Distributed Systems
CS6421

The End

Prof. Tim Wood

Semester Review

Topic 1 Intro to Distributed Systems and the Cloud

Topic 2 Network Programming (The Internet, Sockets)

Topic 3 Servers and Virtualization (EC2, VMs, Containers)

Topic 4 Networks (SDN, NFV)

Topic 5 Storage and Fault Tolerance (S3, EBS, Dynamo DB)

Topic 6 Ordering, and Consistency (DynamoDB, Riak)

Topic 7 Scalable Web Services and Serverless (Wikipedia, Netflix)

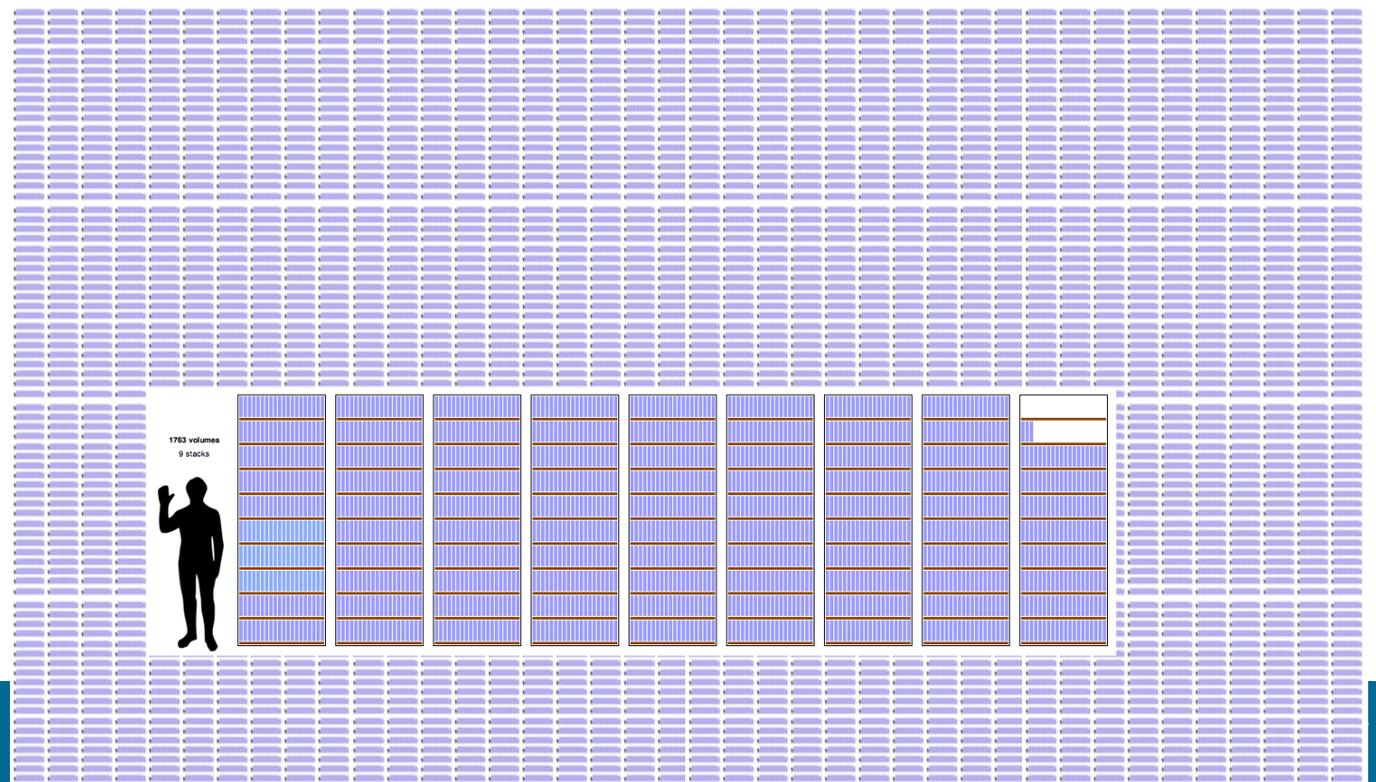
Today … everything else …

Topic 1 Intro to Distributed Systems and the Cloud

Parallelism is hard most of the time

DCs can be a single point of failure for the cloud

Lots of data!

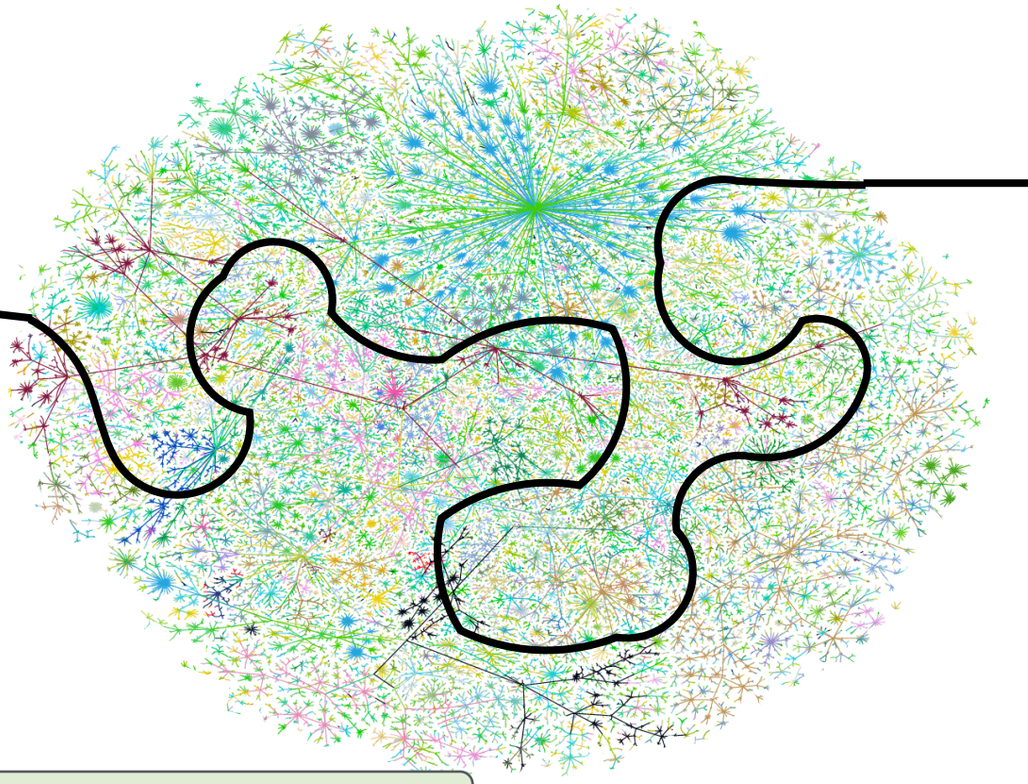


1. 5 + 2 = ?
2. 11 - 2 = ?
3. 14 - 3 = ?
4. -1 + 3 = ?
5. 5*9 = ?
6. 8*8 + 3 = ?

Topic 2 Network Programming (Internet, Sockets)

Networking = lots of layers working together

Protocols are important



161.253.78.16

185.199.109.153

```
GET /timwood/simple.html HTTP/1.1
Host: faculty.cs.gwu.edu
(blank line)
```

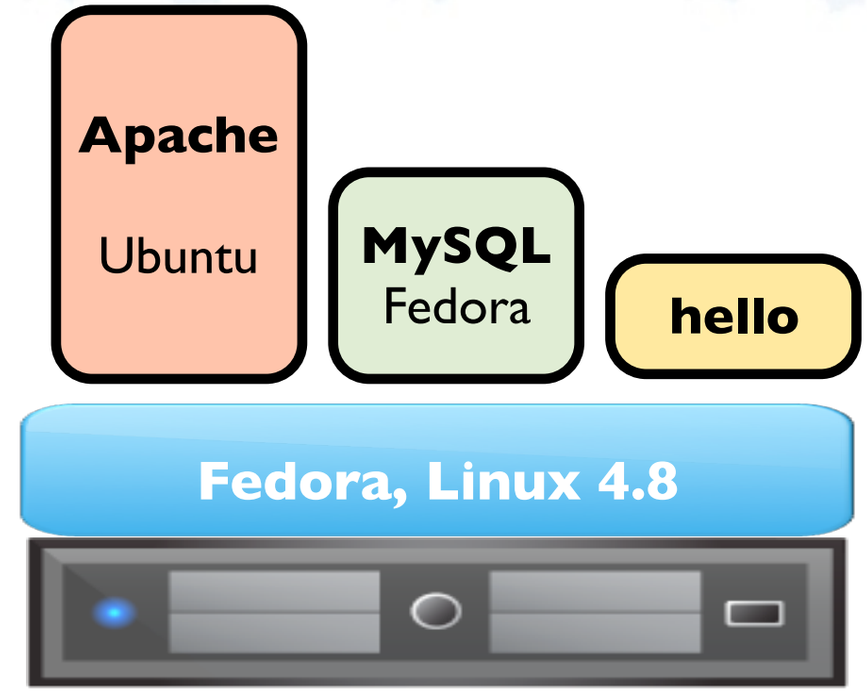
Topic 3 Servers and Virtualization

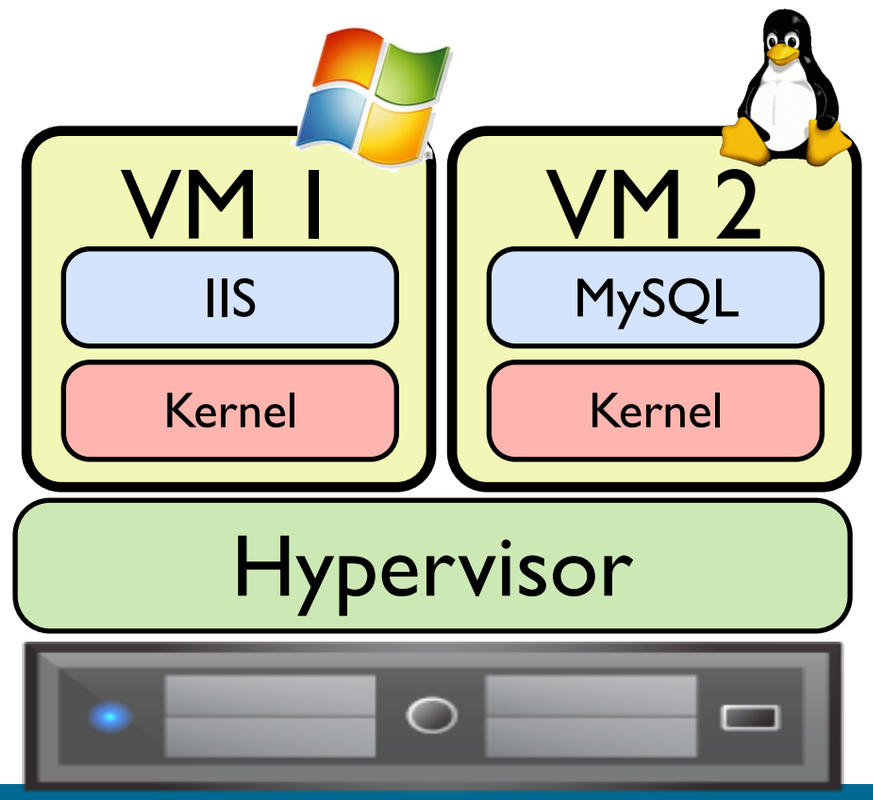
Different types of virtualization:

- Containers, vms, full vs hosted VMs
- Different security/performance/ resource requirements

Page tables provide isolation and nicer abstractions for accessing memory

- Hide physical resources from the virtual layer
-





Topic 4 Networks (SDN, NFV)

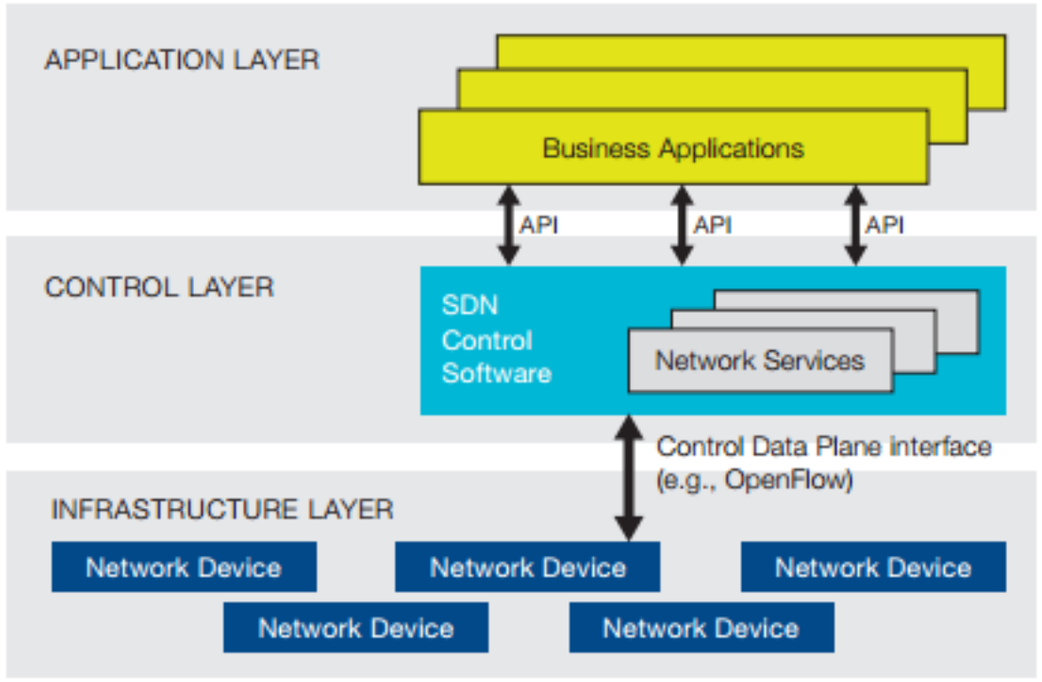
NFV reduces overheads by avoiding kernel

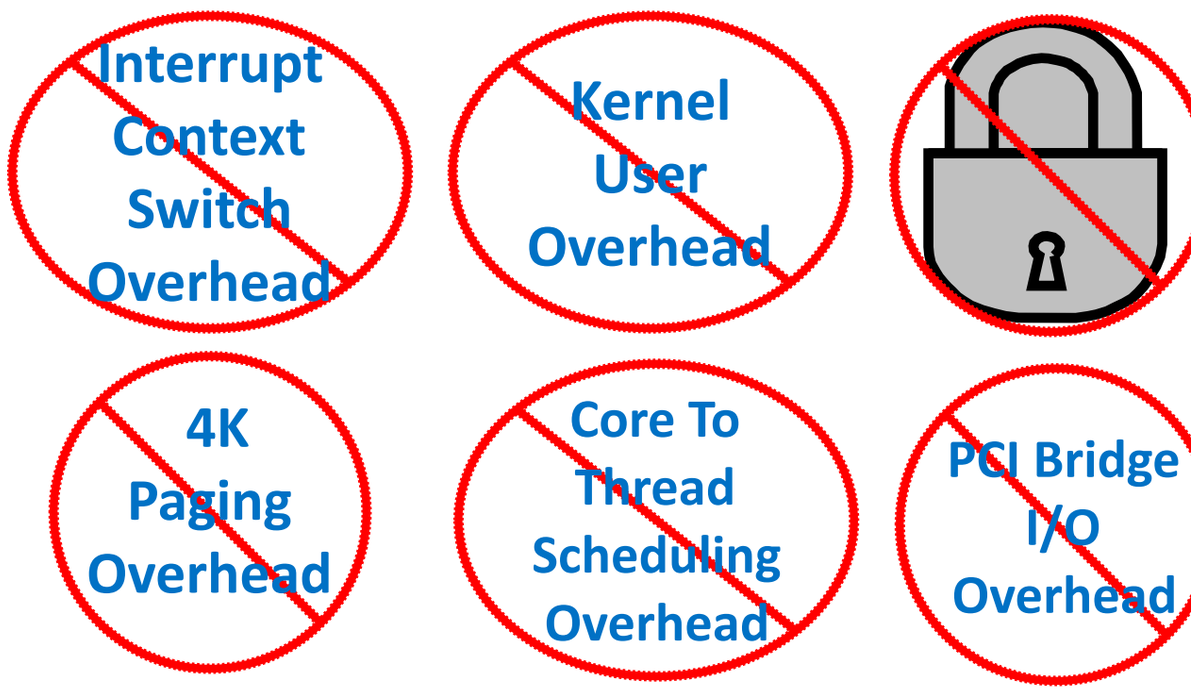
- Use virtualization to share HW
- May be slower than HW, but easier to deploy and manage

SDN allows dynamic control of network flows

- Lets multiple users virtualize (share) the network

Separate the control and data plane





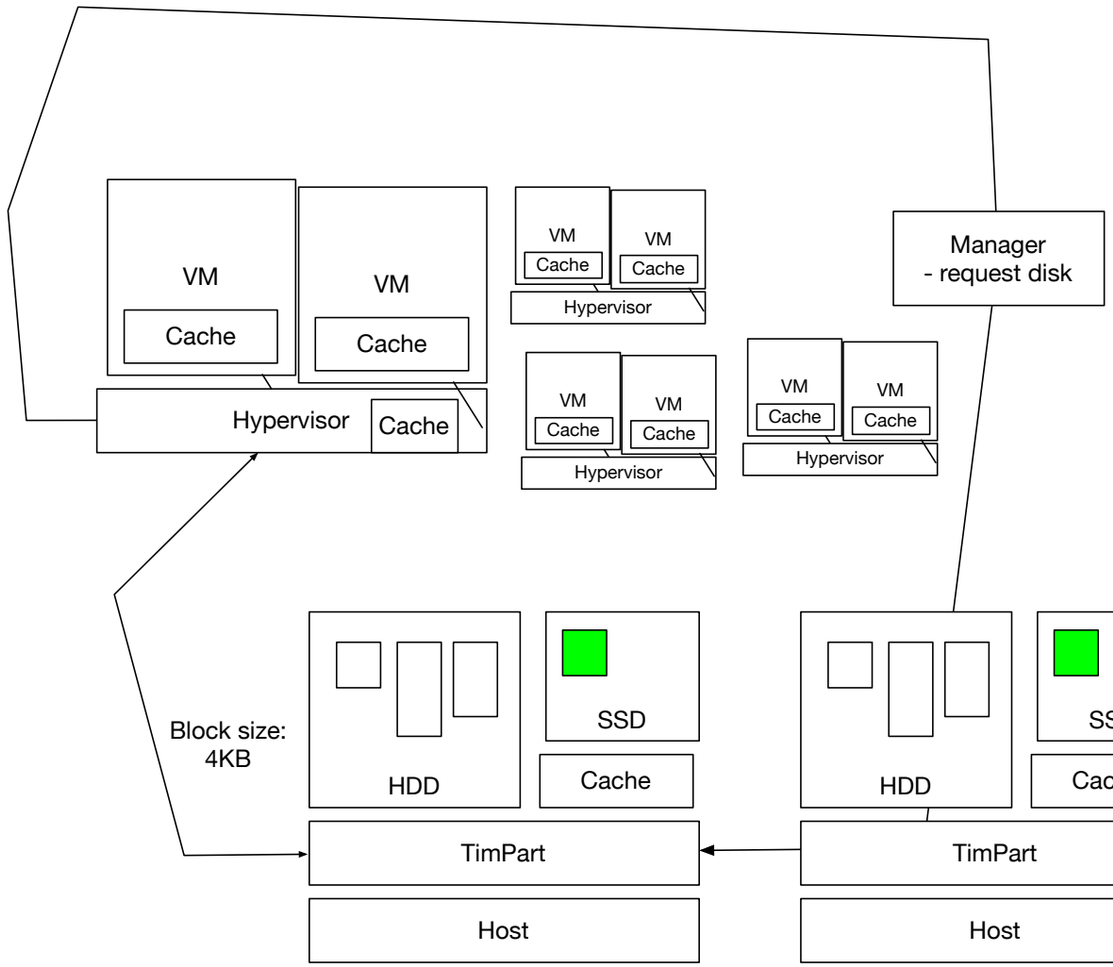
Topic 5 Storage and Fault Tolerance

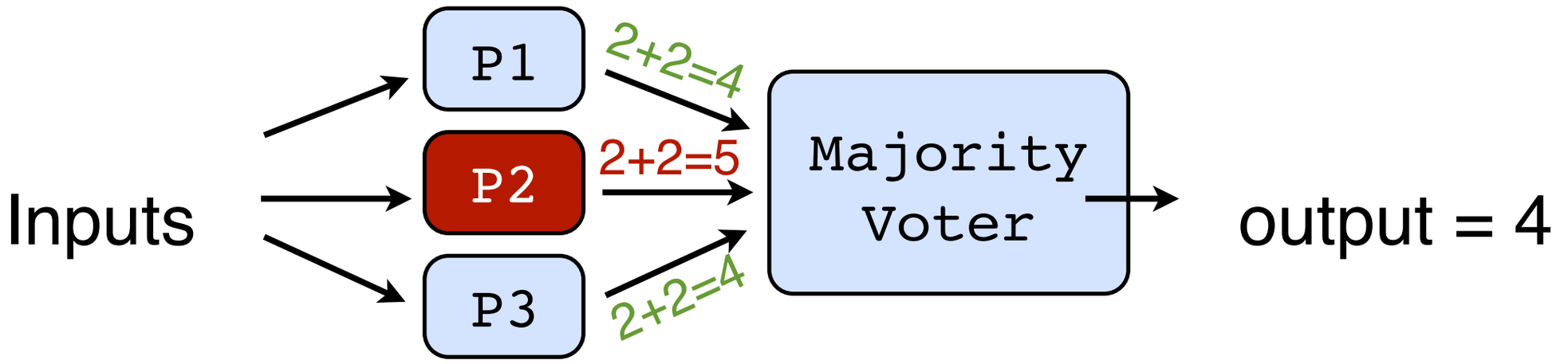
Different types of faults need different levels of replication

Performance numbers are good to know

Block vs object storage

- Different abstractions/interfaces





Topic 6 Ordering, and Consistency

Ordering is easier to track than time

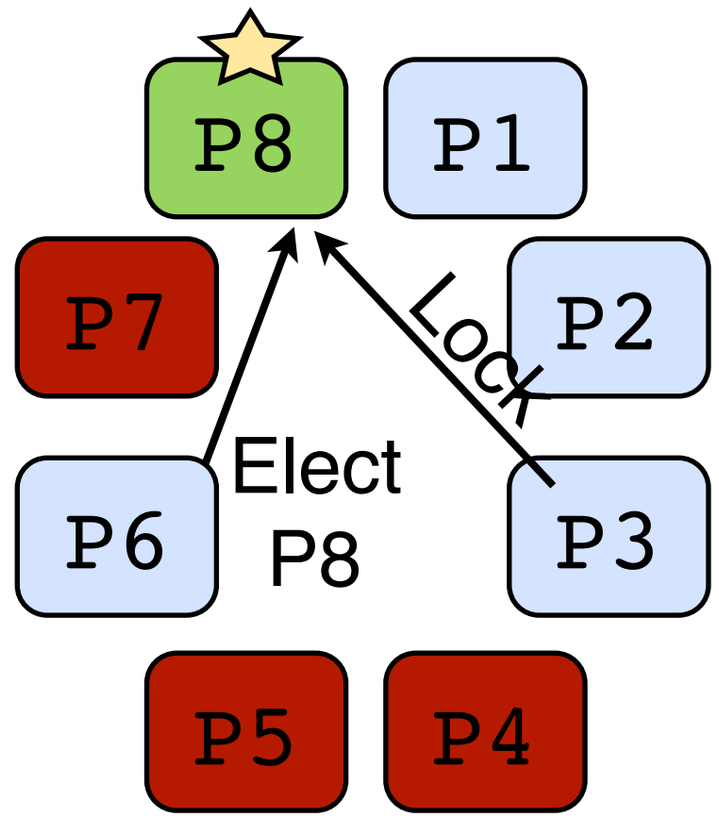
- different clock approaches give you different guarantees

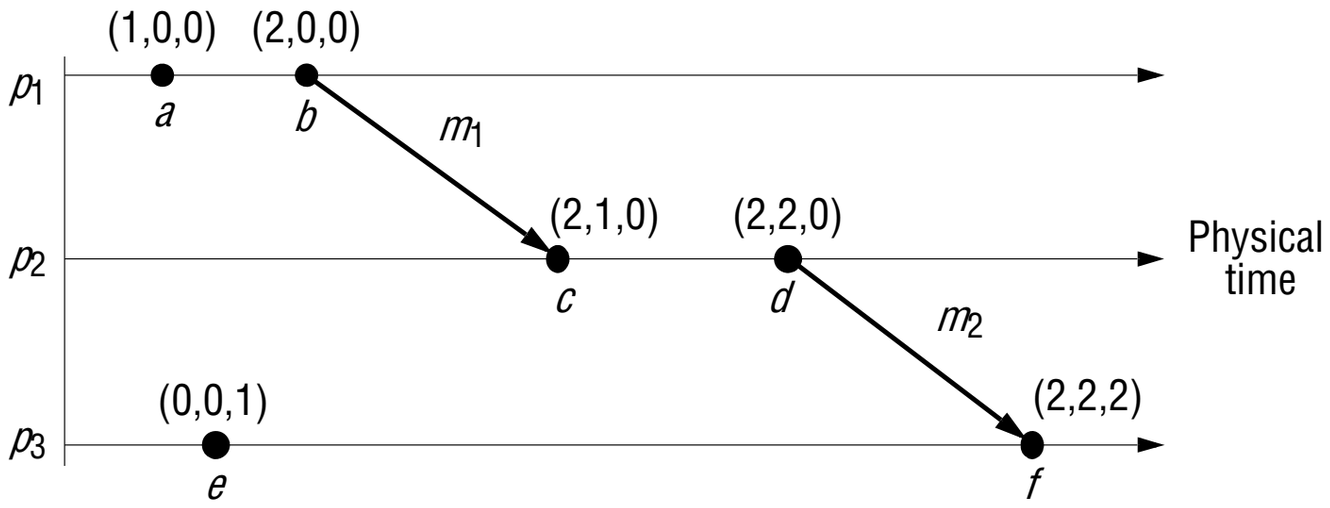
Data Consistency

- Read/write trade-offs
- Quorum based systems

Election algorithms

- Lets us have a centralized leader without it being a single point of failure





Topic 7 Scalable Web Services and Serverless

Multi-tier is a common architecture

Separate stateful vs stateless components

- Different scaling techniques

Microservices divide up into small components

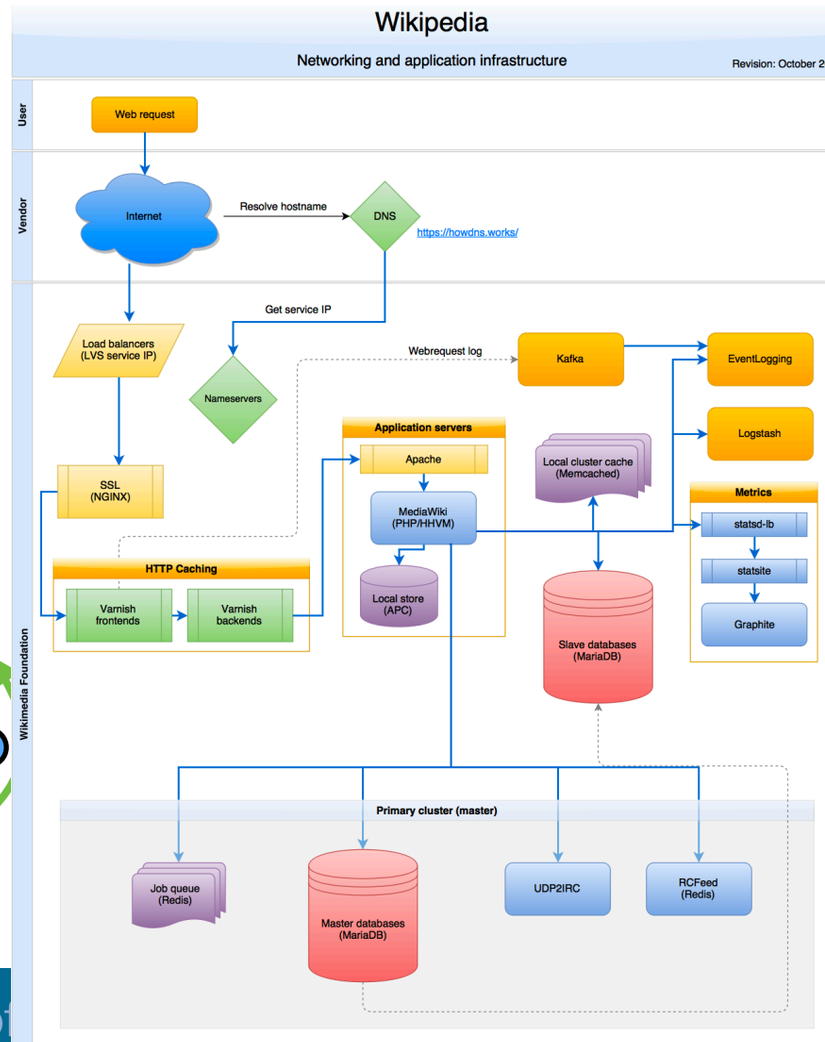
- Easier to scale each piece
- Network connectivity overhead and more complexity

Chaos Monkey

- Break stuff all the time to measure if system is robust

- Serverless computing
  - start containers on demand based on incoming request rate
  - Some requests are slow during scaleup

AWS Lambda

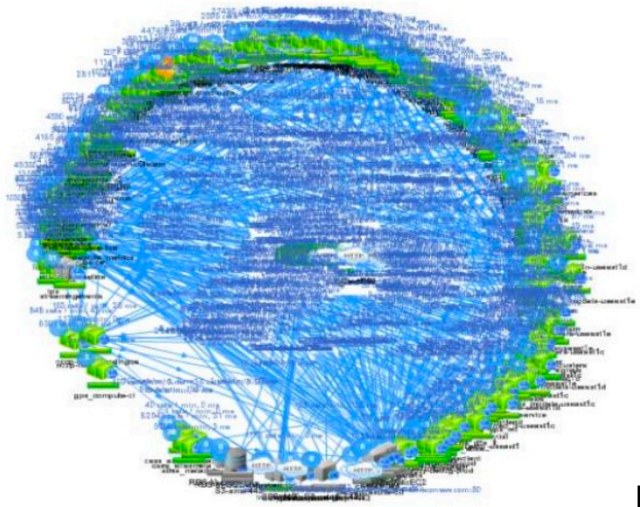


Topic 7 Scalable Web Services and Serverless

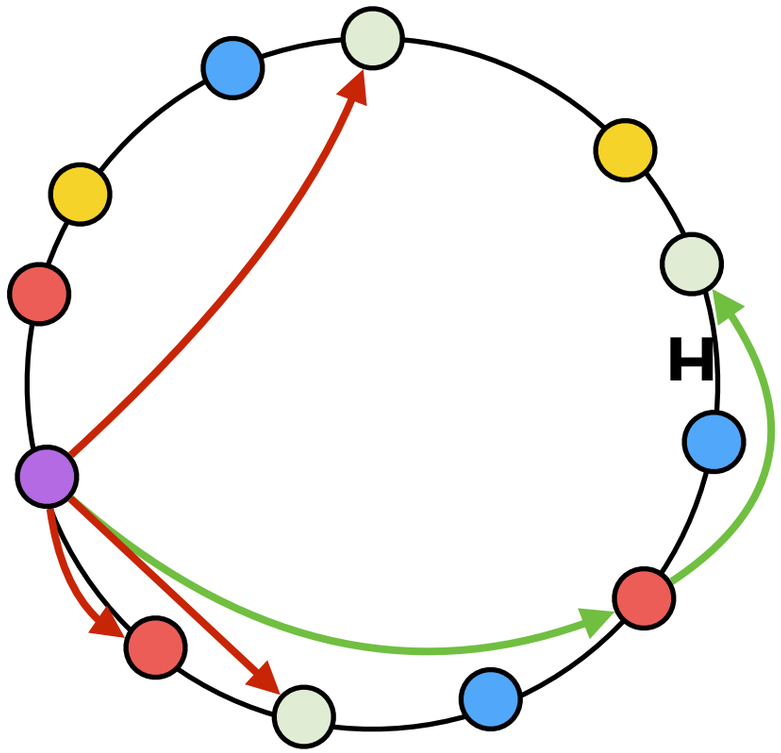
Consistent Hashing

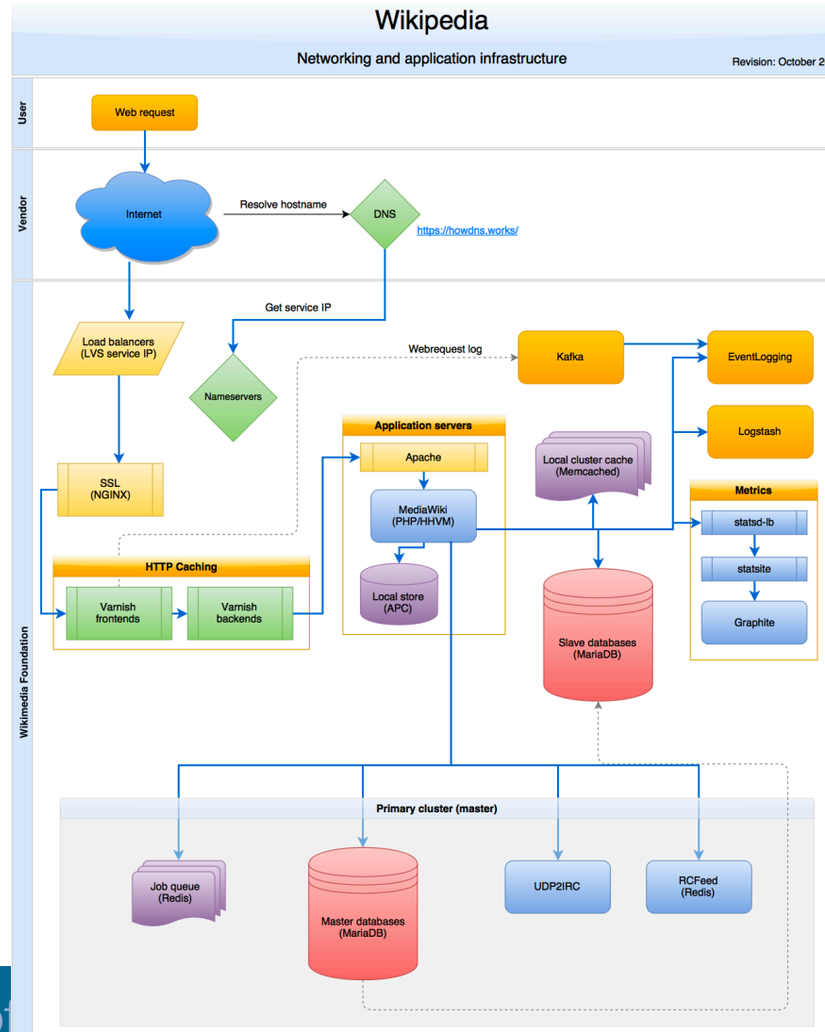
Distributed Hash Table

- Need to be able to scale up services without migrating all the data



AWS Lambda





Challenges

Heterogeneity: VMs/containers

Openness: Network protocols, PaaS

Security: VMs/containers

- Less transparency -> more security

Failure Handling: Replication

Concurrency: Distributed locks

Quality of Service: latency / throughput

Scalability: Performance gain by adding resources

Transparency: IaaS/SaaS/PaaS

Everything else…

(or at least some of it)

High Performance Computing

How is a super computer different from a cloud?

Super Computer

- Entire cluster bought at same time
- High end network, server, and storage HW
- Small number of scientists have access

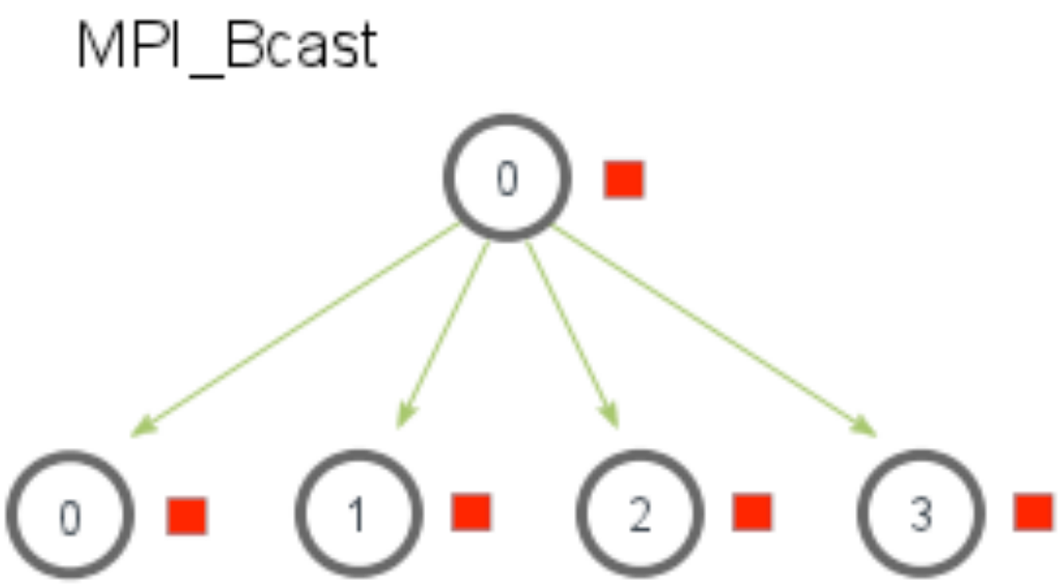
Cloud???

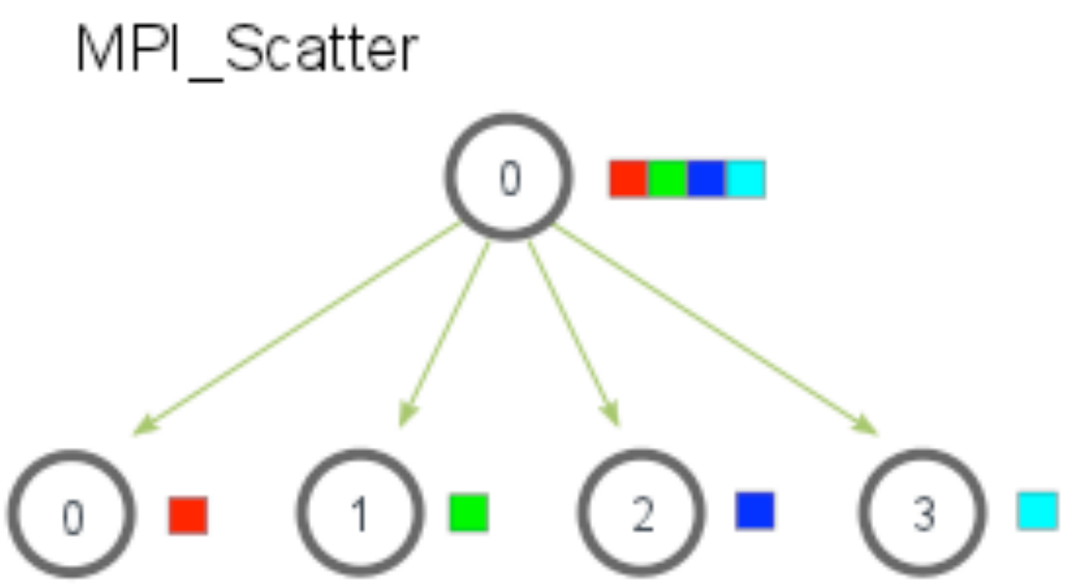
MPI

MPI - Message Passing Interface

- Library standard defined by a committee of vendors, implementers, and parallel programmers
- Used to create parallel programs based on message passing
- Popular for scientific computation being performed on high performance computing clusters (super computers)

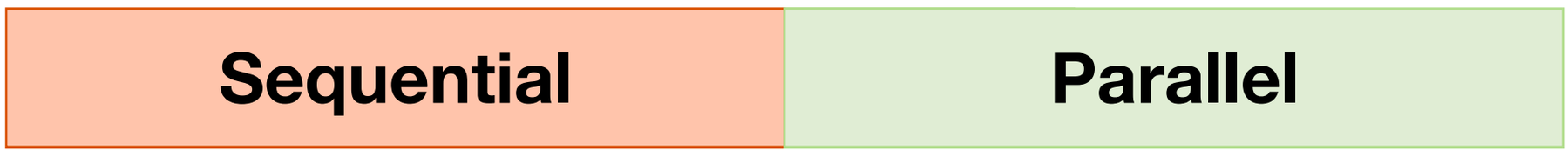
Provides communication primitives for messaging





Amdahl's Law [published 1967]

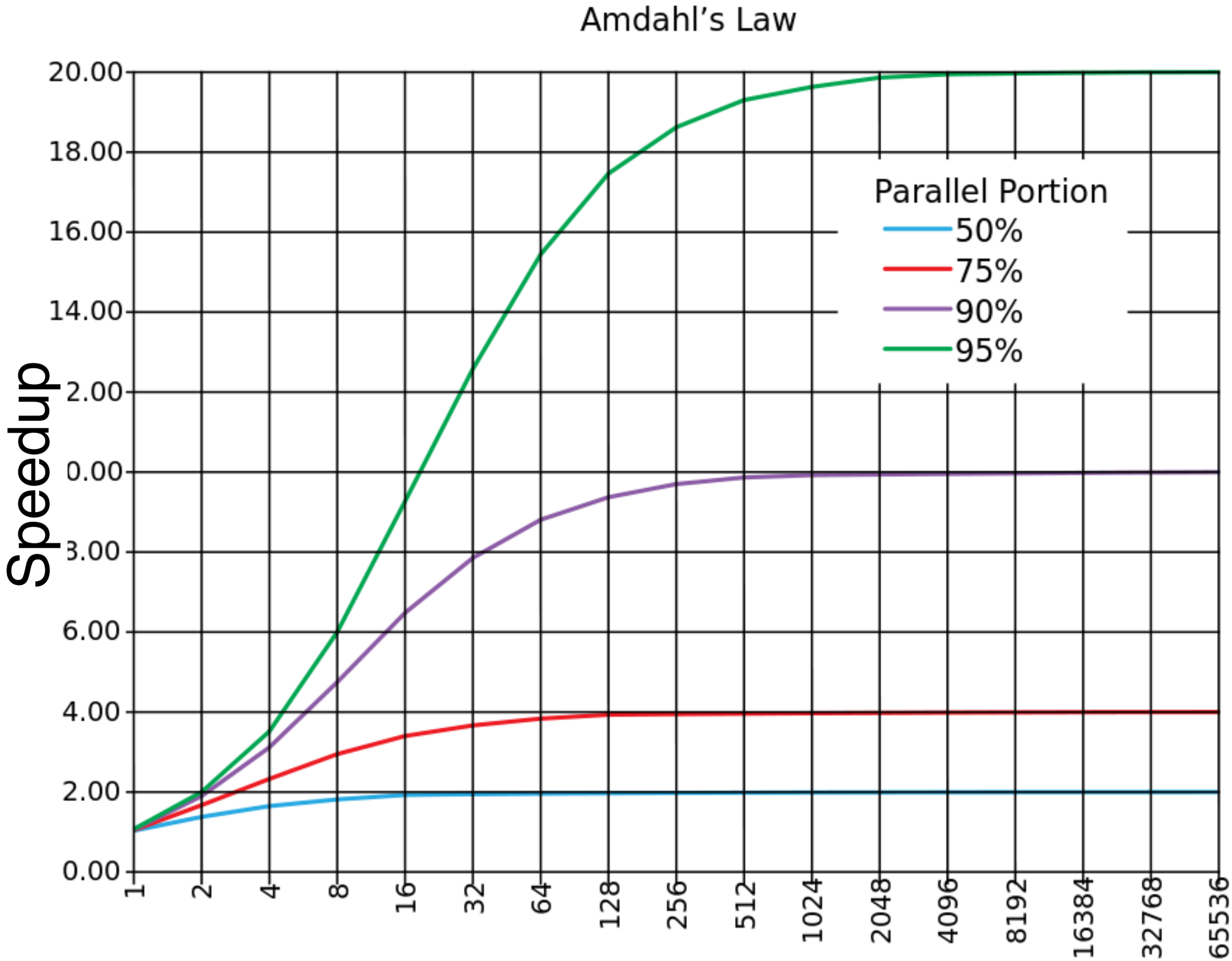
Parts of a program must be run **sequentially** and parts can be run in **parallel**.



Speedup of a parallel application is limited

- **Speedup** $= \dfrac{1}{(1-P) + P/N}$

- P = fraction of program that is parallel
- N = number of processing entities



source: wikipedia

Big Data Analytics

Volume: The amount of data companies want to analyze is growing tremendously

- 40 trillion gigabytes by 2020

Variety: Data is often unstructured and/or user generated

- Tweets, videos, biometrics, much more

Velocity: Analysis must be fast to be useful

- 1TB of new data generated by NY Stock exchange each day

Map Reduce & Hadoop

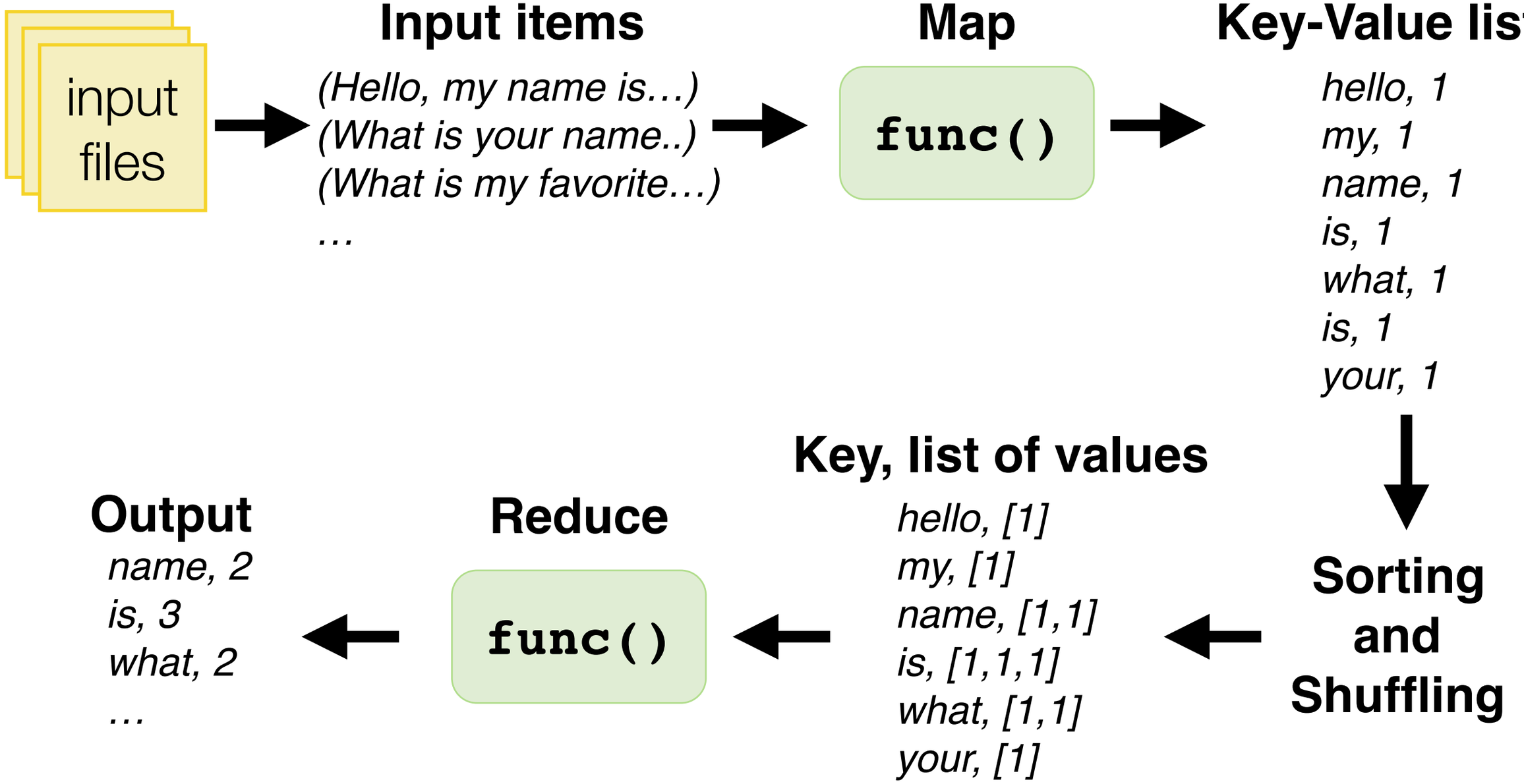
Map Reduce was developed at Google

- Large scale analytics
- Uses commodity servers
- Includes a distributed storage system
- Schedules tasks close to where data is located
- Detects and repeat failed or slow tasks
- New programming model: Map & Reduce

Hadoop is an open source version of Map Reduce

- Ideas are basically interchangeable

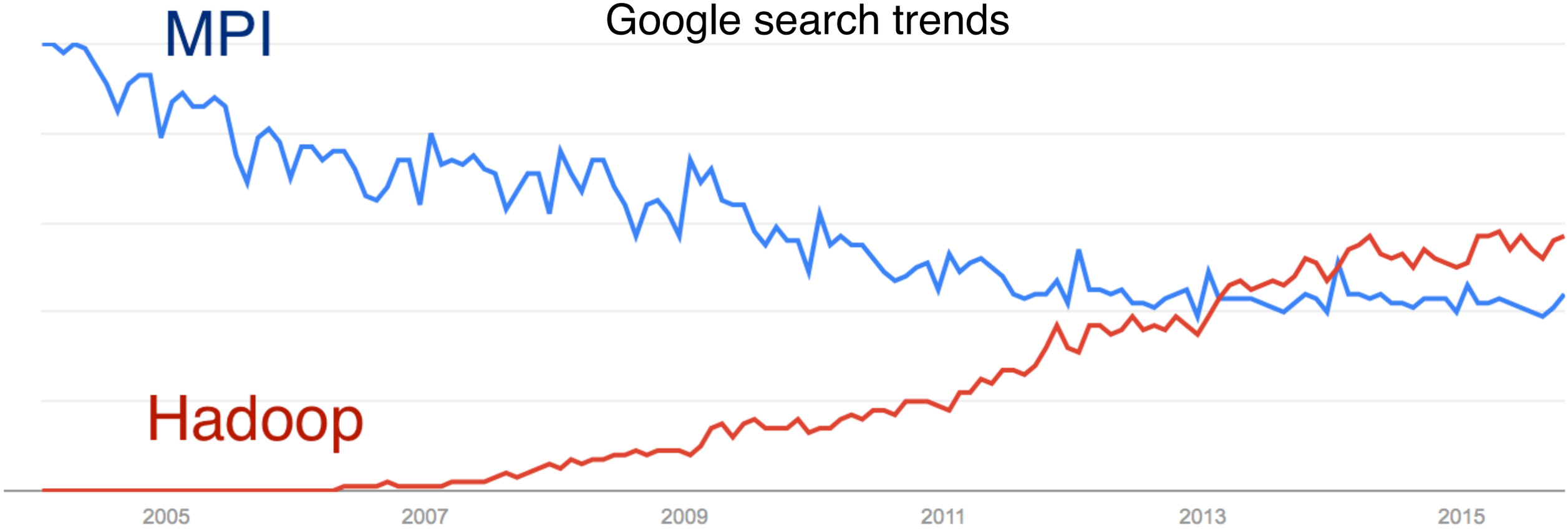
Map Reduce Flow



MPI vs Hadoop

Hadoop is growing

- Used by a much wider range of businesses
- Generally used to solve different problems than MPI would be used for



Storm

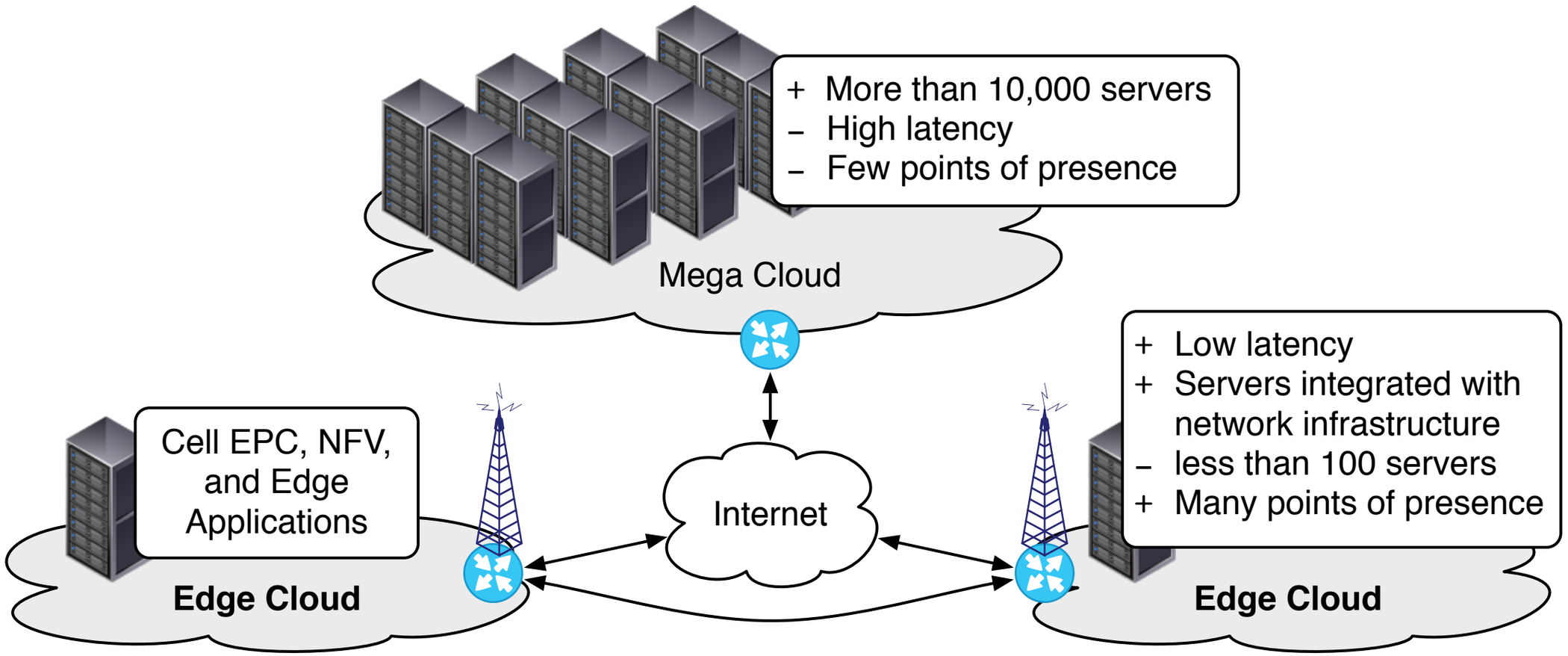
Hadoop is for **batch** processing

Sometimes you want **stream** processing

Storm is basically Hadoop for streams

- Define a graph of processing nodes
- Stream data through the graph
- Manage the workers (each executing a part of the graph)
- Detect failure, carefully buffer data in queues, etc

Edge Computing



Smart Devices
Latency sensitive, mobile, location-aware

IoT, Smart Communities
Bandwidth intensive, M2M communication, stream-oriented

Smart Vehicles
Latency sensitive, mobile, app-specific network QoS

Where is the edge?

Not sure… it doesn't exist yet

Maybe every cell phone tower or "central office" will have a rack of servers

- Who will pay to put them there?
- What will be the killer app?

"Chicken and Egg" problem

- Edge applications are only useful if there is a country-wide edge cloud…
- Nobody will pay to build an edge cloud infrastructure unless there are great edge applications that will pay to use it…

CDNs

Content Delivery Networks are similar to Edge Clouds

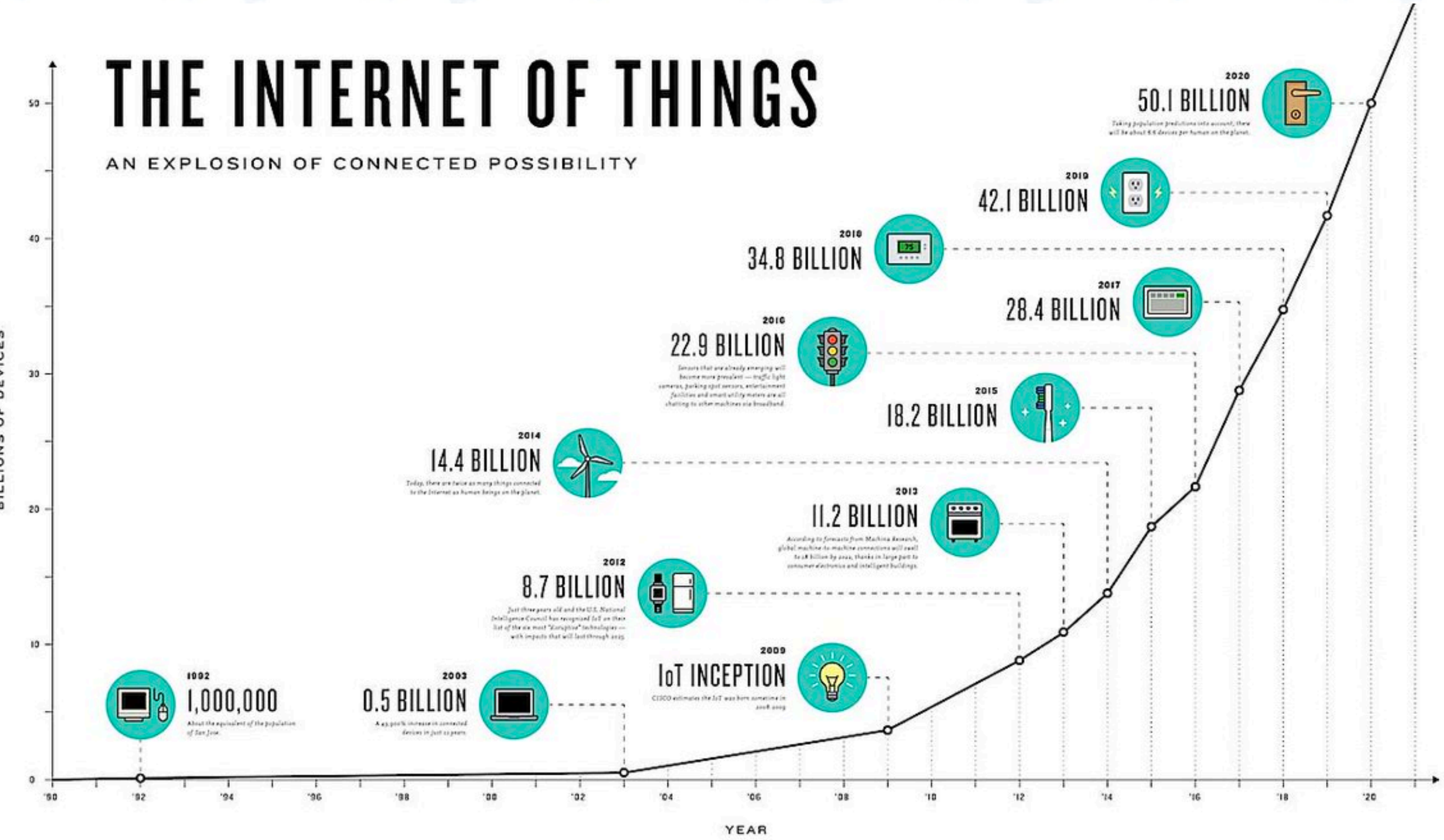
- Provide content closer to the users
- Uses less core bandwidth, lower latency for users

Akamai is the major CDN company in the US

- Originally just hosted static content (videos, images, etc)
- Now supports more dynamic content

My prediction: *Akamai will become an edge cloud provider*

IoT / CPS


THE INTERNET OF THINGS
AN EXPLOSION OF CONNECTED POSSIBILITY
BILLIONS OF DEVICES
50
40
30
20
10
0
'90
'92
'94
'96
'98
'00
'02
'04
'06
'08
'10
'12
'14
'16
'18
'20
YEAR
1992
1,000,000
About the equivalent of the population of San Jose.
2003
0.5 BILLION
A 49,900% increase in connected devices in just 12 years.
2009
IoT INCEPTION
CISCO estimates the IoT was born sometime in 2008-2009.
2010
22.9 BILLION
Sensors that are already emerging will become more prevalent — traffic light cameras, parking spot sensors, entertainment facilities and smart utility meters are all chatting to other machines via broadband.
2012
8.7 BILLION
Just three years old and the U.S. National Intelligence Council has recognized IoT on their list of the six most "disruptive" technologies — with impacts that will last through 2025.
2013
11.2 BILLION
According to forecasts from Machina Research, global machine-to-machine connections will swell to 28 billion by 2020, thanks in large part to consumer electronics and intelligent buildings.
2014
14.4 BILLION
Today, there are twice as many things connected to the Internet as human beings on the planet.
2015
18.2 BILLION
2017
28.4 BILLION
2018
34.8 BILLION
2019
42.1 BILLION
2020
50.1 BILLION
Taking population predictions into account, there will be about 6.6 devices per human on the planet.

What processing does IoT need?

Really streaming big data analytics for IoT sensor information

Low latency edge clouds for IoT control

What's next?

You now know the basic challenges faced by many large scale distributed systems

Find an area that interests you and try to learn more about it!

These skills will be very valuable for getting a job:

- Amazon's cloud services
- Hadoop, big data analysis
- Network programming
- Multi-threading, concurrency
- Virtual machines, containers

There are many free resources to help you learn and try things out. Take advantage of them!