WELCOME TO CLOUD COMUTING!

Dr. Gokop Goteng (Module Organiser)

g.l.goteng@qmul.ac.uk

Dr. Atm Shafiul Alam

a.alam@qmul.ac.uk

Queen Mary University of London

School of Electronic Engineering and Computer Science

Course summary

- Week 1
  - Introduction to cloud computing
  - AWS EC2
  - Cloud Scalability
  - Cloud Security and Trust
- Week 2
  - GPU Architecture
  - GPU CUDA C Programming
- Week 3
  - Big Data Processing
  - Map-Reduce
  - Apache Hadoop
- Week 4
  - Beyond Map-Reduce
  - Content Delivery Network
  - NoSQL
  - Graphs in the Cloud

Overview of Week 1

- Lecture 1: Principles of Map Reduce
- Lecture 2: Apache Hadoop
- Lecture 3: Map Reduce Programming
- Lecture 4: Reliability and Performance
- Lecture 5: Tutorial: Recap, Revision & Question Time

Resources

- Text Books
  - Cloud Computing: Concepts, Technology & Architecture, Authors: Erl, T., Puttini, R., & Mahmood, Z.
  - CUDA Handbook: A Comprehensive Guide to GPU Programming, 1/E, Author: Wilt
  - Hadoop: The Definitive Guide, 4th Edition, Author: Tom White Ed. O' Reilly
- Standard documents [regulatory and government organisation]
- Research articles/technical papers
- **Regularly** check QM+ for updates & additional material

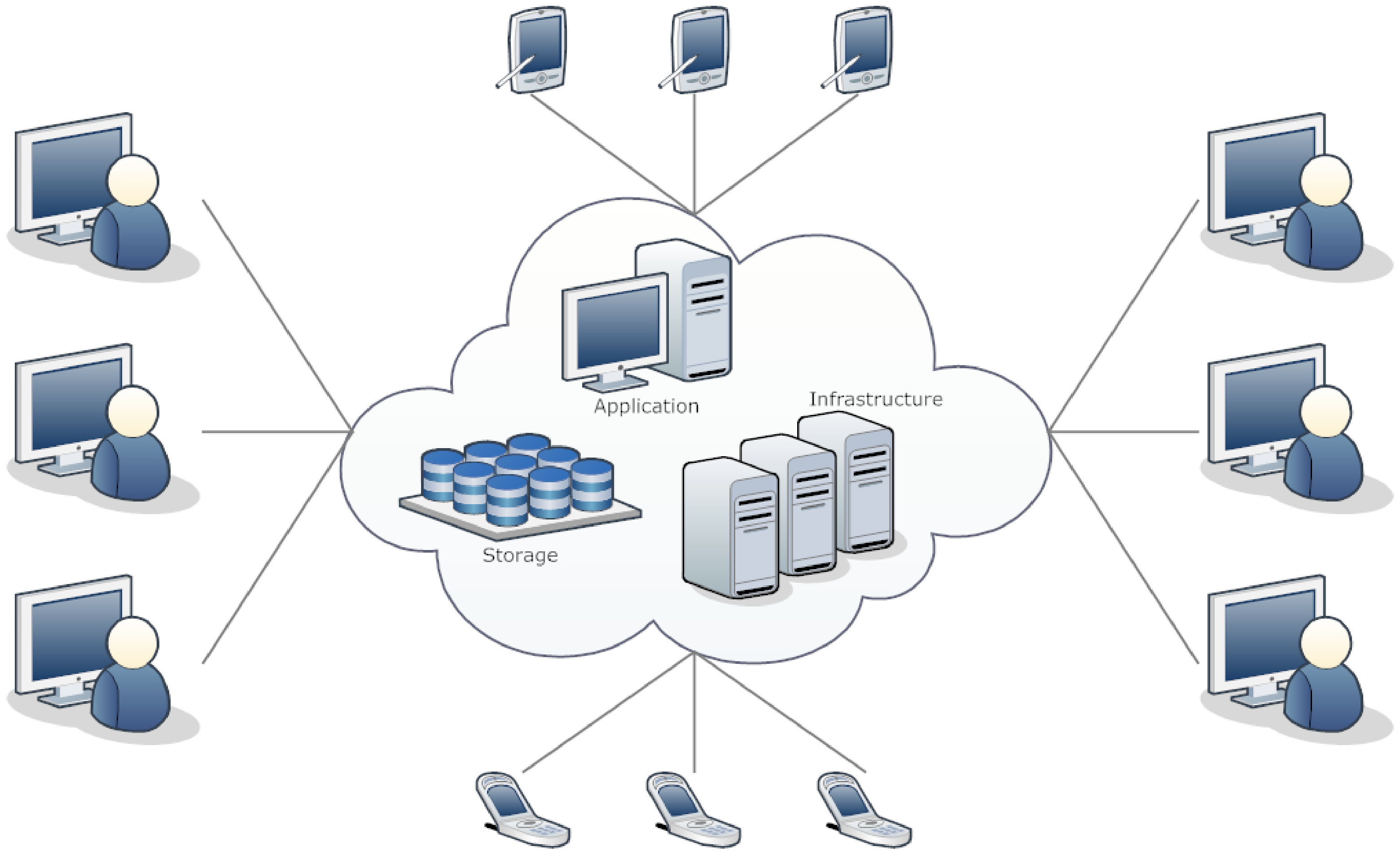
What is cloud computing?

- "***Cloud computing*** *is the* ***on-demand availability of computer system resources*** *especially data storage and computing power, without direct active management by the user, often over the Internet.* Cloud computing relies on sharing of resources to achieve coherence and economies of scale."(1)

- *"Simply put, cloud computing is the* ***delivery of computing services*** *– including servers, storage, databases, networking, software, analytics and intelligence –* ***over the Internet*** *("the cloud") to offer faster innovation, flexible resources and economies of scale. Typically, you only pay for cloud services you use, helping you lower your operating costs, run your infrastructure more efficiently and scale as your business needs change."* (2)

(1)https://en.wikipedia.org/wiki/Cloud_computing
(2) https://azure.microsoft.com/en-gb/overview/what-is-cloud-computing/

What is cloud computing?



What is cloud computing?

- Computing as a utility
  - Utility services, e.g., water, electricity, gas, etc.
  - Consumers pay based on their usage.

- Cloud Computing characteristics:
  - Illusion of infinite resources
  - No up-front cost
  - Fine-grained billing (e.g., hourly)

- In simple terms: “Cloud computing is **the delivery of on-demand computing services** -- from **applications to storage and processing power** -- typically over the internet and on a pay-as-you-go basis.”

What does a cloud (physically) look like?

You can 'hire' one or more of these computers to run your code…i.e. there is no need to buy one permanently!

What is cloud computing?

- Cloud computing is used for many thin
  - Analysing data
  - Hosting w

We're gonna learn about "Big Data" applications

- A mix of several concepts!
  - Datacentres
  - Virtualisation
  - Applications
  - Distributed systems

Who is generating Big Data?

Social

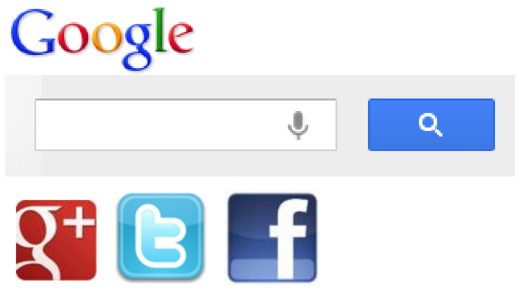
User Tracking & Engagement

Homeland Security

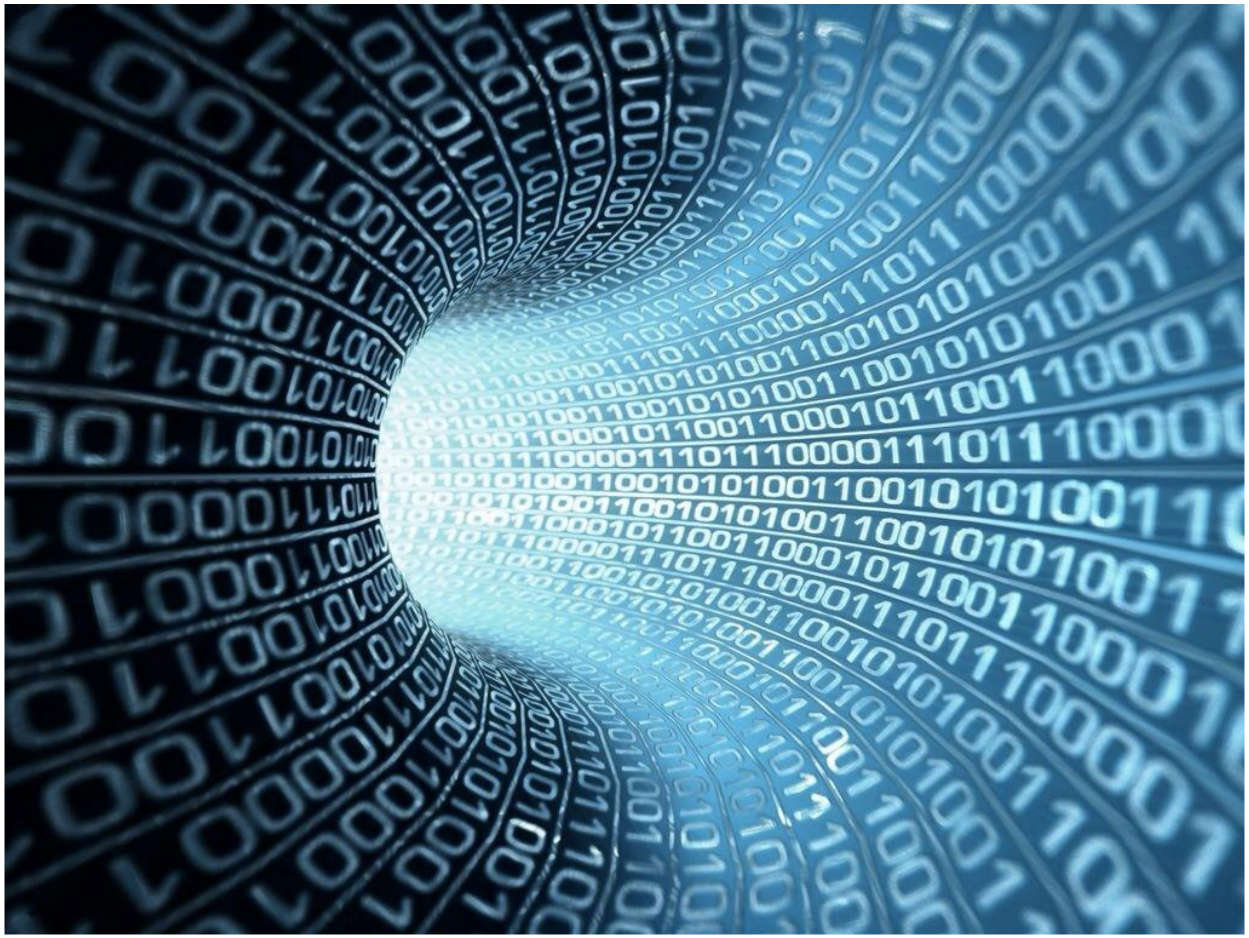
eCommerce

Financial Services

Real Time Search



This is a lot of data...



What does “Big Data” mean?

(1) Collecting large amounts of data

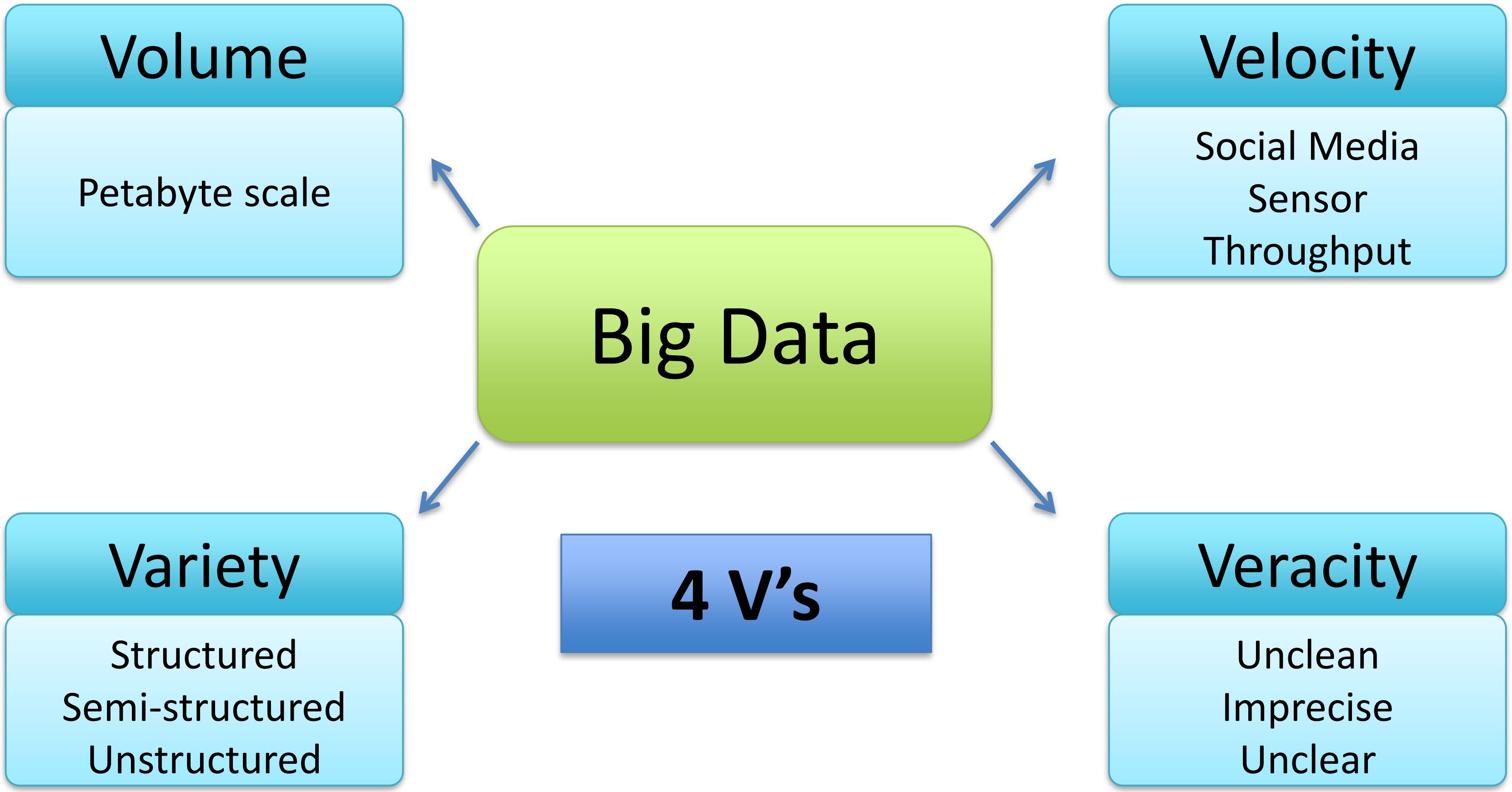
Via computers, sensors, people, events, etc.

(2) Doing something beneficial with it

Making decisions, confirming hypotheses, gaining insights, predicting future …

“Data Science” = Going to (1) to (2)

What are Key Features of Big Data?



What is a distributed system?

- A distributed system consists of
  - Hardware
  - Software
  - Data

Any system that works across multiple computers

d at networked computers, which communicate and coordinate

…their actions using protocols and passing messages

What makes distributed systems interesting?

- Concurrency
  - Multiple components working on different parts of the same task at the same time
- No global clock for time system
  - No guaranteed sequence of events
- No guaranteed reliability
  - Computer and networks fail!
- Price/performance ratio is much better

Distributed processing is non-trivial

- How to assign tasks to different workers in an efficient way?
- What happens if tasks fail?
- How do workers exchange results?
- How to synchronize distributed tasks allocated to different workers?

Image courtesy of Master isolated images at FreeDigitalPhotos.net

Many examples

- …anything that involves multiple computers coordinating
  - Internet (routers, computers around the world)
  - Domain name service (DNS resolvers, clients)
  - Web systems (web server, client browser)
  - Peer-to-peer networks (e.g., BitTorrent)
  - Big Data processing applications (e.g., Hadoop)

INTRODUCTION TO MAP/REDUCE

CLOUD COMPUTING

Dr. Atm Shafiul Alam

a.alam@qmul.ac.uk

Queen Mary University of London

School of Electronic Engineering and Computer Science

Thanks to Dr. Gareth Tyson and Dr. Félix Cuadrado who created the original version of these slides

Contents for today

- **Introduction to parallelism**
- Benefits to parallelism
- The Map/Reduce Programming Model

Parallel computing

- The use of a number of processors, working together, to perform a calculation or solve a problem.
- The calculation will be divided into tasks, sent to different processors.
  - **Processor coordination** will be required
- Processors can be different **cores** in the same machine, and/or different **machines** linked by a network

Parallel computing is hard (but fun?)

- Parallel Computing is generally very hard, because:
  - Many algorithms are **hard to divide** into subtasks (or cannot be divided at all)
  - The subtasks might **use results from each other**, so coordinating the different tasks might be difficult
- Some problem areas are much easier than others to parallelize
  - Imagine organising a party with 10 friends…

Parallel computing is hard (but fun?)

- We can rank tasks based on how easy they are to parallelize (i.e., distribute across multiple machines)

- 2 people arranging to go for coffee?
- 2 people organizing a party for 100 people?
- 2 people trying to sort a list of 50,000 words

Easy parallelization

- **Image processing**: sharpening an image

1. We have a big photograph
2. We divide it into tiles (square patches), and sharpen each tile (separately)
3. Then we adjust the pixels at the edges so that they match up

This works because edge pixels are a small part of the total, and changing them does not affect the other pixels

Not so easy parallelization

- **Path search**: Get best route from Beijing to Shanghai
- How to divide the task in lesser ones? Find intermediate points?
- How to select these intermediate points?
- Need to communicate among processes, what are the intermediate results?
- How can we guarantee that the solution is the best one?

Contents

- Introduction to parallelism
- **Benefits to parallelism**
- The Map/Reduce Programming Model

Why Parallelize?

- Solve larger/more complex problems:
  - There are many problems we cannot solve by running them on a single processor/machine.
- They are:
  - Too large and/or complex (do not fit in one machine)
  - Take tooooooo long

- Parallel computing can provide faster results, and it can even be cheaper $

Queen Mary University of London

Sequential Program Execution

- Basic model based on Von Neumann architecture from the 1940s
- One **instruction** is fetched, decoded and executed at a time (in sequence)
- As processor speed increases, more instructions can be executed in the same time

Single processor limitations

- We have roughly reached the practical limitations in the amount of computing power a single processor can have
  - We cannot make processors much faster, or much bigger
- According to Moore’s law, the number of transistors per chip will continue to increase
- But this means now we have chips containing an increasing amount of processors
  - Multicore chips

SOME EXAMPLE APPLICATIONS…

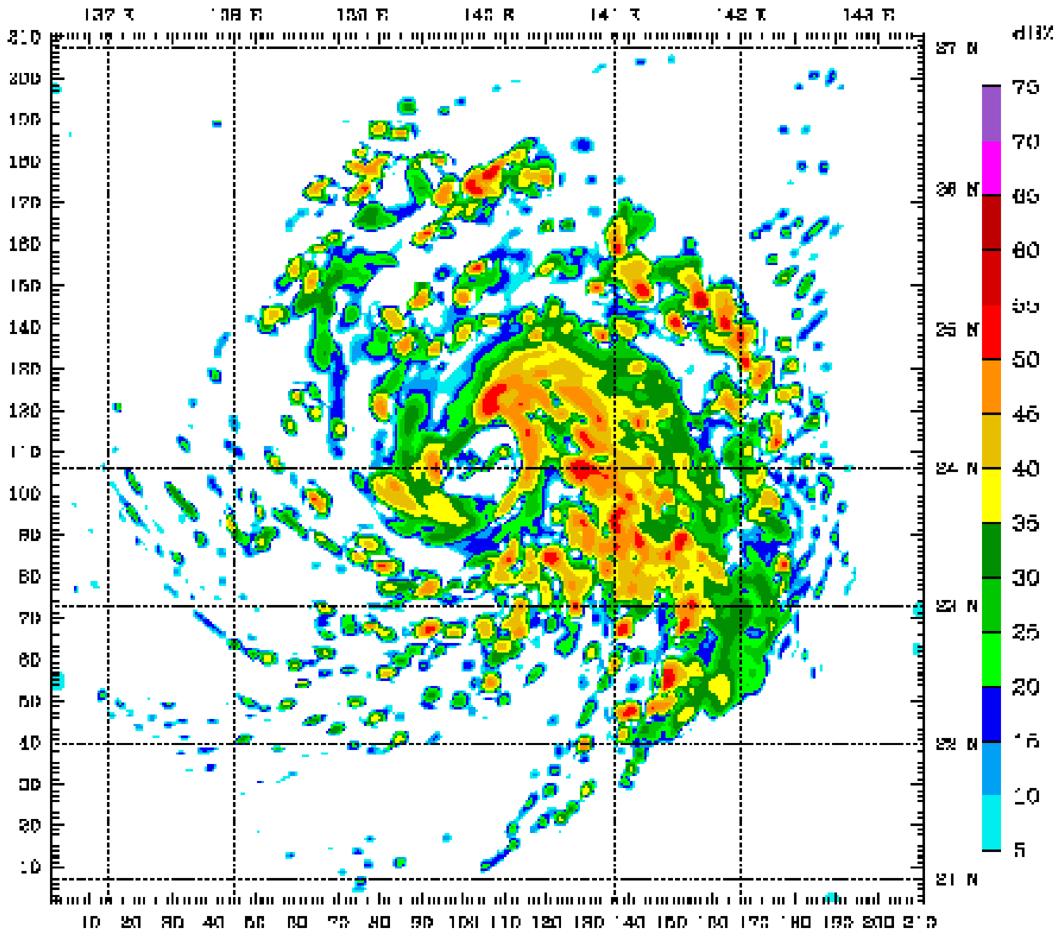
Applications of parallelism: Simulation

We often need to **simulate** physical events:

- What happens when lightning strikes a plane?
- This is important with modern planes, because they are made of plastic, and lightning goes through them (whereas it goes round metal planes)
- Planes are too expensive to conduct a large number of tests with them, but we have to know that they are safe to fly
- So, simulation is a necessity: but it is **computationally difficult**

There are a lot of similar tasks (e.g. nuclear safety) where direct testing is impossible or difficult

Applications of parallelism: Prediction



Applications of parallelism: Prediction

- We would like to predict real-world events (such as the weather)
- This often involves very long and data intensive calculations, just like a simulation does
- Challenging both for algorithm design and implementation

Applications: Data analysis

- Marketers analyze large amounts of data from Twitter in order to find out how their products are doing
- Google analyzes lots of web pages in order to support google search
- The LHC produces huge amounts of data, which need analysis
- Biologists read large quantities of DNA, and they have to work out what it means
- All these applications are computationally demanding, and they also use large amounts of data ("Big Data")

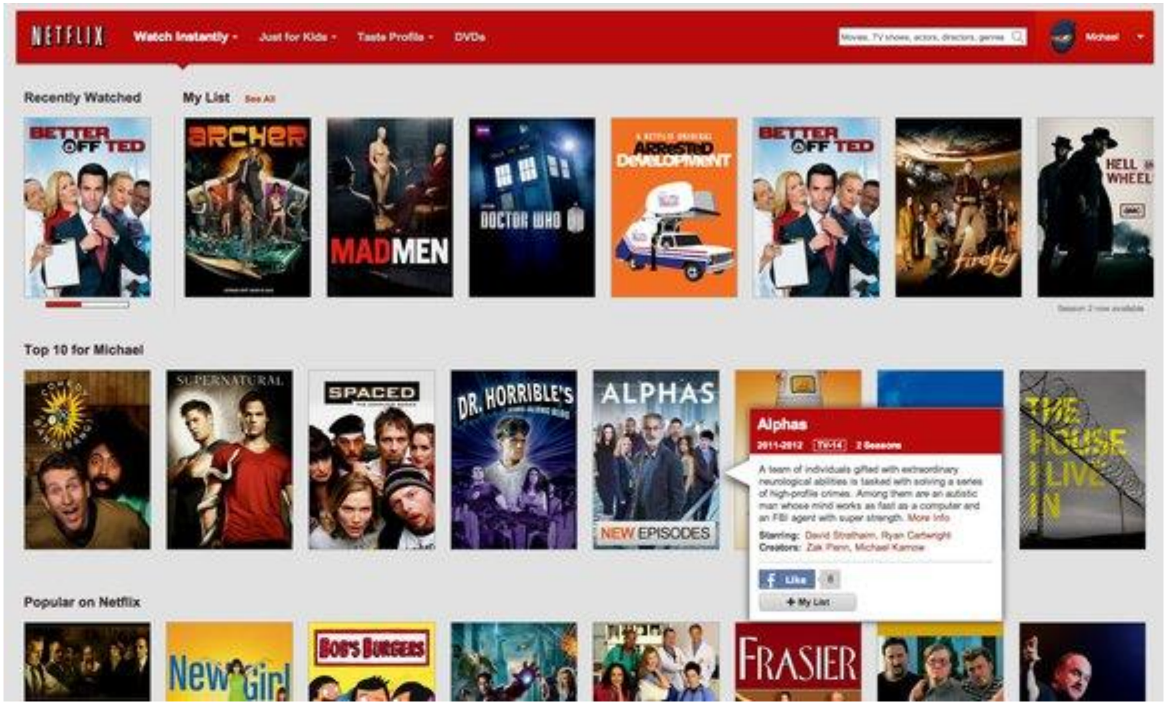
Data analysis (*cont.*)

- Take Netflix and its approach to applying data analysis for maximizing the effectiveness of their video catalog.
- Traditionally, there is a limited subset of movie genres (let's say 100).
- Netflix manages more than 70,000 genres and combines them with collected data about the millions of users that actively consume content through the service.

- With that information, they can provide recommendations considerably more effective than the previous standard
- http://www.theatlantic.com/technology/archive/2014/01/how-netflix-reverse-engineered-hollywood/282679/

Data analysis (*cont.*)

- Good data analysis == money



Alphas
2011-2012 TV-14 2 Seasons
A team of individuals gifted with extraordinary neurological abilities is tasked with solving a series of high-profile crimes. Among them are an autistic man whose mind works as fast as a computer and an FBI agent with super strength. More Info
Starring: David Strathairn, Ryan Cartwright
Creators: Zak Penn, Michael Karnow

Language, Systems, Platforms

- Spreadsheets
  - Surprisingly versatile and powerful for data analysis tasks, but not truly big data
- Programming languages with big-data support
  - R Language – powerful statistical features
  - Python – general-purpose language with R-like add-ons (Pandas, SciPy, scikit-learn)

Language, Systems, Platforms

- Relational Database Management Systems
  - Also called RDBMS, SQL Systems
  - Long-standing solution for reliability, efficiency, powerful query processing
  - Works for all but truly extreme data sizes, or highly unstructured data
- “NoSQL” Systems
  - Distributed/scalable processing, unstructured data
  - Key-value row stores (e.g., Cassandra, Dynamo)
  - Document databases (e.g., MongoDB, CouchDB)
  - Graph databases (e.g., Neo4J, Giraph)

Language, Systems, Platforms

- Specialized languages on scalable systems
  - MapReduce / Hadoop
  - Spark generalized data flow
- Systems for data preparation
- Systems for data visualization

Summary

- We now have access to massive quantities of data
  - Social networking and web/mobile apps traces
  - Automated data gathering (Sequencers in biology, particle accelerators in physics, automated cameras on astronomy)
- The data needs analysis for interpreting it
- This is the new "**Big Data**" market for parallel computing

Contents

- Introduction to parallelism
- Benefits to parallelism
- **The Map/Reduce Programming Model**
    - How can we write parallel data analysis code…

Queen Mary University of London

Our first parallel program

- **Task**: Count the number of occurrences of each word in one document
- **Input**: A text document
- **Output**: List of `<word, count>`

The	56
School	23
Queen	10
…	

Program Input

"QMUL has been ranked 9th among multi-faculty institutions in the UK, according to tables published today in the Times Higher Education.

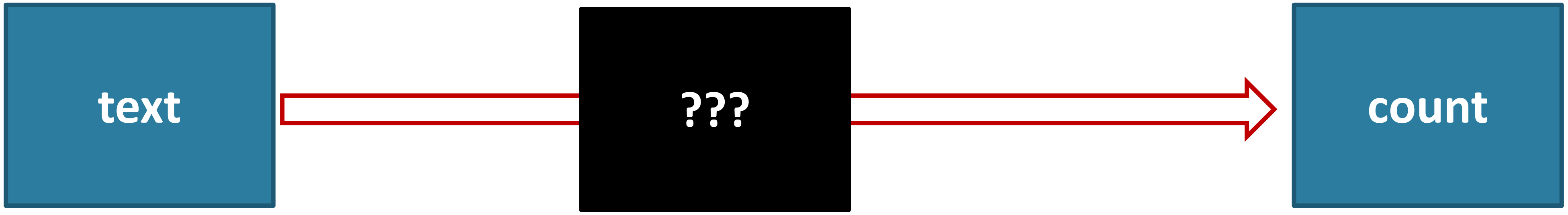
A total of 154 institutions were submitted for the exercise.

The 2008 RAE confirmed Queen Mary to be one of the rising stars of the UK research environment and the REF 2014 shows that this upward trajectory has been maintained.

Professor Simon Gaskell, President and Principal of Queen Mary, said: "This is an outstanding result for Queen Mary. We have built upon the progress that was evidenced by the last assessment exercise and have now clearly cemented our position as one of the UK's foremost research-led universities. This achievement is derived from the talent and hard work of our academic staff in all disciplines, and the colleagues who support them."

The Research Excellence Framework (REF) is the system for assessing the quality of research in UK higher education institutions. Universities submit their work across 36 panels of assessment. Research is judged according to quality of output (65 per cent), environment (15 per cent) and, for the first time, the impact of research (20 per cent)."

How to solve the problem?



QMUL has been ranked 9th among multi-faculty institutions in the UK, according to tables published today in the Times Higher Education.

A total of 154 institutions were submitted for the exercise.

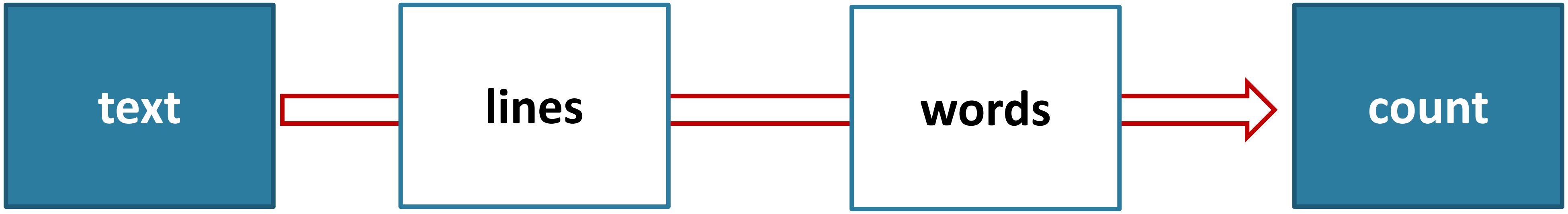
The 2008 RAE confirmed Queen Mary to be one of the rising stars of the UK research environment and the REF 2014 shows that this upward trajectory has been maintained.

Professor Simon Gaskell, President and Principal of Queen Mary, said: “This is an outstanding result for Queen Mary. We have built upon the progress that was evidenced by the last assessment exercise and have now clearly cemented our position as one of the UK’s foremost research-led universities. This achievement is derived from the talent and hard work of our academic staff in all disciplines, and the colleagues who support them.”

The Research Excellence Framework (REF) is the system for assessing the quality of research in UK higher education institutions. Universities submit their work across 36 panels of assessment. Research is judged according to quality of output (65 per cent), environment (15 per cent) and, for the first time, the impact of research (20 per cent).

The	**56**
School	**23**
Queen	**10**
.....	

How to solve the problem?



How to solve the problem on a single processor?

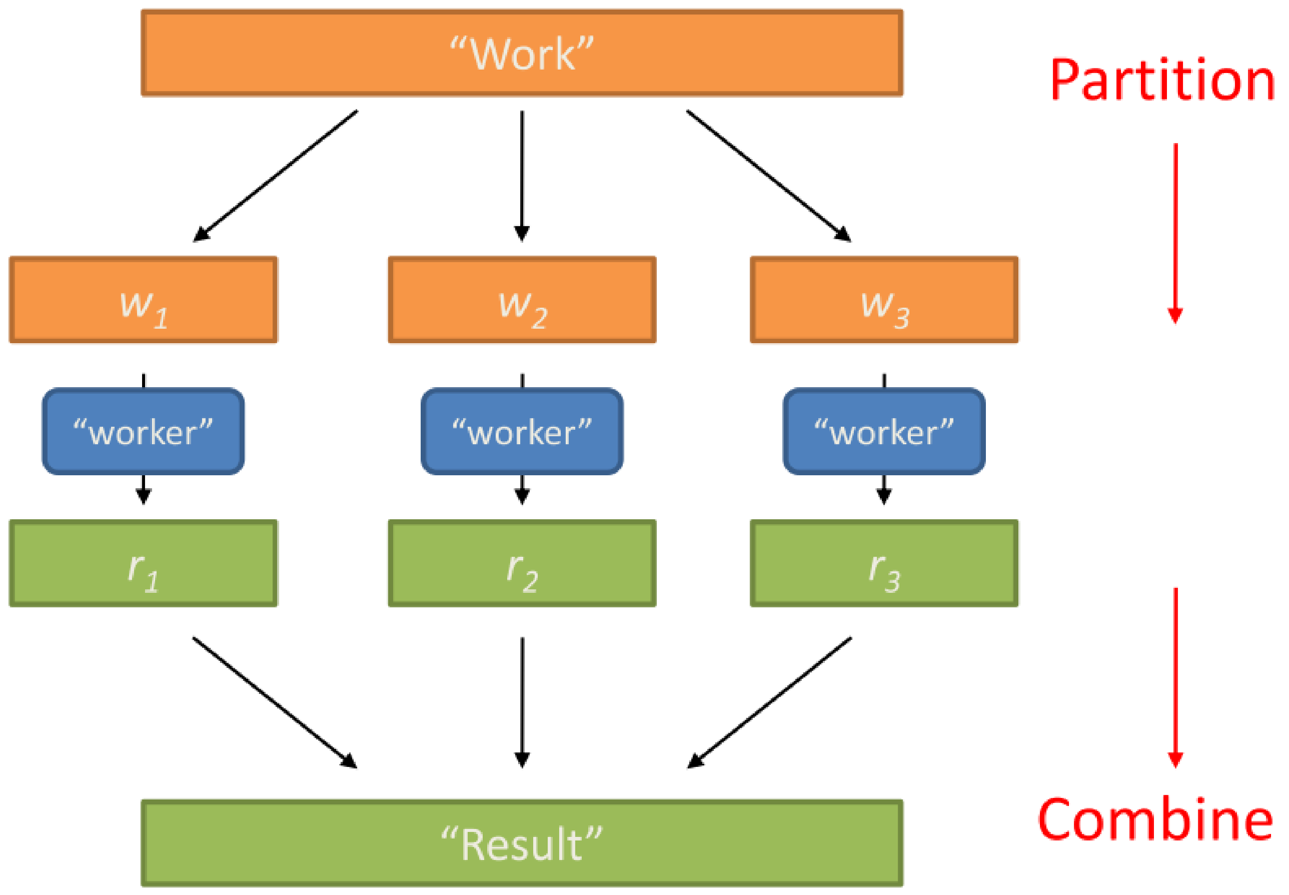
```
List<String> words= text.split();

Hashtable<String,Integer> count = new Hashtable();

for (String word: words){
  if(count.containsKey(word){
    count.put(word, count.get(word)+1);
  }
  else{
    count.put(word,1)
  }
}
```

But what if there are 10 billion lines of text?

Parallelizing the problem

- How would you do it in parallel?
- Splitting the load on subtasks:
  - **Split** sentences/lines into words
  - Count all the occurrences of each word on separate machines (in parallel!)

Divide and Conquer



But…

- …what do we do with the intermediate results?
  - We must merge each output into a single collection
  - Possibly requires parallelism too

- 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?

Image courtesy of Master isolated images at FreeDigitalPhotos.net

Enter **MapReduce**

- *"A simple and powerful interface that enables automatic parallelization and distribution of large-scale computations, combined with an implementation of this interface that achieves high performance on large clusters of commodity PCs."* [1]
- More simply, MapReduce is:
  - A parallel programming model and associated implementation.

[1]*Dean and Ghermawat, "MapReduce: Simplified Data Processing on Large Clusters", Google Inc.*

MapReduce

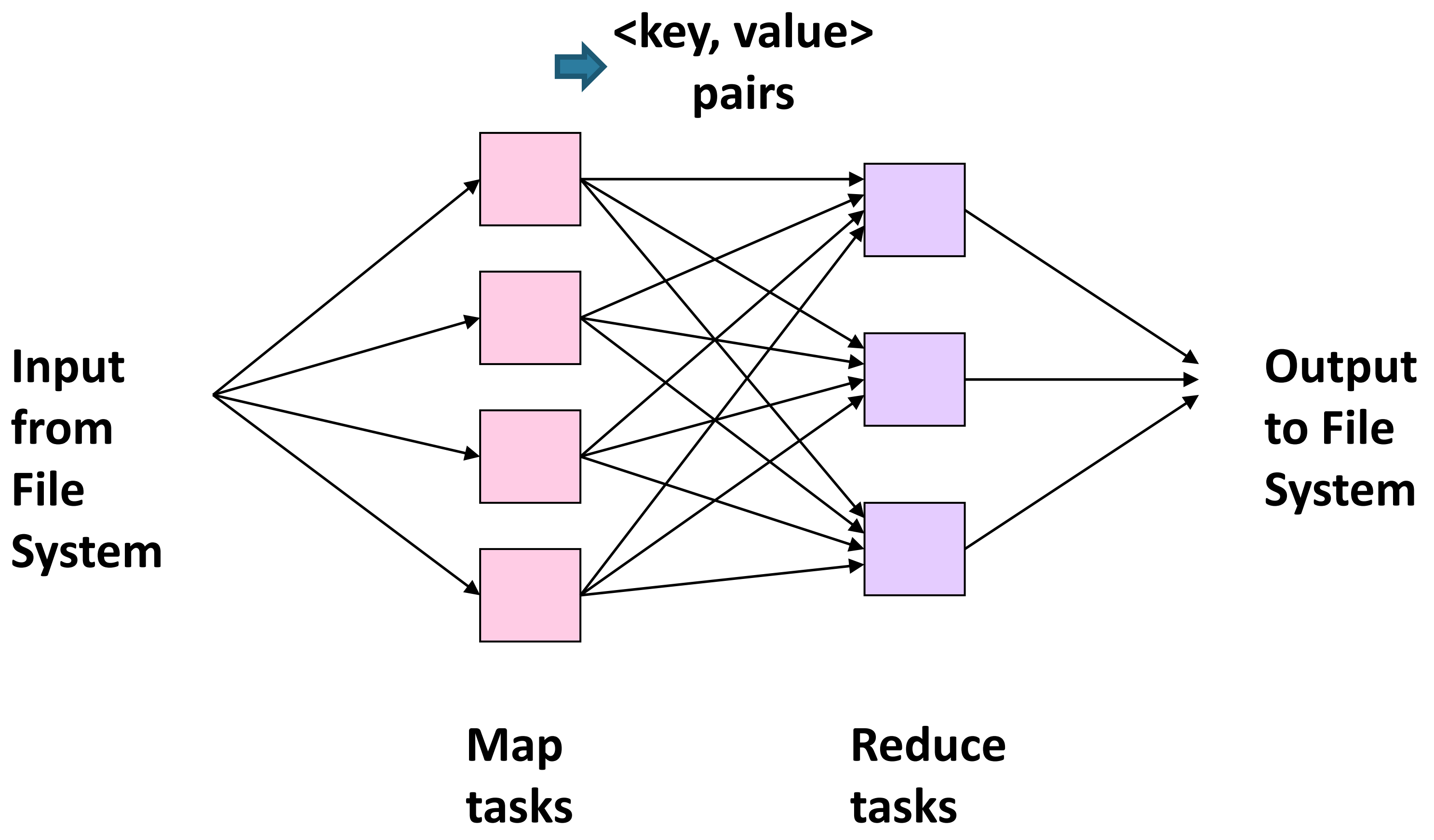
- **Key properties**
  - Google has used successfully for processing its “big-data” sets (~ 20000 peta bytes per day).
  - Users specify the computation in terms of a map and a reduce function.
  - Underlying runtime system automatically parallelizes the computation across large-scale clusters of machines.
  - Underlying system also handles machine failures, efficient communications, and performance issues.

MapReduce can refer to…

- The programming model
- The execution framework (aka “runtime”)
- The specific implementation

Usage is usually clear from context!

Map-Reduce Pattern



What is MapReduce Used For?

- At Google:
  - Index building for Google Search
  - Article clustering for Google News
  - Statistical machine translation
- At Yahoo!:
  - Index building for Yahoo! Search
  - Spam detection for Yahoo! Mail
- At Facebook:
  - Data mining
  - Advert optimisation
  - Spam detection

A brief history

- Inspiration from functional programming (e.g. Lisp)
  - **map()** function
    - Applies a function to each **individual** value of a sequence to create a **new list** of values
    - Example: square x = x * x
      map square [1,2,3,4,5] returns [1,4,9,16,25]
  - **reduce()** function
    - Combines all elements of a sequence using a binary operator
    - Example: sum = (each elem in arr, total +=)
      reduce [1,2,3,4,5] returns 15 (the sum of the elements)

MapReduce programming model (1/2)

- Process data using special **map**() and **reduce**() functions
  - The **map()** function is called on every item in the input and **emits** a series of intermediate <**key-value**> pairs
    - All values associated with a given key are grouped together
  - The **reduce()** function is called on every unique key, and its value list, and emits a value that is added to the output

MapReduce programming model (2/2)

- **Input:** a set of key/value pairs
- User supplies two functions:
  - **map()** function
    - map(k,v) → list(k1,v1)
  - **reduce()** function
    - reduce(k1, list(v1)) → v2
- (k1,v1) is an intermediate key/value pair
- Output is the set of (k1,v2) pairs

Example: Word count

```
public void Map (String filename,
                                  String text) {

  List<String> words = text.split();
  for (String word: words){
    emit(word, 1)  //Records the output (for reducer)
    }

}
```

Outputs <word,1> for each word it sees, e.g. <"cat", 1>

Example: Word count

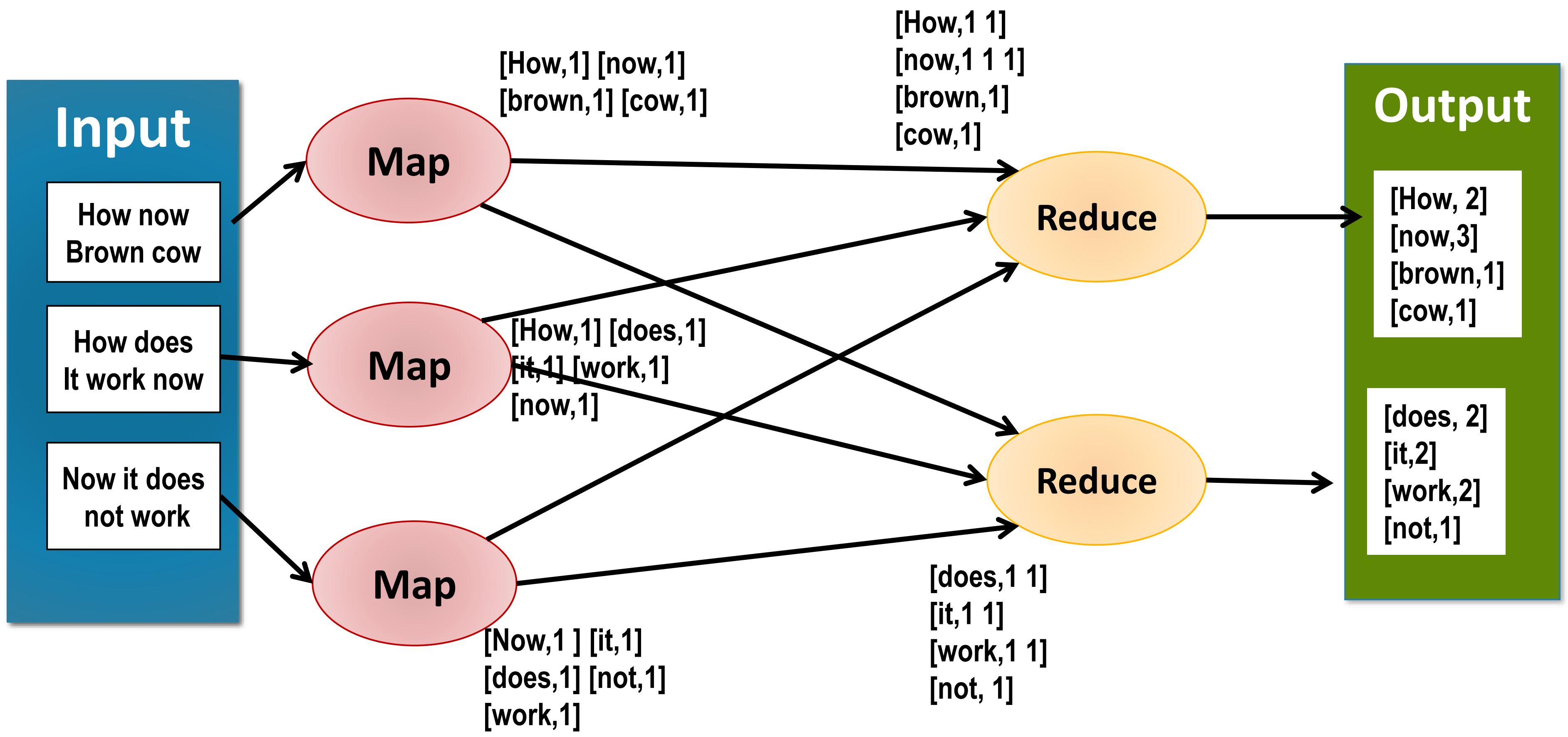
```
public void Reduce (String key,
                    List<Integer> values) {

  int sum = 0;
  for (Integer count: value
    sum+=count;
    }
  emit(key, sum);//Reco
}
```

Receives a list of all values associated with key (the word), e.g. “cat”. All values will be 1

How MapReduce parallelizes

- Input data is **partitioned** into processable chunks
- **One** Map job is executed **per chunk**
  - All can be parallelized (depends on number of nodes)
- One Reduce Job is executed for each distinct key emitted by the Mappers
  - All can be parallelized (partitioned ‘evenly’ among nodes)
- Computing nodes first work on Map jobs. After all have completed, a synchronization step occurs, and they start running Reduce jobs

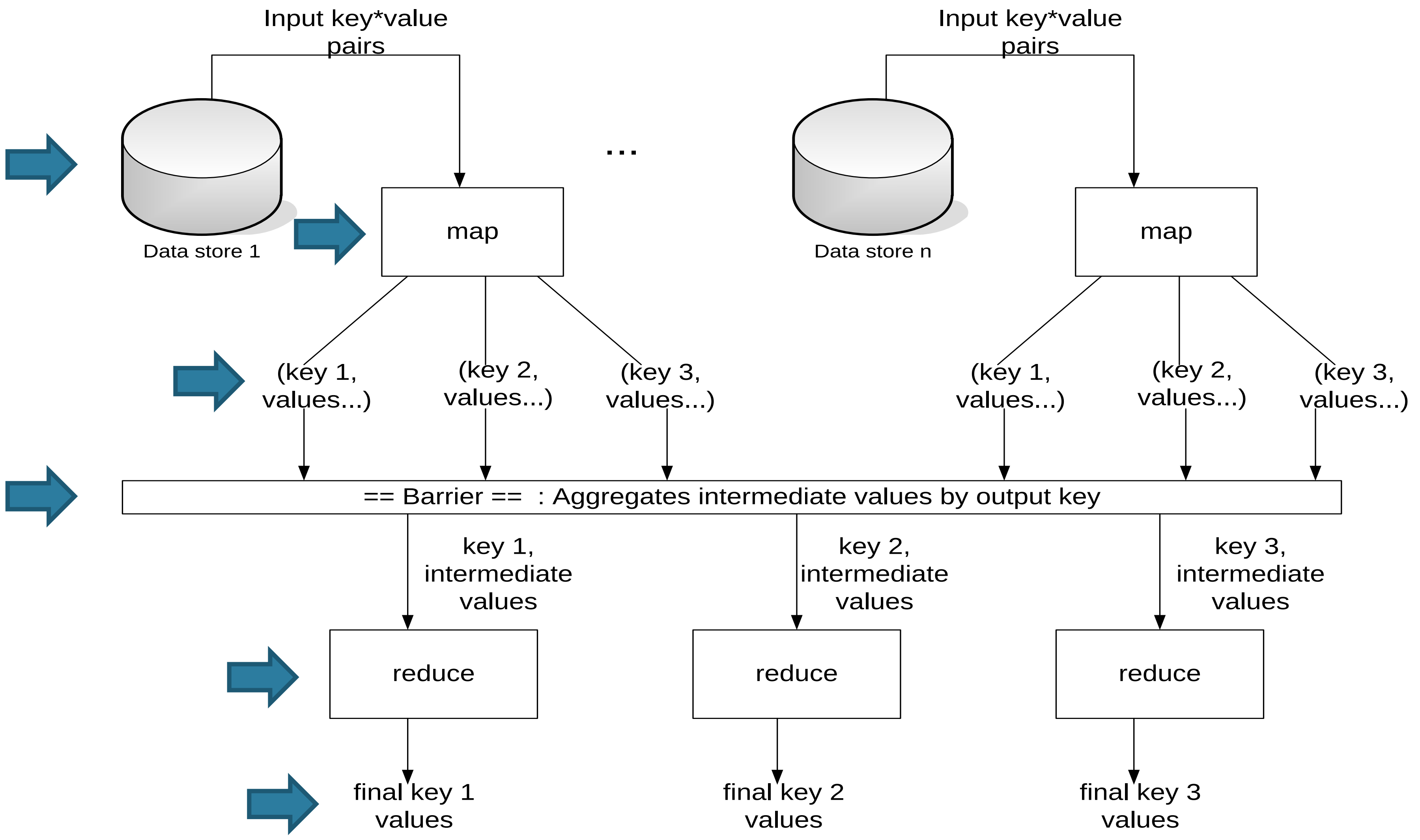
Word Count Example


Input
How now Brown cow
How does It work now
Now it does not work
Map
Map
Map
[How,1] [now,1] [brown,1] [cow,1]
[How,1] [does,1] [it,1] [work,1] [now,1]
[Now,1] [it,1] [does,1] [not,1] [work,1]
[How,1 1] [now,1 1 1] [brown,1] [cow,1]
[does,1 1] [it,1 1] [work,1 1] [not, 1]
Reduce
Reduce
Output
[How, 2] [now,3] [brown,1] [cow,1]
[does, 2] [it,2] [work,2] [not,1]

MapReduce benefits

- High-level parallel programming abstraction
- Framework implementations provide good performance results
- Scalability close to linear with increase in cluster size
- Greatly reduces parallel programming complexity
- However, it is not suitable for every parallel programming algorithm!

Synchronisation and message passing



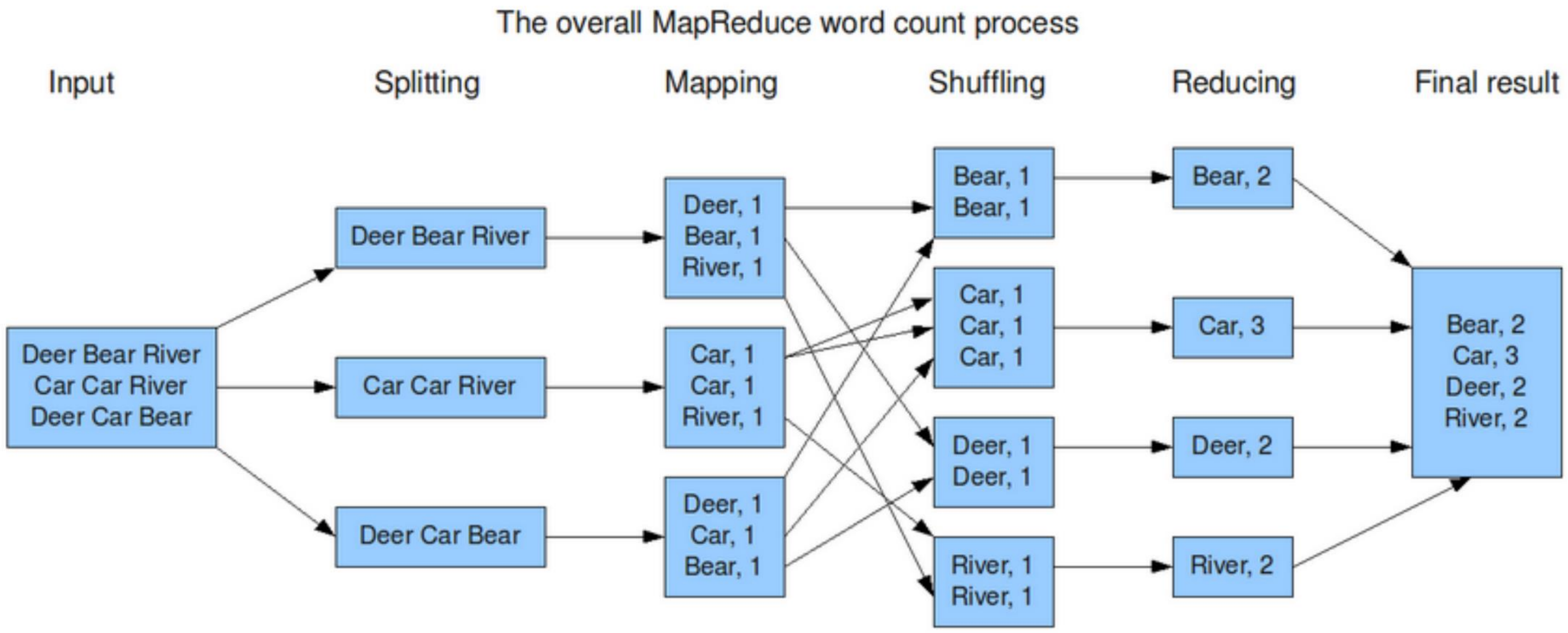
Shuffle and Sort steps

- Every key-value item generated by the mappers is collected
    - Items are transferred over the network
- Same key items are grouped into a list of values
- Data is partitioned among the number of Reducers
- Data is copied (**shuffled**) over the network to each Reducer
- The data provided to each Reducer is **sorted** according to the keys

MapReduce Runtime System

- Partitions input data
- Schedules execution across a set of machines
- Handles load balancing
- Shuffles, partitions and sorts data between Map and Reduce steps
- Handles machine failure transparently
- Manages inter process communication

MapReduce Process Example



MapReduce Implementations

- Google MapReduce
  - Not available outside Google
- Hadoop
  - An open-source implementation in Java
  - Development led by Yahoo!, used in production
  - Now an Apache project
  - Rapidly expanding software eco-system
- Custom research implementation
  - For GPUs, cell processors, etc.

Who uses Hadoop?

- Amazon/A9
- Facebook
- Google
- IBM
- Joost
- Last.fm
- New York Times
- PowerSet
- Veoh
- Yahoo!

Summary

- Introduction to parallelism
- Benefits to parallelism
- The Map/Reduce Programming Model