SUSTAINED PERFORMANCE SCALING IN THE LIGHT OF BOUNDED GROWTH AND TECHNOLOGY CONSTRAINTS

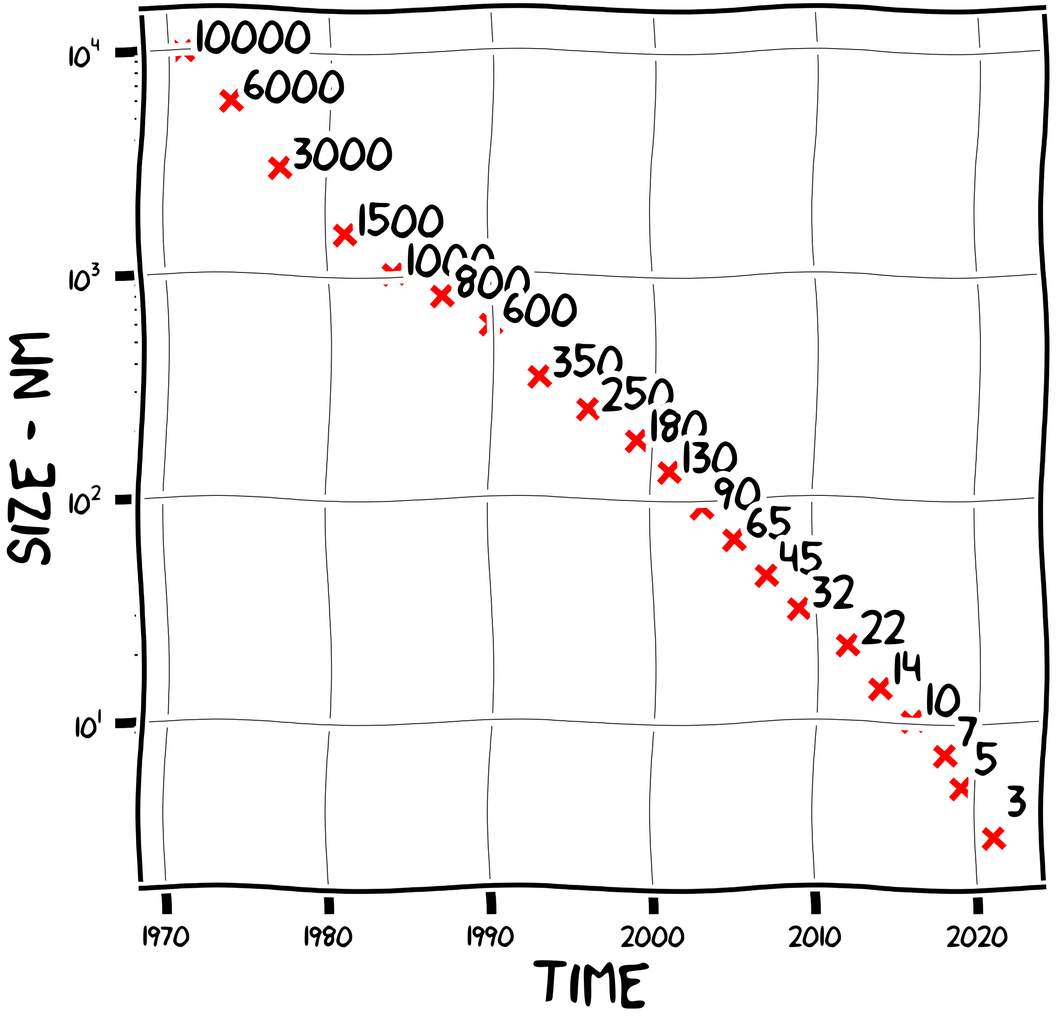
Holger Fröning
Computing Systems Group, Institute of Computer Engineering
Heidelberg University, Germany

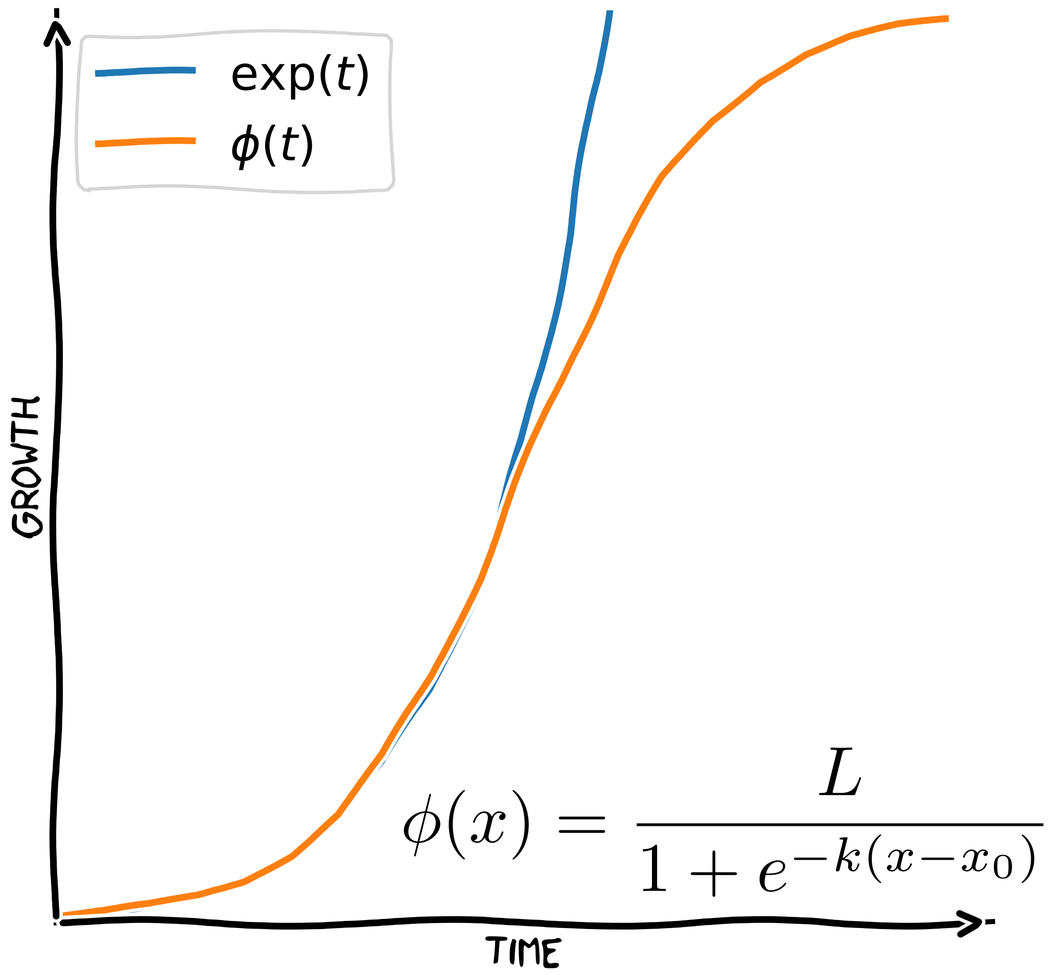
InvasIC Seminar, Friedrich-Alexander-Universität Erlangen-Nürnberg, Dec 16, 2020

ABOUT US

- Professor at Heidelberg University, Germany (since 2019)
  - Currently advising 9 PhDs, 4 being external (3 on ML)
  - PhD from Mannheim University (Interconnection Networks), advised by Ulrich Brüning (2007)
  - Post-Doc, TU Valencia & Heidelberg University (FPGAs, ASICs), advised by José Duato (2008-2011)
  - Visiting professor, TU Graz, sponsored by Gernot Kubin (2015 - first contact with ML)
  - Visiting scientist, NVIDIA Research, sponsored by Bill Dally, reporting to Larry Dennison (Santa Clara, CA, 2016)
- Performance, energy efficiency & programmability for future & emerging technologies
  - Computer architects with a focus on low-level software layers
  - High-performance computing, machine learning & data analytics
  - Bipolar computing landscape: super small & super big

EXPONENTIAL GROWTH IS AN ILLUSION





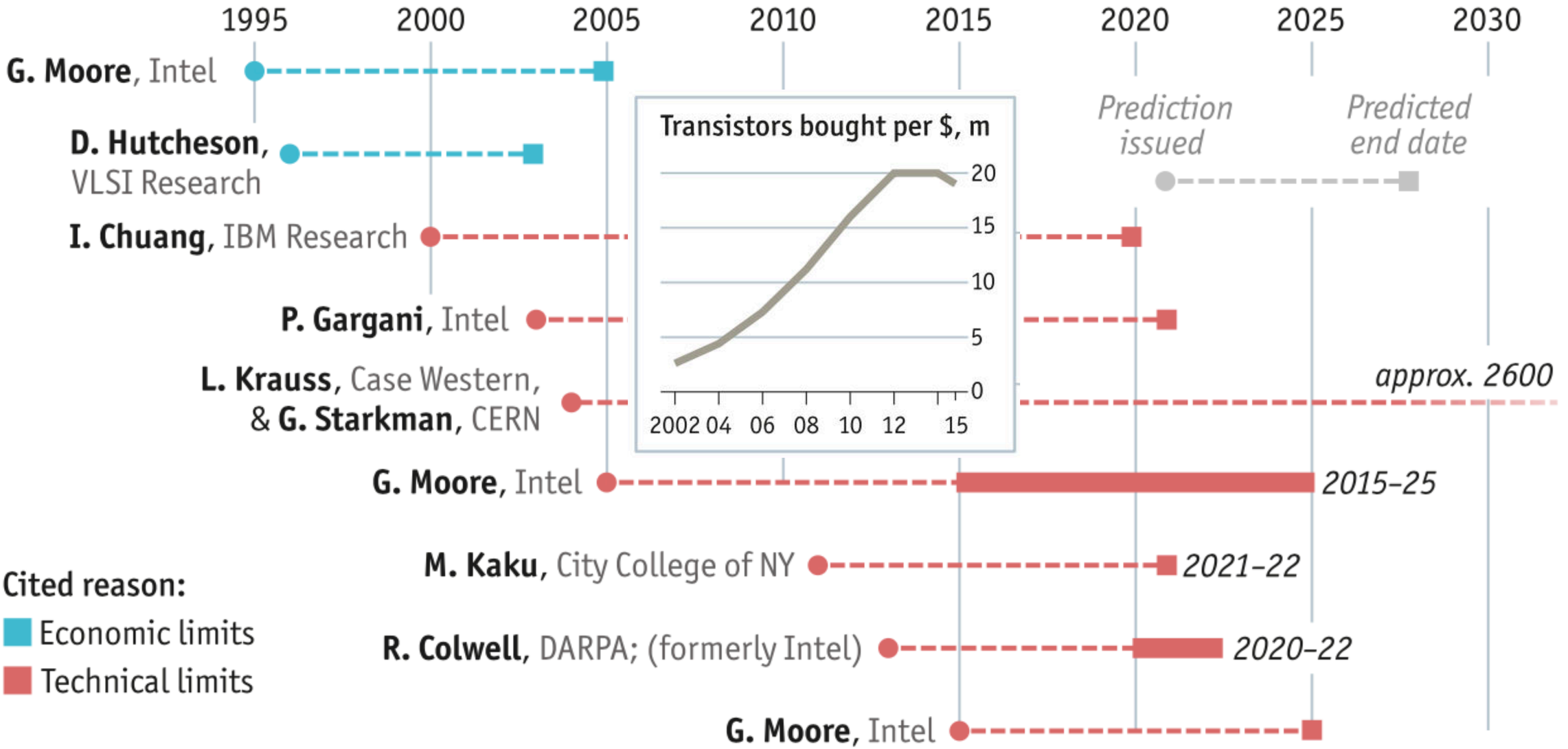
*S. K. Moore, "**The node is nonsense**," in IEEE Spectrum, vol. 57, no. 8, pp. 24-30, Aug. 2020, doi: 10.1109/MSPEC.2020.9150552.*

-> LMC: density of logic (DL), of main memory (DM), and interconnect (DC).

https://www.economist.com/technology-quarterly/2016-03-12/after-moores-law

Faith no Moore

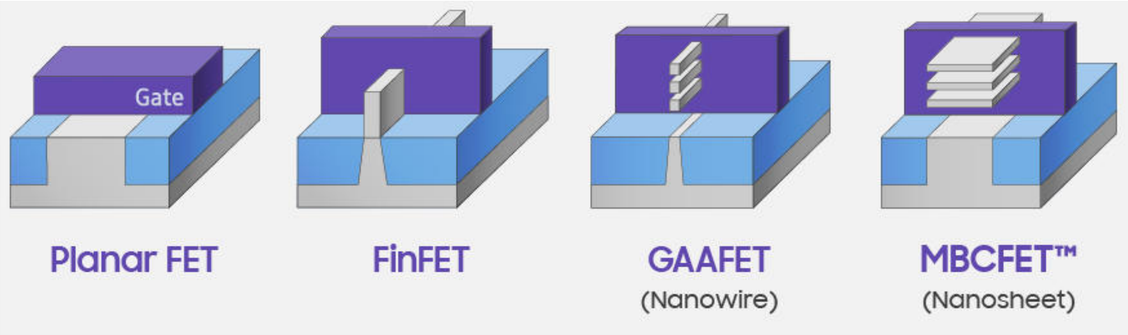
Selected predictions for the end of Moore's law



Sources: Intel; press reports; *The Economist*

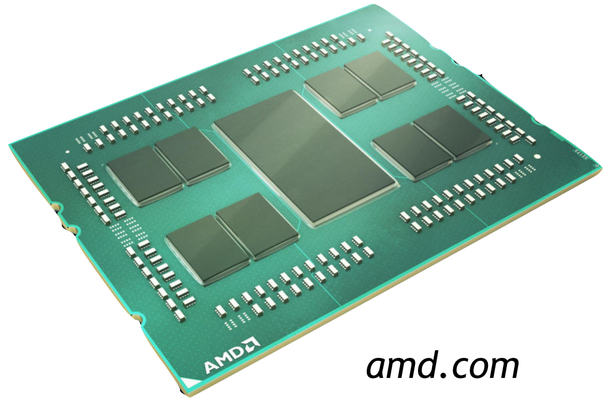
MOORE'S LAW AS OF TODAY

- Transistors still getting smaller
  - Albeit end in sight (probably 3nm)
  - Careful: prototyping != mass production
  - New devices: TFET [1], FEFET [2][3] (Thomas Theis, 2017)



samsung.com

- Chips get larger
  - Size limited by reticle (some NVIDIA GPUs are already at the max)
- Chiplets (lego-like silicon bricks)
  - See Multi-Chip Modules (MCMs) from the 80s/90s
  - 2nd gen AMD Epyc: 8 chiplets



amd.com

[1] E. Memisevic, et al., "Vertical InAs/GaAsSb/GaSb tunneling field-effect transistor on Si with S = 48 mV/decade and Ion = 10 μA/μm for Ioff = 1 nA/μm at Vds = 0.3 V," 2016 IEEE International Electron Devices Meeting (IEDM)

[2] S. Salahuddin and and Supriyo Datta, Use of Negative Capacitance to Provide Voltage Amplification for Low Power Nanoscale Devices, Nano Letters 2008 8 (2), 405-410

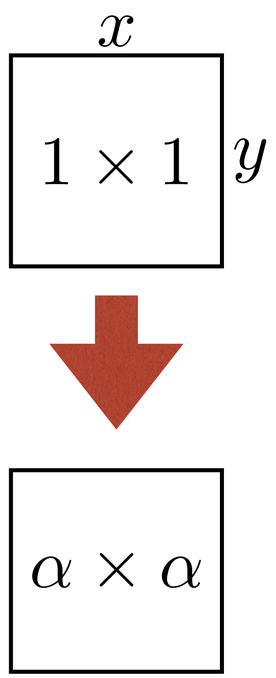
[3] Z. Krivokapic, et al., 14nm Ferroelectric FinFET Technology with Steep Subthreshold Slope for Ultra Low Power Applications, 2017 IEEE International Electron Devices Meeting (IEDM)

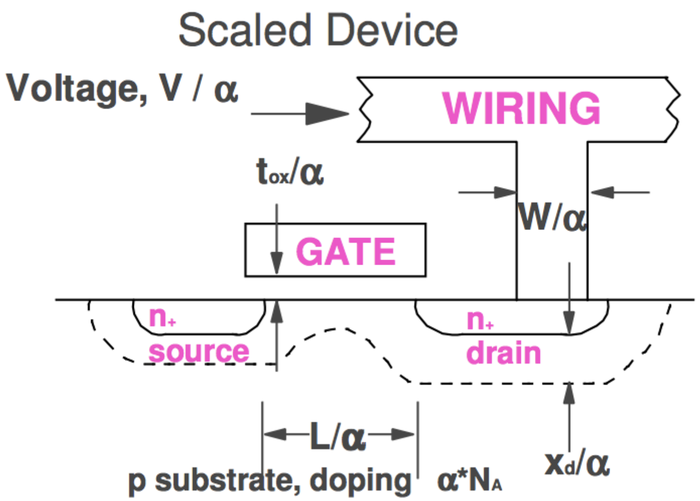
DENNARD SCALING

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

Results of classic scaling (scaling factor: α)

- Higher density (α^2)
- Lower voltage, capacity (each 1/α)
- Higher speed (α)
- Active power density (1)

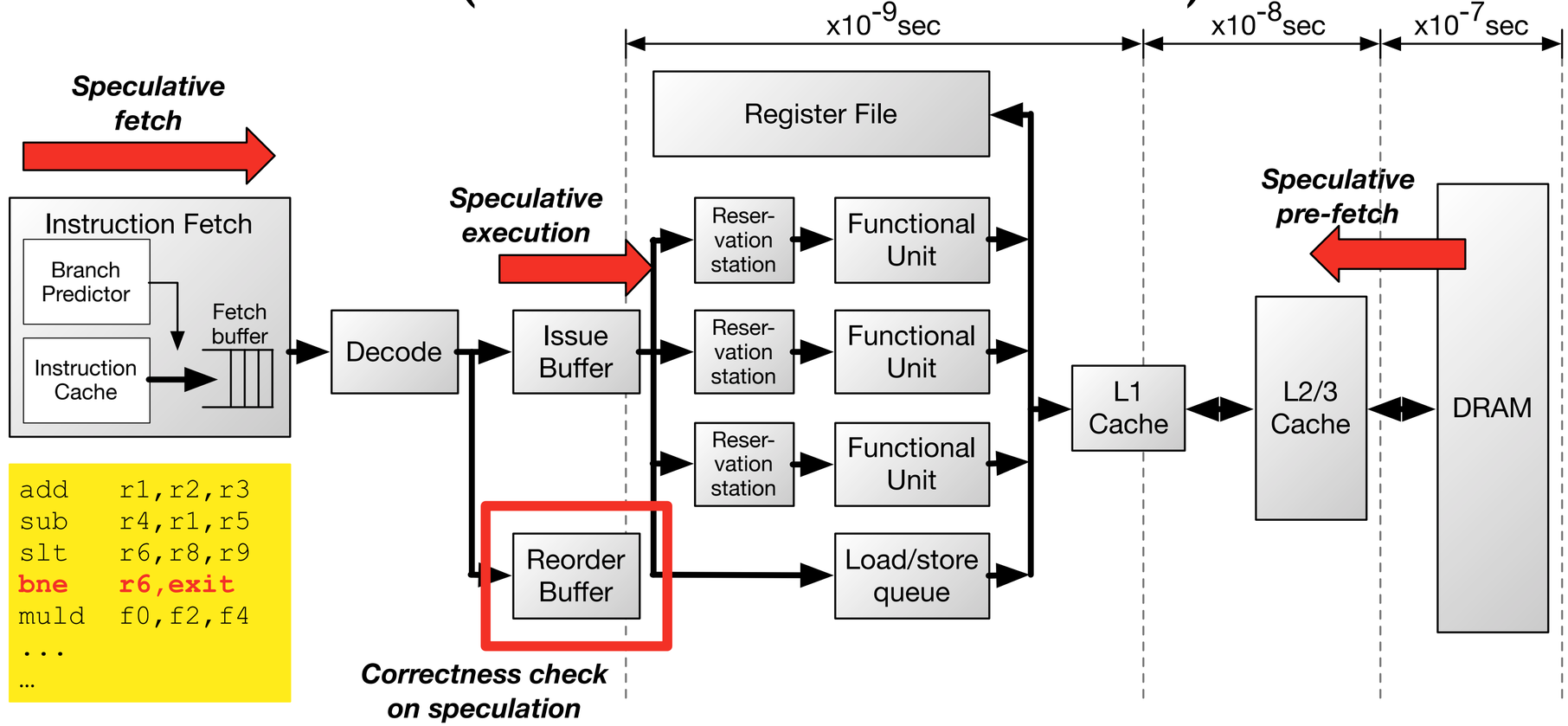




Classic Dennard
Oxide: t_{ox} /α
Wire width: W/α
Gate width: L/α
Diffusion: x_d/α
Substrate: α * NA
Voltage: V/α
Current: I/α

Pradip Bose, IBM Research, 2014

MICROARCHITECTURE EXAMPLE DRIVEN BY MOORE (& DENNARD SCALING)



More transistors
Frequency scaling
Locality (spatial/temporal)
Predictable control flow

Multiple, deep pipelines
Latency minimization & hiding
Speculation everywhere

Partly by Sudha Yalamanchili, Architectural Alternatives for Energy Efficient Performance Scaling, VLSI Conference, 2013

DENNARD SCALING

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

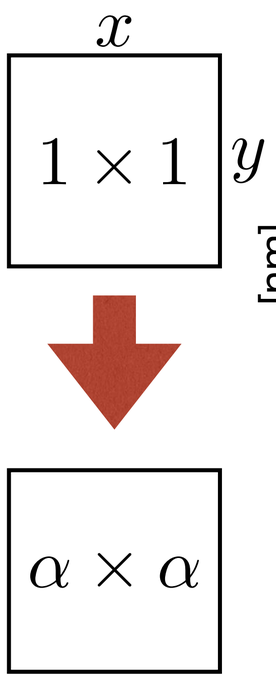
Results of classic scaling (scaling factor: α)

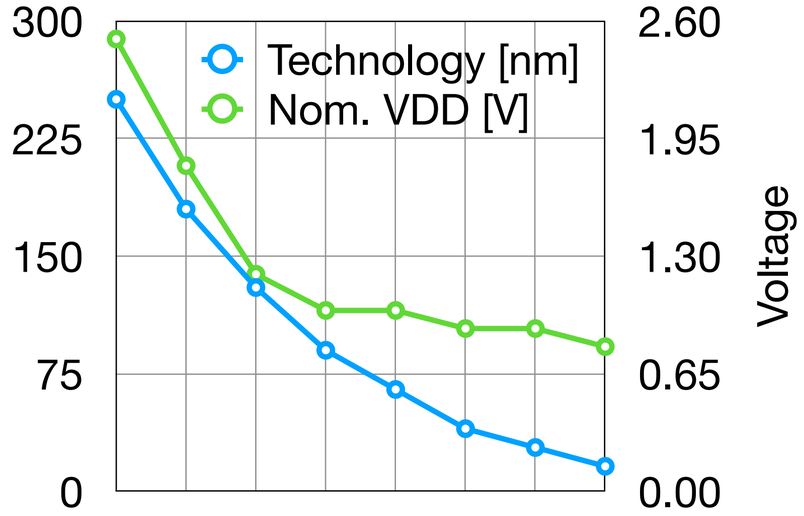
- Higher density (α^2)
- Lower voltage, capacity (each 1/α)
- Higher speed (α)
- Active power density (1)

Reality today: Post-Dennard

- Wiring & leakage power became a major issue
- Voltage scaling no longer possible
  - $V > V_{TH}$; $V_{TH} > 0.25\text{-}0.3V$
- Power no longer remains constant!

Power budget limits device count (Moore vs. Dennard)





Classic Dennard
Oxide: t_{ox} /α
Wire width: W/α
Gate width: L/α
Diffusion: x_d/α
Substrate: α * NA
~~Voltage: V/α~~
Current: I/α

Pradip Bose, IBM Research, 2014

PERFORMANCE SCALING

$$Perf(\frac{ops}{s}) = \underbrace{\frac{Instructions}{cycle}}_{\propto PipelineCount \cdot PipelineDepth} \cdot \underbrace{frequency}_{\text{scales with feature size}}$$

CLASSICAL DENNARD SCALING

$$Perf(\frac{ops}{s}) = \underbrace{Power(W)}_{\text{fixed}} \cdot Efficiency(\frac{ops}{Joule})$$

POST DENNARD SCALING

REGIME I

operator cost —> *Specialization* —> *heterogeneity and asymmetry*

GPU COMPUTING

PROGRAMMING COMPLEXITY

NUMA EFFECTS & LATENCY HIDING

\+ *data movement cost*

REGIME II

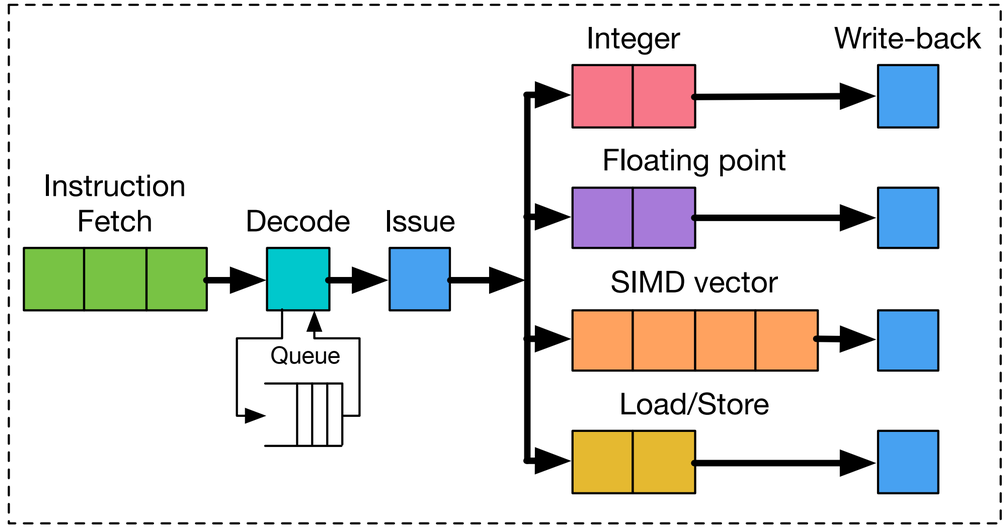
Partly by Bill Dally, Sudha Yalamanchili (UCAA Workshop, 2012)

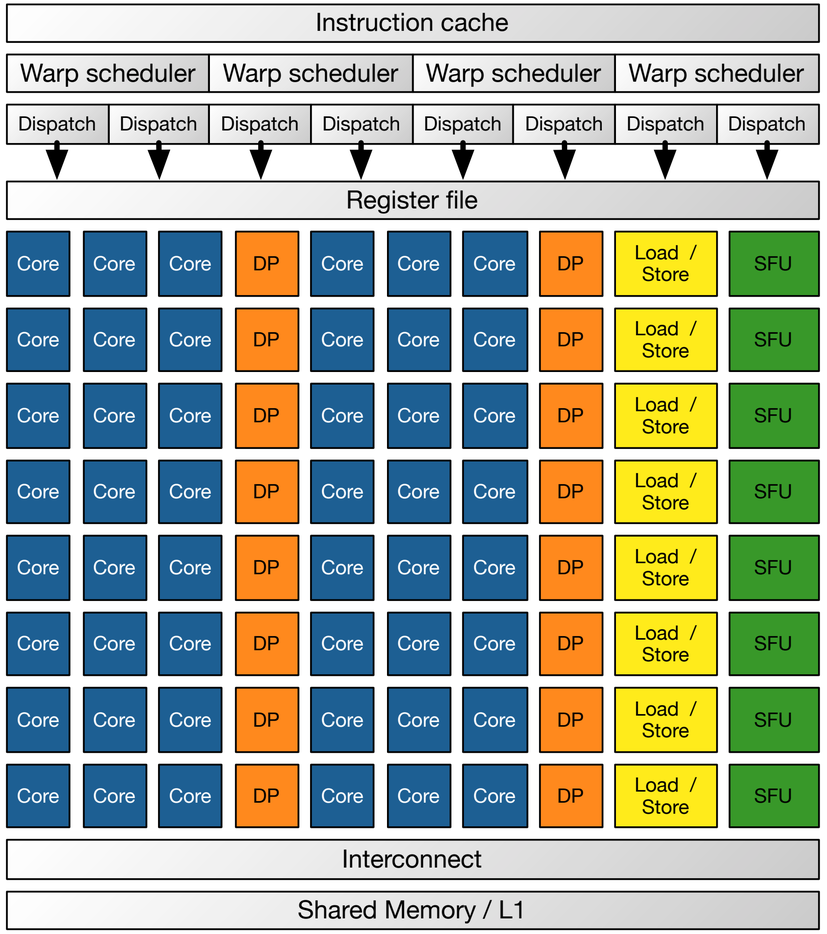
REGIME I - COMPUTE SPECIALIZATION

POST-DENNARD: TRANSITION TO MASSIVELY PARALLEL MICROARCHITECTURES

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

$$\propto f$$





Frequency reduction
In-order pipelines

Replication

Massively parallel
Energy efficient

REMINDER: BULK-SYNCHRONOUS PARALLEL

In 1990, Valiant already described GPU computing pretty well

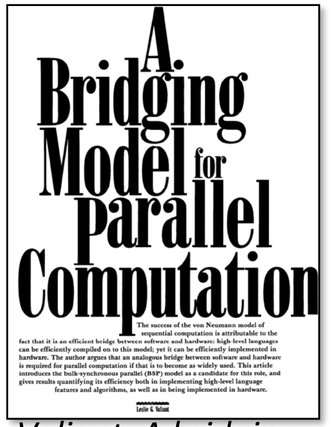
Superstep

- Compute, communicate, synchronize

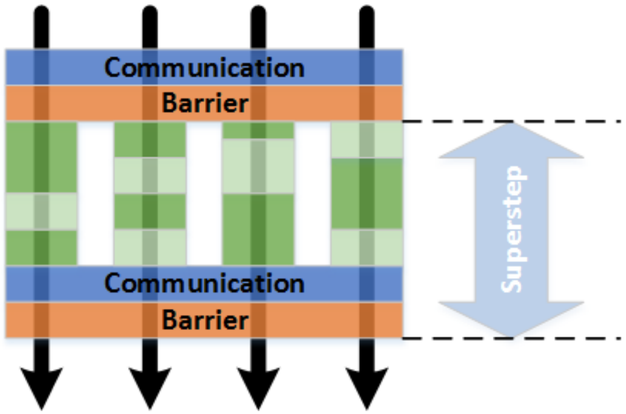
Parallel slackness: # of virtual processors v, physical processors p

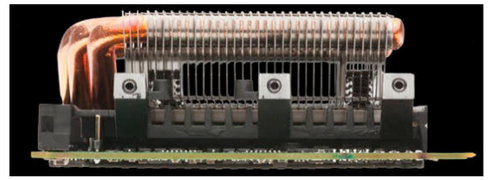
- v = 1: not viable
- v = p: unpromising wrt optimality
- v >> p: leverage slack to schedule and pipeline computation and communication efficiently

Extremely scalable, bad for unstructured parallelism



Leslie G. Valiant, A bridging model for parallel computation, Communications of the ACM, Volume 33 Issue 8, Aug. 1990





OUR VIEW OF A GPU

Software view: a programmable many-core scalar architecture

- Huge amount of scalar threads to exploit parallel slackness, operates in lock-step
- SIMT: single instruction, multiple threads

IT'S A (ALMOST) PERFECT INCARNATION OF THE BSP MODEL

Hardware view: a programmable multi-core vector architecture

- SIMD: single instruction, multiple data
- Illusion of scalar threads: hardware packs them into compound units

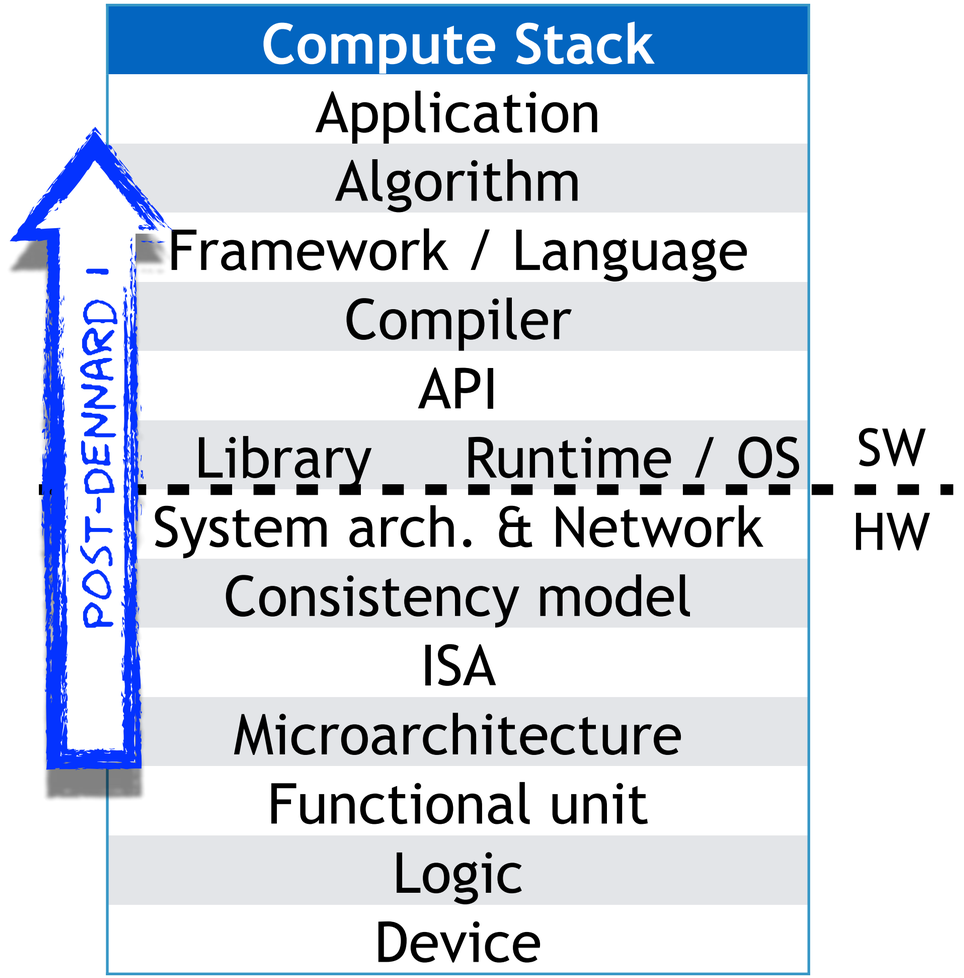
IT'S A VECTOR ARCHITECTURE THAT HIDES ITS VECTOR UNITS

COMPUTE STACK

Post-Dennard I (specialization): today

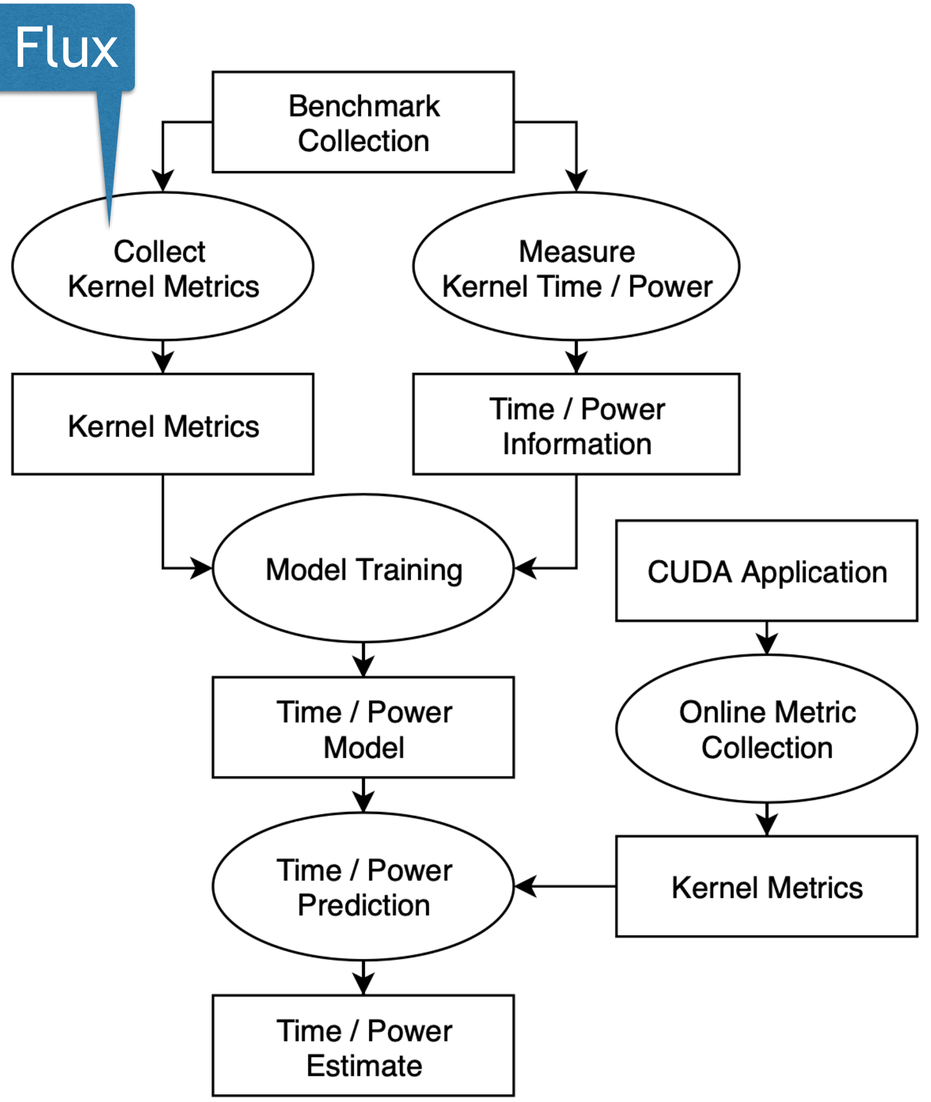
Some uarchs tend to actually converge, but programming models diverge => **replicated compute stacks**

Massive parallelization requires **structure** for efficient execution



HETEROGENEITY AND PORTABILITY

- Predictions about execution time and power consumption
  - Runtime/scheduling decisions
  - Provisioning decisions
  - Performance portability explorations
- State-of-the-art: 25 publications investigated
  - Methods: analytical (9) vs. learning (10) vs. others
  - Representativeness (1-169 kernels/apps)
  - Portability (1-9 GPUs)
  - Availability (only 2 models published)
  - DVFS support (6)
  - Time (21); power consumption (10)

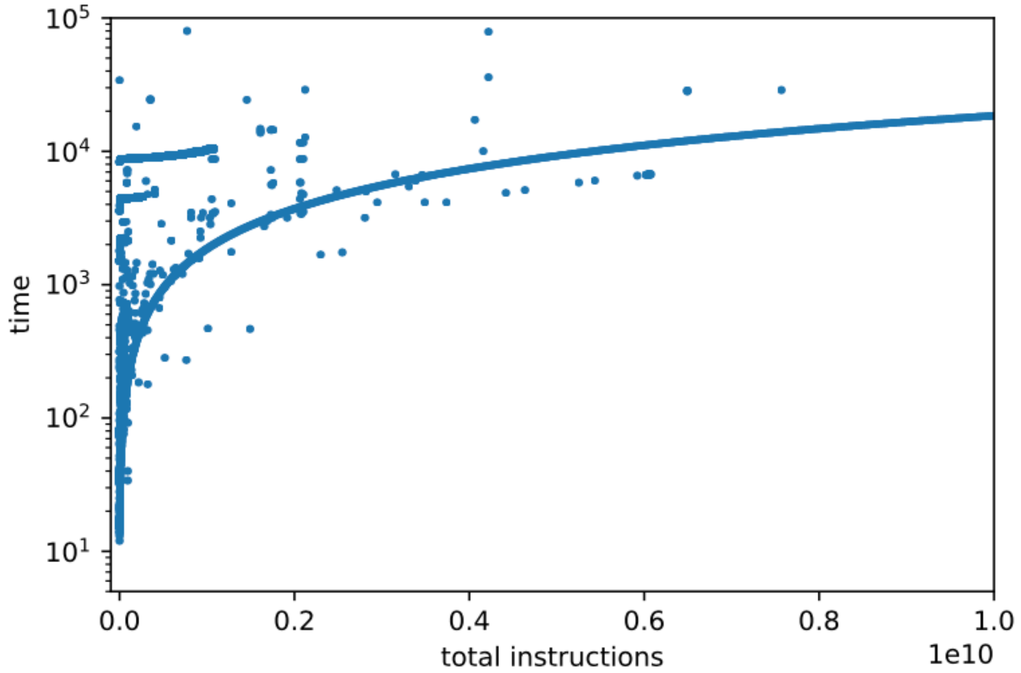


Lorenz Braun, Holger Fröning, CUDA Flux: A Lightweight Instruction Profiler for CUDA Applications, PMBS Workshop@SC2019
https://github.com/UniHD-CEG/cuda-flux

GPU MANGROVE: PORTABLE, FAST, SIMPLE

Which metrics make good features?

- Instructions executed
- FLOPs
- Memory footprint
- Kernel launch configuration
- Computational intensity
- Synchronizations



Portable code features only depend on the kernel and the data handed to it

- Hardware metrics like cache-hit rates not allowed
- Creation of models for new GPUs requires only time and power measurements
- Instruction counter are essential; represent actual work of the processing units

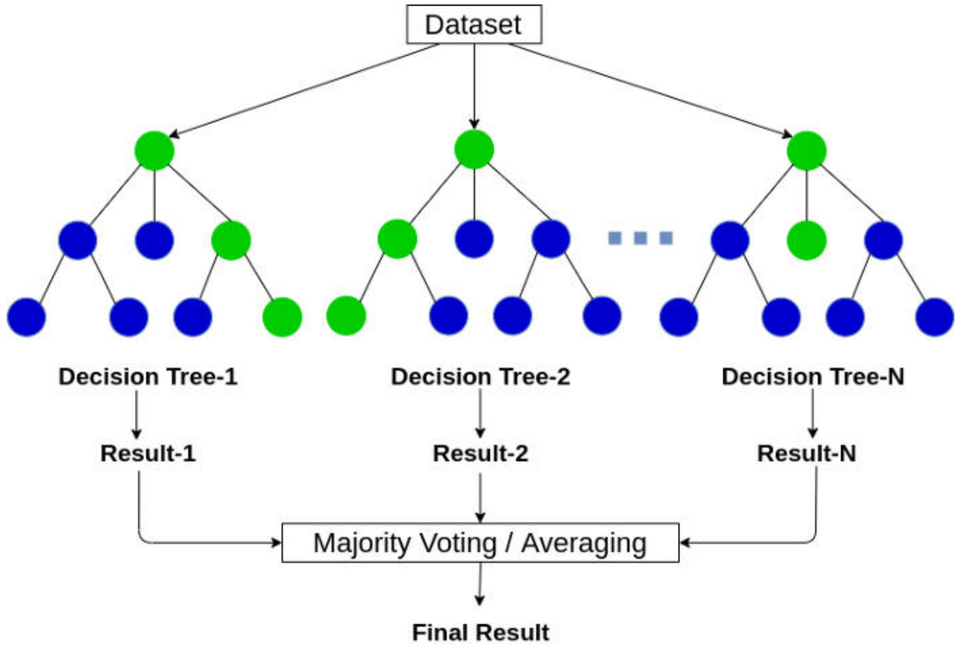
RandomForests

Light computational workload

Likely to over-fit (but can be improved by training method)

Works well with even few samples

Interpolation outside range of training data is difficult



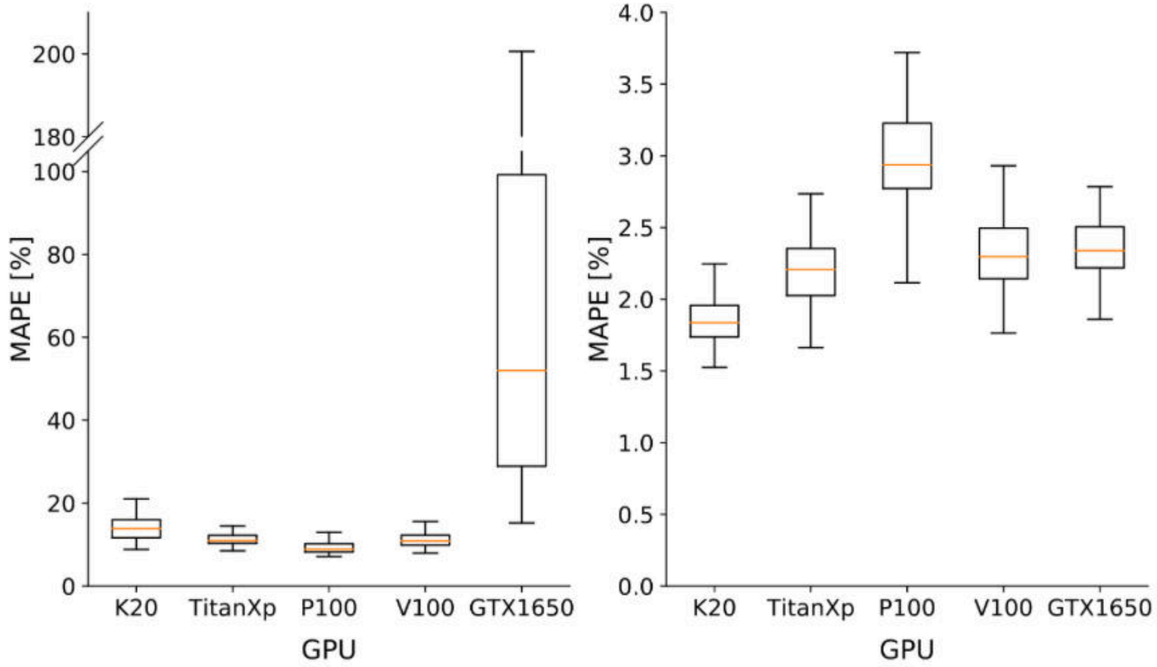
https://medium.com/@gupta020295/random-forest-easily-explained-4b8094feb90

Methodology

189 unique kernels from Parboil, Rodinia, Polybench-GPU and SHOC

Prediction accuracy: 8.86–52.0% for time, 1.84–2.94% for power, across five different GPU

Prediction latency: 15-108ms (not optimized)



Time prediction

Power prediction

Accepted at TACO2021: http://arxiv.org/abs/2001.07104

https://github.com/UniHD-CEG/gpu-mangrove

Tutorial @ HiPEAC2021: https://www.hipeac.net/2021/budapest/#/program/sessions/7856/

REGIME II: DATA MOVEMENT & LOCALITY

PERFORMANCE SCALING

$$Perf(\frac{ops}{s}) = \underbrace{\frac{Instructions}{cycle}}_{\propto PipelineCount \cdot PipelineDepth} \cdot \underbrace{frequency}_{\text{scales with feature size}}$$

CLASSICAL DENNARD SCALING

$$Perf(\frac{ops}{s}) = \underbrace{Power(W)}_{\text{fixed}} \cdot Efficiency(\frac{ops}{Joule})$$

POST DENNARD SCALING

REGIME I

operator cost

Specialization —> heterogeneity and asymmetry

GPU COMPUTING

PROGRAMMING COMPLEXITY

NUMA EFFECTS & LATENCY HIDING

\+

REGIME II

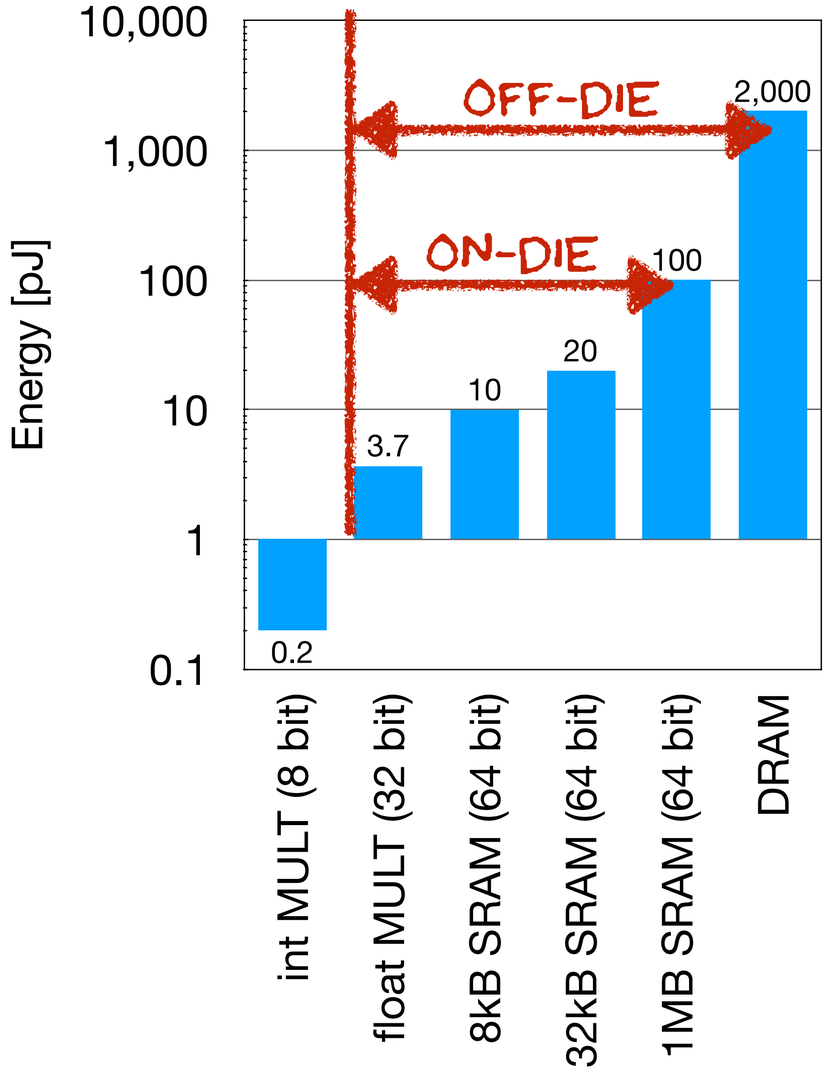
data movement cost

3 operands x 64bit/operand

$$Energy = \#bits \cdot dist[mm] \cdot energy_{per\ bit, per\ mm}[\frac{J}{mm}]$$

Partly by Bill Dally, Sudha Yalamanchili (UCAA Workshop, 2012)

ENERGY DOMINATED BY DATA MOVEMENTS



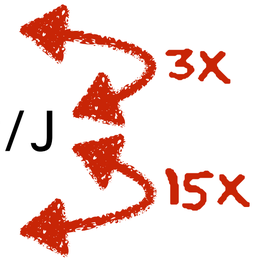

```
void saxpy (...) {
  int i;
  for ( i = 0; i < N; i++ ) {
    y[i] = alpha * x[i] + y[i];
  }
}
```

2 SP FLOPs, 3 references

All registers: 2OP/37.4pJ = 53.48 GOP/J

One miss to LLC cache: 2OP/127.4pJ = 15.7 GOP/J

One off-die access: 2OP/2027.4pJ = 0.99 GOP/J



Performance: multiply with power budget

Optimization toolbox?

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

REMINDER: LATENCY TOLERANCE TECHNIQUES

Property	Relaxed Consistency Models	Prefetching	Multi-T…	Block Data Transfer
Types of latency tolerated	Write (blocking read processors) Read and write (dynamically scheduled processors)	W…	…ad …chronization	Write Read
Software requirements	Labeling synch… op…	…ictability	Explicit extra concurrency	Identifying and orchestrating block transfers
Extra hardware support		Little	Substantial	Not in processor, but in memory system
Supported in commercial systems?	Yes	Yes	Yes	Yes

ALL FAIL TO HIDE ENERGY

David E. Culler, Jaswinder Pal Singh, Anoop Gupta, Parallel Computer Architecture: A Hardware/Software Approach, Morgan Kaufmann,1998

COMPUTE STACK

Post-Dennard I (specialization): today

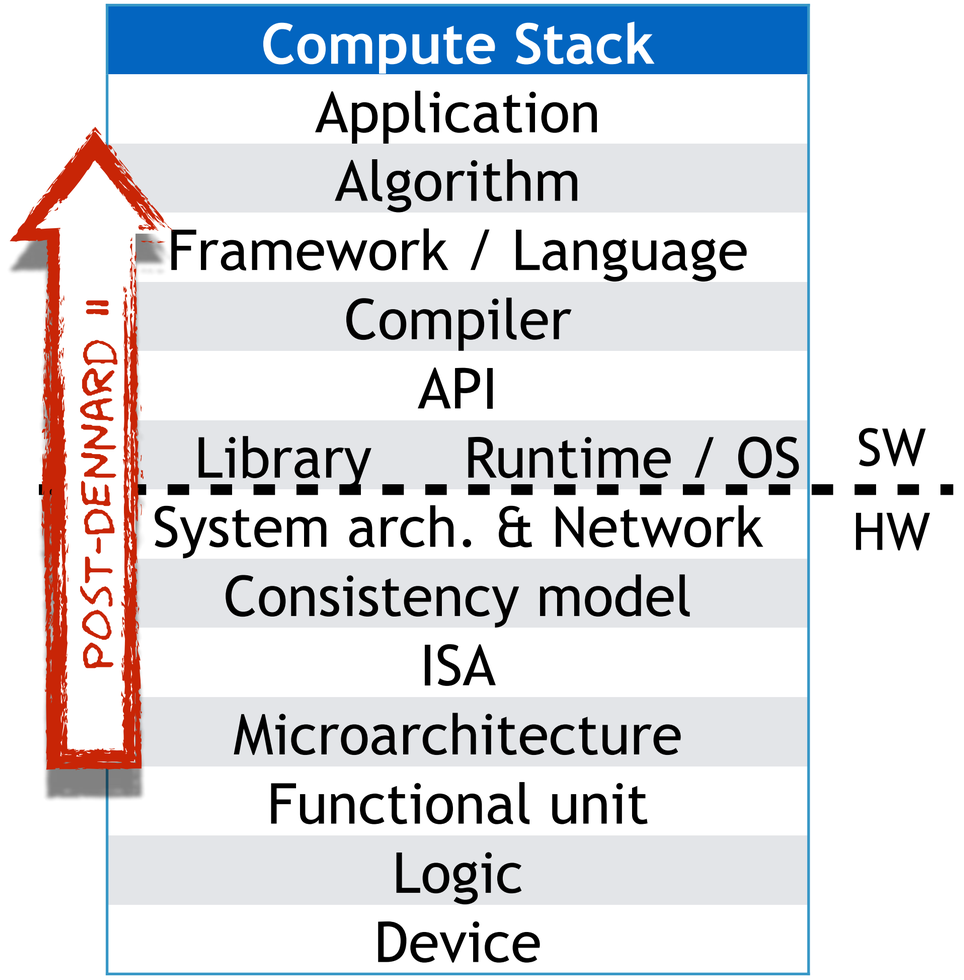
Some uarchs tend to actually converge, but programming models diverge => **replicated compute stacks**

Massive parallelization requires **structure** for efficient execution

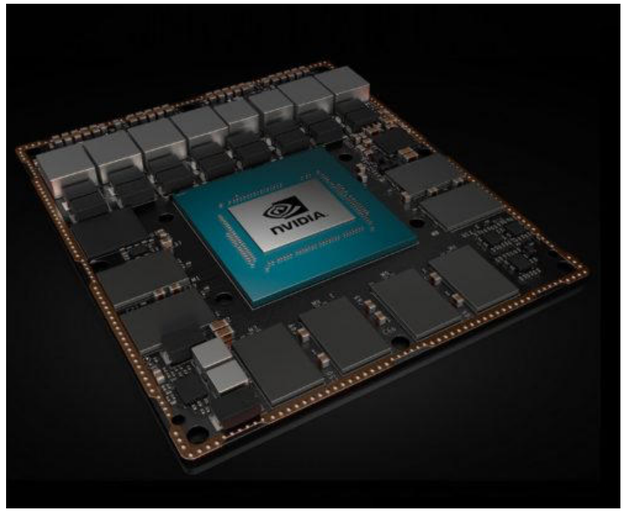
Post-Dennard II (locality-centric): emerging

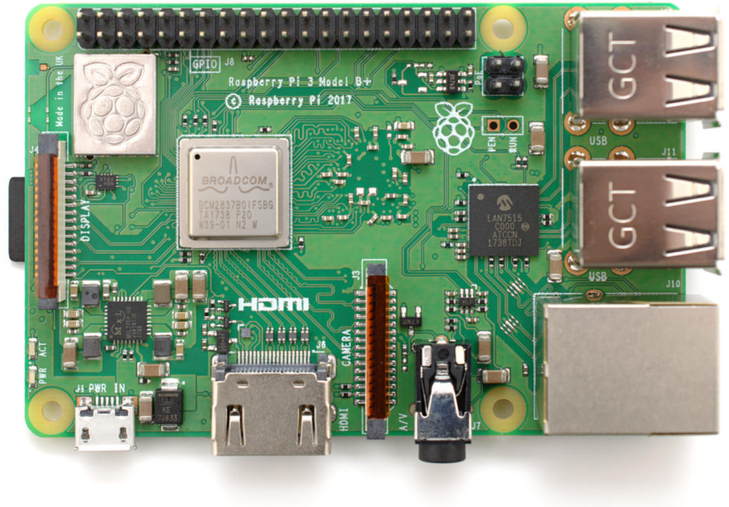
Energy = locality, but missing notion of locality hinders optimizations

Severity of implications demands for both safe and unsafe optimizations



DL AND MOBILE COMPUTING





	NVIDIA Xavier	XILINX Zynq Ultrascale+ ZU19EG	Raspberry Pi 3 B+
Wattage	30W	~10W	6W
Peak GFLOP/s	1,300 (325 images/s[1])	difficult	5.6 (1.4 images/s[1])
Total memory	16GB	2GB	1GB
In-core memory	2.9MB (2.8%[2])	4.3MB (4.2%[2])	2.3MB (2.3%[2])

[1] based on theoretical peak GFLOP/s performance, [2] weights only, both for ResNet-50/ImageNet

Extreme mismatch between DNN complexity and mobile processor capability

DEEPCHIP PROJECT

DL-based speech & image processing for resource-constrained devices

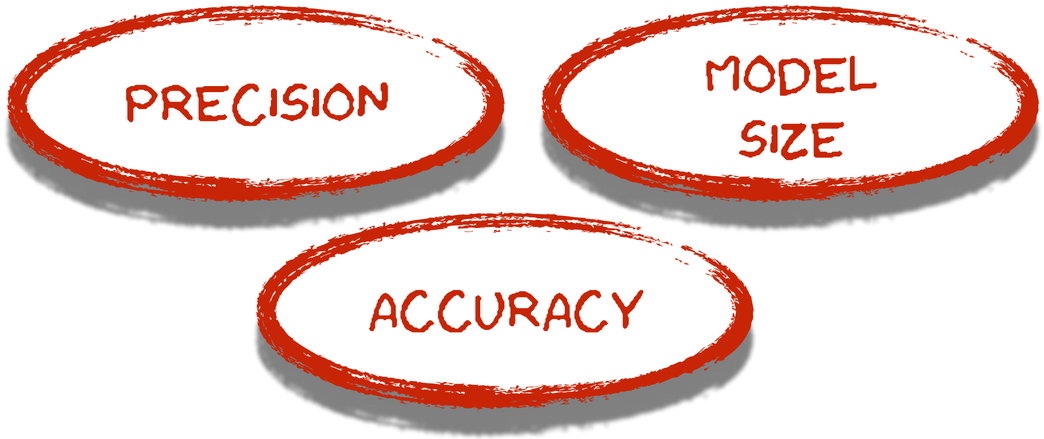
- Trading among precision, model size and accuracy
- Preferred: no accuracy loss compared to state-of-the-art

Reduced precision (quantization), sparsity and asynchrony

1. Inference architecture suitable for various embedded processors
2. New neural networks concepts with particular low requirements
3. Software inference architecture based on quantization and pruning
4. Exploring applicability to various processors

Collaboration with SPSC group @ TU Graz

http://www.deepchip.org



REDUCE & SCALE

Weight quantization to ternary values according to TTQ

- Trained scale factors W^p and W^n
- Bounding activations, quantization to fixed point (flexible bit-width k, DoReFa)

Run-length encoding of weight matrix (reduced cardinality) & Huffman coding

=> Saving complexity: reduced precision, sparsity, only partial sums (+2 mults.)

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

$$w'_{i,l} = \begin{cases} W_l^p : w_{i,l} > \Delta_l \\ 0 : |w_{i,l}| \leq \Delta_l \\ -W_l^n : w_{i,l} < -\Delta_l \end{cases}$$

$$\Delta_l = t \cdot max(|w|); t \in [0, 1]$$

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

$$\mathbf{i}_l^p = \{i | w_{i,l} = W_l^p\} \quad \text{and} \quad \mathbf{i}_l^n = \{i | w_{i,l} = W_l^n\}$$

AlexNet/ImageNet

	Baseline	BNN	INT8	DeepChip
Top-5 Accuracy [%]	78.3	56.4	-[1]	79.0
Sparsity [%]	0.0	0.0	0.0	63.0
Inference Rate [FPS]	4	22	7	8
Memory [MB]	244	24	61	25

[1] *Authors claim no change in accuracy*

Günther Schindler, et al, Towards Efficient Forward Propagation on Resource-Constrained Systems, ECML-PKDD 2018

N-ARY QUANTIZATION (NAQ)

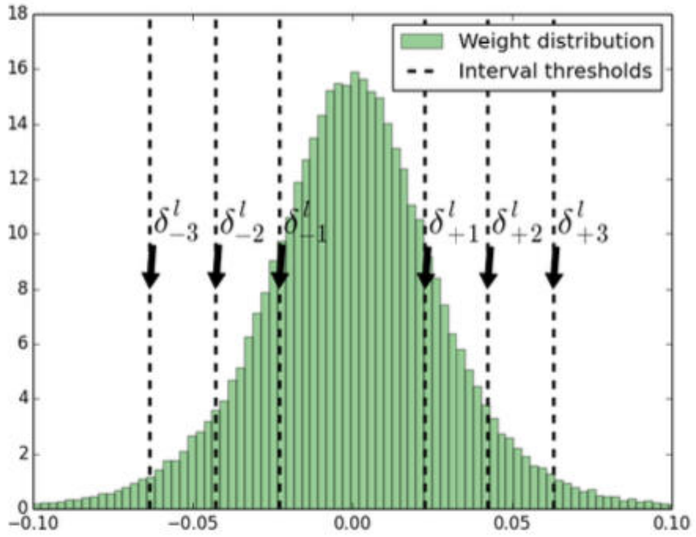
Up to now: all good for ConvNet+SVHN, AlexNet+ImageNet, ResNet-44+CIFAR-10

I.e., complex model + simple data, or simple model + complex data

But: quantization depends on complexity(data) & complexity(model)

Non-uniform n-ary weight representations

Multiple scale factors, cost-effective nested-means clustering



ResNet-18/ImageNet

	Weights [bit]	Activations [bit]	Training	Top-5 [%]
LQNet	2	2	2.3x	85.9
DeepChip - ternary	2	2	1.2x	86.7
LQNet	3	32	1.7x	88.8
DeepChip - quinary	3	32	2.0x	89.0

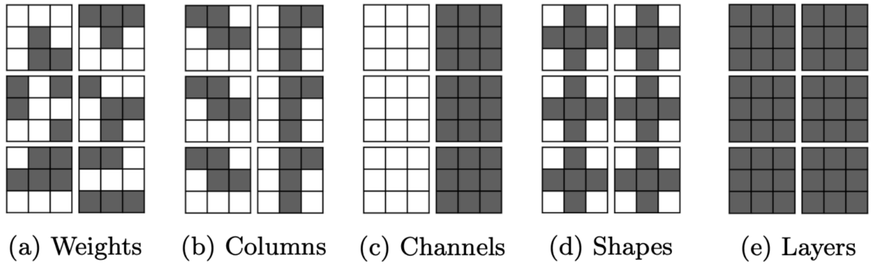
STRUCTURED PRUNING

Unstructured sparsity for HW

- Systolic-array based accelerators: dense computation and predictability
- CPUs/GPUs: usually only sparsity > 90% effective (various reasons)

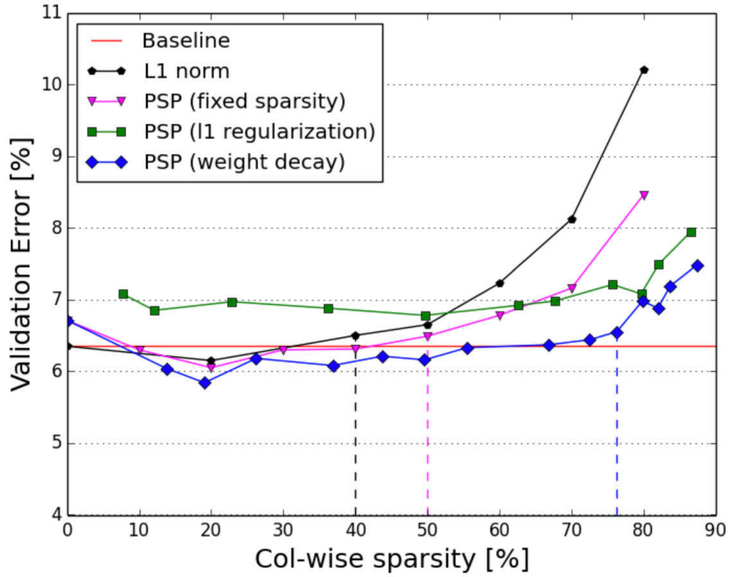
Parametrized structured pruning (PSP) with trainable parameters

- Consider weight tensor W, input tensor x, linear operation $\oplus$, non-linear function g: learn a structured substitute Q for the weight tensor W resp. sub-tensors w_i
- Gradient of α_i is calculated following the chain rule
- Trained together with weight using gradient descent, but regulated and pruned independently
- If $\alpha_i < t$ (tunable threshold parameter), prune the corresponding structure (post-training)



$$z = g(\mathbf{W} \oplus \mathbf{x})$$

$$\mathbf{q}_i = \mathbf{w}_i \alpha_i$$



Günther Schindler, et al., Parameterized Structured Pruning for Deep Neural Networks, 6th International Conference on Machine Learning, Optimization, and Data Science (LOD 2020). Best paper finalist.

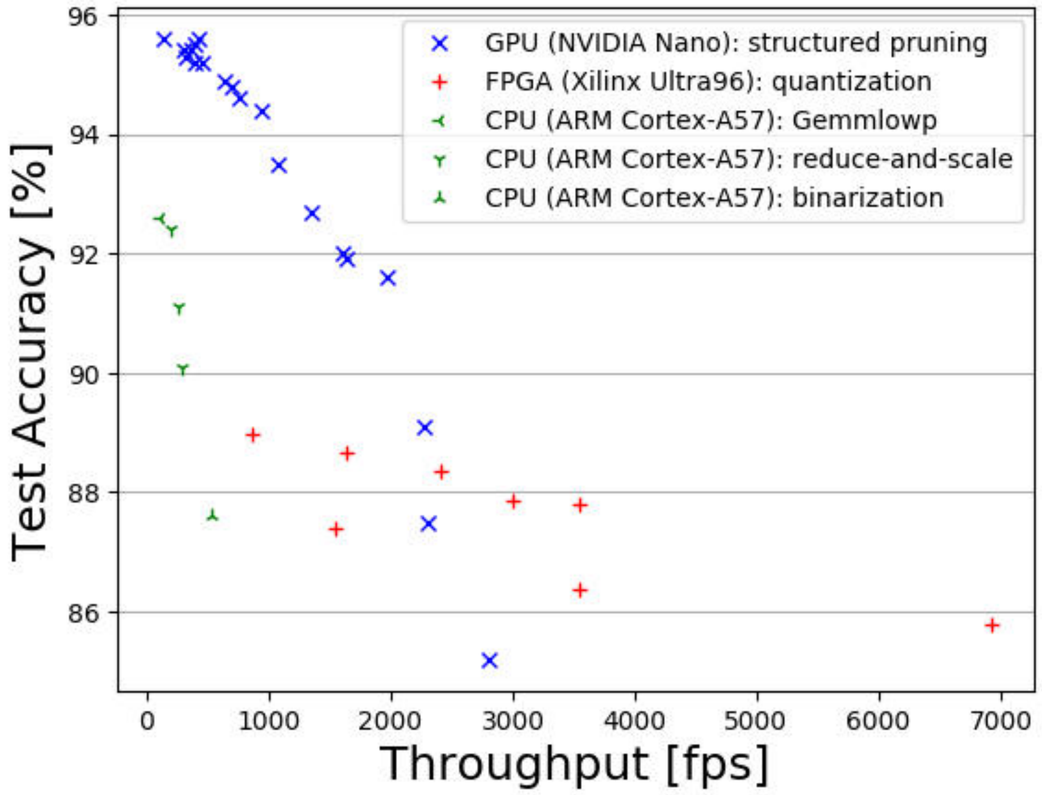
NN COMPRESSION IN A NUTSHELL

Meantime

- (Restricted) architecture search
- PSP for FPGAs, extending FINN
- Auto-generation of quantized operators for ARM processors (AccML@HiPEAC2021)
- Emulation of TF operators on systolic arrays for explorations of architectural parameters (Camuy)

NN compression is dependent on {Data,Model,HW}

- ResNet on CIFAR-10
- Different options @5W



WRAPPING UP

WRAPPING UP

Moore’s law will deliver, Dennard scaling is failing

- Anticipated growth in device count for another 5-10 years

Specialization will dominate computer architecture => tension in between performance (specialization) and programmability (generality)

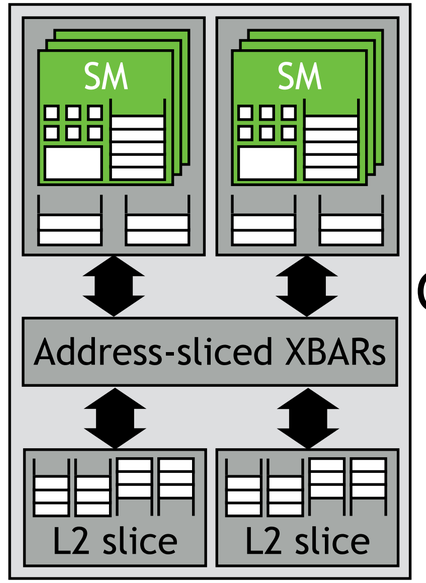
- Post Dennard Regime 1 => **structure**
- Post Dennard Regime 2 => **locality & predictability**

Bill Dally: “locality is efficiency, efficiency is power, power is performance, performance is king” (2011) => **safe & unsafe optimizations**

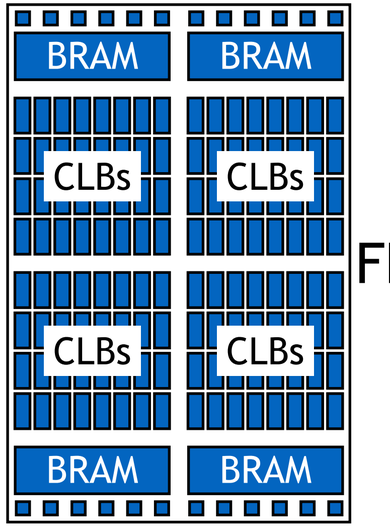
- picojoule replaces nanosecond (there be dragons [1])

Our main research concept: **app+data=architecture**

- In spirit of algorithms + data structures = programs [2]



GPU



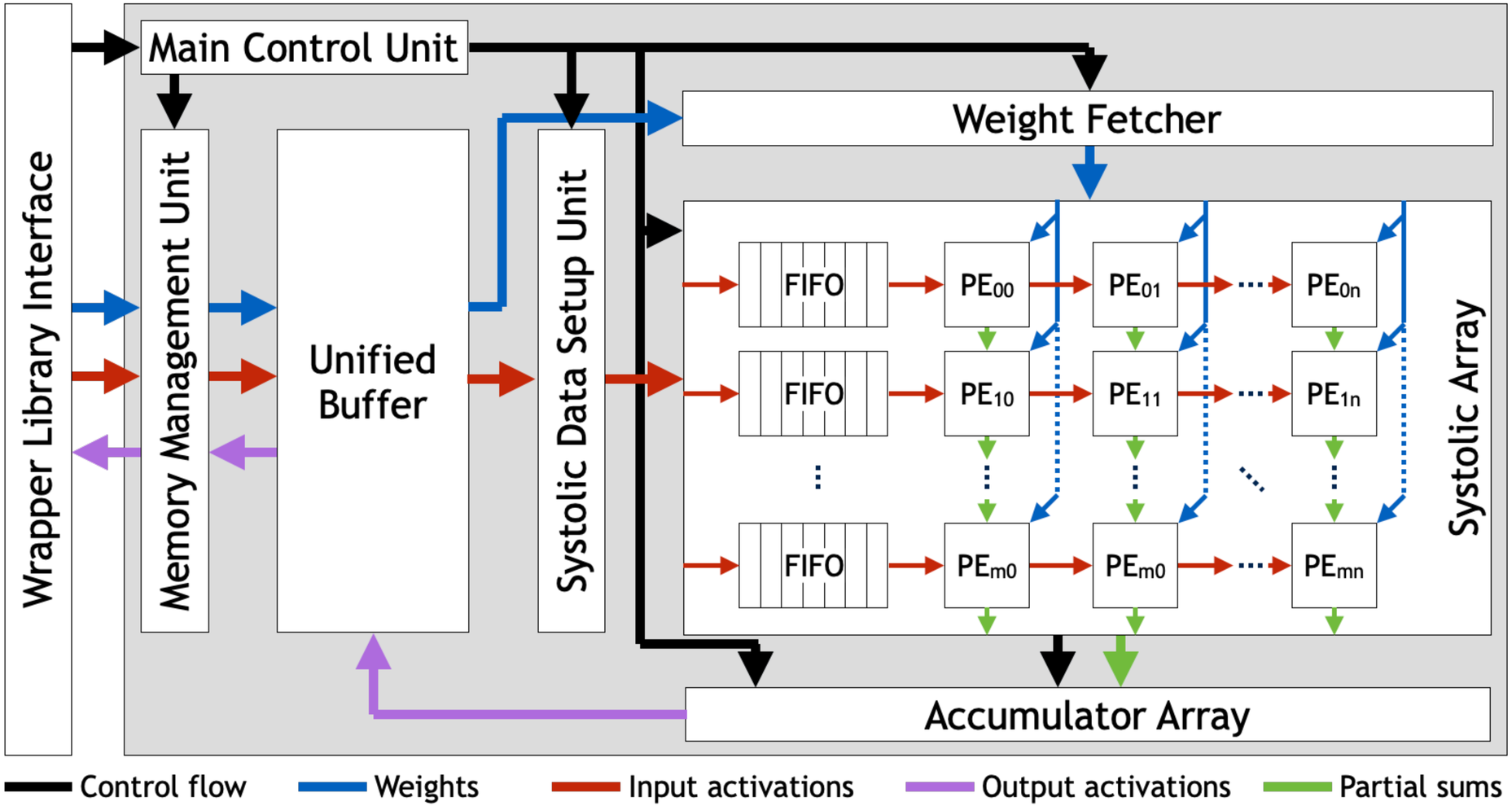
FPGA

[1] "Here be dragons" (Latin: ‘hic sunt dracones’): dangerous or unexplored territories

[2] N. Wirth. *Algorithms + Data Structures = Programs*. Prentice-Hall, 1976.

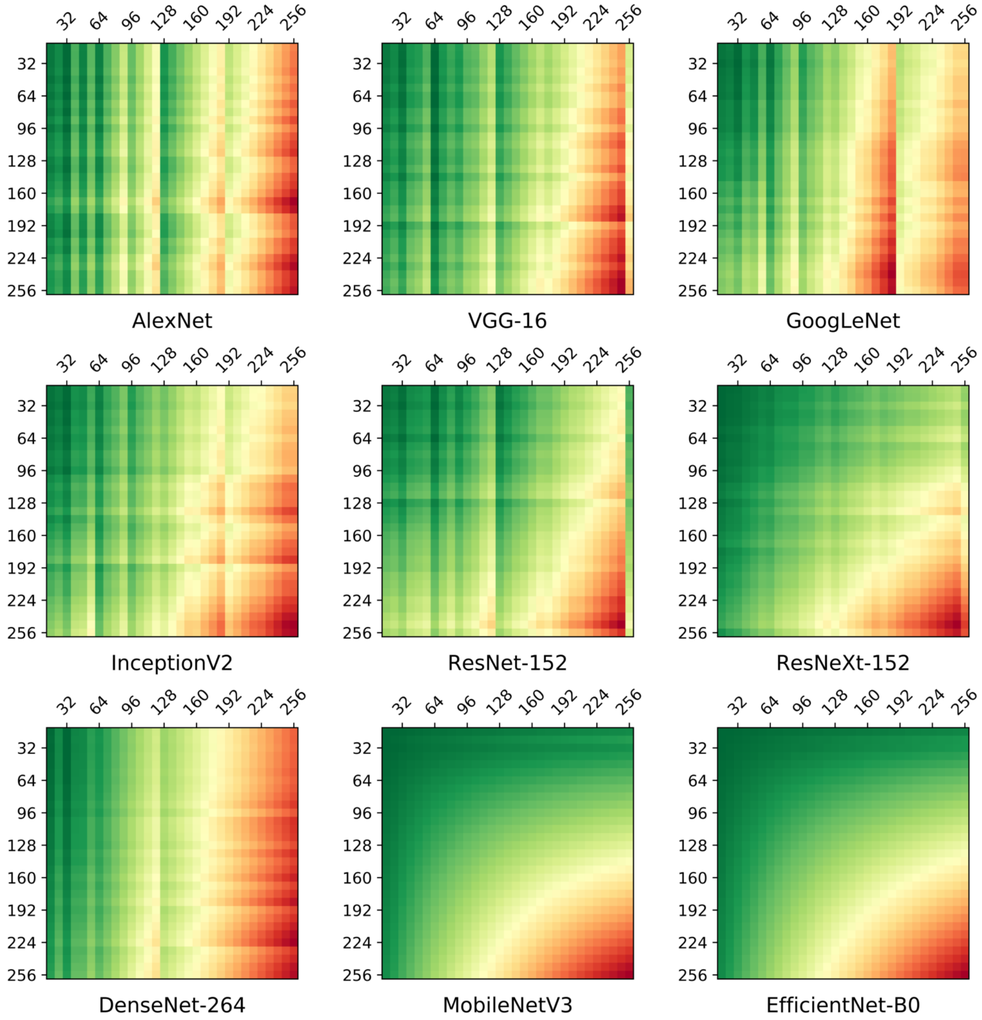
BACKUP

CAMUY EMULATOR

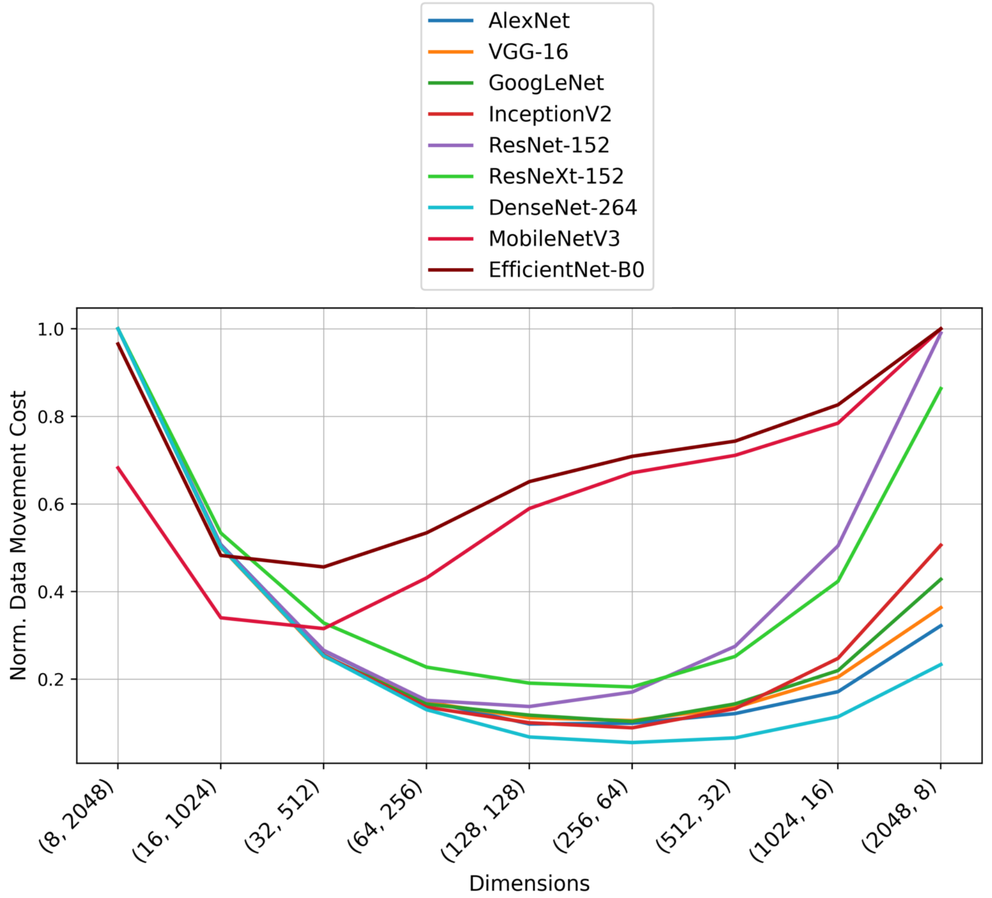


Kevin Stehle, Günther Schindler, Holger Fröning, On the Difficulty of Designing Processor Arrays for Deep Neural Networks, ITEM Workshop@ECML-PKDD2020.
https://arxiv.org/abs/2006.14008
https://github.com/UniHD-CEG/camuy

CAMUY RESULTS



Comparison of data movement heat maps for different CNN models, for different SA dimensions



Normalized data movement cost using systolic array dimensions with equal PE counts