CS-E4640 - Big Data Platforms
Lecture 1

Keijo Heljanko

Department of Computer Science
University of Helsinki & Aalto University
keijo.heljanko@helsinki.fi

12.9-2018

CS-E4640 - Big Data Platforms (5 cr)

- This course focuses on advanced scalable cloud computing technologies and on key algorithmic ideas and methods used to implement them.
- Lectures: Wed 16:15–18:00 in Lecture Hall TU2 (Exception: 26.9 in CS building T1)
- Home Assignment Tutorials: Fri 14:15–15:00 in Lecture Hall TU2, first Home Assignment Exercise on 28th of September.
- No Lecture on 31.10 due to Lecturer travel
- Substitutes the courses: CS–E4120/CSE–E5430/T–79.5308 Scalable Cloud Computing P (5 cr), T–79.5307 Distributed Computing P (5 cr) and T-106.5250 Distributed Systems
- Course homepage available in MyCourses portal: `https://mycourses.aalto.fi/`

Course Personnel

- Lectures: Prof. Keijo Heljanko
  - Email: keijo.heljanko@helsinki.fi
- Tutorials and Home Assignments: M.Sc. Ilari Maarala
  - Email: ilari.maarala@aalto.fi

Course Requirements

To pass the course you have to:

- Pass the exam.
- Get enough points from Apache Spark programming Home Assignments:
  - $\geq$ 50% of points from Home Assignments to pass,
  - $\geq$ 80% of points gives +1 to exam grade (not to 0 or 5).

  The assignments should be done individually, no exercise groups/sharing of solutions allowed.

Home Assignment Schedule

The deadlines are tight!

- Friday 28.9 at 14:00 - Assignment 1 distributed
- Friday 19.10 at 14:00 - Deadline of Assignment 1, Assignment 2 distributed
- Friday 9.11 at 14:00 - Deadline of Assignment 2, Assignment 3 distributed
- Friday 30.11 at 14:00 - Deadline of Assignment 3

The deadlines are tight!

Course Material

- Home Assignments are to be done using an automated home assignment grading system. This is the second year the system in used in this course. Still consider yourselves as beta testers of the system, so please be patient and expect some minor glitches.
- All material needed for the exam will be distributed through the MyCourses portal.
- Material will mostly consist of the lecture slides.
- Also there will be tutorial materials to study weekly in MyCourses, with correct answers added a week later.
- There is no single book the course will be based on but we can recommend some additional reading on the topic.

Cloud Computing

- The NIST Definition of Cloud Computing
  Authors: Peter Mell and Tim Grance
  Version 15, 10-7-09
  http://www.nist.gov/itl/cloud/upload/cloud-def-v15.pdf

  - Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model promotes availability and is composed of five essential **characteristics**, three **service models**, and four **deployment models**.

Cloud Computing (cnt.)

- **Essential characteristics**: On-demand self-service, Broad network access, Resource pooling, Rapid elasticity, and Measured Service.
- **Service Models**: Cloud Software as a Service (SaaS), Cloud Platform as a Service (PaaS), and Cloud Infrastructure as a Service (IaaS).
- **Deployment Models**: Private cloud, Community cloud, Public cloud, and Hybrid cloud.

Business Drivers of Clouds

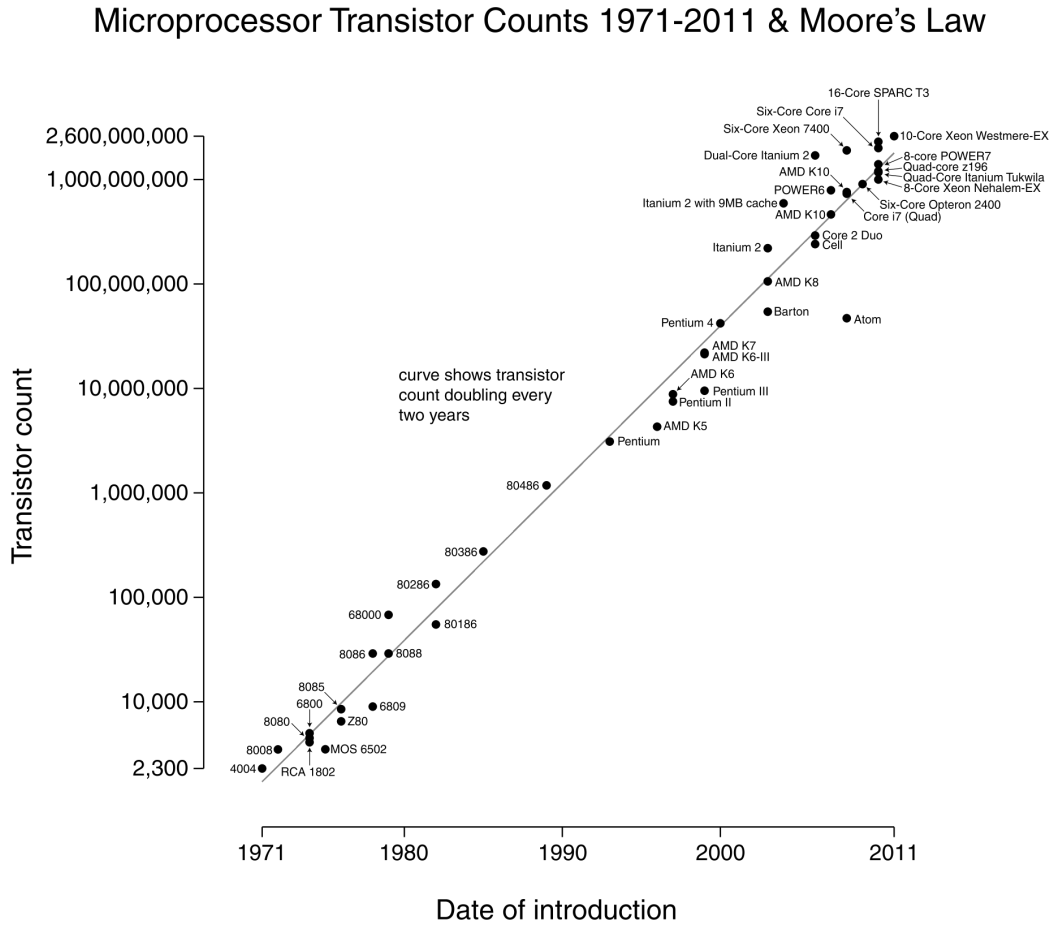
- Large data centers allow for economics of scale
  - Cheaper hardware purchases
  - Cheaper cooling of hardware
    - Example: Google paid 40 MEur for a Summa paper mill site in Hamina, Finland: Data center cooled with sea water from the Baltic Sea
  - Cheaper electricity
  - Cheaper network capacity
  - Smaller number of administrators / computer
- Unreliable commodity hardware is used
- Reliability obtained by replication of hardware components and a combined with a fault tolerant software stack

Cloud Computing Technologies

A collection of technologies aimed to provide elastic "pay as you go" computing

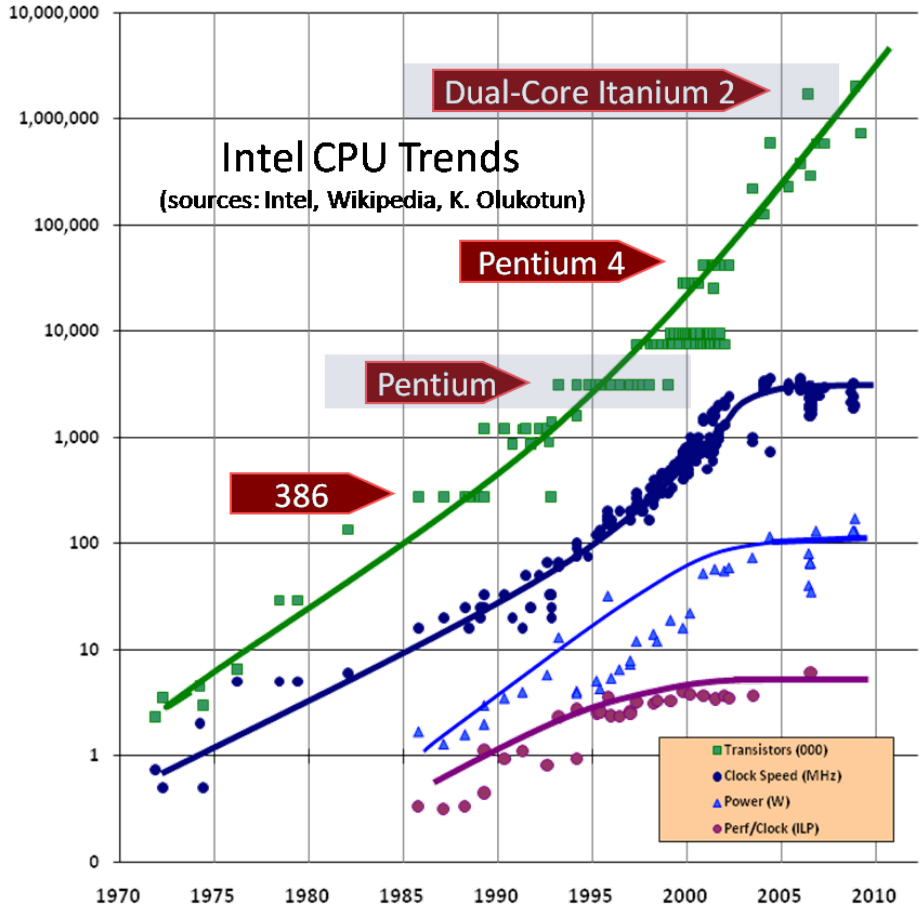
- Virtualization of computing resources: Amazon EC2, Eucalyptus, OpenNebula, Open Stack Compute, ...
- Scalable file storage: Amazon S3, GFS, HDFS, ...
- Scalable batch processing: Google MapReduce / Apache Hadoop, Apache Flink, Microsoft Dryad, Google Pregel, Apache Spark, ...
- Scalable datastore: Amazon Dynamo, Apache Cassandra, Riak, Google Bigtable, HBase, Chubby, Apache Zookeeper...
- Scalable Web applications hosting: Google App Engine, Microsoft Azure, Heroku, ...

Moore's Law (from Wikipedia)



- Gordon Moore of Intel 1965: the number of transistors in integrated circuits doubles roughly every 18-24 months

Clock Speed of Processors



- Herb Sutter: The Free Lunch Is Over: A Fundamental Turn Toward Concurrency in Software. Dr. Dobb's Journal, 30(3), March 2005 (updated graph in August 2009).

Implications of the End of Free Lunch

- The clock speeds of microprocessors are not going to improve much in the foreseeable future
  - The efficiency gains in single threaded performance are going to be only moderate
- The number of transistors in a microprocessor is still growing at a high rate
  - One of the main uses of transistors has been to increase the number of computing cores the processor has
  - The number of cores in a low end workstation (as those employed in large scale datacenters) is going to keep on steadily growing
- Programming models need to change to efficiently exploit all the available concurrency - scalability to high number of cores/processors will need to be a major focus

Dark Silicon - End of Moore's law in Sight?!

- Even worse news ahead, computing will be hitting a power wall: Silicon is becoming energy constrained
- We will have much more transistors than what we can switch on and off at each clock cycle: Dark Silicon
- For implications of dark silicon, see:
  - Hadi Esmaeilzadeh, Emily R. Blem, Rene St. Amant, Karthikeyan Sankaralingam, Doug Burger: Dark Silicon and the End of Multicore Scaling. IEEE Micro 32(3): 122-134 (2012)
  - Michael B. Taylor: A Landscape of the New Dark Silicon Design Regime. IEEE Micro 33(5): 8-19 (2013)

Dark Silicon - Improving Energy Efficiency

- To combat the power wall more energy efficient computing is needed
- GPGPU computing is improving the energy efficiency of training deep neural networks
- Special purpose hardware (ASICs) for inference of deep neural networks: Google Tensor Processing Unit v1 and v2
- Similar special purpose accelerators will become a necessity also to other fields than machine learning

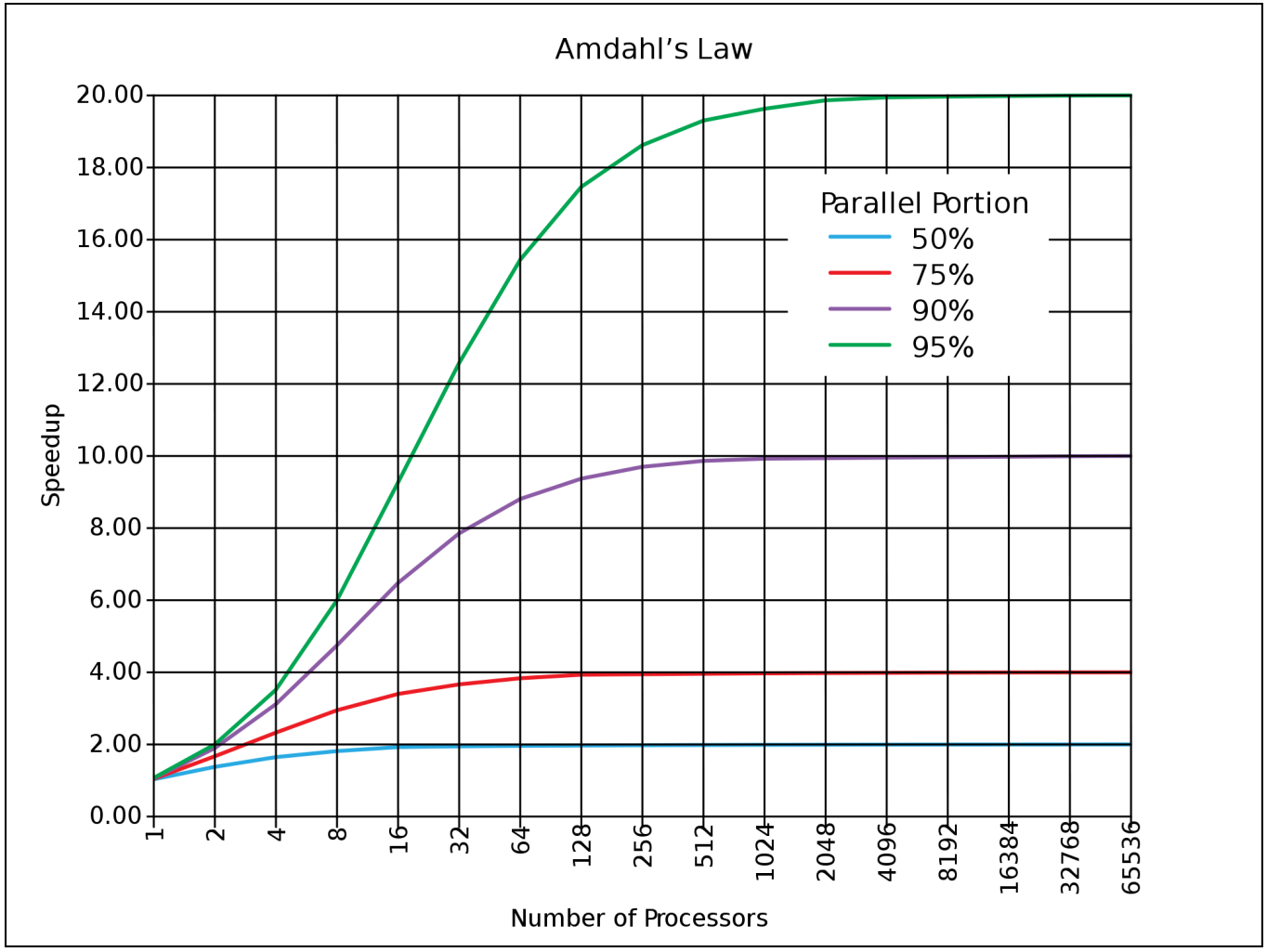
Amdahl's Law

- In order to keep the ever increasing number of cores busy, the applications must be extremely parallel to enjoy the benefits of increasing transistor counts
- Unfortunately, the Amdahl's Law still applies:
  - Amdahl's law states that if P is the proportion of a program that can be made parallel (i.e. benefit from parallelization), and $(1 - P)$ is the proportion that cannot be parallelized (remains serial), then the maximum speedup that can be achieved by using N processors is:

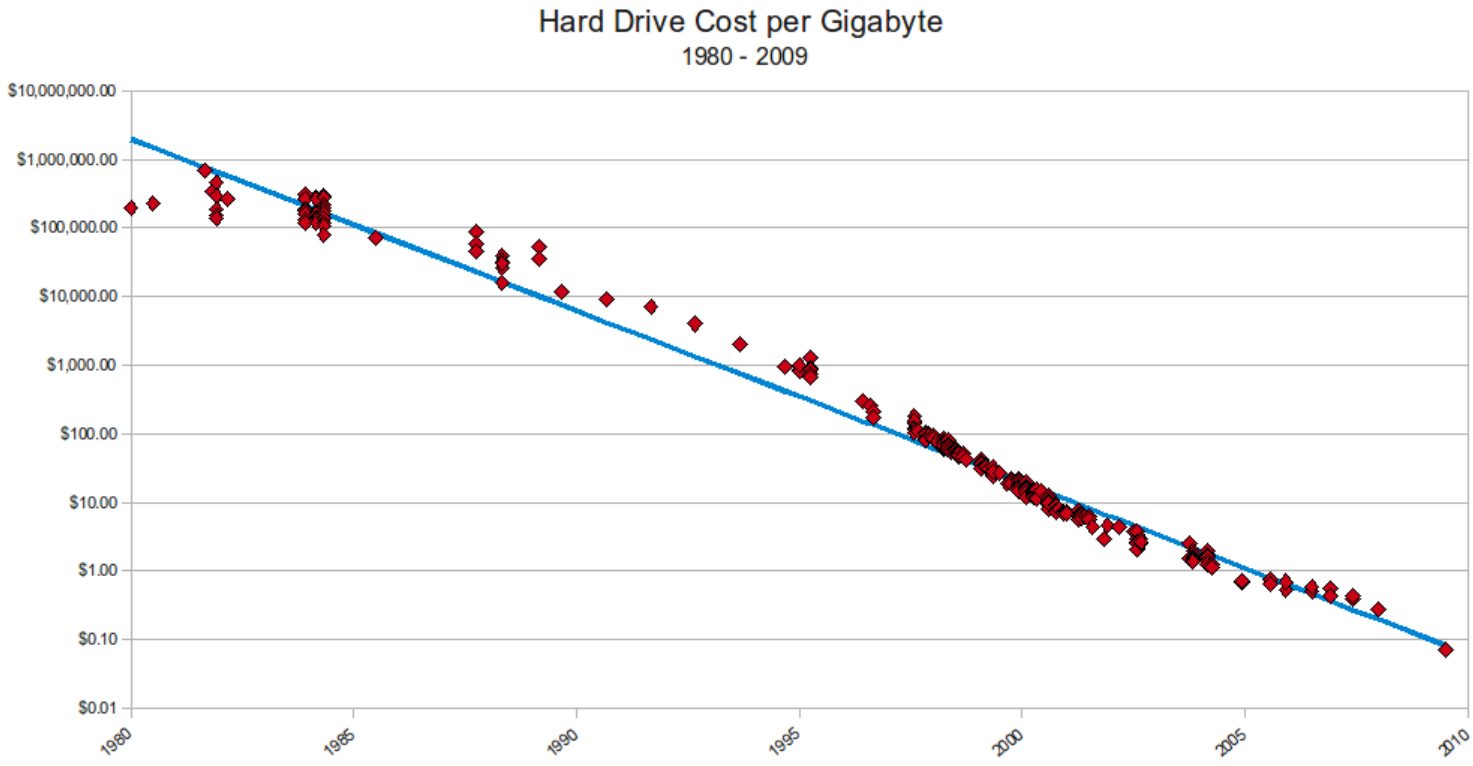
$$\frac{1}{(1 - P) + \frac{P}{N}}$$

http://en.wikipedia.org/wiki/Amdahl%27s_law

Amdahl's Law (Example from Wikipedia)



Kryder's Law



- Cost of storing a bit on a hard disk halves every 14 months (though this improvent has slowed down recently): `http://www.mkomo.com/cost-per-gigabyte`

Scaling Up vs Scaling Out

- Scaling up: When the need for parallelism arises, a single powerful computer is added with more CPU cores, more memory, and more hard disks
- Scaling out: When the need for parallelism arises, the task is divided between a large number of less powerful machines with (relatively) slow CPUs, moderate memory amounts, moderate hard disk counts
- Scalable cloud computing is trying to exploiting scaling out instead of scaling up

Pros and Cons of Scaling Up vs Scaling Out

- Scaling up is more expensive than scaling out. Big high-end systems have much higher pricing for a given: CPU power, memory, and hard disk space
- Scaling out is more challenging for fault tolerance. A large number of loosely coupled systems means more components and thus more failures in hardware and in networking. Solution: Software fault tolerance
- Scaling out is more challenging for software development due to larger number of components, larger number of failures both in nodes and networking connecting them, and increased latencies. Solution: Scalable cloud platforms

Warehouse-scale Computing (WSC)

- The smallest unit of computation in Google scale is: Warehouse full of computers
- [WSC]: Luiz André Barroso, Jimmy Clidaras, and Urs Hölzle: *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines, Second edition* Morgan & Claypool Publishers 2013

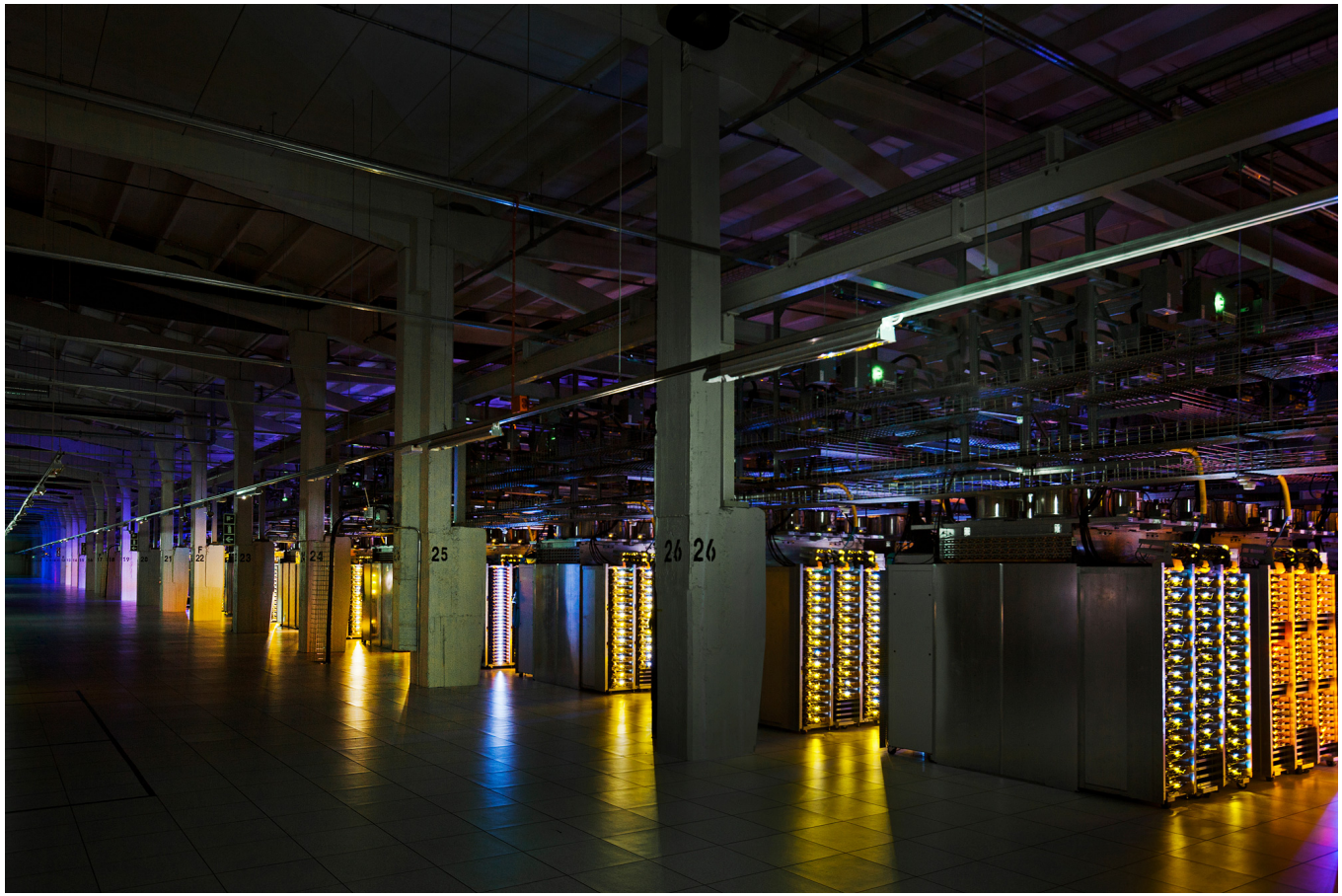
  http://dx.doi.org/10.2200/S00516ED2V01Y201306CAC024
- The WSC book says: "...we must treat the datacenter itself as one massive warehouse-scale computer (WSC)."

Google Summa Warehouse Scale Computer

Google Summa from the Roof Level

Google Summa Sea Water Cooling

Google Summa Computer Hall



Elements of WSC

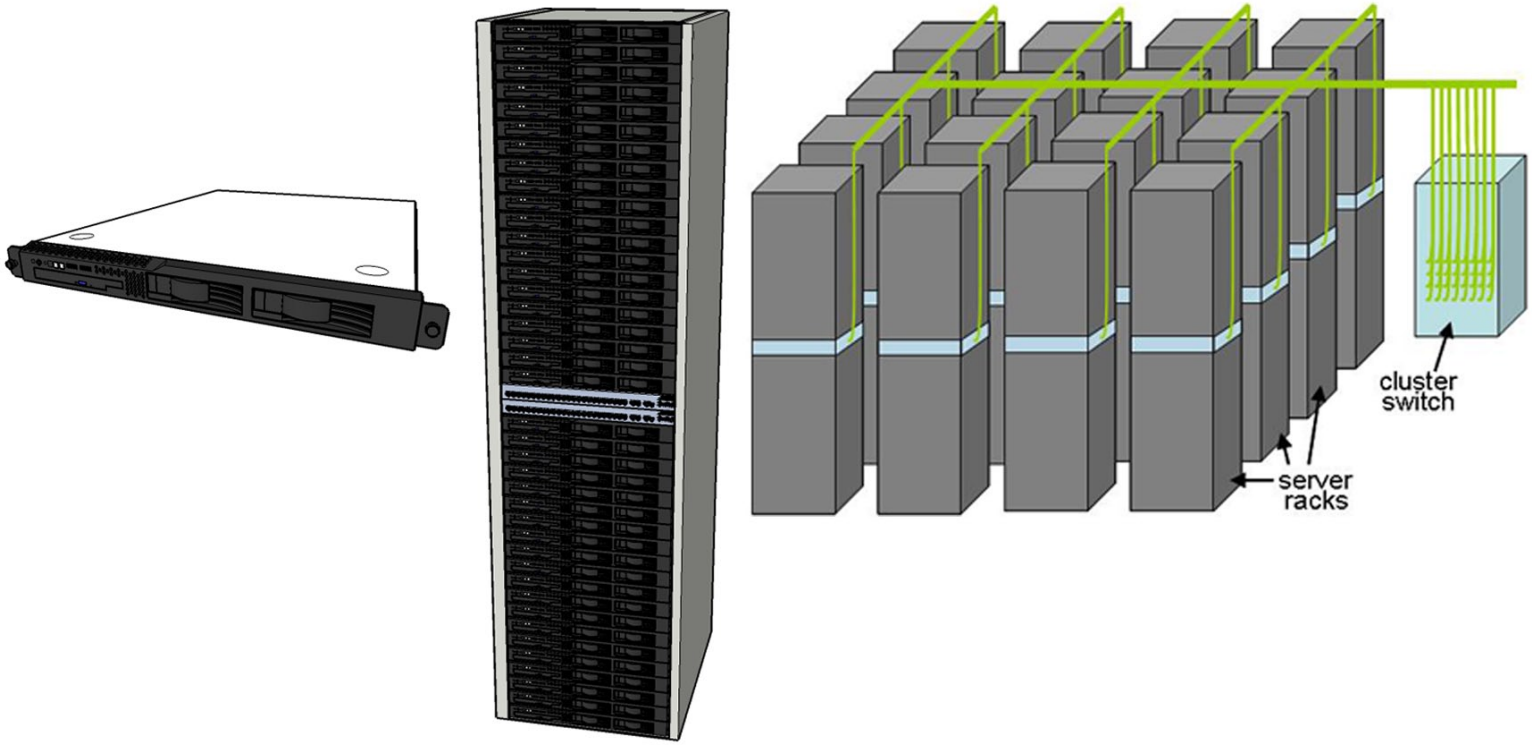


Figure: Typical elements in WSC, Figure 1.1 of [WSC]

Google WSC from 2012

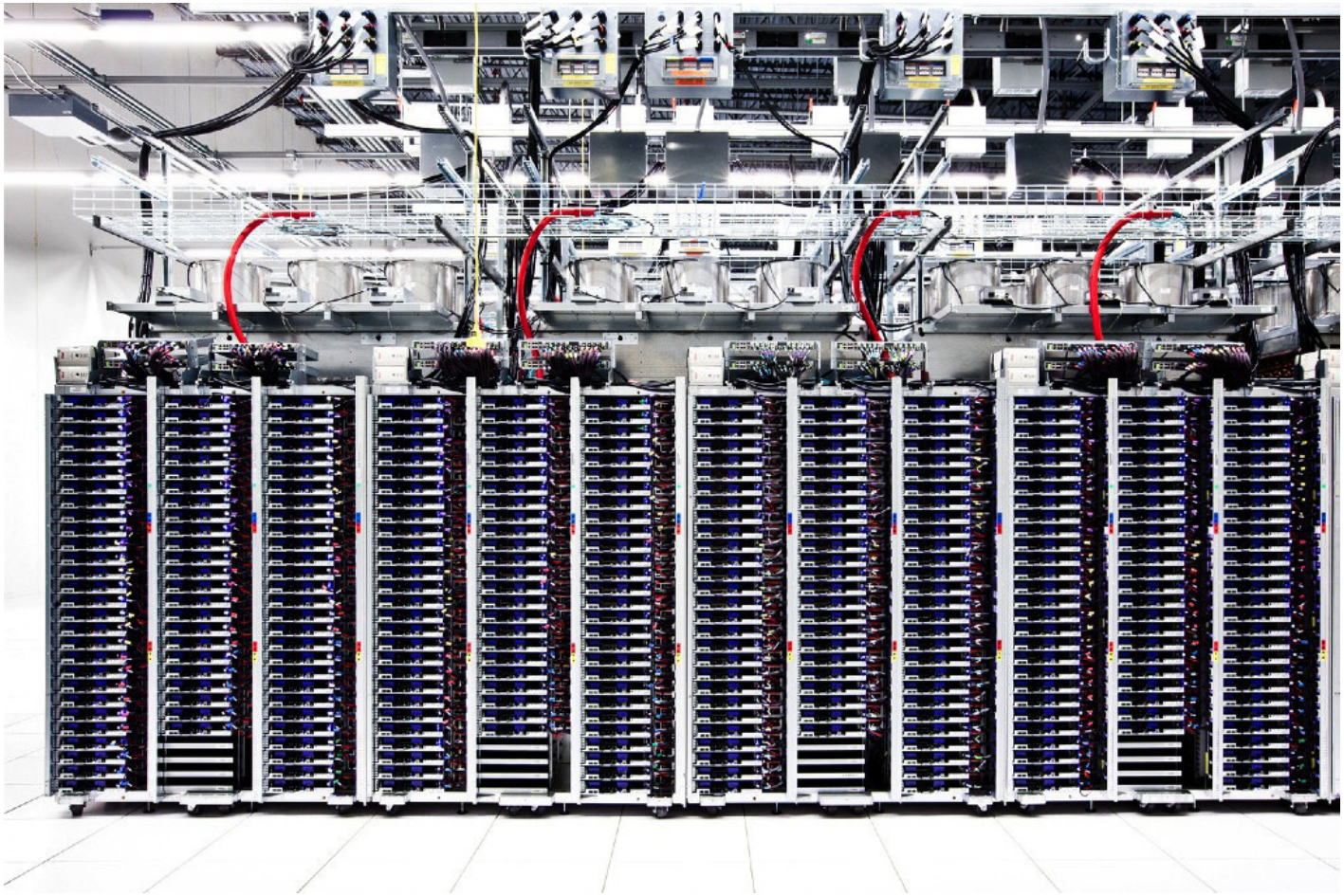


Figure: Picture of a row of servers from Google, Figure 1.2 of [WSC]

A Classical Datacenter

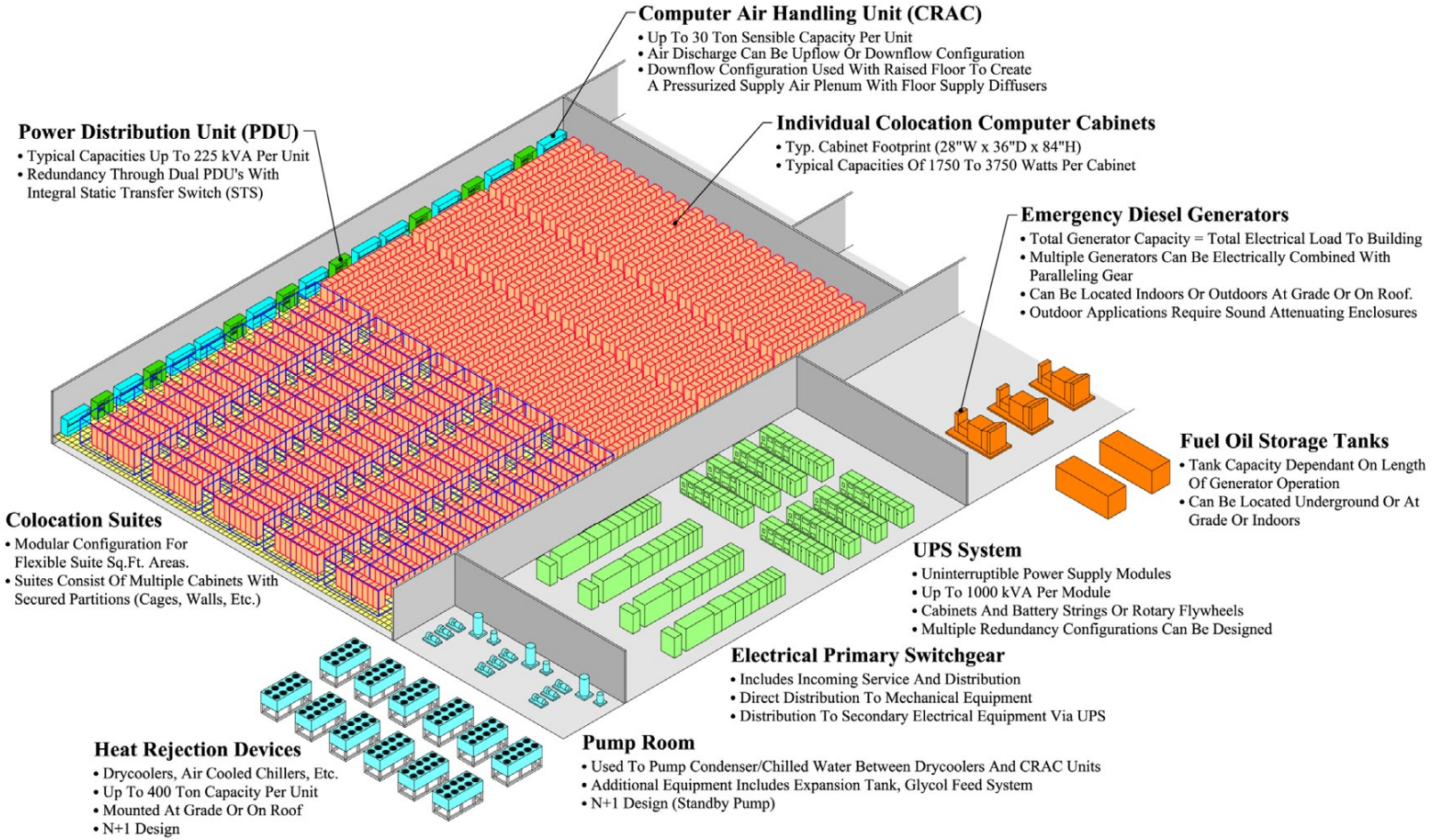


Figure: A Classical Datacenter, Figure 4.1 of [WSC]

A Classical Cooling System

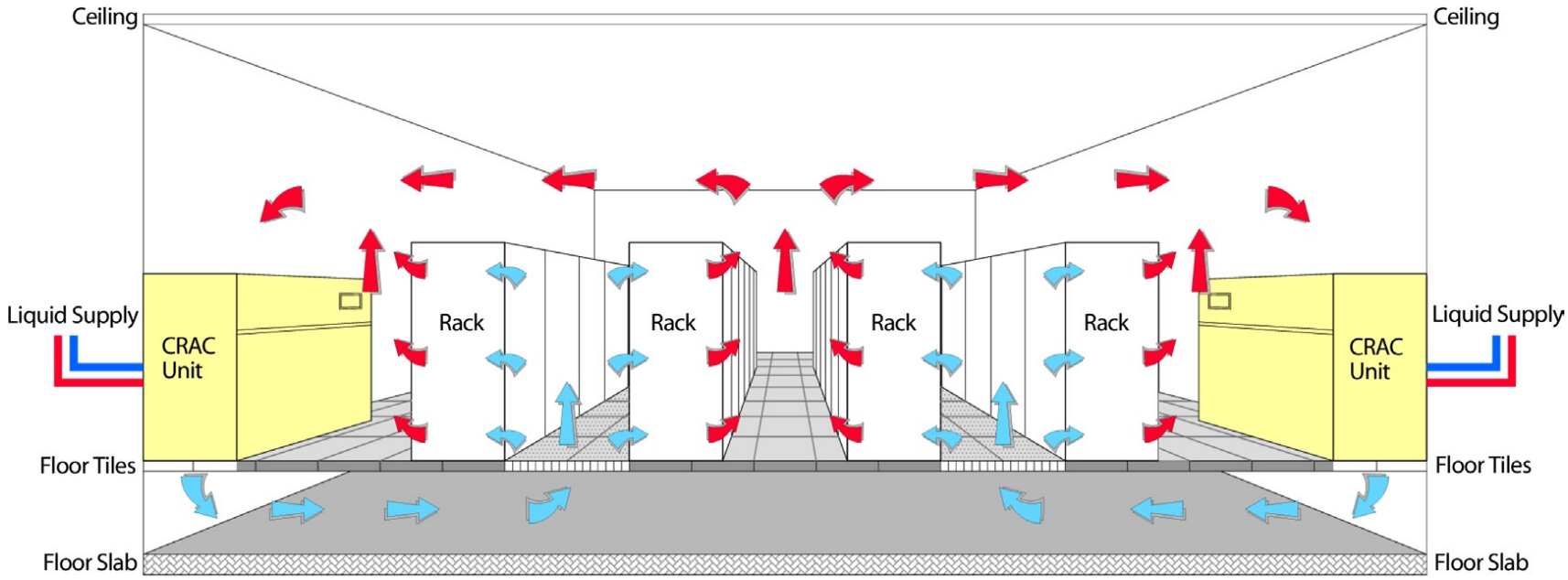


Figure: A Classical Cooling System, Figure 4.6 of [WSC]

Power Usage Breakdown of Servers

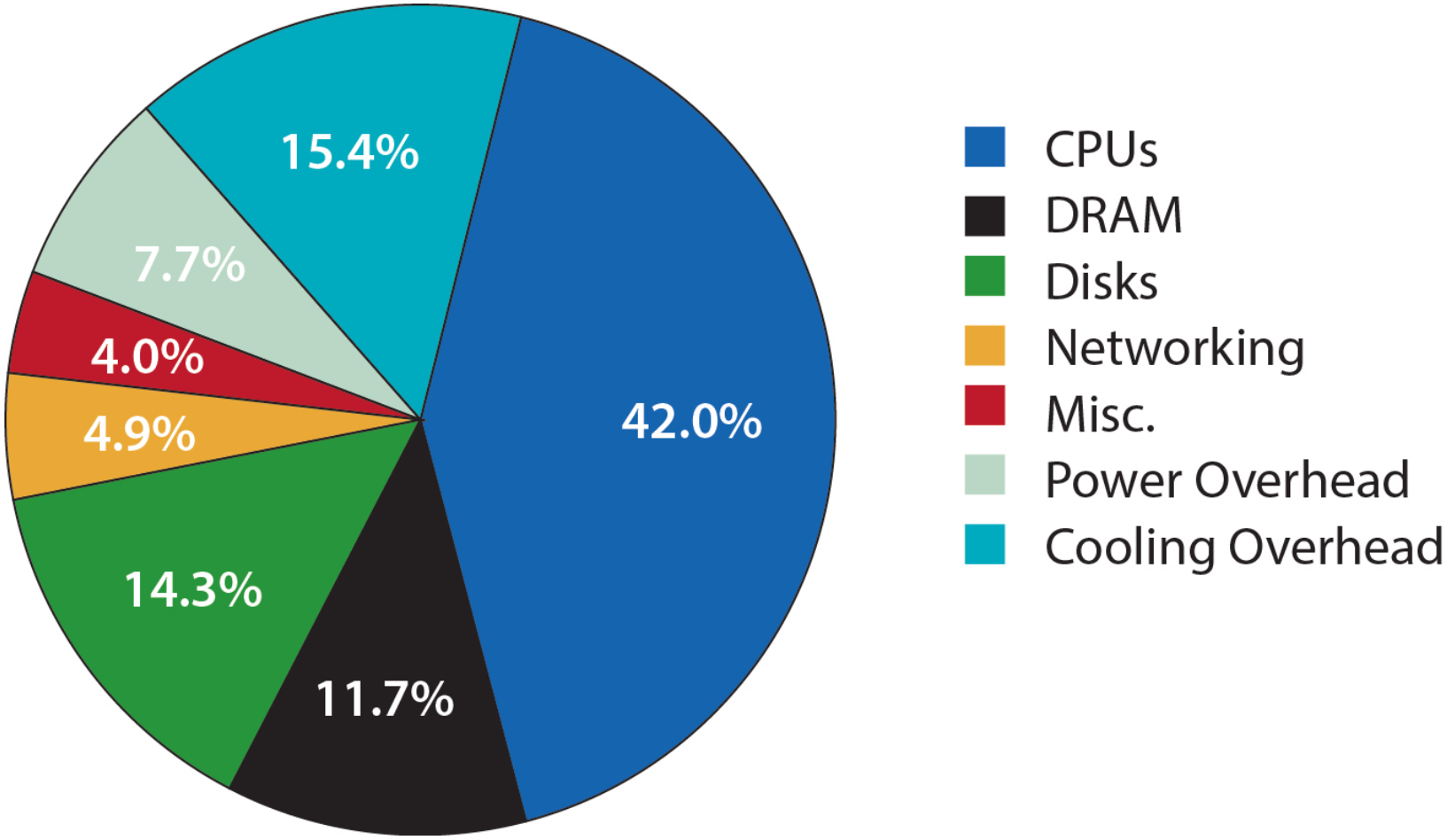


Figure: Power Usage Breakdown for Two Socket Servers (from 2012), Figure 1.6 of [WSC]

Storage Hierarchy of WSC

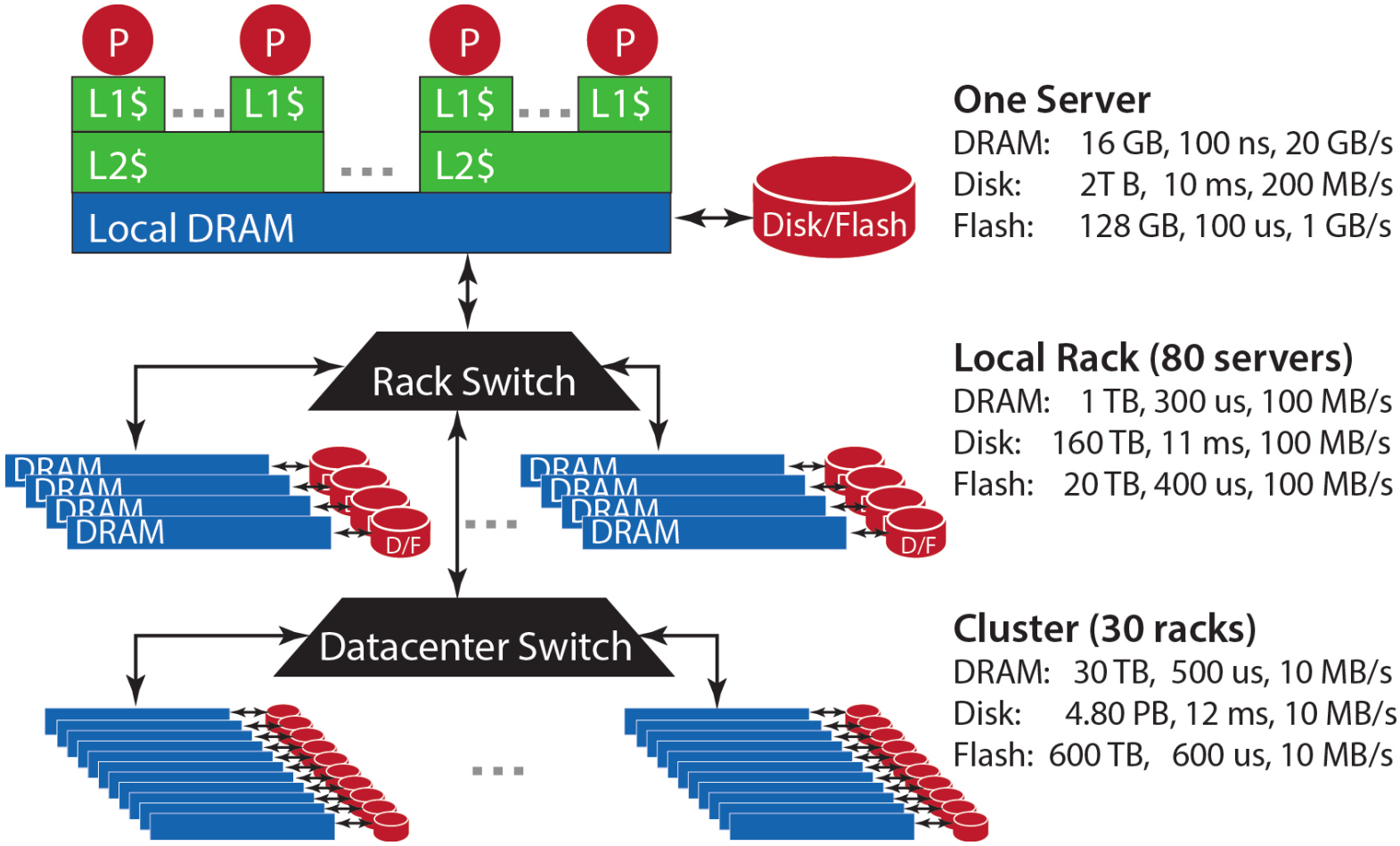


Figure: Storage hierarchy of WSC, Figure 1.3 of [WSC]

WSC latency, bandwidth, and capacity

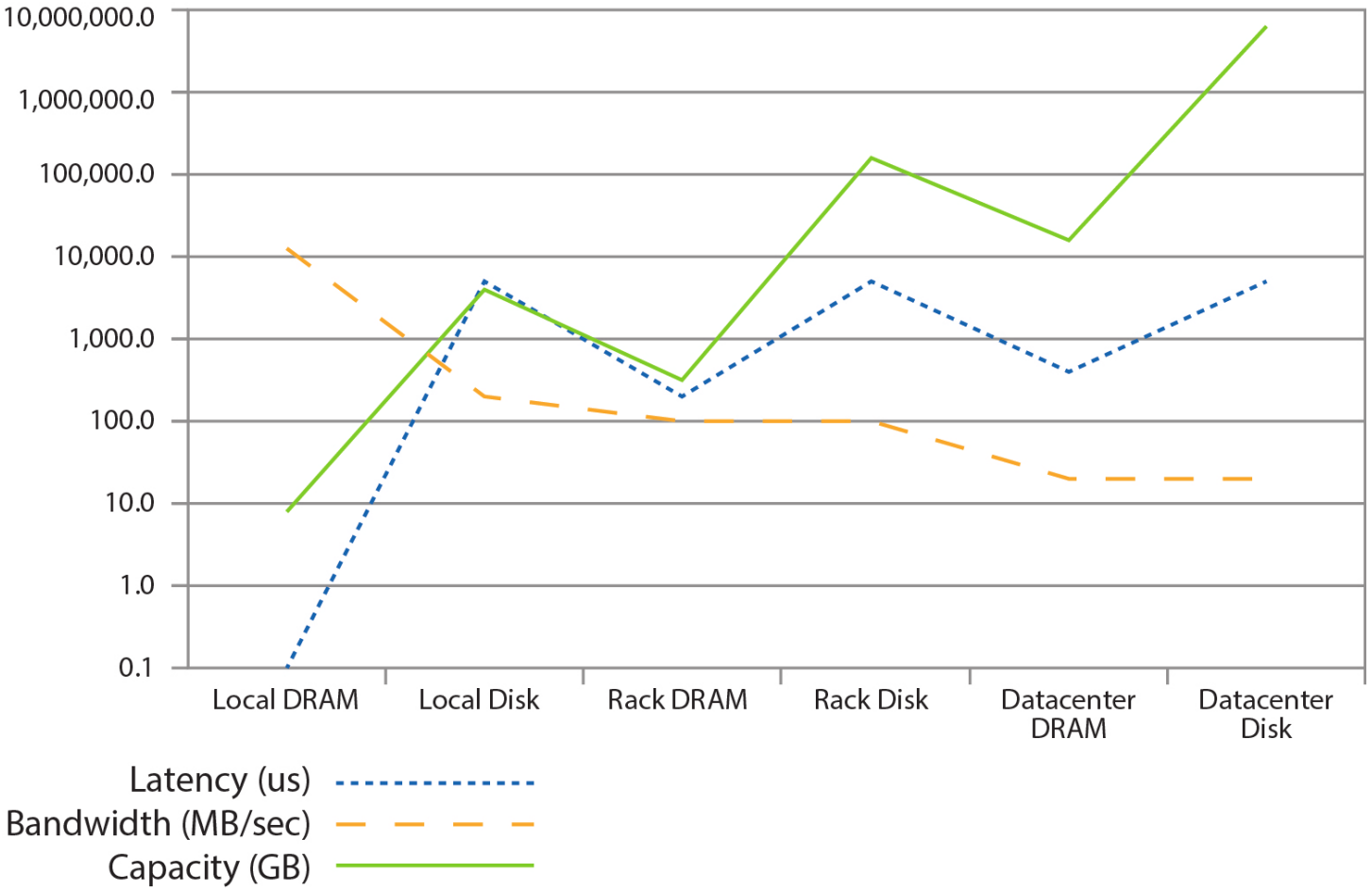


Figure: WSC latency, bandwidth, and capacity, Figure 1.4 of [WSC]

Jeffrey Dean (Google): Rule of Thumb Numbers

- LADIS 2009 keynote: “Designs, Lessons and Advice from Building Large Distributed Systems”
  `http://www.cs.cornell.edu/projects/ladis2009/talks/dean-keynote-ladis2009.pdf`

L1 cache reference	0.5 ns
Branch mispredict	5 ns
L2 cache reference	7 ns
Mutex lock/unlock	25 ns
Main memory reference	100 ns
Compress 1K bytes with Zippy	3,000 ns
Send 2K bytes over 1 Gbps network	20,000 ns
Read 1 MB sequentially from memory	250,000 ns
Round trip within same datacenter	500,000 ns
Disk seek	10,000,000 ns
Read 1 MB sequentially from disk	20,000,000 ns
Send packet CA $\Rightarrow$ Netherlands $\Rightarrow$ CA	150,000,000 ns

Jeffrey Dean (Google): LADIS 2009 keynote failure numbers

- At warehouse-scale, failures will be the norm and thus fault tolerant software will be inevitable
- Hypothetically assume that server mean time between failures (MTBF) would be 30 years (highly optimistic and not realistic number!). In a WSC with 10000 nodes roughly one node will fail per day.
- Typical yearly flakiness metrics from J. Dean (Google, 2009):
  - 1-5% of your disk drives will die
  - Servers will crash at least twice (2-4% failure rate)

Jeffrey Dean (Google): Joys of Real Hardware (in verbatim)

Typical first year for a new cluster:

- $\approx$ 0.5 overheating (power down most machines in $<$ 5 mins $\approx$ 1-2 days to recover)
- $\approx$ 1 PDU failure ($\approx$ 500-1000 machines suddenly disappear, $\approx$ 6 hours to come back)
- $\approx$ 1 rack-move (plenty of warning, $\approx$ 500-1000 machines powered down, $\approx$ 6 hours)
- $\approx$ 1 network rewiring (rolling $\approx$5 % of machines down over 2-day span)
- $\approx$ 20 rack failures (40-80 machines instantly disappear, 1-6 hours to get back)
- $\approx$ 5 racks go wonky (40-80 machines see 50% packetloss)
- $\approx$ 8 network maintenances (4 might cause $\approx$ 30-minute random connectivity losses)

Jeffrey Dean (Google): Joys of Real Hardware (cnt.)

- $\approx$ 12 router reloads (takes out DNS and external vips for a couple minutes)
- $\approx$ 3 router failures (have to immediately pull traffic for an hour)
- $\approx$ dozens of minor 30-second blips for dns
- $\approx$ 1000 individual machine failures
- $\approx$ thousands of hard drive failures slow disks, bad memory, misconfigured machines, flaky machines, etc.
- Long distance links: wild dogs, sharks, dead horses, drunken hunters, etc.

Big Data

- EMC sponsored IDC study estimates the Digital Universe to be 4.4 Zettabytes (4 400 000 000 TB) in 2013
- The amount of data is estimated to grow to 44 Zettabytes by 2020
- Data comes from: Video, digital images, sensor data, biological data, Internet sites, social media, Internet of Things, . . .
- Some examples:
  - Netflix is collecting 1 PB of data per month from its video service user behaviour
  - Rovio is collecting in the order of 1 TB of data per day of games logs
- The problem of such large data massed, termed Big Data calls for new approaches to analyzing these data masses

Example: Simple Word Search

- Example: Suppose you need to search for a word in a 2TB worth of text that is found only once in the text mass using a compute cluster
- Assuming 100MB/s read speed, in the worst case reading all data from a single 2TB disk takes $\approx$ 5.5 hours
- If 100 hard disks can be used in parallel, the same task takes less than four minutes
- Scaling using hard disk parallelism is one of the design goals of scalable batch processing in the cloud

Scalable Batch Processing

- For large enough data sets we must use hard disk parallelism to do processing in reasonable time
- MapReduce is a programming paradigm developed at Google for scalable batch processing, initially developed to compute the Web search indexes for Google
- The Apache Spark system is a new framework which is currently replacing MapReduce as the distributed Big Data processing system of choice
- A high performance fault tolerant distributed file system is needed to support either Spark or MapReduce, for example Google GFS, or its open source clone Hadoop distributed file system (HDFS)
- Apache Spark framework will be used in the home assignments of the course