Hewlett Packard Enterprise

THE BRAVE NEW WORLD OF EXASCALE COMPUTING: COMPUTATION IS FREE, DATA MOVEMENT IS NOT

Utz-Uwe Haus, Head of HPE HPC/AI EMEA Research Lab

2021-03-03

TRR154/MINOA conference “Trends in Modelling, Simulation and Optimisation: Theory and Applications”

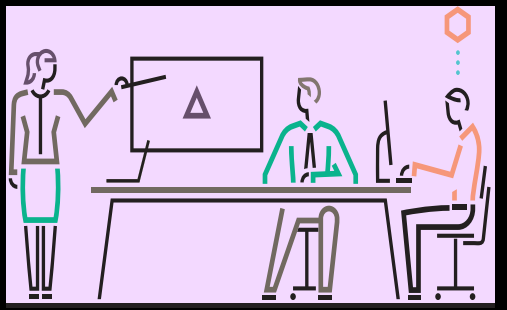
Supported by the European Union’s Horizon 2020 research and innovation program through grant agreement 801101.

MAESTRO DATA ORCHESTRATION

HPE HPC/AI EMEA RESEARCH LAB

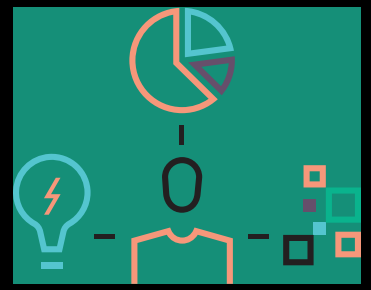
Deep Technical Collaboration

- HPE & Customers work together
- Focus on new technologies
- Drive future HPE products
- Long term technical relationship



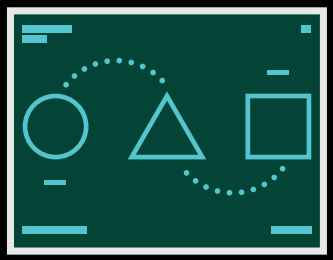
Research Interests

- Memory hierarchy
- Data Movement and Workflows
- Novel accelerators, highly heterogenous systems
- Compilers and mathematical optimisation
- HPC in Cloud, AI and Big Data
- System and site monitoring and data analysis



Engagement Models

- Advanced Collaboration Centers in Centers of Excellence
- Value Add projects
- EU H2020 research projects



ADVANCED COLLABORATION CENTERS IN EMEA

ARCHER/ARCHER2, UK

- LASSi - IO Monitoring and Analytics
- Application tuning (XC30/EX)
- IO Performance Optimisation

KAUST, KSA

- Numerical linear algebra libraries
- Asynchronous tasking
- Deep Learning for Bio-Science

GW4

- ARM system tuning
- ARM ecosystem development
- Joint ARM, Cavium partnership

Coming up: LUMI and HLRS

CURRENT H2020 PROJECTS

- EXPERTISE
- MAESTRO
- EPIGRAM-HS
- SODALITE
- Plan4Res
- Funded PhD secondments

Imperial College London
Barcelona Supercomputing Center
Centro Nacional de Supercomputación
HLRS High-Performance Computing Center | Stuttgart
MAVEL
Politecnico di Torino
ECMWF
SEAGATE
CEA
appentra
CSCS
JÜLICH Forschungszentrum
epcc
ETH Zürich
Fraunhofer ITWM
KTH Vetenskap och Konst
HLRS High-Performance Computing Center | Stuttgart
ADAPTANT
XLAB
IBM
Politecnico Milano 1863
iti
CERTH Centre for Research & Technology Hellas
Atos
JADS
Imperial College London
RWTH Aachen University
ZIB
ICOOR
EDF
SIEMENS Ingenuity for life
EuroPLEx
MINOA Mixed-Integer Non-Linear Optimisation: Algorithms and Applications

WHAT IS A SUPERCOMPUTER?

Any of a class of extremely powerful computers. The term is commonly applied to the fastest high-performance systems available at any given time.

- Britannica

[A device for] processing of massively complex or data-laden problems using the concentrated compute resources of multiple computer systems working in parallel.

- HPE

[a device to] handle and compute on volumes of data at speeds hundreds to millions of times faster than on a typical data center server

- nimbix

A supercomputer is a computer with a high level of performance as compared to a general-purpose computer.

- Wikipedia

A supercomputer is scientific instrument.

- folklore

A supercomputer is a device for turning compute-bound problems into I/O-bound problems.

- Ken Batcher

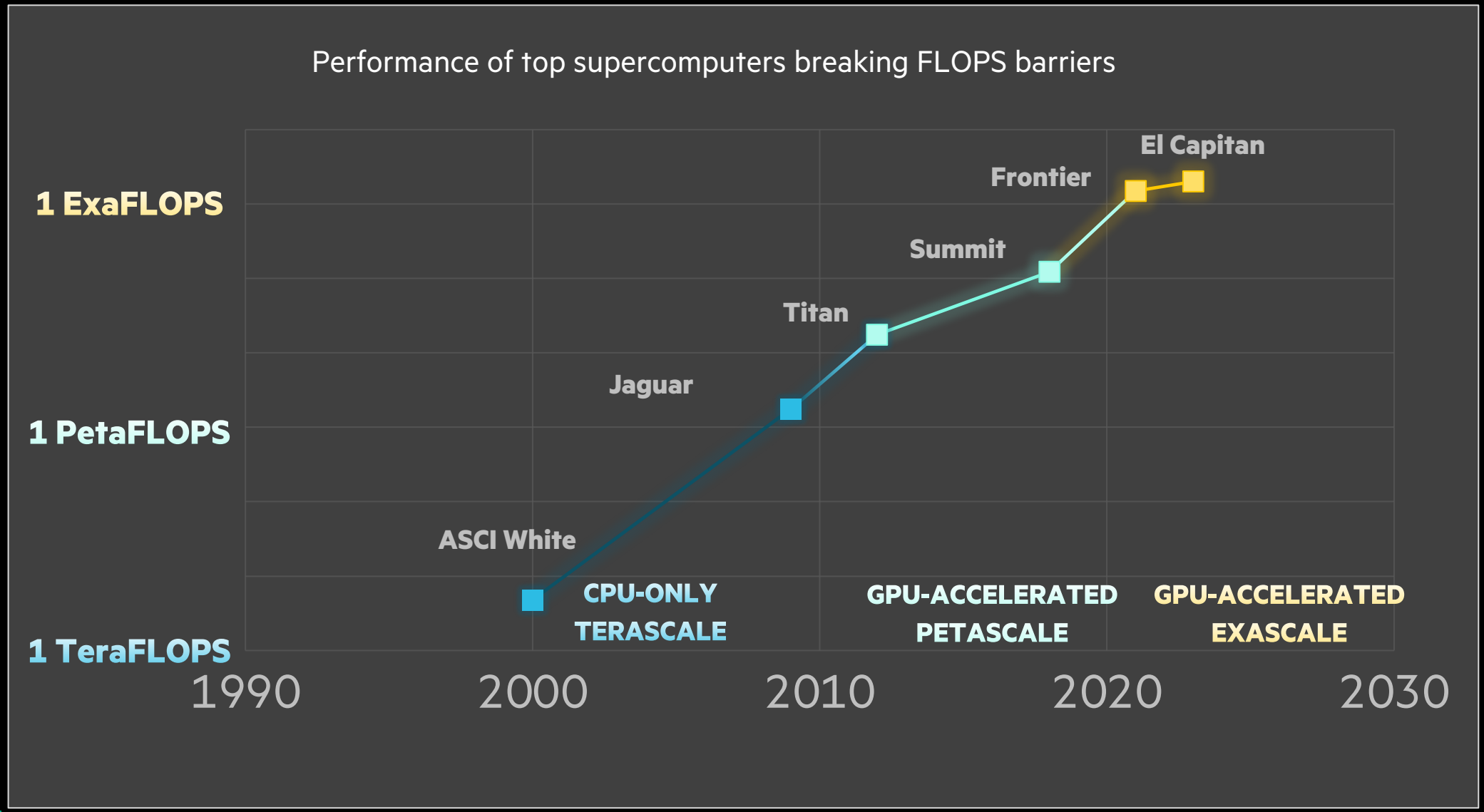
EXA..WHAT?

- Exascale computing: 10^{18} Floating Point Operations per second (FLOPS)
  - Measured by LINPACK:
    - solve $Ax = b$ for dense A
    - using LU factorization with partial pivoting
    - with $\frac{2}{3}n^3 + O(n^2)$ operations
    - In double-precision IEEE floating point
- Theoretical peak performance R^{peak}
  - ignoring communication between compute units

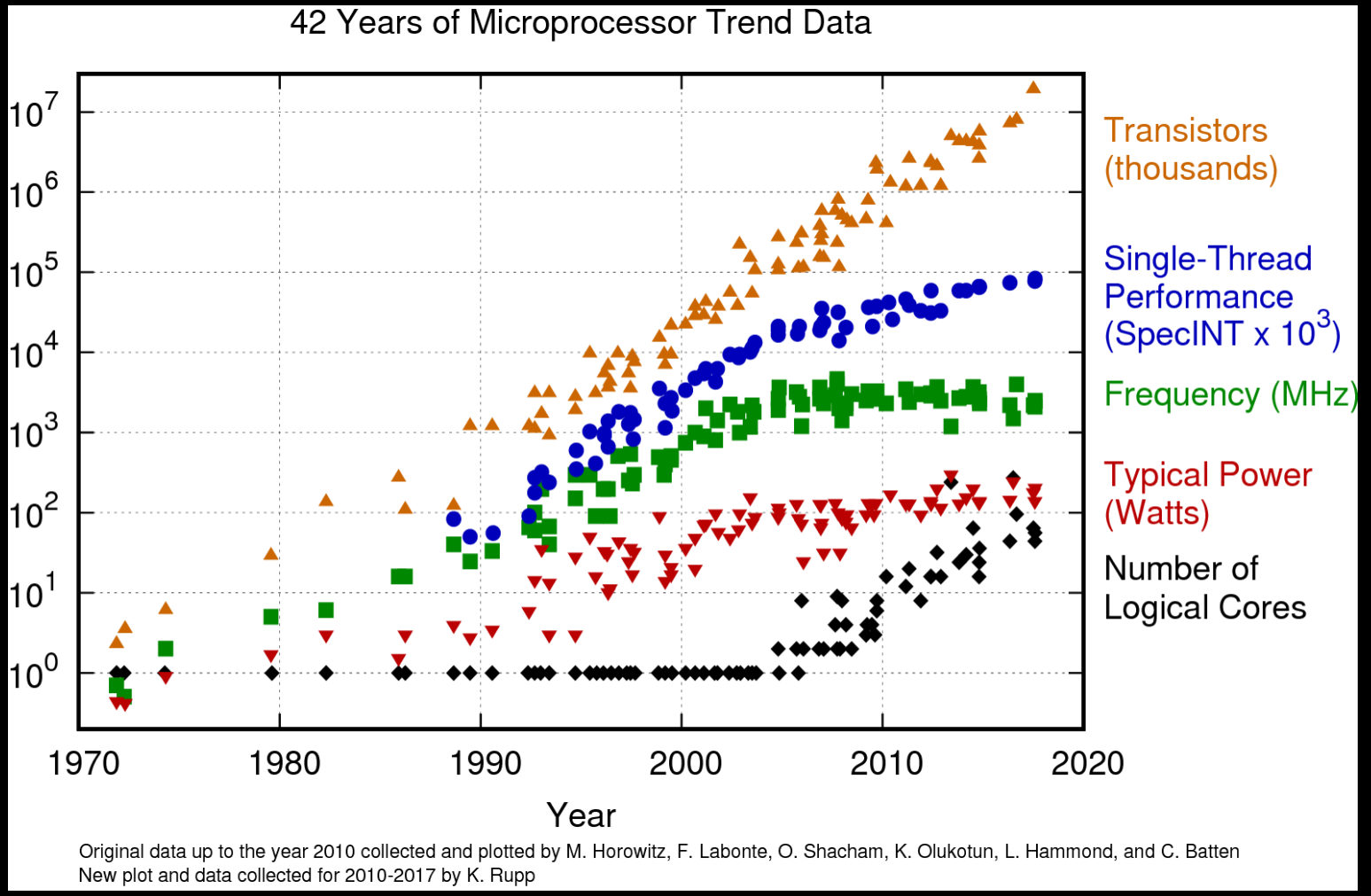
Raspberry Pi-4B	13.5 GF
iPhone 11 A13	0.8 TF
Nvidia Titan V	110 TF

https://www.top500.org/

MAJOR LEAPS IN PERFORMANCE: 3 ERAS OF SUPERCOMPUTING



INGREDIENTS IN A CRISIS CURRENTLY HAPPENING



https://www.karlrupp.net/2018/02/42-years-of-microprocessor-trend-data/

MEMORY V.S. COMPUTATION: THEN AND NOW

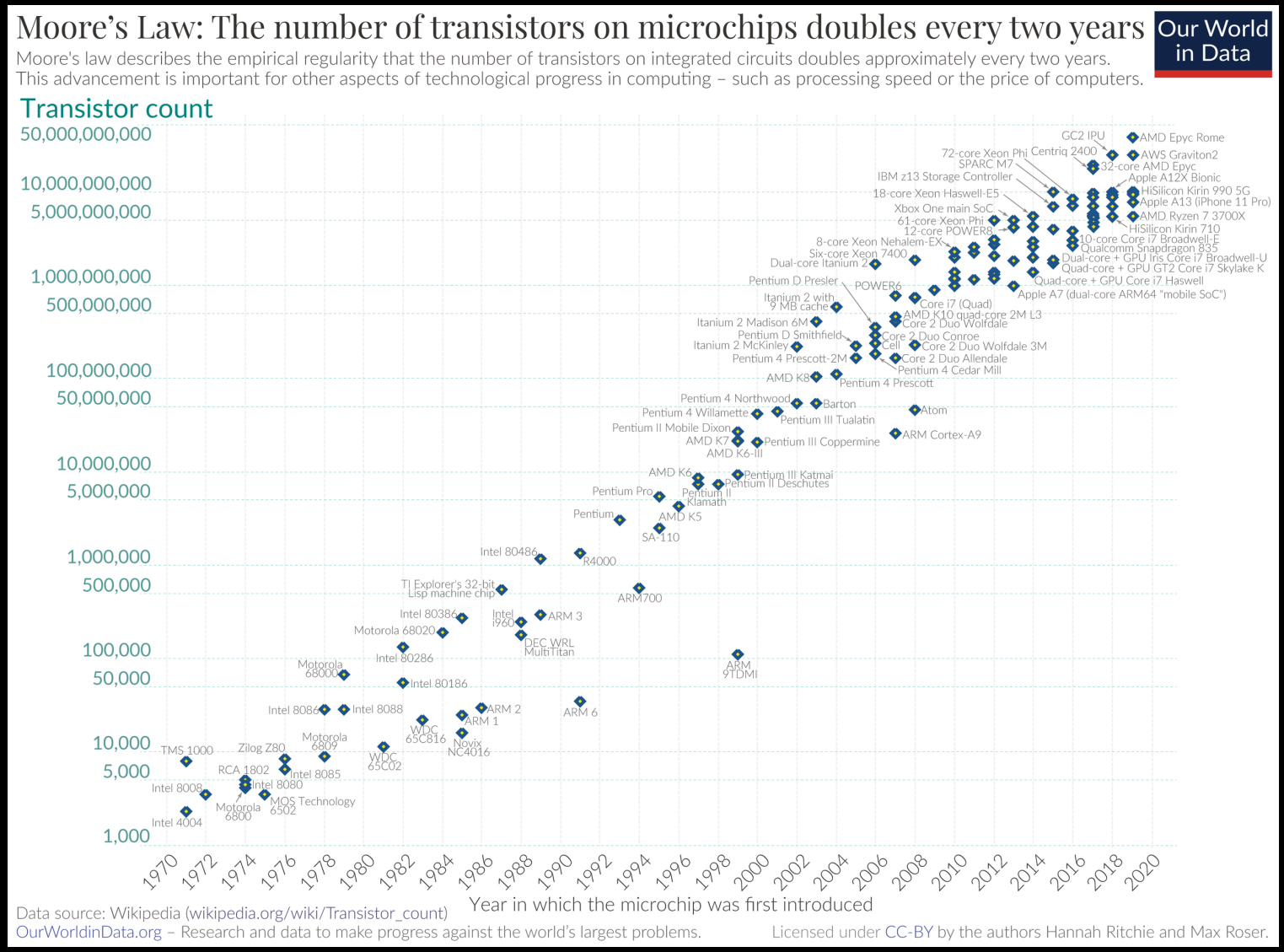
Cray 1, 1975

Fugaku, 2020

System Performance	160 MFLOPS	537 PFLOPS	**3356250000x**
Perf/node	160 MFLOPS	3.38 TFLOPS	**21125x**
Memory capacity /node	8 MB	32 GB	**4000x**
Memory bandwidth /node	640 MB/s	1024 GB/s	**1600x**
Memory bandwidth / flop	4	0.3	**1/13 x**

SEMICONDUCTOR WISDOMS (1)

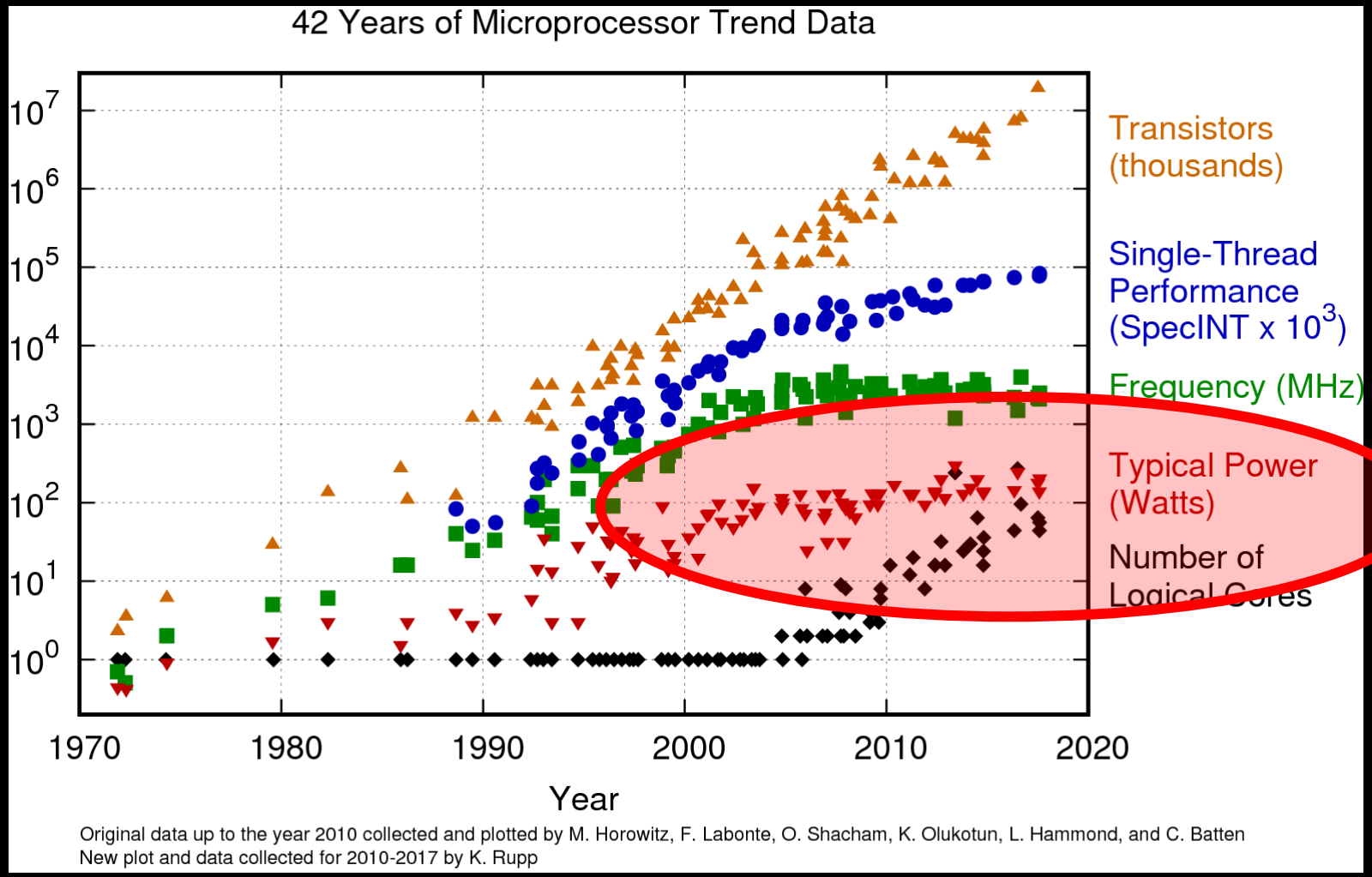
Moore’s Law



“The number of transistors in a dense integrated circuit (IC) doubles about every two years.

SEMICONDUCTOR WISDOMS (2)

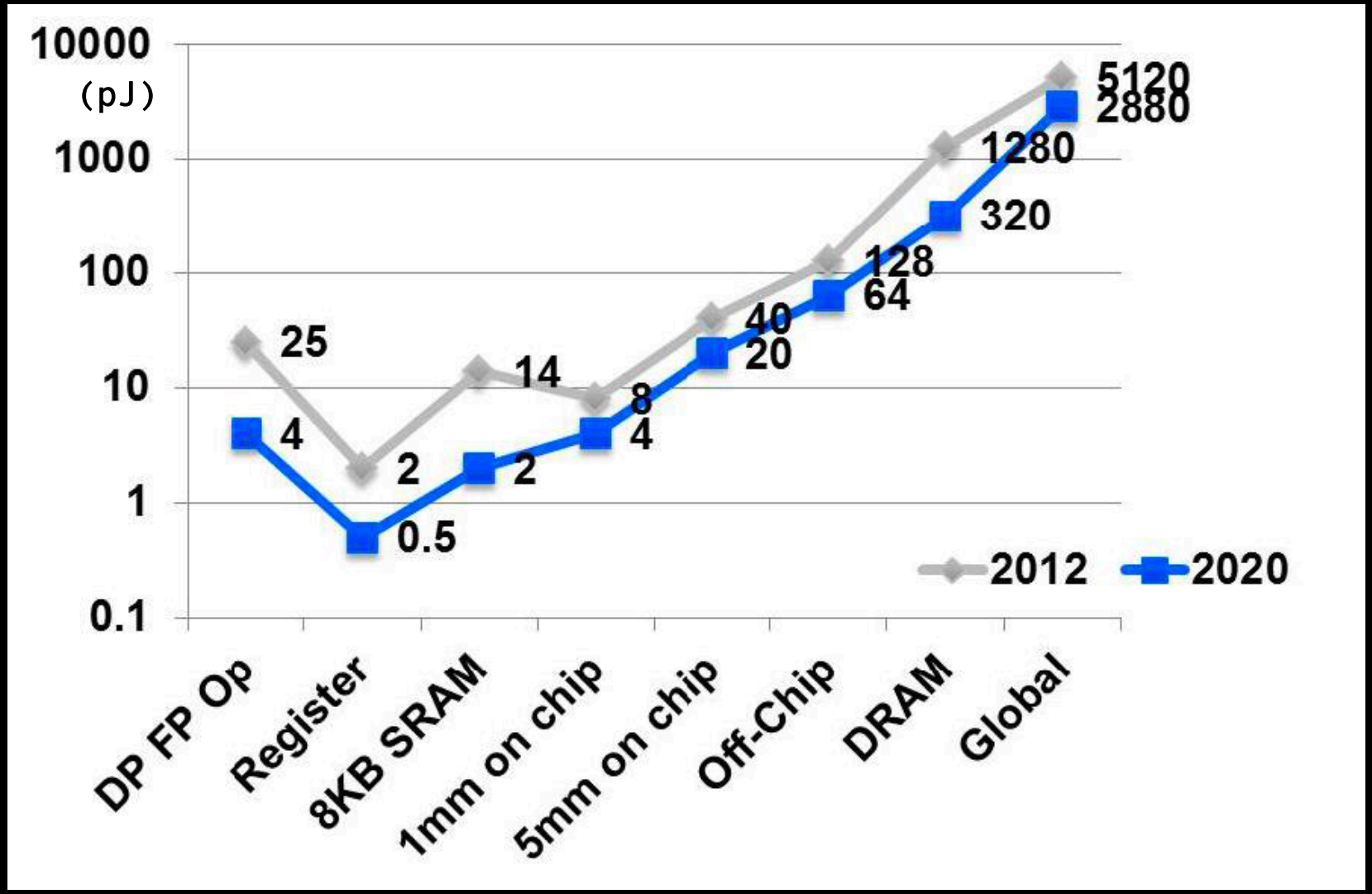
The end of Dennard Scaling (2006)



"As transistors get smaller, their power density stays constant, so that the power use stays in proportion with area; both voltage and current scale (downward) with length"

https://www.karlrupp.net/2018/02/42-years-of-microprocessor-trend-data/

DATA MOVEMENT COST



Energy cost, in picojoules (pJ) per 64-bit floating-point operation.

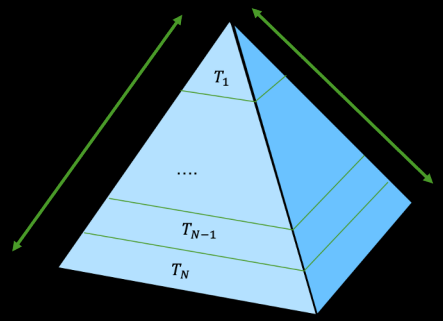
Note that the double-precision floating-point arithmetic (DP FP Op) energy cost is comparable to that for moving the same data 1mm–5mm on chip.

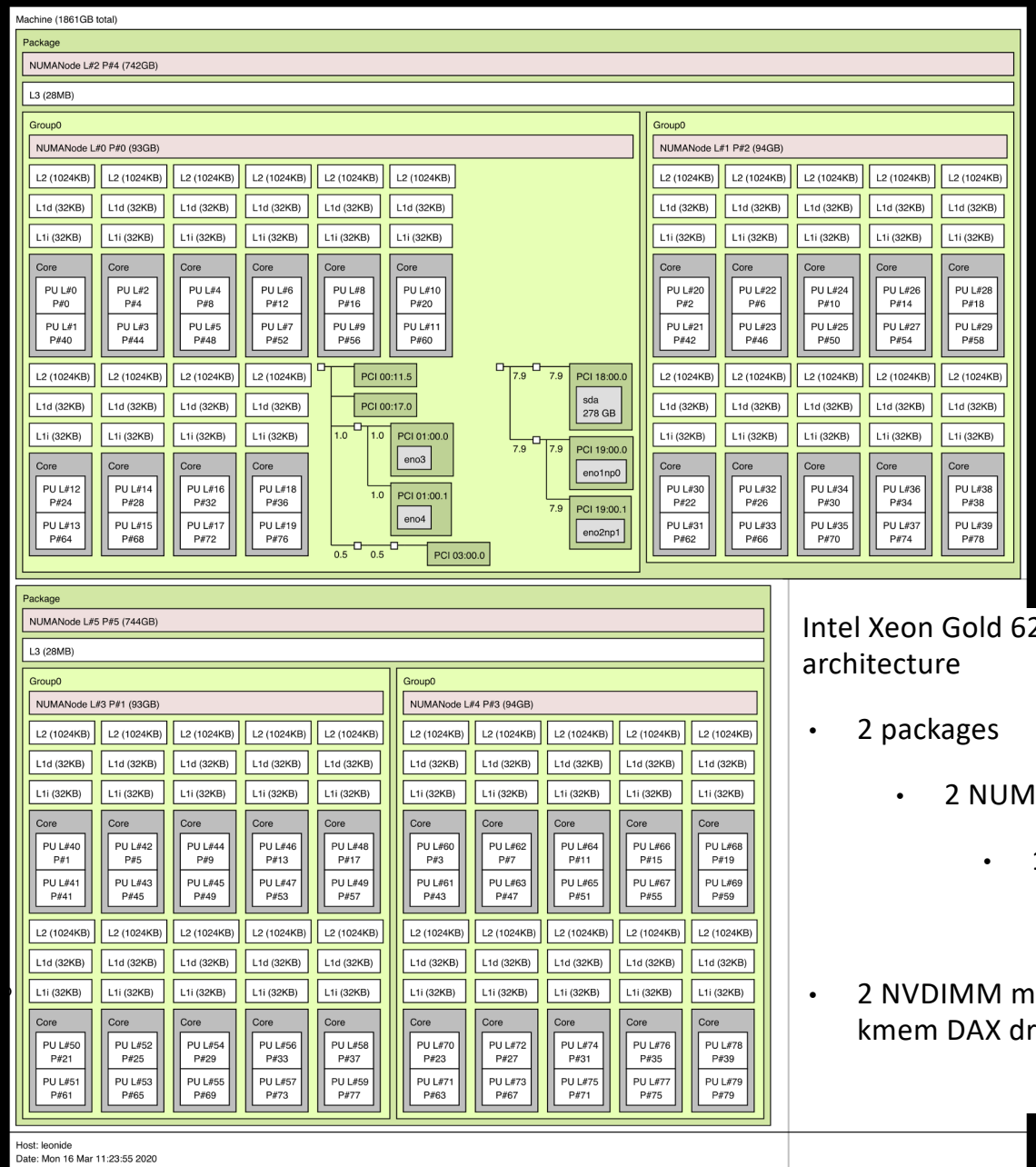
That cost is dwarfed by the cost of any movement of this same data off chip.

Leland et al., report SAND2016-9583

MEMORY IS DIVERSE

- Caches L1,L2,L3
- DRAM
- GDRAM
- NUMA domains
- HBM/MCDRAM
- NVDIMM
- Node-local SSD
- …
- Object Storage
- GFS





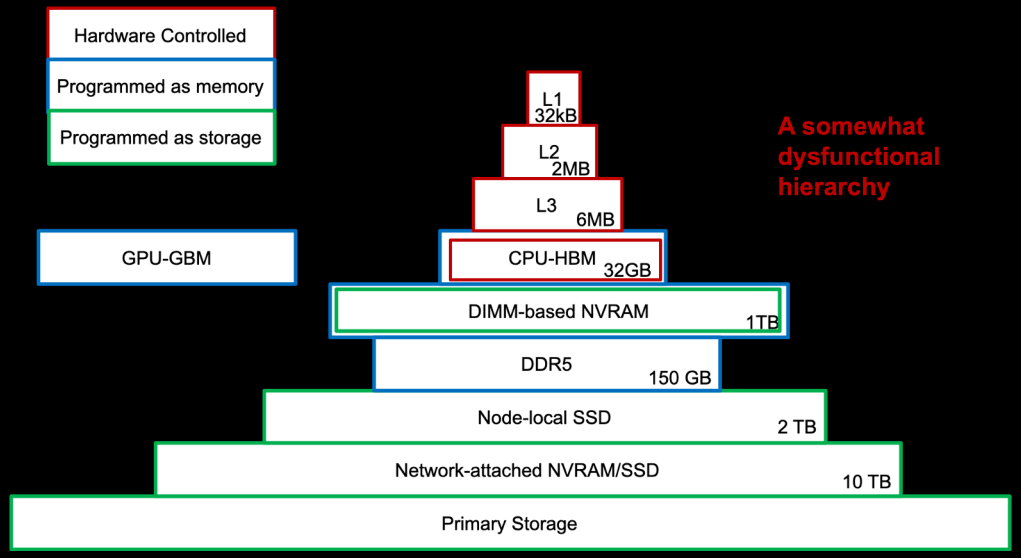
Intel Xeon Gold 6230 Cascade Lake architecture

- 2 packages
  - 2 NUMA nodes
    - 10 cores
      - 2 threads
- 2 NVDIMM managed with kmem DAX driver

DATA MOVEMENT IS HARD AND A MAJOR PERFORMANCE BOTTLENECK

Is it still a hierarchy?

- Latency, bandwidth, capacity numbers not monotone anymore
- Some have separate address spaces
- Some not under our control



It's a dynamic (robust) vehicle routing problem with inventories and splittable resources. Contains

- Splittable flow
- Packing
- Job shop scheduling

➤ Mathematically hard; hard to approximate

➤ Practically hard:

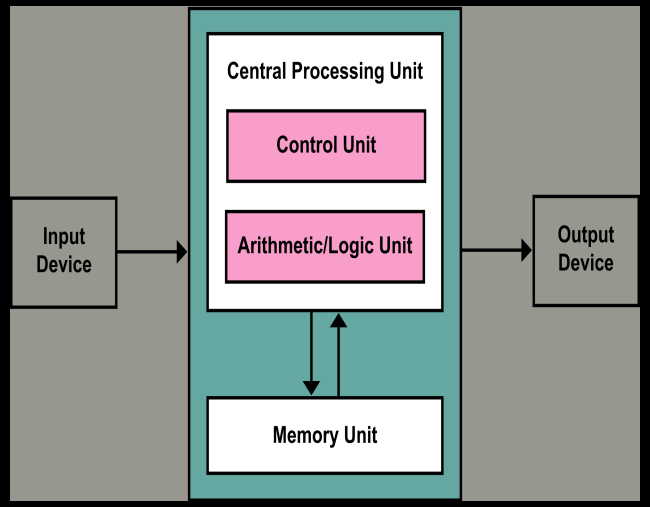
- No uniform programming model
- no system model to compute optimal schedules at scale

INGREDIENTS FOR A SOLUTION

Move computations, not data

Decouple instructions (code) and data movement

Beware of von Neumann architecture bottleneck, in particular the word-by-word traffic paradigm



Wikimedia, CC BY-SA 3.0

Count energy, not instructions

Remember when + and * were counted with different costs multipliers in CS101?

Our complexity classes don't capture instruction sets well where computation and data movement have exponentially different cost.

Communication-avoiding algorithms are a niche topic.

Recompute instead of reload

Overlapping data movement and compute is no longer sufficient.

Ghosting, cache-oblivious or write-avoiding algorithms, architecture/topology-aware implementations are limited (and not future-proof)

Functional programming is the right abstraction: well-defined side-effect-free/closure-confined computations.

TASKING – A PARADIGM FOR DATA-AWARE PROGRAMMING

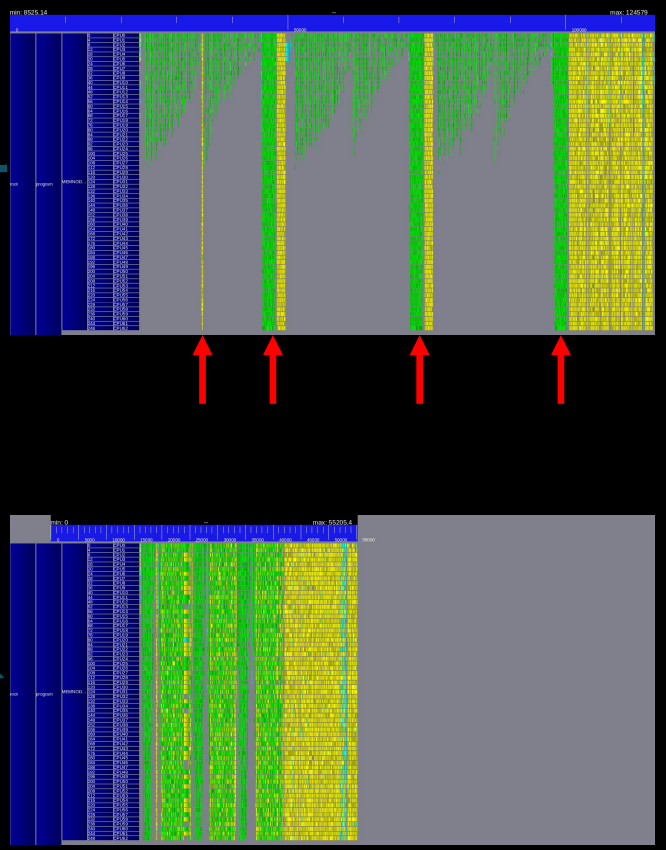
Data dependency driven programming abounds

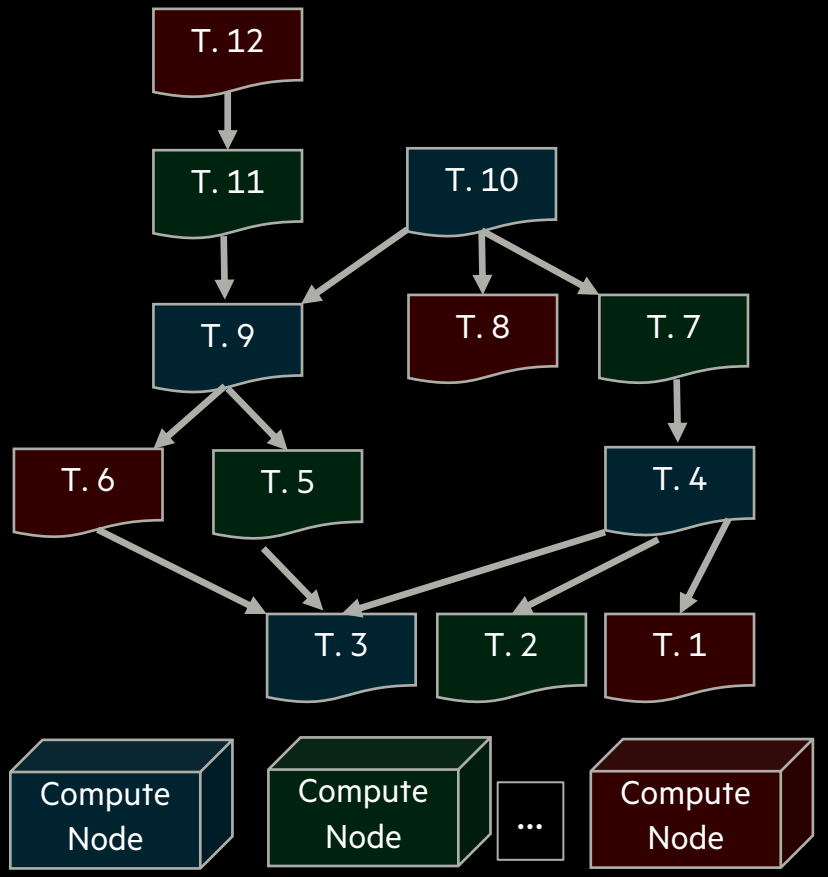
- At workflow level, explicit
  - Rise of heterogeneous coupled applications
    - Analytics, Systems Biology, Live “Big Data” processing
- In HPC
  - Parallel File System as backbone of implicit workflows
    - Simulation-analysis coupling, checkpointing, archiving
    - Coupler frameworks/middlewares
- In Programming Environment
  - Distributed tasking: HPX, PaRSEC, Legion, swift-lang, (StarPU), …
  - On-node tasking: OMP tasking, StarPU; pthreads, ArgoBots, UPC
    - Often lacking data locality information and data movement cost metrics
  - Functional paradigms entering mainstream languages (C++xx)
  - IO abstractions: Dataspaces, H5FDDSM, ADIOS2
- In Hardware abstraction
  - Dataflow architectures

DATA DRIVEN TASKING 101

- Decompose program into tasks that are coupled by input-output relations
  - A directed graph, tasks as nodes, data as arc labels, δ^+, δ^- as outputs/inputs
  - Program execution:
    - Marking of initial tasks as ‘ready to run’
    - Executing (some) ‘ready-to-run’ tasks
    - Marking successors of completed tasks as ‘ready-to-run’
  - Acyclic in purely functional programming, cyclic with bounded number of cycle repeats for terminating non-pure programs
- Example: $\boldsymbol{b = Ax}$ decomposes into $\boldsymbol{2 + m}$ tasks:
  - T_0 : Scatter rows of $\mathbf{A}$
  - T_1 … T_m: $\boldsymbol{m}$ scalar product tasks $\boldsymbol{A_i.x}$
  - T_{m+1}: gather results into b

WHY?




T. 12
T. 11
T. 10
T. 9
T. 8
T. 7
T. 6
T. 5
T. 4
T. 3
T. 2
T. 1
Compute Node
Compute Node
...
Compute Node

SCHEDULING?

It's a kind of VRP, but then again not

Vehicle routing model
Route code along data

- Data immutable
  - Explicit duplication operations allowed
- Hypergraph in bipartite representation
  - "substances" (data objects)
  - "reactions" (functional transformations)
    - A subset of transformations: data movement
- Transformation cost function (energy, time, …)
  - Looks a lot like a Petri net

Job shop Scheduling:
Assign tuples of data and code to compute resources

- Ordering constraints
- Machine-dependent execution times
- Data handling implicit in
  - Machine-dependent setup times
  - Sequence-dependent setup times
  - Reconfigurable machines: Data hierarchy access

Packing:
Handling multiple concurrent workflows

- Inside one problem instance
  - 'Machines' have setup times, amortized cost
  - Competition for resources
- Across workflows
  - Non-cooperating users
  - Different/contradicting objectives (makespan, energy, …)
- In time, on resources, partially splittable
- Often online

WHERE'S THE INSTANCE DATA?

System Monitoring

- Too coarse or too fine grained
- Based on hardware/software parameters that often are not suitable for a-priory models
- Often cannot be attributed to **tasks**
- Congestion vs. nominal data

Feedback profiling

- Not automatic
- Post-mortem
- Resulting models not sufficiently data-dependent

Machine models

- Nominal behavior/stochastic data
- Usable only at compiler or HPC workload manager level
- Very complex for modern large systems

Building a middleware (dataplane) that operates at application-defined object level permits

- data location awareness
- measuring at object level
- Compiler/application/workflow level optimal scheduling

MAESTRO
DATA ORCHESTRATION

https://www.maestro-data.eu/

OUTLOOK

- HPC is a great target for operations research techniques
  - Hardware and software system
  - Interconnects
  - Programming level

- Computational models disrupted
  - Data centricity
  - Heterogeneity up to the ‘Cloud-to-Edge’ level

- Many well-contained optimization problems and some extremely general ones

- After Exascale there’ll be Zetascale, so there is no shortage of scalable problem instances

… and I’ve not even talked about **using** mathematical programming on HPC systems

THANK YOU

uhaus@hpe.com