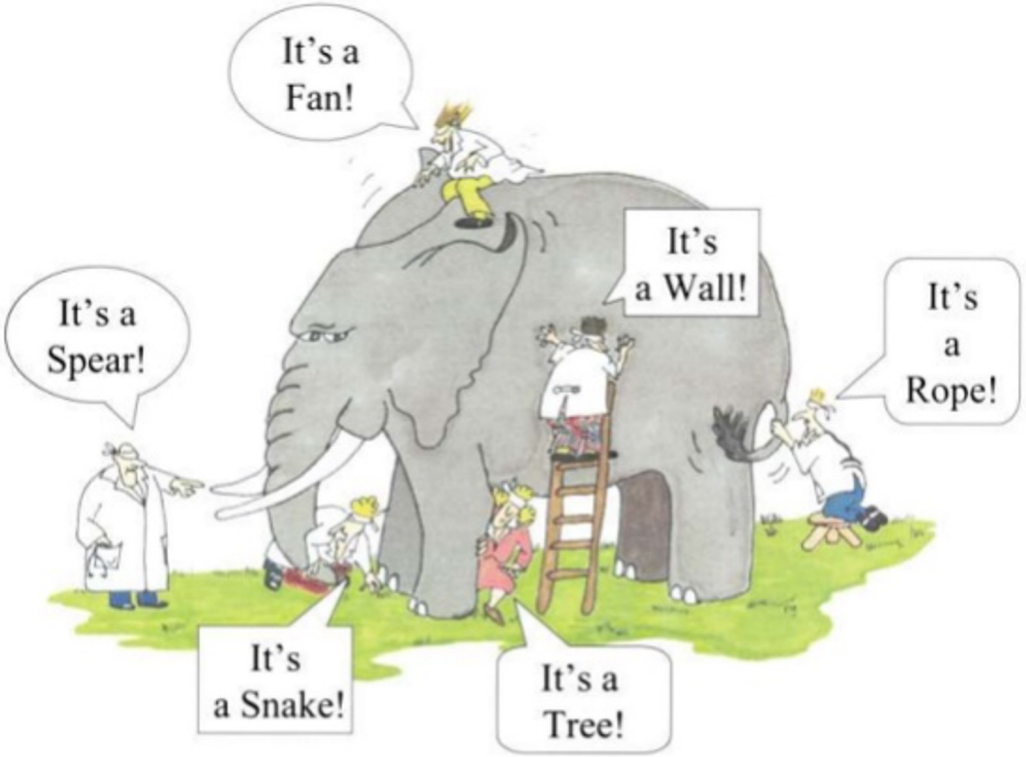
Data Science Spring 2023

Big Data



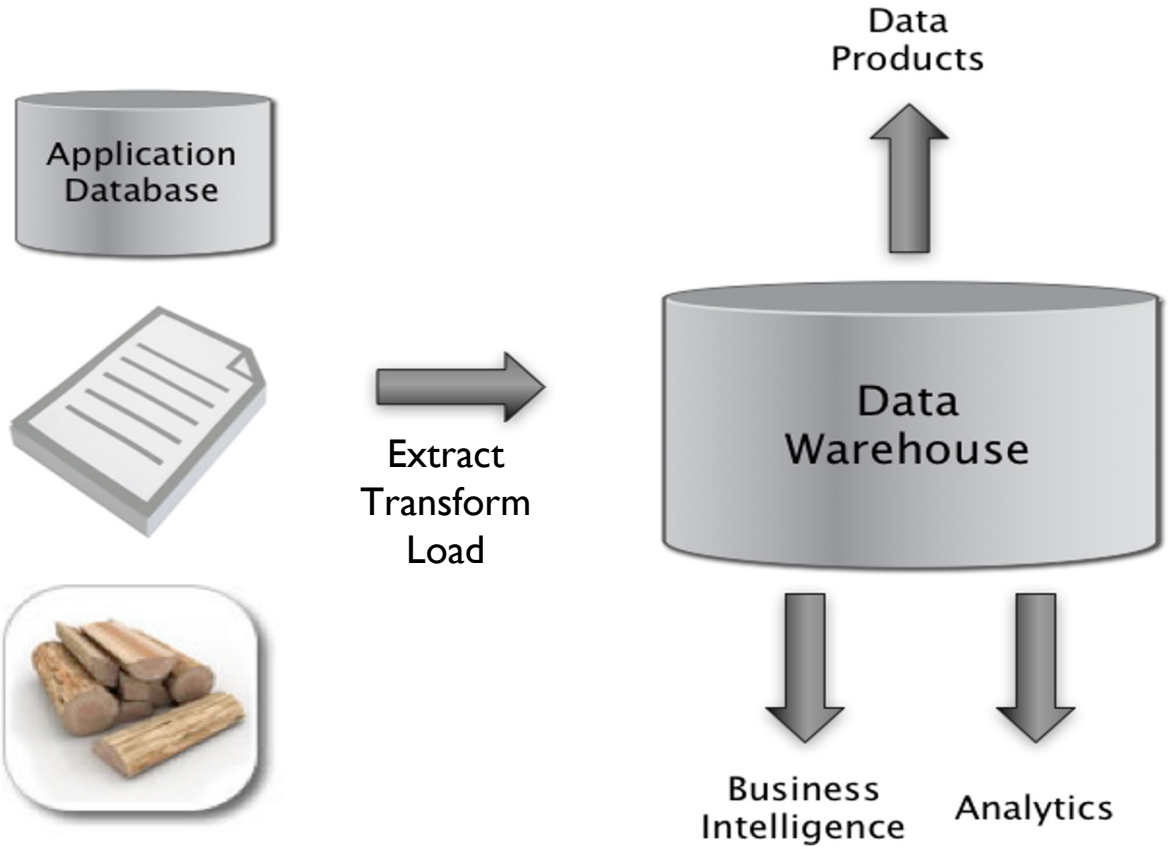
Dept. of Computer Science and Engineering
Korea University

Big Data

- Today
  - Big Data: definitions, examples, analytics
  - Data Warehouses and Data Lakes
  - Introduction to distributed data storage and processing
  - Apache Spark and Modin

What is Big Data?

The Big Picture: Actionable Analytics



Data Lake

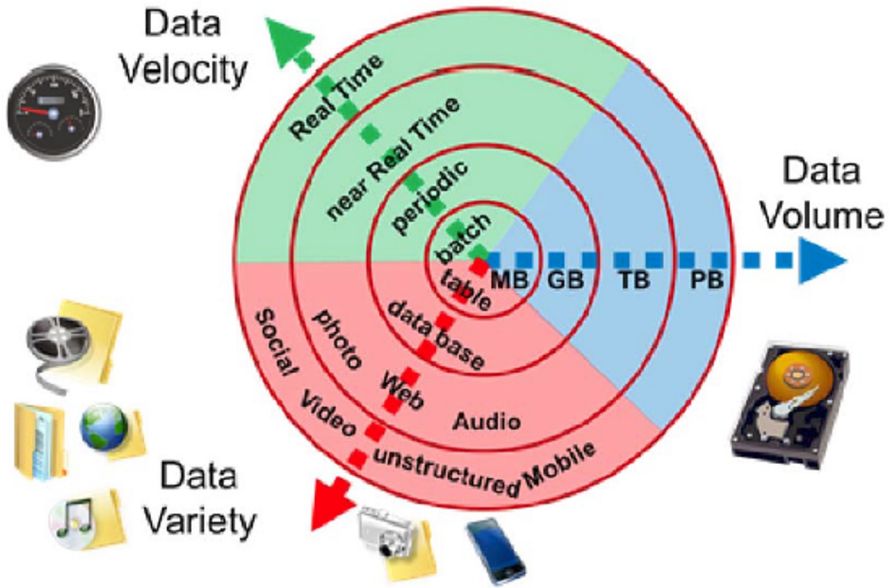
- 80% of worldwide data will be unstructured by 2025[1]
- Unstructured data creates a unique challenges:
  - can't easily be stored in a database
  - its attributes make it a challenge to search for, edit, analyze, especially on the fly
- Large companies already reached that critical mass already

[1]https://solutionsreview.com/data-management/80-percent-of-your-data-will-be-unstructured-in-five-years/

What is Big Data?

- Data of sizes beyond ability of traditional SW tools to quickly capture, curate, manage, and process
- Unstructured, semi-structured and structured data
- Data requiring parallel computing tools to process

Big Data



By Ender005 - Own work, CC BY-SA 4.0,
https://commons.wikimedia.org/w/index.php?curid=49888192

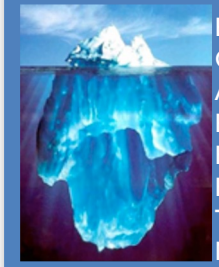
Using Big Data: Actionable Analytics

- Descriptive – What is happening now?
- Predictive – What will happen?
- Prescriptive – What should I do about it?

Sources of Big Data

Big Data is Everywhere!

It's All Happening On-line



Every:
Click
Ad impression
Billing event
Fast Forward, pause,…
Friend Request
Transaction
Network message
Fault
…

User Generated (Web & Mobile)




…


Open Data Sources

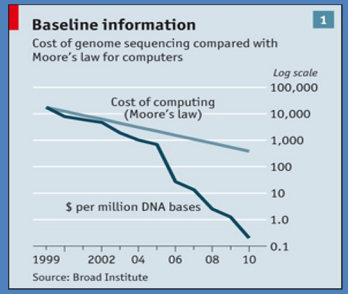
DATA.GOV

DataSF

HealthData.gov

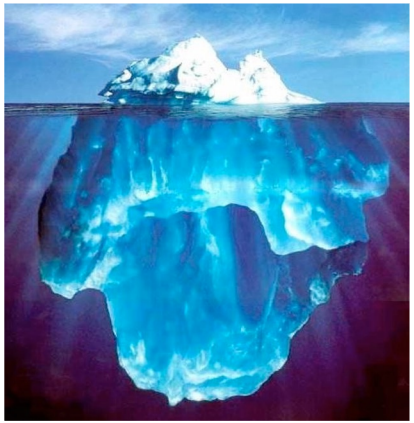
Internet of Things / M2M

Scientific Computing



Where Does Big Data Come From?

- It is all happening online – could record every:
  - Click
  - Ad impression
  - Billing event
  - Fast Forward, pause,…
  - Server request
  - Transaction
  - Network message
  - Fault
  - …



Facebook's daily logs: **>60 TB**

Where Does Big Data Come From?

The number of IoT devices could rise to 41.6 billion by 2025.[3]

YouTube has over 500 hours of new content uploaded every minute.[2]

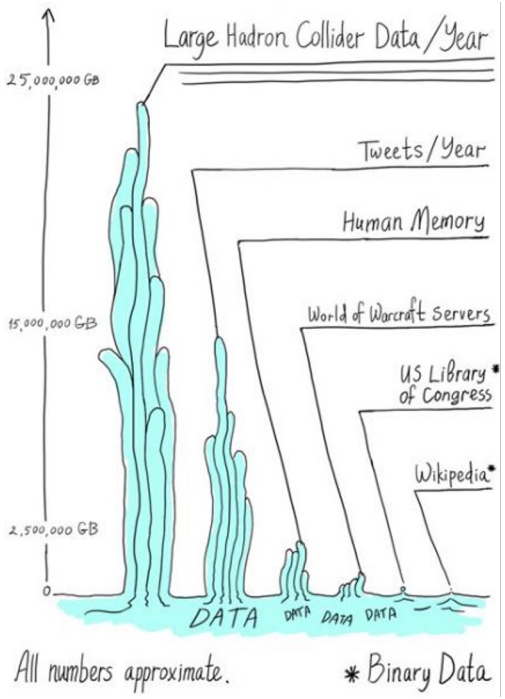
Image credit: IBM, YouTube
https://www.tubefilter.com/2019/05/07/number-hours-video-uploaded-to-youtube-per-minute/

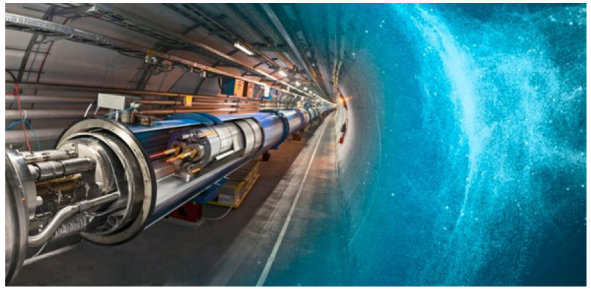
Where Does Big Data Come From?

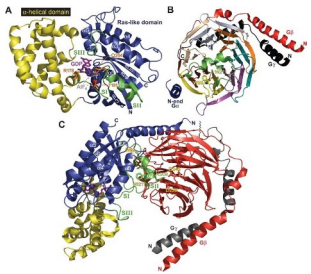
- User Generated Content (Web & Mobile)
  - >> Facebook
  - >> Instagram
  - >> Google
  - >> Yelp
  - >> TripAdvisor
  - >> Twitter
  - >> YouTube
  - >> …

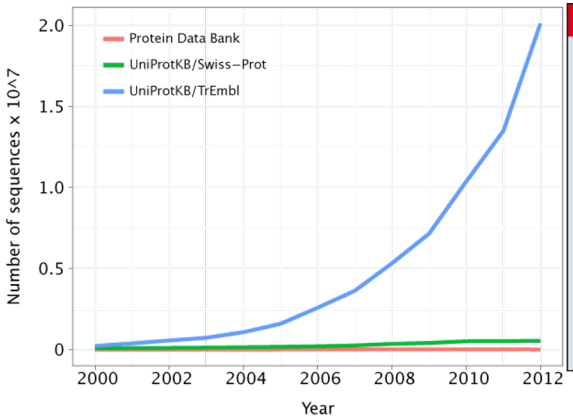
Google web index: **10+ PB**

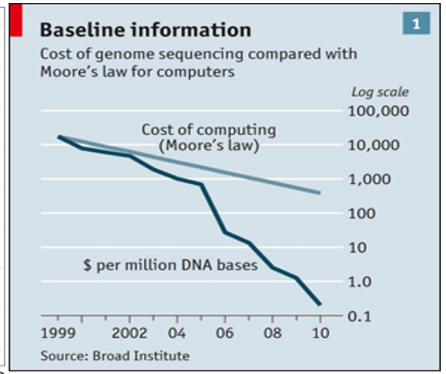
Health and Scientific Computing











Images: http://www.economist.com/node/16349358
http://gorbi.irb.hr/en/method/growth-of-sequence-databases/
http://www.symmetrymagazine.org/article/august-2012/particle-physics-tames-big-data

Graph Data

Lots of interesting data has a graph structure:

- Social networks
- Telecom Networks
- Computer Networks
- Road networks
- Collaborations
- Relationships
- …

Some of these graphs can get quite large
(e.g., Facebook user graph)

Log Files – Apache Web Server Log

```
uplherc.upl.com - - [01/Aug/1995:00:00:07 -0400] "GET / HTTP/1.0" 304 0
uplherc.upl.com - - [01/Aug/1995:00:00:08 -0400] "GET /images/ksclogo-medium.gif HTTP/1.0" 304 0
uplherc.upl.com - - [01/Aug/1995:00:00:08 -0400] "GET /images/MOSAIC-logosmall.gif HTTP/1.0" 304 0
uplherc.upl.com - - [01/Aug/1995:00:00:08 -0400] "GET /images/USA-logosmall.gif HTTP/1.0" 304 0
ix-esc-ca2-07.ix.netcom.com - - [01/Aug/1995:00:00:09 -0400] "GET /images/launch-logo.gif HTTP/1.0" 200 1713
uplherc.upl.com - - [01/Aug/1995:00:00:10 -0400] "GET /images/WORLD-logosmall.gif HTTP/1.0" 304 0
slppp6.intermind.net - - [01/Aug/1995:00:00:10 -0400] "GET /history/skylab/skylab.html HTTP/1.0" 200 1687
piweba4y.prodigy.com - - [01/Aug/1995:00:00:10 -0400] "GET /images/launchmedium.gif HTTP/1.0" 200 11853
tampico.usc.edu - - [14/Aug/1995:22:57:13 -0400] "GET /welcome.html HTTP/1.0" 200 790
```

Log Files – Machine Syslog File

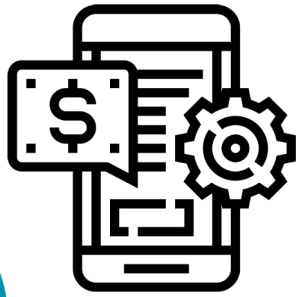
```
dhcp-47-129:CS100_1> syslog -w 10
Feb  3 15:18:11 dhcp-47-129 Evernote[1140]
<Warning>: -[EDAMAccounting read:]: unexpected field
ID 23 with type 8.  Skipping.
Feb  3 15:18:11 dhcp-47-129 Evernote[1140]
<Warning>: -[EDAMUser read:]: unexpected field ID 17
with type 12.  Skipping.
Feb  3 15:18:11 dhcp-47-129 Evernote[1140]
<Warning>: -[EDAMAuthenticationResult read:]:
unexpected field ID 6 with type 11.  Skipping.
Feb  3 15:18:11 dhcp-47-129 Evernote[1140]
<Warning>: -[EDAMAuthenticationResult read:]:
unexpected field ID 7 with type 11.  Skipping.
Feb  3 15:18:11 dhcp-47-129 Evernote[1140]
<Warning>: -[EDAMAccounting read:]: unexpected field
ID 19 with type 8.  Skipping.
Feb  3 15:18:11 dhcp-47-129 Evernote[1140]
<Warning>: -[EDAMAccounting read:]: unexpected field
ID 23 with type 8.  Skipping.
```

Images: Tesla

Open Data Sources

- Lots of open government and other data sources
  - US Data.gov, US HealthData.gov, CIA World Factbook, EU Open Data Portal
  - Public Data Portal (https://www.data.go.kr)
  - Many cities and states have open data portals
  - Huge list of public datasets: https://github.com/awesomedata/awesome-public-datasets

Big Data Usage is Ubiquitous!



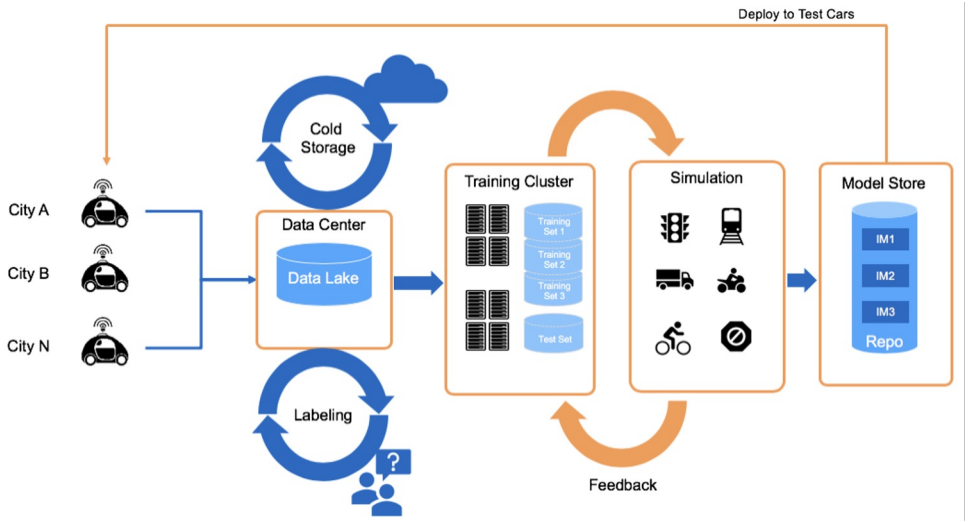
Big Data: Aerospace

- Massive Internet of Things data collection
  - GE jet engines: 5,000 data points/second
  - Boeing 787 generates ~500GB per flight
  - Airbus A380 has ~25,000 sensors

Big Data: Aerospace

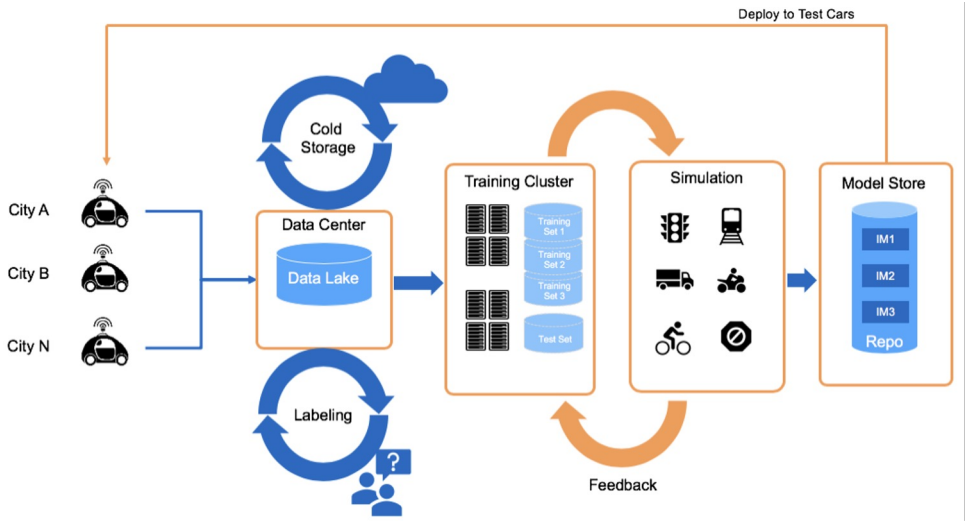
- Optimize fleet maintenance operations
  - >> Plan maintenance, position spare parts, anticipate component failure
- Improve operational performance across flights/fleet
  - >> Data-driven pilot training for fuel savings and component fatigue reduction

Big Data: Autonomous Vehicles



- 1.4-19TB/hr/car
- 300TB to 2+PB data/car/year
  >> ~1B images, >3M images labeled by humans

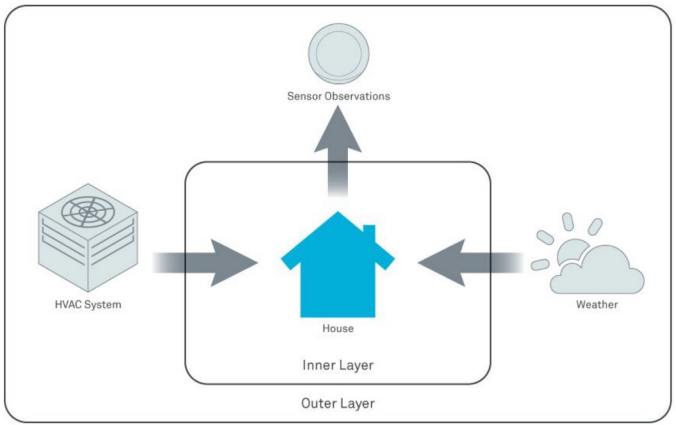
Big Data: Autonomous Vehicles



- Waymo
  - >> Each vehicle: 11-152 TB per day
  - >> Fleet: 2.2 to 30.4PB per day
- DNN Training: 800 GPUs per test car

Big Data: Smart Homes

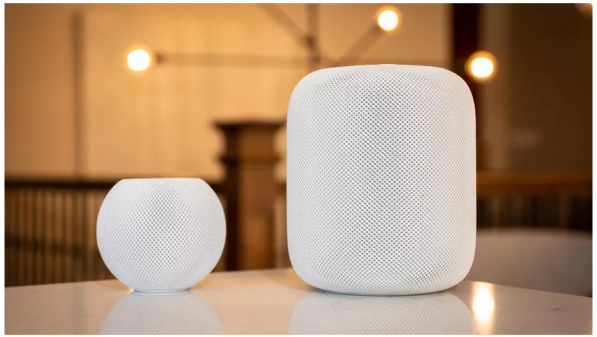
- HVAC[1]: Machine Learning at scale
  - Reduce energy consumption/increase comfort
  - Monitor activity/usage/location patterns



[1] Heating, Ventilation, and Air Conditioning (HVAC)

Big Data: Smart Homes

- NEST devices
  - Build thermal models across many homes/types
  - AI anomaly detection: Urgent Alerts and Early Warnings



Big Data: e-Commerce

- Predictive Analytics:
  - Analyze millions to billions of transactions

- Alibaba's 3 year smart garment factory: $328B market in China
  - 1 in 4 garments are shipped via Alibaba platforms
  - Overall 75% reduction in production lead times
  - AI fabrics for dye simulation
  - AI-powered cutting machines and networked sewing machines

https://www.bloomberg.com/news/articles/2020-11-01/alibaba-s-secret-three-year-experiment-to-reinvent-the-factory

Big Data: e-Commerce

- Predictive Analytics:
  - Analyze millions to billions of transactions
- Target retailer:
  - Market to women in 2nd trimester (peak buying time)
  - Guest ID linked to Target and third-party data
  - Buying larger quantities of unscented lotion at 2nd trimester start

Big Data: Precision Medicine

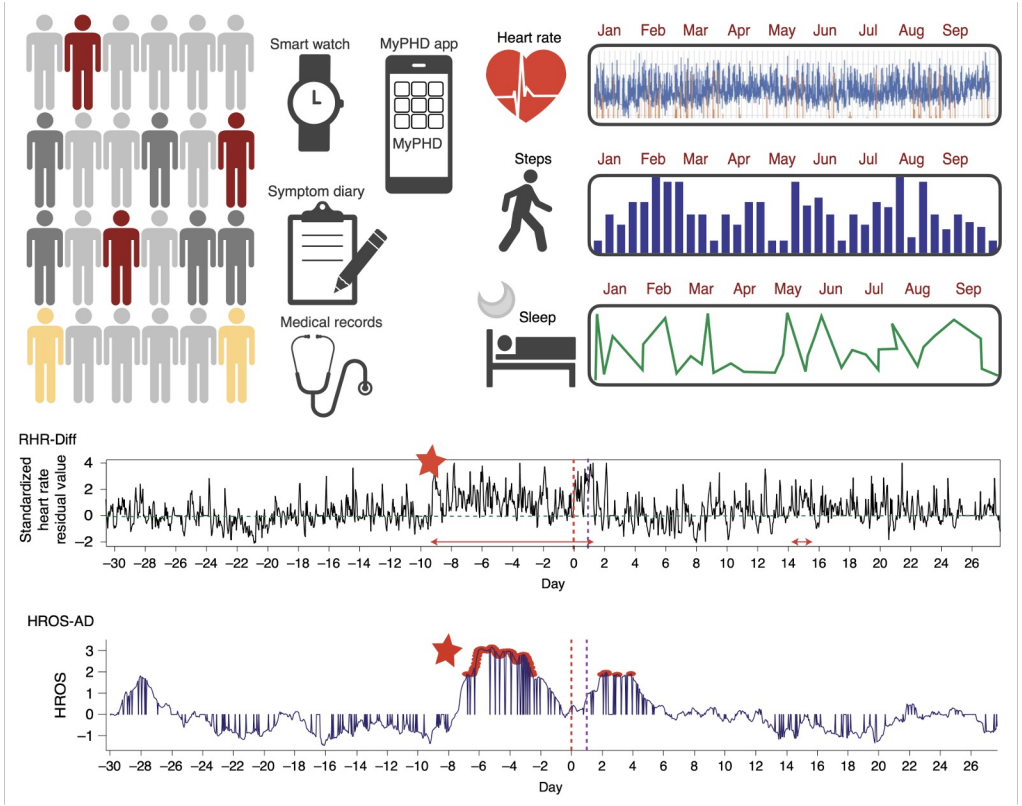
- NYU COVID-19 severity score model:
  - Data from 160 hospitalized patients in Wuhan, China
  - ML identified 4 significantly elevated biomarkers in patients who died versus those who recovered
  - Built model using biomarkers, as well as age and sex
  - Validated model in New York City

- Goal:
  - Give providers data to make informed care decisions, leading to better outcomes for patients

Why Big Tech Wants to Access to Your Medical Records

https://www.wsj.com/articles/google-strikes-deal-with-hospital-chain-to-develop-healthcare-algorithms-11622030401

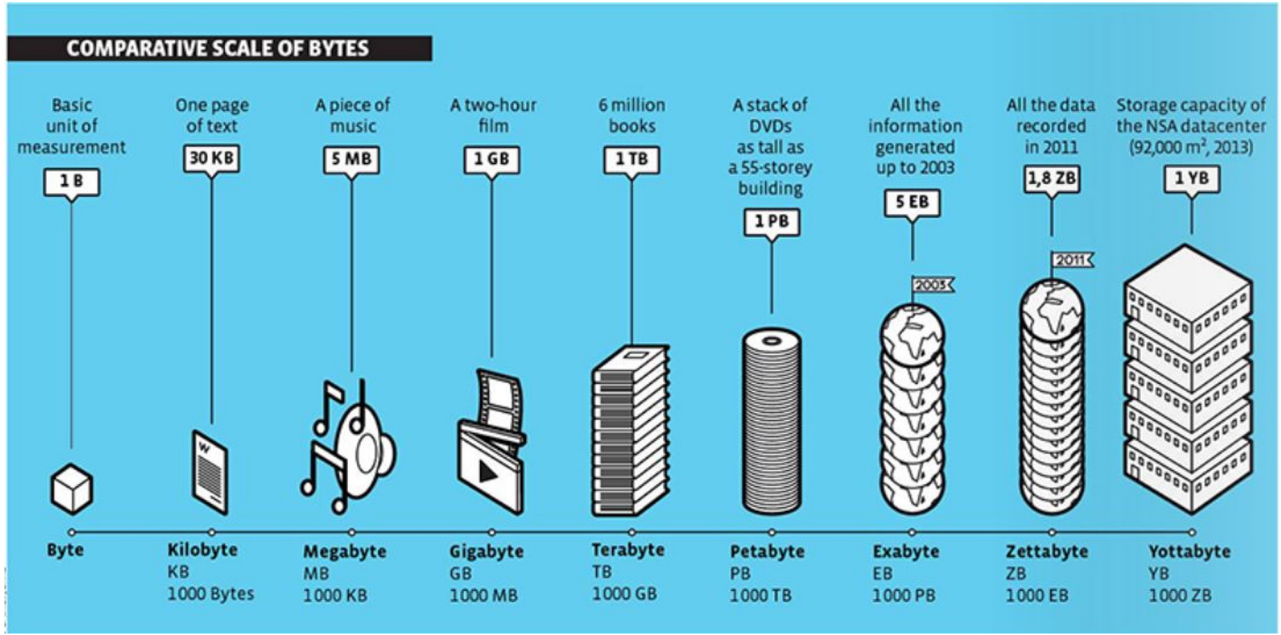
Pre-symptomatic detection of COVID-19 from smartwatch



Mishra et al., “Pre-symptomatic detection of COVID-19 from smartwatch data,” Nature.

Big Data: Genomics

- Human Genome Sequencing: 40 EB/year by 2025



Big Data: Genomics

- Human Genome Sequencing: 40 EB/year by 2025
  - Research:
    - better understanding of diseases
      - Causes of genetic diseases, severity, progression rate
    - improving agriculture
      - >50% more food to feed 9B people by 2050
      - Climate change could reduce crop yields by 25%
  - Clinical:
    - early/better treatment of diseases
      - Identifying hazardous mutations enables early treatment
    - rapidly identifying infection pathogens

Data in the Organization

Inventory

How we like to think of data in the organization

The reality…

Sales (Asia)

Sales (US)

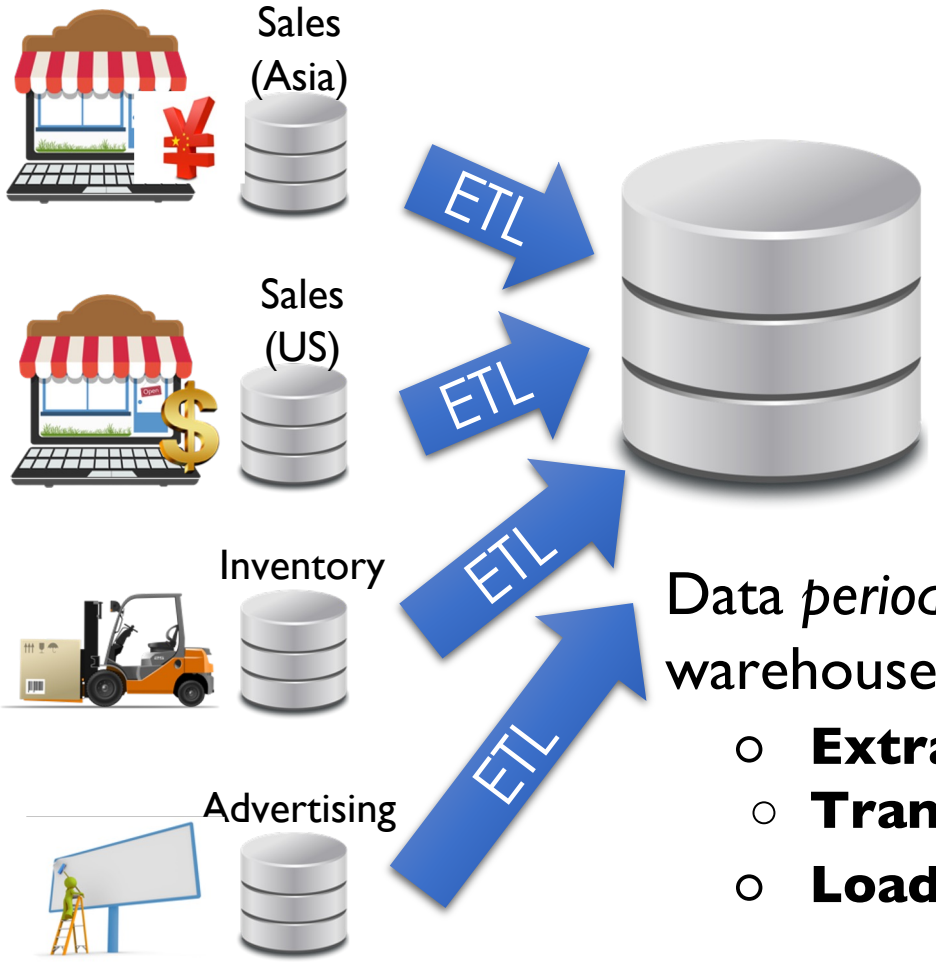
Inventory

Advertising

Operational Data Stores

- Capture **the now**
- Many different databases across an organization
- Mission critical… be careful!
  - Serving live ongoing business operations
  - Managing inventory and powering e-commerce & PoS
- Different formats (e.g., currency)
  - Different schemas (acquisitions ...)
- Live systems often don't maintain history

We would like a consolidated, clean, historical snapshot of the data



Data Warehouse

Collects and organizes historical data from multiple sources

Data *periodically* **ETL**'d into data warehouse:

- **Extracted** from remote sources
- **Transformed** to standard schemas
- **Loaded** into (typically) relational (SQL) DB

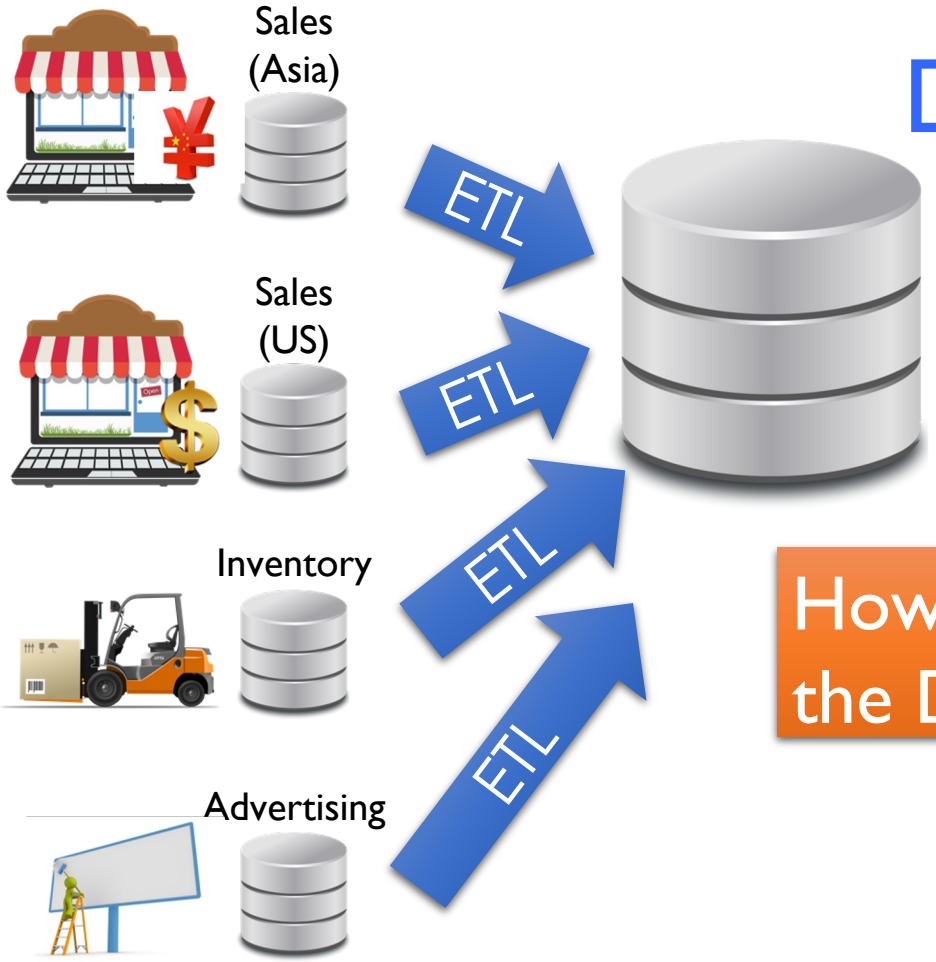
Extract ⇒ Transform ⇒ Load (ETL)

Extract & Load: provides a snapshot of operational data

- Historical snapshot
- Data in a single system
- Isolates analytics queries (e.g., Deep Learning) from business critical services (e.g., processing user purchases)
- Easy!

Transform: clean and prepare data for analytics in a unified representation

- **Difficult** ⇒ often requires specialized code and tools
- Different schemas, encodings, granularities



Data Warehouse

Collects and organizes historical data from multiple sources

How is data organized in the Data Warehouse?

Example Sales Data

pname	category	price	qty	date	day	city	state	country
Corn	Food	25	25	3/30/16	Wed.	Omaha	NE	USA
Corn	Food	25	8	3/31/16	Thu.	Omaha	NE	USA
Corn	Food	25	15	4/1/16	Fri.	Omaha	NE	USA
Galaxy	Phones	18	30	1/30/16	Wed.	Omaha	NE	USA
Galaxy	Phones	18	20	3/31/16	Thu.	Omaha	NE	USA
Galaxy	Phones	18	50	4/1/16	Fri.	Omaha	NE	USA
Galaxy	Phones	18	8	1/30/16	Wed.	Omaha	NE	USA
Peanuts	Food	2	45	3/31/16	Thu.	Seoul		Korea
Galaxy	Phones	18	100	4/1/16	Fri.	Seoul		Korea

Example Sales Data

pname	category	price	qty	date	day	city	state	country
Corn	Food	25	25	3/30/16	Wed.	Omaha	NE	USA
Corn	Food	25	8	3/31/16	Thu.	Omaha	NE	USA
Corn	Food	25	15	4/1/16	Fri.	Omaha	NE	USA
Galaxy	Phones	18	4	1/30/16	Wed.	Omaha	NE	USA
Galaxy	Phones	18	9	3/31/16	Thu.	Omaha	NE	USA
Galaxy	Phones	18	5	4/1/16	Fri.	Omaha	NE	USA
Galaxy	Phones	18	8	1/30/16	Wed.	Omaha	NE	USA
Peanuts	Food	2	45	3/31/16	Thu.	Seoul		Korea

- **Big** table: many *columns* and *rows*
  - Substantial redundancy ⇒ expensive to store and access
  - Make mistakes while updating
- Could we organize the data more efficiently?

Multidimensional Data Model

Sales **Fact Table**

pid	timeid	locid	sales
11	1	1	25
11	2	1	8
11	3	1	15
12	1	1	30
12	2	1	20
12	3	1	50
12	1	1	8
13	2	1	10
13	3	1	10
11	1	2	35
11	2	2	22
11	3	2	10
12	1	2	26
12	2	2	45

Locations

locid	city	state	country
1	Omaha	Nebraska	USA
2	Seoul		Korea
5	Richmond	Virginia	USA

Products

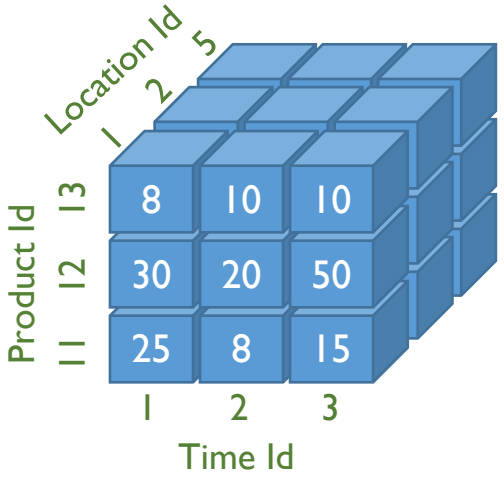
pid	pname	category	price
11	Corn	Food	25
12	Galaxy 1	Phones	18
13	Peanuts	Food	2

Time

timeid	Date	Day
1	3/30/16	Wed.
2	3/31/16	Thu.
3	4/1/16	Fri.

Dimension Tables

- Multidimensional "Cube" of data



Multidimensional Data Model

Sales **Fact Table**

pid	timeid	locid	sales
11	1	1	25
11	2	1	8
11	3	1	15
12	1	1	30
12	2	1	20
12	3	1	50
12	1	1	8
13	2	1	10
13	3	1	10
11	1	2	35
11	2	2	22
11	3	2	10
12	1	2	26

Locations

locid	city	state	country
1	Omaha	Nebraska	USA
2	Seoul		Korea
5	Richmond	Virginia	USA

Products

pid	pname	category	price
11	Corn	Food	25
12	Galaxy 1	Phones	18
13	Peanuts	Food	2

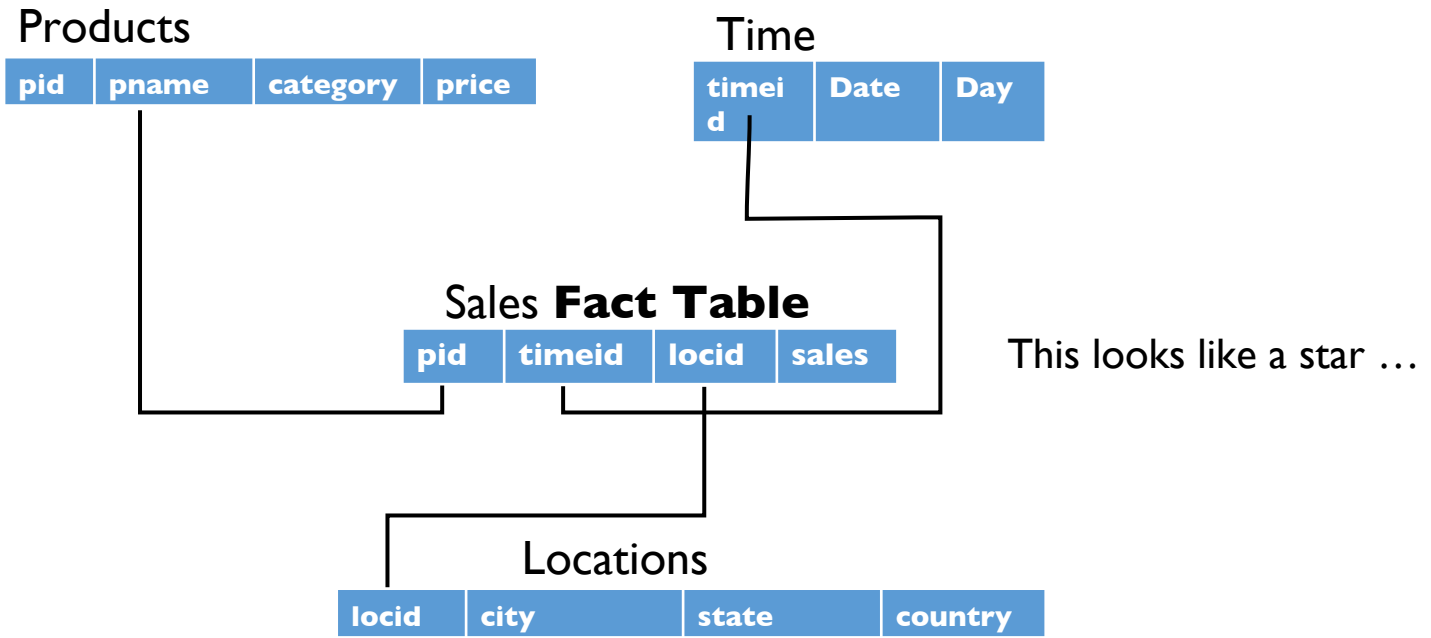
Time

timeid	Date	Day
1	3/30/16	Wed.
2	3/31/16	Thu.
3	4/1/16	Fri.

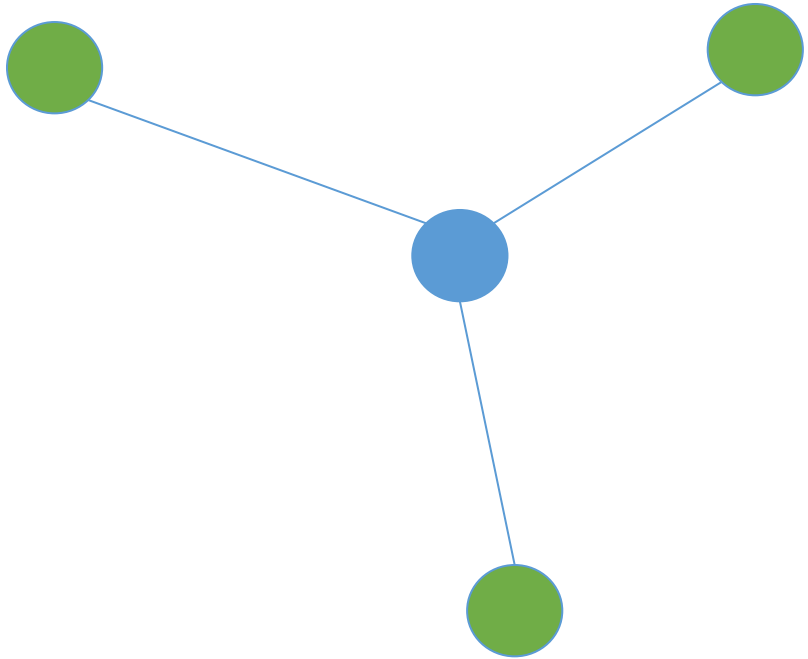
Dimension Tables

- Fact Table
  - Minimizes redundant info
  - Reduces data errors
- Dimensions
  - Easy to manage and summarize
  - Rename: Galaxy1 Phablet
- Normalized Representation
- How do we do analysis?
  - **Joins!**

The Star Schema

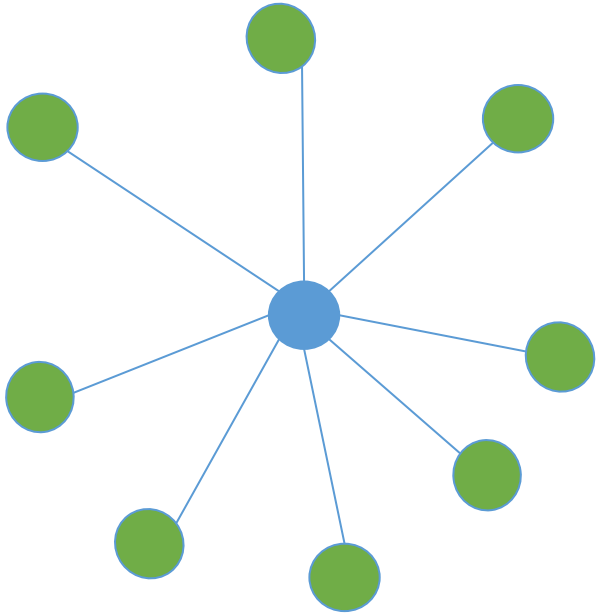


The Star Schema



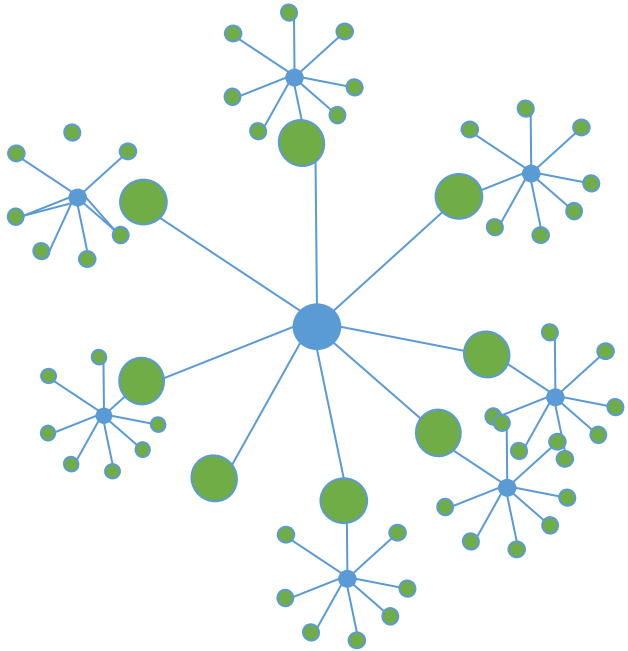
This looks like a star …

The Star Schema



This looks like a star …

The Snowflake Schema



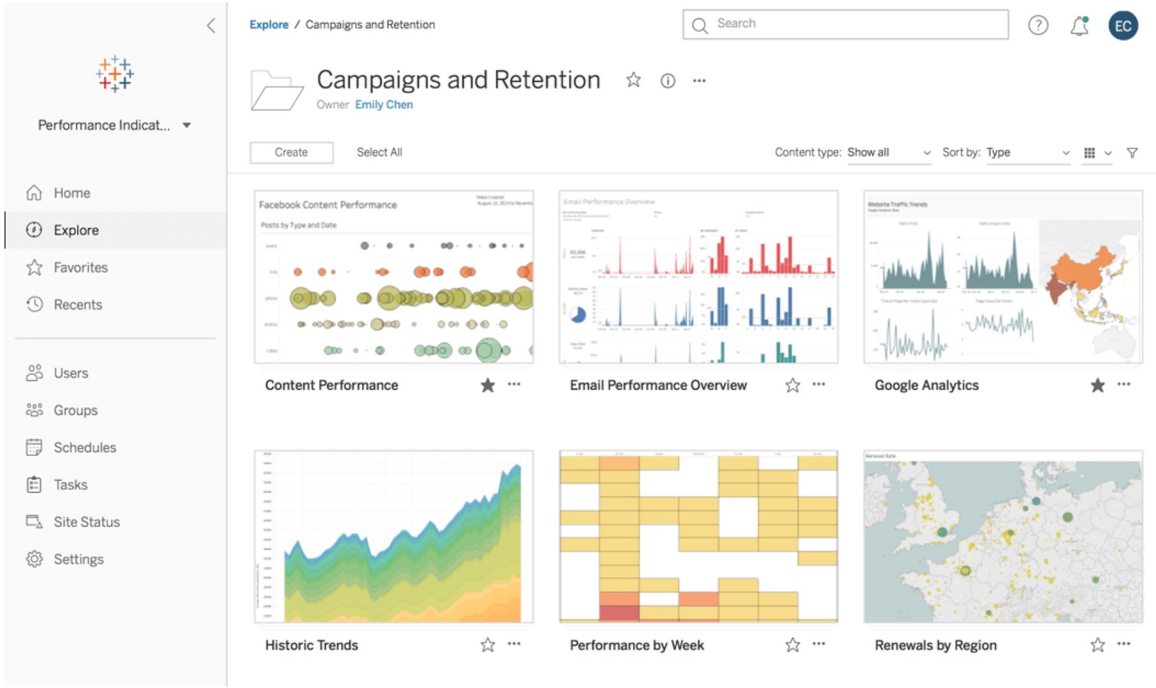
This looks like a snowflake …

More details in COSE371 & COSE444

OnLine Analytics Processing (**OLAP**)

Users interact with multidimensional data:

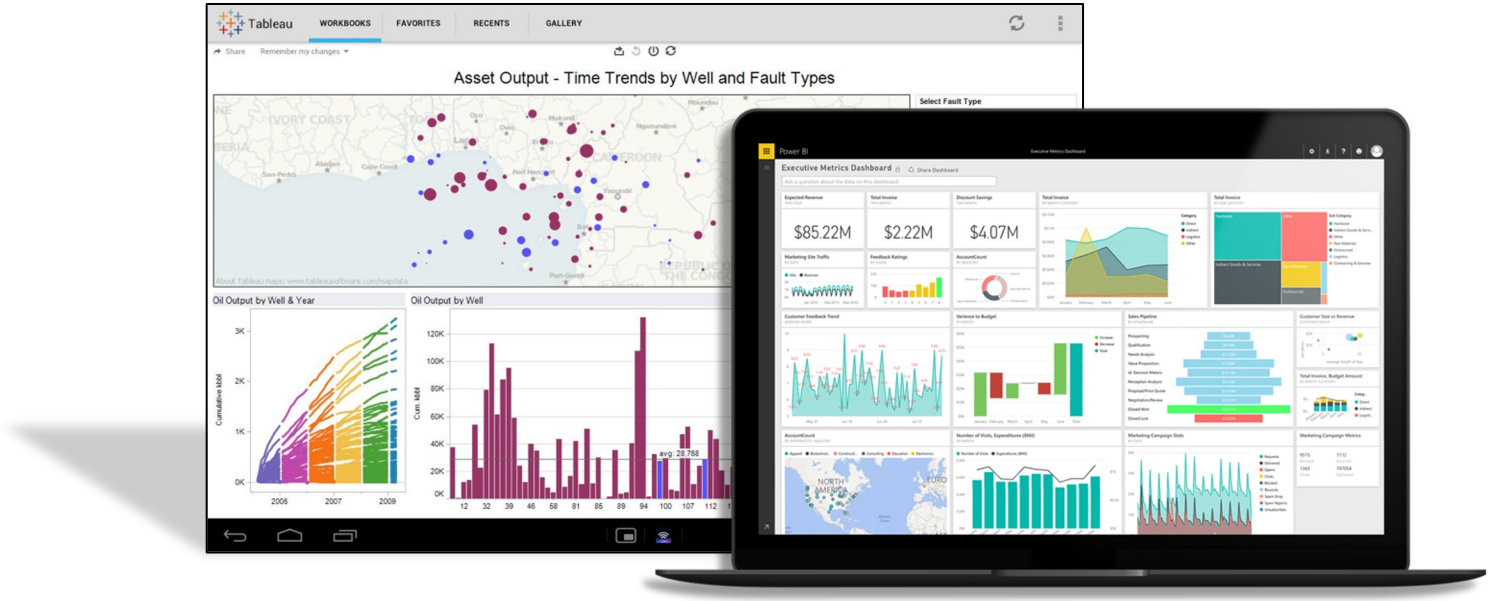
- Constructing ad-hoc and often complex SQL queries
- Using graphical tools that to construct queries
  - e.g., Tableau



Reporting and Business Intelligence (BI)

- Use high-level tools to interact with their data:
  - Automatically generate SQL queries
    - Queries can get big!

- Common!



Data Warehouse

Collects and organizes historical data from multiple sources

- How do we deal with semi-structured and unstructured data?
- Do we really want to force a schema on load?

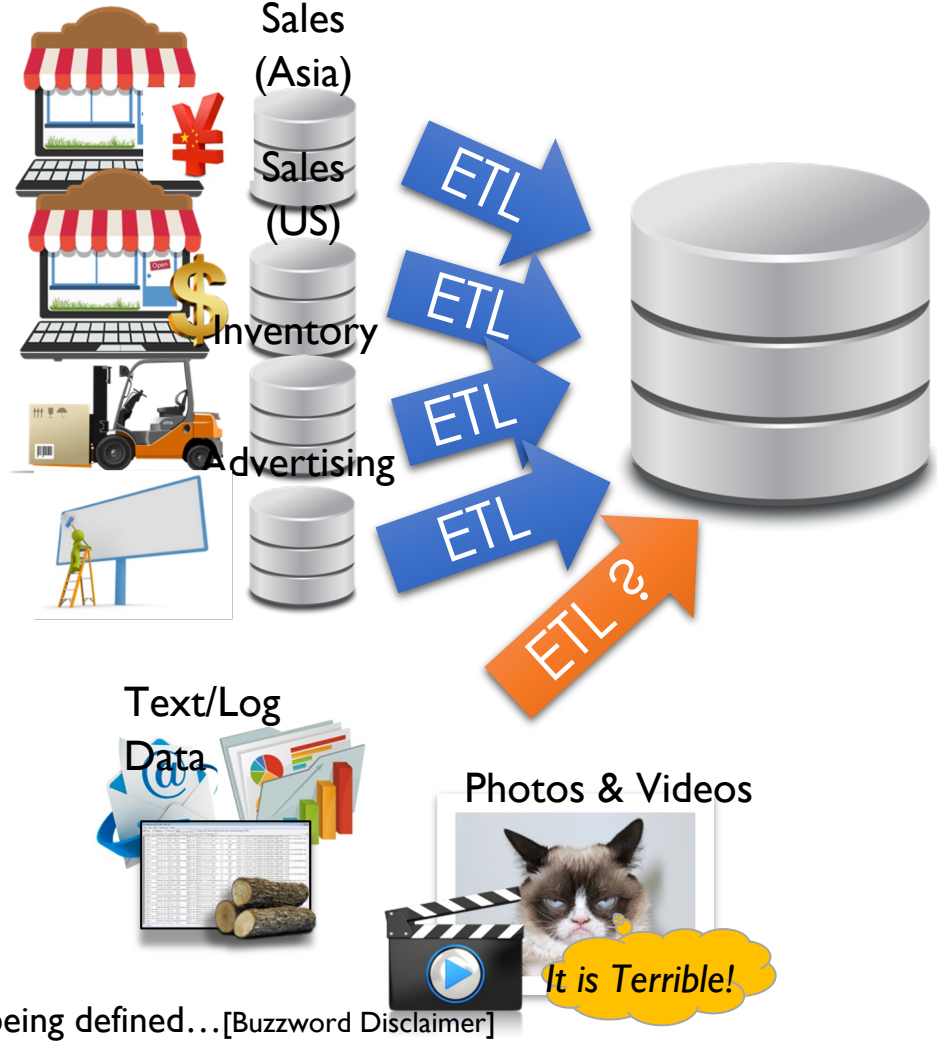
Data Warehouse

Collects and organizes historical data from multiple sources

- How do we deal with semi-structured and unstructured data?
- Do we really want to force a schema on load?

iid	date_taken	is_cat	is_grumpy	image_data
45123 1333	01-22-2016	1	1	
47234 2122	06-17-2017	0	1	
57182 7231	03-15-2009	0	0	
23847 2733	05-18-2018	0	0	

Unclear what a good schema for this image data might look like. Something like above will work, but it is inflexible!



Data Lake*

- Store a copy of all the data
  - in one place
  - in its original "natural" form
- Enable data consumers to choose how to transform and use data
  - *Schema on Read*

What could go wrong?

*Still being defined…[Buzzword Disclaimer]

The Dark Side of Data Lakes

- Cultural shift: *Curate* ⇒ *Save Everything!*
  - Noise begins to dominate signal
- Limited data governance and planning
  - **What** does it contain?
  - **When** and **who** created it?
- No cleaning and verification ⇒ lots of dirty data
- New tools are more complex and old tools no longer work

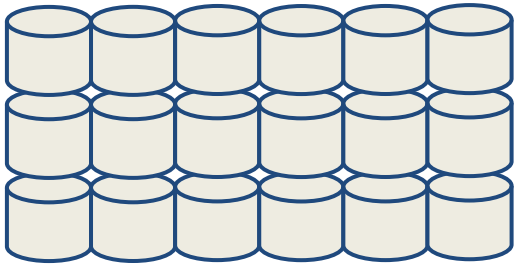
Enter the data scientist

A Brighter Future for Data Lakes

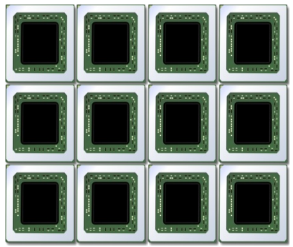
Enter the data scientist

- Data scientists bring new skills
  - Distributed data processing and cleaning
  - Machine learning, computer vision, and statistical sampling
- Technologies are improving
  - SQL over large files
  - Self describing file formats (e.g., Parquet) & catalog managers
- Organizations are evolving
  - Tracking data usage and file permissions
  - New job title: data scientist/engineer

Hardware for Big Data



Lots of hard drives



… and CPUs

How do we **store** and **compute** on large unstructured datasets?

- Requirements:
  - Handle very **large files** spanning **multiple computers**
  - Use **cheap** commodity devices that **fail frequently**
  - **Distributed data processing** quickly and **easily**

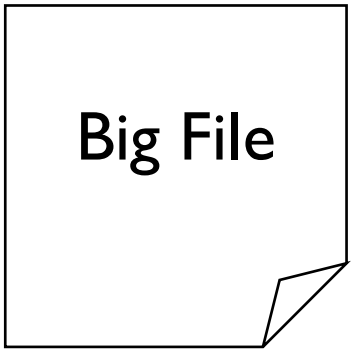
How do we **store** and **compute** on large unstructured datasets?

- Solutions:
  - **Distributed file systems** ⇒ spread data over multiple machines
    - Assume machine **failure** is common ⇒ **redundancy**
  - **Distributed computing** ⇒ load and process files on multiple machines concurrently
    - Assume machine **failure** is common ⇒ **redundancy**
    - **Functional programming** computational pattern ⇒ **parallelism**

Distributed File Systems

Storing very large files

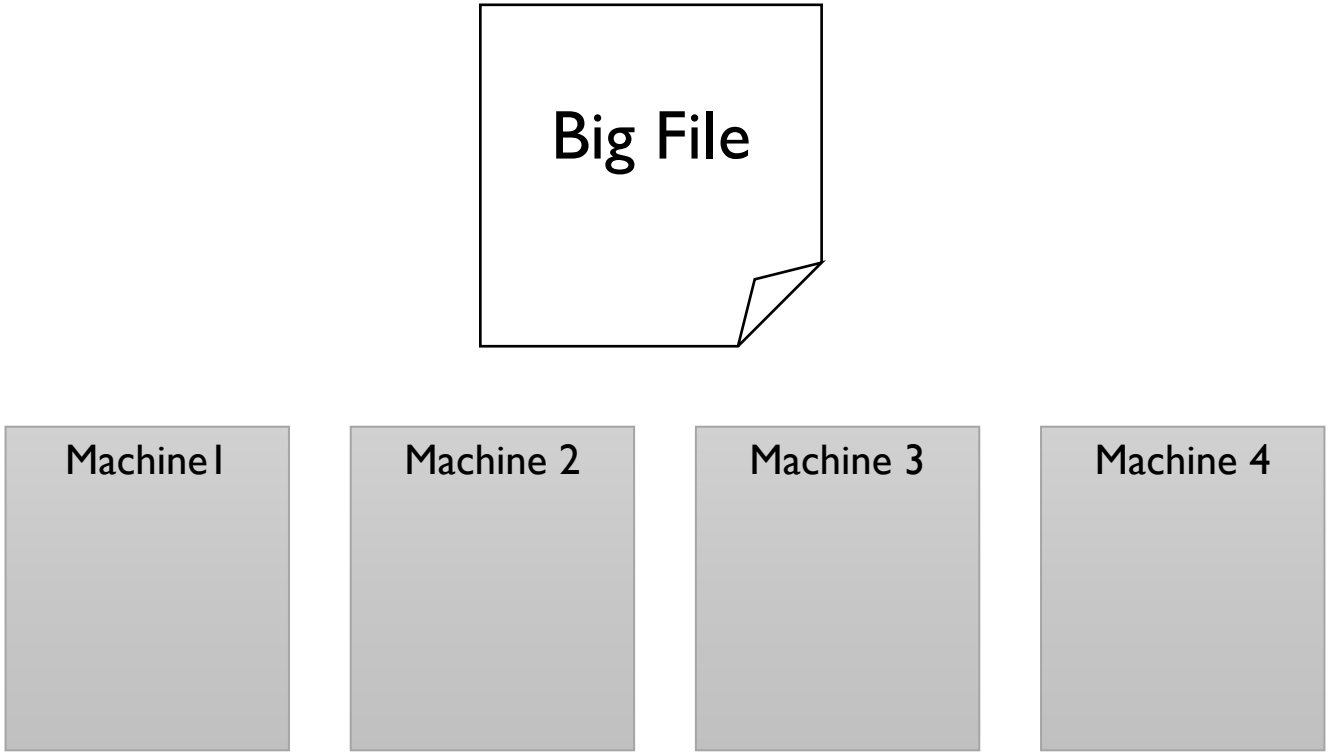
Fault Tolerant Distributed File Systems



How do we **store** and **access** very **large files** across **cheap** commodity devices ?

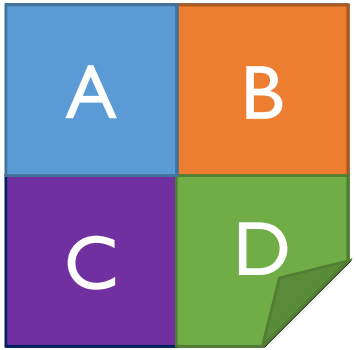
[Ghemawat et al., SOSP'03]

Fault Tolerant Distributed File Systems

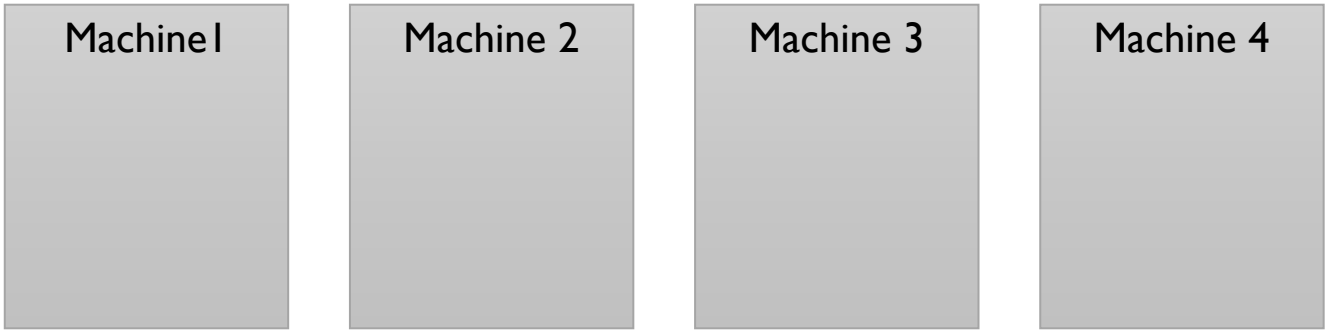


[Ghemawat et al., SOSP'03]

Fault Tolerant Distributed File Systems

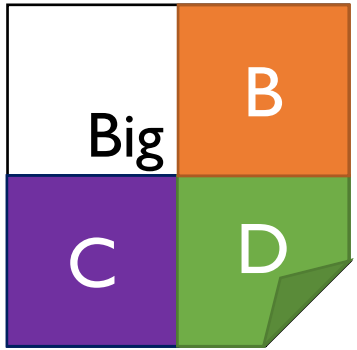


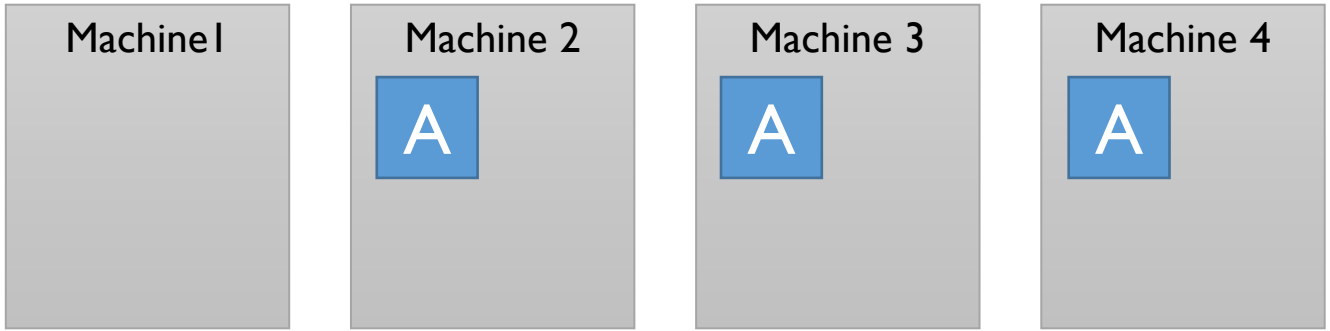
- Split the file into smaller parts
- How?
  - Ideally at record boundaries
  - What if records are big?



[Ghemawat et al., SOSP'03]

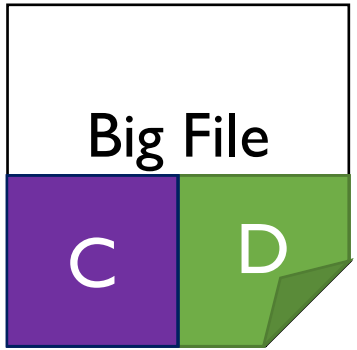
Fault Tolerant Distributed File Systems

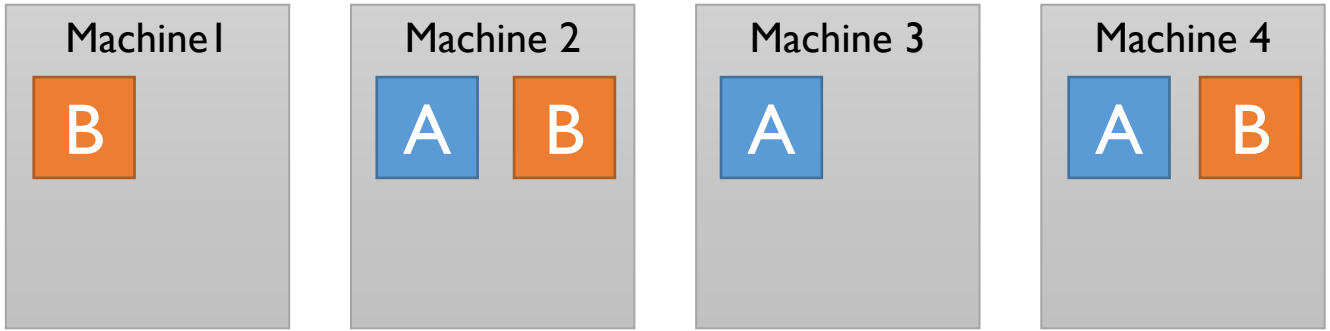




[Ghemawat et al., SOSP'03]

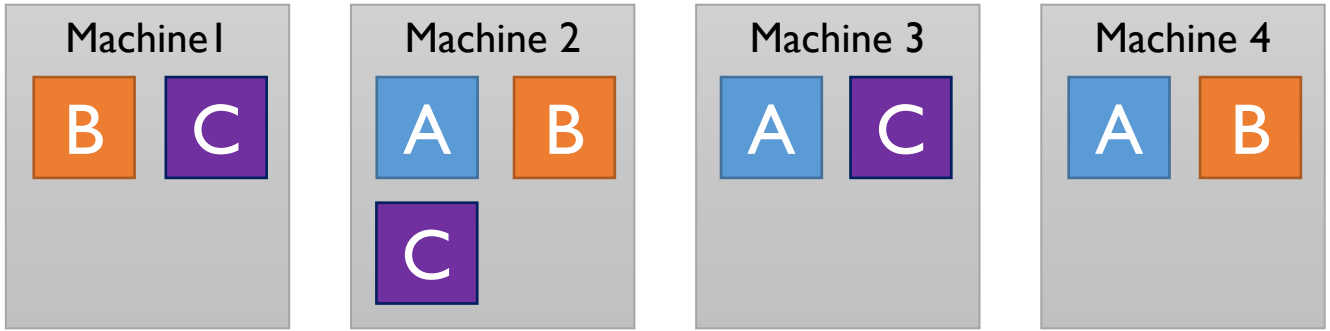
Fault Tolerant Distributed File Systems





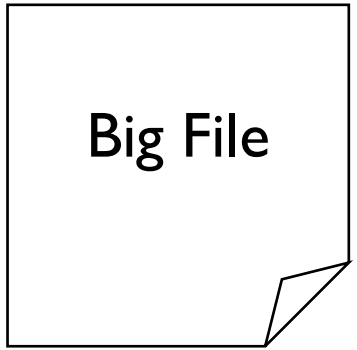
[Ghemawat et al., SOSP'03]

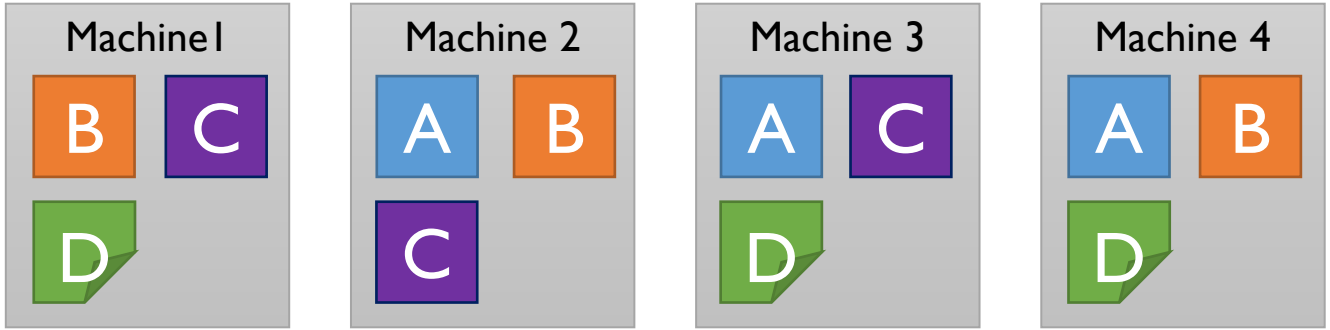
Fault Tolerant Distributed File Systems



[Ghemawat et al., SOSP'03]

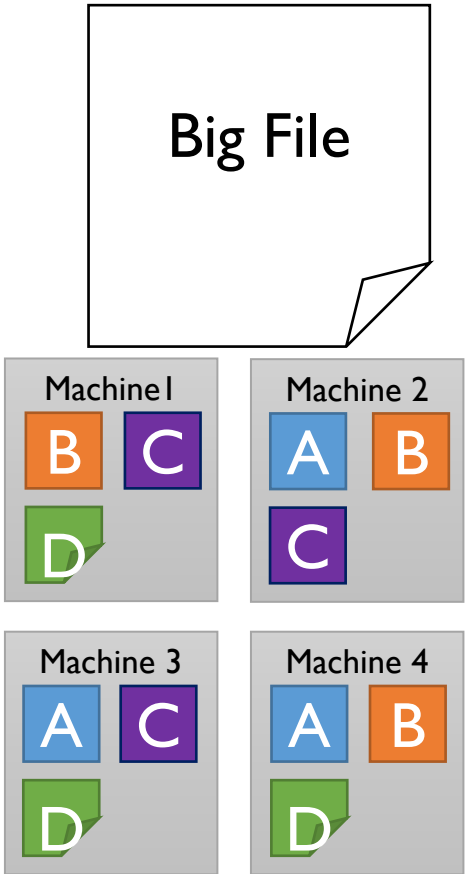
Fault Tolerant Distributed File Systems





[Ghemawat et al., SOSP'03]

Fault Tolerant Distributed File Systems



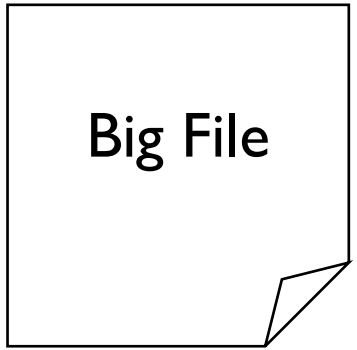
- Split large files over multiple machines
  - Easily support massive files spanning machines
- Read parts of file in parallel
  - Fast reads of large files
- Often built using cheap commodity storage devices

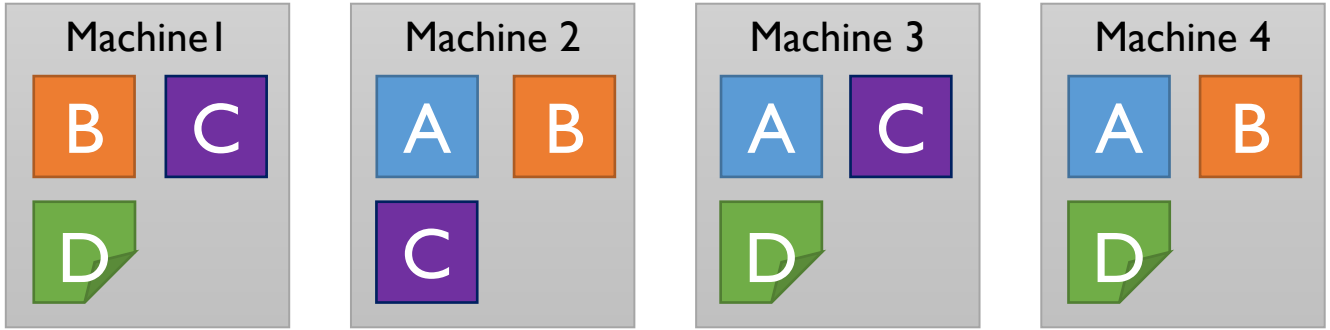
Cheap commodity storage devices will fail!

[Ghemawat et al., SOSP'03]

Fault Tolerant Distributed File Systems

Failure Event

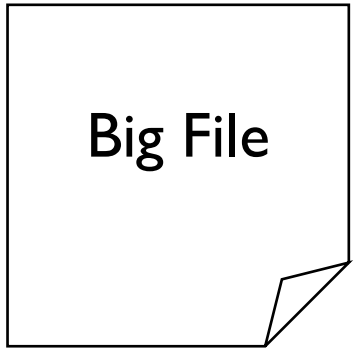


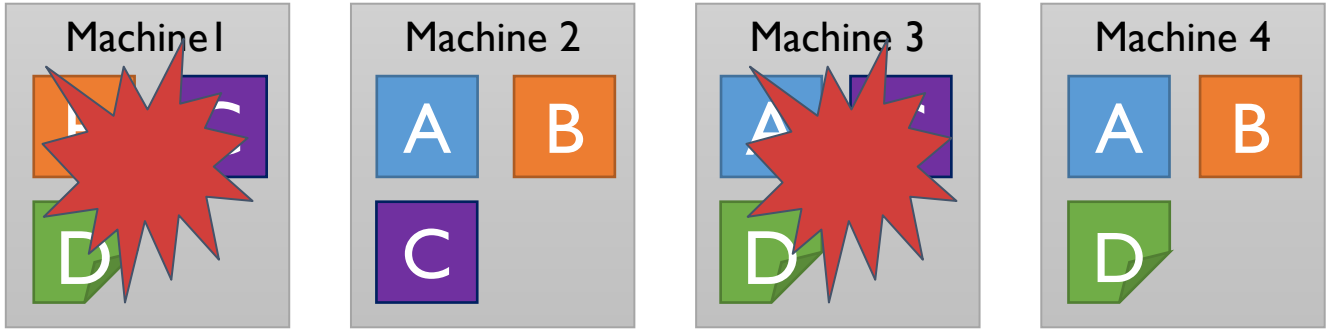


[Ghemawat et al., SOSP'03]

Fault Tolerant Distributed File Systems

Failure Event

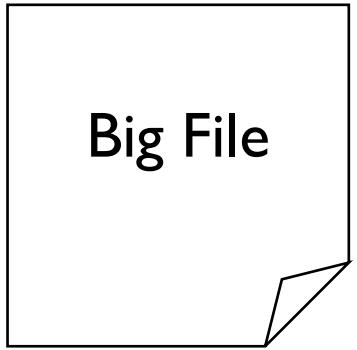


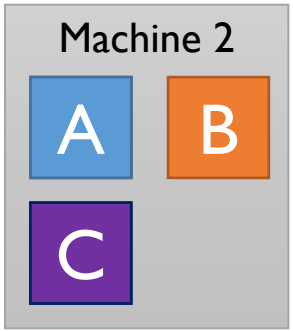


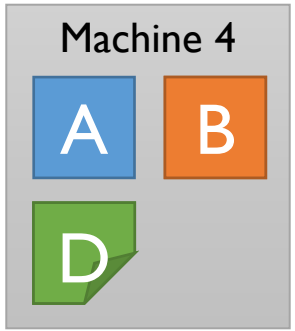
[Ghemawat et al., SOSP'03]

Fault Tolerant Distributed File Systems

Failure Event



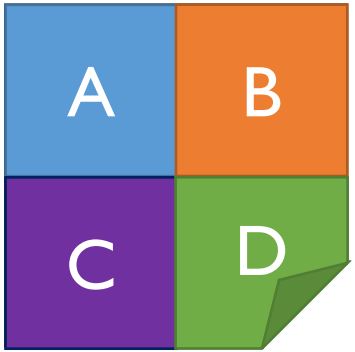


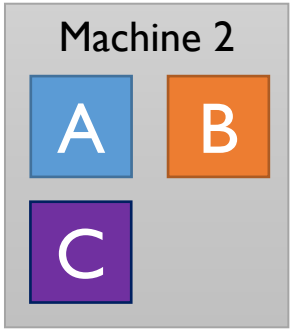


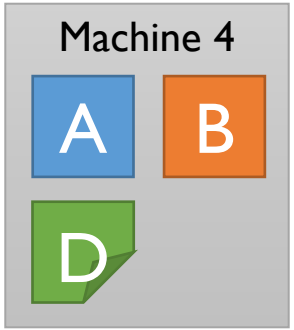
[Ghemawat et al., SOSP'03]

Fault Tolerant Distributed File Systems

Failure Event







[Ghemawat et al., SOSP'03]

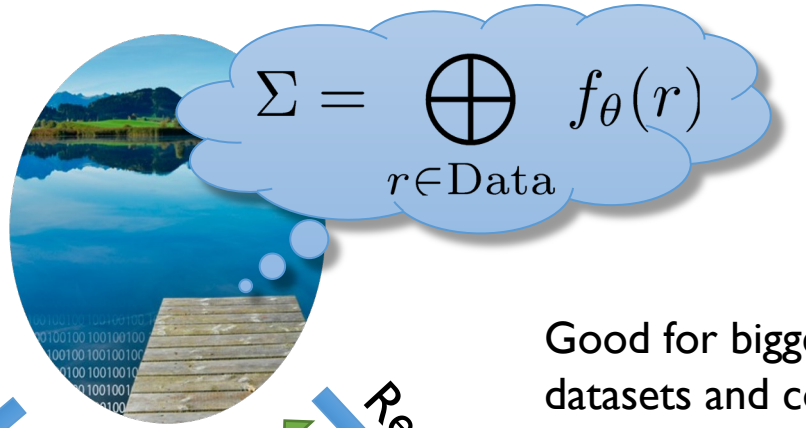
Map-Reduce Distributed Aggregation

Computing across very large files

Interacting With the Data

Good for smaller datasets

- Faster more natural interaction
- Lots of tools!



Good for bigger datasets and compute intensive tasks

Request Data Sample

Sample of Data

Query: f_θ

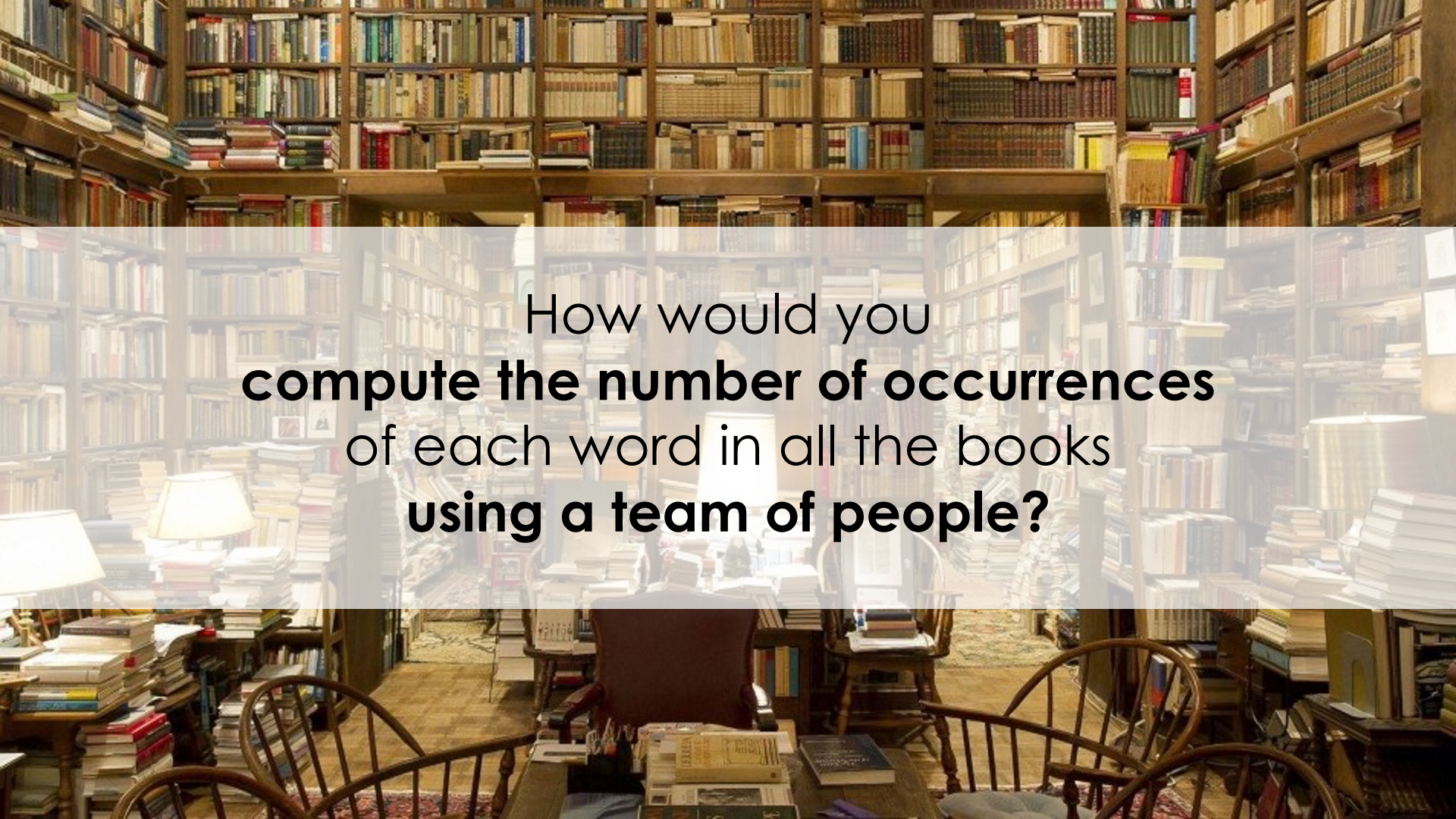
Response: Σ

Compute Locally

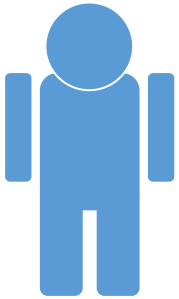
Algorithm

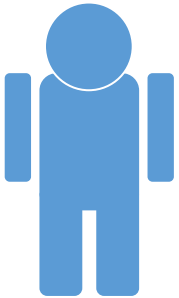
Algorithm

Cluster/Cloud Compute

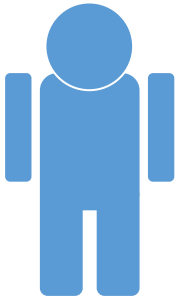

How would you
compute the number of occurrences
of each word in all the books
using a team of people?

Simple Solution

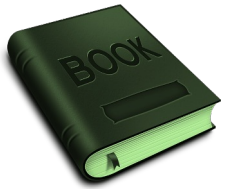


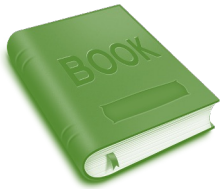


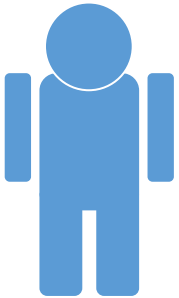
Simple Solution

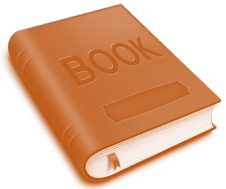


1) Divide Books Across Individuals

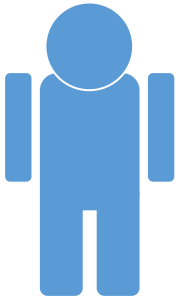




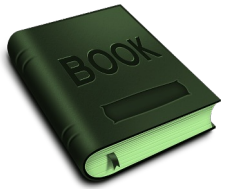


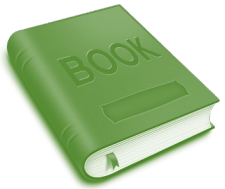


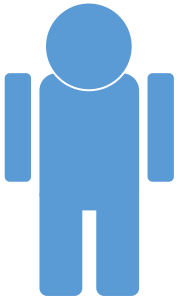
Simple Solution

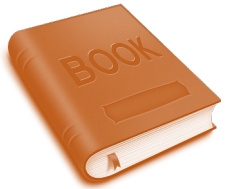


1) Divide Books Across Individuals







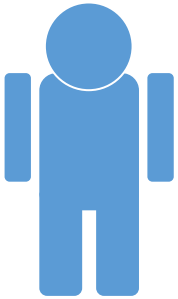


2) Compute Counts Locally

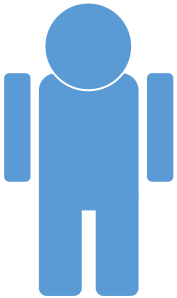
Word	Count
Apple	2
Bird	7
…	

Word	Count
Apple	0
Bird	1
…	

Simple Solution



1) Divide Books Across Individuals



2) Compute Counts Locally

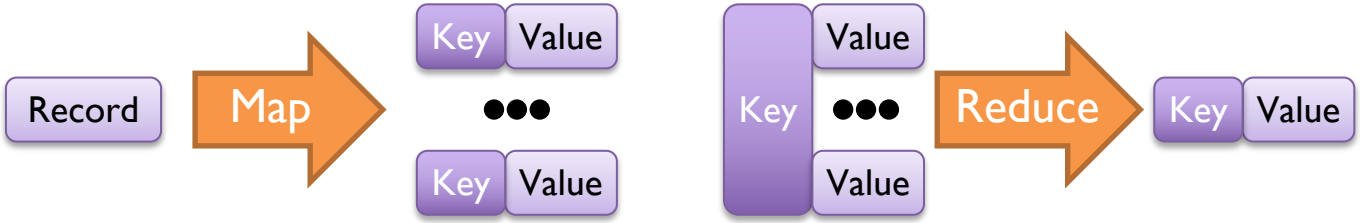
Word	Count
Apple	2
Bird	7
…	

Word	Count
Apple	0
Bird	1
…	

3) Aggregate Tables

Word	Count
Apple	2
Bird	8
…	

The Map Reduce Abstraction



Example: *Word-Count*

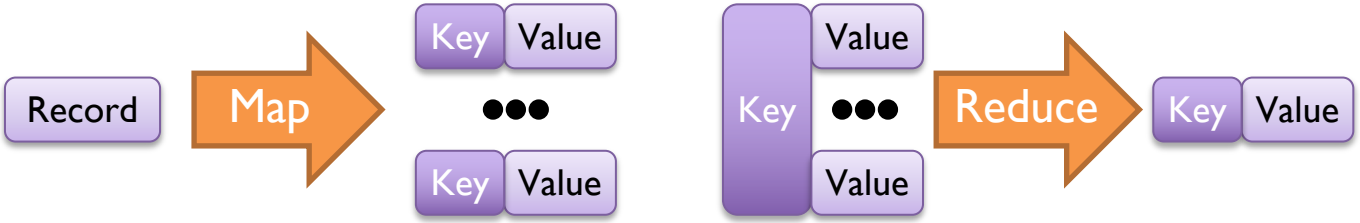
```
Map(book):
  for (word in set(book)):
    emit (word, book.count(word))
```

Key Value

```
Reduce(word, counts) {
  sum = 0
  for count in counts:
    sum += count
  emit (word, SUM(counts))
}
```

[Dean & Ghemawat, OSDI'04]

The Map Reduce Abstraction (Simpler)



Example: *Word-Count*

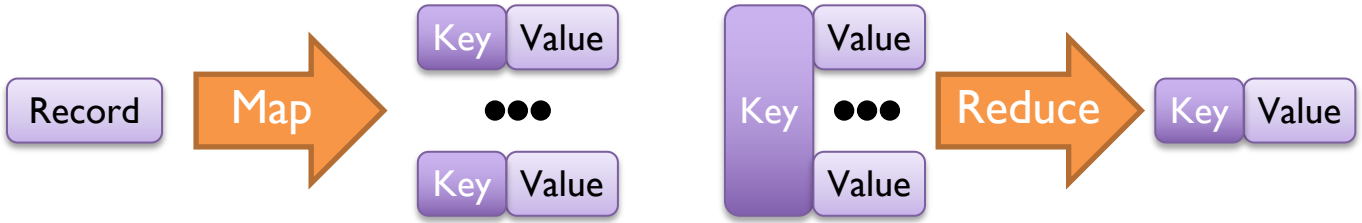
```
Map(book):
  for (word in book):
    emit (word, 1)
```

Key Value

```
Reduce(word, counts) {
  sum = 0
  for count in counts:
    sum += count
  emit (word, SUM(counts))
}
```

[Dean & Ghemawat, OSDI'04]

The Map Reduce Abstraction (General)



Example: *Word-Count*

```
Map(record, f):
  for (key in record):
    emit (key, f(key))
```

Key Value

```
Reduce(key, values, f) {
  agg = f(values[0], values[1])
  for value in values[2:]:
      agg = f(agg, value)
  emit (word, agg)
}
```

Map: Deterministic
Reduce: Commutative and Associative

[Dean & Ghemawat, OSDI'04]

Key properties of Map And Reduce

- **Deterministic Map:** allows for re-execution on failure
  - If some computation is lost we can always re-compute
  - Issues with samples? *Can use random seeds to mitigate*
- **Commutative Reduce:** *allows for re-order of operations*
  - Reduce(A, B) = Reduce(B, A)
  - Example (addition): A + B = B + A
- **Associative Reduce:** *allows for regrouping of operations*
  - Reduce(Reduce(A, B), C) = Reduce(A, Reduce(B, C))
  - Example (max): max(max(A, B), C) = max(A, max(B, C))
    - Warning: Floating point operations (e.g., addition) are not guaranteed associative

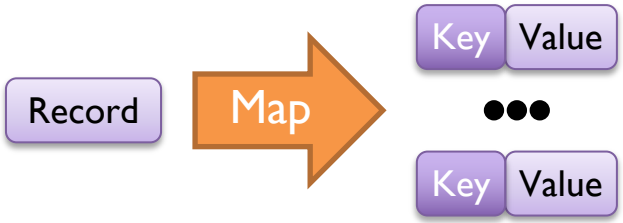
Question

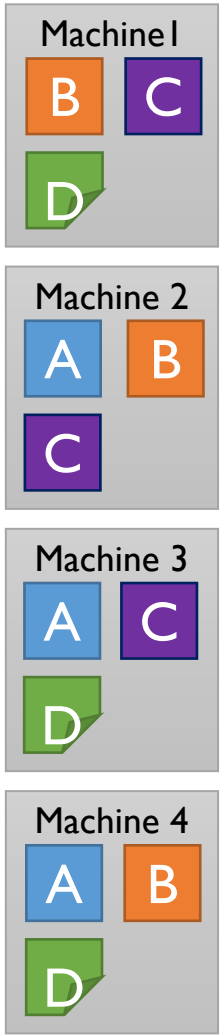
- Suppose our reduction function computes a*b + 1
- Suppose we have 3 values associated with the key ‘cat’. What is the result of the reduction operation?

Question

- Suppose our reduction function computes a*b + 1
- Suppose we have 3 values associated with the key ‘cat’. What is the result of the reduction operation?
  - It depends!
    - 3*2 + 1 ⇒ 7, and then 7*1 + 1 = 8
    - 1*2 + 1 ⇒ 3, and then 3*3 + 1 = 10
    - 3*1 …

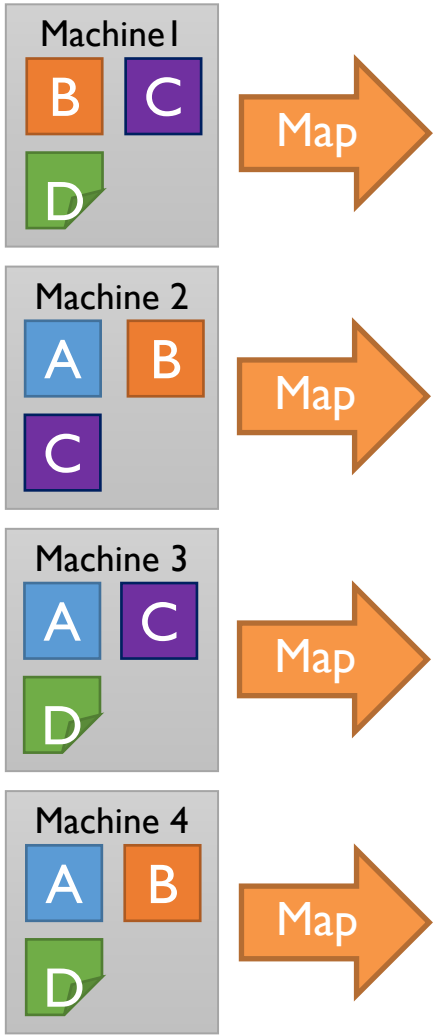
Executing Map Reduce





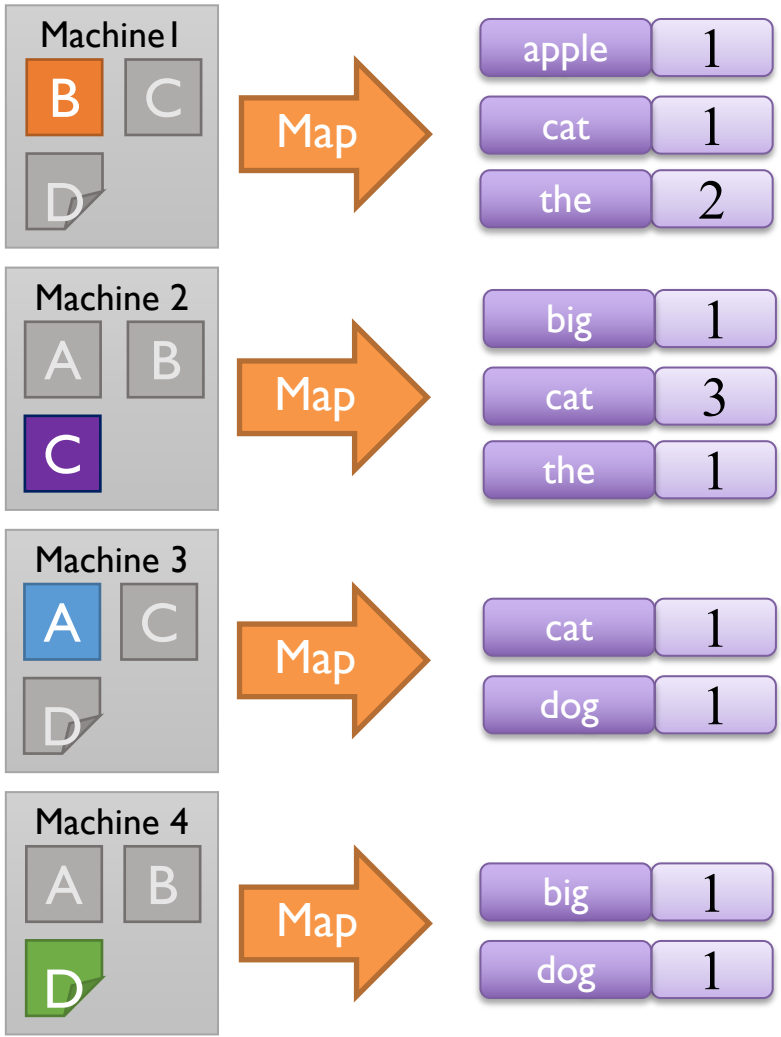
Executing Map Reduce

Distributing the Map Function



Executing Map Reduce

Distributing the Map Function



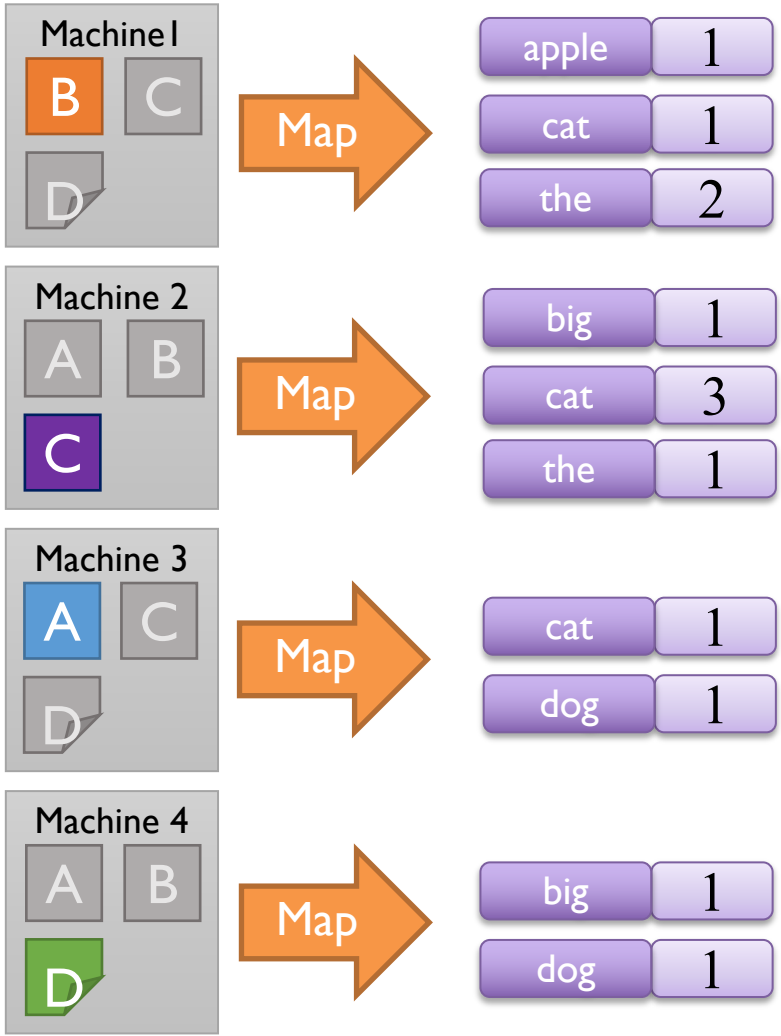
Executing Map Reduce

Output is cached for fast recovery on node failure

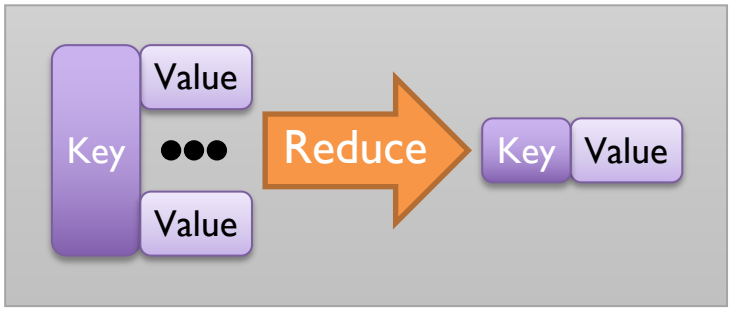
The map function applied to a local part of the big file

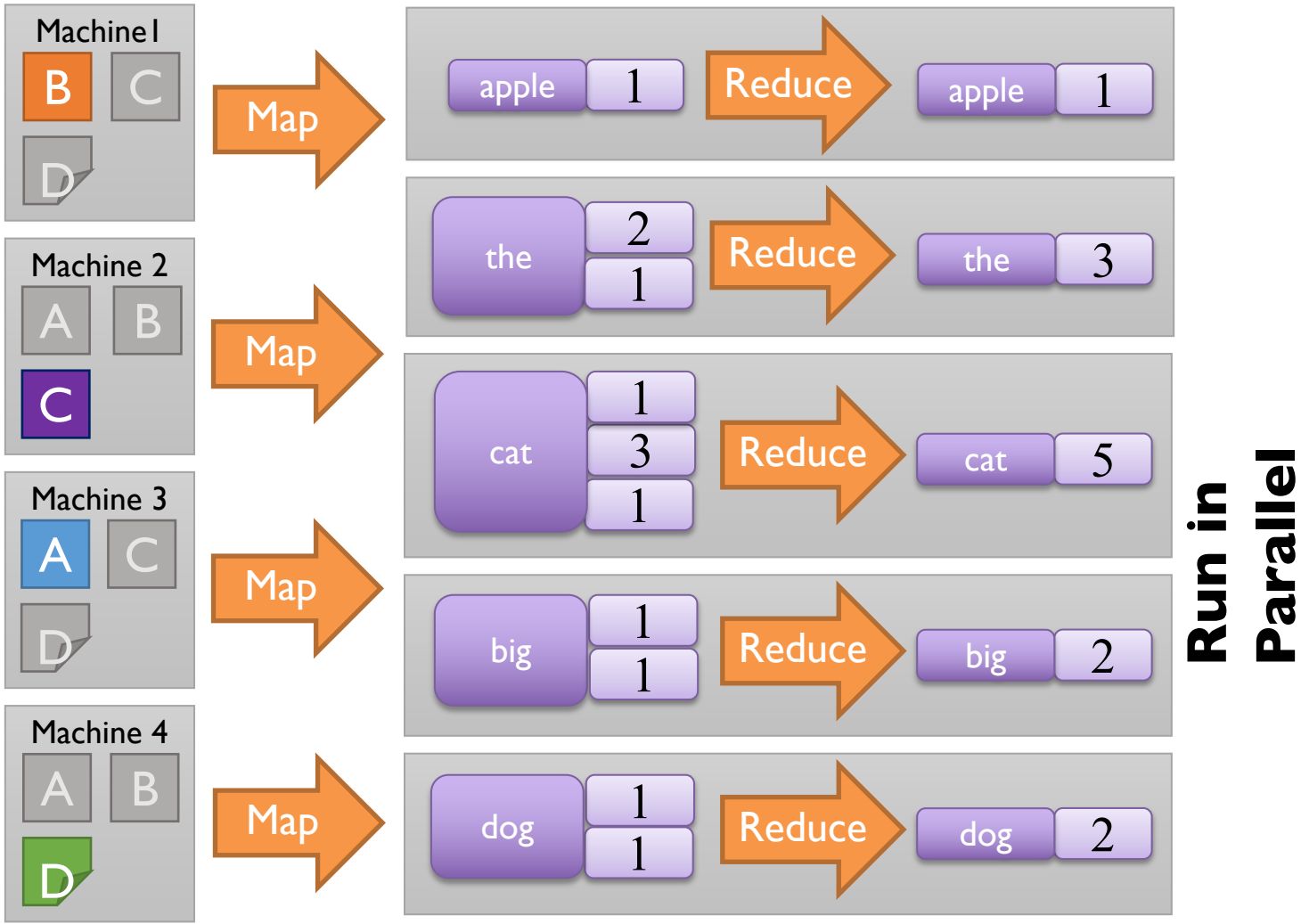
Run in Parallel

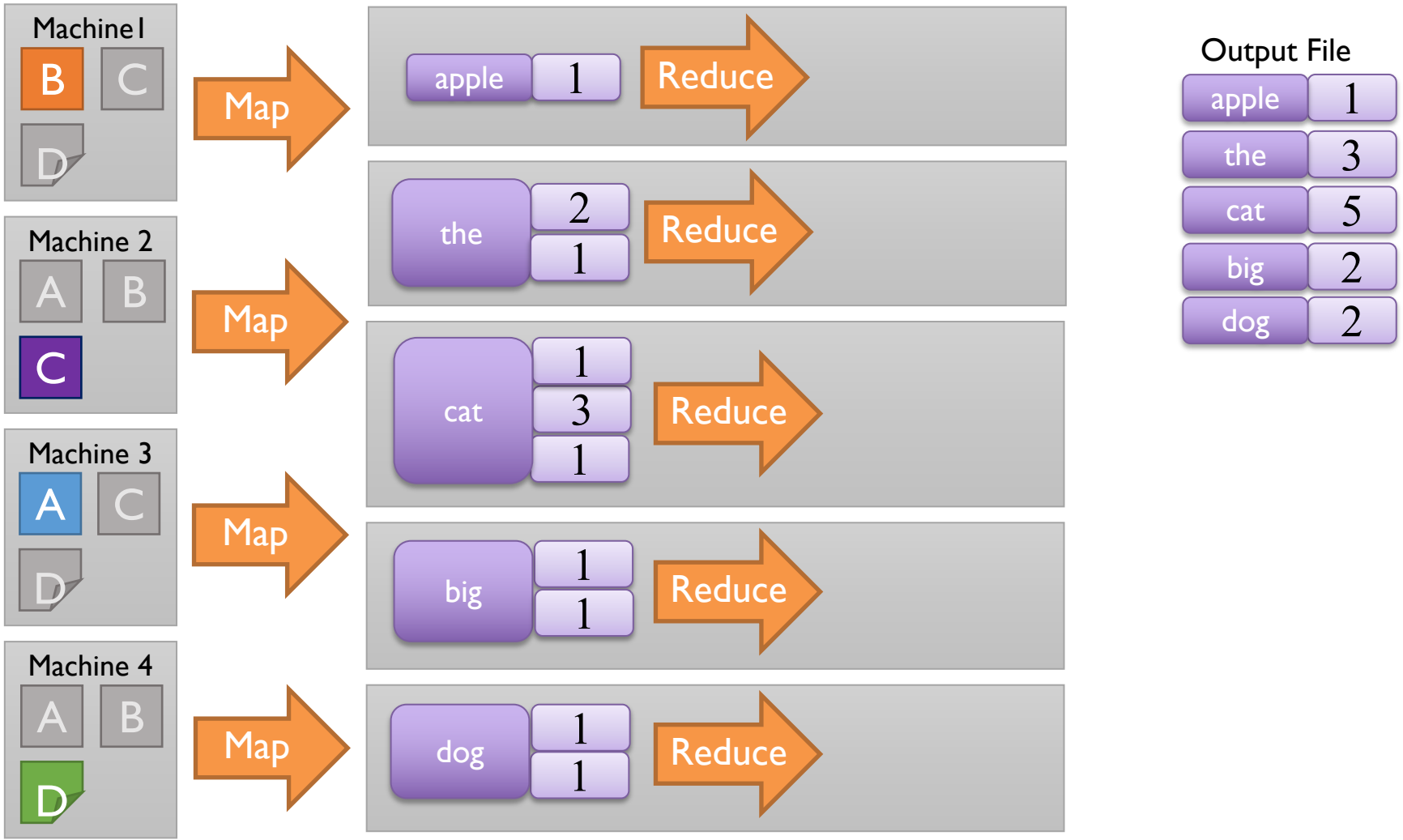
Executing Map Reduce

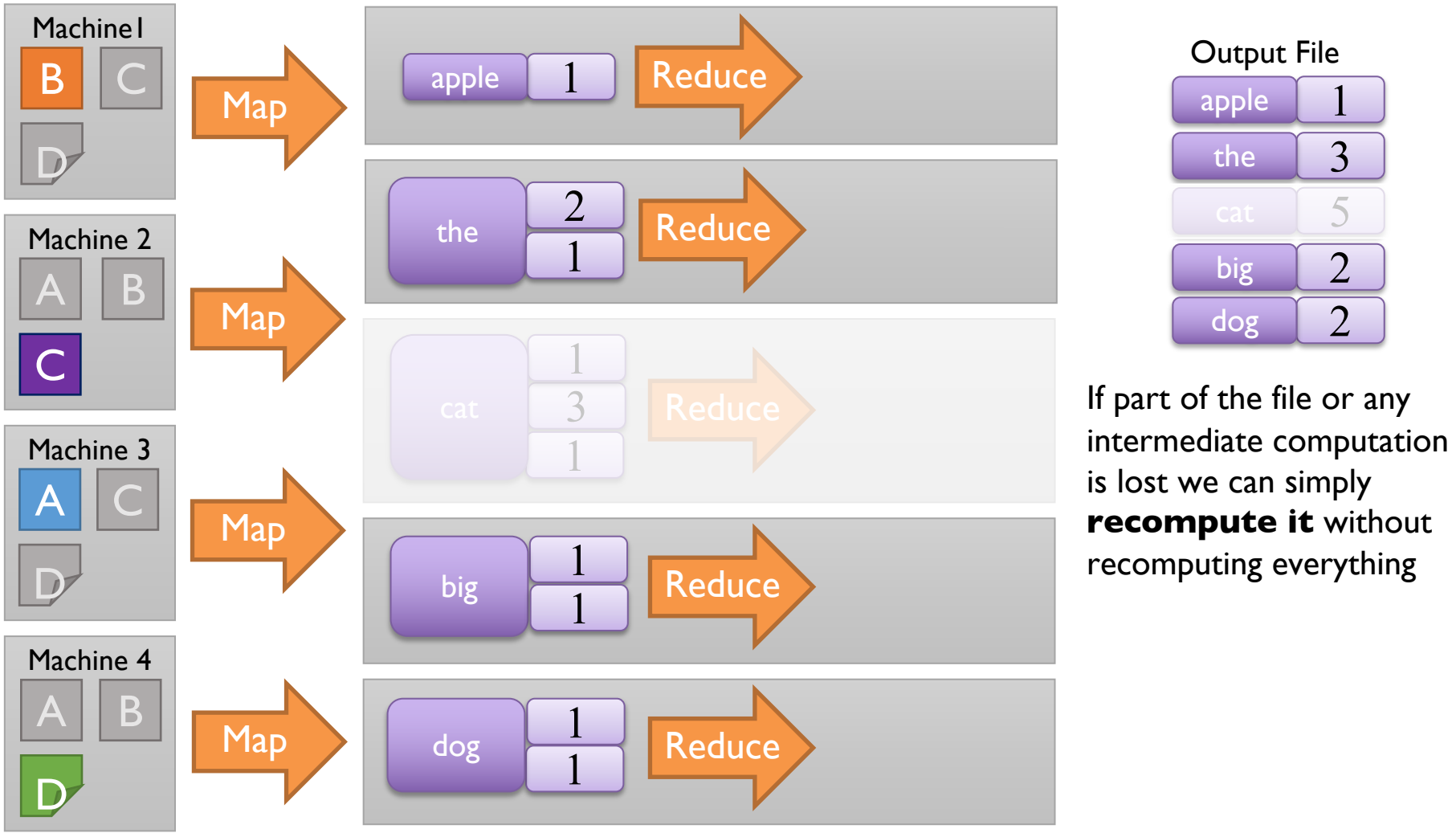


Reduce function can be run on many machines …




Machine 1
B
C
D
Machine 2
A
B
C
Machine 3
A
C
D
Machine 4
A
B
D
Map
Map
Map
Map
apple
1
Reduce
apple
1
the
2
1
Reduce
the
3
cat
1
3
1
Reduce
cat
5
big
1
1
Reduce
big
2
dog
1
1
Reduce
dog
2
Run in Parallel

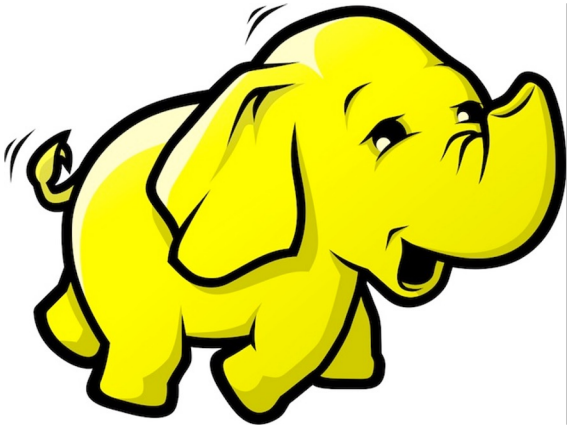

Machine 1
B
C
D
Machine 2
A
B
C
Machine 3
A
C
D
Machine 4
A
B
D
Map
Map
Map
Map
apple
1
Reduce
the
2
1
Reduce
cat
1
3
1
Reduce
big
1
1
Reduce
dog
1
1
Reduce
Output File
apple
1
the
3
cat
5
big
2
dog
2



If part of the file or any intermediate computation is lost we can simply **recompute it** without recomputing everything

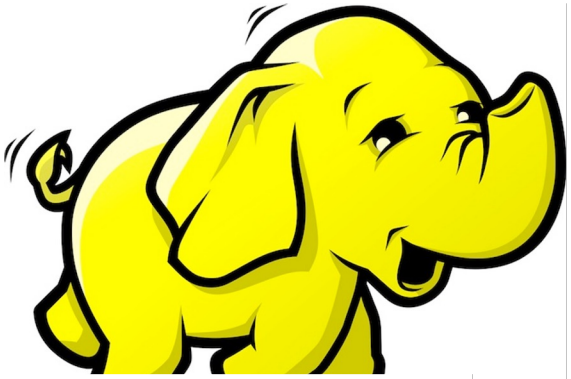
Map Reduce Technologies (Optional)

Apache Hadoop



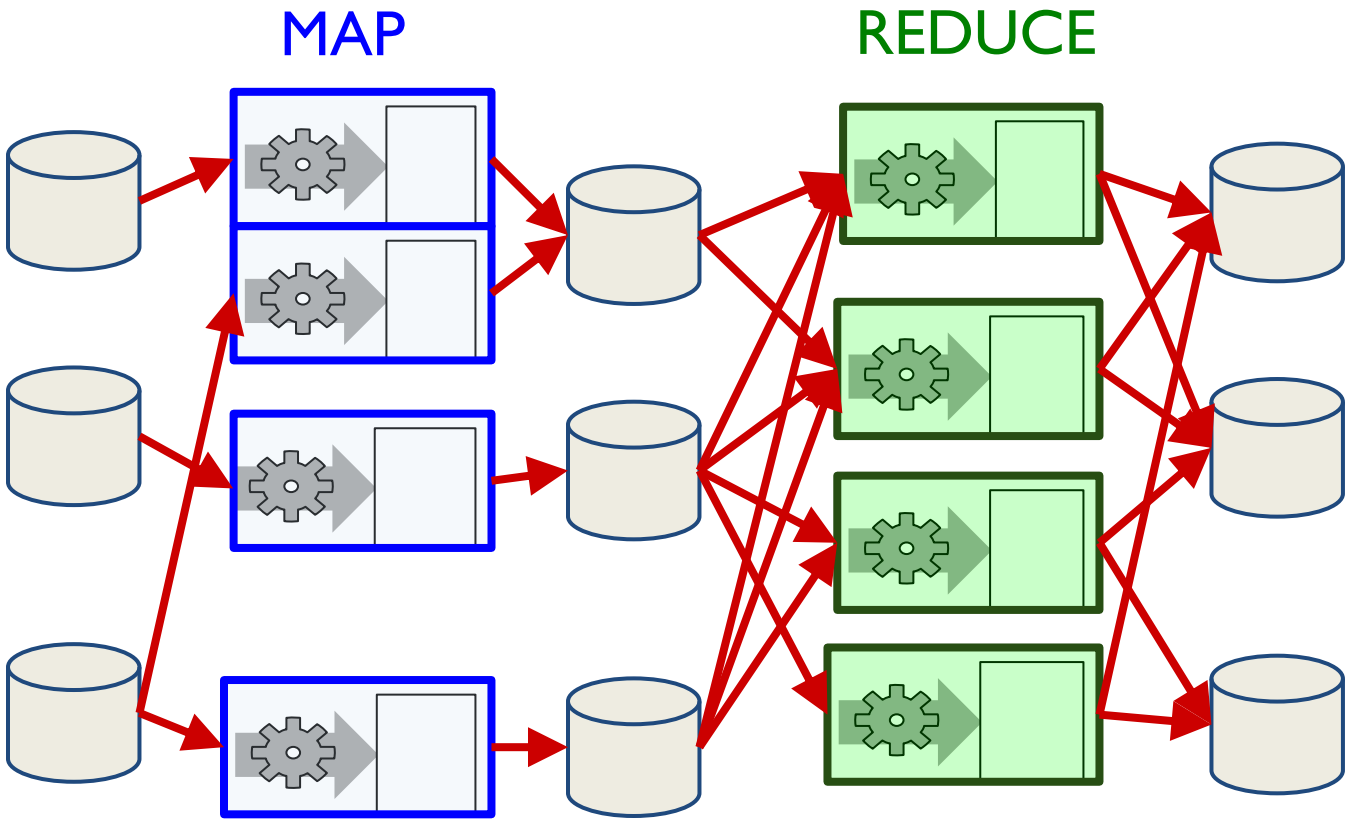
- First open-source map-reduce software
  - Managed by Apache foundation
- Based on Google's approaches
  - Google File System
  - MapReduce
- Companies formed around Hadoop:
  - Cloudera
  - Hortonworks
  - MapR

Apache Hadoop



- Very active open source ecosystem
- Several key technologies
  - **HDFS:** Hadoop File System
  - **MapReduce:** map-reduce compute framework
  - **YARN:** Yet another resource negotiator
  - **Hive:** SQL queries over MapReduce
  - …
- Downside: Tedious to use!
  - Joey: Word count example from before is 100s of lines of Java code.

Map Reduce/Apache Hadoop: Distributed Execution

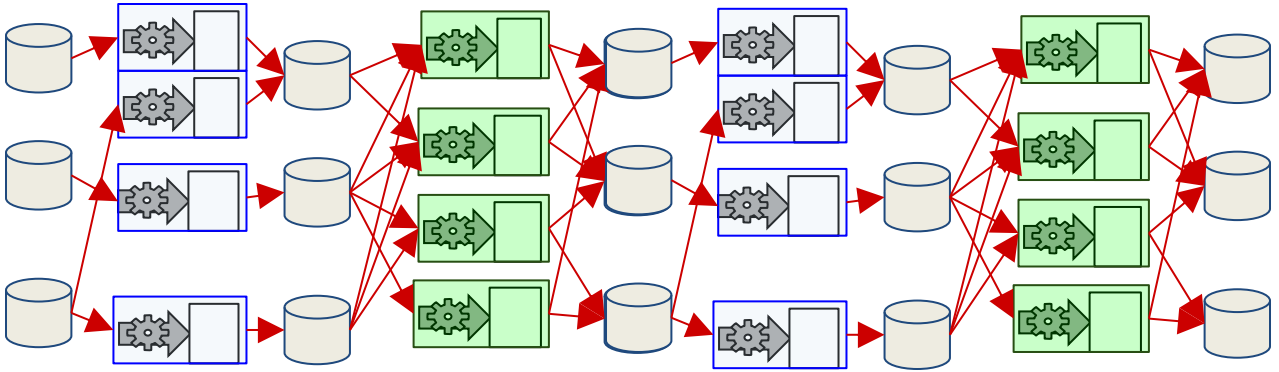


Each stage passes through the hard drives

https://hadoop.apache.org/

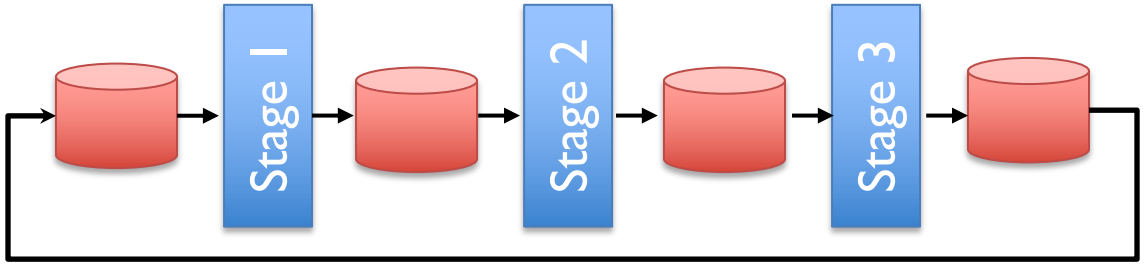
Map Reduce: Iterative Jobs

- Iterative jobs involve a lot of disk I/O for each repetition



Map Reduce: Iterative Jobs

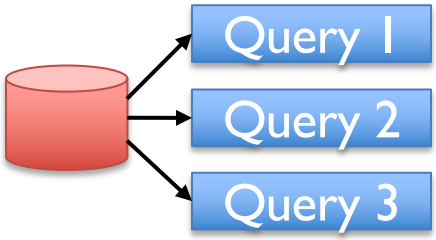
- Iterative jobs involve a lot of disk I/O for each repetition

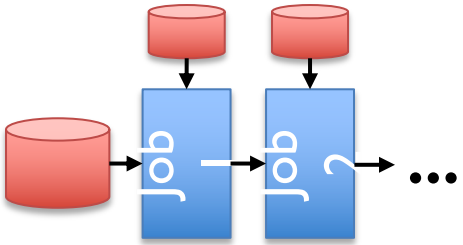


Disk I/O is very slow!

Map Reduce: Issues

- Using Map Reduce for complex jobs, interactive queries and online processing involves ***lots of disk I/O***





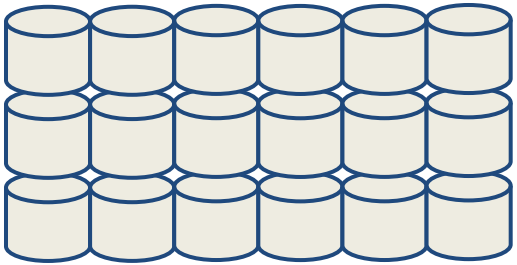
Interactive mining

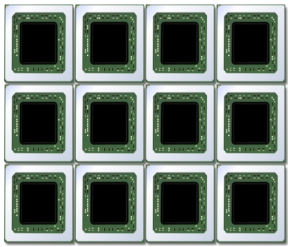
Stream processing

Also, iterative jobs

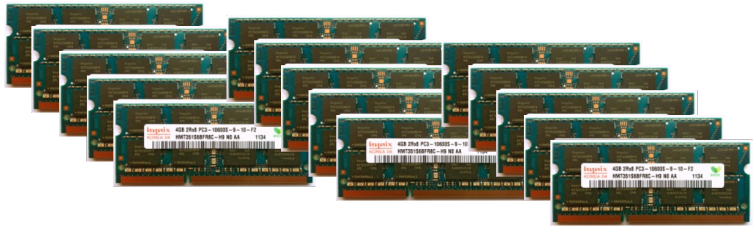
Disk I/O is very slow

Today's Hardware for Big Data





Lots of hard drives … and CPUs



… and memory!

Opportunity

- Keep more data ***in-memory***
- Create new distributed execution engine:

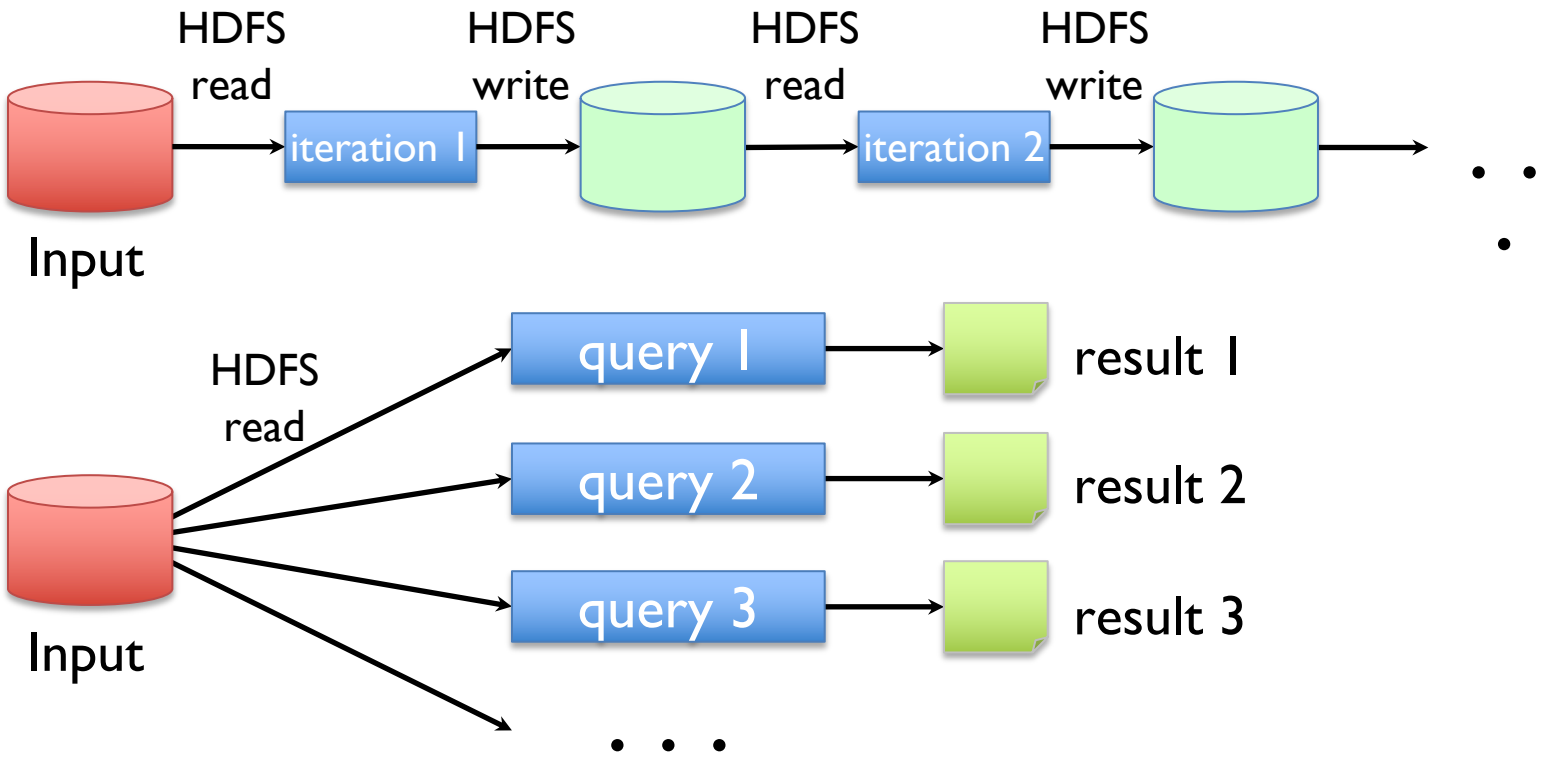
Apache  In-Memory Dataflow System

Developed in the UC Berkeley AMP Lab

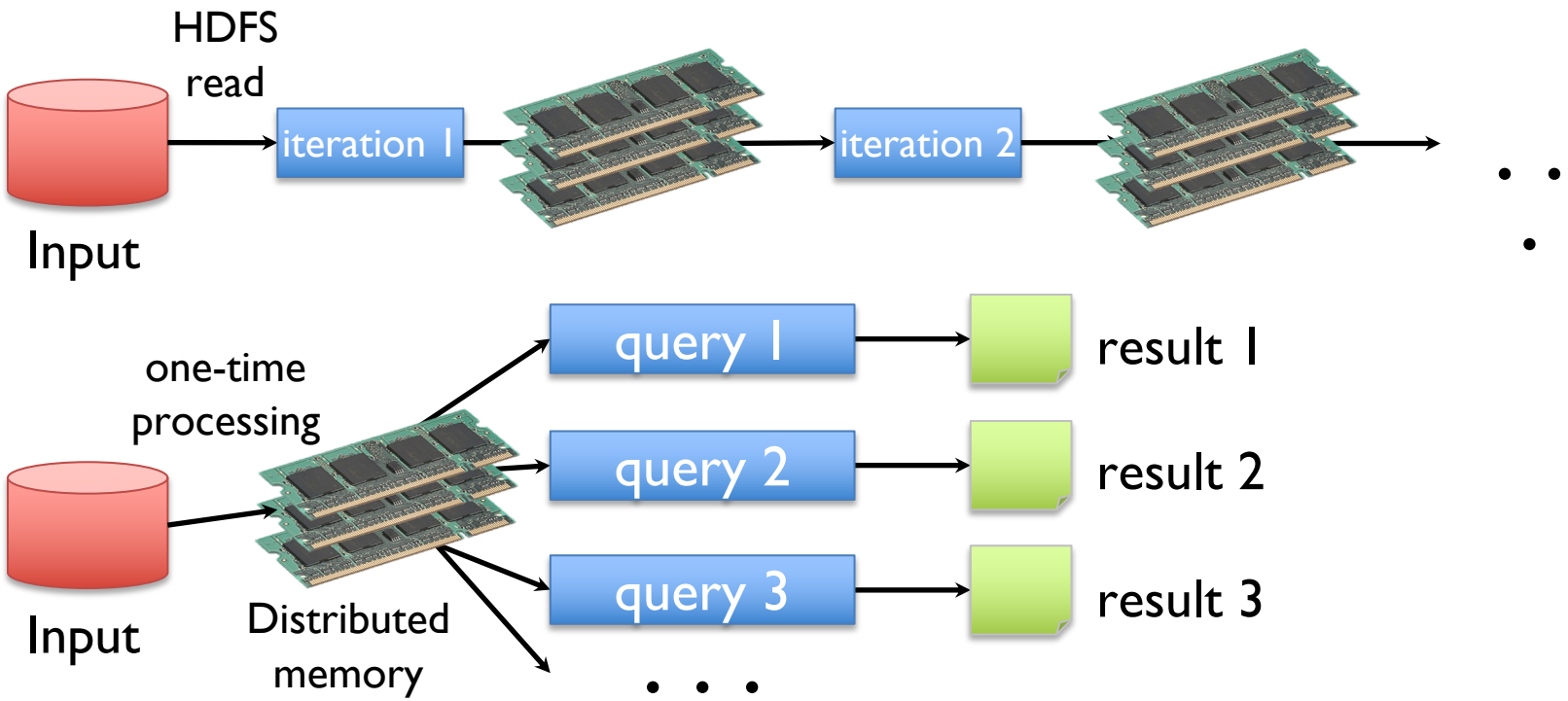
M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica. *Spark: cluster computing with working sets*. HotCloud'10

M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M.J. Franklin, S. Shenker, I. Stoica. *Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing*, NSDI 2012

Use Memory Instead of Disk



In-Memory Data Sharing



Up to100x faster than network and disk

What Is 

- Parallel execution engine for big data processing
- **General**: efficient support for multiple workloads
- **Easy** to use: 2-5x less code than Hadoop MR
  - High level API's in Python, Java, and Scala
- **Fast**: up to 100x faster than Hadoop MR
  - Can exploit in-memory when available
  - Low overhead scheduling, optimized engine

Spark Programming Abstraction

- *Write programs in terms of transformations on distributed datasets*
- Resilient Distributed Datasets (RDDs)
  - Distributed collections of objects that can stored in memory or on disk
  - Built via parallel transformations (map, filter, …)
  - Automatically rebuilt on failure

Slide provided by M.

RDD: Resilient Distributed Datasets

- Collections of objects partitioned & distributed across a cluster
  - Stored in RAM or on Disk
  - Resilient to failures
- Operations
  - Transformations
  - Actions

Operations on RDDs

- Transformations f(RDD) ⇒ RDD
  - Lazy (not computed immediately)
  - E.g., “map”, “filter”, “groupBy”

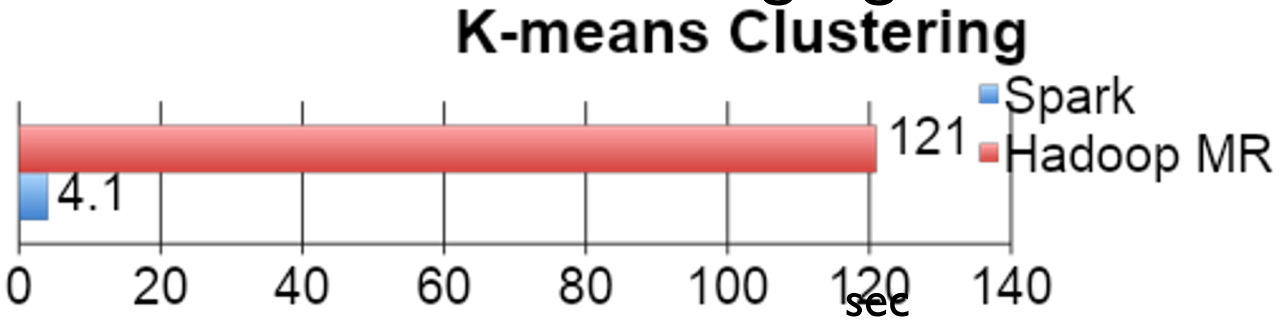
- Actions:
  - Triggers computation
  - E.g. “count”, “collect”, “saveAsTextFile”

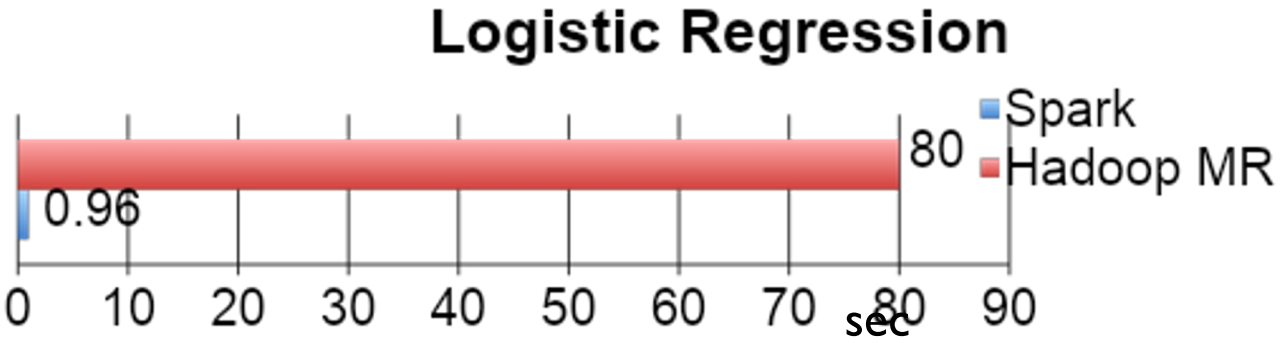
Spark and Map Reduce Differences

	Hadoop Map Reduce	Spark
Storage	Disk only	In-memory or on disk
Operations	Map and Reduce	Map, Reduce, Join, Sample, etc…
Execution model	Batch	Batch, interactive, streaming
Programming environments	Java	Scala, Java, R, and Python

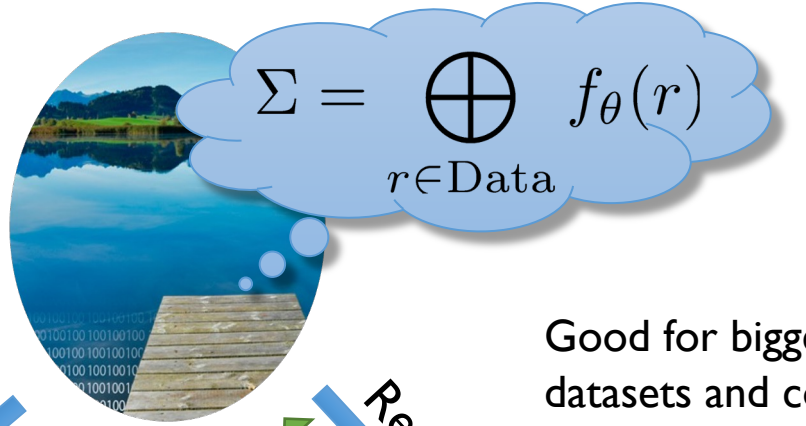
In-Memory Can Make a Big Difference

Two iterative Machine Learning algorithms:





Interacting With the Data

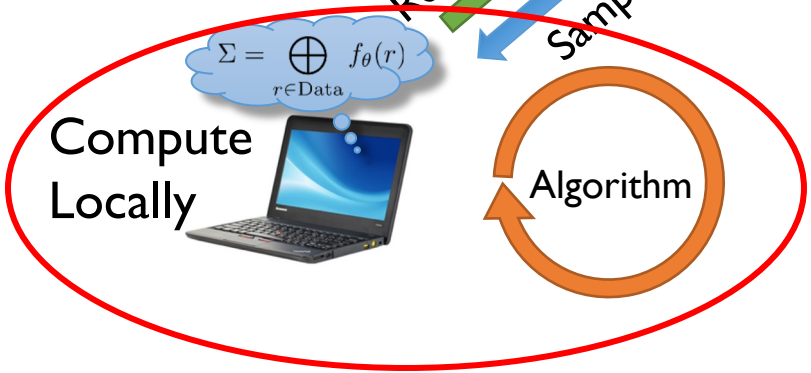


Good for smaller datasets

- Faster more natural interaction
- Lots of tools!

Request Data Sample

Sample of Data



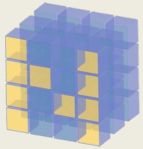
Good for bigger datasets and compute intensive tasks

Query: f_θ

Response: Σ

Data Science Tools

Tools efficient for O(1MB)

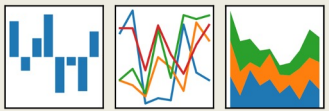


NumPy

matplotlib

pandas

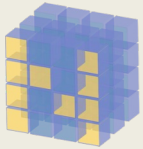
$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$



Data Science Tools

Data Science Tools

Tools efficient for O(1MB)

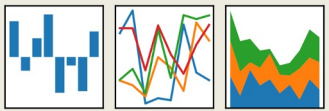


NumPy

matplotlib

pandas

$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$



Tools efficient for O(100s GB+)

Data Science Landscape: Today

Tools efficient for O(1MB)

NumPy

scikit learn

matplotlib

pandas

$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$

Tools efficient for O(100s GB+)

hadoop

DASK

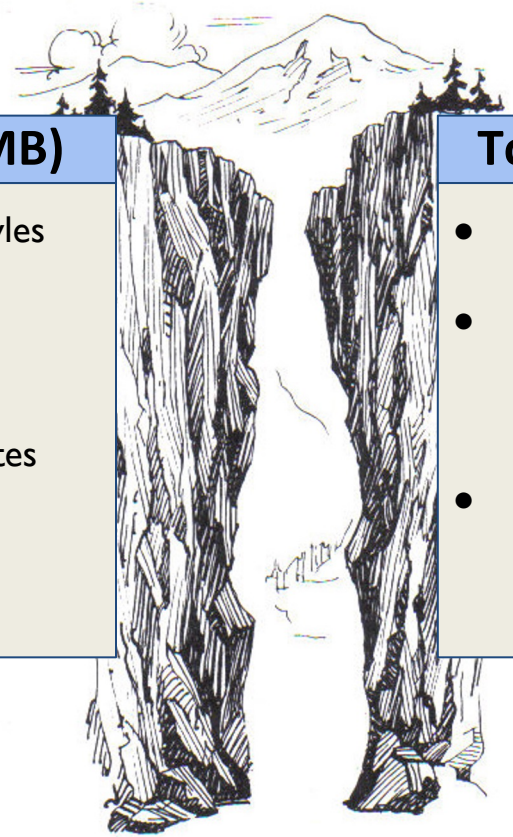
HIVE

APACHE Spark™

The world has changed since these tools were built

- Many cores
- Lots of memory
- Lots of data

Data Science Landscape: Today



Tools efficient for O(1MB)

- Follow typical programming styles
- Tools are widely used and understood - in production
- The majority of college graduates will already know these tools
- No scalability

Tools efficient for O(100s GB+)

- Difficult to debug
- Requires distributed computing knowledge
  - Must understand partitioning
  - Lazy evaluation - hated
- Designed by systems people for systems people
  - New APIs that do the same thing

Data Science Landscape: Today

MODIN

Tools efficient for O(1MB)

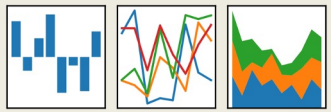
NumPy

scikit learn

matplotlib

pandas

$y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$



Tools efficient for O(100s GB+)

hadoop

DASK

HIVE

APACHE Spark™

Drop-in replacement for pandas

Modin Next Generation Dataframe

Accelerate your pandas workloads by changing one line of code

```
# import pandas as pd
import modin.pandas as pd
```

Modin is a DataFrame for datasets from 1MB to 1TB+

- 4x speedup on 4-core laptop

Installation

Modin can be installed from PyPI:

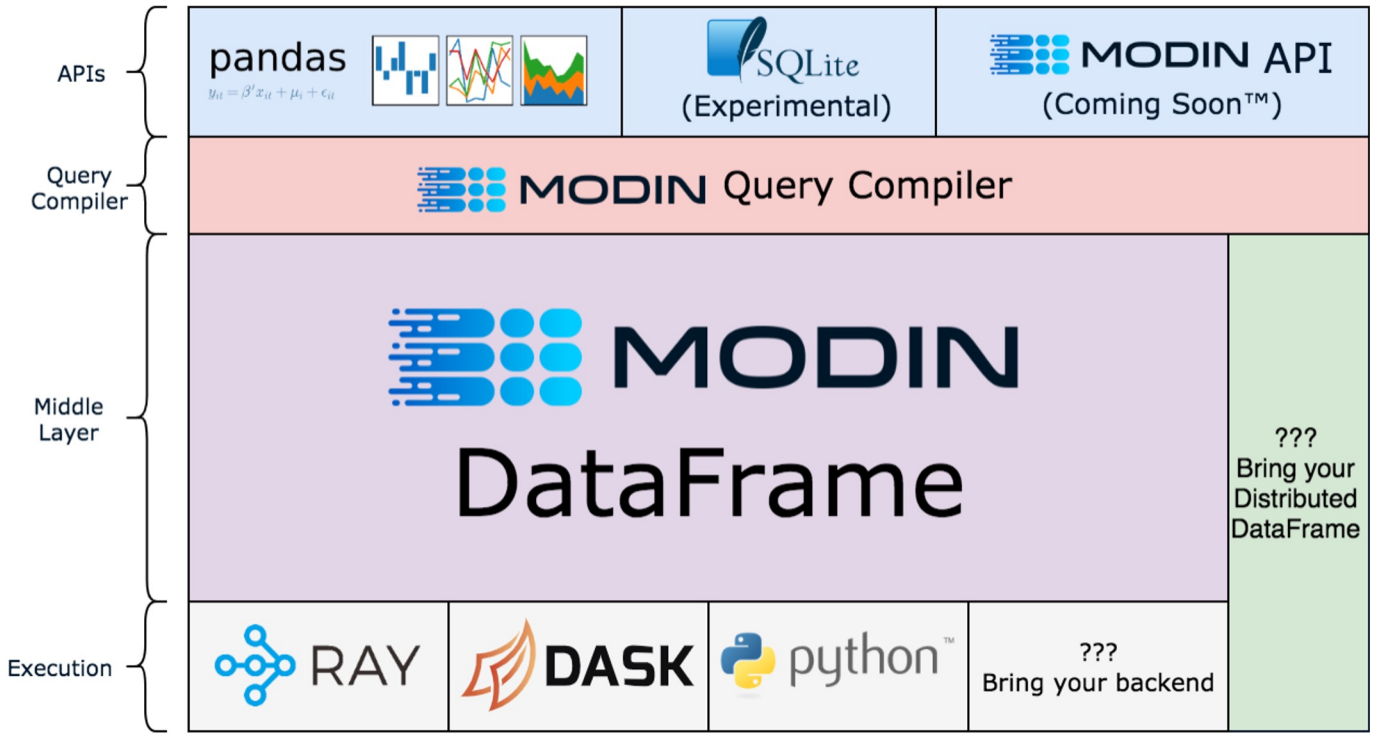
```
pip install modin
```

https://modin.readthedocs.io/en/latest/

Modin - Impact of the pandas API

- 5300+ Github stars
- 41k installs/month, >400k total (since July 2018 start)
- Reached overall trending on GitHub multiple times (top starred repos of the day)
- Some of the users:
  - Tesla
  - DoD
  - Oak Ridge National Lab
  - Splunk
  - NVIDIA
  - Intel

Modin Architecture



https://modin.readthedocs.io/en/latest/

Summary (1/2)

- ETL is used to bring data from operational data stores into a data warehouse
  - Many ways to organize tabular data warehouse, e.g., star and snowflake schemas
- Online Analytics Processing (OLAP) techniques let us analyze data in data warehouse
- Unstructured data is hard to store in a tabular format in a way that is amenable to standard techniques,
  e.g., finding pictures of cats
  - Resulting new paradigm: The Data Lake

Summary (2/2)

- Data Lake is enabled by two key ideas:
  - Distributed file storage, Distributed computation
- Distributed file storage involves replication of data
  - Better speed and reliability, but more costly
- Distributed computation made easier by map reduce
  - Apache Hadoop: Open-source implementation of distributed file storage and computation
  - Apache Spark: Typically faster and easier to use than Hadoop
- Modin: Accelerating local data exploration