IOWA

CS5630

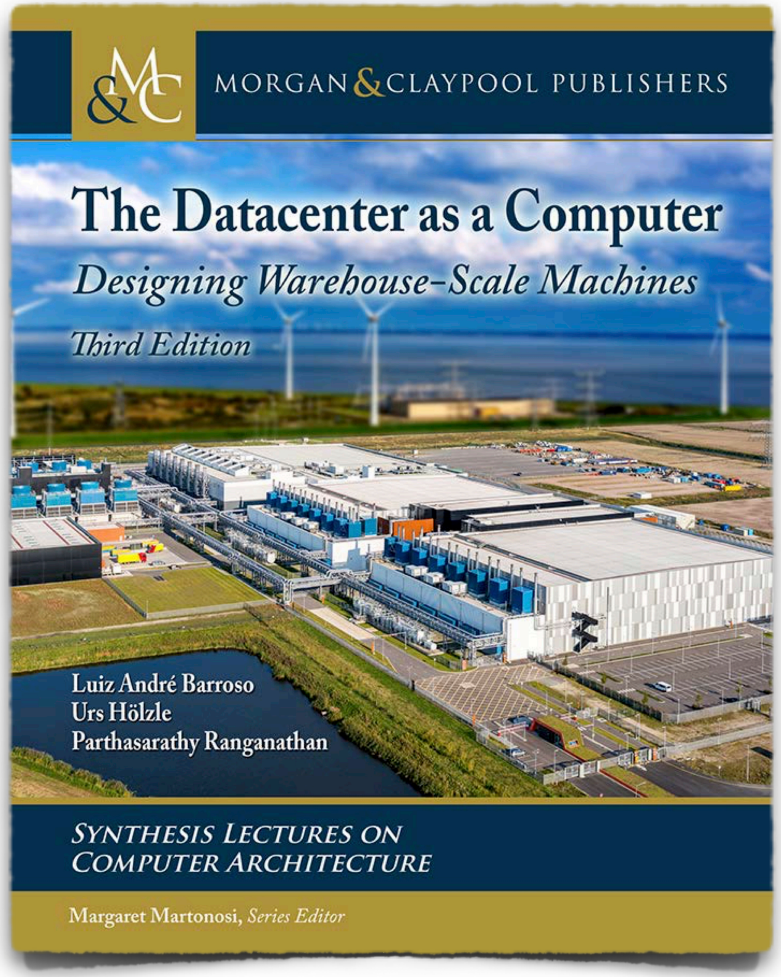
# Datacenters (3): Hardware Infrastructure

**Prof. Supreeth Shastri**
*Computer Science*
*The University of Iowa*

# Lecture goals

*Technical overview of datacenter hardware*

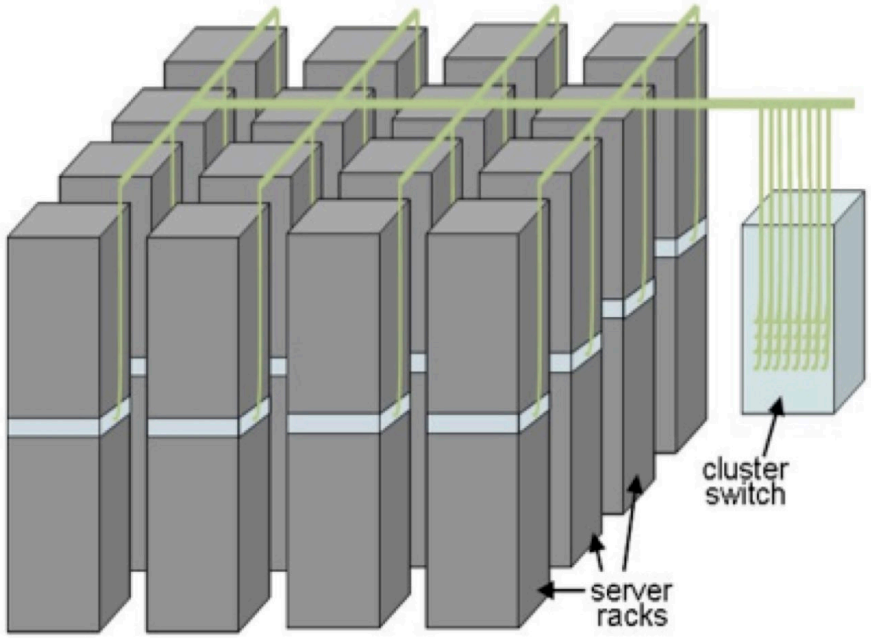
- *Computer servers*
- *Storage systems*
- *Networking gear*
- *Balanced design*



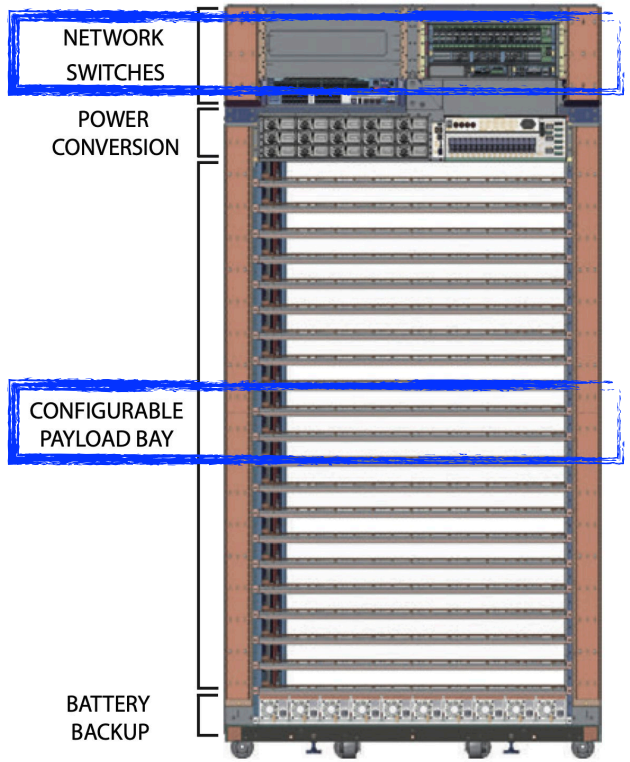

Chapter 3

IOWA

# Datacenter-level Organization

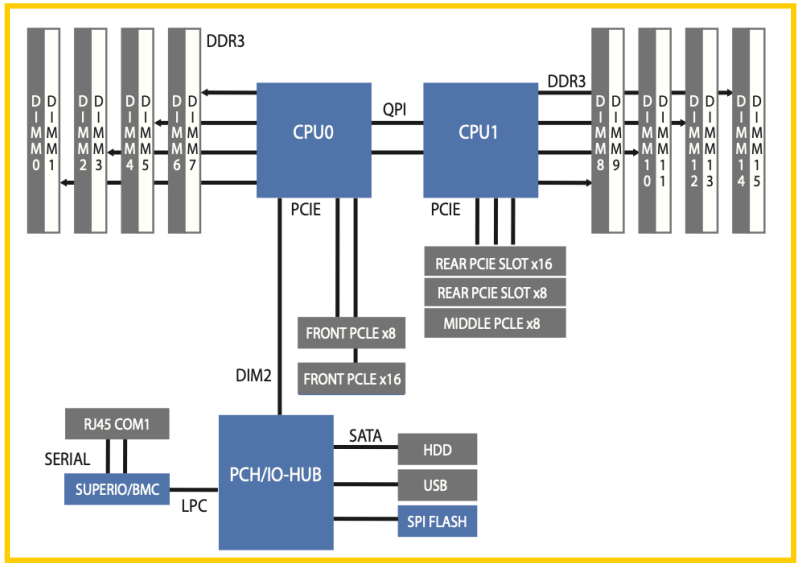



**Cluster of racks**

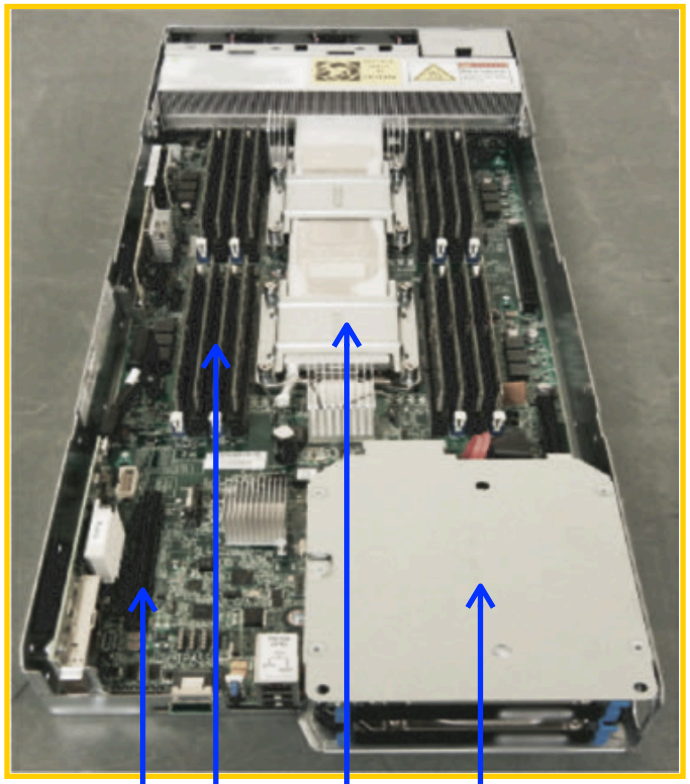



**Rack of servers and switches**

# Server Trays
## *(or Blade Servers)*




**High-level block diagram**



## Example: Intel x86 server

- 2x Intel Haswell CPU sockets (each with 18-cores)
- 16x DIMM slots that can hold two DRAM per slot
- 80 PCIe lanes for connecting SSD, HDD, NIC, accelerators etc
- SATA ports for direct attached storage

# **Server Choice:**
# cores per processor

- Shared Memory Processors (SMP) come w/ varying number of cores. *What drives this choice?*
- WSC apps are unlikely to fit within a single SMP. *So, we need to analyze at the cluster-level.*

## Modeling a representative WSC application

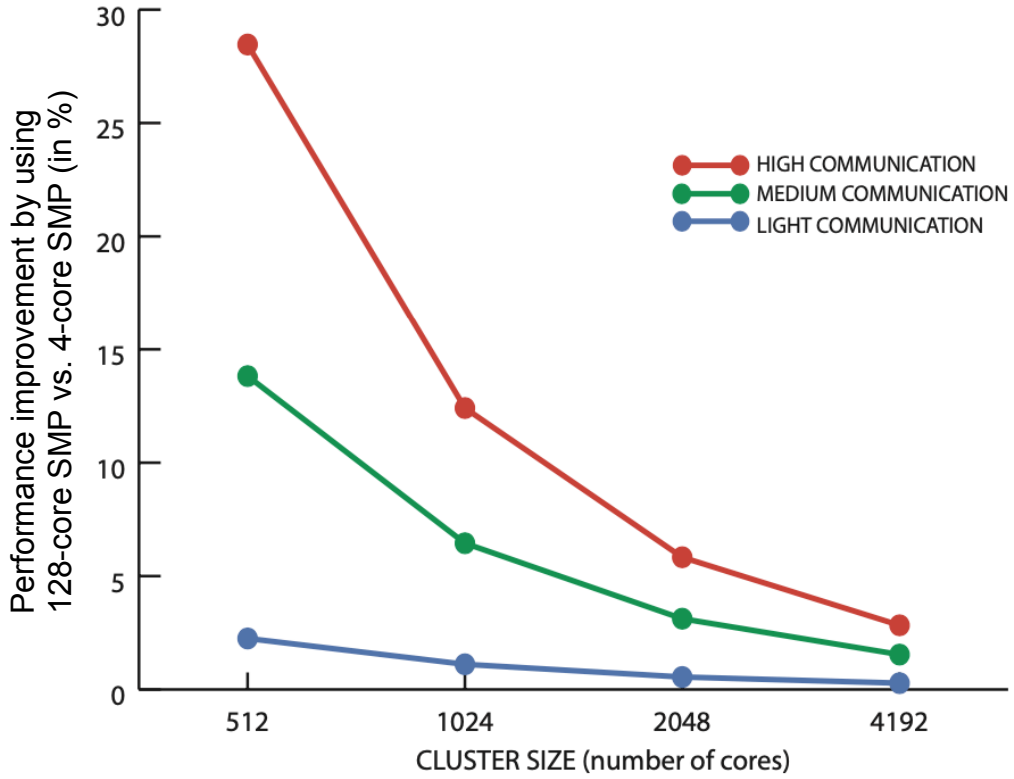
- The app takes **1 ms** for its computation and requires **f** units of data
- The data is spread across multiple cluster nodes. Local data accesses take ~100 ns (RAM), and global data accesses take ~100 µs (remote memory via Ethernet)
- Data is distributed uniformly i.e., with N nodes in the cluster, (1/N) accesses will be local and (1-1/N) will be global

$$\text{Execution time} = 1\ \text{ms} + f * [100\ \text{ns} * (1/N) + 100\ \mu\text{s} * (1 - 1/N)]$$

# **Server Choice:**
# cores per processor

### Three sub-class of WSC applications

- Light communications (f = 1)
- Medium communications (f = 10)
- High communications (f = 100)




### Key observations

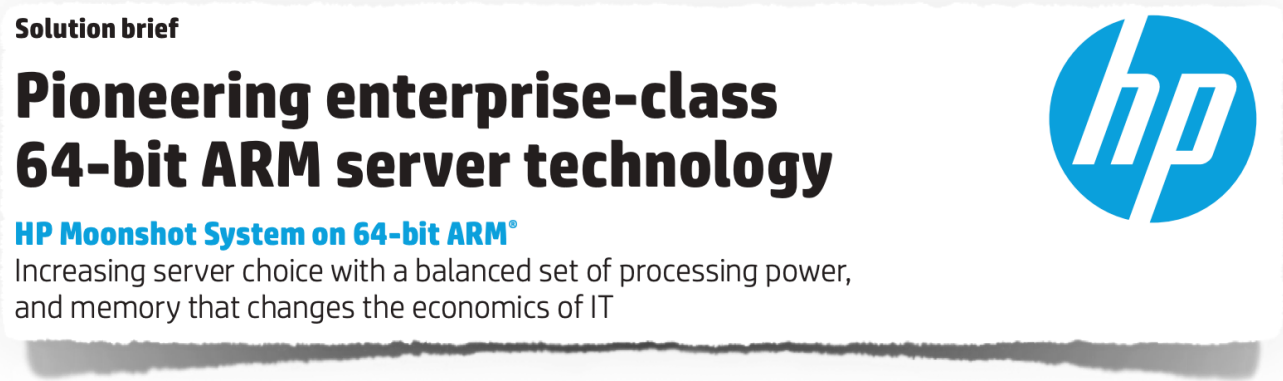
- Performance advantage of large SMPs quickly decreases as cluster size is increased
- If an application requires >2K cores, a cluster built with low-end servers (4-core SMP) performs within 10% of one built with high-end (128-core SMP) servers, while the latter is 4-20x more expensive

# **Server Choice:** brawny vs. wimpy

*Based on the the previous analysis, is it better to get away from server-grade CPUs altogether and instead use wimpy processors (e.g., mobile processors)?*

## Advantages of wimpy servers

- Multicore CPUs carry a price-performance premium over lower-end processors *so that the same amount of throughput can be bought 2-5X cheaper with multiple smaller CPUs*
- Many applications are memory- or I/O-bound *so that faster CPUs do not scale well for large applications*
- Slower CPUs tend to be more power efficient. *CPU power decreases by $O(k^2)$ when CPU frequency decreases by a factor of k*



# **Server Choice:**
# brawny vs. wimpy

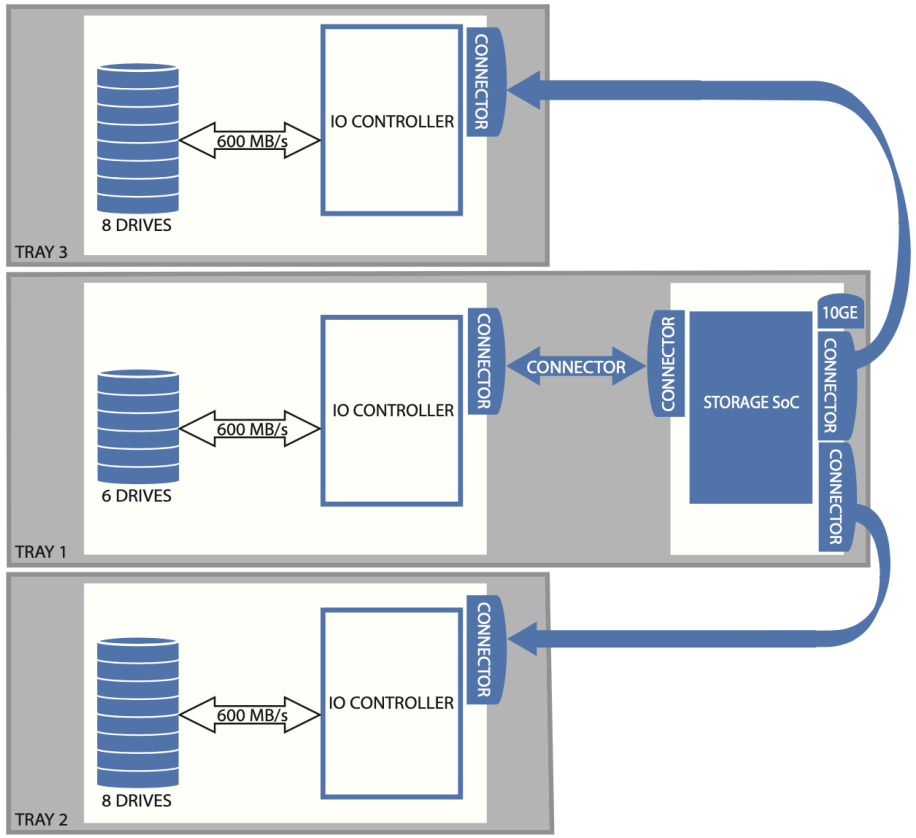
## Shortcomings of wimpy servers

- WSC applications will eventually hit limits on their request- and data-level parallelism *(due to Amdahl's law)*
- Using a large number of less-powerful threads (i.e., threads of wimpy servers) increases response variability, thus *affecting tail latency*
- The need to intensively parallelize and optimize applications may *increase software engineering costs*
- Housing a large number of (small) servers will result in *degraded network performance*, or require *additional network infrastructure*
- Applications are run on servers using bin-packing algorithms, which perform poorly as bin sizes decrease. This may lead to *lower utilization* at the datacenter level

***Amdahl's law:***
***limits the maximum speedup a program can achieve via parallelization.***

*For e.g., if a program needs 20 hours to complete using a single thread, but a one-hour portion of the program cannot be parallelized, then regardless of how many threads are devoted to a parallelized execution of this program, the minimum execution time cannot be less than one hour.*

# Disk Trays




**High-level block diagram**

- Shown is a custom designed Google tray
- Belongs to category of ***Network Attached Storage (NAS)***
- Tray provides power, Ethernet, and management services
- Runs a storage stack that handles requests over RPC
- Enabling the trend towards *diskless servers* (why?)

# Datacenters and the disk market!

- Traditionally, disks were designed for enterprise and consumer markets
- This started to change circa 2000, when most data started residing in the cloud
- Brewer et. al., wrote one of the most lucid papers on this: ***Disks for Datacenters [FAST 2016]***

## How are datacenter requirements different?

- WSC applications value higher IOPS vs sequential read/write throughput
- Require control/guarantees over tail latency
- Lower cost per stored bit (can trade durability for cost)
- Security requirements (bug-free firmware, built-in encryption etc)

# Datacenter networking

**Cisco Nexus 3500**
(a Top of the Rack switch)

- 1 rack unit (height)
- 48 ports
- 1-40G Ethernet ports
- Price ~$15-25K

**Cisco ASR 9922 family**
(for datacenter core)

- 44 rack units (height)
- up to 20 linecards/switches
- 400G Ethernet ports
- Price ~$300-500K

# Datacenter networking

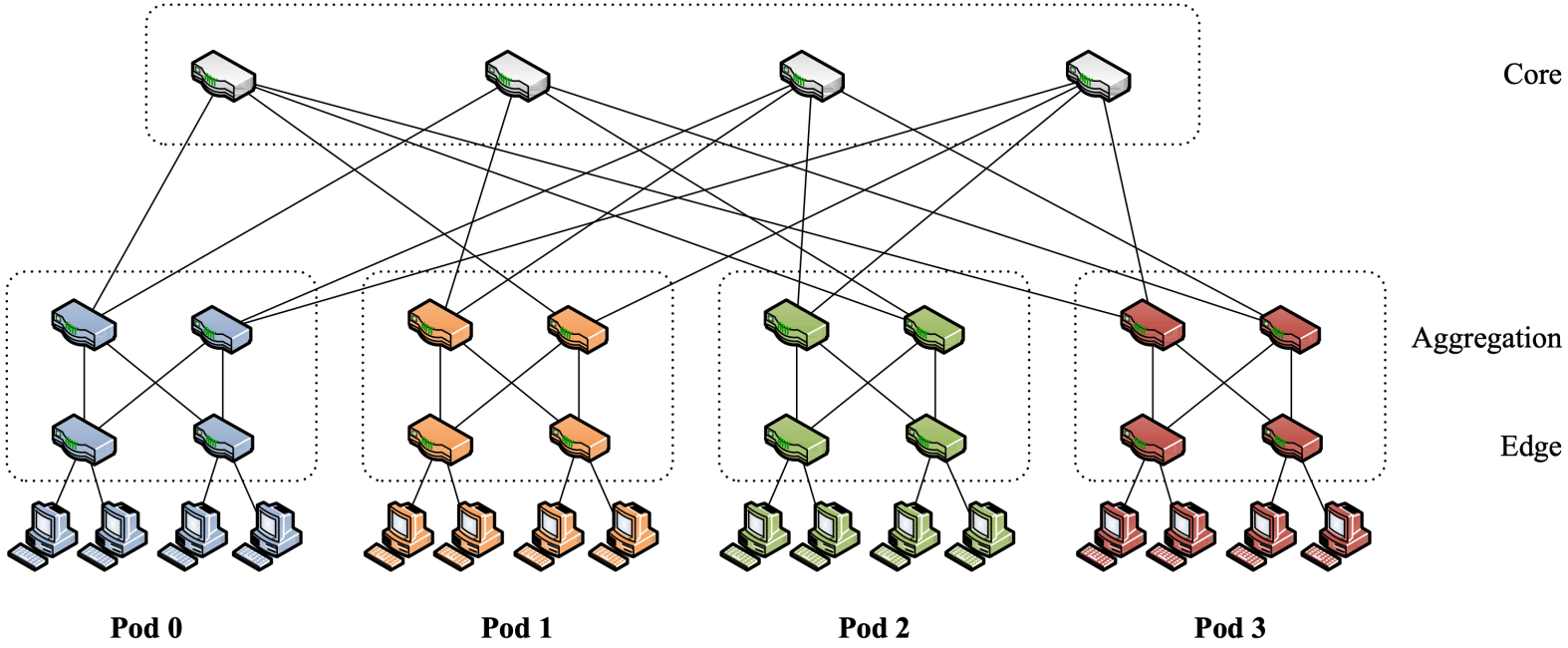
## Difficulty of horizontal scaling

- Compute and storage scale well horizontally i.e., *if you need extra aggregate capacity, simply add more boxes*
- In networking, simply increasing the bandwidth of the leaf node is not enough (why?)
- **Solution**: increase the bisection bandwidth of the network i.e., *bandwidth across the narrowest line that divides the cluster into two parts*
- **Challenge**: we cannot increase bisection bandwidth by simply making or buying arbitrarily large switches and routers (limitations in physics and manufacturing)
- **Solution**: build novel network topologies; for e.g., fat-tree, or CLOS

*Why bisection bandwidth?*
*Assuming uniformly random communication between servers,* we expect every node to send about half its bits across the "middle" of the network

# Datacenter networking

**Three-stage fat tree network**
*(built with 4-port switches)*




- A fat tree built with k-port switches can support full throughput for $k^3/4$ servers
- Problem with fat trees: every communication path involves multiple switch ports (=> *more money, more delay*)
- To reduce costs, datacenters over-subscribe the top-of-the-rack switches i.e., ToR switches are configured to provide more downward bandwidth than upward bandwidth
- *For example, a 48-port switch could connect 40 servers to 8 uplinks, for a 5:1 oversubscription*

# Balanced Designs

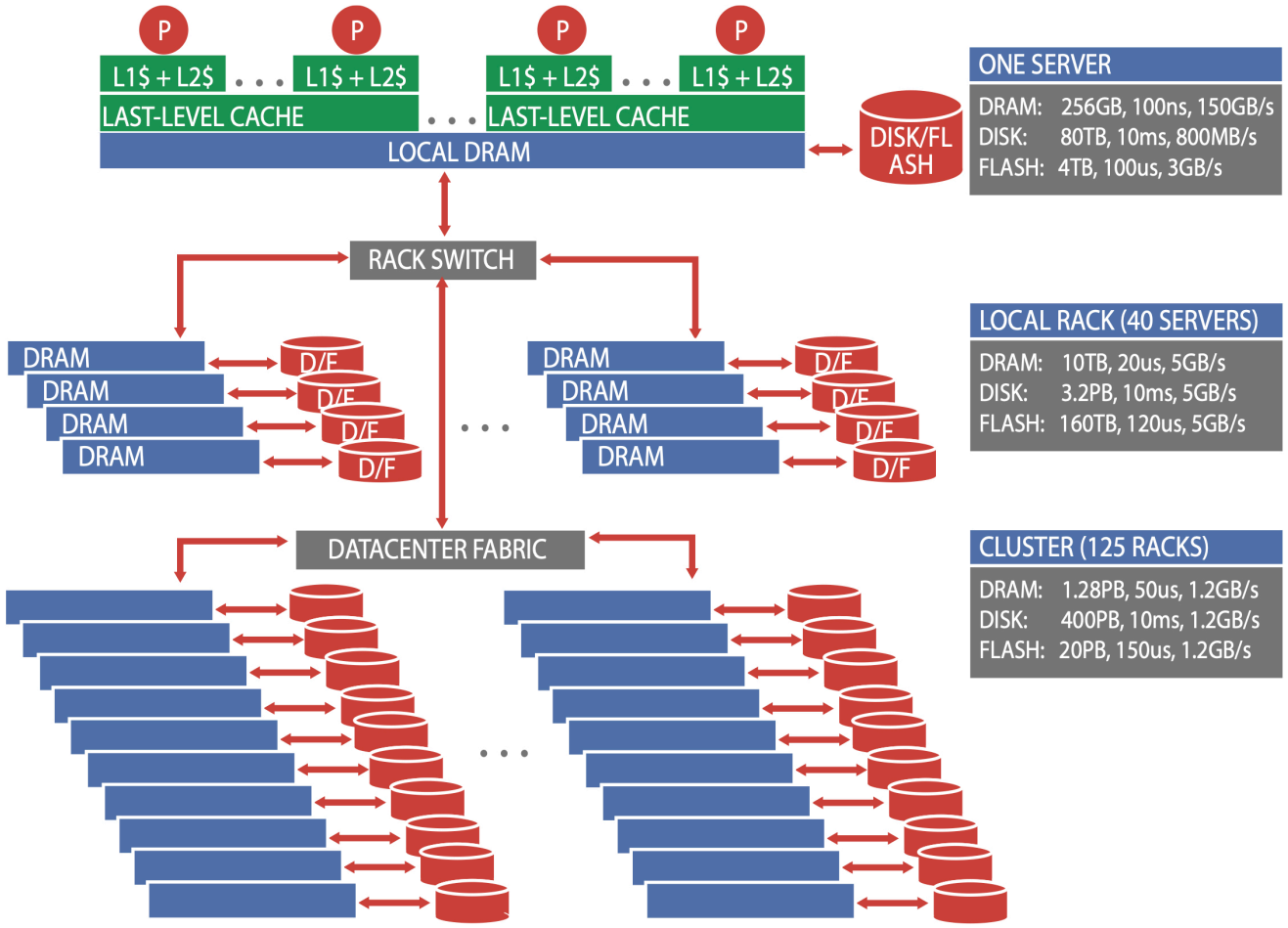
# Latencies

| Operation | Time |
|---|---|
| L1 cache reference | 1 ns |
| L2 cache reference | 5 ns |
| L3 cache reference | 25 ns |
| Main memory reference | 100 ns |
| NVM reference | 1,000 ns |
| Read 1MB from memory | 12,000 ns |
| Read 1MB seq from SSD | 500,000 ns |
| Read 1MB seq from 10Gbps network | 1,000,000 ns |
| Read 1MB seq from hard drive | 10,000,000 ns |
| Packet roundtrip CA —> Netherlands —> CA | 150,000,000 ns |

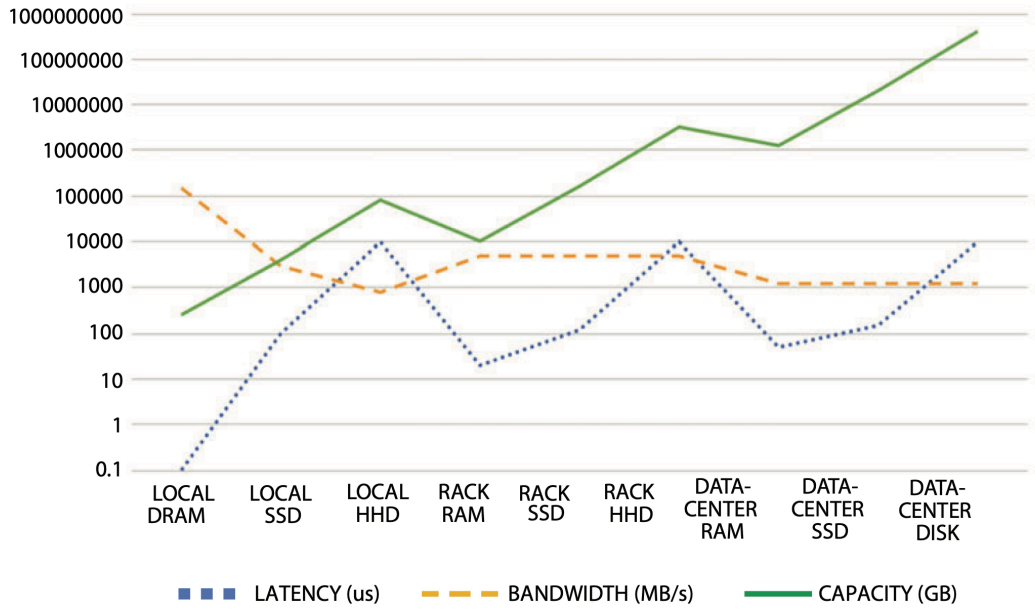
# A View of Resources at the Datacenter Level

*As we move from top towards the bottom,*

- ➡ Access latency increases
- ➡ Application complexity may increase
- ➡ Capacity (available storage) increases
- ➡ Cost (price per bit of data) reduces
- ➡ Throughput (aggregate bits/second) may increase



# Balanced Design




**Hardware-software co-design**

*e.g., restructuring algorithms to benefit from inexpensive hardware*

**One WSC but many WSC apps**

ability to reason w/ an aggregate view of applications while designing WSCs

**Fungible resources and efficiency**

e.g., intelligent/adaptive systems software that can expose underlying tradeoffs to applications

# Essay (ICON)

In lieu of spot quiz but it is worth 2x spot quiz grades