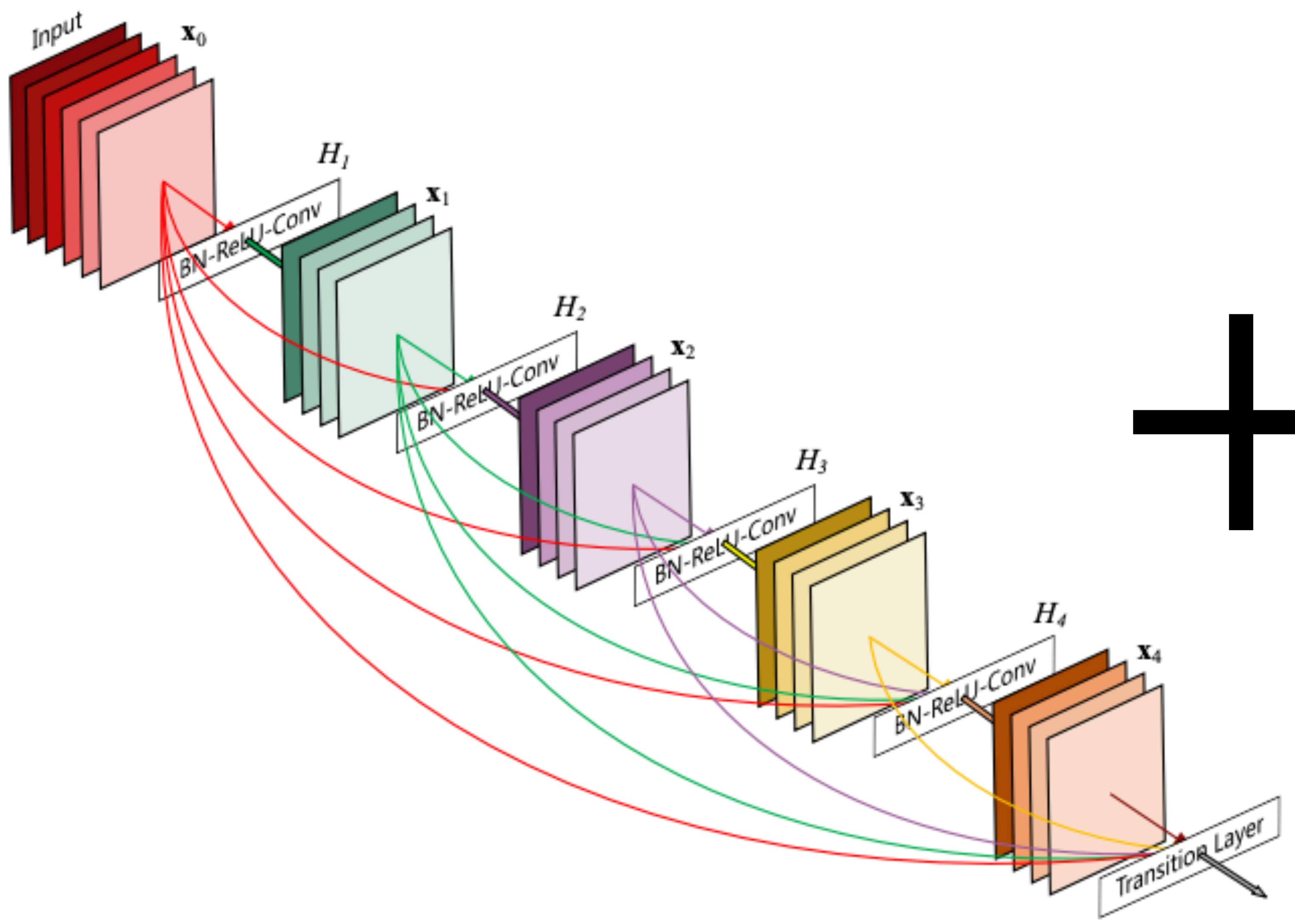
[1] Günther Schindler, Matthias Zöhrer, Franz Pernkopf and Holger Fröning, Towards Efficient Forward Propagation on Resource-Constrained Systems, ECML 2018, https://doi.org/10.1007/978-3-030-10925-7_26

[2] Günther Schindler, Wolfgang Roth, Franz Pernkopf and Holger Fröning, Parameterized Structured Pruning for Deep Neural Networks, LOD 2020, http://arxiv.org/abs/1906.05180

[3] Torben Krieger, Bernhard Klein and Holger Fröning, Towards Hardware-Specific Automatic Compression of Neural Networks, PracticalDL Workshop @ AAAI 2023, https://arxiv.org/abs/2212.07818

REASONING ABOUT UNCERTAINTY?

SOTA NN arch



+

Image

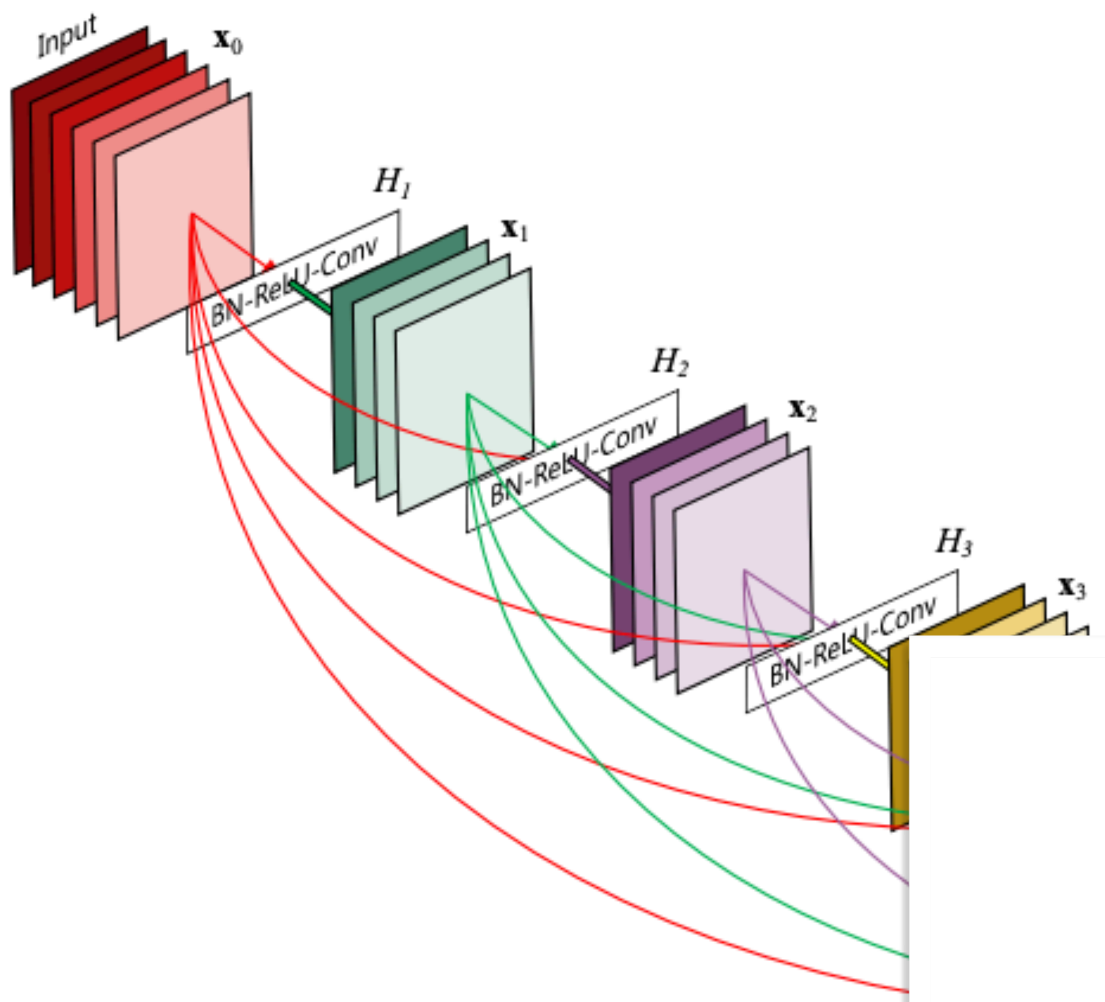
=

Top-10 Classification
1: Persian cat (65.3%)
2: tabby (11.9%)
3: lynx (11.6%)
4: tiger cat (7.6%)
5: Egyptian cat (1.8%)
6: computer keyboard (0.2%)
7: lion (0.1%)
8: carton (0.1%)
9: plastic bag (0.1%)
10: washer (0.1%)

Based on pretrained ResNet18/torchvision

REASONING ABOUT UNCERTAINTY? 👀

SOTA NN arch

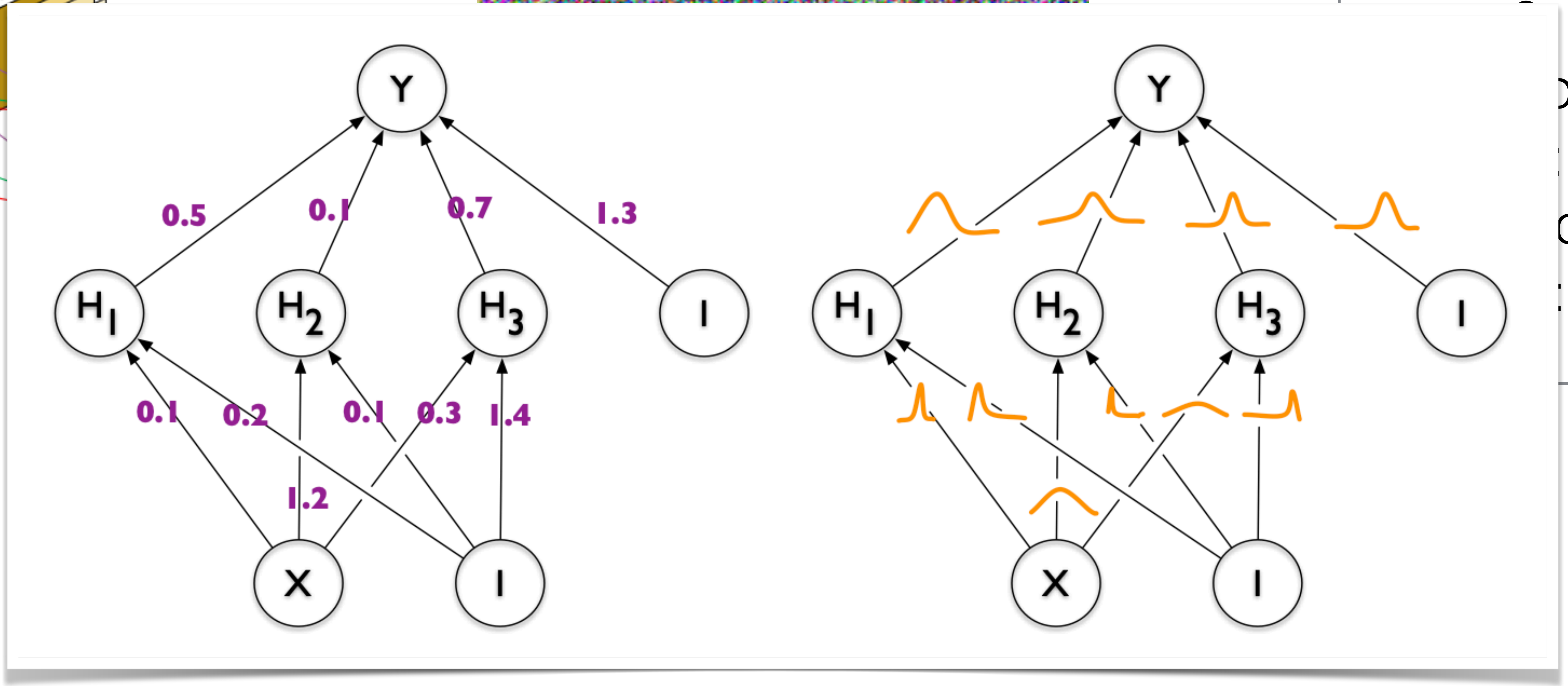


+

One noisy boi

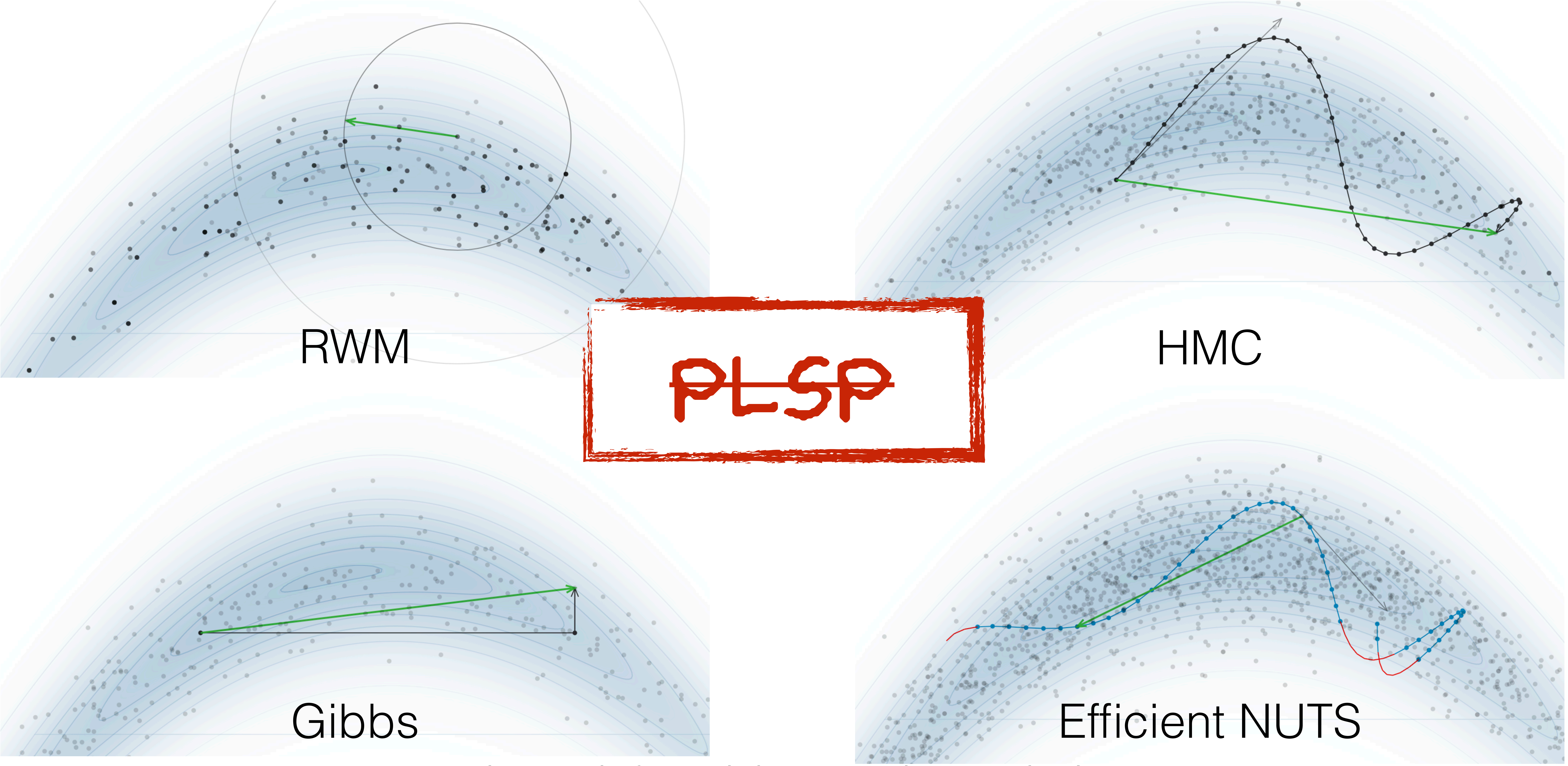
=

Top-10 Classification
1: jellyfish (13.1%)
2: hammerhead (3.7%)
3: jigsaw puzzle (3.5%)
4: electric ray (2.6%)
5: sea snake (2.4%)
stingray (2.3%)
rayer rug (2.0%)
starfish (2.0%)
coral reef (1.5%)
doormat (1.4%)



Based on pretrained ResNet18/torchvision

SAMPLING GALLERY OF 2D PROBABILITY DISTRIBUTION (BANANA)



https://chi-feng.github.io/mcmc-demo/app.html

"If you have a vector problem, build a vector processor"

–Jim Smith/Wisconsin

"If you have a dataflow problem (DNN), build a dataflow processor"

–Kunle Olukotun/Stanford (Keynote ISCA 2023)

So should we build a Bayesian Machine?

$$\mu_y, \sigma_y := \sum_N \Phi(\mathbf{W} \oplus \mathbf{x}), \mathbf{W} \sim \mathscr{P}_W$$

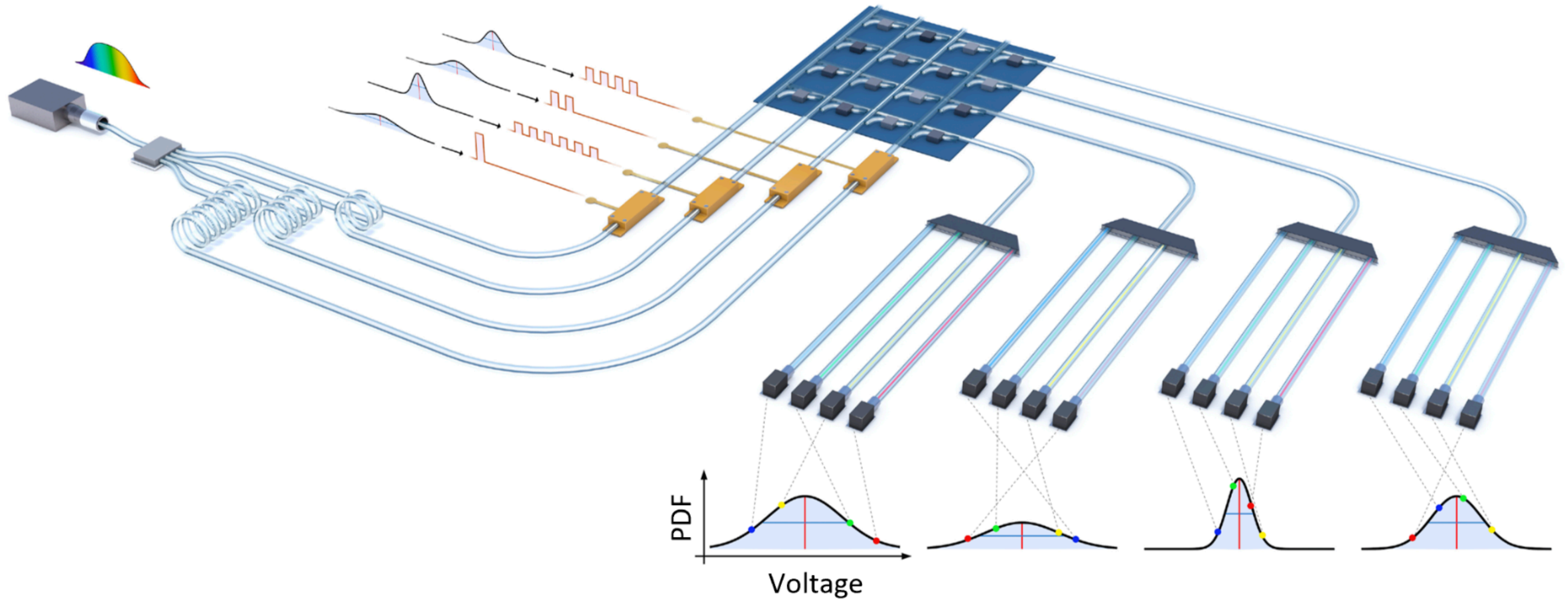
$$\mu_y, \sigma_y := \sum_N \Phi(\mathbf{W} \oplus \mathbf{x}) + \mathbf{v}, \mathbf{v} \sim \mathcal{N}(0, \sigma_v)$$

$$\mu_y, \sigma_y := \sum_N \Phi(\mathbf{W} \oplus \mathbf{x}), \mathbf{W} \sim \mathcal{N}(\mu_w, \sigma_w)$$

BAYESIAN MACHINES
(COLLAB. WITH WOLFRAM PERNICE/HEIDELBERG UNIV.)

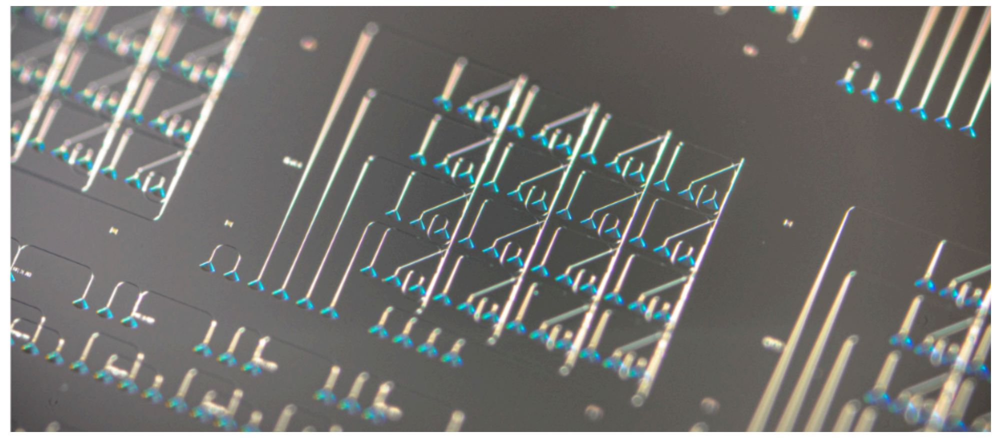
Analog processors are promising in energy efficiency, but inherently come with noise

- Let’s use noise as a source of randomness
- Caveat: we need some control over the noise 🤕

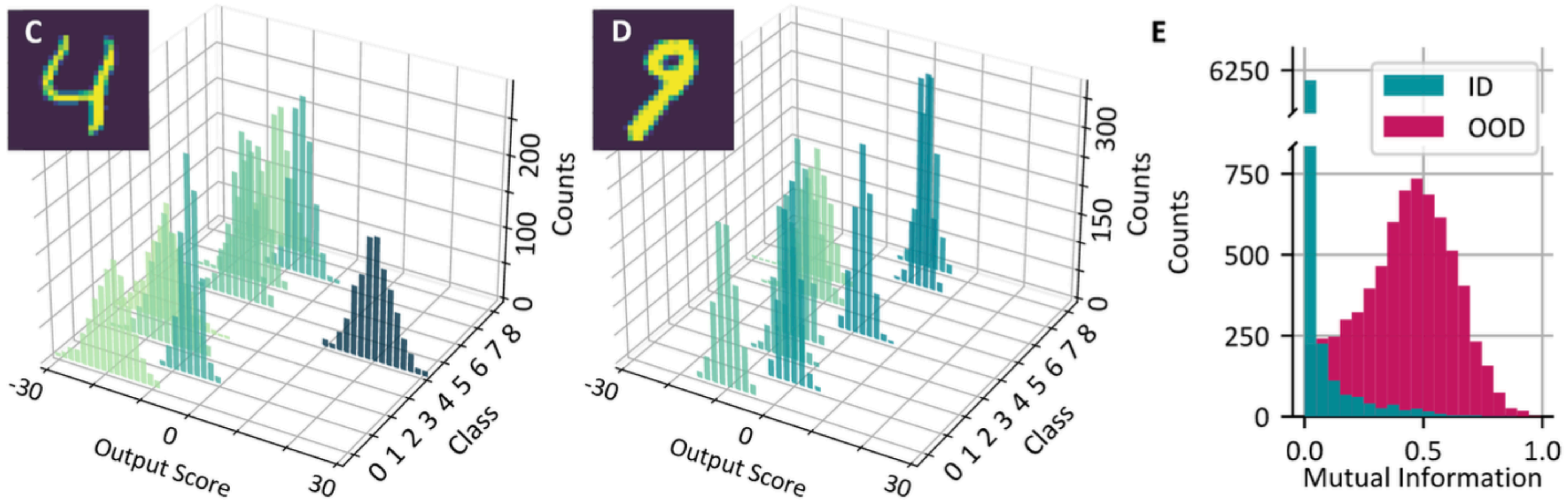


Photonic probabilistic processor

- Chaotic light source
- Coding as noise control
- DNN model can now say “I don’t know” 😍



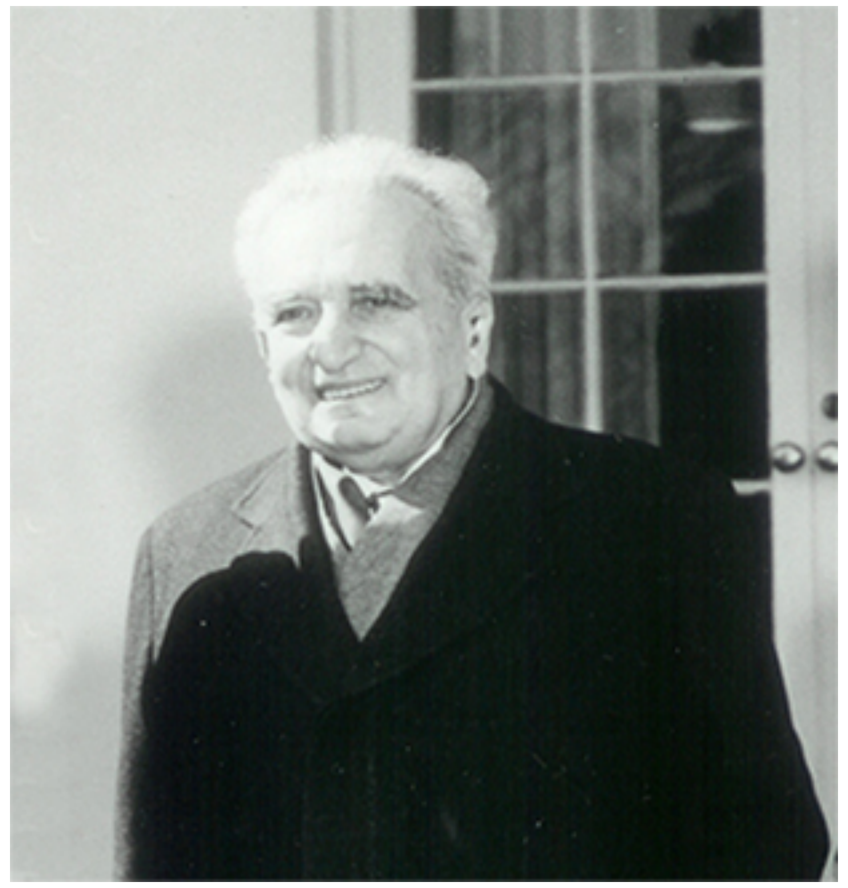
9-class MNIST: do not show 9 during training, but test for it



Hendrik, Monday

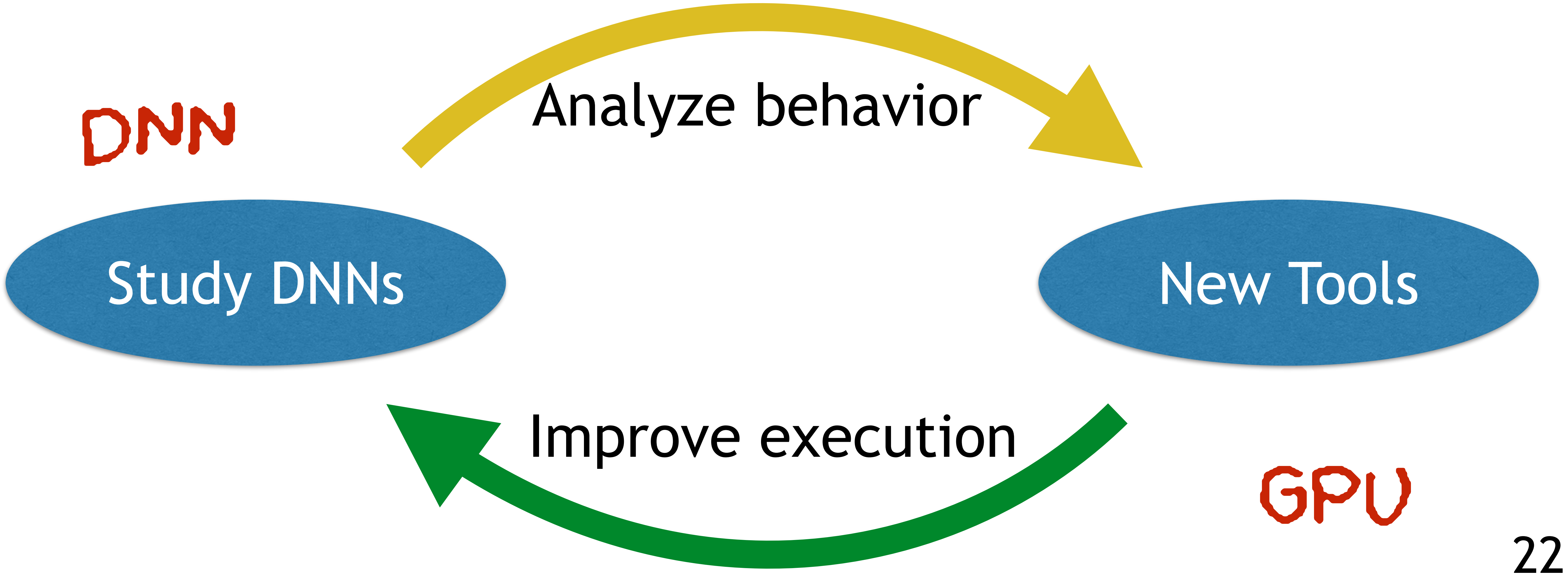
WRAPPING UP

Founding event Faculty of Engineering Sciences, 2022

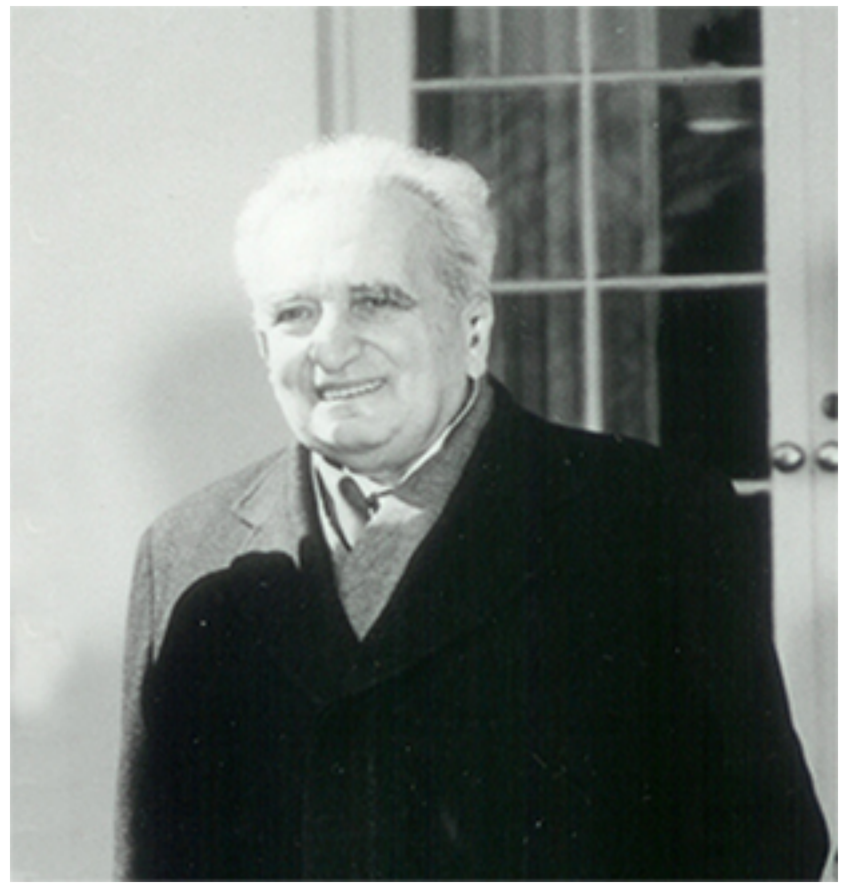


“Scientists study the world as it is, engineers create the world that never has been.”

–Theodore von Kármán (1881-1963)

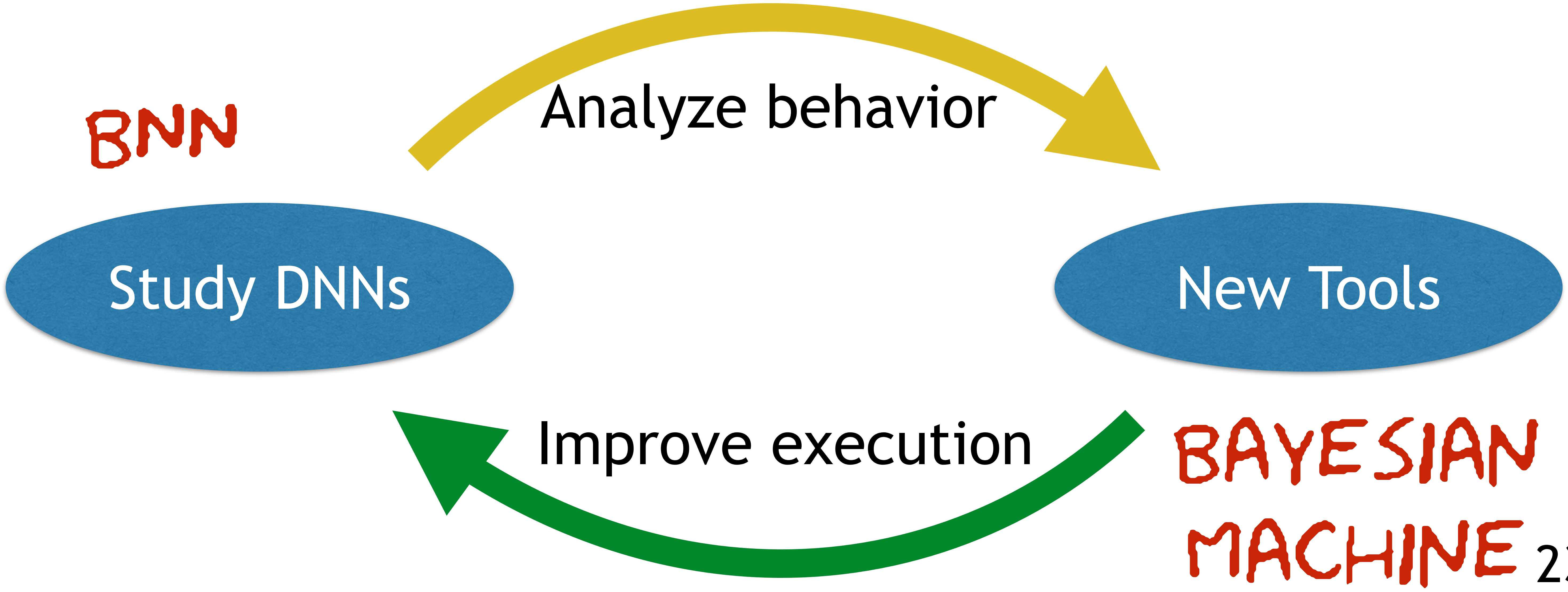


Founding event Faculty of Engineering Sciences, 2022



“Scientists study the world as it is, engineers create the world that never has been.”

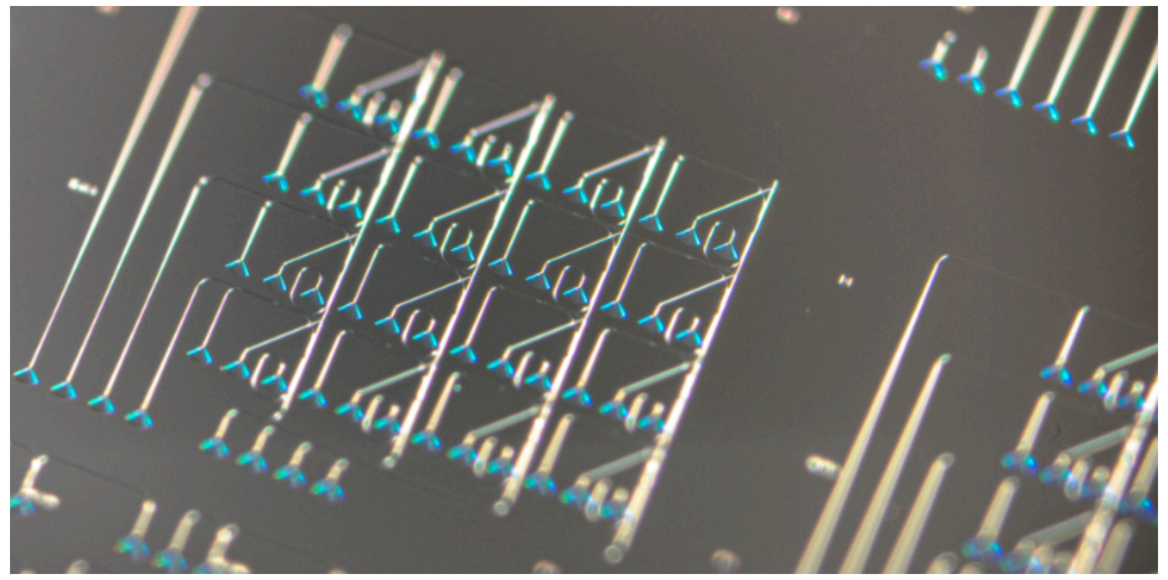
–Theodore von Kármán (1881-1963)



WRAPPING UP

CMOS is stuttering, but future scaling demands for **<u>parallelism, locality, structure and predictability (PLSP)</u>**

- Due to different economic settings still alive -> cloud, hyperscalers
- DNNs very inline with PLSP, variants such as BNNs not



pJ as interface in between architecture and device technology

- Simple, easy to reason about, abstract

Bayesian Machines can be promising to leverage inherent noise in analog computing as a benefit

- Caveat: control over noise required

Sara Hooker. 2021. The hardware lottery. Commun. ACM 64, 12 (December 2021). https://doi.org/10.1145/3467017