Software in the sixth Epoch of Distributed Computing

Tim Mattson Human Learning Group

Five Epochs of Distributed Computing*: the first three

Epoch starting date	Defining limitations	Application	Interaction time and Network performance	Capability
First 1970	Rare connections to expensive computers	FTP, telnet, email	100 ms Low bandwidth High latency	People to computers
Second 1984	I/O wall, disks can't keep up	RPC, Client Server	10 ms 10 mbps	Computer to computer
Third 1990	Networking wall	MPP HPC, three-tier datacenter networks	1 ms 100 mbs → 1 Gbs	Services to services People to static data

*The five Epochs of distributed computing, Amin Vahdat of Google: SIGCOMM Lifetime achievement award keynote, 2020. https://www.youtube.com/watch?v=27zuReojDVw

FTP: File Transfer Protocol, MPP: Massively parallel processor, RPC: Remote Procedure Call

The Eight Fallacies of Distributed Computing

(Peter Deutsch of Sun Microsystems, 1994 … item 8 added in 1997 by James Gosling)

Essentially everyone, when they first build a distributed application, makes the following eight assumptions. All prove to be false in the long run and all cause *big* trouble and *painful* learning experiences.

1. The network is reliable
2. Latency is zero
3. Bandwidth is infinite
4. The network is secure
5. Topology doesn't change
6. There is one administrator
7. Transport cost is zero
8. The network is homogeneous

https://en.wikipedia.org/wiki/Fallacies_of_distributed_computing

The Eight Fallacies of Distributed Computing

(Peter Deutsch of Sun Microsystems, 1994 … item 8 added in 1997 by James Gosling)

Essentially everyone, when they first build a distributed application, makes the following eight assumptions. All prove to be false in the long run and all cause *big* trouble and *painful* learning experiences.

1. The network is reliable
2. Latency is low and fixed
3. Bandwidth is high and fixed
4. The network is secure
5. Topology doesn't change
6. There is one administrator
7. Transport cost is negligible
8. The network is homogeneous

https://en.wikipedia.org/wiki/Fallacies_of_distributed_computing

The Eight Fallacies of Distributed Computing

(Peter Deutsch of Sun Microsystems, 1994 … item 8 added in 1997 by James Gosling)

Essentially everyone, when they first build a distributed application, makes the following eight assumptions. All prove to be false in the long run and all cause *big* trouble and *painful* learning experiences.

Data Center

1. ~~The network is reliable~~
2. ~~Latency is low and fixed~~
3. ~~Bandwidth is high and fixed~~
4. ~~The network is secure~~
5. ~~Topology doesn't change~~
6. ~~There is one administrator~~
7. ~~Transport cost is negligible~~
8. ~~The network is homogeneous~~

HPC Cluster

1. ✓ The network is reliable
2. ✓ Latency is low and fixed
3. ✓ Bandwidth is high and fixed
4. ✓ The network is secure
5. ✓ Topology doesn't change
6. ✓ There is one administrator
7. ~~Transport cost is negligible~~
8. ✓ The network is homogeneous

https://en.wikipedia.org/wiki/Fallacies_of_distributed_computing

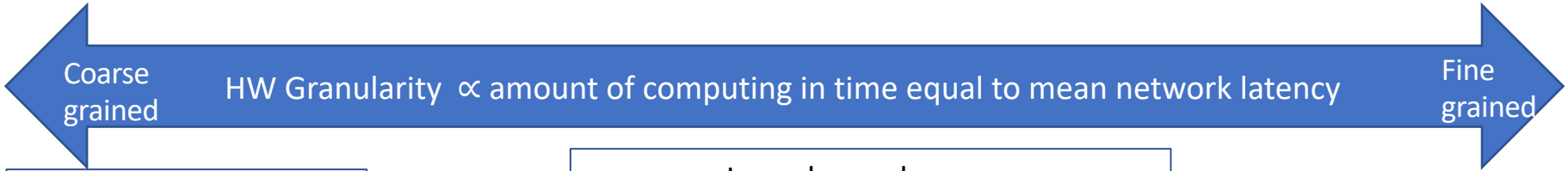
Where can we run HPC applications?

Data Center

1. ~~The network is reliable~~
2. ~~Latency is low and fixed~~
3. ~~Bandwidth is high and fixed~~
4. ~~The network is secure~~
5. ~~Topology doesn't change~~
6. ~~There is one administrator~~
7. ~~Transport cost is negligible~~
8. ~~The network is homogeneous~~

HPC Cluster

1. The network is reliable
2. Latency is low and fixed
3. Bandwidth is high and fixed
4. The network is secure
5. Topology doesn't change
6. There is one administrator
7. ~~Transport cost is negligible~~
8. The network is homogeneous



Embarrassingly Parallel

Loosely synchronous

Asynchronous

Synchronous

Five Epochs of Distributed Computing*

Epoch starting date	Defining limitations	Application	Interaction time and Network performance	Capability
First 1970	Rare connections to expensive computers	FTP, telnet, email	100 ms Low bandwidth High latency	People to computers
Second 1984	I/O wall, disks can't keep up	RPC, Client Server	10 ms 10 mbps	Computer to computer
Third 1990	Networking wall	MPP HPC, three-tier datacenter networks	1 ms 100 mbs → 1 Gbs	Services to services People to static data
Fourth 2000	Dennard scaling wall … per core plateau	Web search, planet-scale services	100 μs 10 Gbps flash	People to people People to interactive results

*The five Epochs of distributed computing, Amin Vahdat of Google: SIGCOMM Lifetime achievement award keynote, 2020.

FTP: File Transfer Protocol, MPP: Massively parallel processor, RPC: Remote Procedure Call

The cloud takes over

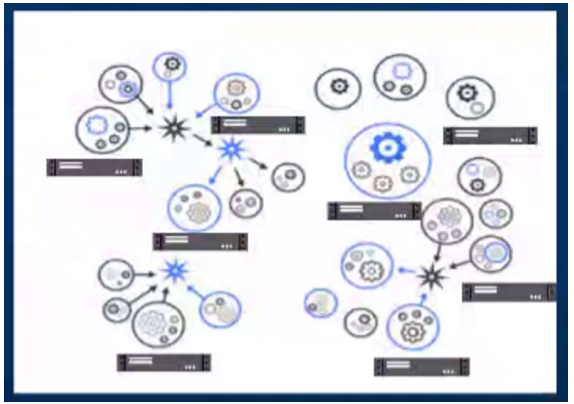
- On premises data centers give way to distributed resources “in the cloud”.
- Starts with virtual machines (infrastructure as a service) and grows into a whole new architecture for software … microservices.

The old way: Monolithic applications

Single program composed of many functions interacting through memory/messages.

Latencies O(microseconds) or less

The new, cool way: microservices

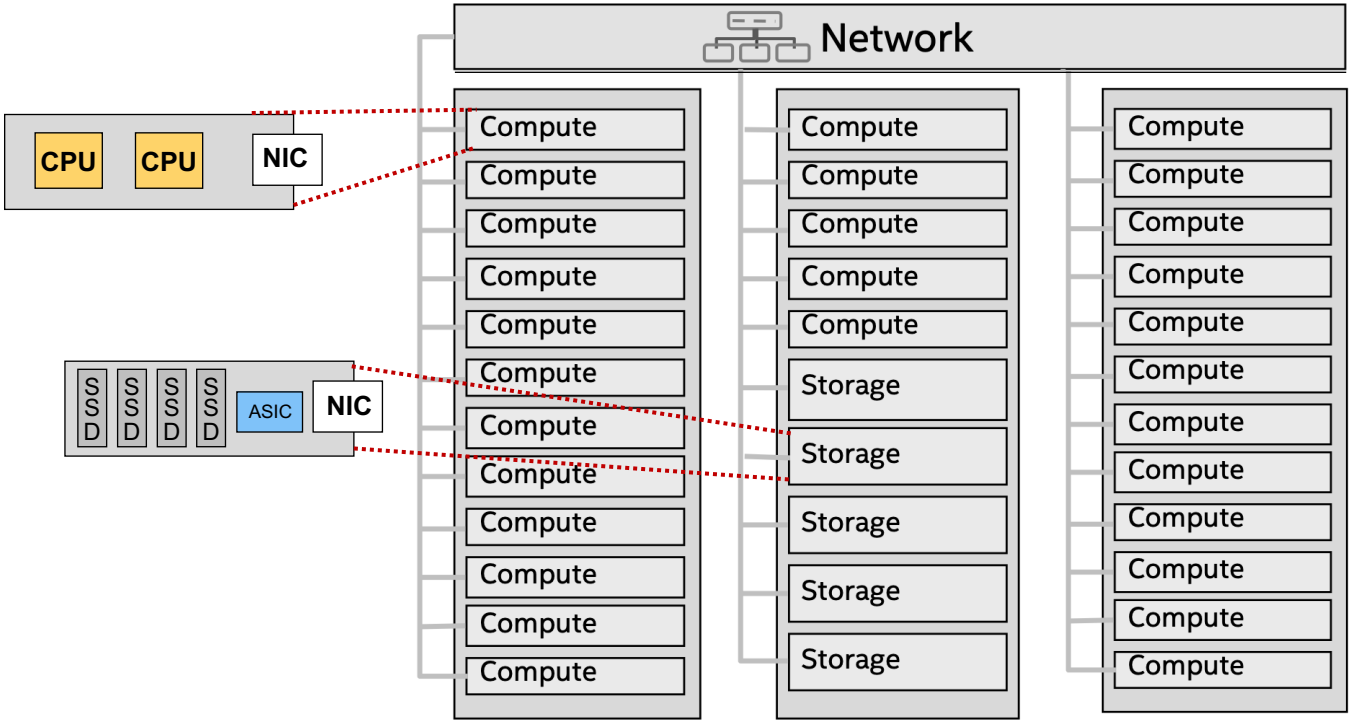


Many small independent programs interacting through remoter procedure calls (RPC).

Latencies O(milliseconds) to O(seconds) … the tail latency problem

Hardware in a cloud data center

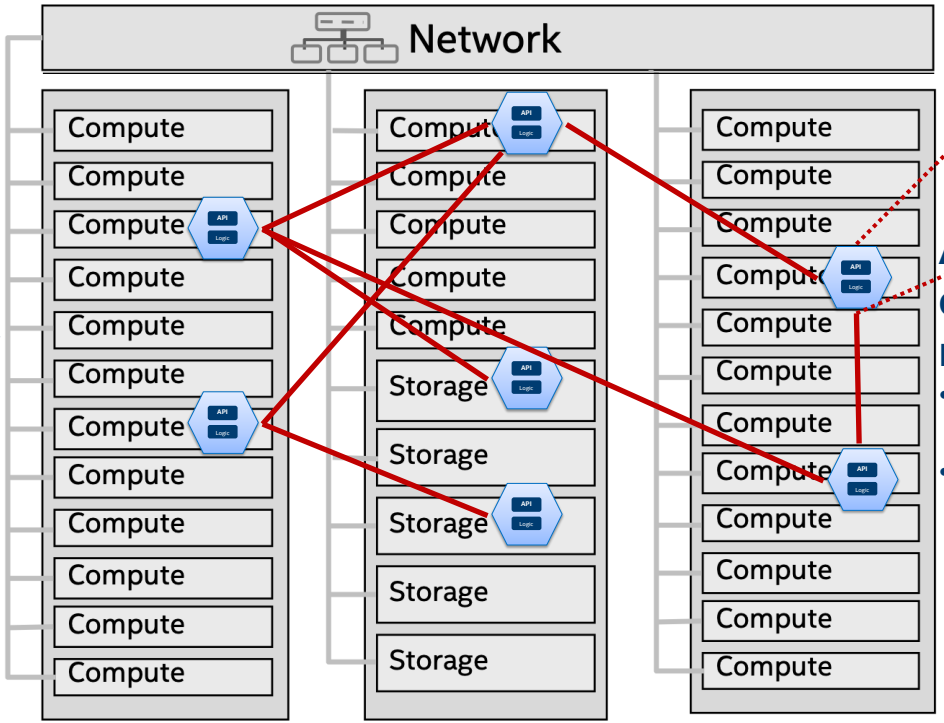
- The unit of hardware replication is the server node with memory, I/O, and CPUs.



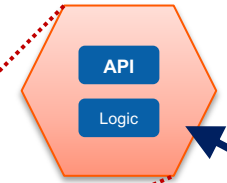
NIC: Network Interface Controller

Hardware in a cloud data center

- Applications composed of many (O(100)) small independent programs … ship functions as needed. They interact through remote procedure calls (RPCs).



Microservice



Apps@scale as collections of microservices

- Discrete units of functionality
- Continuous integration
- Continuous upgrades
- Resilient workflows

A Program embedded inside a microservice

- Tightly coupled parallelism.
- Interacts directly with the Hardware

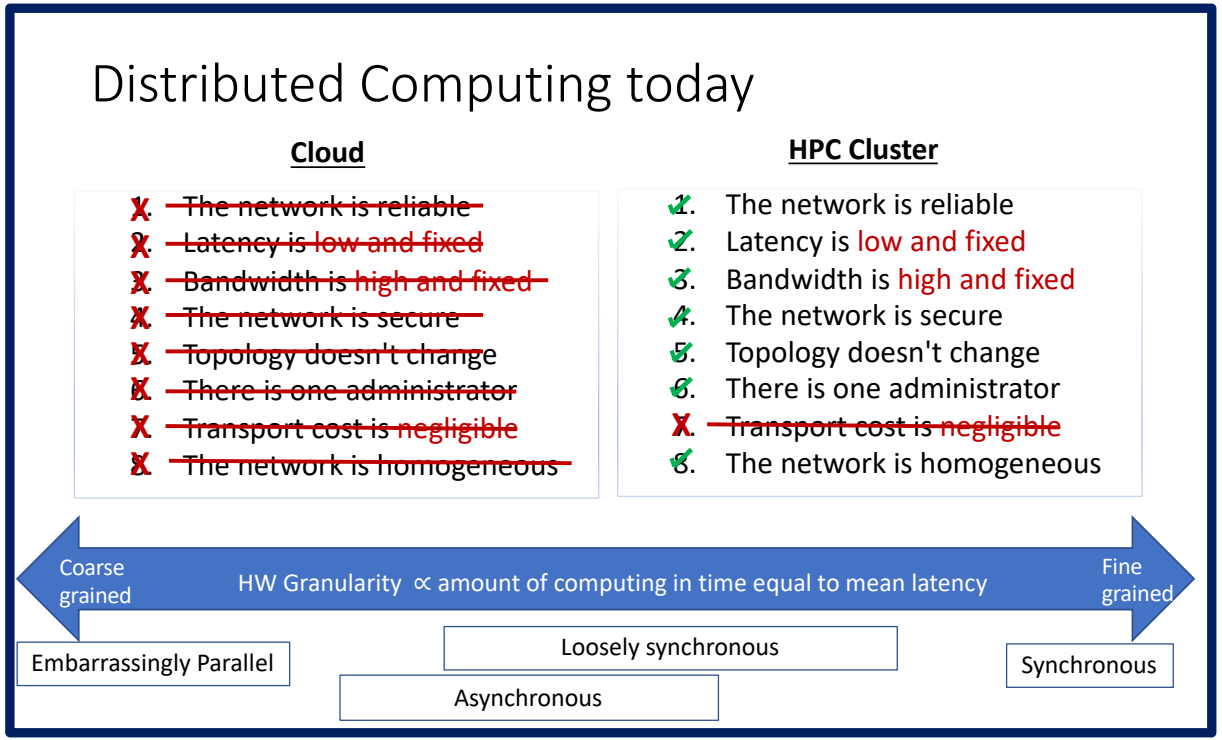
NIC: Network Interface Controller

Programming Distributed computers

There is a clean split between applications that run in the cloud and those that need a dedicated HPC cluster.

This is reflected in the programming models used:

- Cloud: Remote Procedure Call (RPC), distributed object store distinct from tasks, execution flows as task graphs for Function as a service. Heavy use of microservices.
- HPC Cluster: SPMD design pattern with MPI … also PGAS with SHMEM.



The three domains of parallel programming

Platform*	Laptop or server	HPC Cluster	Cloud
Execution Agent	Threads	Processes	Microservices
Memory	Single Address Space	Distributed memory, local memory owned by individual processes	Distributed object store (in memory) backed by a persistent storage system
Typical Execution Pattern	Fork-join	SPMD	Event driven tasks, FaaS, and Actors

Laptop/server and cluster models work well together.

An impenetrable wall separates them from the cloud-native world

Five Epochs of Distributed Computing*

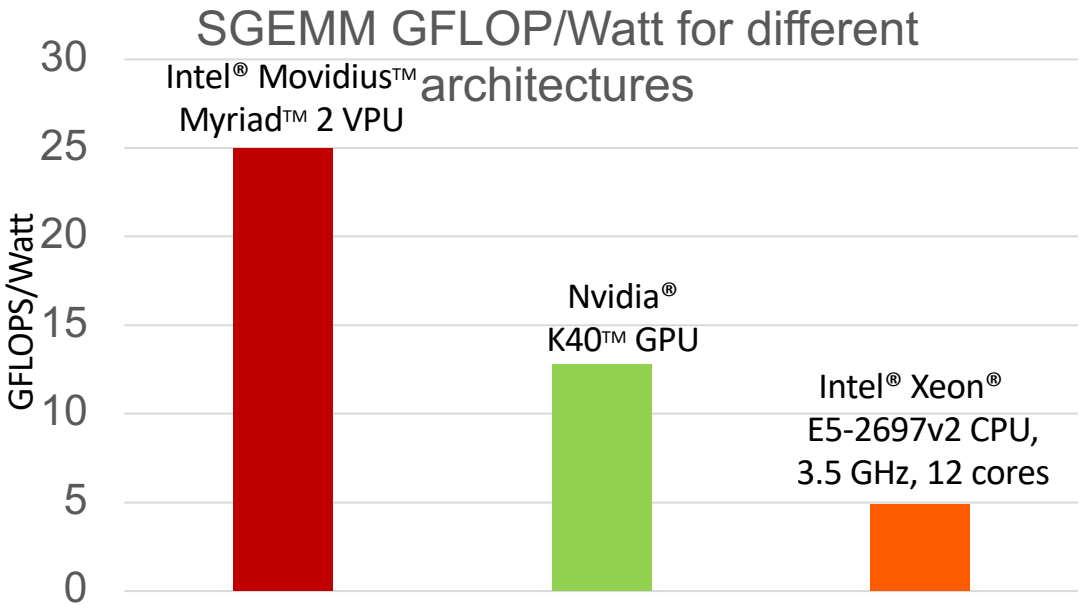
Epoch starting date	Defining limitations	Application	Interaction time and Network performance	Capability
First 1970	Rare connections to expensive computers	FTP, telnet, email	100 ms Low bandwidth High latency	People to computers
Second 1984	I/O wall, disks can't keep up	RPC, Client Server	10 ms 10 mbps	Computer to computer
Third 1990	Networking wall	MPP HPC, three-tier datacenter networks	1 ms 100 mbs → 1 Gbs	Services to services People to static data
Fourth 2000	Dennard scaling wall … per core plateau	Web search, planet-scale services	100 μs 10 Gbps flash	People to people People to interactive results
Fifth 2015	Per socket wall … accelerators take off	Machine Learning, data centric computing	10 μs 200 Gbps → 1 Tbps	People to insights

*The five Epochs of distributed computing, Amin Vahdat of Google: SIGCOMM Lifetime achievement award keynote, 2020.

FTP: File Transfer Protocol, MPP: Massively parallel processor, RPC: Remote Procedure Call

If you care about power, the world is heterogeneous?

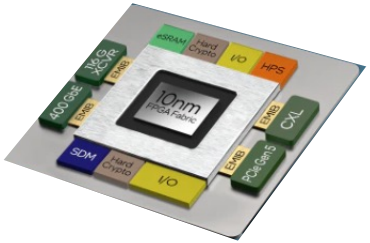
Specialized processors doing operations suited to their architecture are more efficient than general purpose processors.



Hence, future systems will be increasingly heterogeneous … GPUs, CPUs, FPGAs, and a wide range of accelerators

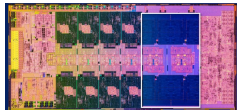
Source: Suyash Bakshi and Lennart Johnsson, "A Highly Efficient SGEMM Implementation using DMA on the Intel/Movidius Myriad-2. IEEE International Symposium on Computer Architecture and High Performance Computing, 2020

A New Golden Age for Computer Architecture

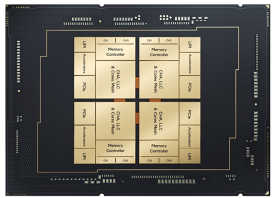


Intel® Agilex™ FPGAs

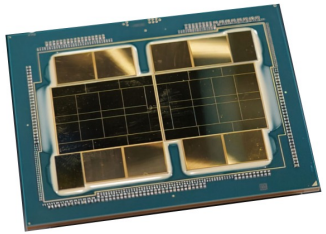
Habana® Gaudi® 2
deep learning accelerator



13th gen Intel® Core™ CPU
Hybrid Architecture with 16 efficiency cores and 8 performance cores + integrated GPU



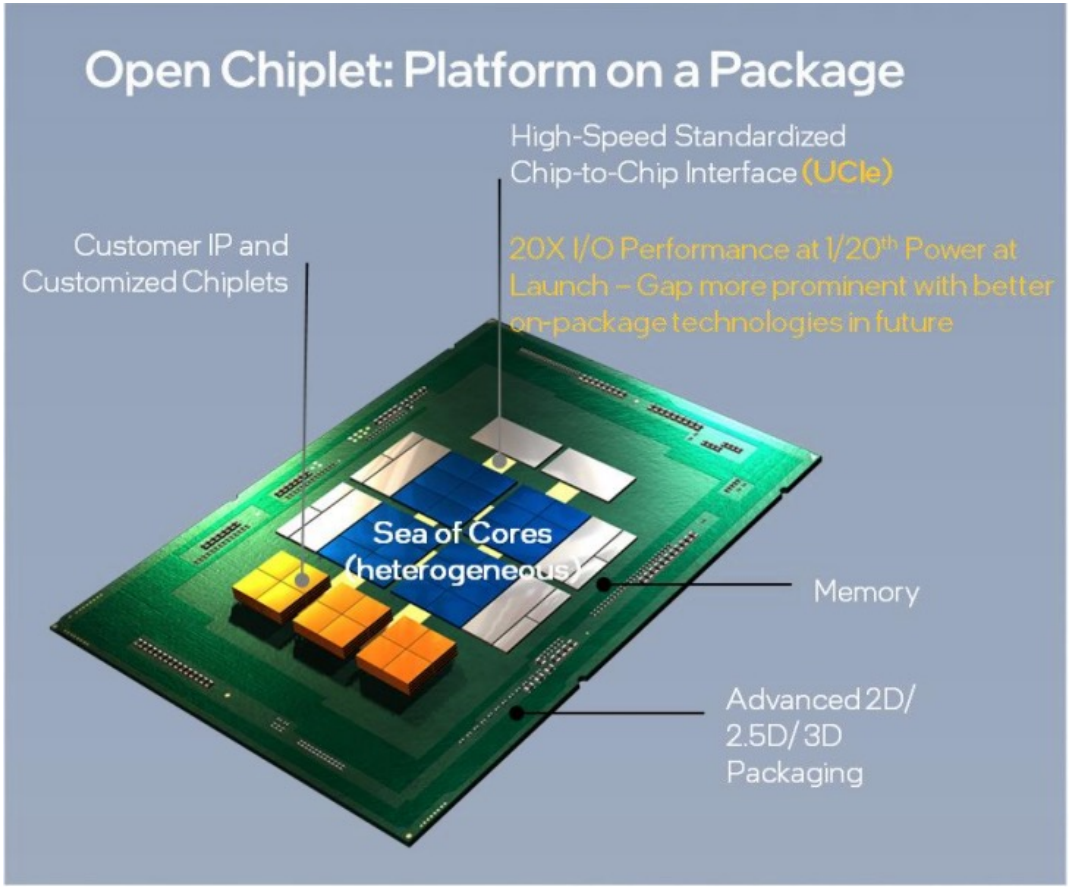
4th Gen Intel® Xeon® CPU with 56 cores and Novel On-Die Accelerators



Intel® Xe HPC GPU

Image sources: Intel

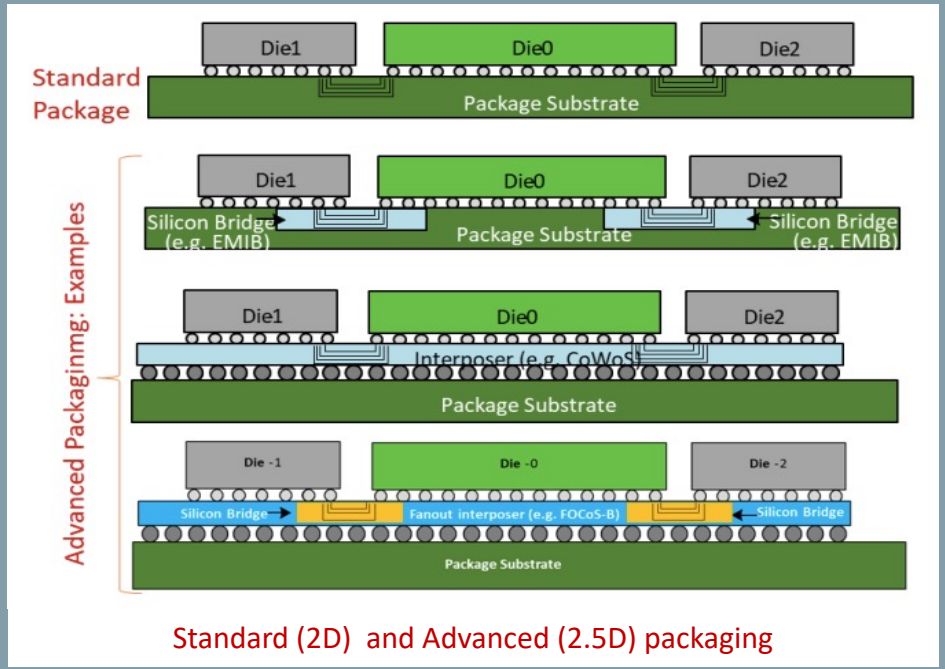
A New Golden Age for Computer Architecture



Source: uciexpress.org white paper,

- Chiplet-based architectures … building a package placed in a socket composed of distinct little chips (the "chiplet").
- Connected by high speed in package interconnects … lets chiplets from multiple fabs fit into one package.
- The Universal Chiplet Interconnect Express effort defines a standard for how to connect chiplets.
- The result … multi-chiplet packages in a socket with heterogeneous devices from multiple vendors.

A New Golden Age for Computer Architecture



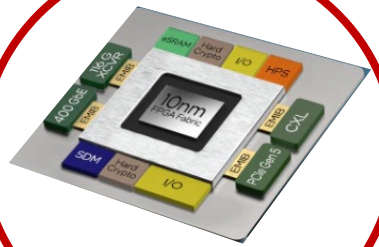
Standard (2D) and Advanced (2.5D) packaging

- Chiplet-based architectures … building a package placed in a socket composed of distinct little chips (the "chiplet").
- Connected by high speed in package interconnects … lets chiplets from multiple fabs fit into one package.
- The Universal Chiplet Interconnect Express effort defines a standard for how to connect chiplets.
- The result … multi-chiplet packages in a socket with heterogeneous devices from multiple vendors.

Membership includes Intel, Nvidia, Samsung, Qualcomm, Alibaba, Google, Microsoft, Meta, Arm …

Source: uciexpress.org white paper,

A New Golden Age for Computer Architecture



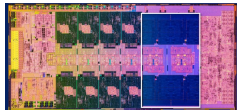
Intel® Agilex™ FPGAs

Habana® Gaudi® 2
deep learning accelerator

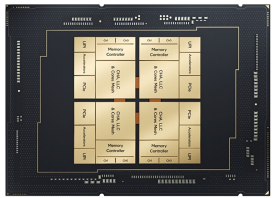
UCIe™
Universal Chiplet
Interconnect Express

Multi-chiplet packages with components from multiple vendors

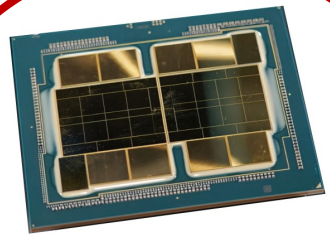
Many of Intel's products already use chiplet technology



13th gen Intel® Core™ CPU
Hybrid Architecture with 16 efficiency cores and 8 performance cores + integrated GPU



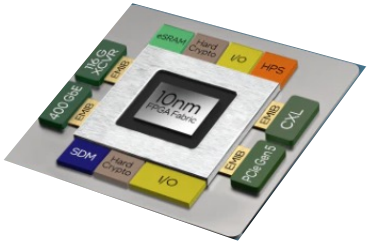
4th Gen Intel® Xeon® CPU with 56 cores and Novel On-Die Accelerators



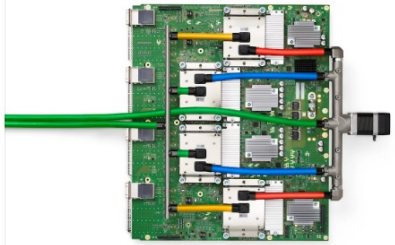
Intel® Xe HPC GPU

Image sources: Intel, TPU from https://cloud.google.com/tpu

A New Golden Age for Computer Architecture



Intel® Agilex™ FPGAs



Google® Tensor Processing Unit

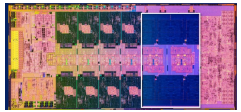
Habana® Gaudi® 2 deep learning accelerator

UCIe™ Universal Chiplet Interconnect Express

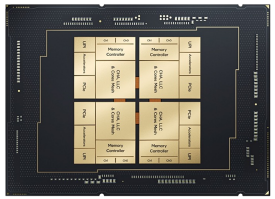
Multi-chiplet packages with components from multiple vendors

RISC-V®

arm



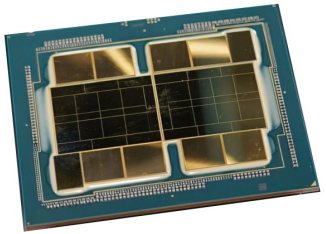
13th gen Intel® Core™ CPU Hybrid Architecture with 16 efficiency cores and 8 performance cores + integrated GPU



4th Gen Intel® Xeon® CPU with 56 cores and Novel On-Die Accelerators

AMD

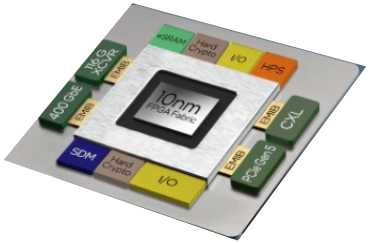
- CPUs
- Discreate GPU
- CPU + integrated GPU



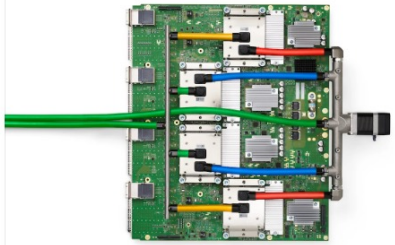
Intel® Xe HPC GPU

Image sources: Intel, TPU from https://cloud.google.com/tpu

A New Dark Age for Computer Programmers



Intel® Agilex™ FPGAs



Google® Tensor Processing Unit

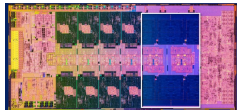
Habana® Gaudi® 2 deep learning accelerator

UCIe™ Universal Chiplet Interconnect Express

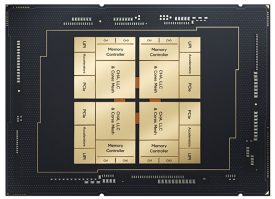
Multi-chiplet packages with components from multiple vendors

RISC-V®

arm



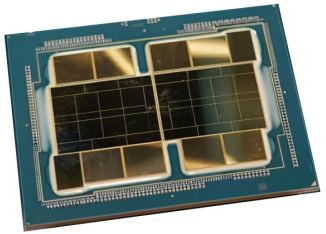
13th gen Intel® Core™ CPU Hybrid Architecture with 16 efficiency cores and 8 performance cores + integrated GPU



4th Gen Intel® Xeon® CPU with 56 cores and Novel On-Die Accelerators

AMD

- CPUs
- Discreate GPU
- CPU + integrated GPU

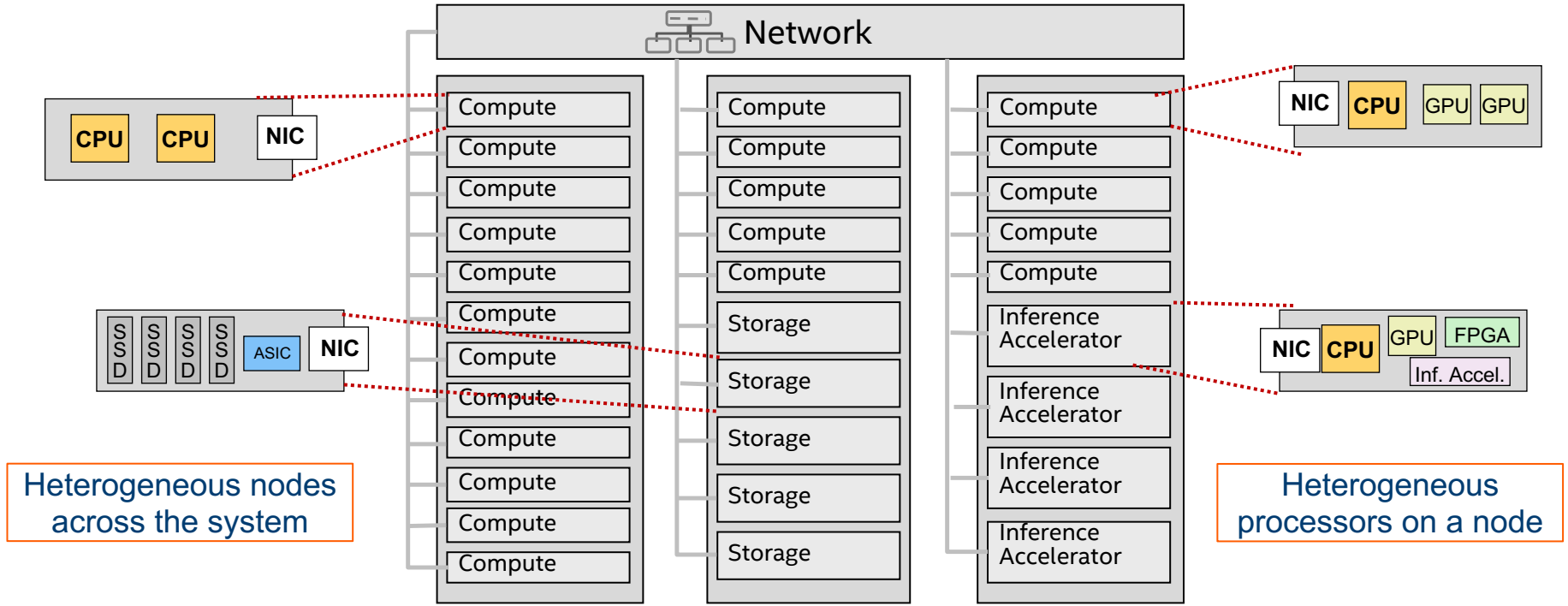


Intel® Xe HPC GPU

Image sources: Intel, TPU from https://cloud.google.com/tpu

Hardware in a cloud data center

- The unit of hardware replication is the server node with memory, I/O, processors, and accelerators.



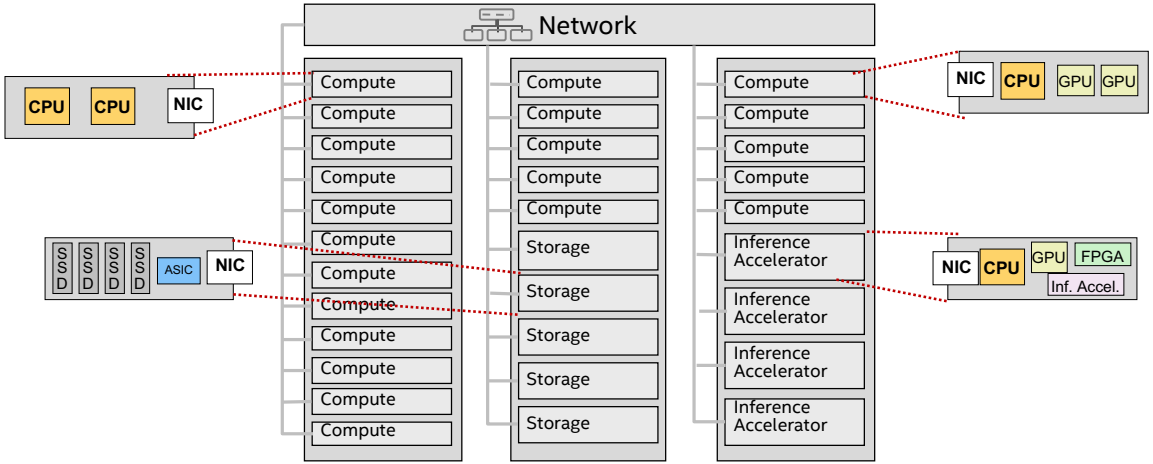
NIC: Network Interface Controller

… But there are problems in our cloudy paradise

Wasted Resources in the cloud

How resources are made available is associated with hardware on a node.

For example, if all the cores are allocated to customer VMs, memory on the node not used by the VMs is unavailable for other jobs. It is **stranded**.



Analysis of production traces from Azure, shows that stranding is the dominant source of memory waste. They found that up to 25% of DRAM at any time is stranded (i.e. wasted).*

A similar analysis$ of traces from Google's data centers found the average DRAM utilization was only 40%.

* "Pond: CXL based memory pooling system for cloud platforms" Huaicheng Lie, et. al. https://arxiv.org/abs/2203.00241

$ "Borg: the Next Generation", Muhammad Tirmazi, et. al., https://storage.googleapis.com/pub-tools-public-publication-data/pdf/5bf4ebfbf98ead7f6ee1552860fab88e75a5ed7e.pdf

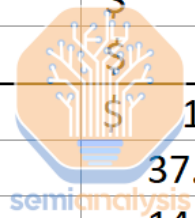
Cost break down for a typical server node

For Azure, DRAM can be 50% of server cost*

Other analyses finds it at 40% of the cost

Whether its 40% or 50% of total cost, it is still an expensive resource to waste

2x Intel Sapphire Rapids Server	
Component	AI Server
CPU	$ 1,850
8 GPU + 4 NVSwitch Baseboard	$ -
Memory	$ 3,930
Storage	$ 1,536
SmartNIC	$ 654
Chassis (Case, backplanes, cabling)	$ 395
Motherboard	$ 350
Cooling (Heatsinks+fans)	$ 275
Power Supply	$ 300
Assembly and Test	$ 495
Markup	$ 689
Total Cost	$ 10,474
DRAM BOM %	37.5%
NAND BOM %	14.7%
Memory BOM %	52.2%



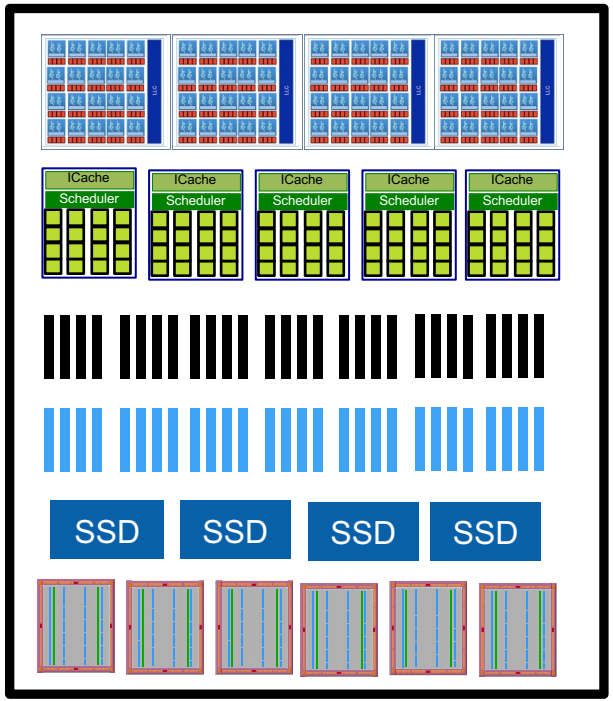
* “Pond: CXL based memory pooling system for cloud platforms” Huaicheng Lie, et. al. https://arxiv.org/abs/2203.00241

*Dylan Patel and Gerald Wong, May 29, 2023. https://www.semianalysis.com/p/ai-server-cost-analysis-memory-is

Disaggregated Computing for SW Defined Servers (SDS)

Consider a Rack composed of multiple pools

Dynamically compose across pools to match a software defined server to the workload



CPU pool

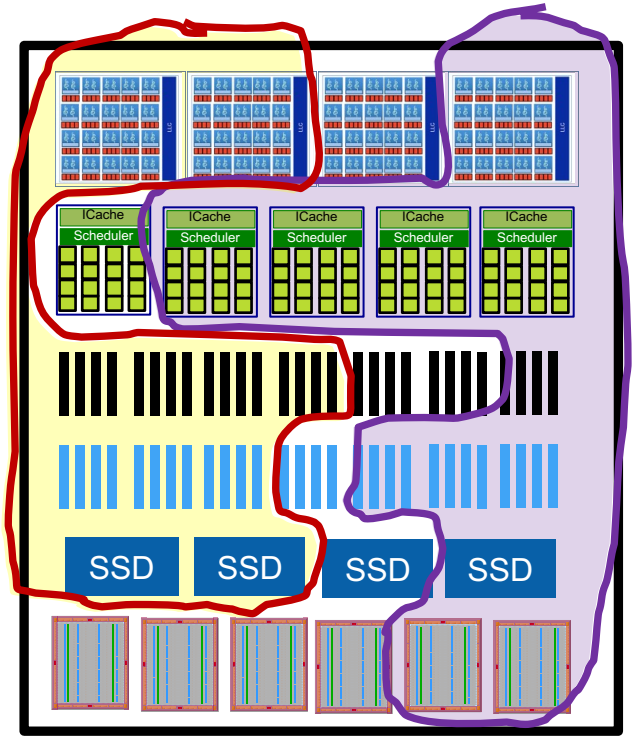
GPU pool

DRAM pool

NVRAM pool

SSD pool

FPGA pool

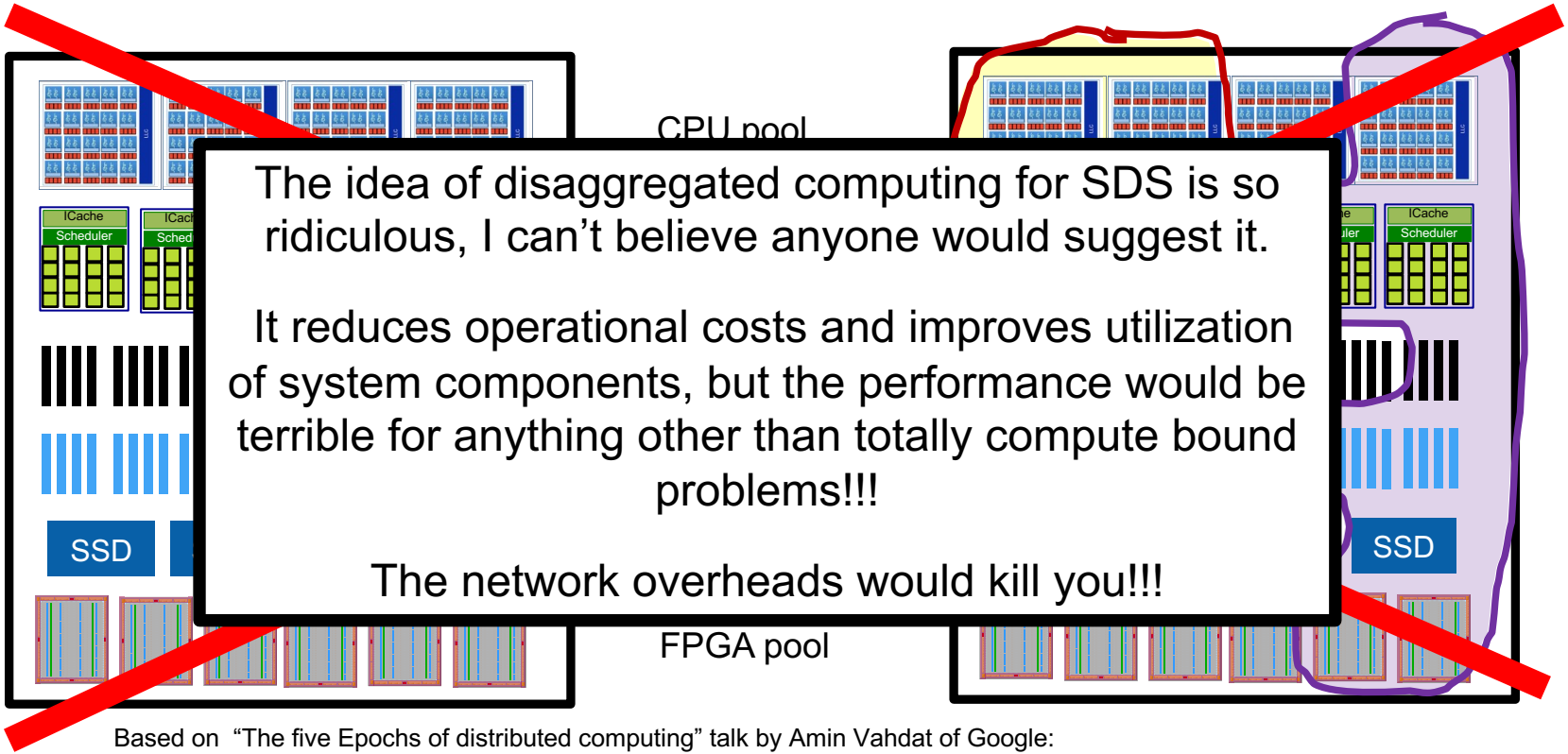


Based on "The five Epochs of distributed computing" talk by Amin Vahdat of Google:

Disaggregated Computing for SW Defined Servers (SDS)

Consider a Rack composed of multiple pools

Dynamically compose across pools to match a software defined server to the workload



The idea of disaggregated computing for SDS is so ridiculous, I can't believe anyone would suggest it.

It reduces operational costs and improves utilization of system components, but the performance would be terrible for anything other than totally compute bound problems!!!

The network overheads would kill you!!!

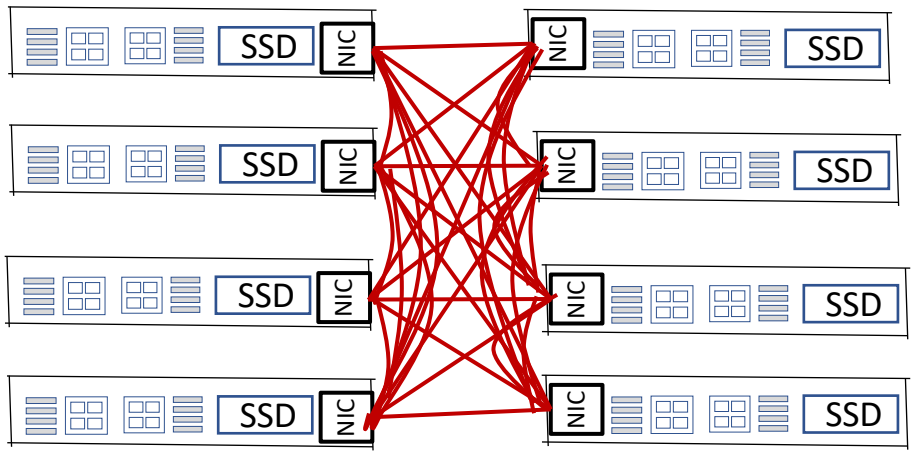
Based on "The five Epochs of distributed computing" talk by Amin Vahdat of Google:

The sixth Epoch of Distributed Computing

Epoch starting date	Defining limitations	Application	Interaction time and Network performance	Capability
First 1970	Rare connections to expensive computers	FTP, telnet, email	100 ms Low bandwidth High latency	People to computers
Second 1984	I/O wall, disks can't keep up	RPC, Client Server	10 ms 10 mbps	Computer to computer
Third 1990	Networking wall	MPP HPC, three-tier datacenter networks	1 ms 100 mbs → 1 Gbs	Services to services People to static data
Fourth 2000	Dennard scaling wall … per core plateau	Web search, planet-scale services	100 μs 10 Gbps flash	People to people People to interactive results
Fifth 2015	Per socket wall … accelerators take off	Machine Learning, data centric computing	10 μs 200 Gbps → 1 Tbps	People to insights
Sixth 2025	Speed of light	Dynamic, real-time AI from data-center to the edge with SDE*	100 ns 10 Tbs	People to experiences

* SDE: Software defined Everything, i.e. software defined networking, software defined infrastructure, software defined servers ... All at the same time … to dynamically construct systems to meet the needs of workloads.

Networking technology… replace generic data center network with a cluster of cliques



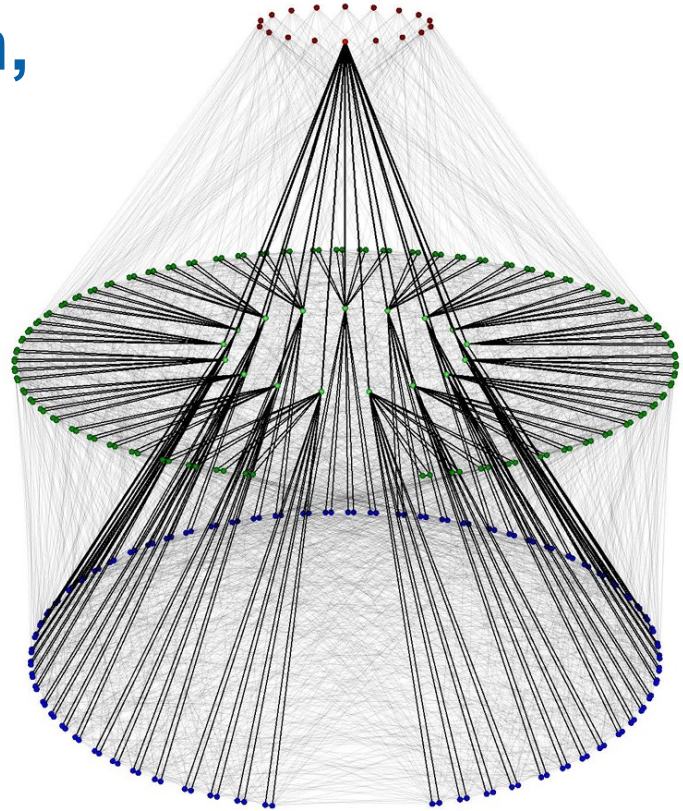
A clique: A graph where every vertex is connected to every other vertex

A Clique: a network of diameter one with $O(\frac{1}{4}N^2)$ bisection bandwidth

Combine with next generation optical networks to hit latencies close to DRAM latencies (100 ns)

Future: scale networks across the data center with next generation, point-to-point optical networks

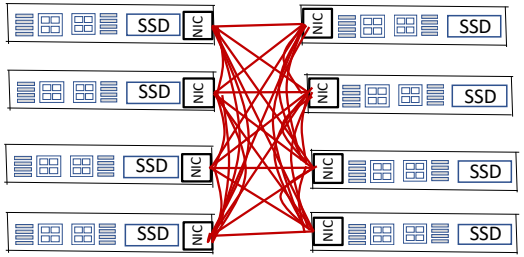
PolarFly network: Over 150 nodes connect by a network of diameter 2 … that is, any pair of nodes can be reached by two hops on the network



PolarFly: A cost-effective and Flexible Low-Diameter Toploty, K. Lakhotia, M. Besta, L. Monroe, K. Isharm, P. Iff, T. Hoefler, and F. Petrini, arXiv:2208.01695v4, 2023

Latencies every engineer should know …

L1 cache reference 1.5 ns
L2 cache reference 5 ns
Branch misprediction 6 ns
Uncontended mutex lock/unlock 20 ns
L3 cache reference 25 ns
Main memory reference 100 ns
“Far memory”/Fast NVM reference 1,000 ns (1us)
Read 1 MB sequentially from memory 12,000 ns (12 us)
SSD Random Read 100,000 ns (100 us)
Read 1 MB bytes sequentially from SSD 500,000 ns (500 us)
Read 1 MB sequentially from 10Gbps network 1,000,000 ns (1 ms)
Read 1 MB sequentially from disk 10,000,000 ns (10 ms)
Disk seek 10,000,000 ns (10 ms)
Send packet California→Netherlands→California (150 ms)



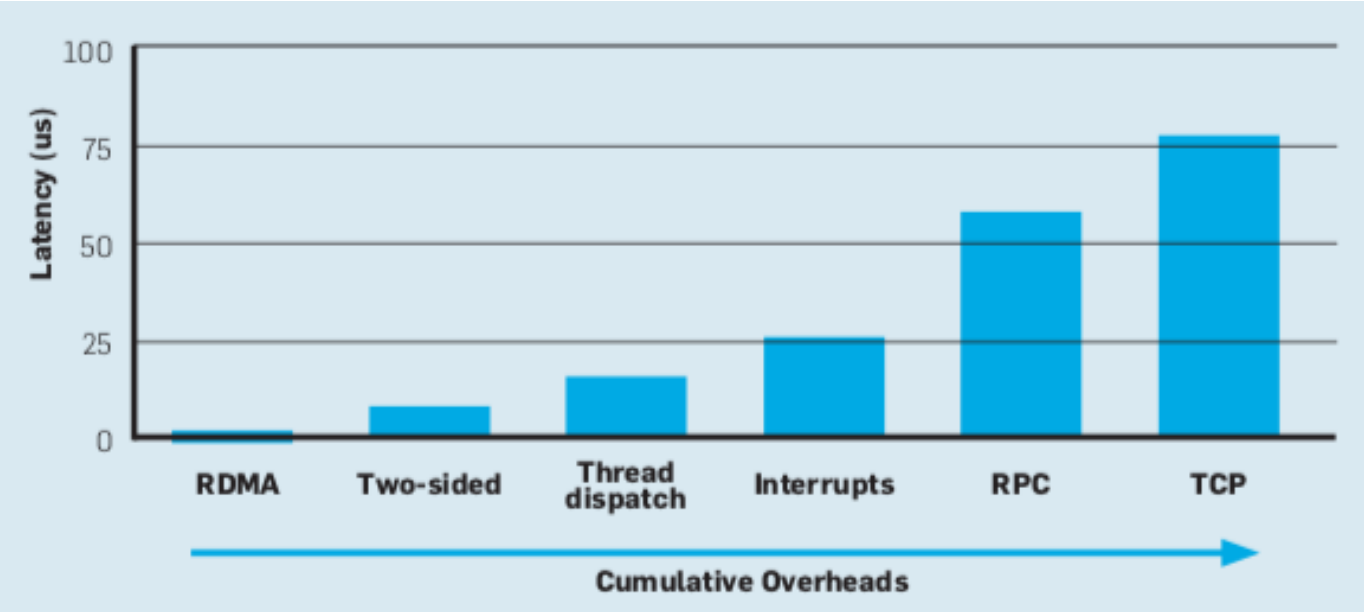
A cluster of nodes with a Clique network topology and low latency optical network…

Yields one hop network latencies on par with DRAM access latencies.

Source: **The Datacenter as a Computer: Designing Warehouse-Scale Machines**, Luiz Andre Barroso, Urs Holzle, Parthasarathy Ranganathan, 3rd edition, Morgan & Claypool, 2019.

Take out the big stuff & you're left with lots of µs overheads

All those SW overheads add up … like bricks that combine to build a networking-wall …
turning a 2 µs network into a 100 µs network…



Computer Scientists need to rethink system SW stacks to minimize latencies …
fast RDMA, reduce sync contention, low latency interrupt handlers, and more ….
All to hit O(µs) latencies.

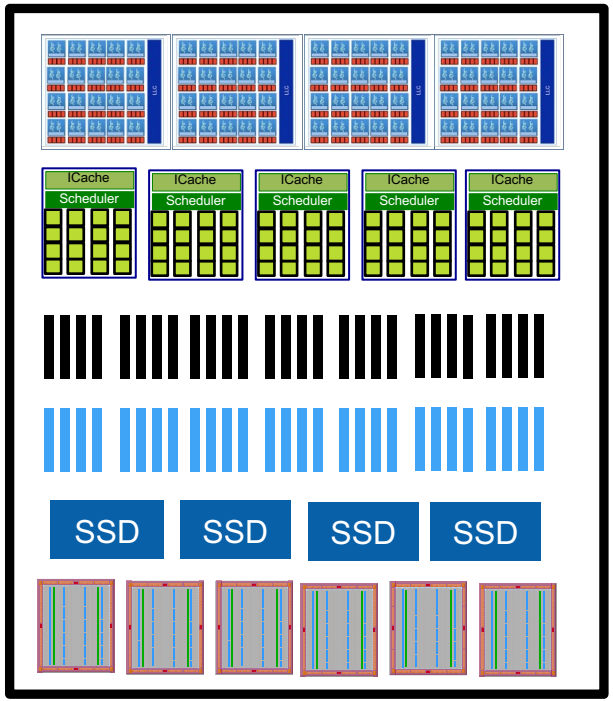
Source: Fig 1 from "Attack of the Killer Microseconds", Barroso, Marty, Patterson, and Ranganathan, Comm ACM vol 60, # 4, p. 48, 2017

Disaggregated Computing for SW Defined Servers (SDS):

This idea works in the sixth Epoch

Consider a Rack composed of multiple pools

Dynamically compose across pools to match a software defined server to the workload



CPU pool

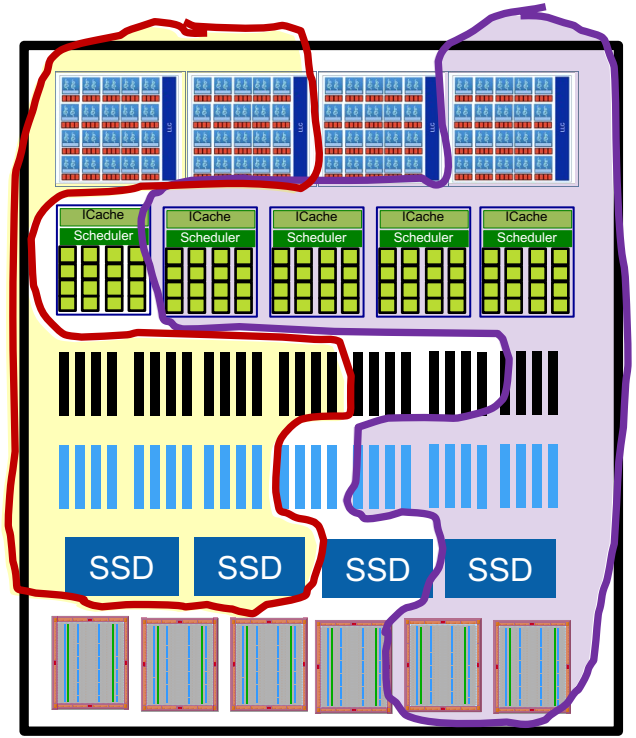
GPU pool

DRAM pool

NVRAM pool

SSD pool

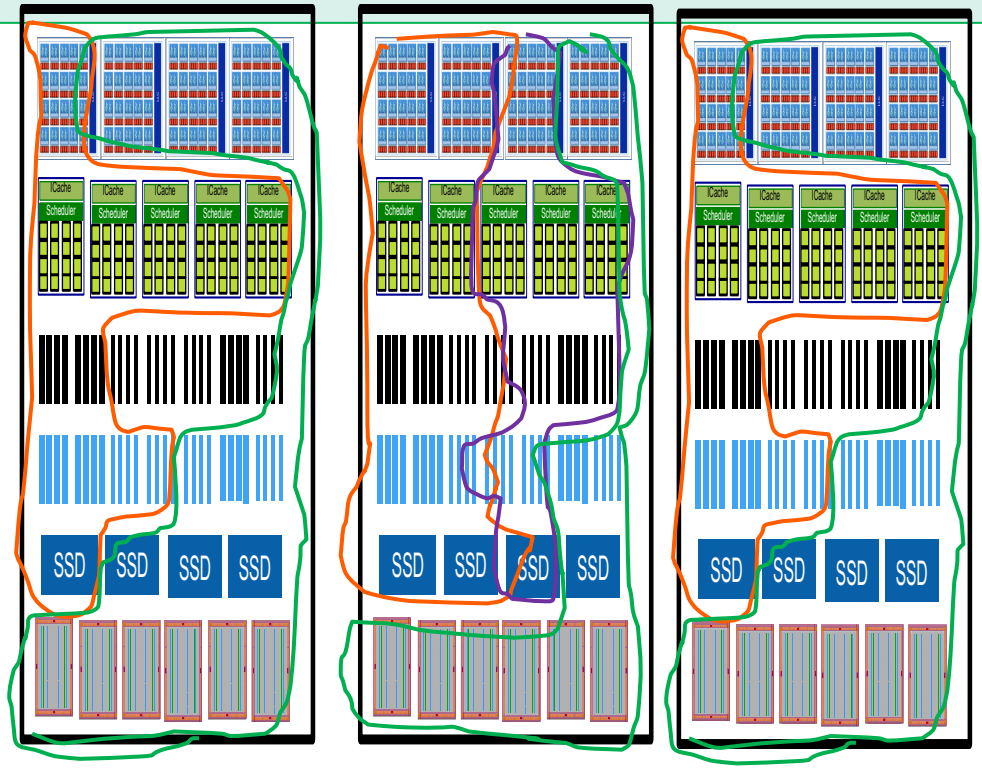
FPGA pool



Based on "The five Epochs of distributed computing" talk by Amin Vahdat of Google:

SW Defined clusters of SW defined Servers

Low latency, high bandwidth network between cliques



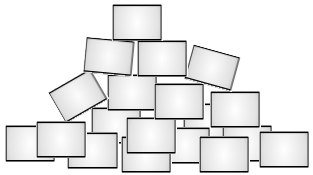
. . .

- Dynamic … changing from one job to the next.
- SW defined severs composed of heterogeneous components
- Dynamically composed into a cluster
- Integrated over a 5G network to devices (and people) at the edge

The three domains of parallel programming

Platform*	Laptop or server	HPC Cluster	Cloud
Execution Agent	Threads	Processes	Microservices
Memory	Single Address Space	Distributed memory, local memory owned by individual processes	Distributed object store (in memory) backed by a persistent storage system
Typical Execution Pattern	Fork-join	SPMD	Event driven tasks, FaaS, and Actors

Advances in networking technology plus low-overhead software stacks optimized to reduce tail-latency will shatter the wall between cloud computing and t5he other domains



Six Epochs of Distributed Computing … SW implications

Epoch starting date	Defining limitations	Application	Interaction time and Network performance	Capability
First 1970	Rare connections to expensive computers	FTP, telnet, email		People to computers
Second 1984	I/O wall, disks can't keep up	RPC, Client Server		Computer to computer
Third 1990		MPP HPC, three-tier datacenter networks		
Fourth 2000	Dennard Scaling Wall … per core plateau	Web search, planet-scale services	100 μs	
Fifth 2015	Per socket wall … accelerators take off	Machine Learning, data centric computing		
Sixth 2025	Speed of light	Dynamic AI, integrated f data-center to the edge.		

Concurrency: none

Concurrency: event driven

Concurrency: Multithreaded for latency hiding

Concurrency: Multithreaded for performance

Concurrency: Multi-server, multi-threaded

Concurrency: real-time, hierarchical & heterogenous

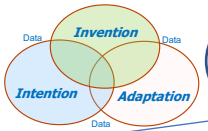
The five Epochs of distributed computing, Amin Vahdat of Google: SIGCOMM Lifetime achievement award keynote, 2020.

Writing Parallel Distributed Applications

Network technology evolution:
- Lower and more predictable latencies
- Erase distinctions between HPC clusters & the cloud

In response … we must support:
- One code base → multiple execution models

We call this ***machine programming***



- Application task-groups → microservices
- Data structures → distributed object store
- Durable store: Persistent cloud store (e.g. S3)

Cloud Native HPC

- Application task-groups → processes
- Data structures → in process memory
- Durable Store: Cluster file system

HPC Cluster

- Applications task-groups → threads
- Data structures → process heap
- Durable store: local file system

Laptop/server

Conclusion

- We are on the threshold of the sixth Epoch of distributed computing.
- This will shatter the wall separating traditional HPC from the cloud.
- As optical networks become the norm, much of what we do in HPC will be driven to the cloud.
- The issue for the HPC community … will our software be ready for this brave new world?