Big Data Computing

The Apollo Guidance Computer (AGC)

The computer installed on each command and lunar module of all the Apollo program's missions

The Apollo Guidance Computer (AGC)

The computer installed on each command and lunar module of all the Apollo program's missions

A few numbers:

- ~2 MHz CPU clock frequency
- 16-bit architecture
- 3,840 bytes of main memory (RAM)
- 69,120 bytes of non-volatile read-only memory (ROM)

The Apollo Guidance Computer (AGC)

The computer installed on each command and lunar module of all the Apollo program's missions

A few numbers:

- ~2 MHz CPU clock frequency
- 16-bit architecture
- 3,840 bytes of main memory (RAM)
- 69,120 bytes of non-volatile read-only memory (ROM)

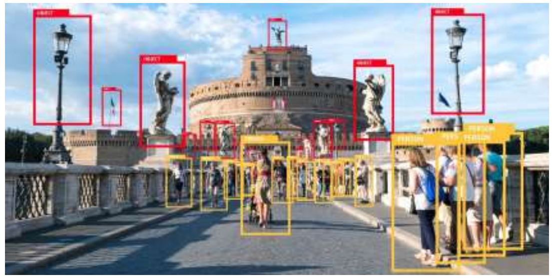
All the running software was written in AGC assembly language, now also available on [GitHub]

Almost 55 Years Have Passed…

… And The World Has Changed

… And The World Has Changed

… And The World Has Changed



AGC vs. Our Smartphone

- Most recent smartphones have
  - >3 GHz CPU clock frequency
  - 4÷16 GB of RAM
  - 64÷1000 GB of internal storage (**don't** call it ROM!)

AGC vs. Our Smartphone

- Most recent smartphones have
  - >3 GHz CPU clock frequency
  - 4÷16 GB of RAM
  - 64÷1000 GB of internal storage (**don't** call it ROM!)

~3 orders of magnitude faster (~1,000x)

~6÷7 orders of magnitude larger RAM and internal storage (up to 10,000,000x)

A Side Note on Units

Prefixes for multiples of bits (bit) or bytes (B)

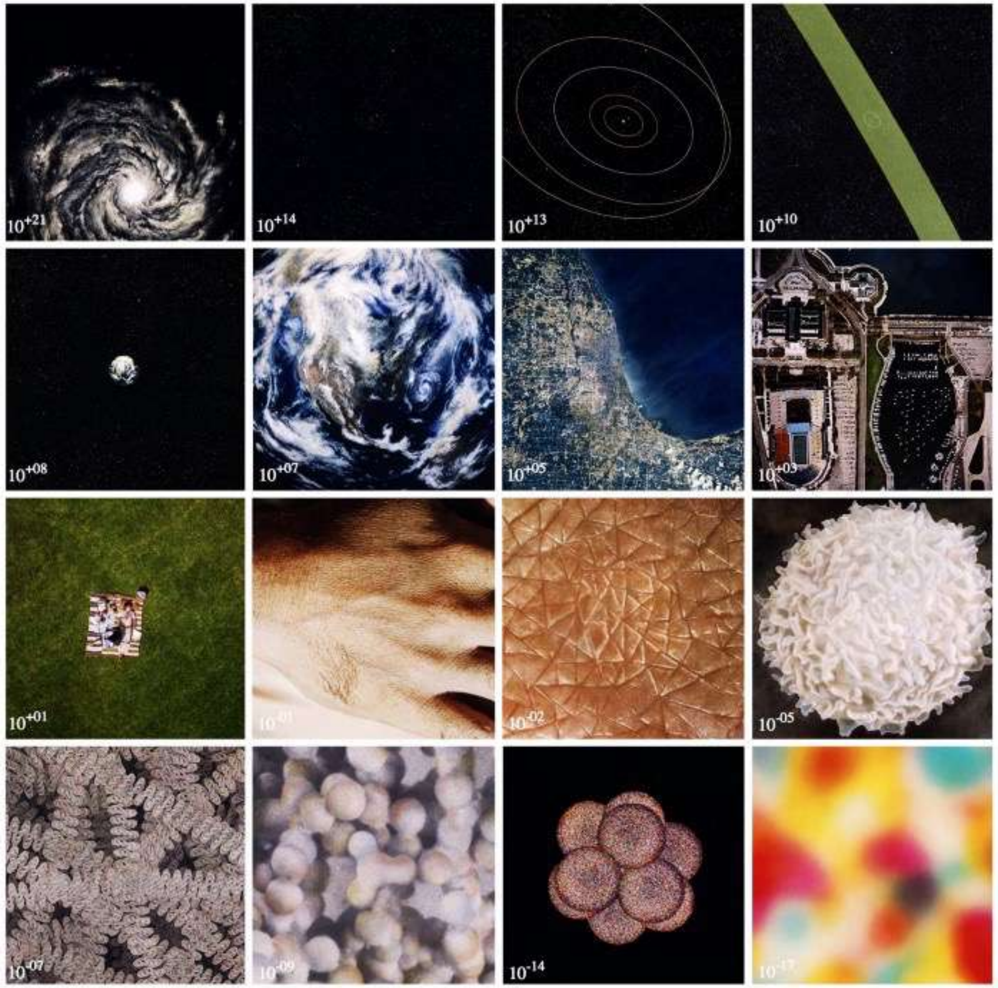
Decimal				Binary					
Value		SI		Value		IEC		JEDEC	
1000	10^3	k	kilo	1024	2^{10}	Ki	kibi	K	kilo
1000^2	10^6	M	mega	1024^2	2^{20}	Mi	mebi	M	mega
1000^3	10^9	G	giga	1024^3	2^{30}	Gi	gibi	G	giga
1000^4	10^{12}	T	tera	1024^4	2^{40}	Ti	tebi	–	
1000^5	10^{15}	P	peta	1024^5	2^{50}	Pi	pebi	–	
1000^6	10^{18}	E	exa	1024^6	2^{60}	Ei	exbi	–	
1000^7	10^{21}	Z	zetta	1024^7	2^{70}	Zi	zebi	–	
1000^8	10^{24}	Y	yotta	1024^8	2^{80}	Yi	yobi	–	

Orders of Magnitude

source: https://www.youtube.com/watch?v=Ww4gYNrOkkg

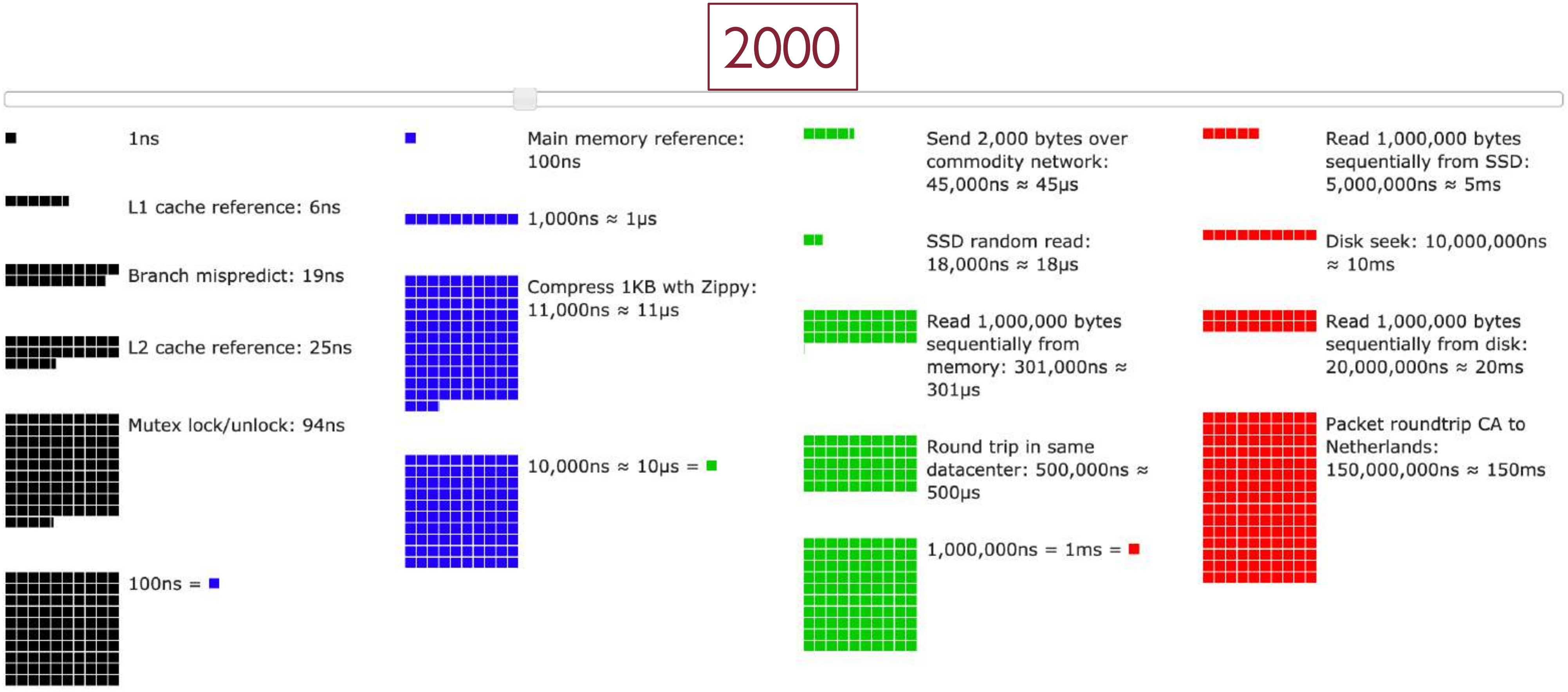
$10^0 = 1$

Orders of Magnitude



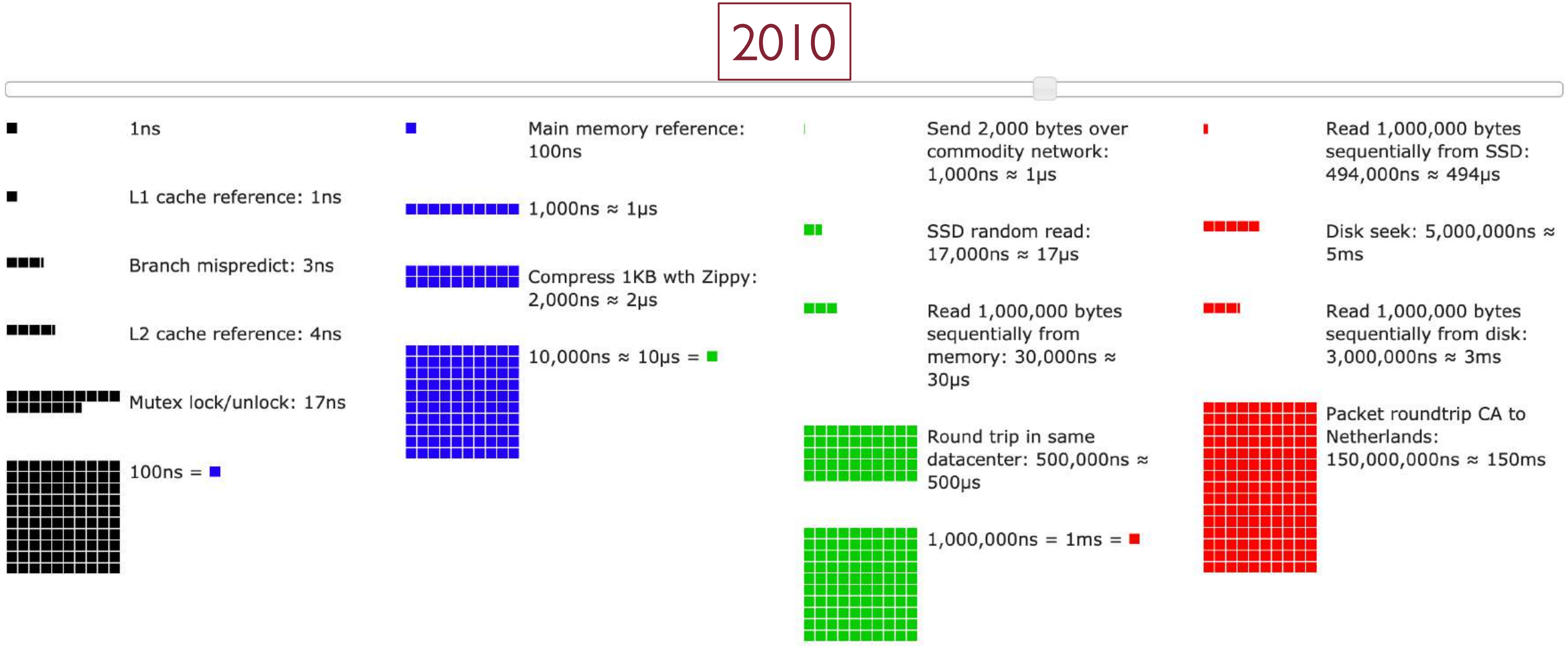
Numbers Every Computer Scientist Should Know

Colin Scott's updated and interactive version of Jeff Dean's previous one



Numbers Every Computer Scientist Should Know

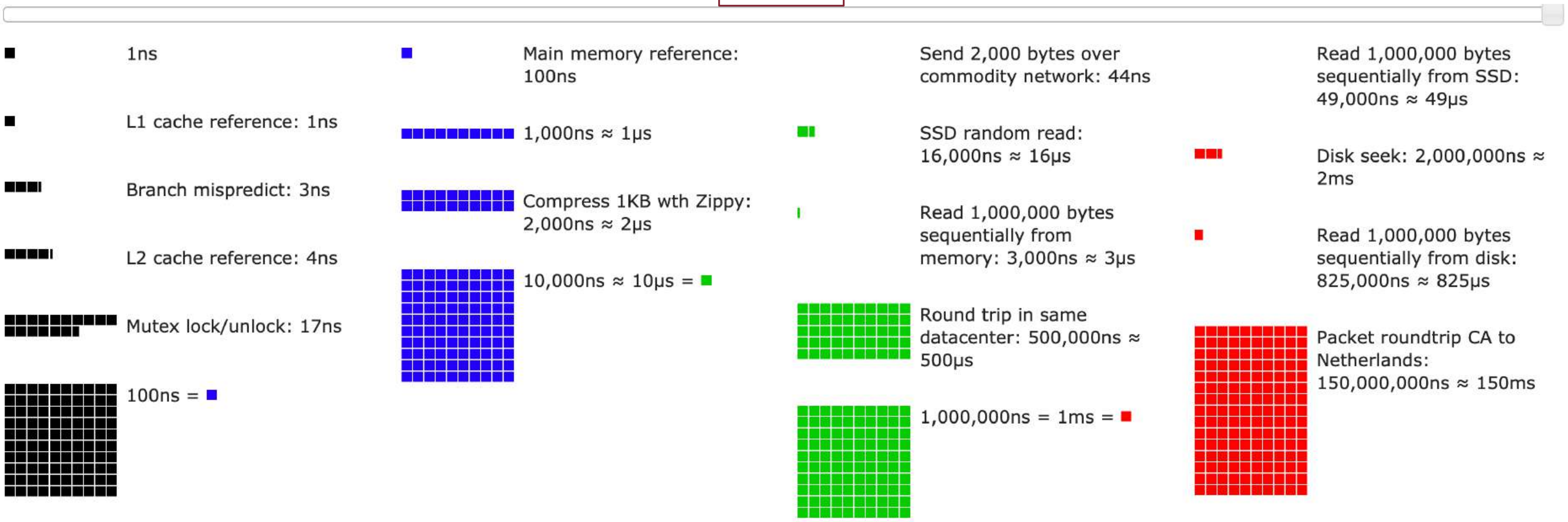
[Colin Scott]'s updated and interactive version of [Jeff Dean]'s previous one



Numbers Every Computer Scientist Should Know

Colin Scott's updated and interactive version of Jeff Dean's previous one

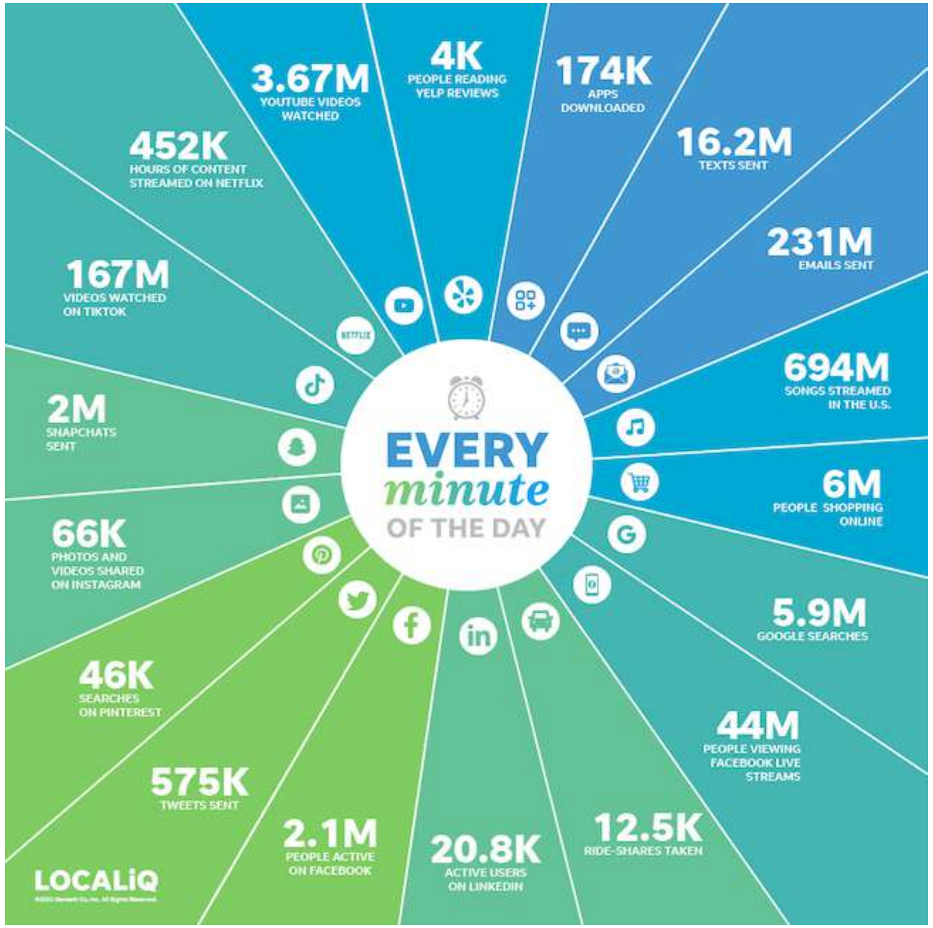
2020



The Information Technology (IT) Revolution

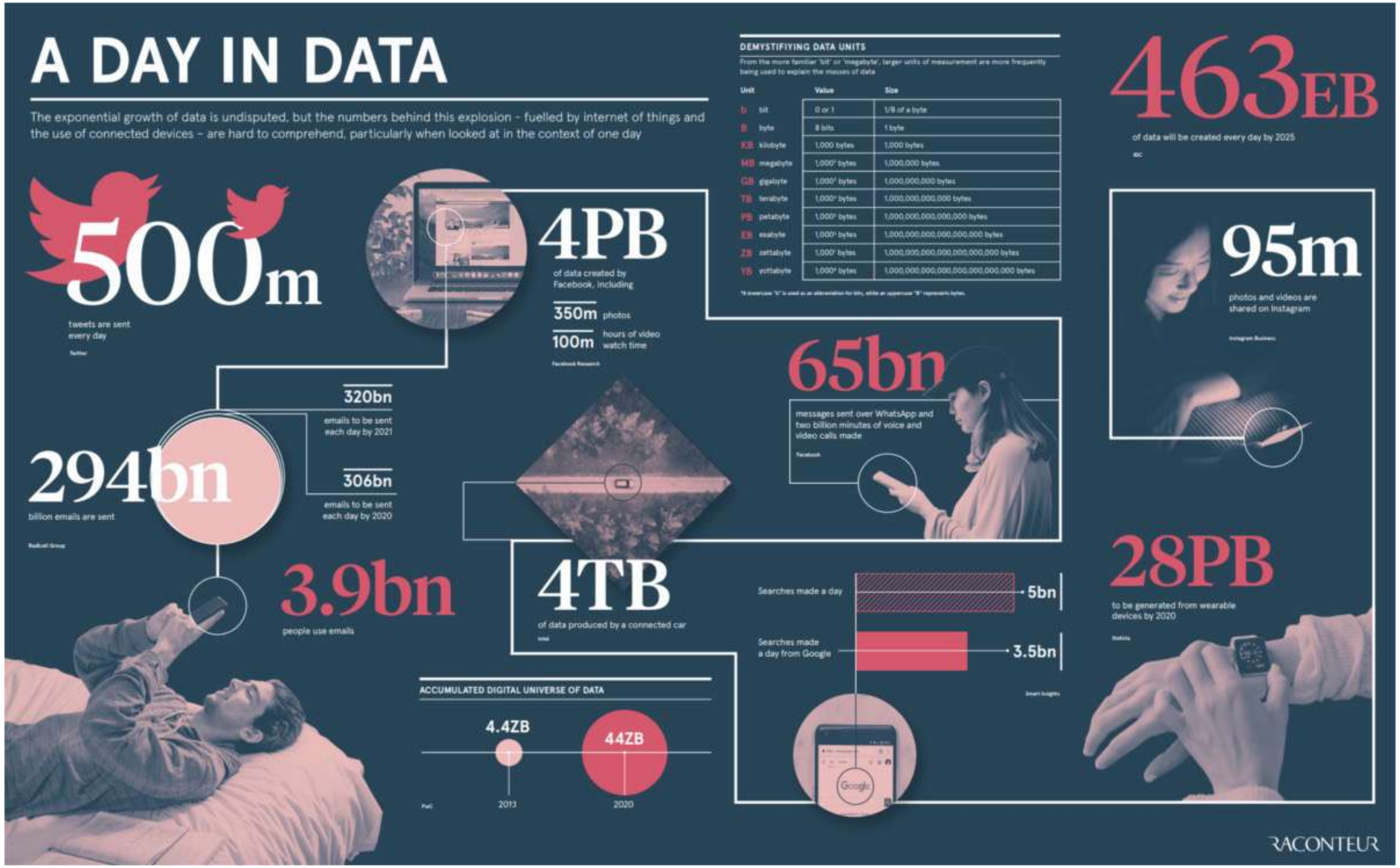
- Started almost 60 years ago and still rocketing
- Driven by:
  - Science/Engineering
  - Business
  - Society

What Happens on the Internet in 1 Minute?



source: LocaliQ

How Much Data is Generated Each Day?



source: VisualCapitalist

What is Big Data?

- Sometimes a buzzword yet describing an actual phenomenon

What is Big Data?

- Sometimes a buzzword yet describing an actual phenomenon
- 4V's (sometimes, 5, 6 or even 7!)

What is Big Data?

- Sometimes a buzzword yet describing an actual phenomenon
- 4 V's (sometimes, 5, 6 or even 7!)
  - Volume → very large amount of data (orders of TB or PB)

What is Big Data?

- Sometimes a buzzword yet describing an actual phenomenon
- 4V's (sometimes, 5, 6 or even 7!)
  - Volume → very large amount of data (orders of TB or PB)
  - Variety → different formats of data: structured (relational tables), semi-structured (JSON files), and unstructured (text/audio/video)

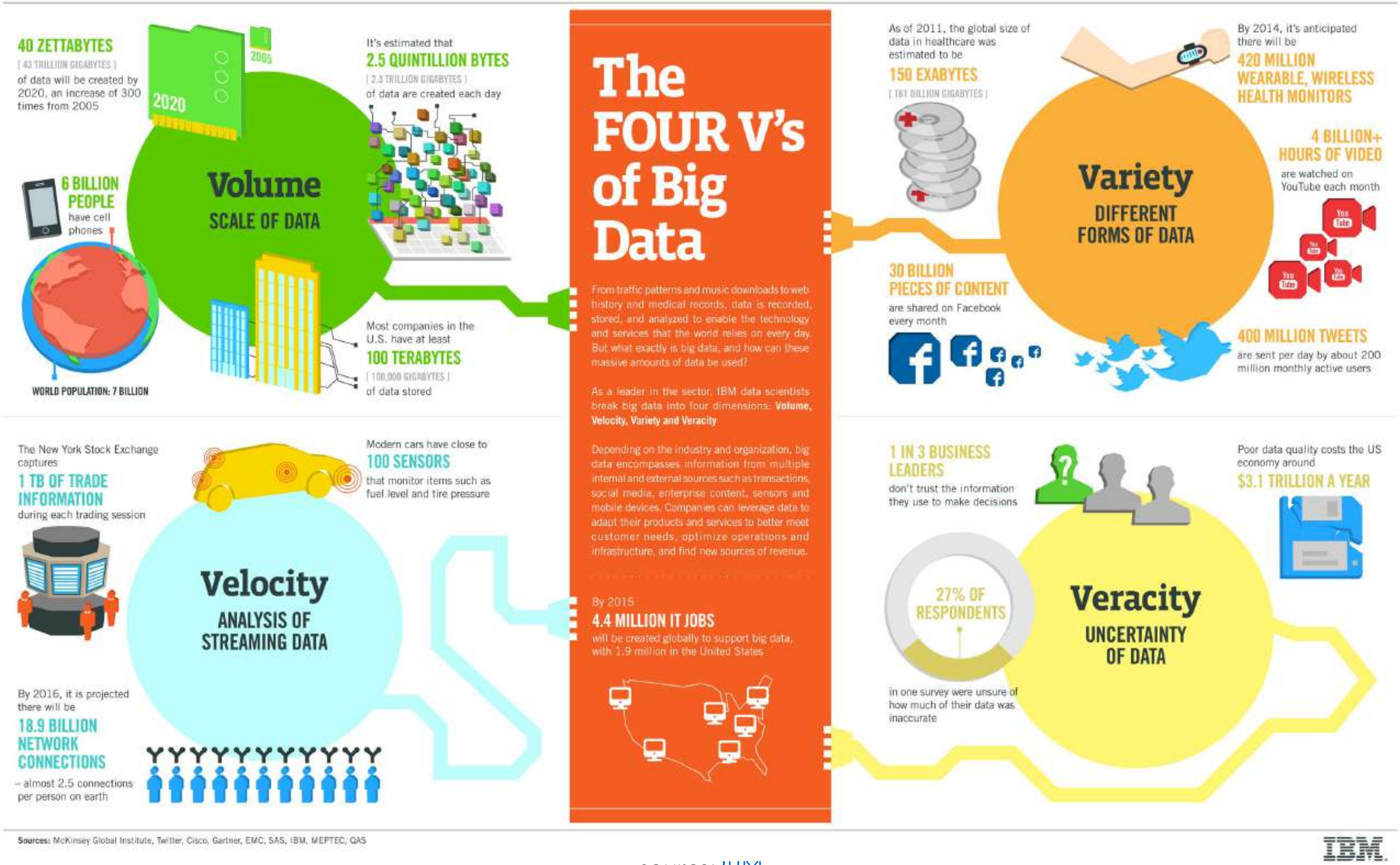
What is Big Data?

- Sometimes a buzzword yet describing an actual phenomenon
- 4V's (sometimes, 5, 6 or even 7!)
  - Volume → very large amount of data (orders of TB or PB)
  - Variety → different formats of data: structured (relational tables), semi-structured (JSON files), and unstructured (text/audio/video)
  - Velocity → insane speed at which data is generated (e.g., Twitter stream)

What is Big Data?

- Sometimes a buzzword yet describing an actual phenomenon
- 4 V's (sometimes, 5, 6 or even 7!)
  - Volume → very large amount of data (orders of TB or PB)
  - Variety → different formats of data: structured (relational tables), semi-structured (JSON files), and unstructured (text/audio/video)
  - Velocity → insane speed at which data is generated (e.g., Twitter stream)
  - Veracity → reliability of the data used to drive decision processes

The 4 V's of Big Data



source: IBM

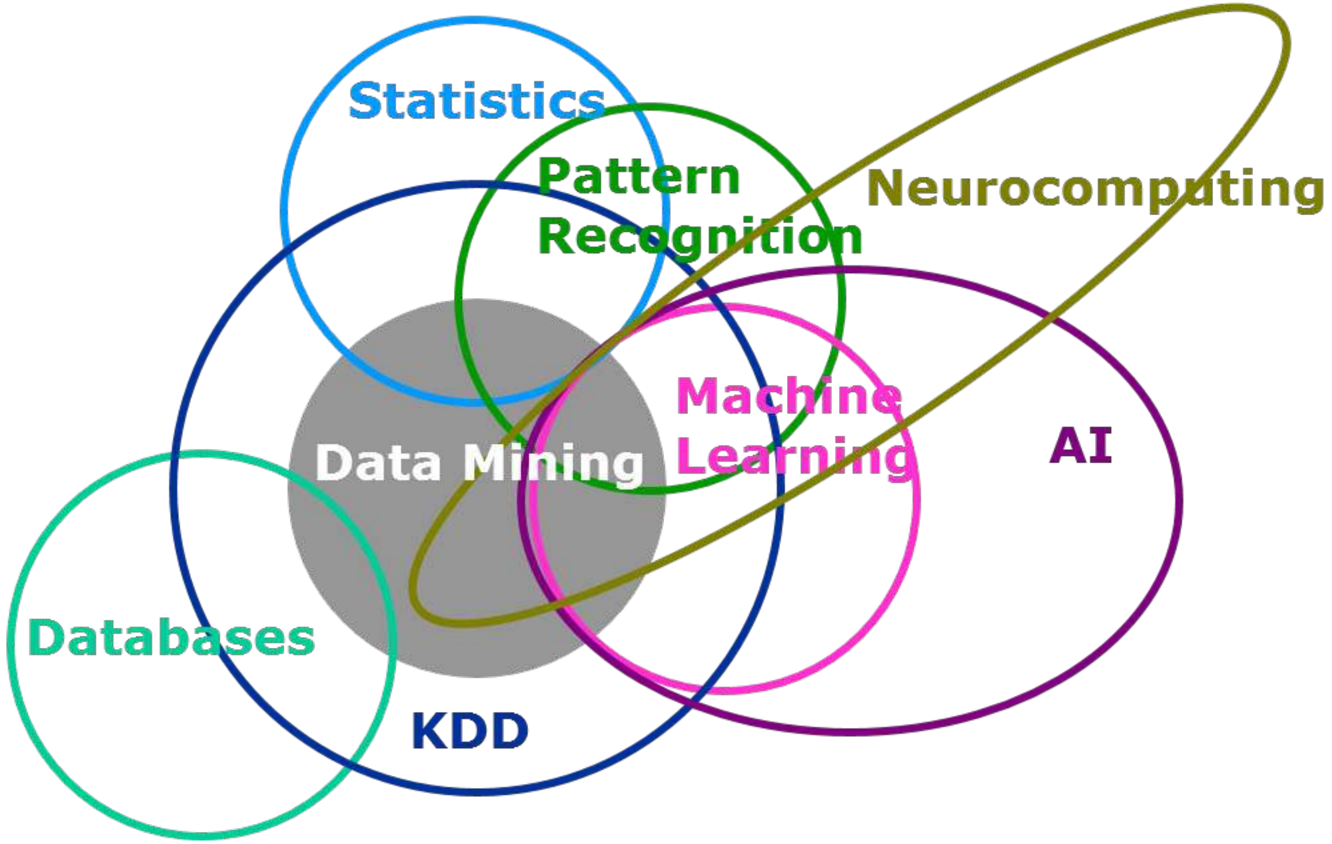
The Value of Big Data

- Extracting knowledge from data is incredibly valuable
  - 5 out of 6 of the biggest companies in the world are "data companies"

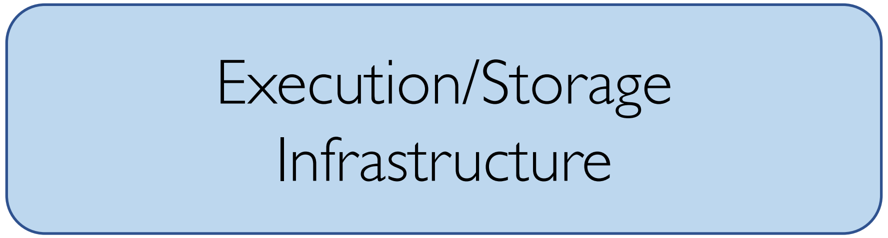
The Value of Big Data

- Extracting knowledge from data is incredibly valuable
  - 5 out of 6 of the biggest companies in the world are "data companies"
- To get the most value out of it, data has to be:
  - Stored
  - Managed
  - Analyzed

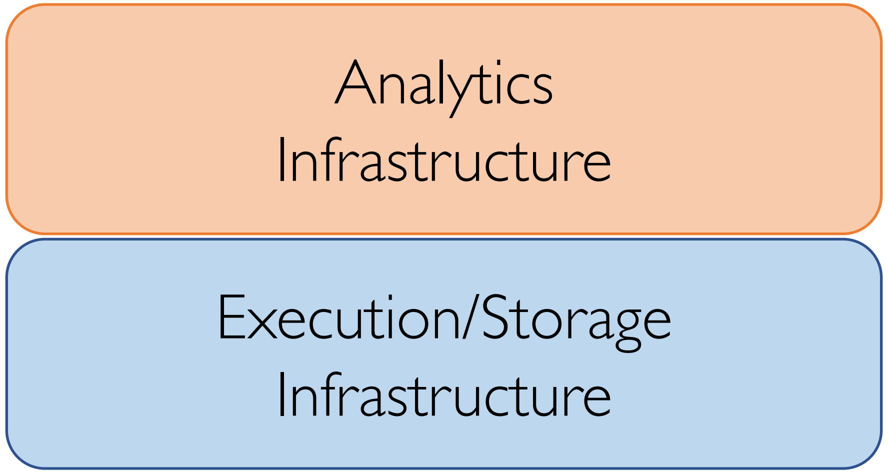
Big Data Analysis: Landscape



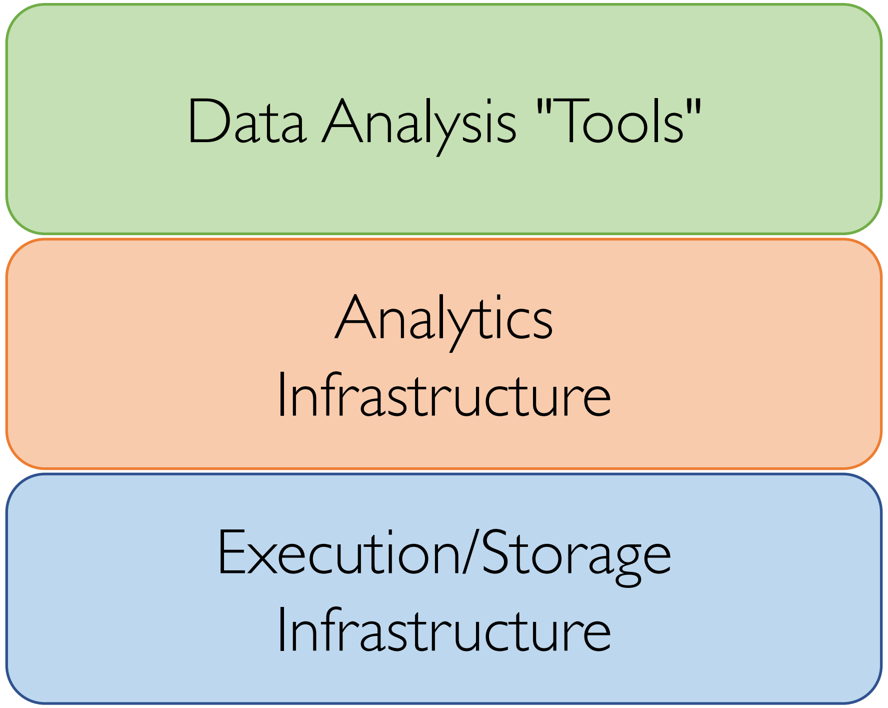
Big Data Analysis Stack



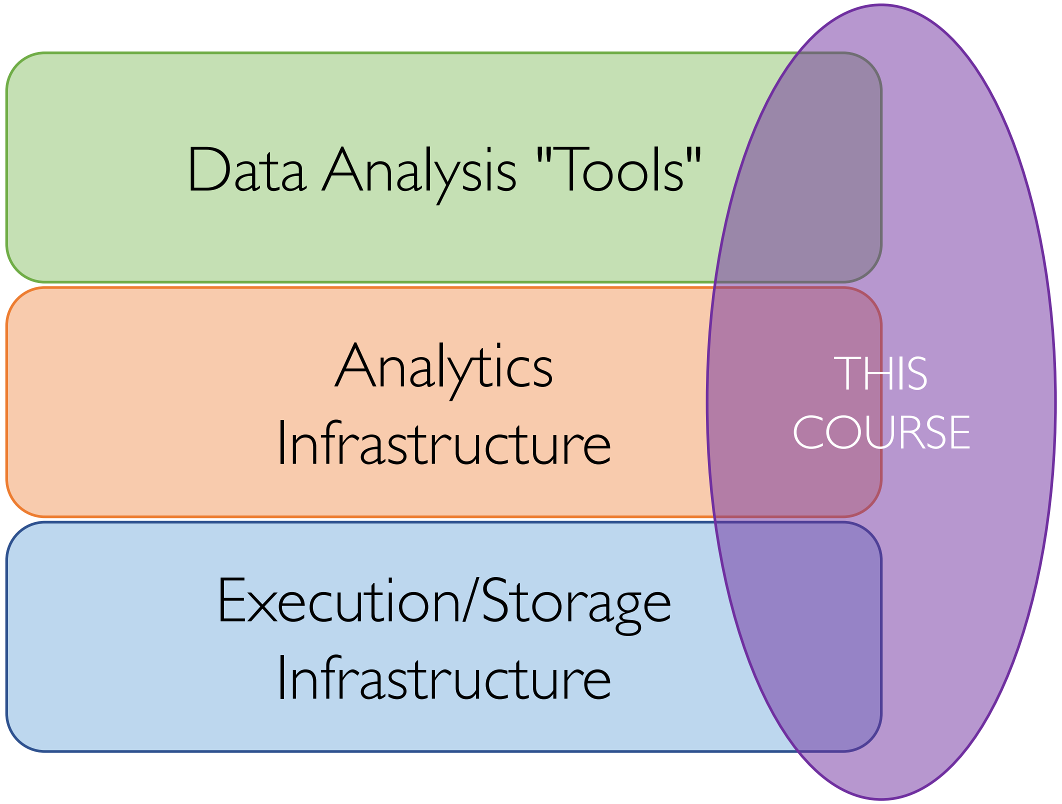
Big Data Analysis Stack



Big Data Analysis Stack



Big Data Analysis Stack



What Will We Learn?

- To extract knowledge from different types of data
  - High-dimensional
  - Unlabeled/Labeled
  - Graph-based
  - Infinite/never-ending streams

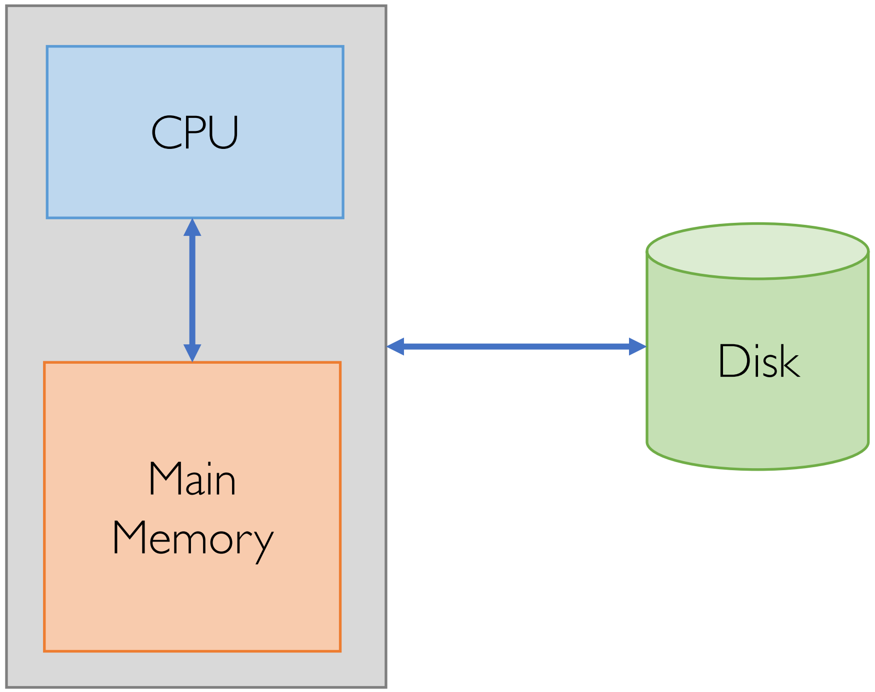
What Will We Learn?

- To use different models of computation
  - MapReduce
  - Streams and online algorithms
  - Single machine in-memory

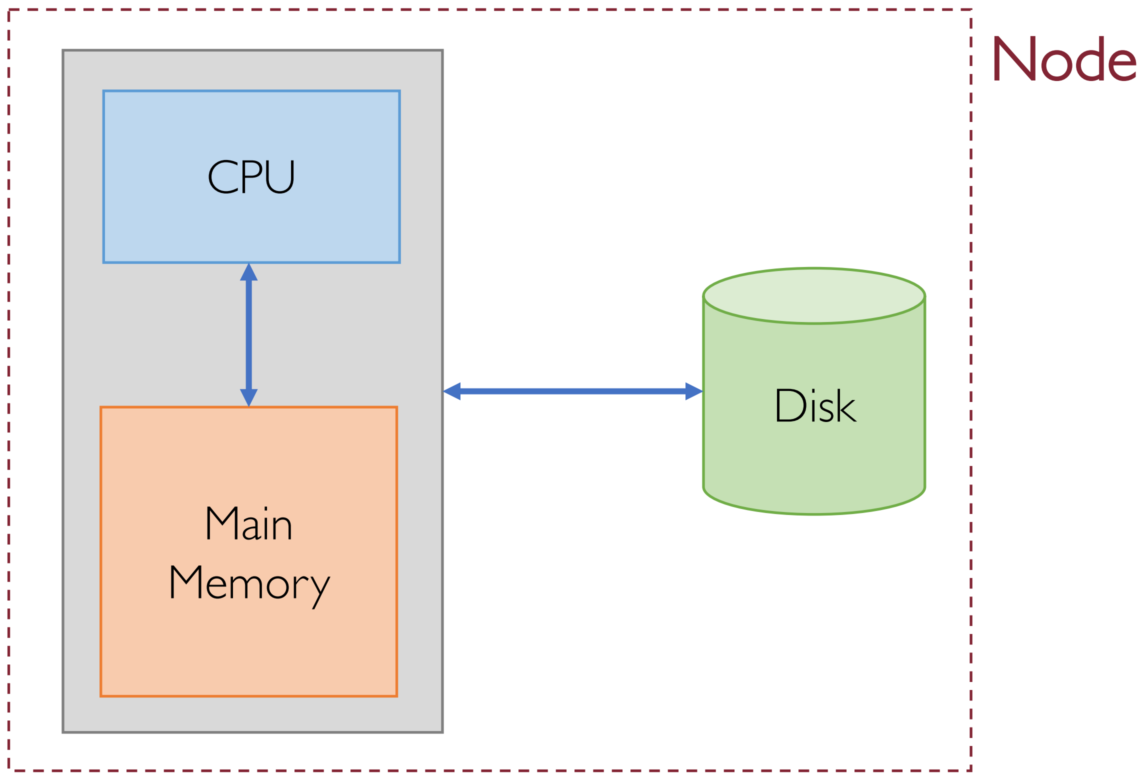
What Will We Learn?

- To apply big data analysis to actually solve real-world problems
  - Clustering
  - Predictive Analysis
  - Recommender Systems
  - Graph Analysis
  - Stream Processing
  - …

The Single-Node Architecture

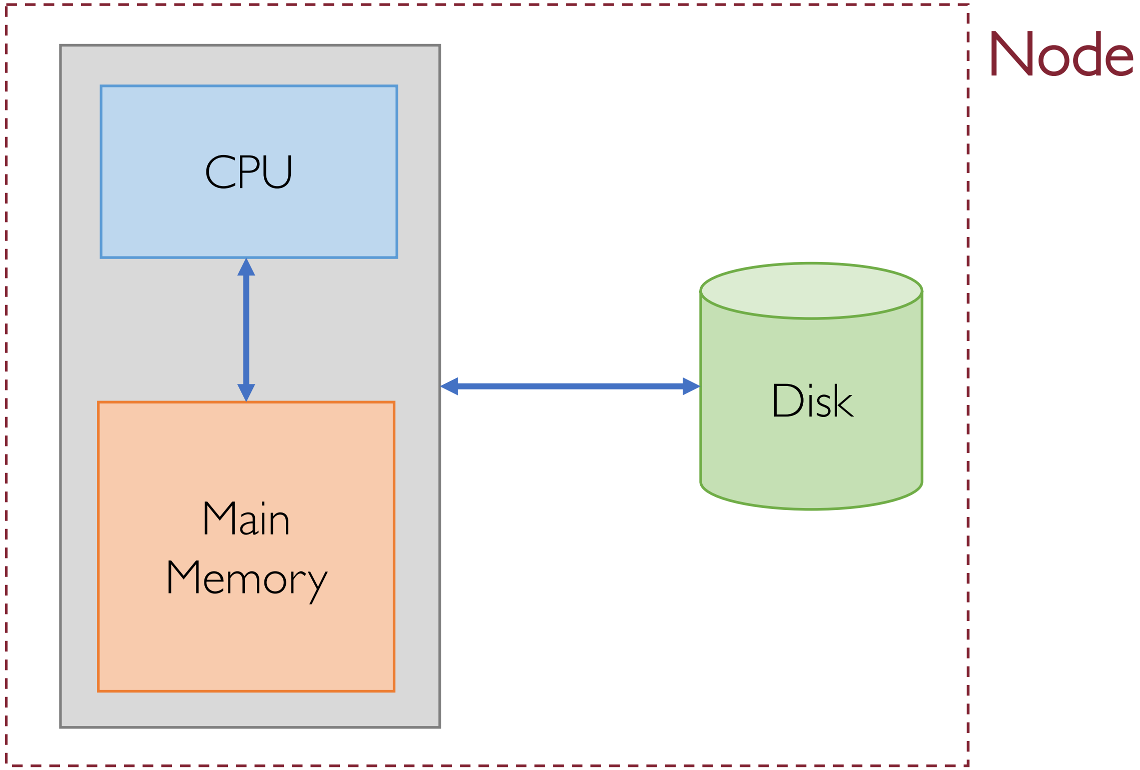


The Single-Node Architecture



The Single-Node Architecture

Everything is ok as long as data fits entirely into main memory
(few accesses to the disk are still tolerated)



Example: Google (Toy) Index

- Google has crawled 50 million web pages (a tiny fraction of the Web!)
- The average size of each web page (HTML only) is ~100 KB
- The total size of the index will be

Example: Google (Toy) Index

- Google has crawled 50 million web pages (a tiny fraction of the Web!)
- The average size of each web page (HTML only) is ~100 KB
- The total size of the index will be

$5\text{x}10^7 \times 10^5$ bytes = $5\text{x}10^{12}$ bytes = **5 TB**

Example: Google (Toy) Index

- Google has crawled 50 million web pages (a tiny fraction of the Web!)
- The average size of each web page (HTML only) is ~100 KB
- The total size of the index will be

$5 \times 10^7 \times 10^5$ bytes $= 5 \times 10^{12}$ bytes $=$ **5 TB**

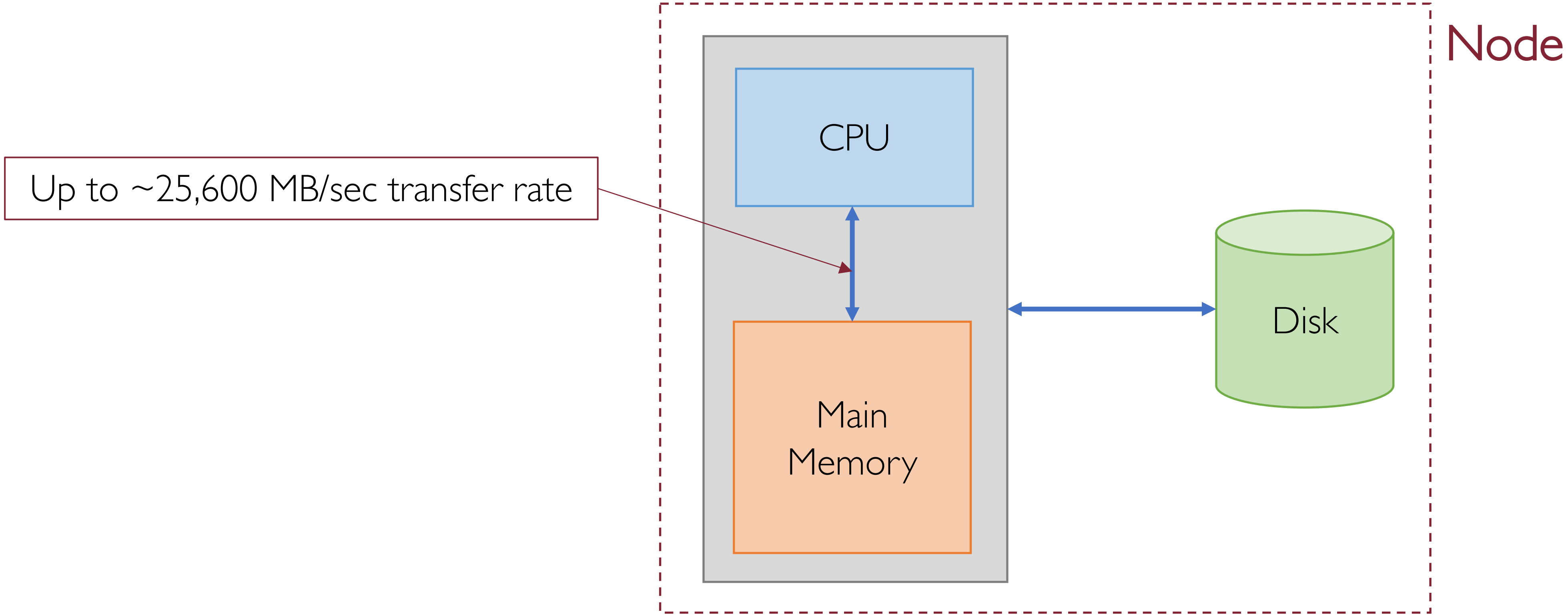
Main Memory

Example: Google (Toy) Index

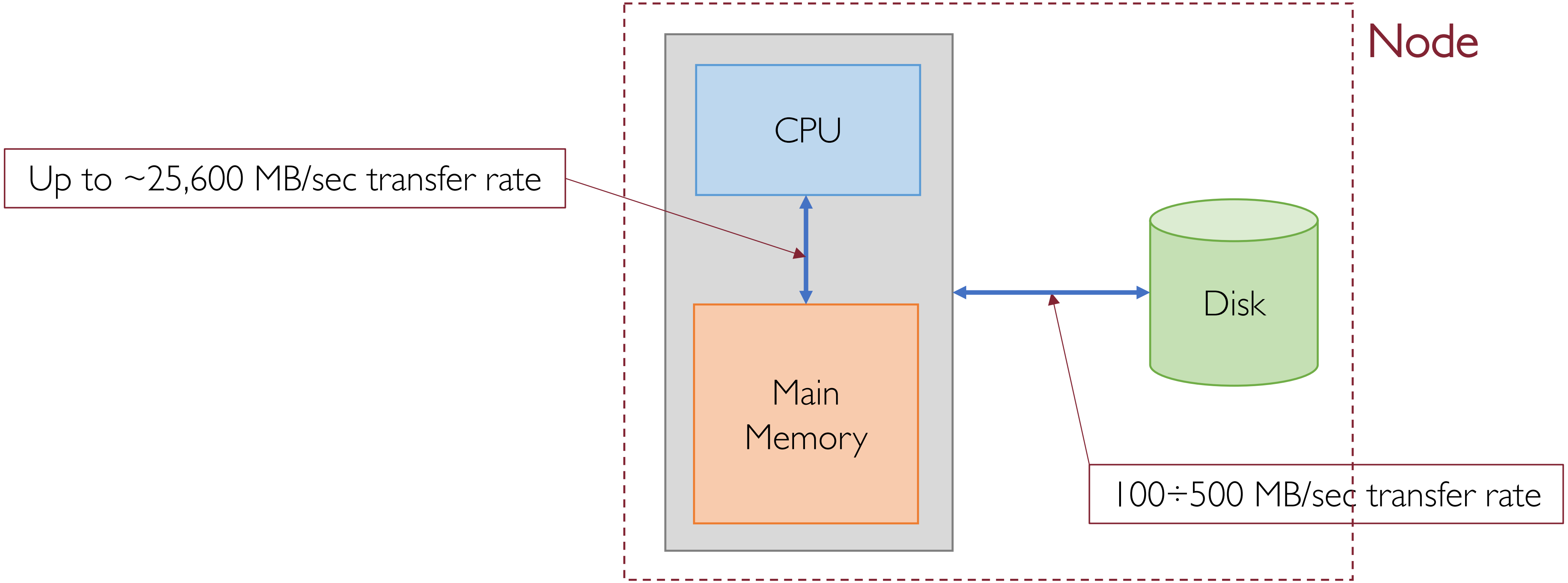
- Google has crawled 50 million web pages (a tiny fraction of the Web!)
- The average size of each web page (HTML only) is ~100 KB
- The total size of the index will be

$5 \times 10^7 \times 10^5$ bytes $= 5 \times 10^{12}$ bytes $=$ **5 TB**

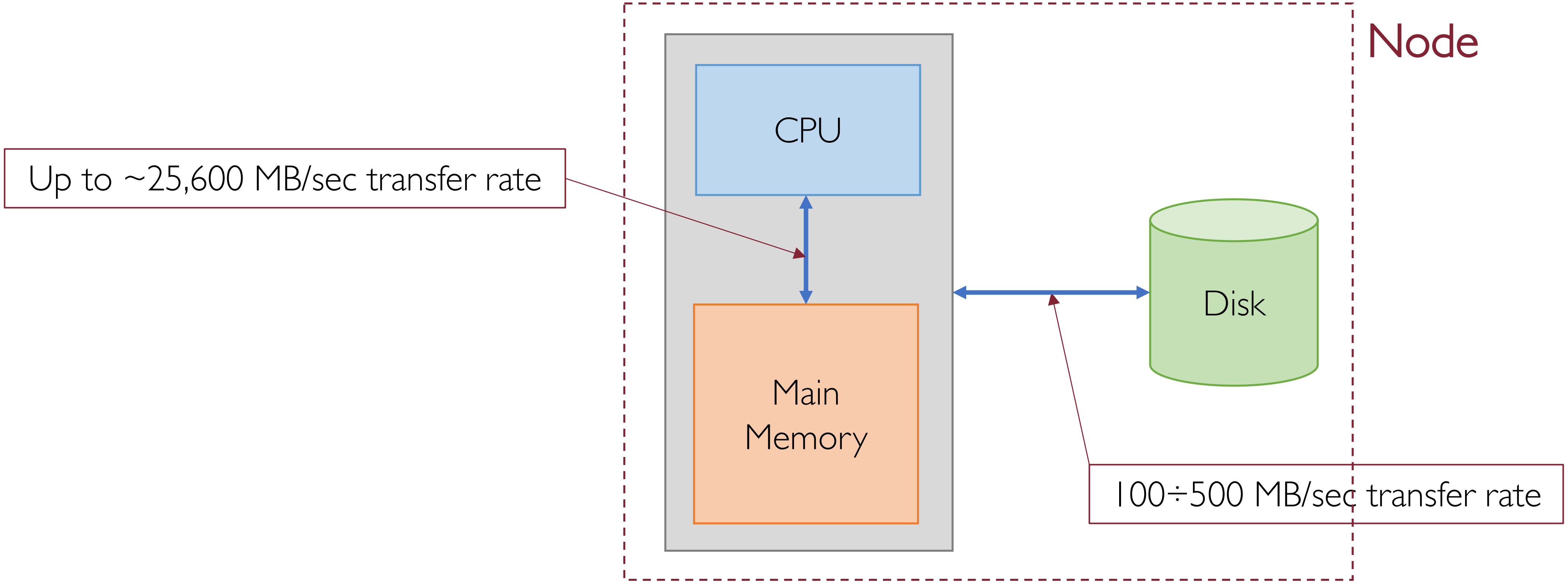
The Single-Node Architecture



The Single-Node Architecture



The Single-Node Architecture



2 orders of magnitude difference between data transfer rate

Example: Google (Toy) Index

- Assuming the disk transfer rate is 100 MB/sec the total time to read the entire index will be:

5×10^{12} bytes/10^8 bytes/sec = 5×10^4 seconds ~**14 hours**

Example: Google (Toy) Index

- Assuming the disk transfer rate is 100 MB/sec the total time to read the entire index will be:

5×10^{12} bytes/10^8 bytes/sec = 5×10^4 seconds ~**14 hours**

- More than half a day to just read the index, without even do any computation on it!

Example: Google (Toy) Index

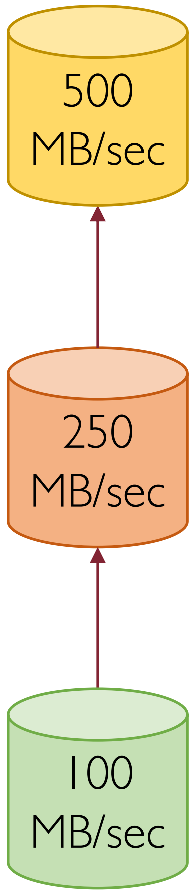
- Assuming the disk transfer rate is 100 MB/sec the total time to read the entire index will be:

$5\text{x}10^{12}$ bytes/10^{8} bytes/sec = $5\text{x}10^{4}$ seconds ~**14 hours**

- More than half a day to just read the index, without even do any computation on it!
- Single-node architecture is clearly not enough here
  - Scaling Up vs. Scaling Out

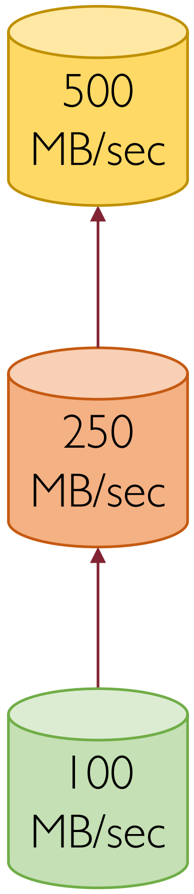
Scaling Up/Vertical Scaling

- Buy a more performing disk (e.g., 250 or 500 MB/sec transfer rate)



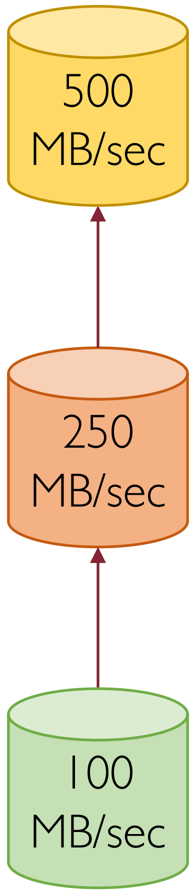
Scaling Up/Vertical Scaling

- Buy a more performing disk (e.g., 250 or 500 MB/sec transfer rate)
- PRO
  - Easiest solution



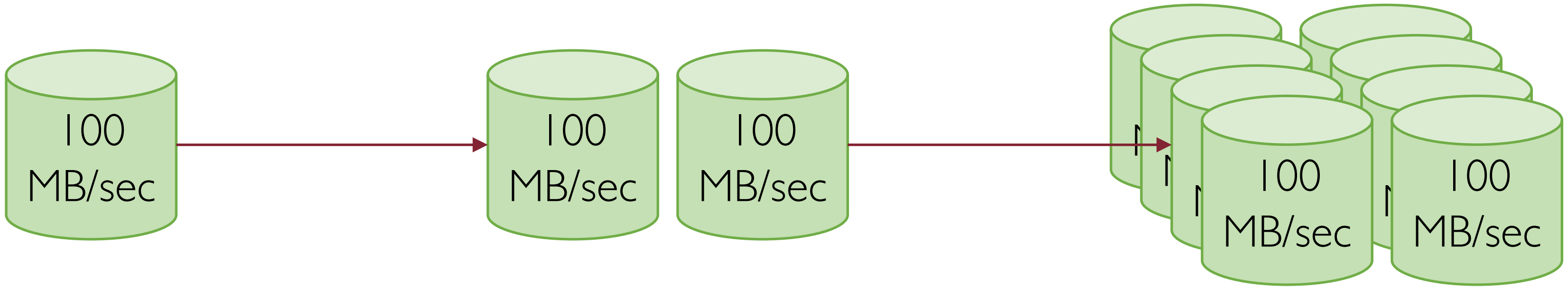
Scaling Up/Vertical Scaling

- Buy a more performing disk (e.g., 250 or 500 MB/sec transfer rate)
- PRO
  - Easiest solution
- CON
  - Improvement is physically-limited (e.g., 2.5x or 5x)
  - Expensive



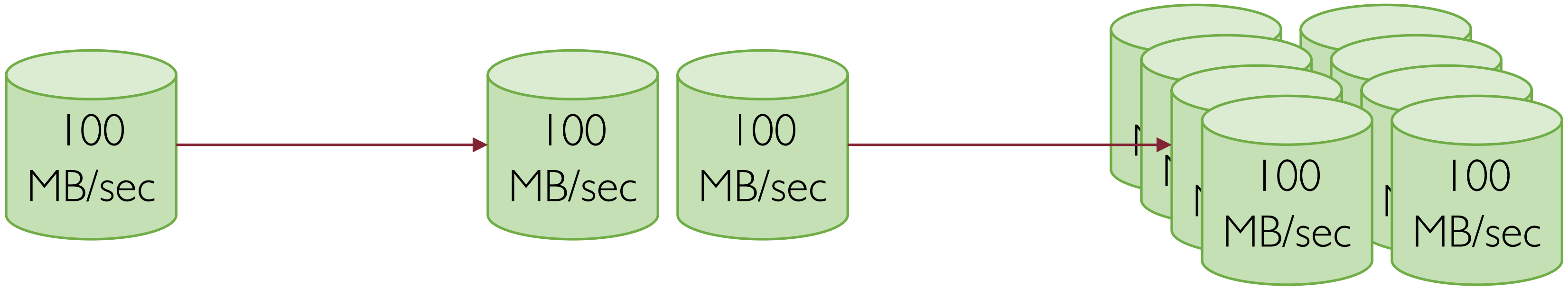
Scaling Out/Horizontal Scaling

- Buy a set of commodity "cheap" disks and let them work in parallel



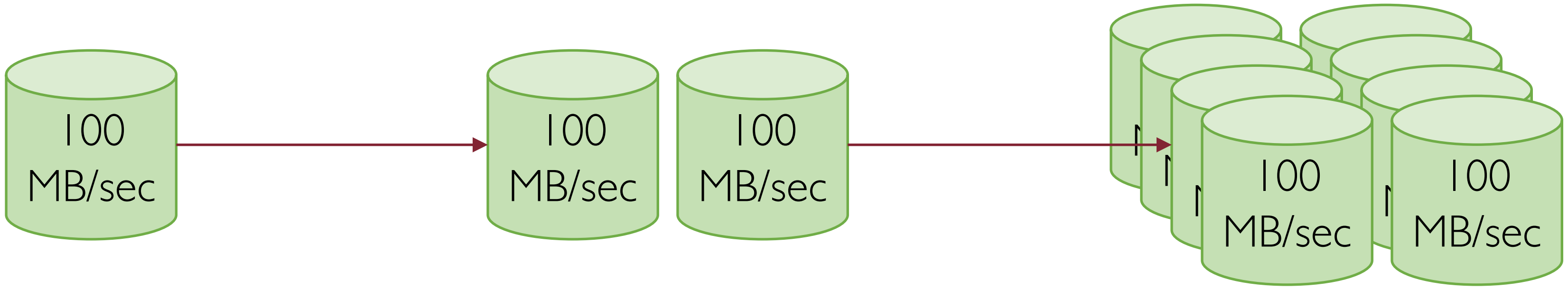
Scaling Out/Horizontal Scaling

- Buy a set of commodity "cheap" disks and let them work in parallel
- PRO
  - Flexibility (improvement is not bound apriori, just add new disks as needed)



Scaling Out/Horizontal Scaling

- Buy a set of commodity "cheap" disks and let them work in parallel
- PRO
  - Flexibility (improvement is not bound apriori, just add new disks as needed)
- CON
  - Extra overhead required to manage parallel work



Cluster Architecture

- Computing architecture based on the scaling out principle

Cluster Architecture

- Computing architecture based on the scaling out principle
- A lot of commodity nodes communicating with each other

Cluster Architecture

- Computing architecture based on the scaling out principle
- A lot of commodity nodes communicating with each other
- Each group of 16÷64 nodes is arranged in a so-called rack

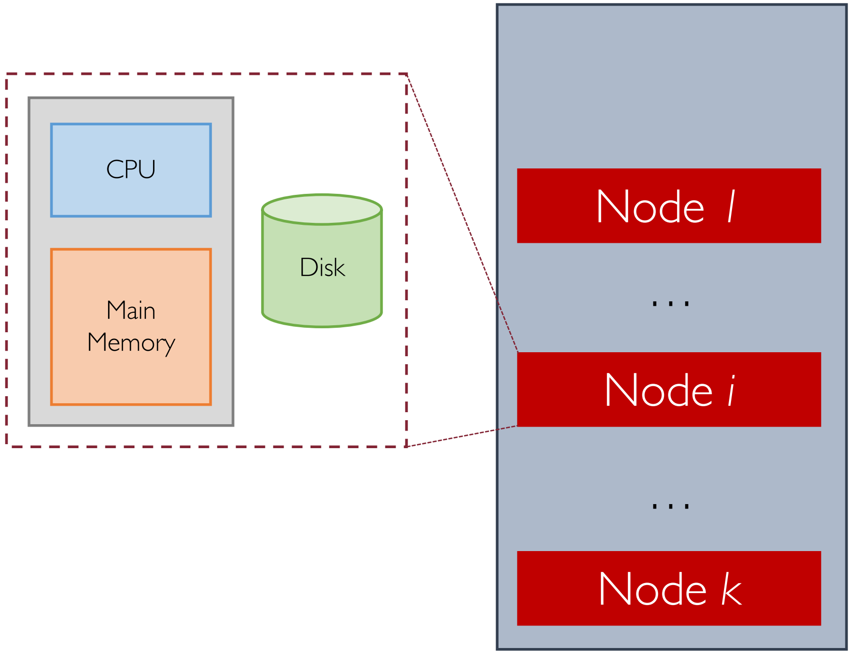
Cluster Architecture

- Computing architecture based on the scaling out principle
- A lot of commodity nodes communicating with each other
- Each group of 16÷64 nodes is arranged in a so-called rack
- A cluster is made of multiple racks

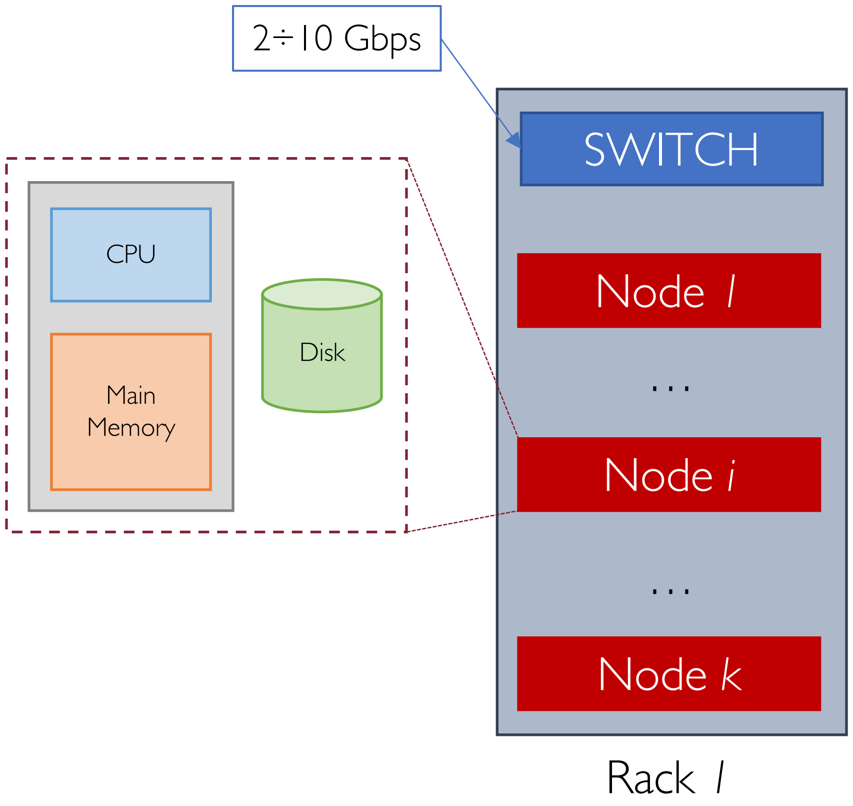
Cluster Architecture

- Computing architecture based on the scaling out principle
- A lot of commodity nodes communicating with each other
- Each group of 16÷64 nodes is arranged in a so-called rack
- A cluster is made of multiple racks
- Network switches enabling node communication
  - 1 Gbps (inter-rack)
  - 2÷10 Gbps (intra-rack)

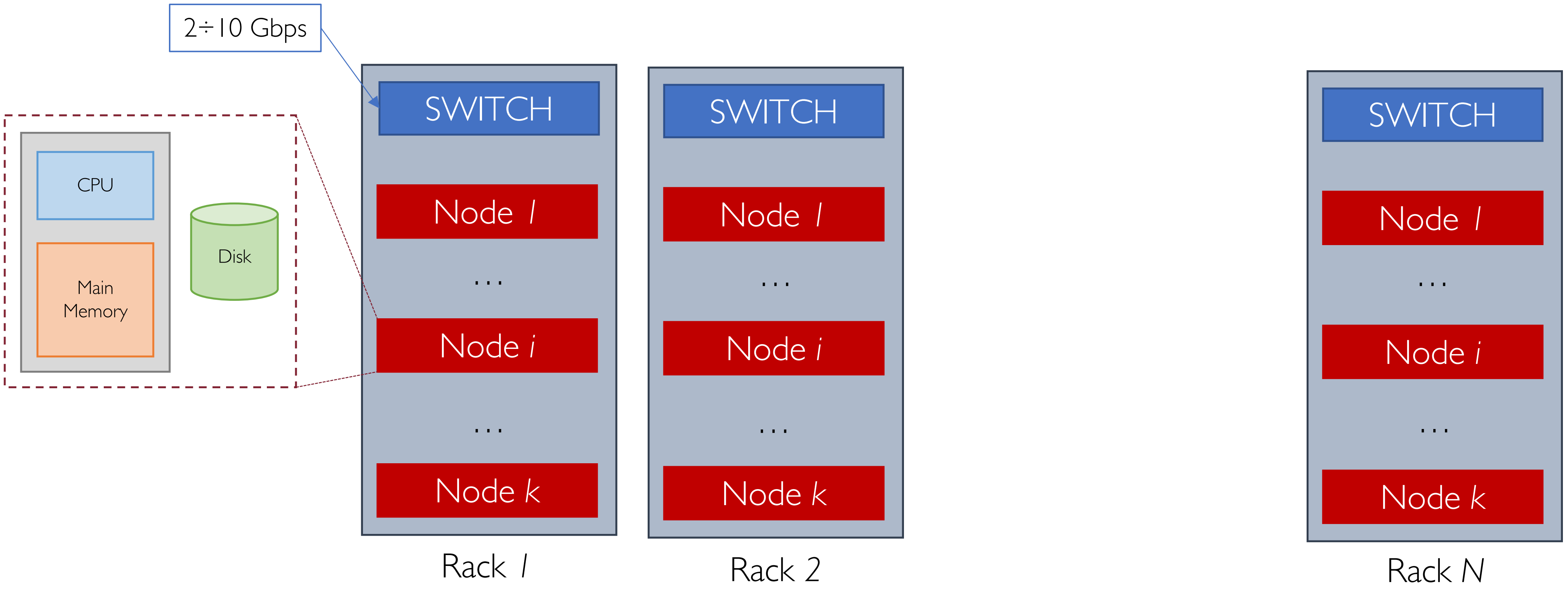
Cluster Architecture



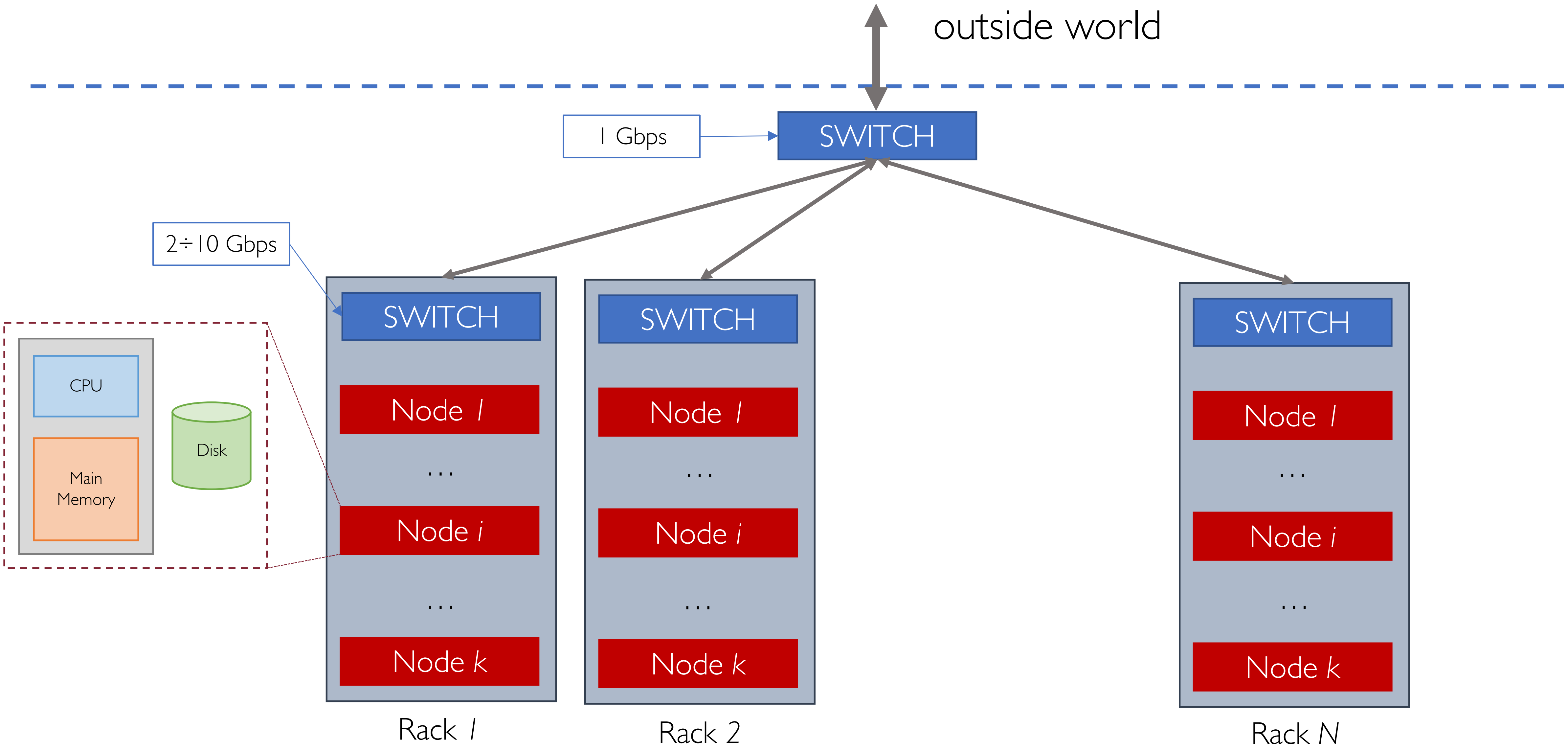
Cluster Architecture



Cluster Architecture



Cluster Architecture



Cluster Architecture: Challenges

- 3 major **challenges** posed by cluster architecture

Cluster Architecture: Challenges

- 3 major challenges posed by cluster architecture
  - Ensure reliability upon node failure

Cluster Architecture: Challenges

- 3 major challenges posed by cluster architecture
  - Ensure reliability upon node failure
  - Minimize network communication bottleneck

Cluster Architecture: Challenges

- 3 major challenges posed by cluster architecture
  - Ensure reliability upon node failure
  - Minimize network communication bottleneck
  - Ease distributed programming model

Challenge: Node Failure

- Suppose we have a cluster of *N* nodes

Challenge: Node Failure

- Suppose we have a cluster of N nodes
- Each node has a Mean Time To Failure (MTTF) = 3 years ~ 1,000 days

$$\boxed{p = P(\text{node}_i \text{ fails}) = 1/1{,}000 = 0.001}$$

Challenge: Node Failure

- Suppose we have a cluster of N nodes
- Each node has a Mean Time To Failure (MTTF) = 3 years ~ 1,000 days

$$\boxed{p = P(\text{node}_i \text{ fails}) = 1/1{,}000 = 0.001}$$

- Associate with each node a random variable $X_{i,t}$

Challenge: Node Failure

- Suppose we have a cluster of N nodes
- Each node has a Mean Time To Failure (MTTF) = 3 years ~ 1,000 days

$$\boxed{p = P(\text{node}_i \text{ fails}) = 1/1{,}000 = 0.001}$$

- Associate with each node a random variable $X_{i,t}$
  - $X_{i,t} \sim \text{Bernoulli}(p)$ outputs 1 (failure) with probability $p = 0.001$ and 0 (working) with probability $(1\text{-}p) = 0.999$

Challenge: Node Failure

- Suppose we have a cluster of N nodes
- Each node has a Mean Time To Failure (MTTF) = 3 years ~ 1,000 days

$$\boxed{p = P(\text{node}_i \text{ fails}) = 1/1{,}000 = 0.001}$$

- Associate with each node a random variable $X_{i,t}$
  - $X_{i,t} \sim \text{Bernoulli}(p)$ outputs 1 (failure) with probability $p = 0.001$ and 0 (working) with probability $(1\text{-}p) = 0.999$
  - Assume for semplicity p is the same for all nodes and independent from each other

Challenge: Node Failure

What is the expected number of failures in a certain day t,
given that the probability of one machine failing is p?”

Challenge: Node Failure

What is the expected number of failures in a certain day t,
given that the probability of one machine failing is p?"

Under the (simplified) assumption that $X_{i,t}$ are all i.i.d.

$$T = X_{1,t} + X_{2,t} + \ldots + X_{N,t}$$

Challenge: Node Failure

What is the expected number of failures in a certain day t,
given that the probability of <u>one</u> machine failing is p?"

Under the (simplified) assumption that $X_{i,t}$ are all i.i.d.

$$T = X_{1,t} + X_{2,t} + \ldots + X_{N,t}$$

$$T \sim \mathit{Binomial}\ (N, p)$$

Challenge: Node Failure

What is the expected number of failures in a certain day t,
given that the probability of <u>one</u> machine failing is p?"

Under the (simplified) assumption that $X_{i,t}$ are all i.i.d.

$$T = X_{1,t} + X_{2,t} + \ldots + X_{N,t}$$

$$T \sim Binomial\ (N, p)$$

$$E[T] = Np$$

Challenge: Node Failure

- A single-node failure on a day may be a quite a rare event (0.1% chance)

Challenge: Node Failure

- A single-node failure on a day may be a quite a rare event (0.1% chance)
- Things are not so infrequent when we deal with several nodes:
  - 1 (expected) failure per day with $N = 1{,}000$ nodes
  - 1,000 (expected) failures per day with $N = 1{,}000{,}000$ nodes

Challenge: Node Failure

- A single-node failure on a day may be a quite a rare event (0.1% chance)
- Things are not so infrequent when we deal with several nodes:
  - 1 (expected) failure per day with N = 1,000 nodes
  - 1,000 (expected) failures per day with N = 1,000,000 nodes

Q1: How to make data and computation resilient to node failures?

Challenge: Network Bottleneck

- Moving data across nodes both intra- and inter racks may be costly

Challenge: Network Bottleneck

- Moving data across nodes both intra- and inter racks may be costly
- For example, if we have to transfer 10 TB of data at 1 Gbps

$$\boxed{8 \times 10^{13} \text{ bits} \ / \ 1 \times 10^{9} \text{ bit/sec} = 8 \times 10^{4} \text{ secs} \sim 1 \text{ day}}$$

Challenge: Network Bottleneck

- Moving data across nodes both intra- and inter racks may be costly
- For example, if we have to transfer 10 TB of data at 1 Gbps

$$\boxed{8 \times 10^{13} \text{ bits} / 1 \times 10^{9} \text{ bit/sec} = 8 \times 10^{4} \text{ secs} \sim 1 \text{ day}}$$

Q2: How to minimize data tranfers so as to reduce network communications?

Challenge: Distributed Programming

- Distributed programming can be really complex

Challenge: Distributed Programming

- Distributed programming can be really complex
- Programmers should focus on the (distributed) task rather than dealing with the complexities of the cluster architecture

Challenge: Distributed Programming

- Distributed programming can be really complex
- Programmers should focus on the (distributed) task rather than dealing with the complexities of the cluster architecture

Q3: How to implement algorithms which take advantage of the distributed infrastructure without worrying about its complexities?

Take-Home Message of Today

- Data is generated at an unprecedented rate → Big Data

Take-Home Message of Today

- Data is generated at an unprecedented rate → Big Data
- Extracting knowledge from such big data is incredibly valuable

Take-Home Message of Today

- Data is generated at an unprecedented rate → Big Data
- Extracting knowledge from such big data is incredibly valuable
- Traditional algorithms/techniques often don't scale very well

Take-Home Message of Today

- Data is generated at an unprecedented rate → Big Data
- Extracting knowledge from such big data is incredibly valuable
- Traditional algorithms/techniques often don't scale very well
- There is the need for new "tools" which allow storing, managing, and analyzing big data painlessly