Advanced BI: the BIG DATA challenge

What's Big Data?

from Wikipedia:

- **Big data** is the term for a collection of data sets so large and complex that it becomes difficult to process using on-hand database management tools or traditional data processing applications.
- The challenges include capture, curation, storage, search, sharing, transfer, analysis, and visualization.
- The trend to larger data sets is due to the additional information derivable from analysis of a single large set of related data, as compared to separate smaller sets with the same total amount of data, allowing correlations to be found

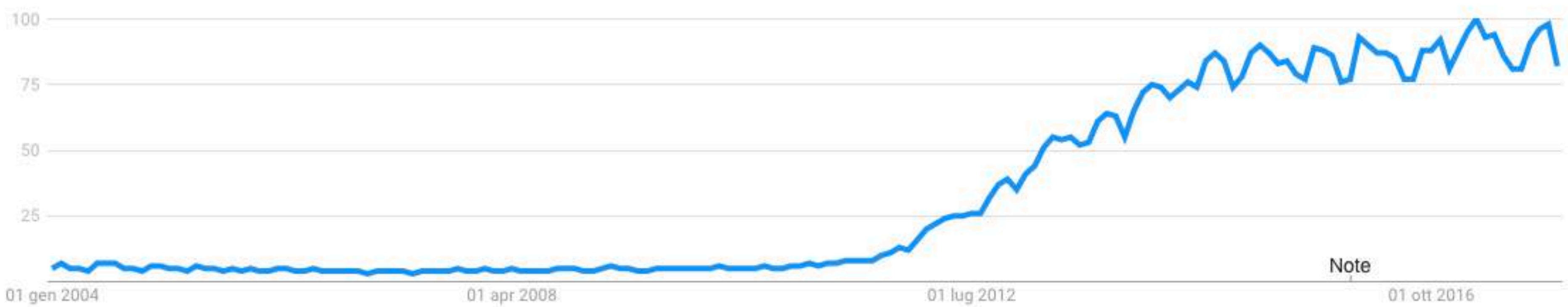
Why is BIG different from SMALL?

- **‘**Big Data**’ is similar to ‘small data**’**, but bigger in** size
- Having data bigger requires different approaches:
  - Techniques, tools and architecture
- Big Data generates value from the storage and processing of very large quantities of digital information **that cannot be analyzed** with traditional computing techniques.

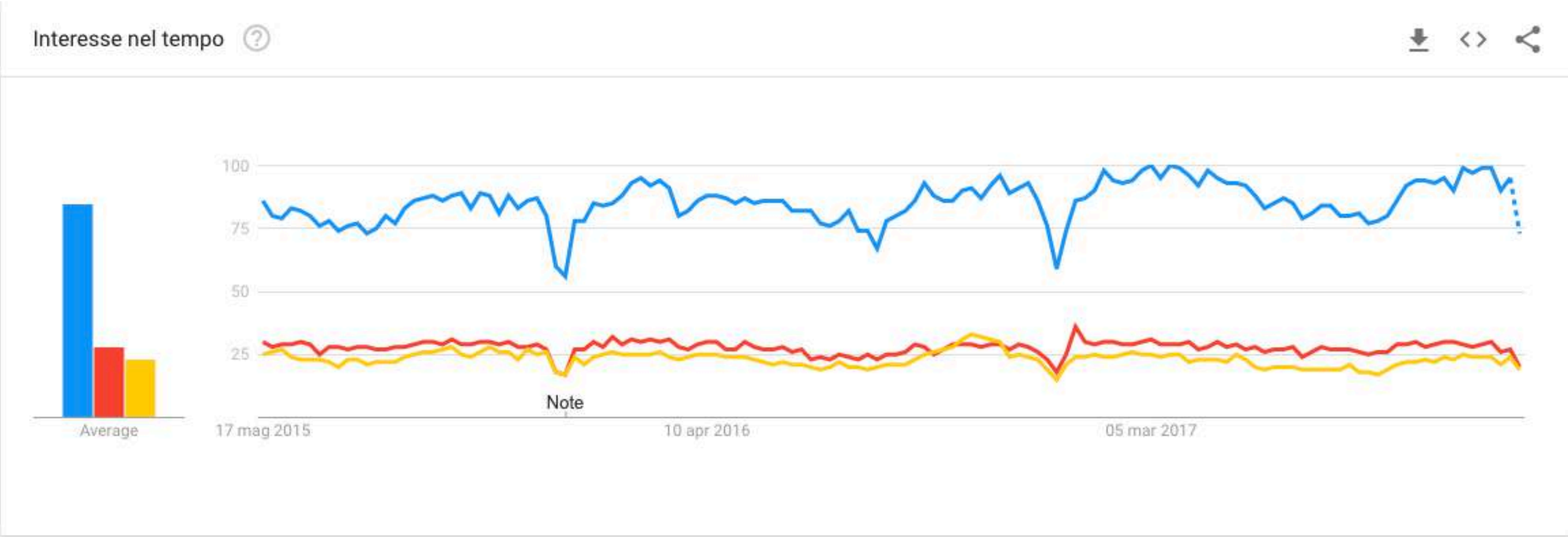
Is it a new challenge?

- Big Data may well be the Next Big Thing in the IT world.
- Big data burst upon the scene in the first decade of the 21st century.
- The first organizations to embrace it were online and startup firms. Firms like Google, eBay, LinkedIn, and Facebook were built around big data from the beginning.
- Like many new information technologies, big data can bring about dramatic cost reductions, substantial improvements in the time required to perform a computing task, or new product and service offerings.

Google trends “Big data” (2004-2017)



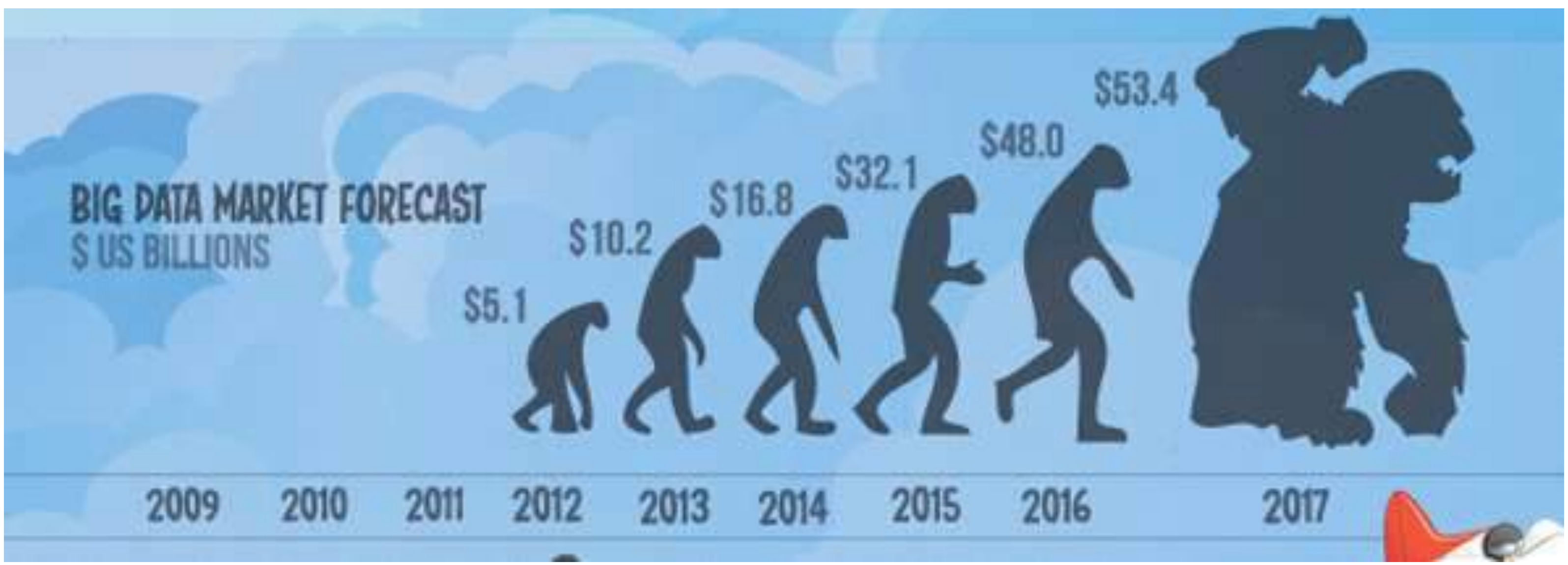
Interest stay stable in the past two years for main technology drivers



Big data
Business Intelligence
Internet of Things

How big is big?

- Walmart handles more than **1 million customer** transactions every hour.
- Facebook handles **40 billion photos** from its user base.
- Decoding the human genome originally took 10 years to process; now it can be achieved in **one week**.



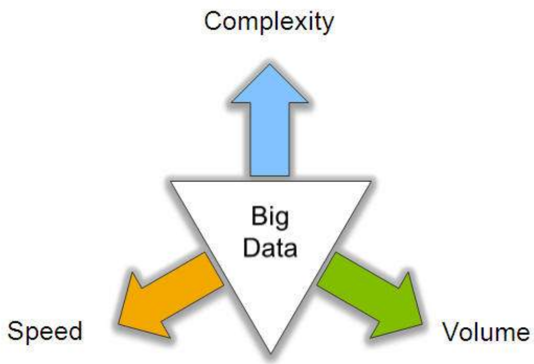
The 3 features of Big Data

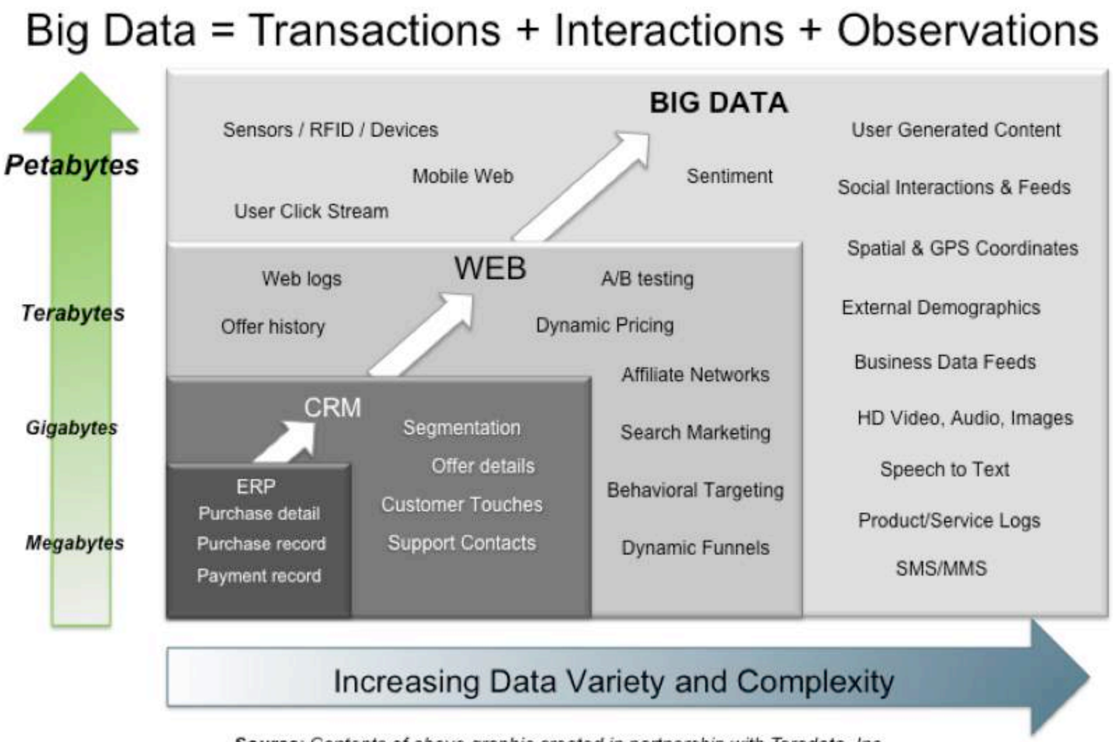
Volume
- Data quantity

Velocity
- Data Speed

Variety
- Data Types

The 3 V (+ 1 C)

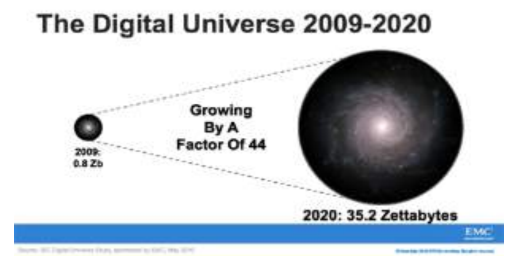




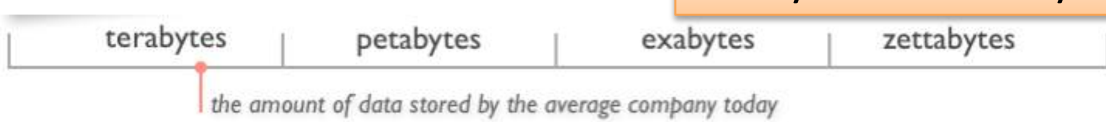
Source: *Contents of above graphic created in partnership with Teradata, Inc.*

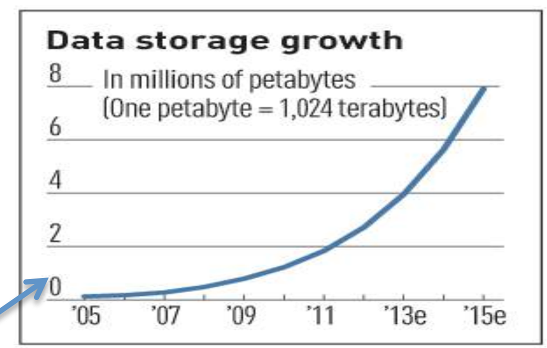
V1: Volume (Scale)

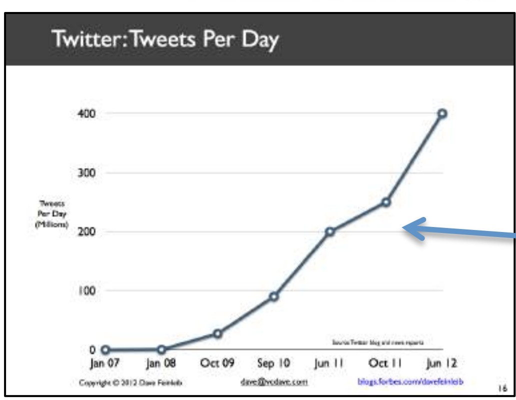
- **Data Volume**
  - 44x increase from 2009 2020
  - From 0.8 zettabytes to 35zb
- Data volume is increasing exponentially



2^{21} byte = 1 zettabyte







Exponential increase in collected/generated data

12+ TBs
of tweet data every day

? TBs of data every day

***25+ TBs* of log data every day**

30 billion RFID tags today (1.3B in 2005)

4.6 billion camera phones world wide

100s of millions of GPS enabled devices sold annually

76 million smart meters in 2009… 200M in 2014

2+ billion people on the Web by end 2011

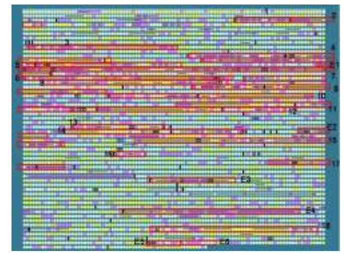
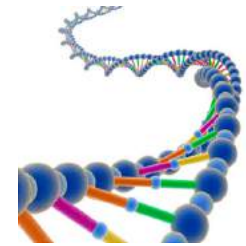
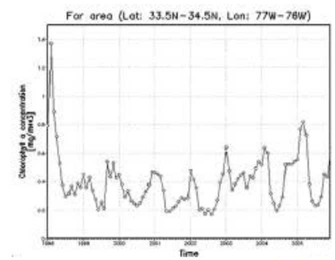
V2: Variety (and Complexity)

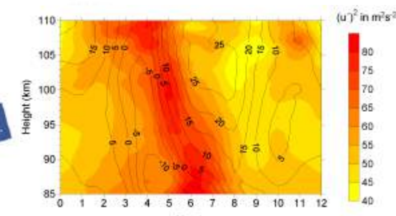
- Relational Data (Tables/Transaction/Legacy Data)
- Text Data (Web)
- Semi-structured Data (XML)
- Graph Data
  - Social Network, Semantic Web (RDF), …

- Streaming Data
  - You can only scan the data once

- A single application can be generating/collecting many types of data

- Big Public Data (on-line, weather, finance, etc)

To extract knowledge → all these types of data need to be linked together

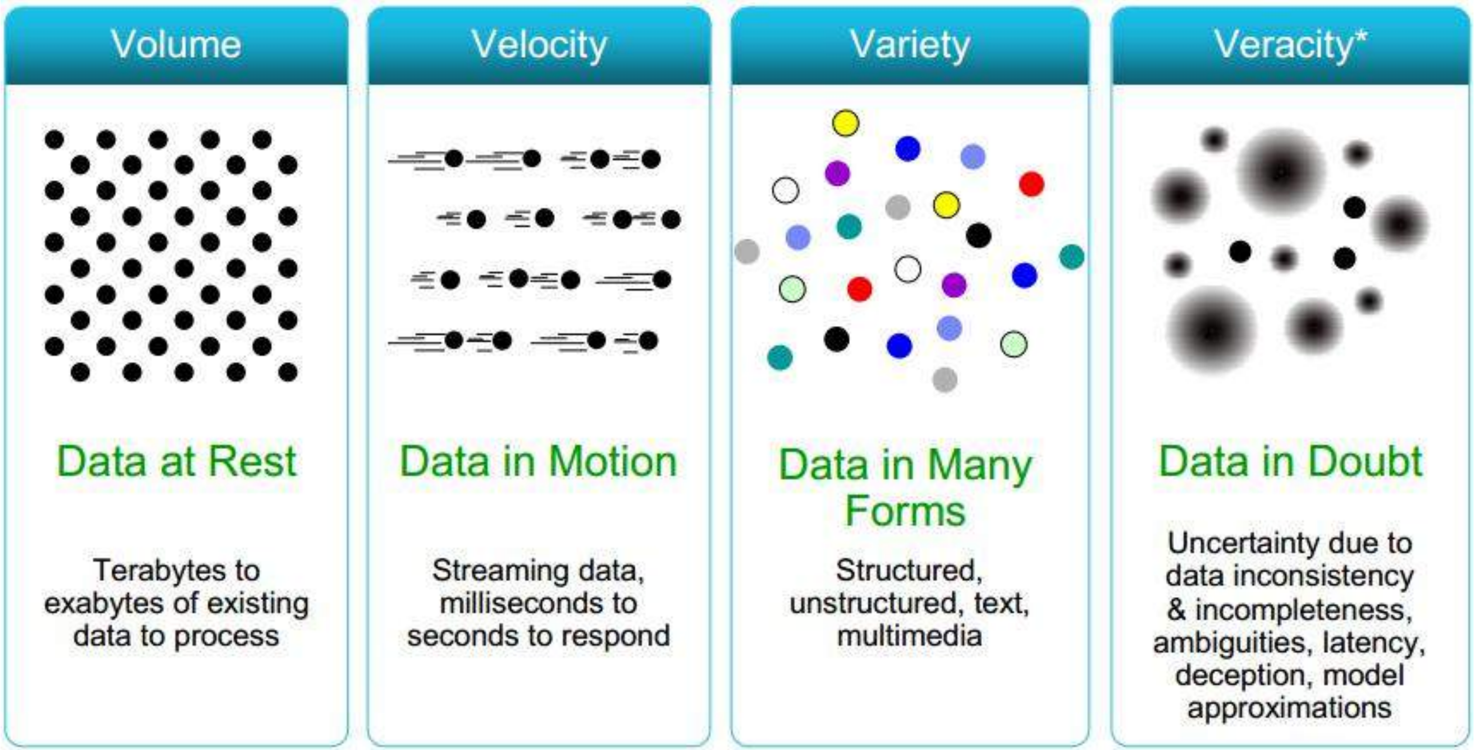
V3: Velocity (Speed)

- Data is begin generated fast and need to be processed fast
- Online Data Analytics
- Late decisions ➔ missing opportunities

- **Examples**
  - **E-Promotions:** Based on your current location, your purchase history, what you like ➔ send promotions **right now** for store next to you
  - **Healthcare monitoring:** sensors monitoring your activities and body ➔ **any abnormal measurements require immediate reaction**

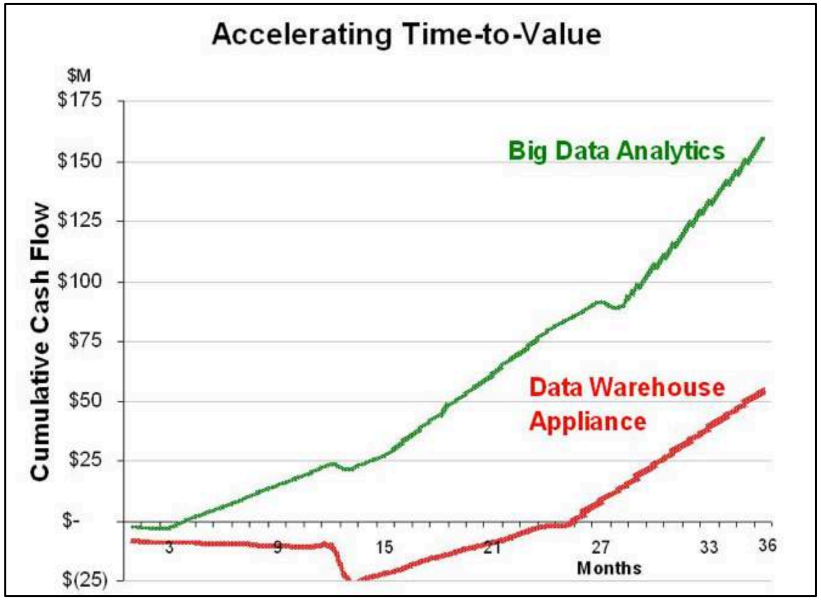
Without velocity, no real-time BI!

Some Make it 4V's: veracity



Big Data Analytics

- Big data is more real-time in nature than traditional DW applications
- Traditional DW architectures (e.g. Exadata, Teradata) are not well-suited for big data apps
- Massively parallel processing architectures must be used for big data apps



Big data: What are the issues?

Where processing is hosted?
- Distributed Servers / Cloud (e.g. Amazon EC2)

Where data is stored?
- Distributed Storage (e.g. Amazon S3)

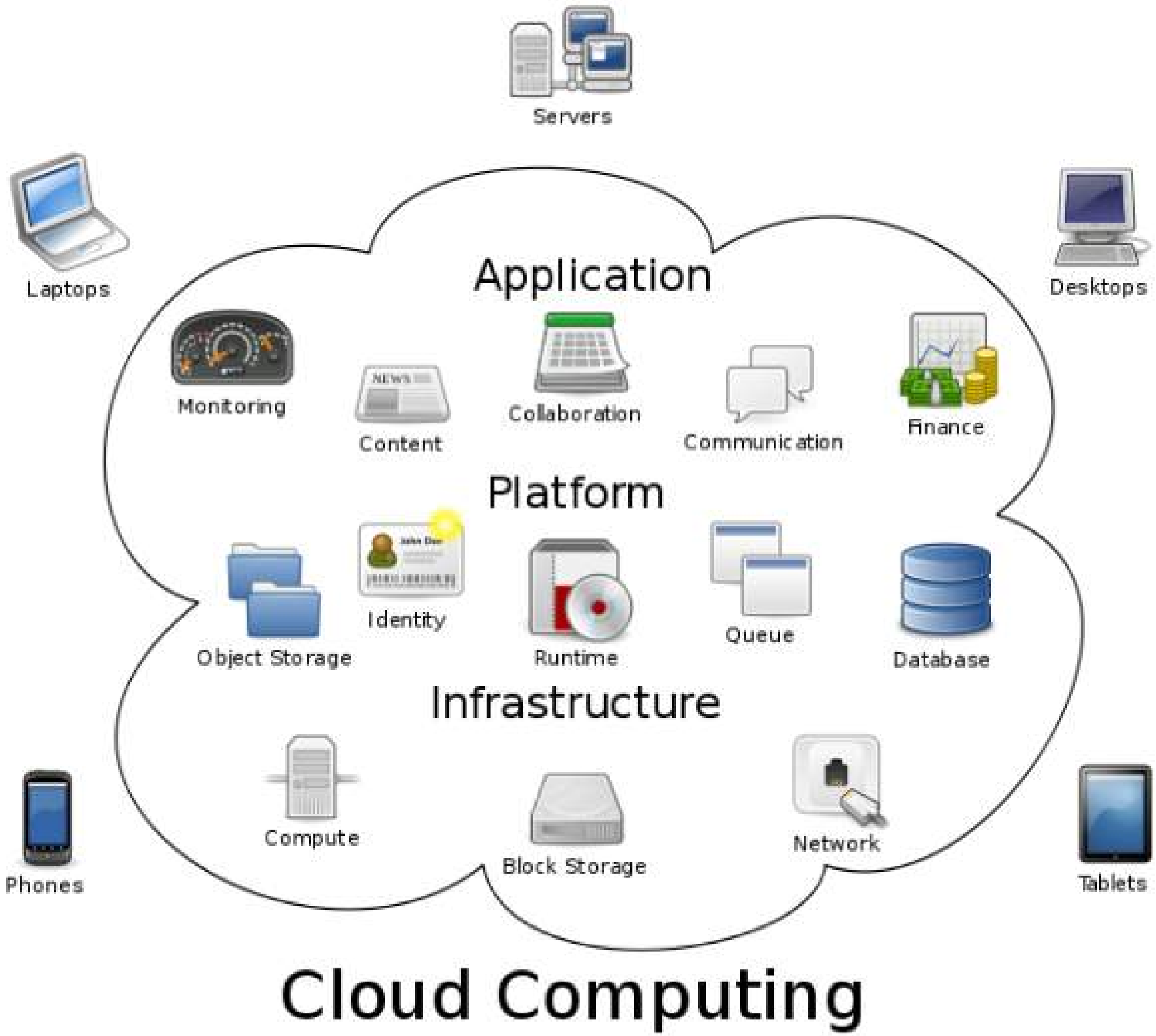
What is the programming model?
- Distributed Processing (e.g. MapReduce)

How data is indexed?
- High-performance schema-free databases (e.g. MongoDB)

What operations are performed on data?
- Analytic / Semantic Processing

1-2. Hosting and storing: solution is Cloud Computing

- IT resources provided “as a service”
  - Compute, storage, databases, queues
- Clouds leverage economies of scale of commodity hardware
  - Cheap storage, high bandwidth networks & multicore processors
  - Geographically distributed data centers
- Offerings from Microsoft, Amazon, Google, …



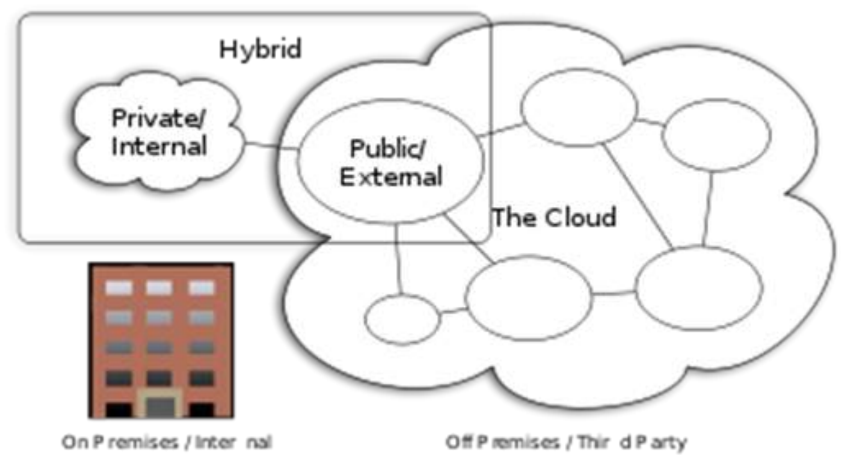
wikipedia:Cloud Computing

Benefits of Cloud Computing

- Cost & management
  - Economies of scale, “out-sourced” resource management
- Reduced Time to deployment
  - Ease of assembly, works “out of the box”
- Scaling
  - **On demand** provisioning, co-locate data and compute
- Reliability
  - Massive, redundant, shared resources
- Sustainability
  - Hardware **not owned**

Types of Cloud Computing

- **Public Cloud**: Computing infrastructure is hosted at the vendor's premises.
- **Private Cloud**: Computing architecture is dedicated to the customer and is **not shared** with other organisations.
- **Hybrid Cloud**: Organisations host some critical, secure applications in private clouds. The not so critical applications are hosted in the public cloud
  - **Cloud bursting**: the organisation uses its own infrastructure for normal usage, but cloud is used for peak loads.



Cloud Computing Types

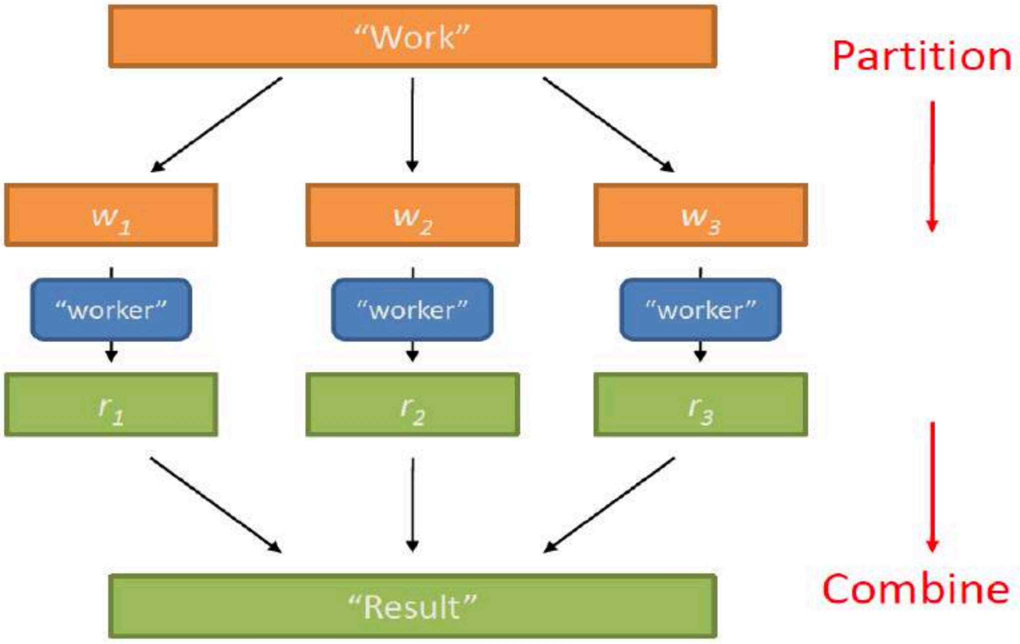
Classification of Cloud Computing based on Service Provided

- Infrastructure as a service (IaaS)
  - Offering hardware related services using the principles of cloud computing. These could include storage services (database or disk storage) or virtual servers.
  - Amazon EC2, Amazon S3, Rackspace Cloud Servers and Flexiscale.
- Platform as a Service (PaaS)
  - Offering a development platform on the cloud.
  - Google's Application Engine, Microsofts Azure, Salesforce.com's force.com .
- Software as a service (SaaS)
  - Including a complete software offering on the cloud. Users can access a software application hosted by the cloud vendor on pay-per-use basis. This is a well-established sector.
  - Salesforce.coms' offering in the online Customer Relationship Management (CRM) space, Googles gmail and Microsofts hotmail, Google docs.

3. Programming models for Big data

- The main computational model is called MapReduce
- Based on "Divide et Impera" (divide and conquer)

1. Work is (conveniently) partitioned and assigned to workers (workers are computing units)
2. Results from each worker (r_i) are (appropriately) combined



Issues with parallelization (1)

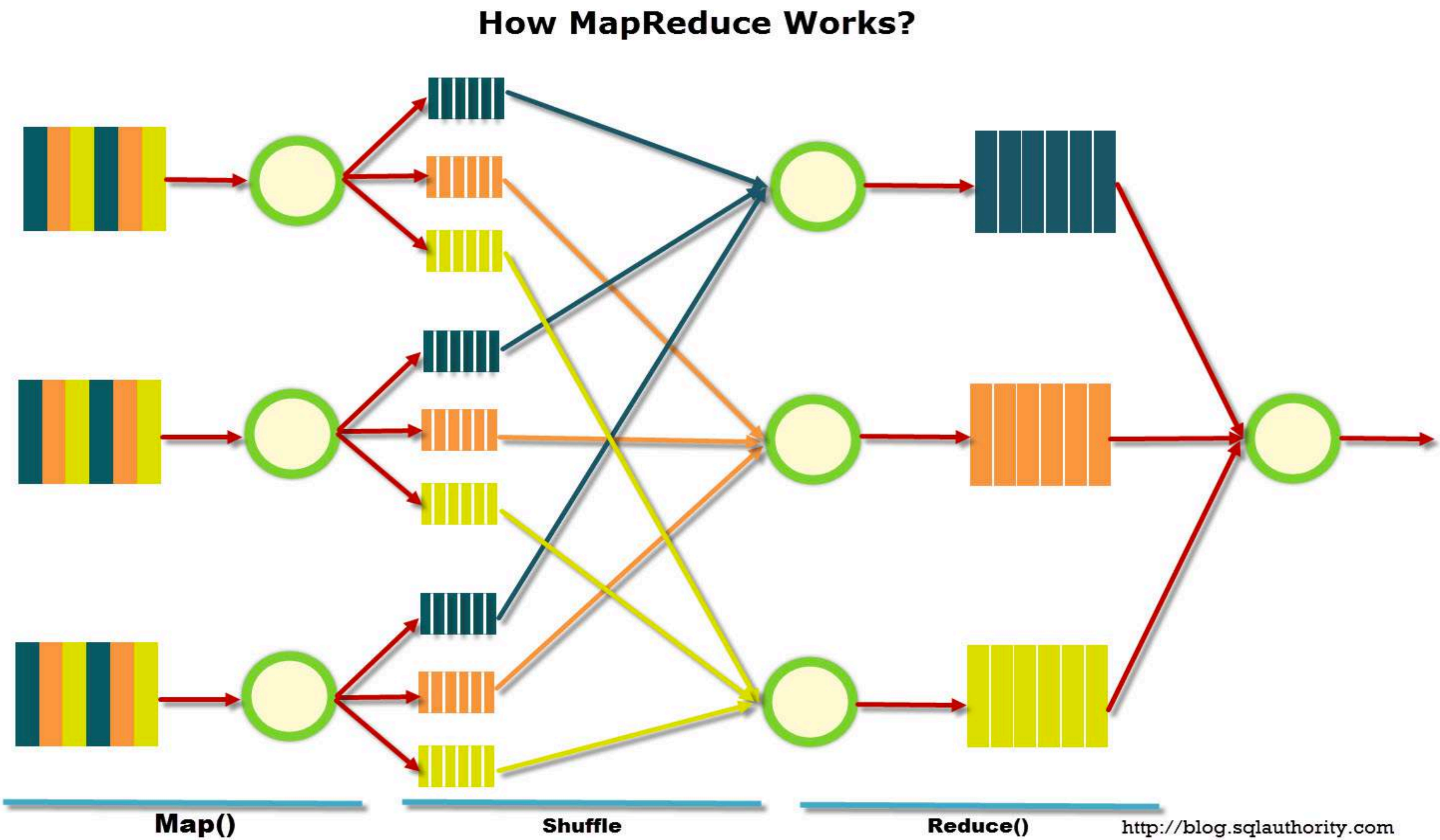
- How do we assign work units to workers?
- What if we have more work units than workers?
- What if workers need to share partial results?
- How do we aggregate partial results?
- How do we know all the workers have finished?
- What if workers “die”?

Issues with parallelization (2)

- **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**

Source: Ricardo Guimarães Herrmann

Big Data Processing computation model: MapReduce



Basic MapReduce steps

MAP:

- Iterate over a large number of records of the database
- Extract "something of interest" from each

REDUCE:

- Shuffle and sort intermediate results
- Aggregate intermediate results
- Generate final output

MapReduce programming model

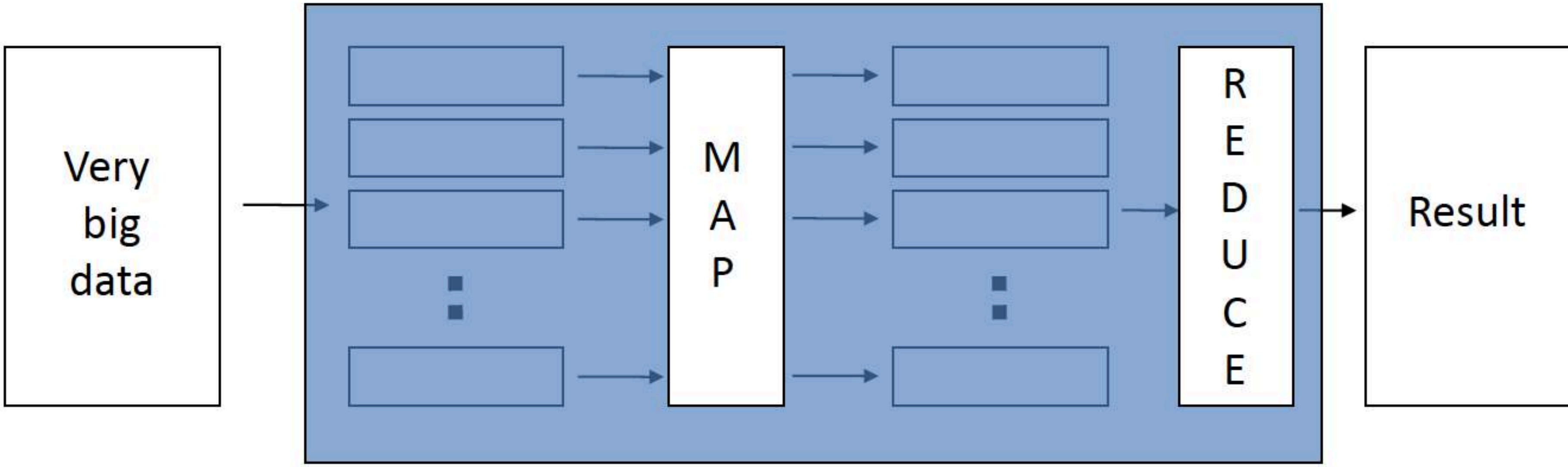
- Programmers specify two functions:

map (k, v) → [(k', v')]

reduce (k', [v']) → [(k', v')]

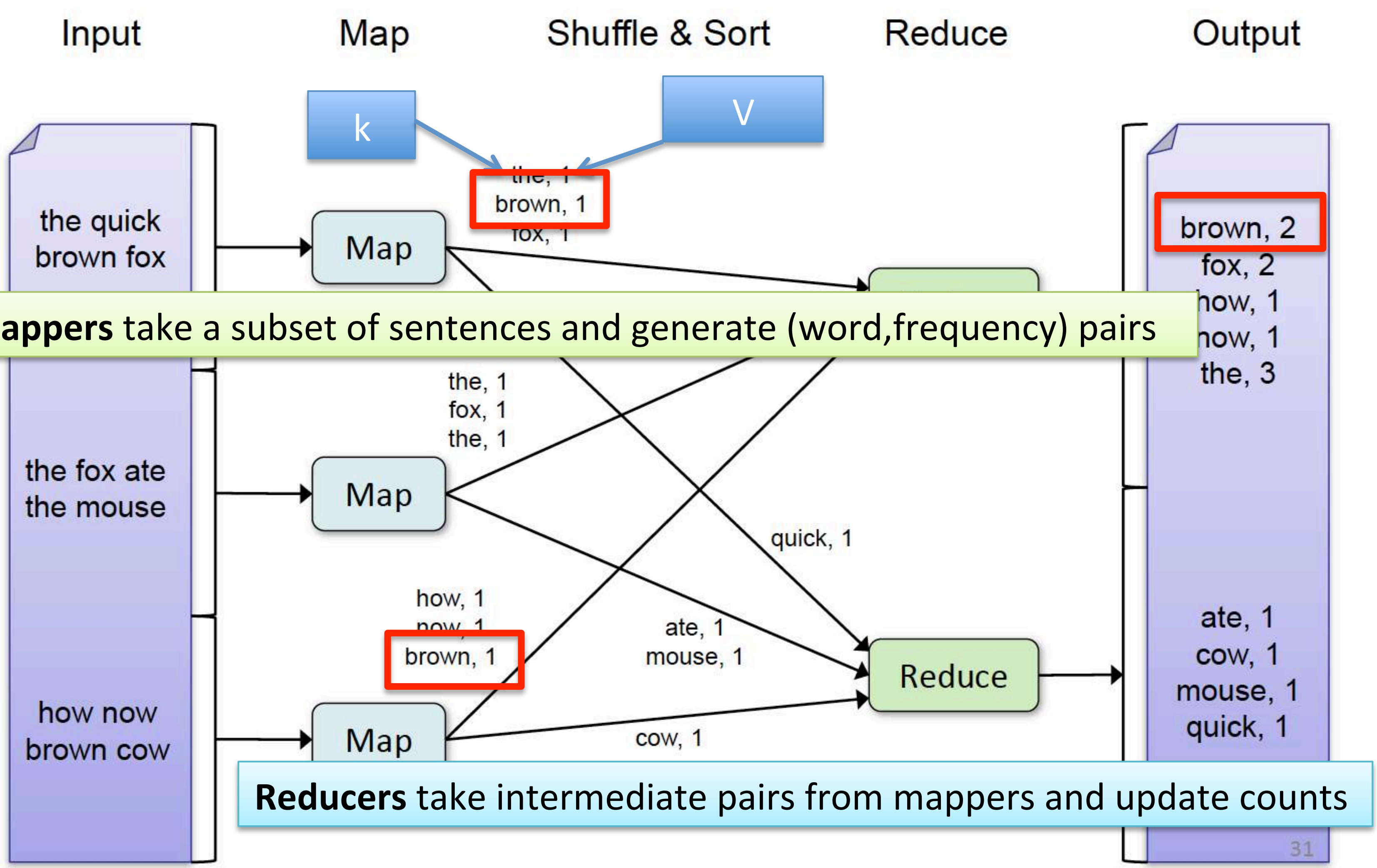
- k,k' are keys (e.g. **attribute names**) v,v' are **values**
- Mappers map a set of (attribute,value) pairs(k,v) into another set of (k',v') pairs, called intermediate pairs.
- All values with the same key k' are sent to the same "reducer "
- The execution handles everything else

MAP+REDUCE



- Map:
  - Accepts *input* key/value pair
  - Emits *intermediate* key/value pair
- Reduce :
  - Accepts *intermediate* key/value* pair
  - Emits *output* key/value pair

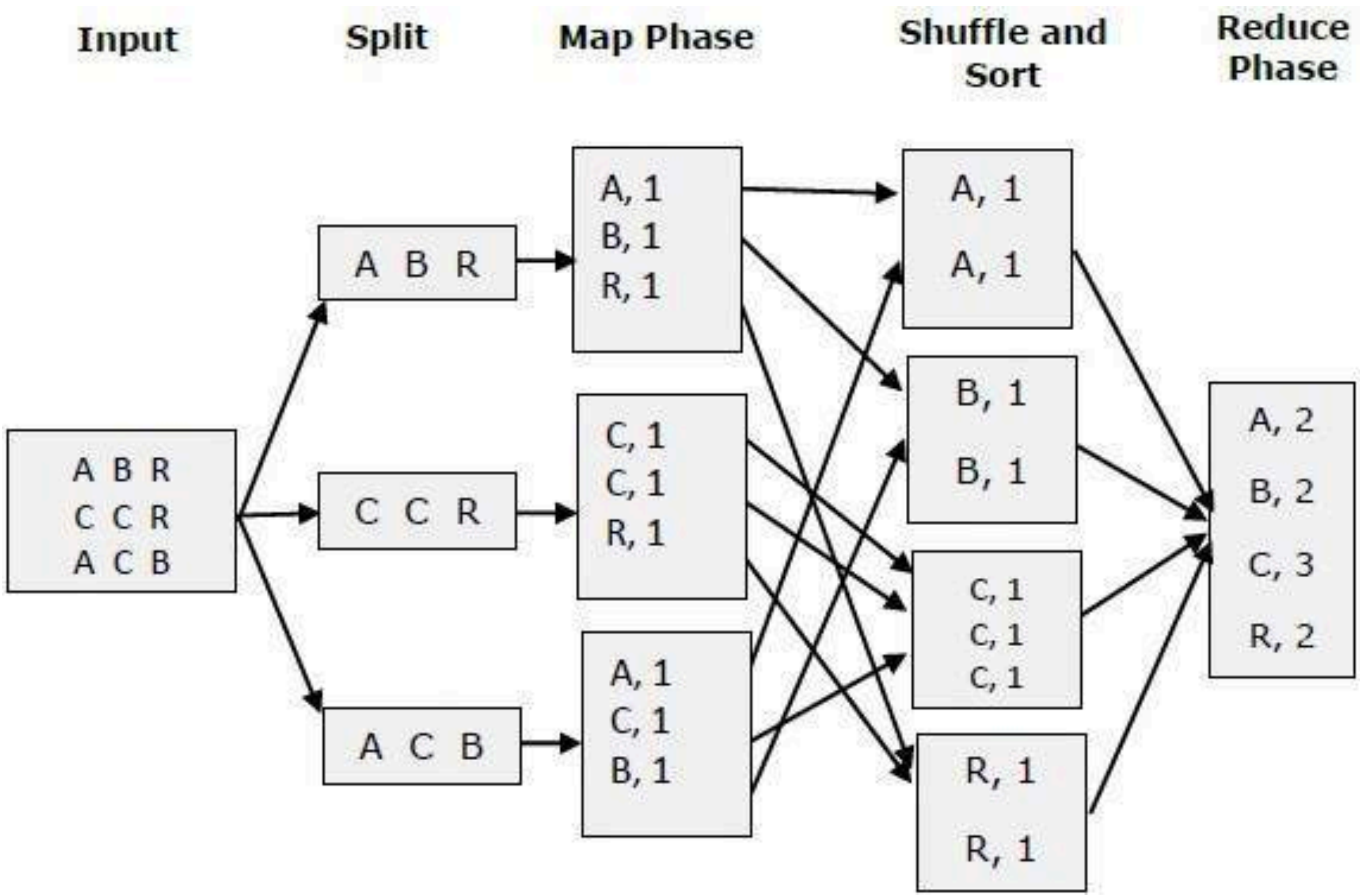
Example 1: counting words in very large textual datasets



Mappers take a subset of sentences and generate (word,frequency) pairs

Reducers take intermediate pairs from mappers and update counts

In a more sketchy way:



Example 2: K Means

Traditional

AssignCluster():

- For each point p

  Assign p the closest c

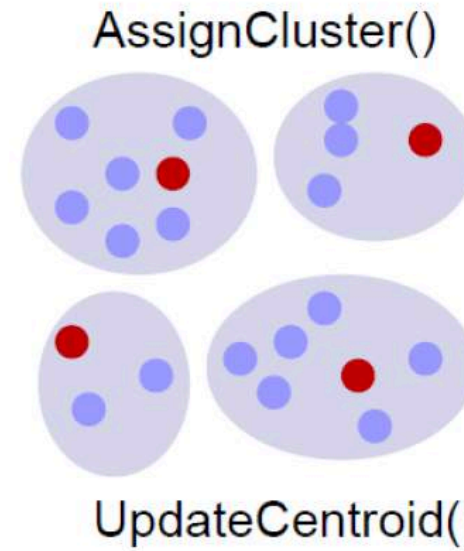
UpdateCentroids ():

- For each cluster

  Update cluster center

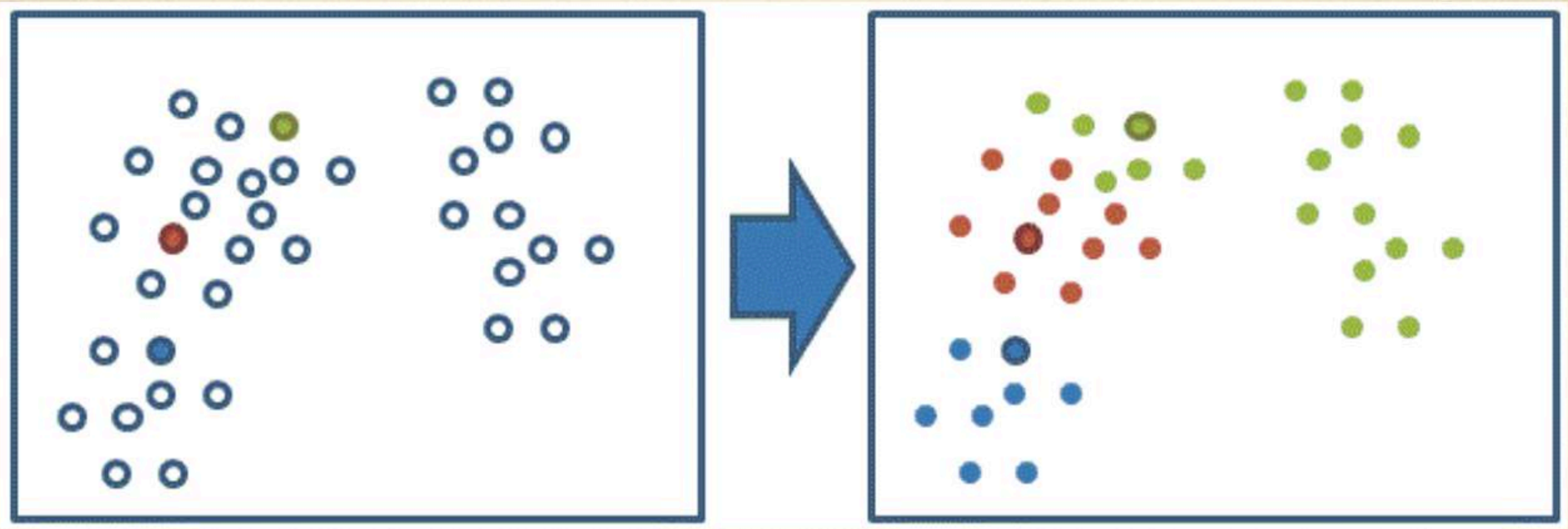
Kmeans ()

- While not converge:
  - AssignCluster()
  - UpdateCentroids()



Traditional K-Means

most intensive calculation to occur is the calculation of distances.



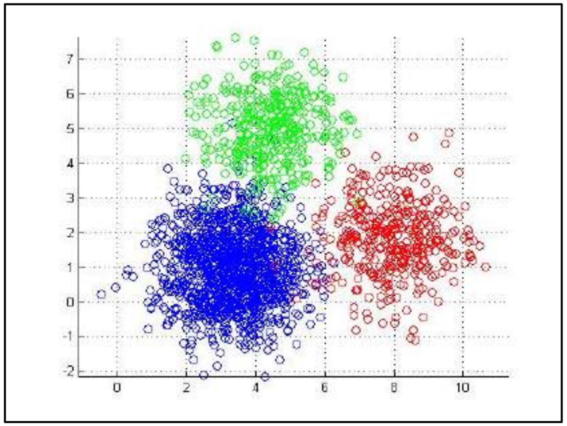
each iteration require nk distance

n=number of records, k number of clusters. For billions of records the algorithm would NOT produce a result

Can we parallelize?

- The distance computations between one node with the (current) centroids **is irrelevant** to the distance computations between other nodes with the corresponding centroids.
- distance computations between different nodes with centroids **can be executed in parallel!**

K-Means in MapReduce

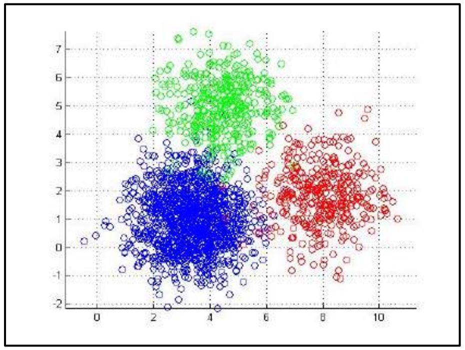


- **Input**
  - Dataset (set of records) --Large
  - Initial centroids (K points) --Small
- **Map Side**
  - Each Mapper reads the K-centroids + one block from dataset. Let’s call “point” a record of the dataset
  - Assign each point to the closest centroid
  - Output a set of <centroid, point> pairs (the (k’v’), where k’ is the centroid and v’ is a point of the dataset)

K-Means in MapReduce (Cont'd)

- **Reduce Side**
  - Every reducer gets **all points** [v'] for a given centroid k' (or for a subset of centroids)
  - Re-compute a new centroid for this cluster
  - Output: <new centroid>

- **Iteration Control**
  - Compare the old and new set of K-centroids
    - If similar ➔ Stop
    - Else
      - If max iterations has reached ➔ Stop
      - Else ➔ Start another Map-Reduce Iteration



Let’s make an example (K=2)

First step is to split the dataset

data

1,1
2,2
3,3
11,11
12,12
13,13

target

1st split

1,1
2,2
3,3

2nd split

11,11
12,12
13,13

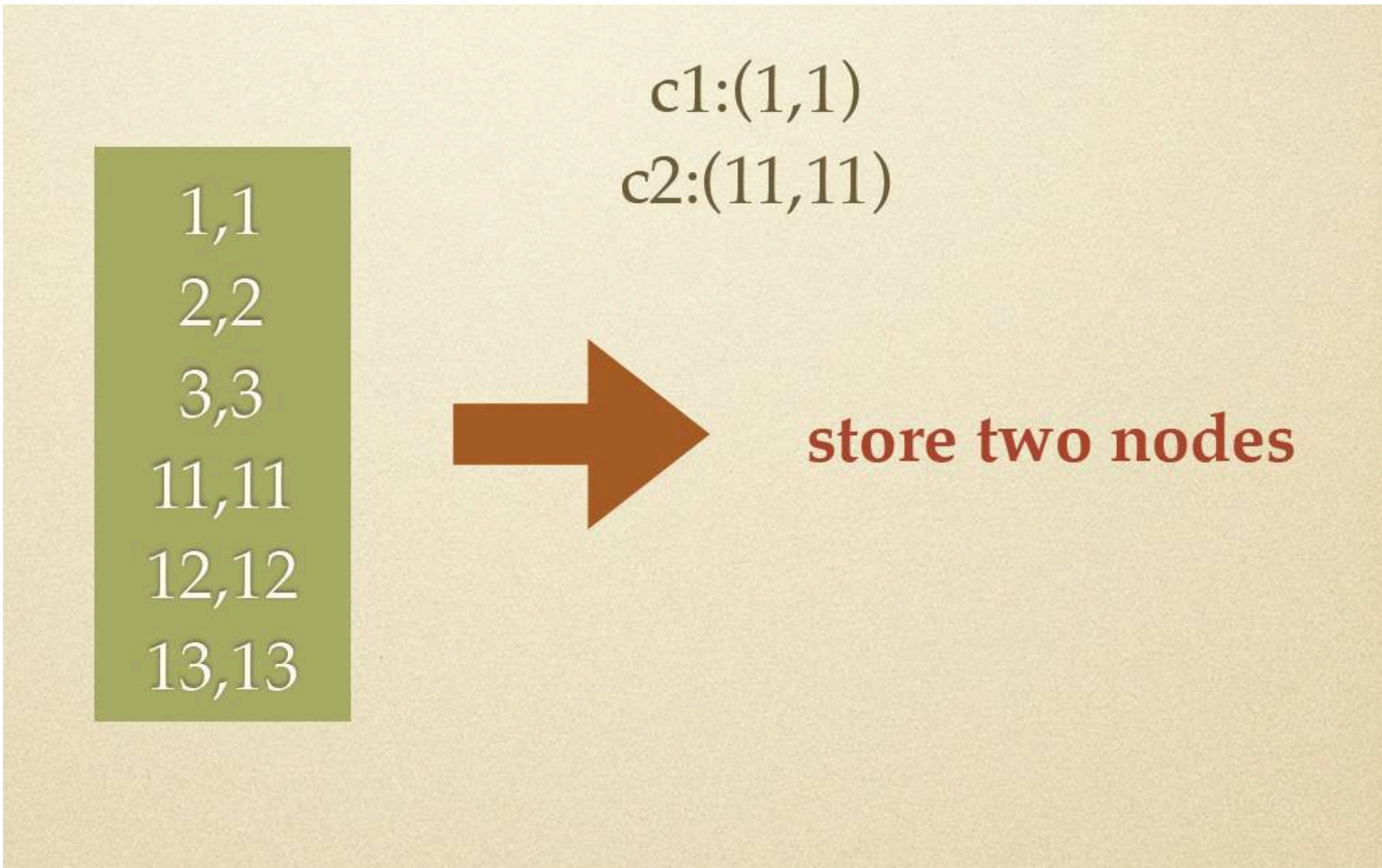
Then, two centroids are randomly chosen

1,1
2,2
3,3
11,11
12,12
13,13

random two centroid

c1:(1,1)
c2:(11,11)

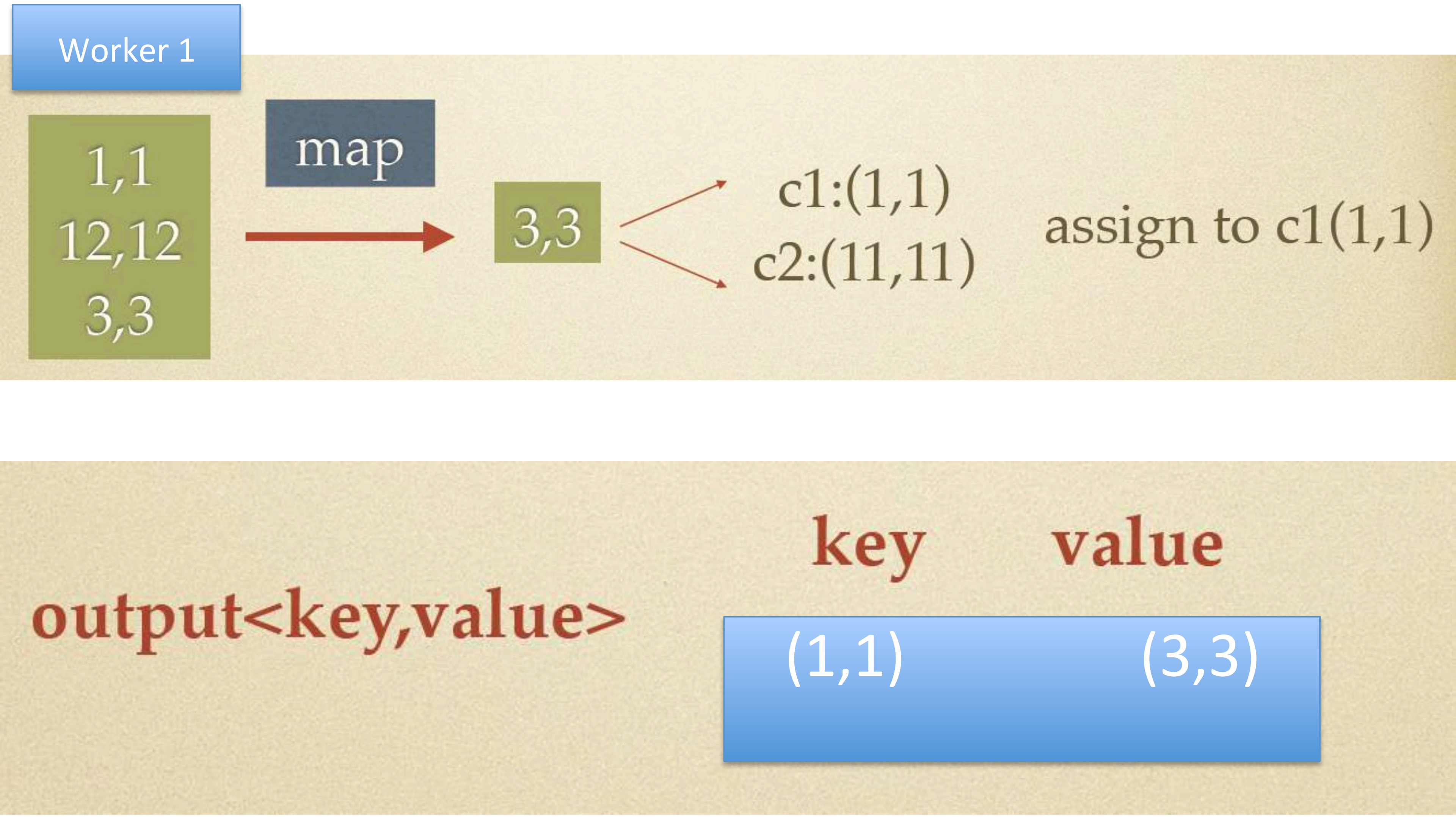
Step 2



Partition the task on 2 “workers”

Each worker receives a subset of the data and a centroid

Workers apply the Map step: tehy assign a value (a point) to each key (a cluster centroid)



Workers who are "reducers" merge all values with the same key

	key	value
output<key,value>		
	(1,1) ,	{(4,4),{(1,1),(3,3)},2}
	(11,11) ,	{(12,12),(12,12),1}

(1,1) **centroid**

{(4,4),{(1,1),(3,3)},2} **cluster members (the last number is the cluster ID)**

Next, new centroids are computed and process is iterated

K-means Clustering Parallel Efficiency

- Shantenu Jha et al. A Tale of Two Data-Intensive Paradigms: Applications, Abstractions, and Architectures. 2014.

Shows results as the number of centroids increase, the number of Cores (computers) increases and the type of platform changes

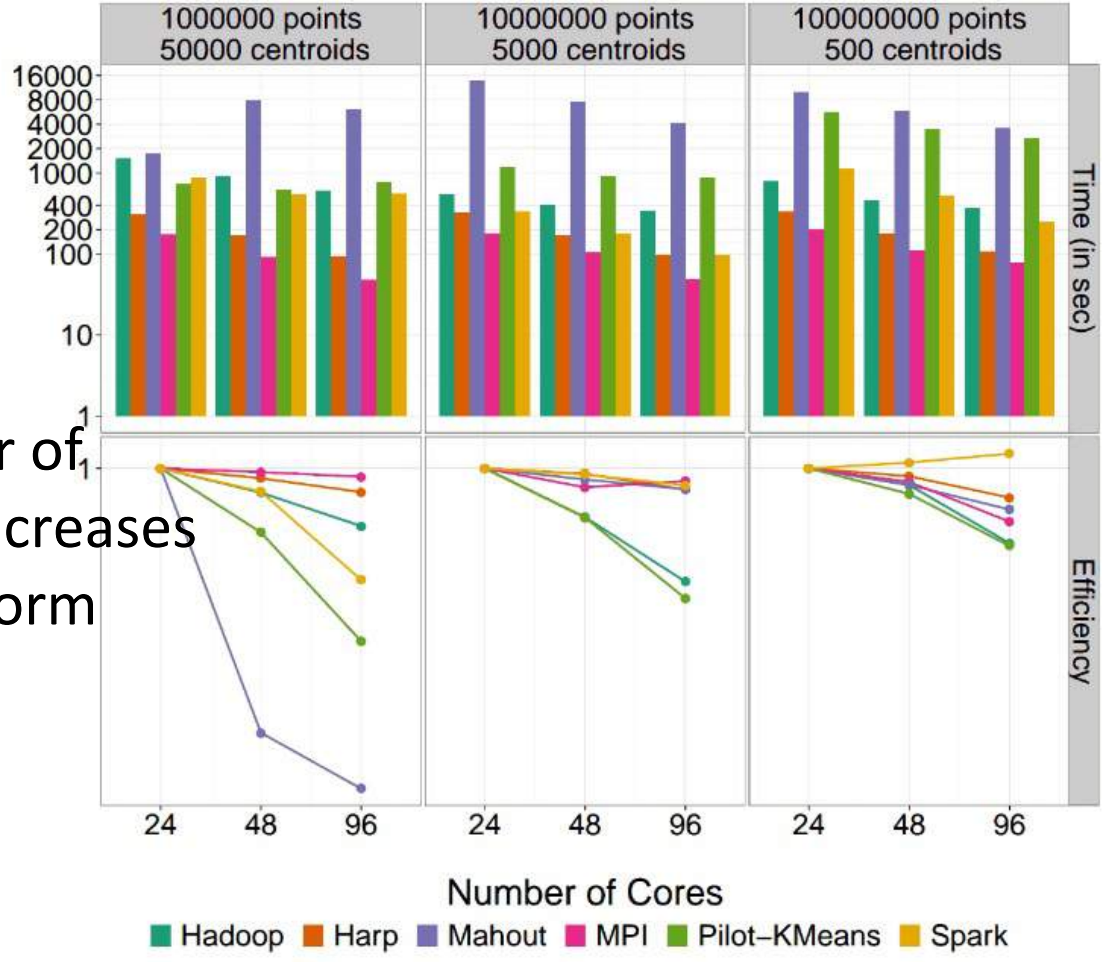


Fig. 5. Time-To-Completion for KMeans on Different Backends

Who is using MapReduce?

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

Applications of BD

Smarter Healthcare

Homeland Security

Traffic Control

Manufacturing

Multi-channel sales

Telecom

Trading Analytics

Search Quality

Risks of big data

- Security:
  - To make more sense from the big data, organizations would need to start integrating parts of their sensitive data into the bigger data.
- Cost: Costs escalate too fast
  - But often isn't necessary to capture 100% of the data
- Privacy: Many sources of big data are private (e.g. health data)
  - Self-regulation
  - Legal regulation

When to adopt Big Data solutions?

- You can’t process the amount of data that you want to because of the limitations of your current platform.
- You can’t include new/contemporary data sources (e.g., social media, RFID, Sensory, Web, GPS, textual data) because it does not comply with the data schema.
- You need to (or want to) integrate data as quickly as possible to be current on your analysis.
- You want to work with a schema-on-demand data storage paradigm because the variety of data types.
- The data is arriving so fast at your organization’s doorstep that your analytics platform cannot handle it.
- …

Critical Success Factors for Big Data Analytics

- A clear business need (alignment with the vision and the strategy)
- Strong, committed sponsorship (executive champion)
- Alignment between the business and IT strategy
- A fact-based decision-making culture
- A strong data infrastructure
- The right analytics tools
- **Right people with right skills**

Critical Success Factors for Big Data Analytics

Main providers of BD as-a-service

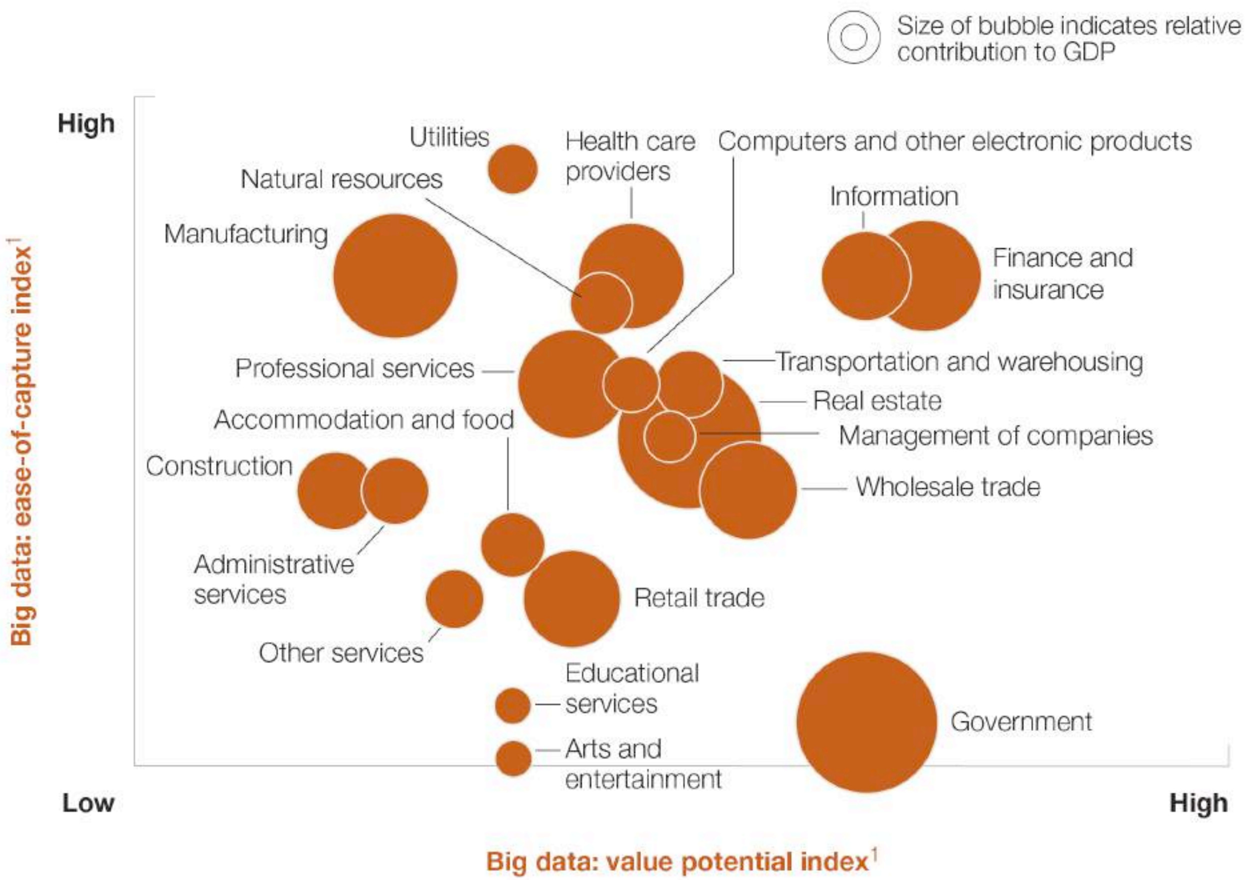
- **Google Cloud Dataproc**
- **Amazon Web Services**
- **Microsoft Azure HDInsight**
- **Salesforce Wave Analytics**
- **Qubole Data Service**
- **IBM BigInsights on Cloud**

Main BD computing frameworks

- Hadoop
- Spark
- Flink
- Storm
- Samza
- See: https://www.kdnuggets.com/2016/03/top-big-data-processing-frameworks.html

Market forecasts

Market, by service type



[1]For detailed explication of metrics, see appendix in McKinsey Global Institute full report *Big data: The next frontier for innovation, competition, and productivity*, available free of charge online at mckinsey.com/mgi.

Source: US Bureau of Labor Statistics; McKinsey Global Institute analysis

Value of BD job titles