MINING OF MASSIVE DATASETS

Zareen Alamgir

MINING OF MASSIVE DATASETS

COURSE OVERVIEW

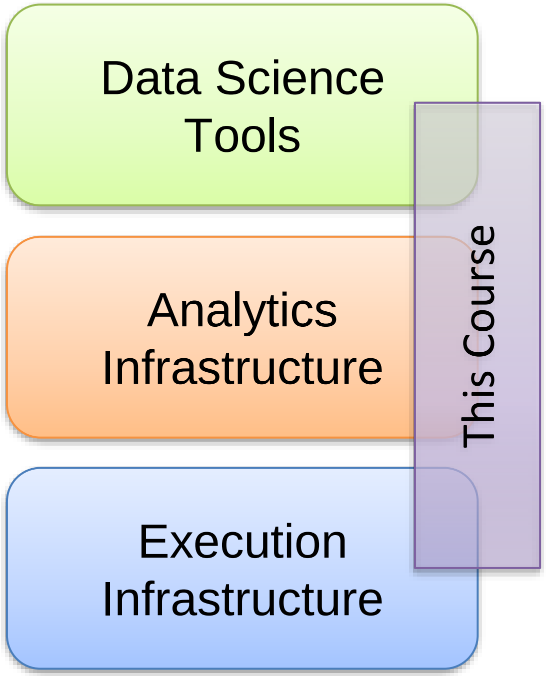
Course Information

- Instructor: Zareen Alamgir
- Email: zareen.alamgir@nu.edu.pk

- Course Information and updates will be posted on ***Google Classroom***
  - Tentative schedule
  - News and announcements
  - Lecture Slides
  - Assignments
  - Books and Reading Material

Course Content

- Introduction
- Infrastructure for Massive data
  - Map Reduce (very brief)
  - Hadoop, HDFS
  - Apache Spark
- Algorithms and Techniques (Tentative)
  - Clustering
  - Graphs - Link Analysis (Page Rank) and Inverted Index
  - Finding Similar items (Locality Sensitive Hashing)
  - Large-Scale Machine Learning (Decision trees)
  - Recommendation Systems (ALS)
  - Frequent Pattern Mining



This course focuses on algorithm design and "thinking at scale"

Textbooks and Readings

- TextBooks
  - **Mining of Massive Data Sets**, by Anand Rajaraman, Jure Leskovec and Jeff Ullman
  - Data Mining: Concepts and Techniques. By Jiawei Han and Micheline Kamber.

- Reference
  - **Learning Spark**, by Holden Karau, Andy Konwinski, Patrick Wendell and Matei Zaharia
  - Introduction to Data Mining. By P.-N. Tan, M. Steinbach and V. Kumar.
  - **Data-Intensive Text Processing with MapReduce**, by Jimmy Lin and Chris Dyer

- All textbooks are free to download
- We will also cover important research papers and tutorials

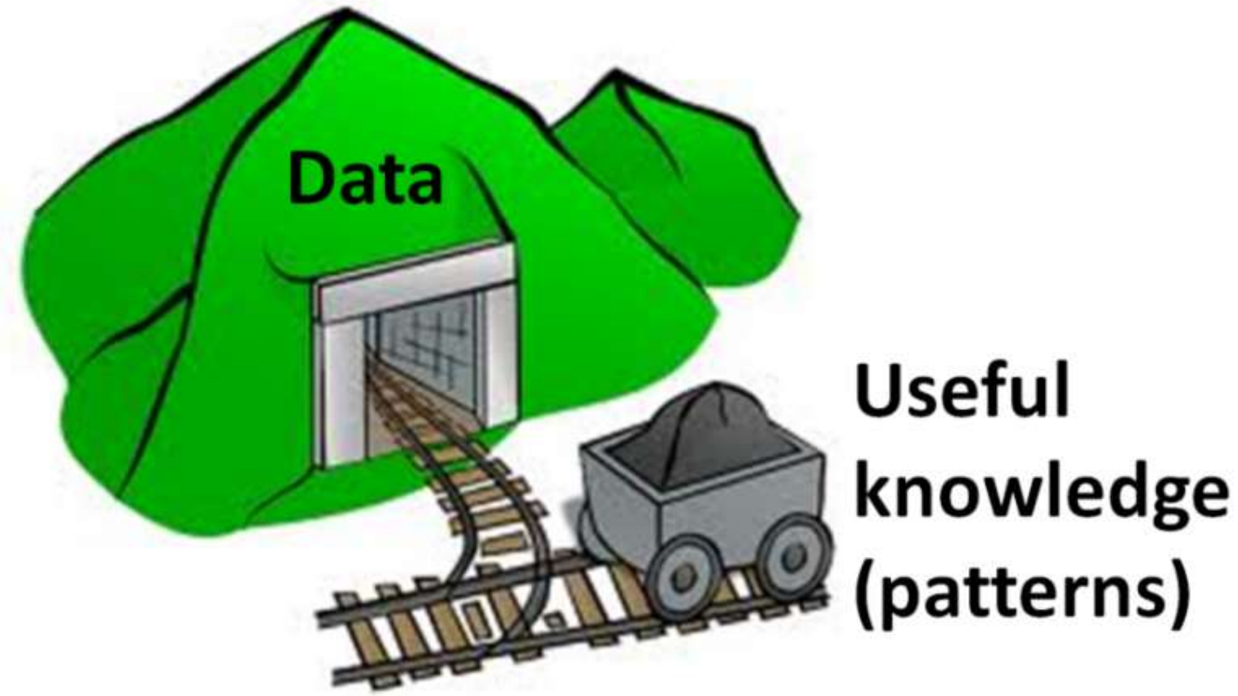
Pre-Requisites

The students should have good background in

- Programming and Data structures
- Database Systems (familiarity with SQL queries)

Tentative Grading Scheme

■ Two Midterms	30%
■ Quizzes	10%
– 5 quizzes or more	
■ Assignments/Project	10%
– Programming Assignments	
– Project/Presentation	
■ Final	50%



WHAT IS DATA MINING?

Knowledge discovery from data

http://data-mining.philippe-fournier-viger.com/introduction-data-mining/

Introduction

- Data is growing at a phenomenal rate
  - Web data, e-commerce
  - purchases at department/grocery stores
  - Bank/Credit Card transactions
  - scientific simulations

UNCOVER HIDDEN INFORMATION

DATA MINING

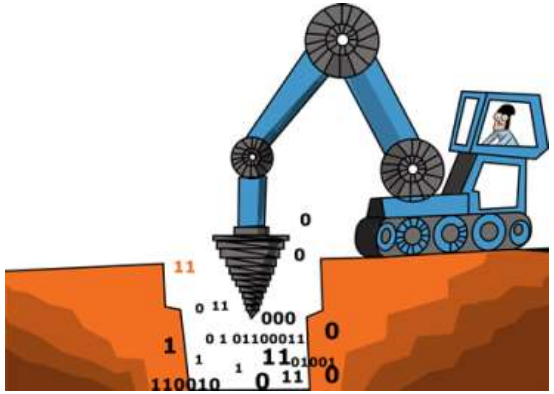
We are drowning in data but starving for knowledge!

Information “hidden” in the data is not readily evident

Human analysts take weeks to discover useful information

https://whatsthebigdata.com

What is Data Mining



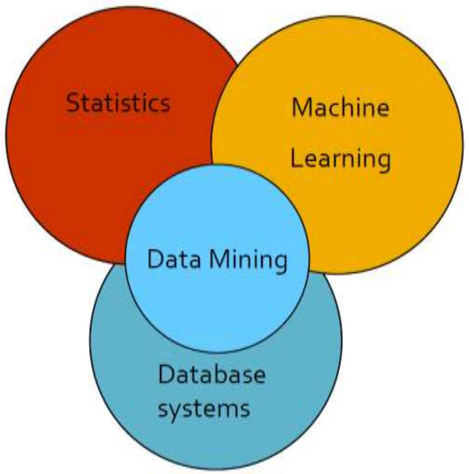
- Data mining (knowledge discovery from data)
  - Extraction of interesting (non-trivial, implicit, previously unknown and potentially useful) patterns or knowledge from huge amount of data

  - Exploration & analysis, by automatic or semi-automatic means, of large quantities of data in order to discover meaningful patterns

http://www.cs.science.cmu.ac.th

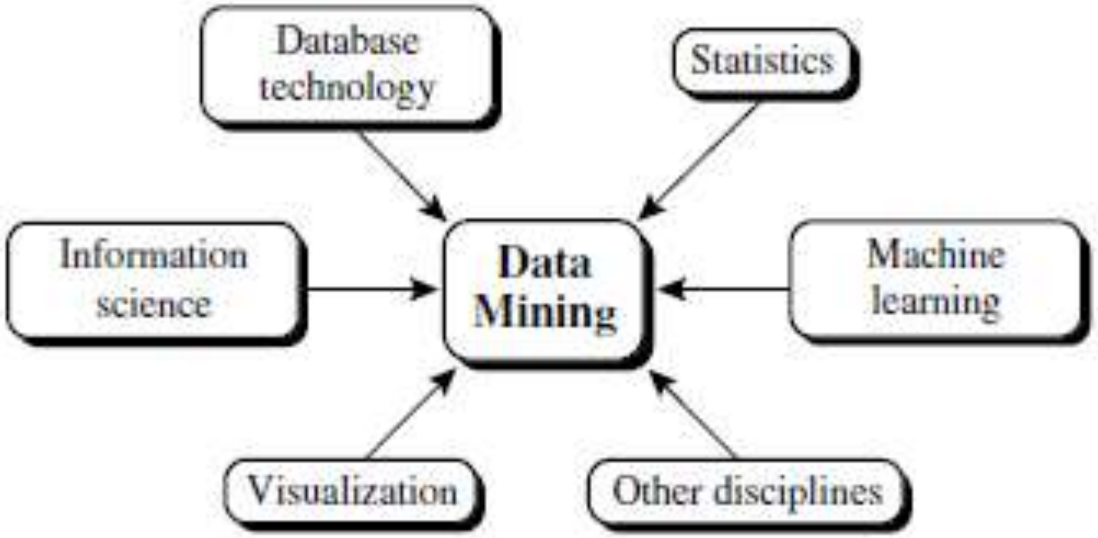
Data Mining and related Disciplines

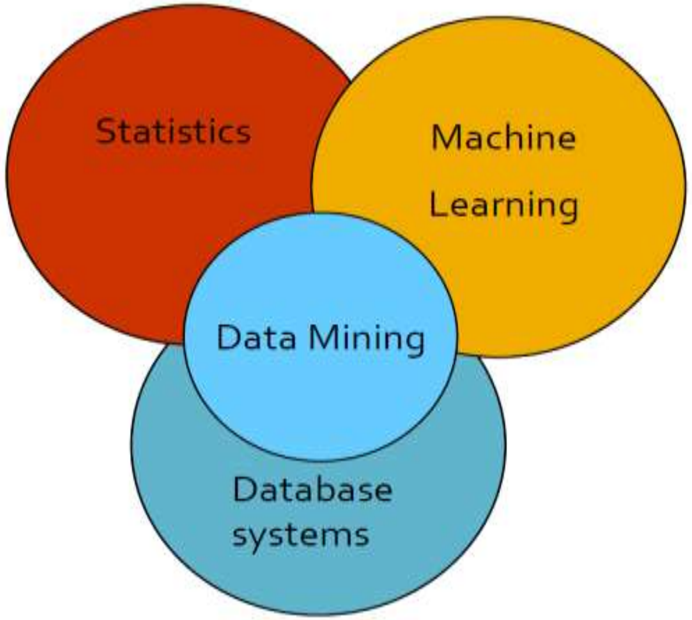
- Data mining overlaps with:
  - **Databases:** Large-scale data, simple queries
  - **Machine learning:** Small data, Complex models
  - **CS Theory:** (Randomized) Algorithms
- **Different cultures:**
  - To a DB person, data mining is an extreme form of **analytic processing** – queries that examine large amounts of data
    - Result is the query answer
  - To a ML person, data-mining is the **inference of models**
    - Result is the parameters of the model



Data Mining and related Disciplines

- Emphasis is on
  - **scalability** of number of features and instances (massive data)
  - stress on **algorithms and architectures**
    - whereas foundations of methods provided by statistics and machine learning
  - automation for handling large, complex and heterogeneous data





Database vs Data Mining

Database	Data Mining
• Find all credit applicants with last name of Smith. • Identify customers who have purchased more than $10,000 in the last month. • Find all customers who have purchased milk	• Find all credit applicants who are poor credit risks. *(classification)* • Identify customers with similar buying habits. *(Clustering)* • Find all items which are frequently purchased with milk. *(association rules)*

Database Processing vs. Data Mining

Query

- Well defined
- SQL

Query

- Poorly defined
- No precise query language

Output

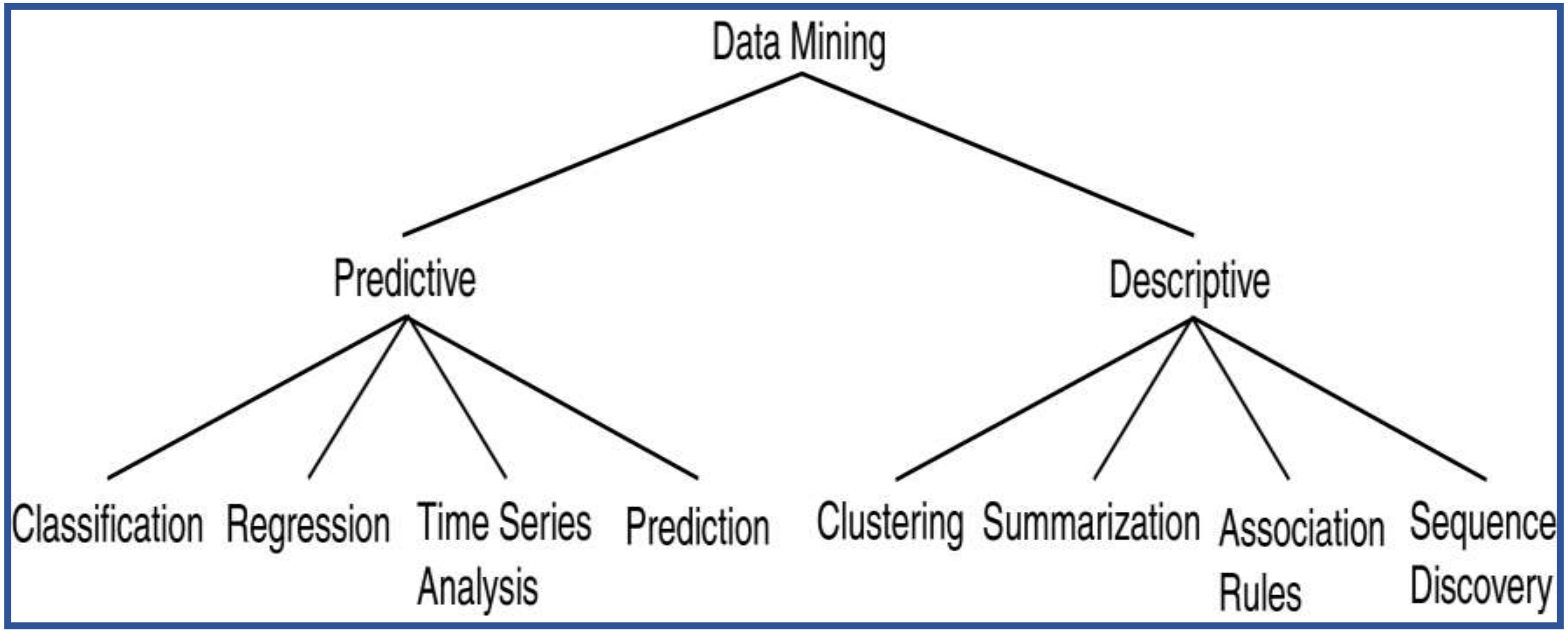
- Precise
- Subset of database

Output

- Fuzzy
- Not a subset of database

Data Mining Models and Tasks

- **Descriptive data mining**:
  - Describe general properties
- **Predictive data mining:**
  - Infer on available data



Data Mining ≈ Predictive Analytics ≈ Data Science ≈ Machine Learning ≈ Data-Centric AI

What this course is about ?

Mining of massive datasets

What this course is about ?

Extraction of actionable information from (usually) very large datasets

It's not all about machine learning
But most of it is!

- Emphasis is on algorithms that **scale**
  - Parallelization often essential

DISTRIBUTED COMPUTING FOR DATA MINING

What is Massive/Big Data?

Too big: petabyte-scale collections or lots of big data sets

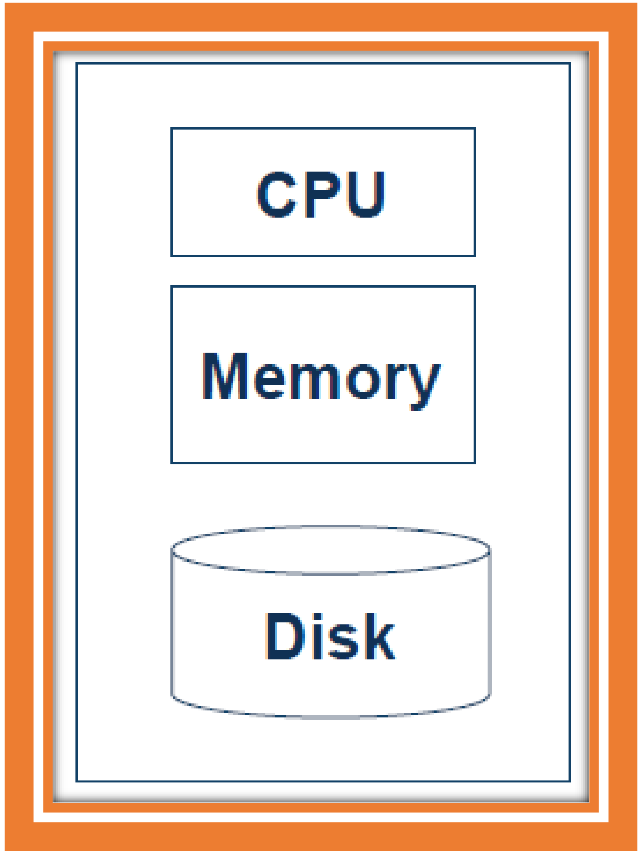
Too hard: does not fit neatly in an existing tool

- *Data sets that need to be cleaned, processed and integrated*
- *E.g., Twitter, news, customer transactions*

Too fast: needs to be processed quickly

Single-node Architecture

Data Analysis → Data Mining → Machine Learning



Motivation: Google Example

20+ billion web pages x 20KB = 400+ TB

1 computer reads 30-35 MB/sec from disk

- *~4 months to read the web*

~1,000 hard drives to store the web

Takes even more to **do** something useful with the data!

A standard architecture for such problems is emerging

- *Cluster of commodity Linux nodes*
- *Commodity network (ethernet) to connect them*

How to handle massive data ?

Platforms for Large-scale Data Mining

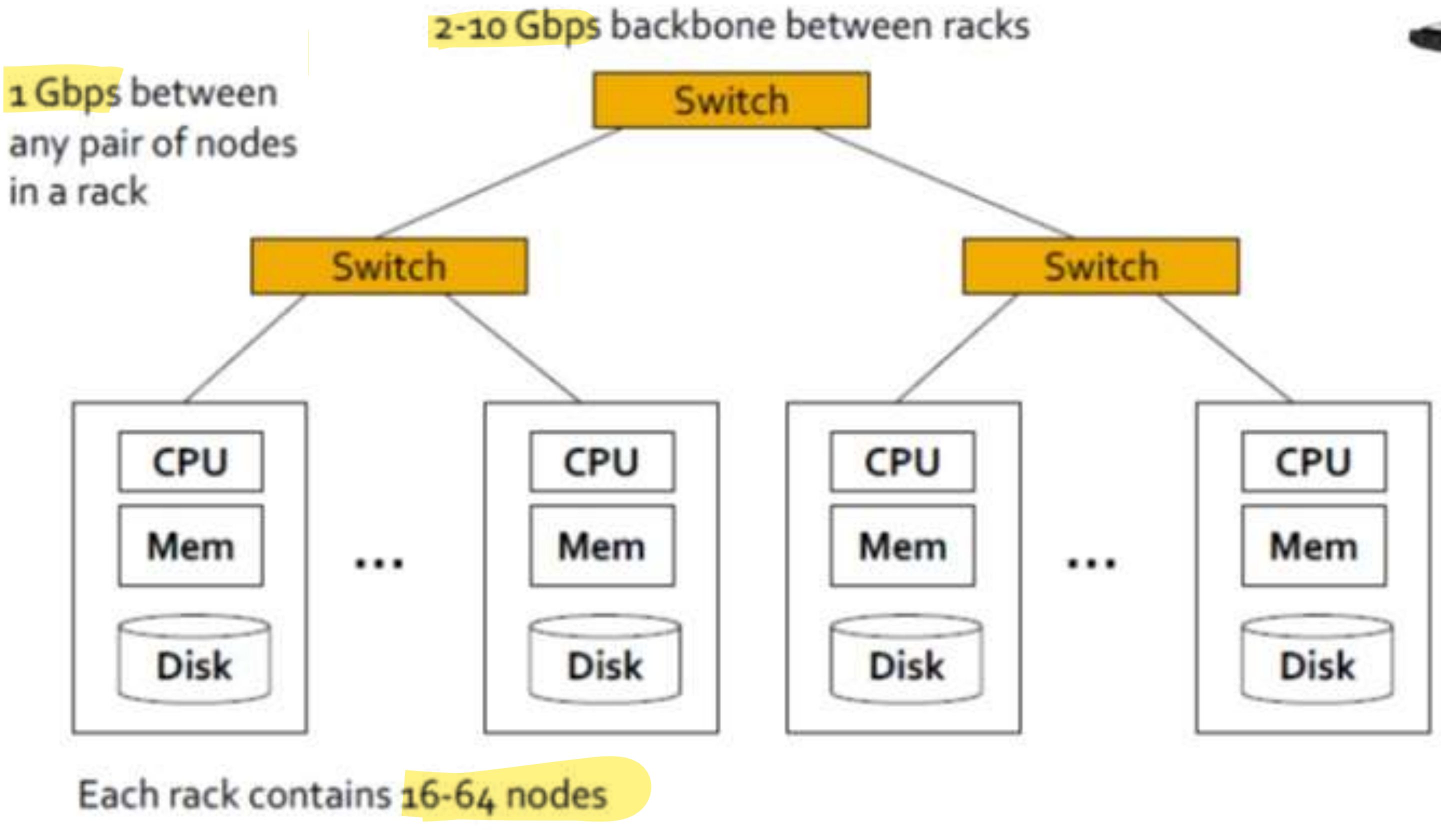
Distributed Infrastructure

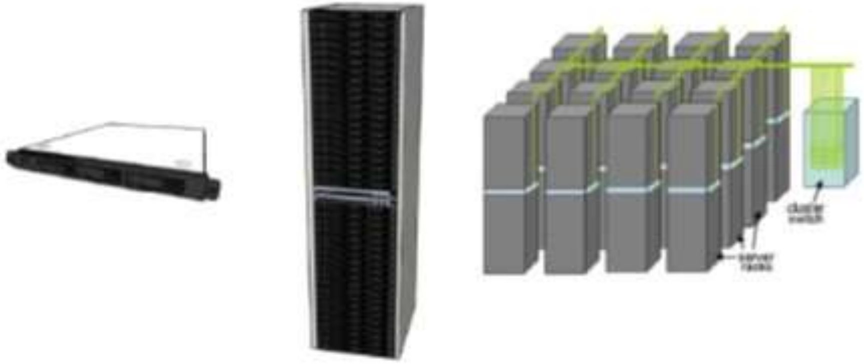
- *HADOOP*
- *HDFS*

Programming Models

- *Map Reduce*
  - pioneered by Google
  - popularized by Yahoo
- *SPARK*

Cluster Architecture



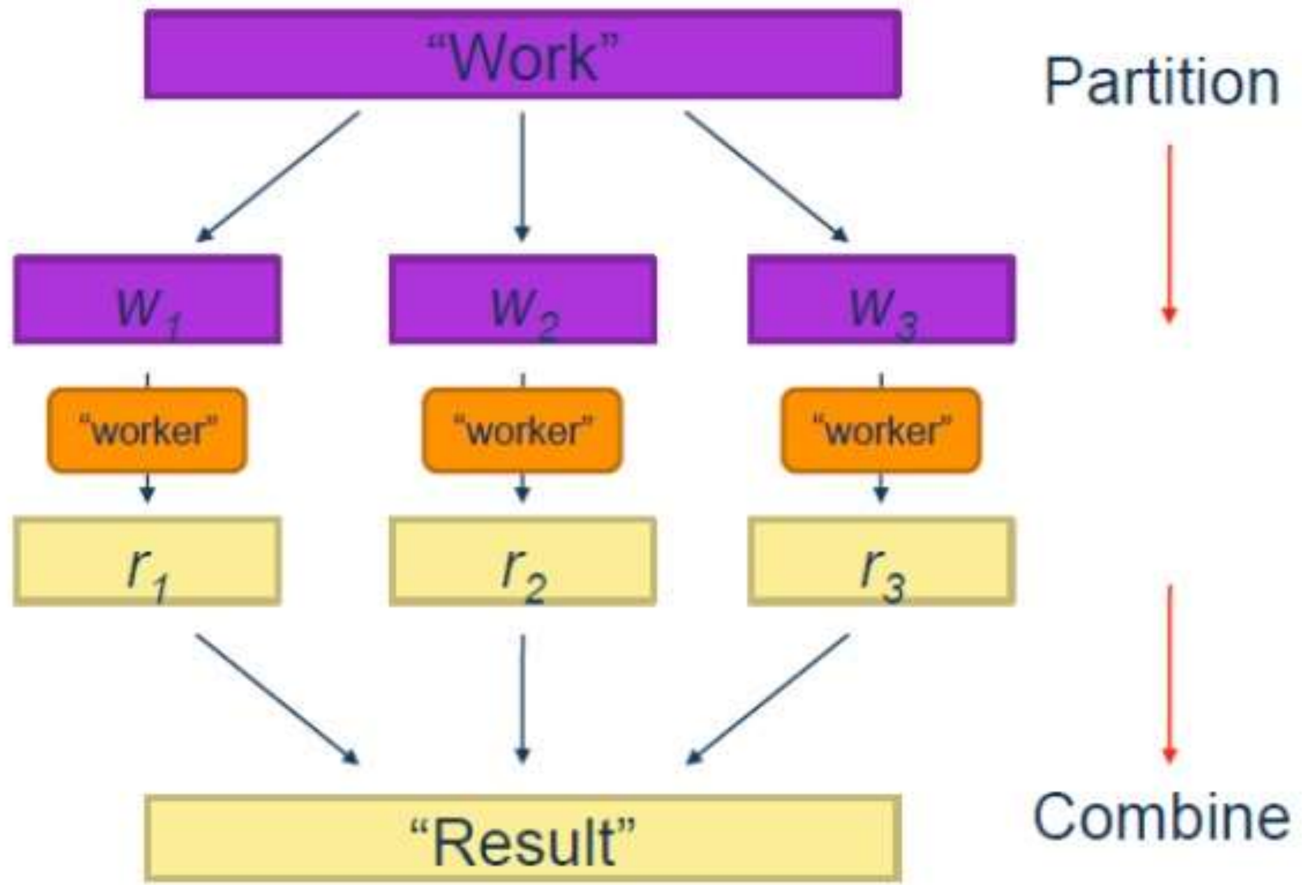


In 2011 it was guestimated that Google had 1M machines, http://bit.ly/Shh0RC

Tackling Big Data

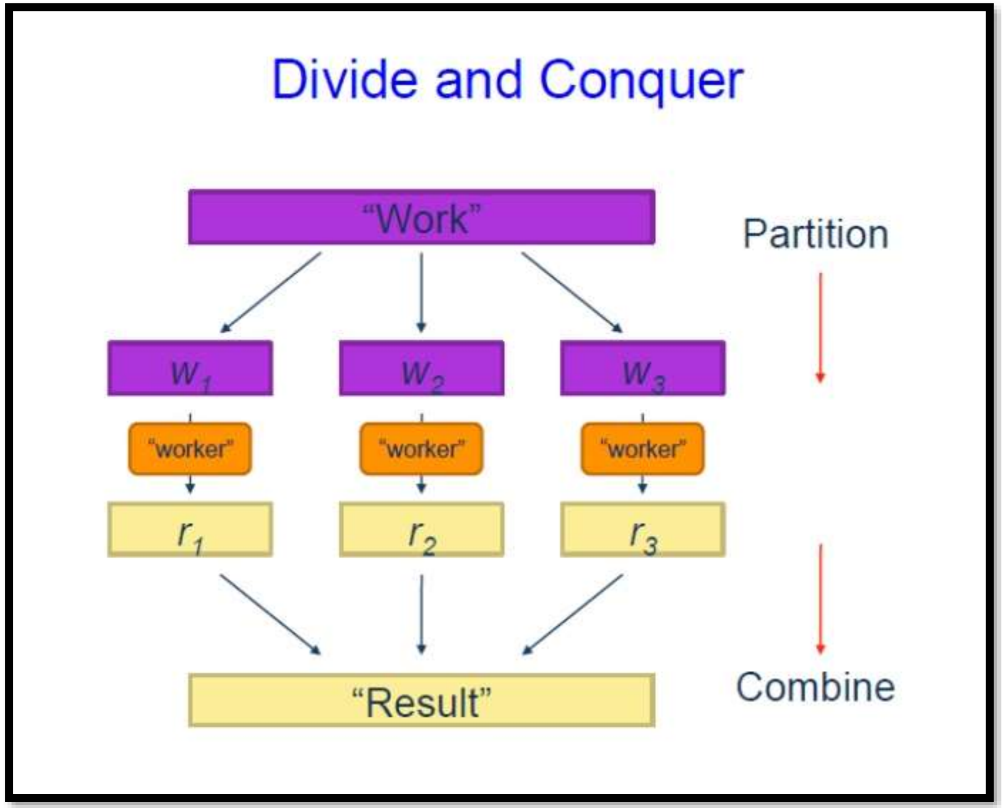
Source: Google

Divide and Conquer



Parallelization Challenges

- How do we assign work units to workers?
- What if we have more work units than workers?
- What if workers need to share partial results?
- How do we aggregate partial results?
- How do we know all the workers have finished?
- What if workers die?
- What is the common theme of all of these problems



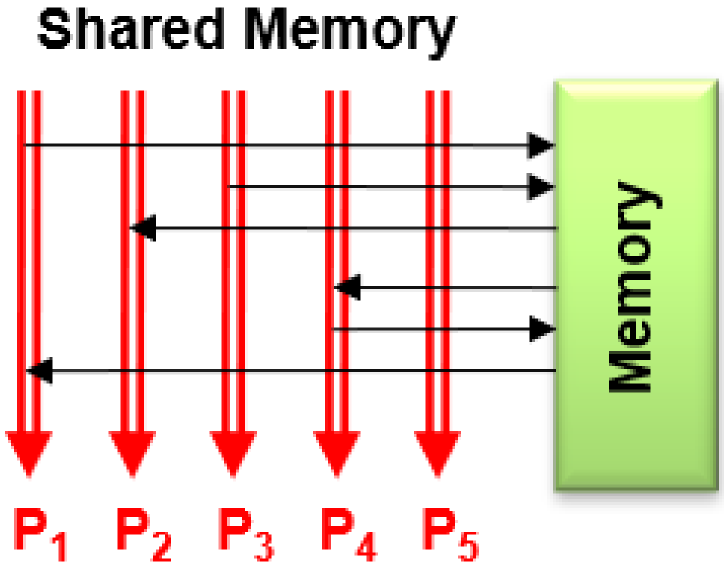
Common Theme?

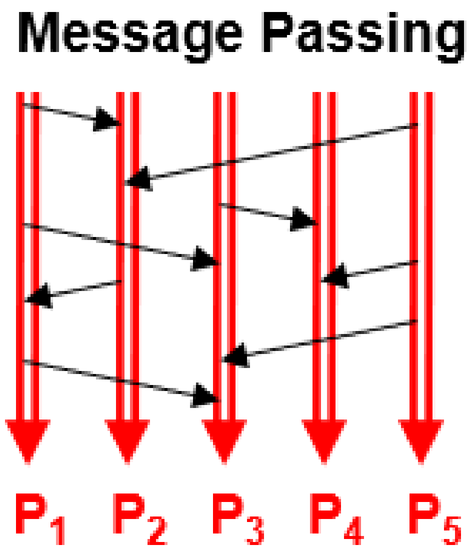
- Parallelization problems arise from:
  - **Communication between workers** (e.g., to exchange state)
  - **Access to shared resources** (e.g., data)
- Thus, we need a synchronization mechanism

Semaphores (lock, unlock)
Conditional variables (wait, notify, broadcast)
Barriers
Still, lots of problems:
Deadlock, livelock, race conditions…
Dining philosophers, sleeping barbers, cigarette smokers…

Current Tools

- What if workers need to share partial results?
- **Programming models**
  - Shared memory (pthreads)
  - Message passing (MPI)





When Theory Meets Practices

Concurrency is already difficult to reason about…

Now throw in:

The scale of clusters and (multiple) datacenters
The presence of hardware failures and software bugs
The presence of multiple interacting services

The reality:

Lots of one-off solutions, custom code
Write you own dedicated library, then program with it
Burden on the programmer to explicitly manage everything

Bottom line: it's hard!

Big Ideas: Abstract System-Level Details

It's all about the right level of abstraction

MapReduce isolates developers from System level details

Separating the What from the How !!!!

- Programmer defines **what** computations are to be performed
- **MapReduce execution framework takes care of how the computations are carried out**

Big Ideas: Scale Out vs. Scale Up

Scale up

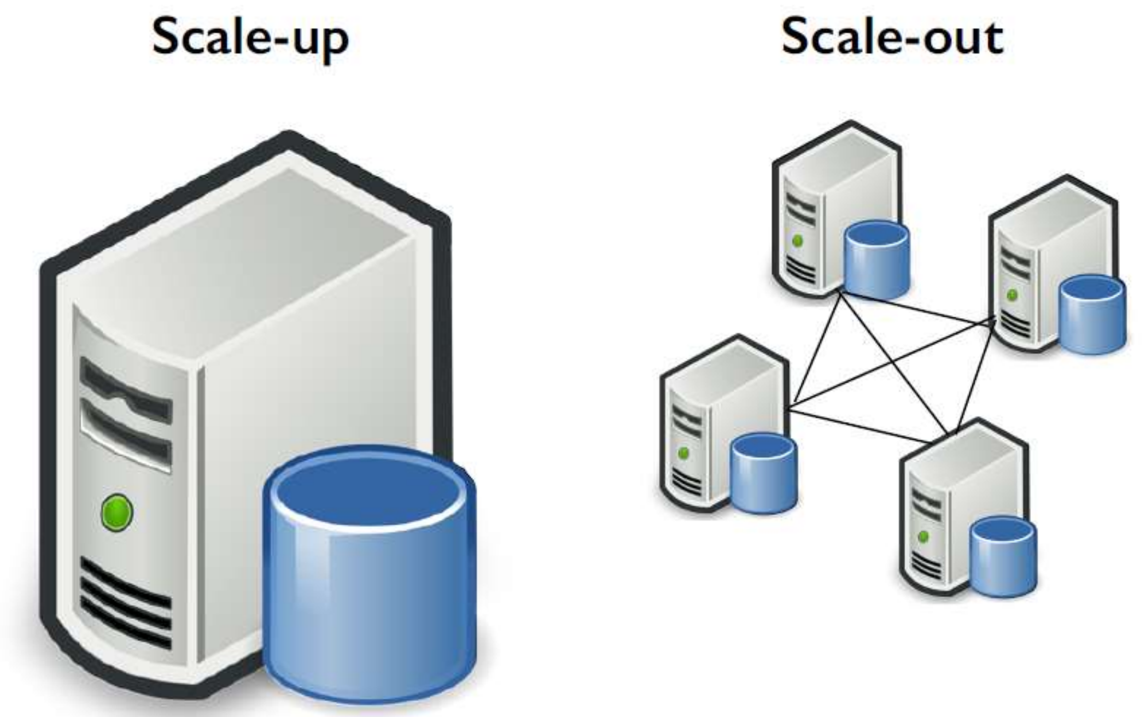
small number of high-end servers

- Symmetric multi-processing (SMP) machines, large shared memory
- Not cost-effective – cost of machines does not scale linearly; and no single SMP machine is big enough

Scale out

Large number of commodity low-end servers is more effective for data-intensive applications

8 128-core machines vs. 128 8-core machines



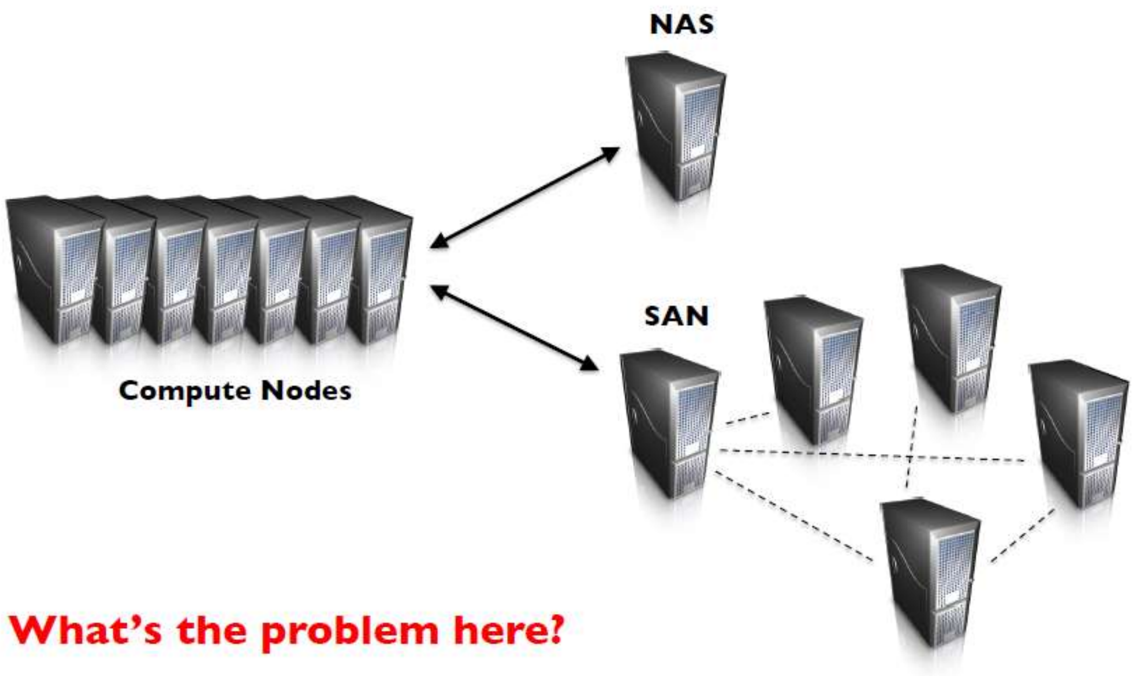
Big Ideas: Failures are Common

- Suppose a cluster is built using machines with a *mean-time between failures (MTBF) of 1000 days*
  - **For a 10,000 server cluster, there are on average 10 failures per day!**
- MapReduce and Spark implementation cope with failures
  - Automatic task restarts

Big Ideas: Move Processing to Data

- Supercomputers often have processing nodes and storage nodes
  - Computationally expensive tasks
  - High-capacity interconnect to move data around
  - Data movement leads to a bottleneck in the network!

How do we get data to the workers?



What's the problem here?

Why does this make sense for compute-intensive tasks?

What's the issue for data-intensive tasks?

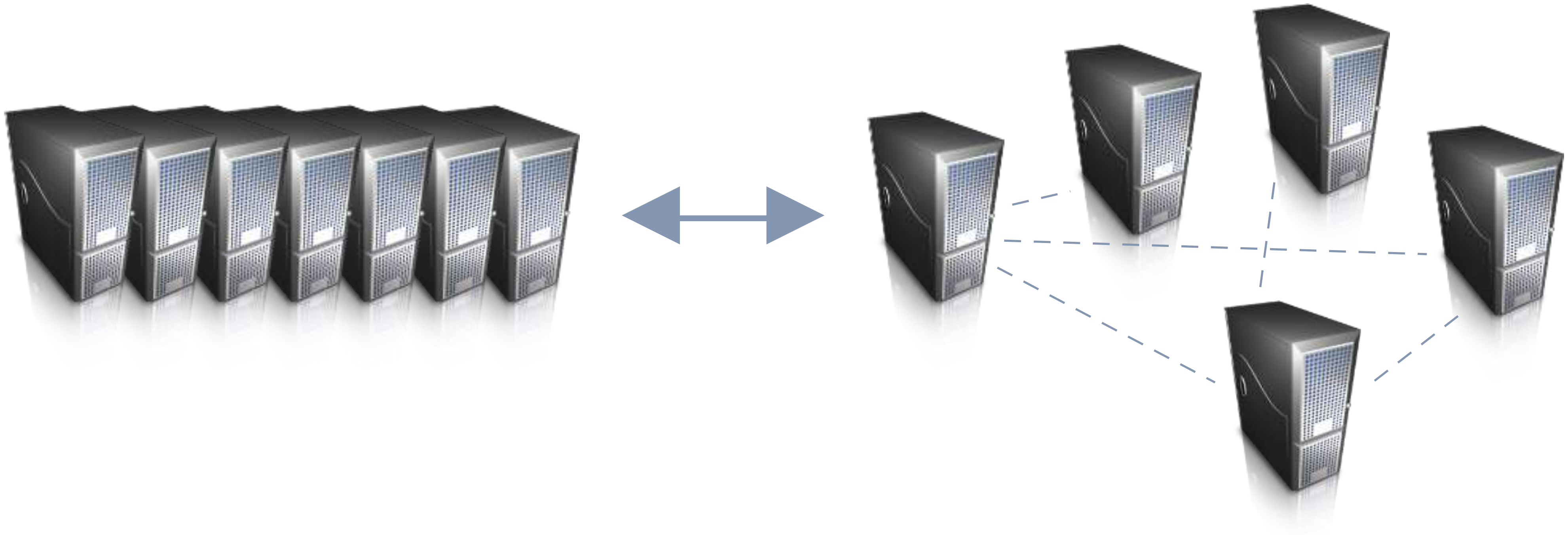
Many data-intensive applications are not very processor-demanding

What's the solution?

Don't move data to workers… move workers to the data!

Key idea: co-locate storage and compute

Start up worker on nodes that hold the data



What's the solution?

Don't move data to workers… move workers to the data!

Key idea: co-locate storage and compute

Start up worker on nodes that hold the data

We need a distributed file system for managing this

GFS (Google File System) for Google's MapReduce

HDFS (Hadoop Distributed File System) for Hadoop