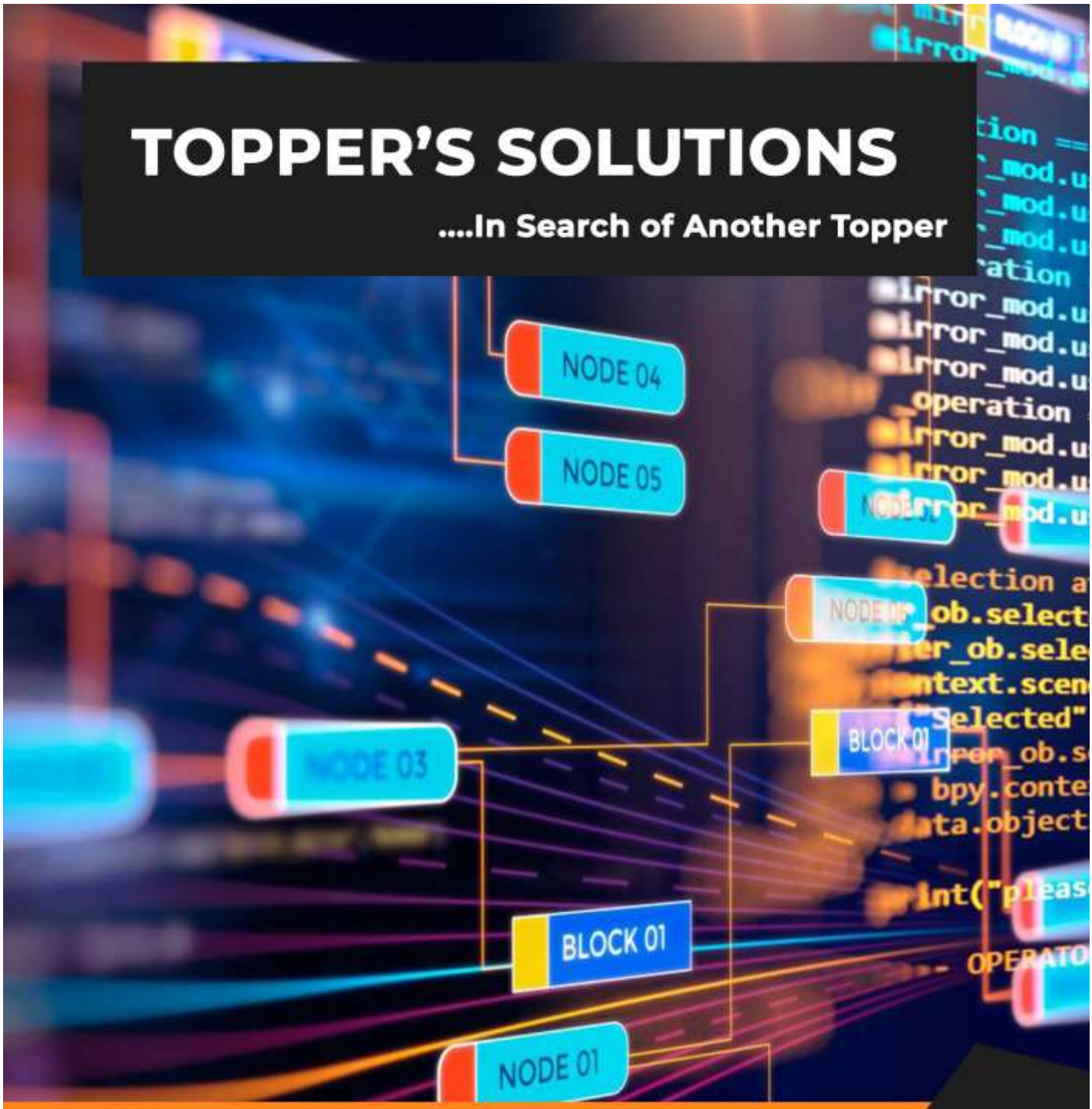
# BDA Notes for the year 2021-22 ( Mumbai University )

Big Data Analytics (University of Mumbai)




# BIG DATA ANALYTICS

## (BE - COMPUTER)

7 SEM

As per Revised Syllabus w.e.f 2019-20

Jul 2019 Edition

# TOPPER'S SOLUTIONS

**....In Search of Another Topper**

There are many existing paper solution available in market, but Topper's Solution is the one which students will always prefer if they refer... ;) Topper's Solutions is not just paper solutions, it includes many other important questions which are important from examination point of view. Topper's Solutions are the solution written by the Toppers for the students to be the upcoming Topper of the Semester.

It has been said that "**Action Speaks Louder than Words**" So Topper's Solutions Team works on same principle. Diagrammatic representation of answer is considered to be easy & quicker to understand. So our major focus is on diagrams & representation how answers should be answered in examinations.

**Why Topper's Solutions:**

- Point wise answers which are easy to understand & remember.
- Diagrammatic representation for better understanding.
- Additional important questions from university exams point of view.
- Covers almost every important question.
- In search of another topper.

**"Education is Free.... But its Technology used & Efforts utilized which we charge"**

It takes lot of efforts for searching out each & every question and transforming it into Short & Simple Language. Entire Community is working out for betterment of students, do help us.

**Thanks for Purchasing & Best Luck for Exams**

**---- In Association with BackkBenchers Community ----**


BACKK BENCHERS

Practice like you never **WON**.

Perform like you never **LOST**.

---- By Anonymous.



**Syllabus:**

| Exam | TT-1 | TT-2 | AVG | Term Work | Oral/Practical | End of Exam | Total |
|---|---|---|---|---|---|---|---|
| Marks | 20 | 20 | 20 | 25 | 25 | 80 | 150 |

| # | Module | Details Contents | No. |
|---|---|---|---|
| 1. | Introduction to Big Data and Hadoop | ▪ Introduction to Big Data<br>▪ Big Data characteristics, types of Big Data<br>▪ Traditional vs. Big Data business approach<br>▪ Case Study of Big Data Solutions<br>▪ Concept of Hadoop<br>▪ Core Hadoop Components; Hadoop Ecosystem | 01 |
| 2. | Hadoop HDFS and MapReduce | ▪ Distributed File Systems: Physical Organization of Compute Nodes, Large-Scale File-System Organization.<br>▪ MapReduce: The Map Tasks, Grouping by Key, The Reduce Tasks, Combiners, Details of MapReduce Execution, Coping With Node Failures.<br>▪ Algorithms Using MapReduce: Matrix-Vector Multiplication by MapReduce, Relational-Algebra Operations, Computing Selections by MapReduce, Computing Projections by MapReduce, Union, Intersection, and Difference by MapReduce<br>▪ Hadoop Limitations | 17 |
| 3. | NoSQL | ▪ Introduction to NoSQL, NoSQL Business Drivers<br>▪ NoSQL Data Architecture Patterns: Key-value stores, Graph stores, Column family (Bigtable) stores, Document stores, Variations of NoSQL architectural patterns, NoSQL Case Study<br>▪ NoSQL solution for big data, Understanding the types of big data problems; Analyzing big data with a shared-nothing architecture; Choosing distribution models: master-slave versus peer-to-peer; NoSQL systems to handle big data problems. | 25 |
| 4. | Mining Data Streams | ▪ The Stream Data Model: A Data-Stream-Management System, Examples of Stream Sources, Stream Queries, Issues in Stream Processing.<br>▪ Sampling Data techniques in a Stream<br>▪ Filtering Streams: Bloom Filter with Analysis.<br>▪ Counting Distinct Elements in a Stream, Count-Distinct Problem, Flajolet-Martin Algorithm, Combining Estimates, Space Requirements<br>▪ Counting Frequent Items in a Stream, Sampling Methods for Streams, Frequent Itemsets in Decaying Windows.<br>▪ Counting Ones in a Window: The Cost of Exact Counts, The Datar-Gionis-Indyk-Motwani Algorithm, Query Answering in the DGIM Algorithm, Decaying Windows. | 33 |
| 5. | Finding Similar Items and Clustering | ▪ Distance Measures: Definition of a Distance Measure, Euclidean Distances, Jaccard Distance, Cosine Distance, Edit Distance, Hamming Distance.<br>▪ CURE Algorithm, Stream-Computing , A Stream-Clustering Algorithm, Initializing & Merging Buckets, Answering Queries | 40 |
| 6. | Real-Time Big Data Models | ▪ PageRank Overview, Efficient computation of PageRank: PageRank Iteration Using MapReduce, Use of Combiners to Consolidate the Result Vector.<br>▪ A Model for Recommendation Systems, Content-Based Recommendations, Collaborative Filtering.<br>▪ Social Networks as Graphs, Clustering of Social-Network Graphs, Direct Discovery of Communities in a social graph. | 48 |

## CHAP - 1: INTRODUCTION T BIG DATA & HADOOP

**Q1. Give difference between Traditional data management and analytics approach Versus Big data approach**

**Ans:** **[5M – DEC 19]**

**COMPARISON BETWEEN TRADITIONAL DATA MANAGEMENT AND ANALYTICS APPROACH & BIG DATA APPROACH:**

| Traditional data management and analytics approach | Big data approach |
|---|---|
| Traditional database system deals with structured data. | Big data system deals with structured, semi structured and unstructured data. |
| Traditional data is generated in enterprise level. | Big data is generated in outside and enterprise level. |
| Its volume ranges from Gigabytes to Terabytes | Its volume ranges from Petabytes to Zettabytes or Exabytes. |
| Data integration is very easy. | Data integration is very difficult. |
| The size of the data is very small. | The size is more than the traditional data size |
| Its data model is strict schema based and it is static. | Its data model is flat schema based and it is dynamic. |
| It is easy to manage and manipulate the data. | It is difficult to manage and manipulate the data. |
| Its data sources includes ERP transaction data, CRM transaction data, financial data, organizational data, web transaction data etc. | Its data sources includes social media, device data, sensor data, video, images, audio etc. |
| The sample from known population is considered as object of analysis. | Entire population is considered as object of analysis. |
| Normal functions can manipulate data. | Special kind of functions can manipulate data. |
| Traditional data base tools are required to perform any data base operation. | Special kind of data base tools are required to perform any data base operation. |
| Traditional data source is centralized and it is managed in centralized form. | Big data source is distributed and it is managed in distributed form. |

**-- EXTRA QUESTIONS --**

**Q1. Explain Big Data & Types of Big Data**

**Ans:** **[P | High]**

**BIG DATA:**

1. Data is defined as the quantities, characters, or symbols on which operations are performed by a computer.
2. Data may be stored and transmitted in the form of electrical signals and recorded on magnetic, optical, or mechanical recording media.
3. Big Data is also **data** but with a **huge size**.
4. Big Data is a term used to describe a collection of data that is huge in size and yet growing exponentially with time.
5. In short such data is so large and complex that none of the traditional data management tools are able to store it or process it efficiently.
6. **Examples:**
   a. The **New York Stock Exchange** generates about **one terabyte** of new trade data per day.
   b. The statistic shows that **500+ terabytes** of new data get ingested into the databases of social media site **Facebook**, every day. This data is mainly generated in terms of photo and video uploads, message exchanges, putting comments etc.

**TYPES:**

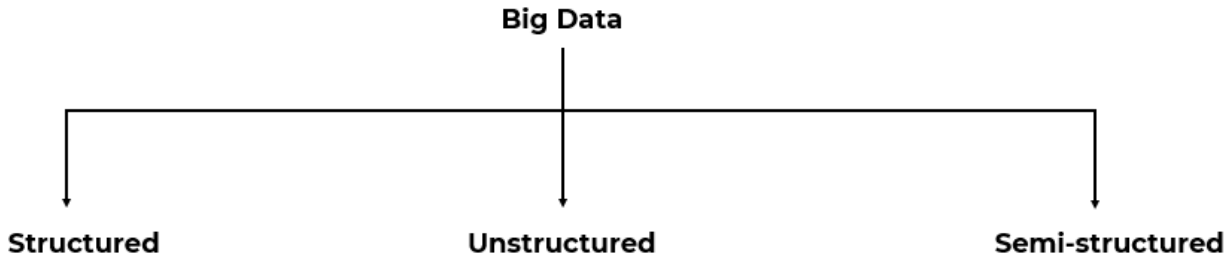



**Figure 1.1: Types of Big Data**

**I) Structured:**

1. Any data that can be stored, accessed and processed in the form of fixed format is termed as a **Structured Data**.
2. It accounts for about 20% of the total existing data and is used the most in programming and computer-related activities.
3. There are two sources of structured data - machines and humans.
4. All the data received from sensors, weblogs, and financial systems are classified under machine-generated data.
5. These include medical devices, GPS data, data of usage statistics captured by servers and applications.
6. Human-generated structured data mainly includes all the data a human input into a computer, such as his name and other personal details.
7. When a person clicks a link on the internet, or even makes a move in a game, data is created.
8. **Example:** An 'Employee' table in a database is an example of Structured Data.

| Employee_ID | Employee_Name | Gender |
|---|---|---|

| | | |
|---|---|---|
| 420 | Angel Priya | Male |
| 100 | Babu Bhaiya | Male |
| 202 | Babita Ji | Female |
| 400 | Jethalal Tapu Ke Papa Gada | Male |
| 007 | Dhinchak Pooja | Female |

9. Tools generates structured data:
   a. Data Marts
   b. RDBMS
   c. Greenplum
   d. TeraData

**II) Unstructured:**

1. Any data with unknown form or the structure is classified as **unstructured data**.
2. The rest of the data created, about 80% of the total account for unstructured big data.
3. Unstructured data is also classified based on its source, into machine-generated or human-generated.
4. Machine-generated data accounts for all the satellite images, the scientific data from various experiments and radar data captured by various facets of technology.
5. Human-generated unstructured data is found in abundance across the internet since it includes social media data, mobile data, and website content.
6. This means that the pictures we upload to Facebook or Instagram handle, the videos we watch on YouTube and even the text messages we send all contribute to the gigantic heap that is unstructured data.
7. Examples of unstructured data include text, video, audio, mobile activity, social media activity, satellite imagery, surveillance imagery etc.
8. The Unstructured data is further divided into:
   **a. Captured data:**
      - It is the data based on the user's behavior.
      - The best example to understand it is GPS via smartphones which help the user each and every moment and provides a real-time output.

   **b. User-generated data:**
      - It is the kind of unstructured data where the user itself will put data on the internet every movement.
      - For example, Tweets and Re-tweets, Likes, Shares, Comments, on YouTube, Facebook, etc.
9. Tools generates unstructured data:
   a. Hadoop
   b. HBase
   c. Hive
   d. Pig
   e. MapR
   f. Cloudera

**III) Semi-Structured:**

1. Semi Structured data is information that does not reside in a RDBMS.
2. Information that is not in the traditional database format as structured data, but contains some organizational properties which make it easier to process, are included in semi-structured data.
3. It may organized in tree pattern which is easier to analyze in some cases.
4. Examples of semi structured data might include XML documents and NoSQL databases.
   Personal data stored in an XML file

```
<rec><name>Angel Priya</name><sex>Male</sex></rec>
<rec><name>Babu Bhaiya</name><sex>Male</sex></rec>
<rec><name>Babita Ji</name><sex>Female</sex></rec>
<rec><name>Jethalal Tapu Ke Papa Gada</name><sex>Male</sex></rec>
<rec><name>Dhinchak Pooja</name><sex>Female</sex></rec>
```

---

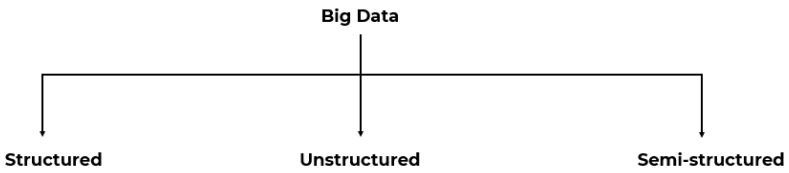
**Q2. Explain Characteristics of Big Data or Define the three V's of Big Data**

**Ans:** **[P | High]**

## CHARACTERISTICS OF BIG DATA:

### I) Variety:

1. Variety of Big Data refers to structured, unstructured, and semi structured data that is gathered from multiple sources.
2. The type and nature of data is having great variety.
3. During earlier days, spreadsheets and databases were the only sources of data considered by most of the applications.
4. Nowadays, data in the form of emails, photos, videos, monitoring devices, PDFs, audio, etc. are also being considered in the analysis applications.




### II) Velocity:

1. The term **velocity** refers to the speed of generation of data.
2. Big Data Velocity deals with the speed at which data flows in from sources like business processes, application logs, networks, and social media sites, sensors, Mobile devices, etc.
3. The flow of data is massive and continuous.
4. The speed of data accumulation also plays a role in determining whether the data is categorized into big data or normal data.
5. As can be seen from the figure 1.2 below, at first, mainframes were used wherein fewer people used computers.
6. Then came the client/server model and more and more computers were evolved.
7. After this, the web applications came into the picture and started increasing over the Internet.

8. Then, everyone began using these applications.
9. These applications were then used by more and more devices such as mobiles as they were very easy to access. Hence, a lot of data!

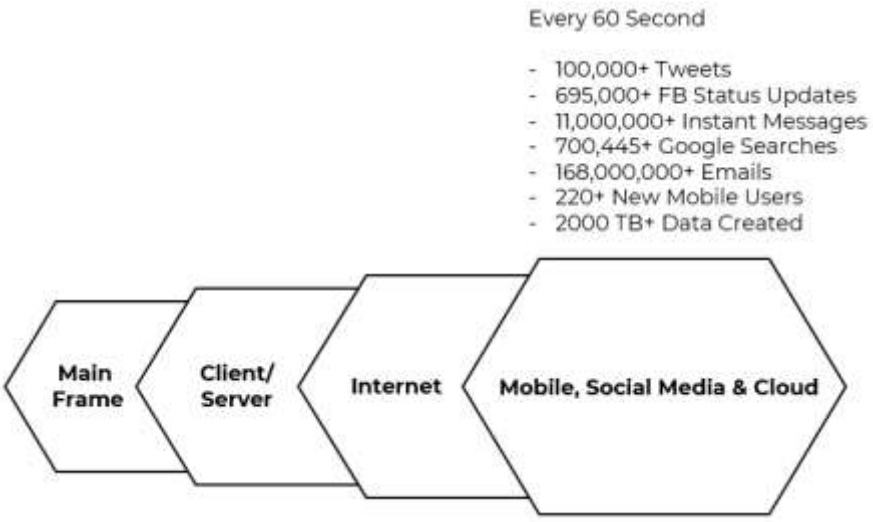



**Figure 1.2: Big Data Velocity**

### III) Volume:

1. The name Big Data itself is related to a size which is enormous.
2. Size of data plays a very crucial role in determining value out of data.
3. Also, whether a particular data can actually be considered as a Big Data or not, is dependent upon the volume of data.
4. Hence, **'Volume'** is one characteristic which needs to be considered while dealing with Big Data.
5. This refers to the data that is tremendously large.
6. As shown in figure 1.3 below, the volume of data is rising exponentially.
7. In 2016, the data created was only 8 ZB and it is expected that, by 2020, the data would rise up to 40 ZB, which is extremely large.

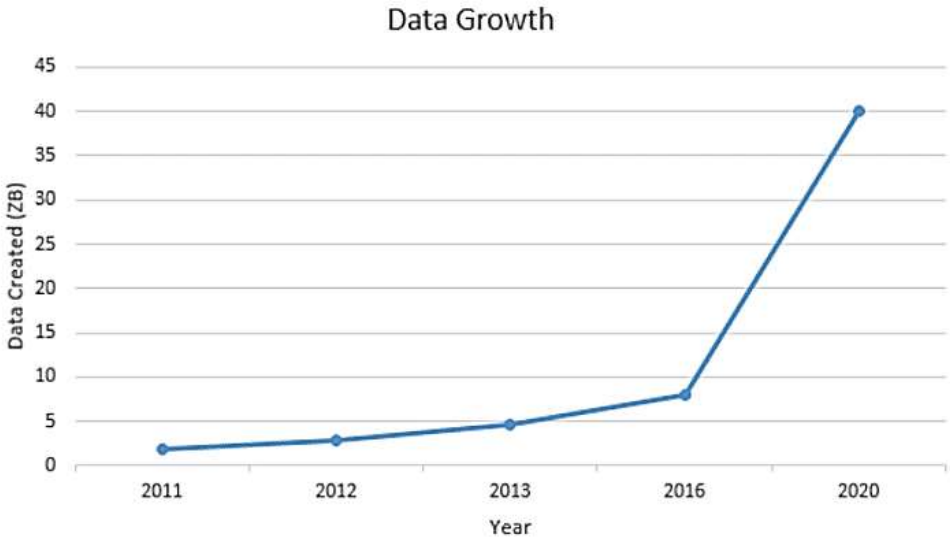



**Figure 1.3: Big Data Volume**

### OTHER CHARACTERISTICS OF BIG DATA:

**I) Programmable:**

1. It is possible with big data to explore all types by programming logic.
2. Programming can be used to perform any kind of exploration because of the scale of the data.

**II) Data Driven:**

1. The data driven approach is possible for scientists.
2. As data collected is huge amount.

**III) Multi Attributes:**

1. It is possible to deal with many gigabytes of data that consist of thousands of attributes.
2. As all data operations are now happening on a larger scale.

**IV) Veracity:**

1. The data captured is not in certain format.
2. Data captured can vary greatly.
3. Veracity means the trustworthiness and quality of data.
4. It is necessary that the veracity of the data is maintained.
5. For example, think about Facebook posts, with hashtags, abbreviations, images, videos, etc., which make them unreliable and hamper the quality of their content.
6. Collecting loads and loads of data is of no use if the quality and trustworthiness of the data is not up to the mark.

---

**Q4. Applications of Big Bata**

**Ans:** **[P | Medium]**

**APPLICATIONS OF BIG BATA:**

**I) Healthcare & Public Health Industry:**

1. Big Data has already started to create a huge difference in the healthcare sector.
2. With the help of predictive analytics, medical professionals and HCPs are now able to provide personalized healthcare services to individual patients.
3. Like entire DNA strings can be decoded in minutes.
4. Apart from that, fitness wearable's, telemedicine, remote monitoring – all powered by Big Data and AI – are helping change lives for the better.

**II) Academia**

1. Big Data is also helping enhance education today.
2. Education is no more limited to the physical bounds of the classroom – there are numerous online educational courses to learn from.
3. Academic institutions are investing in digital courses powered by Big Data technologies to aid the all-round development of budding learners.

**III) Banking**

1. The banking sector relies on Big Data for fraud detection.
2. Big Data tools can efficiently detect fraudulent acts in real-time such as misuse of credit/debit cards, archival of inspection tracks, faulty alteration in customer stats, etc.

**IV) Manufacturing**

1. According to TCS Global Trend Study, the most significant benefit of Big Data in manufacturing is improving the supply strategies and product quality.
2. In the manufacturing sector, Big data helps create a transparent infrastructure, thereby, predicting uncertainties and incompetence's that can affect the business adversely.

**V) IT**

1. One of the largest users of Big Data, IT companies around the world are using Big Data to optimize their functioning, enhance employee productivity, and minimize risks in business operations.
2. By combining Big Data technologies with ML and AI, the IT sector is continually powering innovation to find solutions even for the most complex of problems.

---

**Q4. Write short notes on Hadoop**

**Ans:** **[P | Medium]**

**HADOOP:**

1. Hadoop is an open source software programming framework for storing a large amount of data and performing the computation.
2. Its framework is based on Java programming with some native code in C and shell scripts.
3. **Apache Software Foundation** is the developers of Hadoop, and its co-founders are **Doug Cutting** and **Mike Cafarella**.
4. The Hadoop framework application works in an environment that provides distributed storage and computation across clusters of computers.
5. Hadoop is designed to scale up from single server to thousands of machines, each offering local computation and storage.

**FEATURES:**

1. Low Cost
2. High Computing Power
3. Scalability
4. Huge & Flexible Storage
5. Fault Tolerance & Data Protection

**HADOOP ARCHITECTURE:**

1. Figure 1.4 shows architecture of Hadoop.
2. At its core, Hadoop has two major layers namely:
   a. Processing/Computation layer (MapReduce), and
   b. Storage layer (Hadoop Distributed File System).

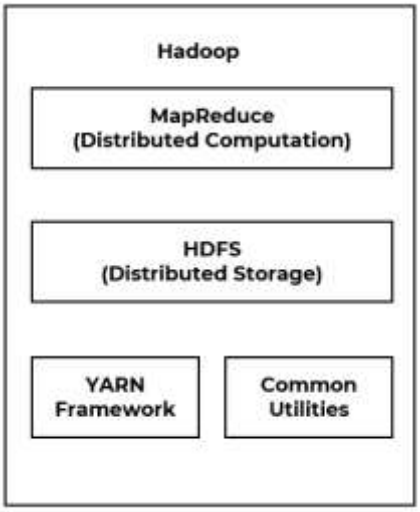



**Figure 1.4: Hadoop Architecture**

**MapReduce:**

1. MapReduce is a parallel programming model for writing distributed applications.
2. It is used for efficient processing of large amounts of data (multi-terabyte data-sets), on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner.
3. The MapReduce program runs on Hadoop which is an Apache open-source framework.

**Hadoop Distributed File System:**

1. The Hadoop Distributed File System (HDFS) is based on the Google File System (GFS)
2. It provides a distributed file system that is designed to run on commodity hardware.
3. It has many similarities with existing distributed file systems.
4. However, the differences from other distributed file systems are significant.
5. It is highly fault-tolerant and is designed to be deployed on low-cost hardware.
6. It provides high throughput access to application data and is suitable for applications having large datasets.

**Hadoop framework also includes the following two modules:**

1. **Hadoop Common:** These are Java libraries and utilities required by other Hadoop modules.
2. **Hadoop YARN:** This is a framework for job scheduling and cluster resource management.

**ADVANTAGES:**

1. Ability to store a large amount of data.
2. High flexibility.
3. Cost effective.
4. High computational power.
5. Tasks are independent.
6. Linear scaling.

**DISADVANTAGES:**

1. Not very effective for small data.

2. Hard cluster management.
3. Has stability issues.
4. Security concerns.

**Q5. Write physical architecture of Hadoop**

**Ans:** **[P | Medium]**

**PHYSICAL ARCHITECTURE OF HADOOP:**

1. Hadoop is an open source software framework which provides huge data storage.
2. Running Hadoop means running a set of resident programs.
3. These resident programs are also known as daemons.
4. These daemons may be running on the same server or on the different servers in the network.
5. Figure 1.5 shows Hadoop cluster topology.

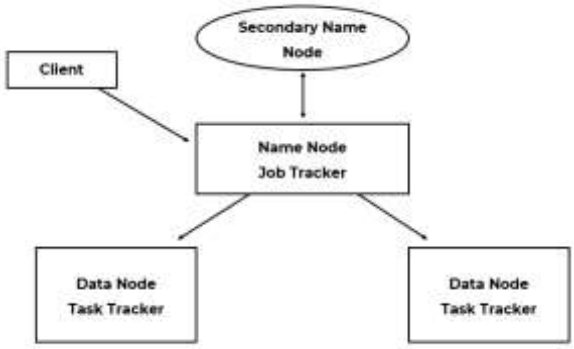



**Figure 1.5: Hadoop Cluster Topology**

**WORKING:**

1. When the client will submit his job, it will go to the name node.
2. Now name node will decide whether to accept the job or not.
3. After accepting the job, the name node will transfer the job to the job tracker.
4. Then the job tracker will divide the job into components and transfer them to data nodes.
5. Now data nodes will further transfer the jobs to the task tracker.
6. Now the actual processing will be done here, means the execution of the job submitted is done here.
7. Now, after completing the part of the jobs assigned to them, the task tracker will submit the completed task to the job tracker via the data node.
8. Now, coming on secondary name node, the task of secondary name node is to just monitor the whole process ongoing.
9. Now, **physical architecture of Hadoop is a Master-slave process**, here name node is a master, job tracker is a part of master and data nodes are the slaves.

**COMPONENTS:**

**I) Name Node:**

1. It is the master of HDFS (Hadoop file system).
2. It contains Job Tracker, which keeps tracks of a file distributed to different data nodes.
3. Name Node directs Data Node regarding the low level I/O tasks.

4. Failure of Name Node will lead to the failure of the full Hadoop system.

**II) Data node:**

1. Data node is the slave of HDFS.
2. A data node can communicate with each other through the name node to avoid replication in the provided task.
3. For replication of data a data node may communicates with other data nodes.
4. Data node continually informs local change updates to name nodes.
5. To create, move or delete blocks, data node receives instructions from the local disk.

**III) Job Tracker:**

1. Job Tracker determines which file to process.
2. There can be only one job tracker for per Hadoop cluster.
3. Job Tracker runs on a server as a master node of the cluster.

**IV) Task Tracker:**

1. Only single task tracker is present per slave node.
2. Task Tracker performs tasks given by job tracker and also continuously communicates with the job tracker.

**V) SSN (Secondary Name Node)**

1. Its main purpose is to monitor.
2. State Monitoring of cluster HDFS is done by SNN.
3. SNN resides on its own machine also.
4. One SSN is present per cluster.

---

**Q6. Write short notes on Core Hadoop Components**

**Ans:** **[P | High]**

**CORE HADOOP COMPONENTS:**

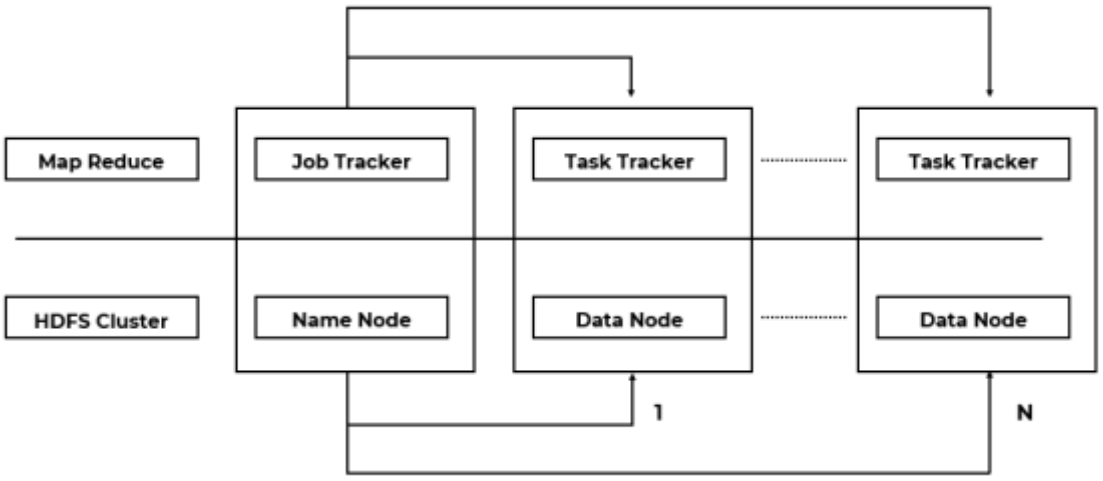



**Figure 1.6: Hadoop Core Components**

1. Hadoop has a **master-slave topology**.
2. In this topology, we have one master node and multiple slave nodes.
3. Master node's function is to assign a task to various slave nodes and manage resources. The slave nodes do the actual computing.
4. Slave nodes store the real data whereas on master we have metadata.
5. Figure 1.6 shows Hadoop core components.

## HADOOP DISTRIBUTED FILE SYSTEM (HDFS):

1. HDFS is a file system for Hadoop.
2. HDFS is based on Google File System (GFS).
3. It runs on clusters on commodity hardware.
4. The file system has several similarities with the existing distributed file systems.

### Characteristics:

1. High Fault Tolerant.
2. High throughput.
3. Supports application with massive datasets.
4. Streaming access to file system data.
5. Can be built out of commodity hardware.

### Architecture:

Figure 1.7 shows HDFS Architecture.

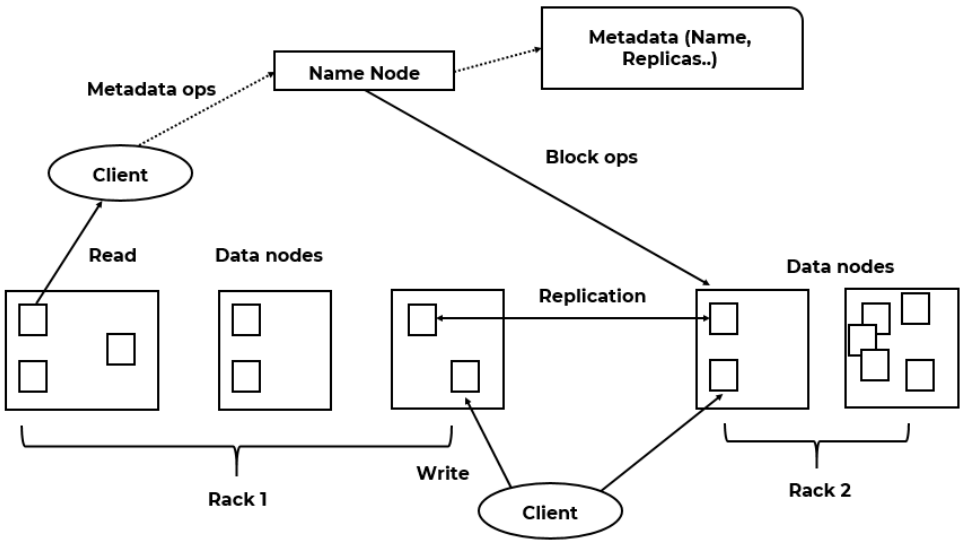



**Figure 1.7: HDFS Architecture**

HDFS follows the master-slave architecture and it has the following elements.

### I) Namenode:

1. It is a deamon which runs on master node of hadoop cluster.
2. There is only one namenode in a cluster.
3. It contains metadata of all the files stored on HDFS which is known as namespace of HDFS.
4. It maintain two files i.e. Edit Log & FsImage.

5. EditLog is used to record every change that occurs to file system metadata (transaction history)
6. FsImage stores entire namespace, mapping of blocks to files and file system properties.
7. The FsImage and the EditLog are central data structures of HDFS.
8. The system having the namenode acts as the master server and it does the following tasks:
   a. Manages the file system namespace.
   b. Regulates client's access to files.
   c. It also executes file system operations such as renaming, closing, and opening files and directories.

**II) Datanode:**

1. It is a deamon which runs on slave machines of Hadoop cluster.
2. There are number of datanodes in a cluster.
3. It is responsible for serving read/write request from the clients. It also performs block creation, deletion, and replication upon instruction from the Namenode.
4. It also sends a Heartbeat message to the namenode periodically about the blocks it hold.
5. Namenode and Datanode machines typically run a GNU/Linux operating system (OS).

**III) Block:**

1. Generally the user data is stored in the files of HDFS.
2. The file in a file system will be divided into one or more segments and/or stored in individual data nodes.
3. These file segments are called as blocks.
4. In other words, the minimum amount of data that HDFS can read or write is called a Block.
5. The default block size is 64MB, but it can be increased as per the need to change in HDFS configuration.

**MAPREDUCE:**

1. MapReduce is a **software framework**.
2. MapReduce is the data processing layer of Hadoop.
3. It is a software framework that allows you to write applications for processing a large amount of data.
4. MapReduce runs these applications in parallel on a cluster of low-end machines.
5. It does so in a reliable and fault-tolerant manner.
6. In MapReduce an application is broken down into number of small parts.
7. These small parts are also called as fragments or blocks.
8. These blocks then can be run on any node in the cluster.
9. Data Processing is done by MapReduce.
10. MapReduce scales and runs an application to different clutter machines.
11. There are two primitives used for data processing by MapReduce known as **Mappers & Reducers**.
12. MapReduce use lists and key/value pairs for processing of data.

**MapReduce Core Functions:**

**I) Read Input:**

1. It divides input into small blocks.
2. These blocks then get assigned to a Map function.

**II) Function Mapping:**

1. It converts file data to smaller, intermediate <key, value> pairs.

**III) Partition, Compare & Sort:**

a. **Partition Function:** With the given key and number of reducers it finds the correct reducer.

b. **Compare Function:** Map intermediate outputs are sorted according to this compare function.

**IV) Function Reducing:**

1. Intermediate values are reduced to smaller solutions and given to output.

**V) Write Output:**

Gives file output

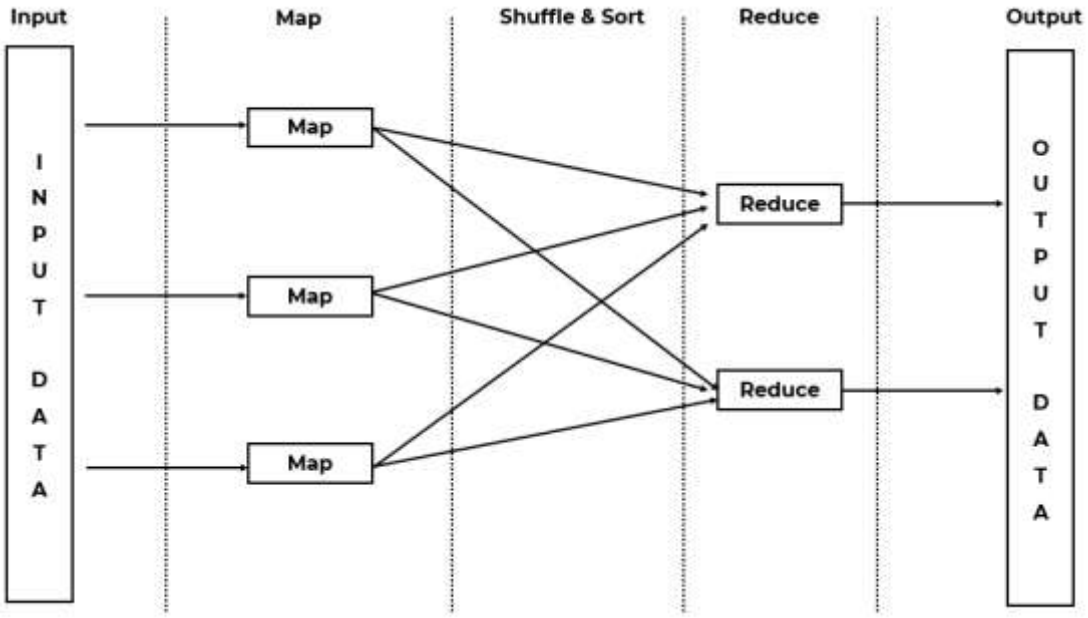



**Figure 1.8: General MapReduce DataFlow**

**Example:**

File 1: "Hello Babita Hello Jethalal"

File 2: "Goodnight Babita Goodnight Jethalal"

**Operations:**

**(1) Map:**

| Map 1 | Map 2 |
|---|---|
| <Hello, 1> | <Goodnight, 1> |
| <Babita, 1> | <Babita, 1> |
| <Hello, 1> | <Goodnight, 1> |
| <Jethalal, 1> | <Jethalal, 1> |

**(2) Combine:**

| Combine Map 1 | Combine Map 2 |
|---|---|
| <Babita, 1> | <Babita, 1> |
| <Jethalal, 1> | <Jethalal, 1> |
| <Hello, 2> | <Goodnight, 2> |

**(3) Reduce**

<Babita, 2>
<Jethalal, 2>
<Goodnight, 2>
<Hello, 2>

---

**Q7. Explain Hadoop Ecosystem**

**Ans:** **[P | Medium]**

**HADOOP ECOSYSTEM:**

1. Core Hadoop ecosystem is nothing but the different components that are built on the Hadoop platform directly.
2. Figure 1.9 represents Hadoop Ecosystem.

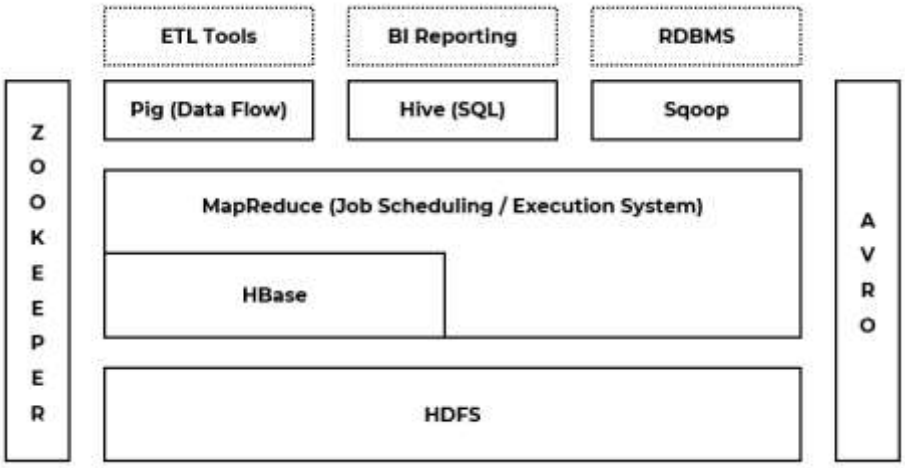



**Figure 1.9: Hadoop Ecosystem.**

**I) Hadoop Distributed File System (HDFS):**

1. **HDFS is the foundation of Hadoop** and hence is a very important component of the Hadoop ecosystem.
2. It is Java software that provides many features like scalability, high availability, fault tolerance, cost effectiveness etc.
3. It also provides robust distributed data storage for Hadoop.
4. We can deploy many other software frameworks over HDFS.

**II) MapReduce:**

1. **MapReduce** is the data processing component of Hadoop.
2. It applies the computation on sets of data in parallel thereby improving the performance.
3. MapReduce works in two phases:
   a. **Map Phase:** This phase takes input as key-value pairs and produces output as key-value pairs. It can write custom business logic in this phase. Map phase processes the data and gives it to the next phase.
   b. **Reduce Phase:** The MapReduce framework sorts the key-value pair before giving the data to this phase. This phase applies the summary type of calculations to the key-value pairs.

### III) Hive:

1. Hive is a data warehouse project built on the top of Apache Hadoop which provides data query and analysis.
2. It has got the language of its own call HQL or **Hive Query Language**.
3. HQL automatically translates the queries into the corresponding map-reduce job.
4. Main parts of the Hive are –
   a. **MetaStore:** It stores metadata
   b. **Driver:** Manages the lifecycle of HQL statement
   c. **Query Compiler:** Compiles HQL into DAG i.e. Directed Acyclic Graph
   d. **Hive Server:** Provides interface for JDBC/ODBC server.

### IV) Pig:

1. Pig is a SQL like language used for querying and analyzing data stored in HDFS.
2. Yahoo was the original creator of the Pig.
3. It uses pig latin language.
4. It loads the data, applies a filter to it and dumps the data in the required format.
5. Pig also consists of JVM called Pig Runtime. Various **features of Pig** are as follows:-
   a. **Extensibility:** For carrying out special purpose processing, users can create their own custom function.
   b. **Optimization opportunities:** Pig automatically optimizes the query allowing users to focus on semantics rather than efficiency.
   c. **Handles all kinds of data:** Pig analyzes both structured as well as unstructured.

### V) HBase:

1. HBase is a NoSQL database built on the top of HDFS.
2. The various **features of** HBase are that it is open-source, non-relational, distributed database.
3. It imitates **Google's Bigtable** and written in Java.
4. It provides real-time read/write access to large datasets.

### VI) Zookeeper:

1. Zookeeper coordinates between various services in the Hadoop ecosystem.
2. It saves the time required for synchronization, configuration maintenance, grouping, and naming.
3. Following are the **features of Zookeeper**:
   a. **Speed:** Zookeeper is fast in workloads where reads to data are more than write. A typical read: write ratio is 10:1.
   b. **Organized:** Zookeeper maintains a record of all transactions.
   c. **Simple:** It maintains a single hierarchical namespace, similar to directories and files.
   d. **Reliable:** We can replicate Zookeeper over a set of hosts and they are aware of each other. There is no single point of failure. As long as major servers are available zookeeper is available.

### VII) Sqoop:

1. Sqoop imports data from external sources into compatible Hadoop Ecosystem components like HDFS, Hive, HBase etc.
2. It also transfers data from Hadoop to other external sources.

3. It works with RDBMS like TeraData, Oracle, MySQL and so on.
4. The major difference between Sqoop and Flume is that Flume does not work with structured data.
5. But Sqoop can deal with structured as well as unstructured data.

# CHAP - 2: HADOOP HDFS & MAPREDUCE

**Q1. What is Hadoop? Describe HDFS architecture with diagram**

**Ans:** **[10M – DEC 19]**

## HADOOP:

1. Hadoop is an open source software programming framework for storing a large amount of data and performing the computation.
2. Its framework is based on Java programming with some native code in C and shell scripts.
3. **Apache Software Foundation** is the developers of Hadoop, and its co-founders are **Doug Cutting** and **Mike Cafarella**.
4. The Hadoop framework application works in an environment that provides distributed storage and computation across clusters of computers.
5. Hadoop is designed to scale up from single server to thousands of machines, each offering local computation and storage.

## FEATURES OF HADOOP:

1. Low Cost
2. High Computing Power
3. Scalability
4. Huge & Flexible Storage
5. Fault Tolerance & Data Protection

## HADOOP DISTRIBUTED FILE SYSTEM (HDFS):

1. HDFS is a file system for Hadoop.
2. HDFS is based on Google File System (GFS).
3. It runs on clusters on commodity hardware.
4. The file system has several similarities with the existing distributed file systems.

## Characteristics:

1. High Fault Tolerant.
2. High throughput.
3. Supports application with massive datasets.
4. Streaming access to file system data.
5. Can be built out of commodity hardware.

## Architecture:

Figure 2.1 shows HDFS Architecture.

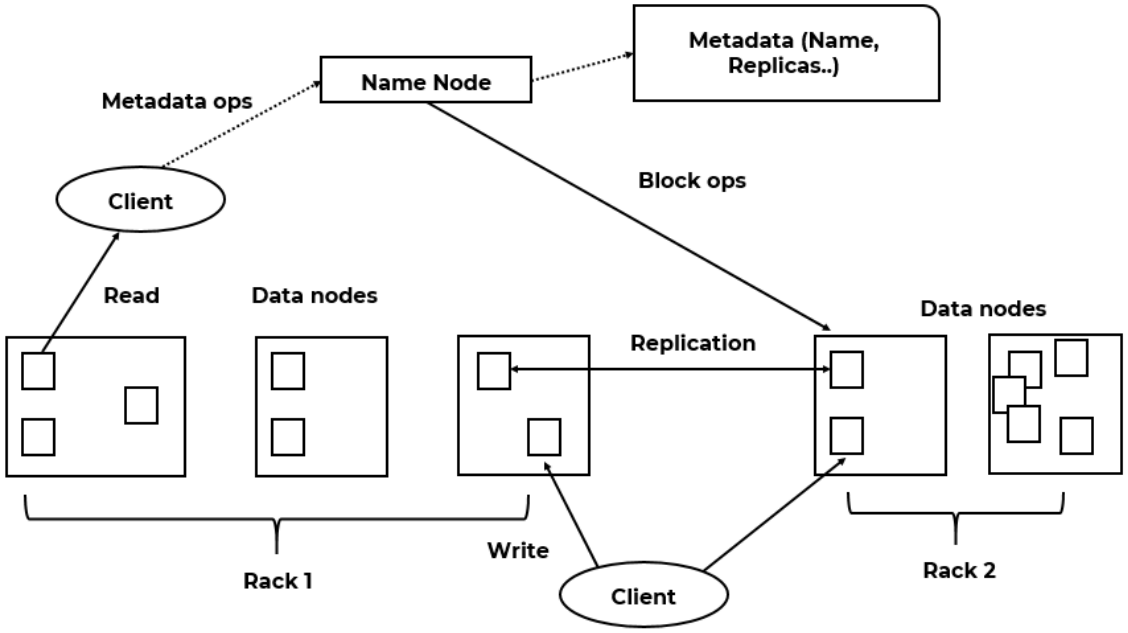



**Figure 2.1: HDFS Architecture**

HDFS follows the master-slave architecture and it has the following elements.

### I) Namenode:

1. It is a deamon which runs on master node of hadoop cluster.
2. There is only one namenode in a cluster.
3. It contains metadata of all the files stored on HDFS which is known as namespace of HDFS.
4. It maintain two files i.e. Edit Log & FsImage.
5. EditLog is used to record every change that occurs to file system metadata (transaction history)
6. FsImage stores entire namespace, mapping of blocks to files and file system properties.
7. The FsImage and the EditLog are central data structures of HDFS.
8. The system having the namenode acts as the master server and it does the following tasks:
   a. Manages the file system namespace.
   b. Regulates client's access to files.
   c. It also executes file system operations such as renaming, closing, and opening files and directories.

### II) Datanode:

1. It is a deamon which runs on slave machines of Hadoop cluster.
2. There are number of datanodes in a cluster.
3. It is responsible for serving read/write request from the clients. It also performs block creation, deletion, and replication upon instruction from the Namenode.
4. It also sends a Heartbeat message to the namenode periodically about the blocks it hold.
5. Namenode and Datanode machines typically run a GNU/Linux operating system (OS).

### III) Block:

1. Generally the user data is stored in the files of HDFS.
2. The file in a file system will be divided into one or more segments and/or stored in individual data nodes.
3. These file segments are called as blocks.

4. In other words, the minimum amount of data that HDFS can read or write is called a Block.
5. The default block size is 64MB, but it can be increased as per the need to change in HDFS configuration.

---

**Q2. What is the role of JobTracker and TaskTracker in MapReduce. Illustrate Map Reduce execution pipeline with Word count example**

**Ans:** **[10M – DEC 19]**

**MAPREDUCE:**

1. MapReduce is a **software framework**.
2. MapReduce is the data processing layer of Hadoop.
3. Similar to HDFS, MapReduce also exploits master/slave architecture in which JobTracker runs on master node and TaskTracker runs on each salve node.
4. Task Trackers are processes running on data nodes.
5. These monitors the maps and reduce tasks executed on the node and coordinates with Job tracker.
6. Job Tracker monitors the entire MR job execution.
7. JobTracker and TaskTracker are 2 essential process involved in MapReduce execution in MRv1 (or Hadoop version 1).
8. Both processes are now deprecated in MRv2 (or Hadoop version 2) and replaced by Resource Manager, Application Master and Node Manager Daemons.

**JOB TRACKER:**

1. JobTracker is an essential Daemon for MapReduce execution in MRv1.
2. It is replaced by Resource Manager / Application Master in **MRv2.**
3. JobTracker receives the requests for MapReduce execution from the client.
4. JobTracker talks to the NameNode to determine the location of the data.
5. JobTracker finds the best Task Tracker nodes to execute tasks based on the data locality (proximity of the data) and the available slots to execute a task on a given node.
6. JobTracker monitors the individual Task Trackers and the submits back the overall status of the job back to the client.
7. If a task fails, the JobTracker can reschedule it on a different TaskTrackers.
8. When the JobTracker is down, HDFS will still be functional but the MapReduce execution cannot be started and the existing MapReduce jobs will be halted.

**TASKTRACKER:**

1. TaskTracker runs on DataNode.
2. TaskTracker is replaced by Node Manager in MRv2.
3. Mapper and Reducer tasks are executed on DataNodes administered by TaskTrackers.
4. TaskTrackers will be assigned Mapper and Reducer tasks to execute by JobTracker.
5. TaskTracker will be in constant communication with the JobTracker signaling the progress of the task in execution.
6. TaskTracker failure is not considered fatal. When a TaskTracker becomes unresponsive, JobTracker will assign the task executed by the TaskTracker to another node.

## A WORD COUNT EXAMPLE OF MAPREDUCE:

1. Let us understand, how a MapReduce works by taking an example where I have a text file called example.txt whose contents are as follows:

**Dear, Bear, River, Car, Car, River, Deer, Car and Bear**

2. Now, suppose, we have to perform a word count on the sample.txt using MapReduce.
3. So, we will be finding the unique words and the number of occurrences of those unique words.

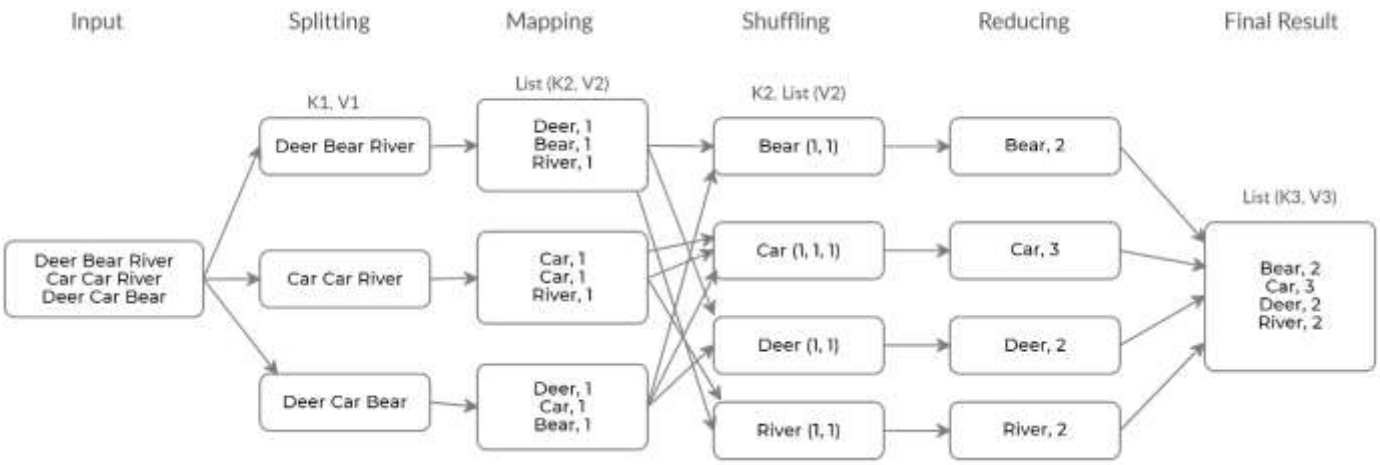



**Figure 2.2: A WORD COUNT EXAMPLE OF MAPREDUCE**

4. First, we divide the input into three splits as shown in the figure 2.2.
5. This will distribute the work among all the map nodes.
6. Then, we tokenize the words in each of the mappers and give a hardcoded value (1) to each of the tokens or words.
7. The rationale behind giving a hardcoded value equal to 1 is that every word, in itself, will occur once.
8. Now, a list of key-value pair will be created where the key is nothing but the individual words and value is one.
9. So, for the first line (Dear Bear River) we have 3 key-value pairs – Dear, 1; Bear, 1; River, 1.
10. The mapping process remains the same on all the nodes.
11. After the mapper phase, a partition process takes place where sorting and shuffling happen so that all the tuples with the same key are sent to the corresponding reducer.
12. So, after the sorting and shuffling phase, each reducer will have a unique key and a list of values corresponding to that very key. For example, Bear, [1,1]; Car, [1,1,1].., etc.
13. Now, each Reducer counts the values which are present in that list of values.
14. As shown in the figure 2.2, reducer gets a list of values which is [1,1] for the key Bear.
15. Then, it counts the number of ones in the very list and gives the final output as – Bear, 2.
16. Finally, all the output key/value pairs are then collected and written in the output file.

## -- EXTRA QUESTIONS --

**Q1. List out limitations of Hadoop**

**Ans:** **[P | Low]**

**HADOOP LIMITATION:**

1. Issue with small files.
2. Slow processing speed.
3. Latency.
4. No Real Time Data Processing.
5. Support for Batch Processing only.
6. No Caching.
7. Not Ease of Use.
8. No Delta Iteration.

---

**Q2. Explain Union, Intersection & Difference operation with MapReduce Techniques**

**Ans:** **[P | Medium]**

**UNION WITH MAPREDUCE:**

1. For union operation, map phase has responsibility of converting the tuple 't' values from Relation 'R' to a key value pair format.
2. Reducer has the task to assign the values to same key 't'
3. Figure 2.3 shows union operation with MapReduce Format.

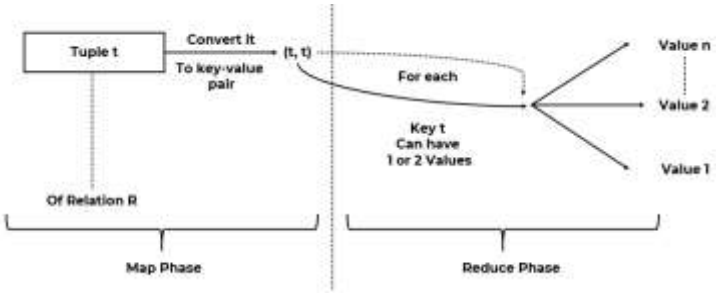



**Figure 2.3: Union Operation with MapReduce Format.**

**INTERSECTION WITH MAPREDUCE:**

1. For intersection operation, map phase has same responsibility as that of union operation.
2. That is to convert tuple 't' values from Relation 'R' to a key value pair format.
3. Reducer is responsible for generating output by evaluating if else condition.
4. That is then only output will be generated in key value format else no output will be produce.
5. Figure 2.4 shows intersection operation with MapReduce Format.

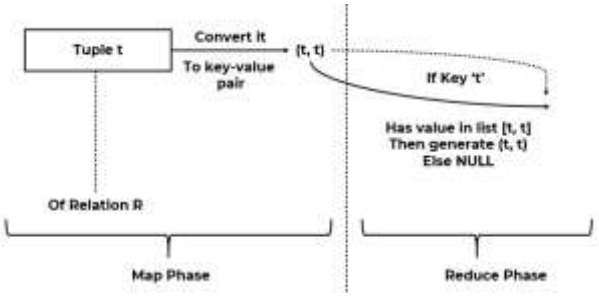



**Figure 2.4: Intersection Operation with MapReduce Format.**

**DIFFERENCE WITH MAPREDUCE:**

1. For difference operation, In Map phase consider two relations namely R & S.
2. Then the tuple 't' in Relation R and tuple 't' in Relation S will produce a key-value pair (t, R) and (t, S) for Relation R and Relation S respectively.
3. Then the output of map phase i.e. key-value pair will be submitted to reducer as an input.
4. Reducer will now generate the key-value pair ($t_1$, t) in generalized form for key 't' of every relation if and only if corresponding relation (R or S) has an associated collection of item i.e. list [R] else NULL will be output.
5. Figure 2.5 shows difference operation with MapReduce.

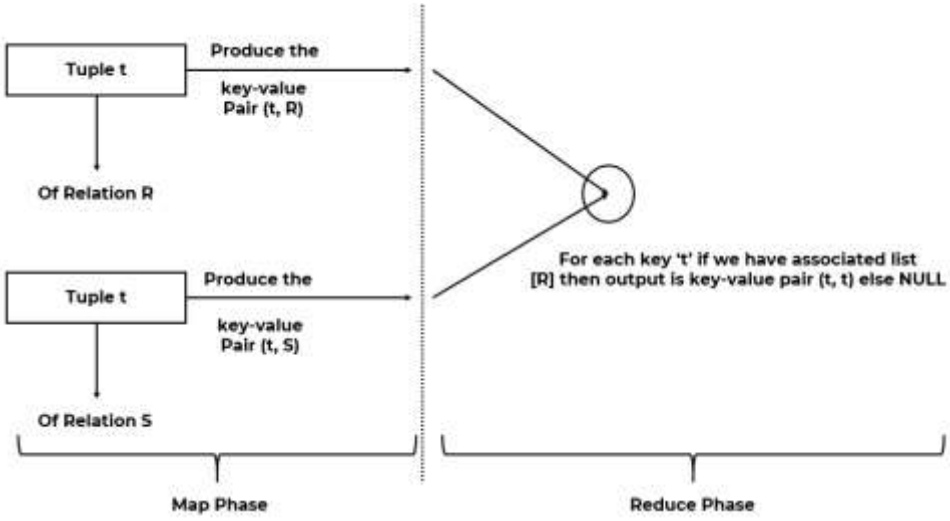



**Figure 2.5: Difference Operation with MapReduce Format.**

**Q3. Explain Matrix Vector Multiplication by MapReduce**

**Ans:** **[P | High]**

**MATRIX VECTOR MULTIPLICATION:**

1. Consider a Matrix 'M' of size n x n.
2. Let 'i' represents the rows and 'j' represents the columns.
3. Any element is Matrix will be represented as $M_{ij}$
4. Assume that there is an n-dimensional vector V, the first j elements denoted as $v_j$.
5. So, matrix M and the vector V is the result of an n-dimensional vector x.
6. The first 'i' elements $X_i$ is given as:

$$x_i = \sum_{j=0}^{n-1} m_{ij}\, v_j$$

7. If n = 100, we do not want to use a DFS or MapReduce for this calculation.
8. But such kind of calculations are required while maintaining the page ranking of web pages for a given search.
9. In real time, the value of n is in some billions.
10. Let us first assume that n is large, but not so large that vector v cannot fit in main memory and thus be available to every Map task.
11. The matrix M and the vector v each will be stored in a file of the DFS.
12. We assume that the row-column coordinates of each matrix element will be discoverable, either from its position in the file, or because it is stored with explicit coordinates, as a triple (i, j, mij).

13. We also assume the position of element vj in the vector v will be discoverable in the analogous way.
14. **The Map Function:**
    a. The Map function is written to apply to one element of Matrix M.
    b. However, if vector v is not already read into main memory at the compute node executing a Map task, then v is first read, in its entirety, and subsequently will be available to all applications of the Map function performed at this Map task.
    c. Each Map task will operate on a chunk of the matrix M.
    d. From each matrix element mij it produces the key-value pair (i, mij* vj).
    e. Thus, all terms of the sum that make up the component xi of the matrix-vector product will get the same key, i.
15. **The Reduce Function:**
    a. The Reduce function simply sums all the values associated with a given key i.
    b. The result will be a pair (i, xi).
    c. We can divide the matrix into vertical stripes of equal width and divide the vector into an equal number of horizontal stripes, of the same height.
    d. Our goal is to use enough stripes so that the portion of the vector in one stripe can fit conveniently into main memory at a compute node.
16. Figure 2.6 shows what the partition looks like if the matrix and vector are each divided into four stripes.

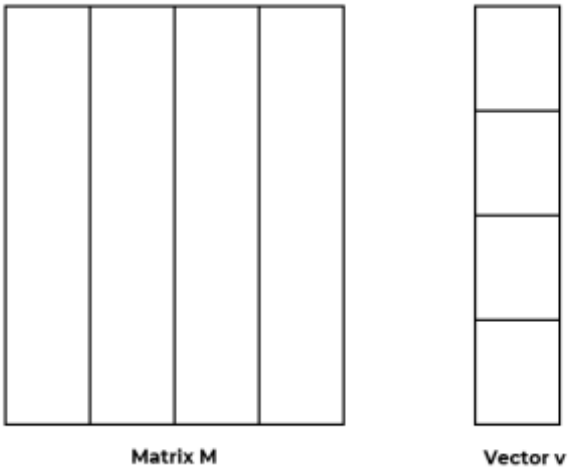



**Figure 2.6: Matrix Vector Multiplication**

**Q4. What are Combiners? Explain the position and significance of combiners**

**Ans:** **[P | Medium]**

**COMBINERS:**

1. A combiner is also known as a **semi-reducer**.
2. It is one of mediator between the mapper phase & the reducer phase.
3. The use of combiners is totally optional.
4. It accepts the output of map phase as an input and pass the key-value pair to the reduce operation.
5. The main function of a Combiner is to summarize the map output records with the same key.
6. It is also known as **grouping by key**.

7. The output (key-value collection) of the combiner will be sent over the network to the actual Reducer task as input.
8. The Combiner class is used in between the Map class and the Reduce class to reduce the volume of data transfer between Map and Reduce.
9. Usually, the output of the map task is large and the data transferred to the reduce task is high.
10. The following figure 2.7 shows position and working mechanism of combiner.

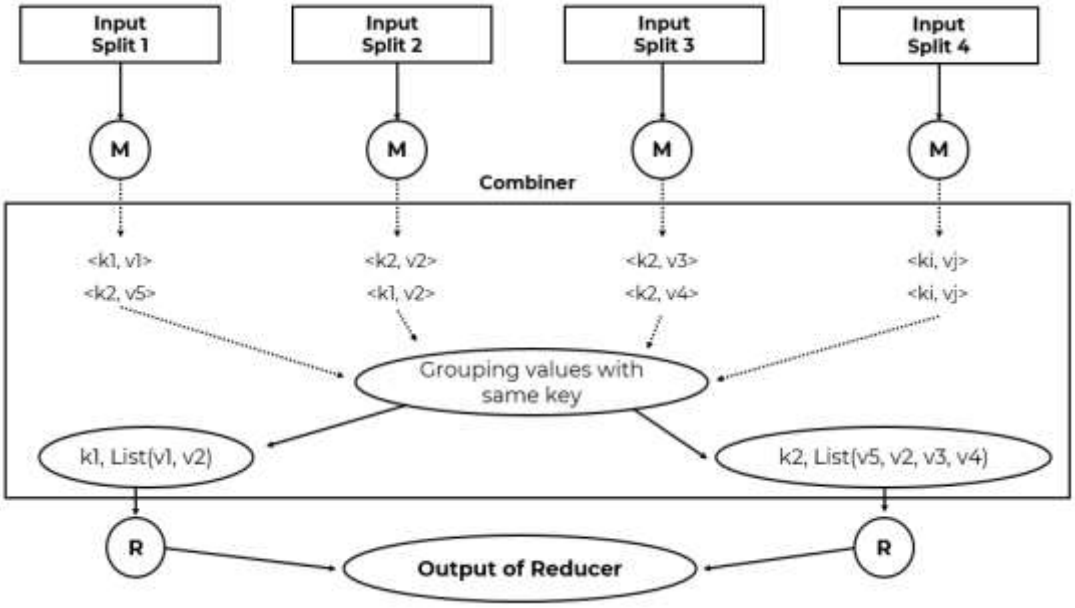



**Figure 2.7: Position and working mechanism of combiner**

**Working:**

1. A combiner does not have a predefined interface and it must implement the Reducer interface's reduce () method.
2. A combiner operates on each map output key.
3. It must have the same output key-value types as the Reducer class.
4. A combiner can produce summary information from a large dataset because it replaces the original Map output.

# CHAP - 3: NOSQL

**Q1. When it comes to big data how NoSQL scores over RDBMS**

**Ans:** **[5M – DEC19]**

**Benefits of NoSQL over RDBMS:**

1. **Schema Less:** NoSQL databases being schema-less do not define any strict data structure.
2. **Dynamic and Agile:**
   a. NoSQL databases have good tendency to grow dynamically with changing requirements.
   b. It can handle structured, semi-structured and unstructured data.
3. **Scales Horizontally:**
   a. In contrast to SQL databases which scale vertically, NoSQL scales horizontally by adding more servers and using concepts of sharding and replication.
   b. This behavior of NoSQL fits with the cloud computing services such as Amazon Web Services (AWS) which allows you to handle virtual servers which can be expanded horizontally on demand.
4. **Better Performance:**
   a. All the NoSQL databases claim to deliver better and faster performance as compared to traditional RDBMS implementations.
   b. Since NoSQL is an entire set of databases (and not a single database), the limitations differ from database to database.
   c. Some of these databases do not support ACID transactions while some of them might be lacking in reliability. But each one of them has their own strengths due to which they are well suited for specific requirements
5. **Continuous Availability:**
   a. The various relational databases may show up modern to high availability for the data transactions while this is much better with the NoSQL databases which excellently show up continuous availability to cope up with different sorts of data transactions at any point of time and in difficult situations.
6. **Ability to handle changes**
   a. The schema-less structure of the NoSQL databases helps it cope up easily with the changes coming with time.
   b. There is a universal index provided for structure, values and text found in the data and hence, it's easy for the organizations to cope with the changes immediately using this information.

**Q2. Explain different ways by which big data problems are handled by NoSQL**

**Q3. Explain 4 ways of NoSQL to operate Big Data Problems**

**Q4. Write short notes on different architectural Patterns in NoSQL**

**Ans:** **[10M – DEC19]**

## DIFFERENT WAYS BY WHICH BIG DATA PROBLEMS ARE HANDLED BY NOSQL:

### I) Key Value Store Databases:

1. It is one of the most basic types of NoSQL databases.
2. This kind of NoSQL database is used as a collection, dictionaries, associative arrays, etc.
3. Data is stored in key/value pairs.
4. It is designed in such a way to handle lots of data and heavy load.
5. Key-value pair storage databases store data as a hash table where each key is unique, and the value can be a JSON, BLOB (Binary Large Objects), string, etc.
6. They work best for shopping cart contents.
7. **Examples:**
   a. Azure Table Storage (ATS)
   b. DyanmoDB
8. **Limitations:**
   a. It may work well for complex queries attempting to connect multiple relations of data.
   b. If data contains lot of many-to-many relationships, a key value store is likely to show poor performance.

| | | | |
|---|---|---|---|
| Key: 1 | ID: 420 | First Name: Jethalal | |
| Key: 2 | Email: BabitaJethalal@gmail.com | Location: Mumbai | Age: 20 |
| Key: 3 | Facebook ID: AngelPriya | Password: AngelPriya98 | Name: Angel |

**Figure 3.1: Example of unstructured data for user records**

### II) Column Store Database:

1. Instead of storing data in relational tuples, it is stored in cells grouped in columns.
2. Column-oriented databases work on columns and are based on BigTable paper by Google.
3. Every column is treated separately.
4. Values of single column databases are stored contiguously.
5. They deliver high performance on aggregation queries like SUM, COUNT, AVG, MIN etc. as the data is readily available in a column.
6. Column-based NoSQL databases are widely used to manage data warehouses, business intelligence, CRM, Library card catalogs,
7. **Examples:**
   a. HBase.
   b. Cassandra

c. Hyper table

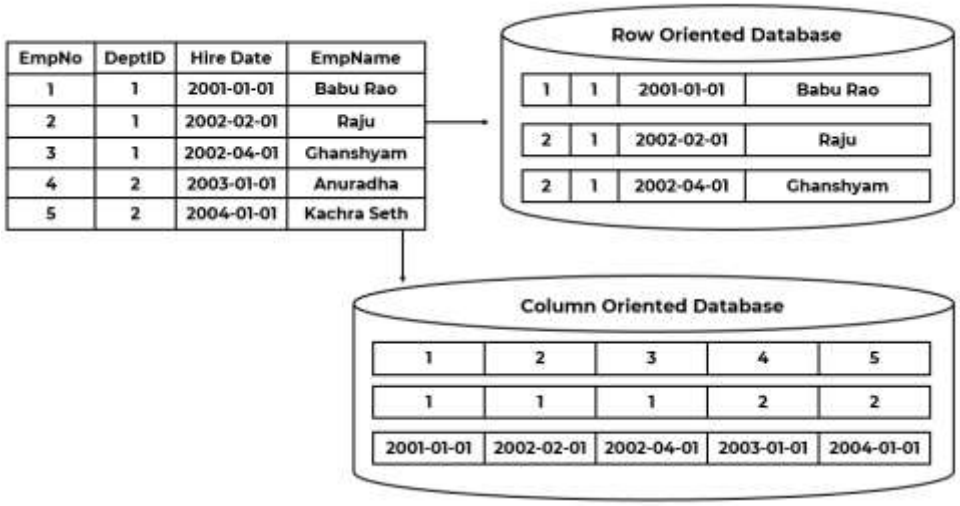



**Figure 3.2: Example of Column store database**

### III) Document Database:

1. Document-Oriented NoSQL DB stores and retrieves data as a key value pair but the value part is stored as a document.
2. The document is stored in JSON or XML formats.
3. Every document contains a unique key, used to retrieve the document.
4. Key is used for storing, retrieving and managing document oriented information also known as semi structured data.
5. **Examples:**
   a. MongoDB
   b. Couch DB
6. **Limitations:**
   a. It's challenging for document store to handle a transaction that on multiple documents.
   b. Document databases may not be good if data is required in aggregation.

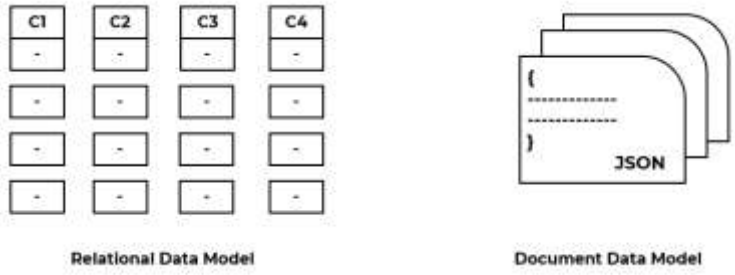



**Figure 3.3: Example of document database**

### IV) Graph Database:

1. A graph type database stores entities as well the relations amongst those entities.
2. The entity is stored as a node with the relationship as edges.
3. An edge gives a relationship between nodes.
4. Every node and edge has a unique identifier.
5. Compared to a relational database where tables are loosely connected, a Graph database is a multi-relational in nature.

6. Traversing relationship is fast as they are already captured into the DB, and there is no need to calculate them.
7. Graph base database mostly used for social networks, logistics, and spatial data.
8. **Examples:**
   a. Neo4J
   b. Infinite Graph.
   c. OrientDB.
   d. FlockDB

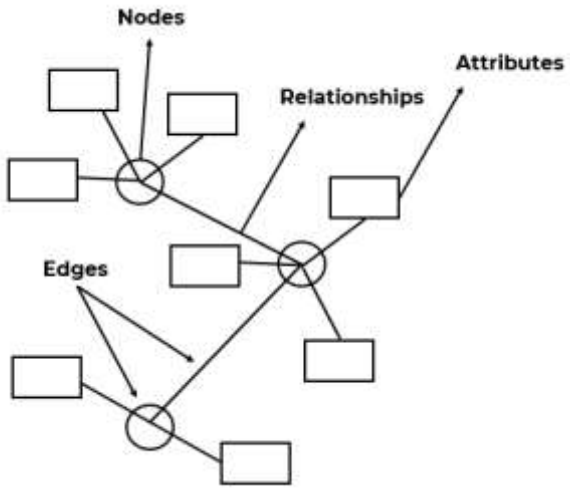



**Figure 3.4: Example of graph database**

## -- EXTRA QUESTIONS --

**Q1. Write short notes on NoSQL**

**Ans:** **[P | Low]**

**NoSQL:**

1. NoSQL stands for Not Only SQL.
2. NoSQL is a non-relational DMS, that does not require a fixed schema, avoids joins, and is easy to scale.
3. NoSQL database is used for distributed data stores with humongous data storage needs.
4. NoSQL is used for Big data and real-time web apps.
5. For example, companies like Twitter, Facebook, Google that collect terabytes of user data every single day.
6. NoSQL generally avoids join operations.

**FEATURES:**

**I) Non-relational:**

1. NoSQL databases never follow the relational model.
2. Never provide tables with flat fixed-column records.
3. Work with self-contained aggregates or BLOBs.
4. Doesn't require object-relational mapping and data normalization.
5. No complex features like query languages, query planners, referential integrity joins, ACID.

**II) Schema-free**

1. NoSQL databases are either schema-free or have relaxed schemas.
2. Do not require any sort of definition of the schema of the data.
3. Offers heterogeneous structures of data in the same domain.

**III) Simple API**

1. Offers easy to use interfaces for storage and querying data provided.
2. APIs allow low-level data manipulation & selection methods.
3. Text-based protocols mostly used with HTTP REST with JSON.
4. Mostly used no standard based query language.
5. Web-enabled databases running as internet-facing services

**IV) Distributed**

1. Multiple NoSQL databases can be executed in a distributed fashion.
2. Offers auto-scaling and fail-over capabilities.
3. Often ACID concept can be sacrificed for scalability and throughput.
4. Mostly no synchronous replication between distributed nodes Asynchronous Multi-Master Replication, peer-to-peer, HDFS Replication.
5. Only providing eventual consistency.
6. Shared Nothing Architecture. This enables less coordination and higher distribution.

**ADVANTAGES:**

1. Good Resources Scalability.
2. No Static Schema.

3. Faster Data Processing.
4. Lower Operational Cost.
5. Support Semi Structure Data.
6. Relatively simple data models.

**DISADVANTAGES:**

1. Not a defined standard.
2. Limited query capabilities.

---

**Q2. Write short notes on DynamoDB**

**Ans:** **[P | Medium]**

**DYNAMODB:**

1. DynamoDB is a fully managed NoSQL database service provided by Amazon.
2. It is database service that provides fast and predictable performance with seamless scalability.
3. It supports key-value and document data structures.
4. DynamoDB data model contains:
   a. Tables
   b. Items
   c. Attributes
5. DynamoDB requires only a primary key and does not require to define all of the attribute names and data types in advance.
6. Each attribute of DynamoDB in an item is a name-value pair.
7. Figure 3.5 shows DynamoDB Namespace.

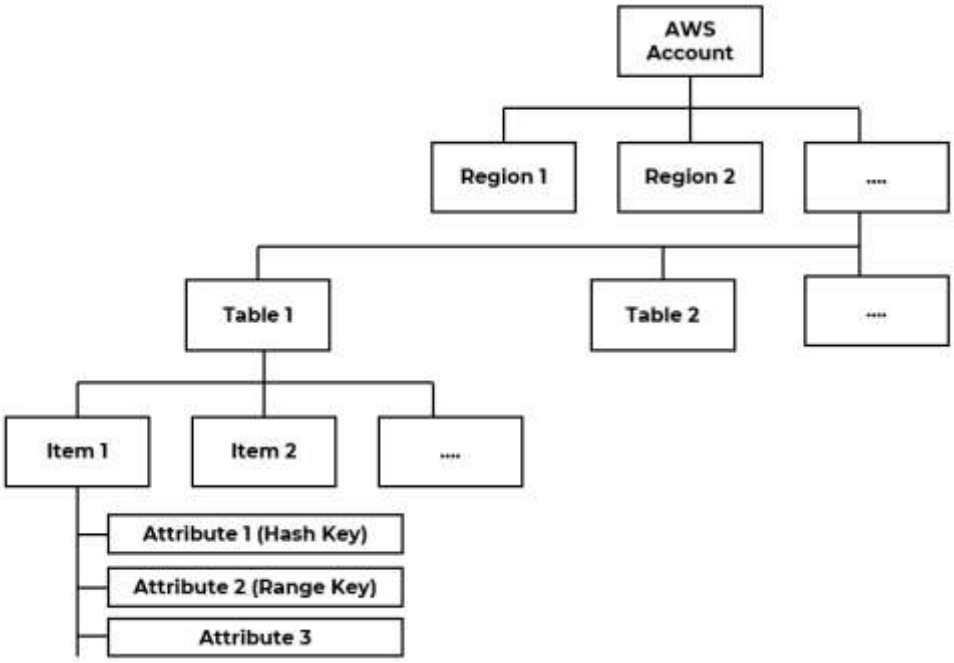



**Figure 3.5: DynamoDB Namespace.**

8. **Example:**

```
{
    ID = 1
```

SubjectName = “Big Data & Analytics”
Authors = [“Sagar Narkar”]
Price = 100
PageCount = 75
Publication = BackkBenchers Publication House
},
{
ID = 2
SubjectName = “Artificial Intelligence & Soft Computing”
Authors = [“Sagar Narkar”]
Price = 100
PageCount = 93
Publication = BackkBenchers Publication House
}

9. **Data type:**
   a. **Scalar types:** Number, String, Binary, Boolean and NULL
   b. **Document types:** List and Map
   c. **Set types:** String Set, Number Set and Binary Set
10. **Data Access:**
   a. DynamoDB is a web service uses HTTP and HTTPS as a transport layer services.
   b. JSON can be used as a message serialization format.
   c. It is possible to use AWS software development kits (SDKs)
   d. Application code makes requests to the DynamoDB web service API.

---

**Q3. Explain Shared-Nothing Architecture in detail**

**Ans:** **[P | Medium]**

**SHARED-NOTHING ARCHITECTURE:**

1. There are three ways that resources can be shared between computer systems: shared RAM, shared disk, and shared-nothing
2. Of the three alternatives, a shared-nothing architecture is most cost effective in terms of cost per processor when you’re using commodity hardware.
3. A shared-nothing architecture (SN) is a distributed-computing architecture.
4. Each processor has its own local memory and local disk.
5. A processor at one node may communicate with another processor using high speed communication network.
6. Any terminal can act as a node which functions as Server for data that is stored on local disk.
7. Interconnection networks for shared nothing systems are usually designed to be scalable.
8. Figure 3.6 shows shared nothing architecture.

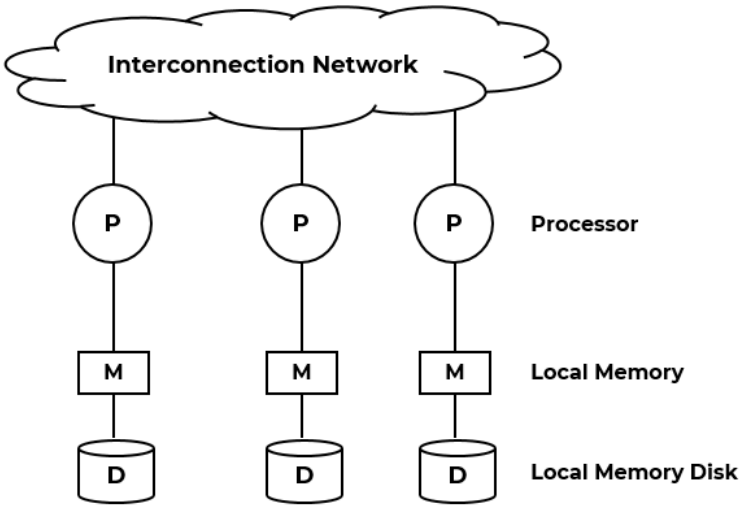



**Figure 3.6: Shared Noting Architecture**

**ADVANTAGES:**

1. It is more scalable.
2. Easily support large numbers of processors.
3. It can achieve high degree of parallelism.

**DISADVANTAGES:**

1. It requires rigid data partitioning.
2. Cost of communication and of non-local disk access is higher.

**APPLICATIONS:**

1. Teradata database machine uses share nothing architecture.
2. Shared-nothing is popular for web development.

# CHAP - 4: MINING DATA STREAMS

**Q1. Explain with block diagram architecture of Data Stream Management System**

**Ans:** **[10M – DEC19]**

## DATA STREAM MANAGEMENT SYSTEM:

1. For handling and controlling the data streams, we require a concrete, standardized model or framework so that data stream will be processed in a properly defined manner.
2. Figure 4.1 shows the block diagram of data stream management system.

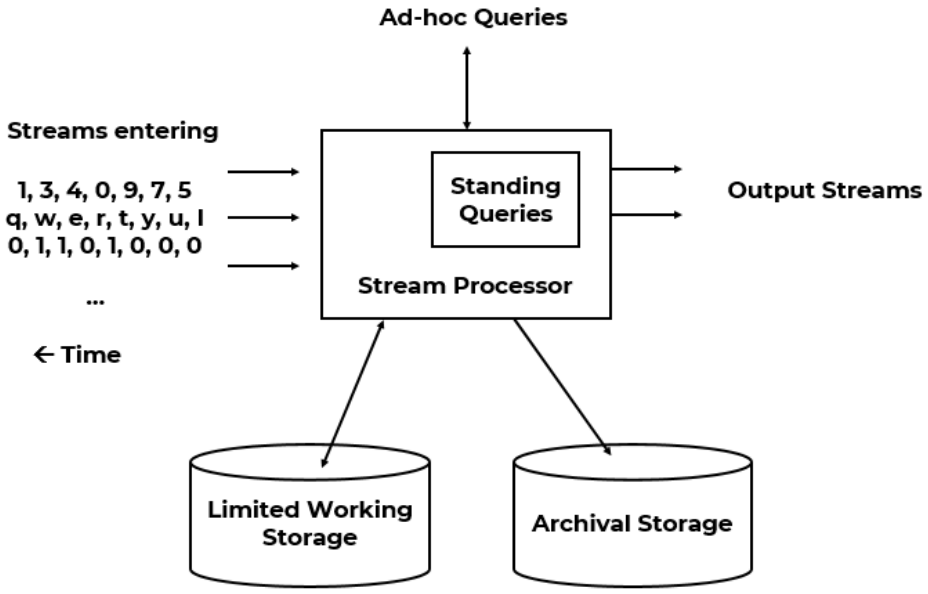



**Figure 4.1: Block diagram of data stream management system**

3. The data-stream management system architecture is very much similar to that of conventional relational data base management system architecture.
4. The basic difference is that processor block or more specifically a query processor (query engine) is replaced with the specialized block known as **stream processor**.
5. The first block in the system architecture shows the input part.
6. Number of data stream generated from different data sources will enter into the system.
7. Every Data stream has its own characteristics such as:
   a. Every data stream can schedule and rearrange its own data items.
   b. Every data stream involved heterogeneity i.e. in each data stream we can found different kinds of data such as numerical data, alphabets, alphanumeric data, graphics data, textual data, binary data or any converted transformed or translated data.
   c. Every data stream has different input data rate.
   d. No uniformity is maintained by the elements of different data streams while entering into the stream processor.
8. In the second block of system architecture, it is abstracted that, there are two different sub systems exists one of which will take care of storing the data stream and other is responsible for fetching the data stream from secondary storage and processing it by loading into main memory.

9. Hence the rate at which stream enters into the system is not the burden of sub-system which is involved in the stream processing.
10. It will be controlled by other sub-system which involved in storage of data stream.
11. The third block represents the active storage or working storage for processing the different data streams.
12. The working storage area may also contain sub-streams which are integral part of main-core stream to generate result for a given query.
13. Working storage basically a main memory but situation demands then data items within the stream can be fetched from the secondary storage.
14. The major issue associated with working storage is its limited size.
15. The fourth block of system architecture is known as archival storage.
16. As name indicates, this block is responsible for maintaining the details of very transaction within the system architecture.
17. It is also responsible to maintain the edit logs.
18. Edit logs are nothing but updation of data (i.e. values form users or system).
19. In some special purpose case, archival storage will be used to process or to retrieve previously used data items which are not present on the secondary storage.
20. The fifth block is responsible for displaying or delivery the output stream generated as a result of processing done by the stream processor usually by taking the support of working storage and occasionally by taking support of archival storage.

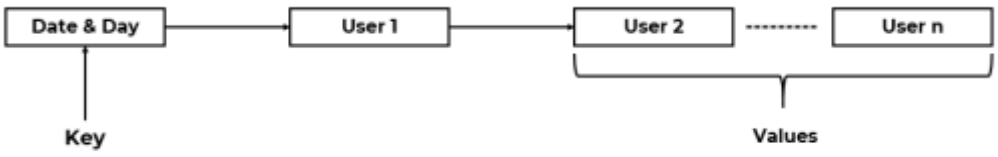
---

**Q2. What do you mean by Counting Distinct Elements in a stream. Illustrate with an example working of a Flajolet — Martin Algorithm used to count number of distinct elements**

**Ans:** **[10M – DEC19]**

**COUNT DISTINCT PROBLEM:**

1. In computer science, the count-distinct problem is the problem of finding the number of distinct elements in a data stream with repeated elements.
2. Consider an example of counting number of newly added users who will view a given web page 1st time in the last month.
3. Clearly, the standard approach to solve this problem is, make a list of all elements of user who has seen before in a given data stream.
4. Convert the data elements in the list into some efficient data structure, more specifically a search structure such as tree.
5. The major benefits of this kind of structure is we can have bucket like structure where element from some category will be found in some bucket.
6. E.g. list of user who has seen a given web page on some data.



7. The addition of new user to an existing list according to decided key is very easy with this structure.
8. So, it seems to be very easy to count the distinct element within the given stream.
9. The major problem arises, when the number of distinct element are too much in number and to make it worst.
10. If we have number of such data stream who will enter into the system at the same time.
11. E.g. If organization like Yahoo or Google requires a count of user who will see their different web pages 1st time for a given month.
12. Here, for every page we need to maintain above said problem which seems to be a complicated problem.
13. To have alternate solution to this problem, we can have "scale out of machines".
14. In this approach, we can add new commodity hardware (an ordinary server) to existing hardware so that load of processing and counting the distinct user on every web page of an organization will be distributed.
15. Another additional thing is we can have the use of secondary memory can batch systems.

## FLAJOLET-MARTIN ALGORITHM:

1. The Problem of counting the distinct element can be solved with the help of ordinary hashing technique.
2. A hash function will be applied to a given set which generate a bit-string as a result of hashing.
3. A constraint should be applied to above process is, there should enough hashing results than elements present inside the set.
4. The general procedure while applying hash function is to pick different hash function for and apply these every element in given data stream.
5. The significant property of flash function is that, whenever applied to the same data element in a given data stream it will generate the same hash value.
6. So, **Flajolet-Martin algorithm** has extended this hashing idea and properties to count distinct elements.
7. The algorithm was introduced by **Philippe Flajolet and G. Nigel Martin** in their 1984
8. Flajolet-Martin algorithm approximates the number of unique objects in a stream or a database in one pass.
9. If the stream contains n elements with m of them unique, this algorithm runs in O(n) time and needs O(log(m)) memory.
10. So the real innovation here is the memory usage, in that an exact, brute-force algorithm would need O(m) memory (e.g. think "hash map").
11. As noted, this is an approximate algorithm. It gives an approximation for the number of unique objects, along with a standard deviation $\sigma$, which can then be used to determine bounds on the approximation with a desired maximum error $\epsilon$, if needed.

## ALGORITHM

1. Create a bit vector of sufficient length L, such that $2L > n$, the number of elements in the stream. Usually a 64-bit vector is sufficient since 264 is quite large for most purposes.

2. The $i^{th}$ bit in this vector/array represents whether we have seen a hash function value whose binary representation ends in 0i. So initialize each bit to 0.
3. Generate a good, random hash function that maps input (usually strings) to natural numbers.
4. Read input. For each word, hash it and determine the number of trailing zeros. If the number of trailing zeros is k, set the k-th bit in the bit vector to 1.
5. Once input is exhausted, get the index of the first 0 in the bit array (call this R). By the way, this is just the number of consecutive 1s (i.e. we have seen 0, 00, ..., $0^{R-1}$ as the output of the hash function) plus one.
6. Calculate the number of unique words as $2R/\phi$, where $\phi$ is 0.77351.
7. The standard deviation of R is a constant: $\sigma(R) = 1.12$. (In other words, R can be off by about 1 for 1-0.68=32% of the observations, off by 2 for about 1-0.95=5% of the observations, off by 3 for 1-0.997=0.3% of the observations using the Empirical rule of statistics).
8. This implies that our count can be off by a factor of 2 for 32% of the observations, off by a factory of 4 for 5% of the observations, off by a factor of 8 for 0.3% of the observations and so on.

**EXAMPLE:**

S=1, 3, 2, 1, 2, 3, 4, 3, 1, 2, 3, 1

Assume |b| = 5

| x | h(x) | Rem | Binary | r(a) |
|---|---|---|---|---|
| 1 | 7 | 2 | 00010 | 1 |
| 3 | 19 | 4 | 00100 | 2 |
| 2 | 13 | 3 | 00011 | 0 |
| 1 | 7 | 2 | 00010 | 1 |
| 2 | 13 | 3 | 00011 | 0 |
| 3 | 19 | 4 | 00100 | 2 |
| 4 | 25 | 0 | 00000 | 5 |
| 3 | 19 | 4 | 00100 | 2 |
| 1 | 7 | 2 | 00010 | 1 |
| 2 | 13 | 3 | 00011 | 0 |
| 3 | 19 | 4 | 00100 | 2 |
| 1 | 7 | 2 | 00010 | 1 |

R = max (r(a)) = 5

So no. of distinct elements = N

$= 2^R$

$=2^5$

=32

## -- EXTRA QUESTIONS --

**Q1. Explain Datar-Gionis-Indyk-Motwani (DGIM) Algorithm in details**

**Ans:** **[P | High]**

**DGIM:**

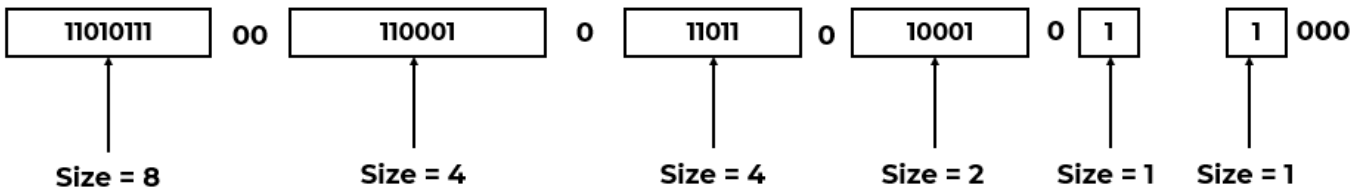
1. DGIM stands for **Datar-Gionis-Indyk-Motwani Algorithm**.
2. It is designed to find the number 1's in a data set.
3. This algorithm uses **O(log²N)** bits to represent a window of N bit, allows to estimate the number of 1's in the window with and error of no more than 50%.
4. In DGIM algorithm, each bit that arrives has a timestamp, for the position at which it arrives.
5. If the first bit has a timestamp 1, the second bit has a timestamp 2 and so on.
6. The positions are recognized with the window size N (the window sizes are usually taken as a multiple of 2).
7. The windows are divided into buckets consisting of 1's and 0's.

**RULES FOR FORMING THE BUCKETS:**

1. The right side of the bucket should always start with 1. (if it starts with a 0, it is to be neglected) E.g. · 1001011 → a bucket of size 4, having four 1's and starting with 1 on it's right end.
2. Every bucket should have at least one 1, else no bucket can be formed.
3. All buckets should be in powers of 2.
4. The buckets cannot decrease in size as we move to the left. (move in increasing order towards left)

**EXAMPLE:**

Consider the following stream of bits: ...1110101110011000101101101 1000




**STORAGE SPACE REQUIREMENTS:**

The storage space required for the DGIM Algorithm can be determined as,

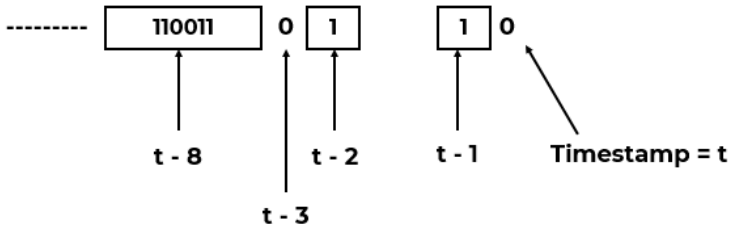
1. A single bit represented as O(Log N)
2. Number of baskets O(Log N)
3. Aggregate space requirement = O(Log2N)

**QUERY – ANSWERING in DGIM:**

1. Given k≤N, we want to know how many of the last k bits were 1-s.
2. To find number of 1's in given stream say last 10 bits.

| 1 1 0 1 0 1 1 1 0 0 1 1 0 0 0 1 0 1 1 0 0 1 1 0 1 1 0 |

3. For last 10 bits, K = 10
4. Therefore, part of stream under consideration will be,




5. This is the basket which is partly in representation because bit left to the bit t – 8 will t – 9 and so on.
6. So till t- 8 will be included.
7. Therefore quoted answer for number of 1's in last 10 bits K = 10 will be 6 but actual answer is 5.

---

**Q2. Explain bloom filter? Explain bloom filtering process with neat diagram**

**Ans:** **[P | Medium]**

**BLOOM FILTER:**

1. Bloom filter is a technique to rectify data stream for said criteria.
2. It is a data structure.
3. The efficiency of bloom filter is that it is probabilistic data structure which tells us that the element is either present in the set or not.
4. The base data structure of a Bloom filter is a **Bit Vector**.
5. Bloom filter consists of:
   a. An Array.
   b. Hash Function.
   c. A Set.
6. Figure 4.2 represents Bloom Filter.

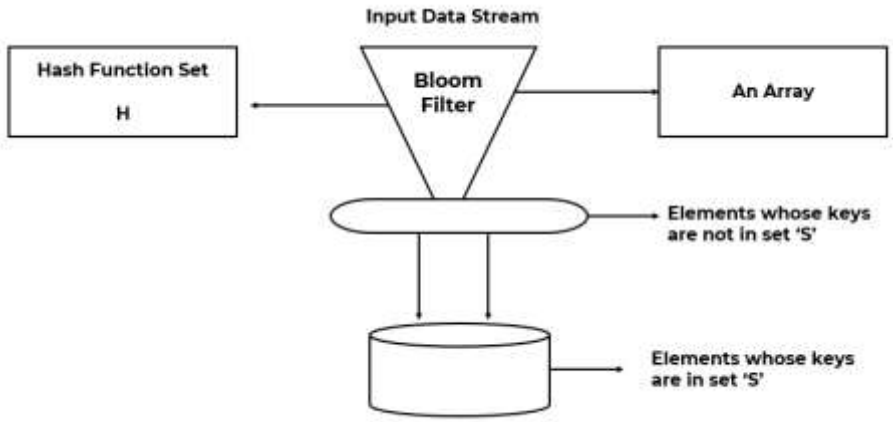



**Figure 4.2: Bloom Filter**

**BLOOM FILTERING PROCESS:**

1. Bloom filtering is an array of 'n' bit size.
2. This array is initialize to '0' for all n bits.
3. Pick every key and value present inside the data stream 's'.
4. Apply every hash function $h_i$ where (i is between 1....K)
5. Set value 1 to every bit of data stream for $h_i(k)$ in the sets 'S'.
6. Check the values for every $h_i$ applied on bit in 'S'.

7. If the value of K is 1 ∀ (For All), $h_i$ – i varies from 1 to k
8. If for all $h_i(k)$ values is 1, then set its bit value to 1 and allow that element of stream to be involved in a sample.
9. If anyone or more $h_i(k)$ value is not evaluates to 1 then it is conducted that 'K' doesn't satisfies the criteria. Hence it will not be in sample.

**STRENGTHS:**

1. **Space-efficient**: Bloom filters take up O(1) space, regardless of the number of items inserted.
2. **Fast**: Insert and lookup operations are both O(1) time.

**WEAKNESSES:**

1. **Probabilistic:** Bloom filters can only definitively identify true negatives. They cannot identify true positives. If a bloom filter says an item is present, that item might actually be present (a true positive) or it might not (a false positive).
2. **Limited Interface**: Bloom filters only support the insert and lookup operations. You can't iterate through the items in the set or delete items.

# CHAP - 5: FINDING SIMILAR ITEMS AND CLUSTERING

**Q1. Explain Edit distance measure with an example**

**Ans:** **[5M – DEC19]**

## EDIT DISTANCE MEASURE:

1. Edit distance is a way of quantifying how dissimilar two strings (e.g., words) are to one another by counting the minimum number of operations required to transform one string into the other.
2. This distance makes sense when points are strings.
3. As in edit distance we have string representation so, it will satisfy negativity, positivity and symmetry and triangle in equating constraints.
4. The distance between two strings x = x1, x2 · · · xn and y = y1, y2 · · · ym is the smallest number of insertions and deletions of single characters that will convert x to y.
5. There are two variants of edit distance i.e. classical method and longest common sequence method.

### I) Classical Method:

1. Consider following representation of points x and y in a given space having string representation as,

   X = ABCDE

   Y = ACFDEG
2. To calculate the distance between x and y we should convert x string to y string.
3. Now comparing the character sequence in both the strings.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| X | = | A | B | C | D | E | | |
| Y | = | A | C | F | D | E | G | |
| | | 1 | 2 | 3 | 4 | 5 | 6 | Position |

4. Clearly, position 2, 3, 6 differ in their contents from x and y. So make necessary insertion and deletions at in accordance with these 3 positions.

| | | |
|---|---|---|
| B | C | - |
| C | F | G |
| 2 | 3 | 6 |

5. Delete the character ‘B’ in string x from position no. 2
6. Shift the other characters to left hand side.
7. Now current status of string x is,

| | | | | | |
|---|---|---|---|---|---|
| X | = | A | C | D | E |
| | | 1 | 2 | 3 | 4 |

8. Now insert the character ‘F’ at position 3 i.e. after character ‘C’ and before character ‘D’
9. Therefore, now the status of string x will be,

| | | | | | | |
|---|---|---|---|---|---|---|
| X | = | A | C | F | D | E |
| | | 1 | 2 | 3 | 4 | 5 |

10. Lastly, append the characters G in the X string get $6^{th}$ position.
11. Status of string X will be,

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| X | = | A | C | F | D | E | G |
| | | 1 | 2 | 3 | 4 | 5 | 6 |

12. Hence in above example, we have
    No. of deletions = 1
    No. of insertion = 2
    Edit distance between X, Y
    d(x, y) = No. of deletion + No. of insertions
    = 1 + 2 = 3

**II) Longest Common Sequence Method:**

1. The longest common sequence can be developed by performing detection operations on the character positions in respective string.
2. Suppose we have two point x and y represented as strings.
3. Therefore, edit distance between points x and y will be,
   d(x, y) = (Length of string x + Length of string y) – 2 x Length of longest common sequence)
4. Suppose,

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| X | = | A | B | C | D | E | |
| Y | = | A | C | F | D | E | G |

5. Here,
   Longest common sequence (LCS) is nothing, but the (a, c, d, e) = 4
   Length of string x = 5
   Length of string y = 6
   Therefore, d(x, y) = 5 + 6 – 2 x 4
   = 5 + 6 – 8
   = 11 – 8
   = 3

---

**Q2. What are the challenges in clustering of Data streams. Explain stream clustering algorithm in detail**

**Ans:** **[10M – DEC19]**

**CHALLENGES IN CLUSTERING OF DATA STREAMS:**

**I) Cluster Validity:**

1. Recent developments in data stream clustering have heightened the need for determining suitable criteria to validate results.
2. Most outcomes of methods are depended to specific application.
3. However, employing suitable criteria in results evaluation is one of the most important challenges in this arena.

**II) Space limitation:**

1. Not only time and concept drift are the main complexity in data stream clustering but also space complexity in some applications (e.g. wireless sensor network monitoring and controlling) can be caused difficulties in processing.

2. Sensors with small memory are not able to keep a big amount of data so new method for data stream clustering should be managed this limitation.

**III) High dimensional data stream:**

1. There are high dimensional data sets (e.g. image processing, personal similarity, customer preferences clustering, network intrusion detection, wireless sensors network and generally time series data) which should be managed through the processing of data stream.
2. In huge databases, data complexity can be increased by number of dimensions.

## BDMO ALGORITHM / STREAM CLUSTERING ALGORITHM:

1. Stream clustering algorithm is also referred as BDMO Algorithm.
2. BDMO Algorithm is designed by Bahcock, Datar, Motwani and OCallaghan.
3. It is based on **K-Means**.
4. The BDMO algorithm follows the concept of 'counting ones' method, which means that there is a window of length N on a binary stream and it counts the number of 1s that comes in the last k bits where k ≤ N.
5. The BDMO algorithm uses the bucket with allowable bucket sizes that forms a sequence where each size is twice of the previous size.
6. In the algorithm, the number of points represents the size of the bucket.
7. It does not consider that the sequence of allowable bucket sizes starts with 1 but consider only forming a sequence such as 2, 4, 6, 8... Where each size is twice the previous size.
8. For maintaining the buckets, the algorithm considers the size of the bucket with the power of two.
9. In addition, the number of buckets of each size is either one or two that form a sequence of non-decreasing size.
10. The buckets that are used in the algorithm, contains the size and timestamp of the most recent points of the stream.
11. Along with this, the bucket also contains a collection of records that represents the clusters into which the points of that bucket have been partitioned.
12. This record contains the number of points in the cluster, the centroid, or clustroid of the cluster, and other parameters that are required to merge and maintain the clusters.
13. The major steps of the BDMO algorithm are as follows:
    a. Initialising buckets
    b. Merging buckets.
    c. Answering queries.

## INITIALISING BUCKETS:

1. Initialisation of the bucket is the first step of the BDMO algorithm.
2. The algorithm uses the smallest bucket size that is p with a power of two.
3. It creates a new bucket with the most recent p points for p stream elements.
4. The timestamp of the most recent point in the bucket is the timestamp of the new bucket.
5. After this, we may choose to leave every point in a cluster by itself or perform clustering using an appropriate clustering method.
6. For example, if k-means clustering method is used, it clusters the k points into k clusters.

7. For the Initialisation of the bucket using selected clustering methods, it calculates the centroid or clustroid for the clusters and counts the points in each cluster.
8. All this information is stored and becomes a record for each cluster.
9. The algorithm also calculates the other required parameters for the merging process.

**MERGING BUCKETS:**

1. After the Initialisation of the bucket, the algorithm needs to review the sequence of a bucket.
2. If there happens to be a bucket with a timestamp more than N time units prior to the current time then nothing of that bucket is in the window.
3. In such a case, the algorithm drops it from the list
4. In case we had created three buckets of size p, then we must merge the oldest two of the three buckets.
5. In this case, the merger can create two buckets of size 2p, this may require us to merge buckets of increasing sizes recursively.
6. For merging two consecutive buckets, the algorithm needs to perform the following steps:
   a. For merging, the size of the bucket should be twice the sizes of the two buckets to be merged.
   b. The timestamp of the merged bucket is the timestamp of the more recent of the two consecutive buckets.
   c. In addition, it is necessary to calculate the parameters of the merged clusters.

**ANSWERING QUERIES:**

1. A query in the stream-computing model is a length of a suffix of the sliding window.
2. Any algorithm takes all the clusters in all the buckets that are at least partially within the suffix and then merges them using some method.
3. The answer of the query is the resulting clusters.
4. For the clustering of the streams, the stream-computing model finds out the answer to the query 'What are the clusters of the last or more recent m points in the stream for in $m \leq N$'.
5. During the Initialisation, the k-means method is used and for merging the buckets timestamp is used.
6. Hence the algorithm is unable to find a set of buckets that covers the last in points.
7. However, we can choose the smallest set of buckets that covers the last m points and include in these buckets no more than the last 2m points.
8. After this, the algorithm generates the answer in response to the query as 'the centroids or clustroids of all the points in the selected buckets'.

## -- EXTRA QUESTIONS --

**Q1. Write short notes on distance measures in big data**

**Ans:** **[P | High]**

### DISTANCE MEASURES:

1. A set of point is known as a **space**.
2. By using the space as a platform we can calculate the distance measure by applying the function 'd' on any given two points x and y in a plane.
3. The distance function d(x, y) generates a real number.
4. Number of constraints can be imposed on the distance measures as follows:
   a. **Negative of Distances:** The distance between any two points say x and y cannot be negative

      **id(x, y) ≥0**

   b. **Positivity of Distances:** The distance between any two points say x and y is said to be zero if and only if x and y has same co-ordinates i.e. x = y

      **d(x, y) = 0 iff x = y**

   c. **Symmetry of Distance:** The distance between any two points say x and y are in one direction. i.e. distance from x and y is same as that of distance from y to x.

      **d(x, y) = d(y, x)**

   d. **Triangular inequality of distances:** When we are dealing with the terminologies like distance. We strive to have the minimum time, distance to From 1 point to other. To achieve the minimum distance between two points if we introduce some other point in between these two point. Then it doesn't prove to be an efficient solution.

      **d(x, y) ≤ d(x, z) + d(z, y)**

### TYPES OF DISTANCE MEASURES:

#### I) Euclidean Distances:

1. Euclidean distance is the fundamental technique to measure the distance.
2. The space in Euclidean distance is known as Euclidean space.
3. Euclidean distance is the "ordinary" straight-line distance between two points in Euclidean space.
4. In this Euclidean space every point is nothing but the vector which contains 'n' real numbers.
5. To calculate, the distance between any given two points we can have following formula known as $L_2$ Norm.

$$d_{euc}(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

#### II) Jaccard Distance:

1. Jaccard distance is a measure of how dissimilar two sets are.
2. It can be calculates with the help of following formula:

   **$J_d$ = d(x, y) = 1 – SIM(x, y)**

3. That is, the Jaccard distance is 1 minus the ratio of the sizes of the intersection and union of sets x & y

### III) Cosine Distance:

1. Cosine distance method is useful in space.
2. Here the space is nothing but the Euclidean space which contain set of points which are represented as vector along with an integer or Boolean values.
3. The cosine distance between any two given vector is nothing but the angle made by these vectors.
4. These angles range from $0^{\circ}$ to $180^{\circ}$ irrespective of no. of dimensions.
5. Hence to calculate cosine distance between any given no, of vectors:
   a. Calculate the angle made by them.
   b. Apply arc cosine function.
   c. Cosine angle is nothing but the dot product of two vectors which should be divided by $L_2$ Norms of x and y. i.e. Euclidean distance.

### IV) Edit Distance:

1. Edit distance is a way of quantifying how dissimilar two strings (e.g., words) are to one another by counting the minimum number of operations required to transform one string into the other.
2. This distance makes sense when points are strings.
3. The distance between two strings x = x1, x2 · · · xn and y = y1, y2 · · · ym is the smallest number of insertions and deletions of single characters that will convert x to y.

### V) Hamming Distance:

1. The space is nothing but collection of points.
2. These points can be represented as vector.
3. Every vector is composed of different components such as magnitude, directions etc.
4. When vector differs in their components then that difference between one or more vectors is known as hamming distance.
5. As this distance calculation depends on the difference operation so it will satisfy all constraints such as negativity of distances, positivity of distance, symmetry and triangle in equality.

**Q2. Write short notes on CURE Algorithm**

**Ans:** **[P | High]**

## CURE ALGORITHM:

1. Cure stands for **Clustering Using Representatives Algorithm**.
2. It is an efficient data clustering algorithm for large databases.
3. CURE Algorithm works better in spherical as well as non-spherical clusters.
4. CURE uses random sampling and partitioning to speed up clustering.

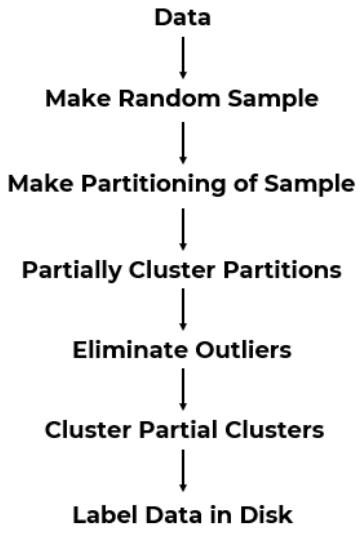



**Figure 5.1: CURE Overview**

5. The CURE algorithm is divided into phases:
   a. Initialization in CURE
   b. Completion of the CURE Algorithm

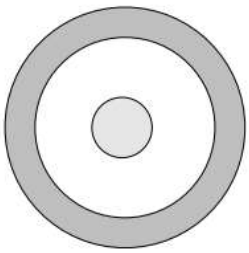


**Figure 5.2: Two clusters, one surrounding the other**

6. Figure 5.2 is an illustration of two clusters.
7. The inner cluster is an ordinary circle, while the second is a ring around the circle.
8. This arrangement is not completely pathological.

## INITIALIZATION IN CURE:

1. Take a small sample of the data and cluster it in main memory.
2. In principle, any clustering method could be used, but as CURE is designed to handle oddly shaped clusters, it is often advisable to use a hierarchical method in which clusters are merged when they have a close pair of points.
3. Select a small set of points from each cluster to be representative points as shown in figure 5.3.

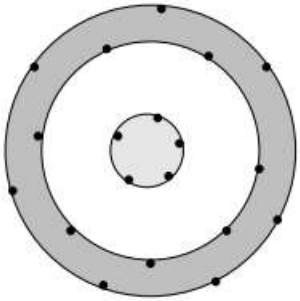


**Figure 5.3: Select representative points from each cluster, as far from one another as possible**

4. These points should be chosen to be as far from one another as possible, using the K-means method.
5. Move each of the representative points a fixed fraction of the distance between its location and the centroid of its cluster.
6. Perhaps 20% is a good fraction to choose. Note that this step requires a Euclidean space, since otherwise, there might not be any notion of a line between two points.

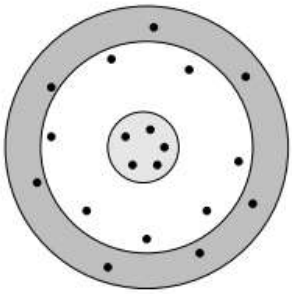


**Figure 5.4: Moving the representative points 20% of the distance to the cluster's centroid**

**COMPLETION OF THE CURE ALGORITHM:**

1. The next phase of CURE is to merge two clusters if they have a pair of representative points, one from each cluster, that are sufficiently close.
2. The user may pick the distance that defines "close."
3. This merging step can repeat, until there are no more sufficiently close clusters.

# CHAP - 6: REAL TIME BIG DATA MODELS

**Q1. Give Applications of Social Network Mining**

**Ans:** **[5M – DEC19]**

## SOCIAL NETWORK MINING AS GRAPHS:

1. Social networks are naturally modeled as graphs, which we sometimes refer to as a social graph.
2. The entities are the nodes, and an edge connects two nodes if the nodes are related by the relationship that characterizes the network.
3. If there is a degree associated with the relationship, this degree is represented by labeling the edges. Often, social graphs are undirected, as for the Facebook friend's graph.
4. But they can be directed graphs, as for example the graphs of followers on Twitter or Google+

## APPLICATIONS:

1. **Telephone Networks:** Here the nodes represent phone numbers, which are really individuals. There is an edge between two nodes if a call has been placed between those phones in some fixed period of time, such as last month, or "ever." The edges could be weighted by the number of calls made between these phones during the period.
2. **Email Networks:** The nodes represent email addresses, which are again individuals. An edge represents the fact that there was at least one email in at least one direction between the two addresses.
3. **Collaboration Networks:** Nodes represent individuals who have published research papers. There is an edge between two individuals who published one or more papers jointly
4. **Other examples include:** information networks (documents, web graphs, patents), infrastructure networks (roads, planes, water pipes, powergrids), biological networks (genes, proteins, food-webs of animals eating each other), as well as other types, like product co-purchasing networks (e.g., Groupon).

---

**Q2. Explain the following terms with diagram**

**1) Hubs and Authorities**

**2) Structure of the Web**

**Ans:** **[10M – DEC19]**

## HUBS AND AUTHORITIES:

1. Hyperlink-Induced Topic Search (HITS; also known as hubs and authorities) is a link analysis algorithm.
2. HITS rates Web pages, developed by Jon Kleinberg.
3. Hubs and authorities are fans and centers in a bipartite core of a web graph.
4. A good hub page is one that points to many good authority pages.
5. A good authority page is one that is pointed to by many good hub pages.
6. Figure 6.1 represents hubs and authorities.

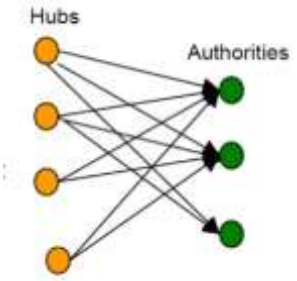



**Figure 6.1: Hubs and authorities**

7. In the HITS algorithm, the first step is to retrieve the most relevant pages to the search query.
8. This set is called the root set and can be obtained by taking the top pages returned by a text-based search algorithm.
9. A base set is generated by augmenting the root set with all the web pages that are linked from it and some of the pages that link to it.
10. The web pages in the base set and all hyperlinks among those pages form a focused subgraph.
11. The HITS computation is performed only on this focused subgraph.
12. According to Kleinberg the reason for constructing a base set is to ensure that most (or many) of the strongest authorities are included.
13. Authority and hub values are defined in terms of one another in a mutual recursion.
14. An authority value is computed as the sum of the scaled hub values that point to that page.
15. A hub value is the sum of the scaled authority values of the pages it points to.
16. Some implementations also consider the relevance of the linked pages.
17. The algorithm performs a series of iterations, each consisting of two basic steps:
    a. **Authority update:** Update each node's authority score to be equal to the sum of the hub scores of each node that points to it. That is, a node is given a high authority score by being linked from pages that are recognized as Hubs for information.
    b. **Hub update:** Update each node's hub score to be equal to the sum of the authority scores of each node that it points to. That is, a node is given a high hub score by linking to nodes that are considered to be authorities on the subject.

## STRUCTURE OF WEB:

1. If we consider web pages as vertices and hyperlinks as edges.
2. Then, the web can be represented as a directed graph.
3. The structure below in figure 6.2 shows a large strongly connected component (SCC), but there were several other portions that were almost as large.
4. The two regions of approximately equal size on the two sides of CORE are named as:
    a. **IN:** Nodes that can reach the giant SCC but cannot be reached from it e.g. new web pages, and
    b. **OUT:** Nodes that can be reached from the giant SCC but cannot reach it e.g. corporate websites.
5. This structure of web is known as the **Bowtie structure**.
6. There are pages that belong to none of IN, OUT, or the giant SCC i.e. they can neither reach the giant SCC nor be reached from it.
7. These are classified as:

a. **Tendrils:** The nodes reachable from IN that can't reach the giant SCC, and the nodes that can reach OUT but can't be reached from the giant SCC. If a tendril node satisfies both conditions then it's part of a tube that travels from IN to OUT without touching the giant SCC, and
b. **Disconnected:** Nodes that belong to none of the previous categories.

8. Taken as a whole, the bow-tie structure of the Web provides a high-level view of the Web's structure, based on its reachability properties and how its strongly connected components fit together.

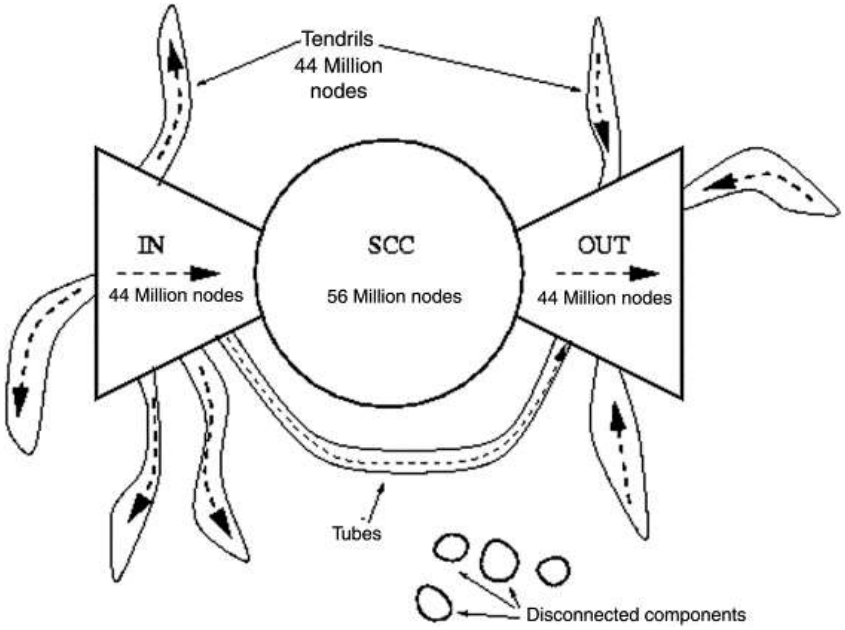



**Figure 6.2: Bow-tie structure of the web**

**Q3. What is the use of Recommender System? How is classification algorithm used in recommendation system?**

**Ans:** **[10M – DEC19]**

## RECOMMENDATION SYSTEMS:

There is an extensive class of Web applications that involve predicting user responses to options. Such a facility is called a recommendation system.

## USE OF RECOMMENDER SYSTEM:

1. Recommender systems are primarily used in **commercial applications**.
2. Recommender systems are utilized in a variety of areas and are most commonly recognized as playlist generators for video and music services like Netflix, YouTube and Spotify, product recommenders for services such as Amazon, or content recommenders for social media platforms such as Facebook and Twitter.
3. **Product Recommendations:** Perhaps the most important use of recommendation systems is at on-line retailers.
4. **Movie Recommendations:** Netflix offers its customers recommendations of movies they might like. These recommendations are based on ratings provided by users
5. **News Articles:** News services have attempted to identify articles of interest to readers, based on the articles that they have read in the past.

## CLASSIFICATION ALGORITHM:

1. Classification algorithm is completely different approach to a recommendation system using item profiles and utility matrices is to treat the problem as one of machine learning.
2. Regard the given data as a training set, and for each user, build a classifier that predicts the rating of all items.
3. A decision tree is a collection of nodes, arranged as a binary tree.
4. The leaves render decisions; in our case, the decision would be “likes” or “doesn’t like.”
5. Each interior node is a condition on the objects being classified; in our case the condition would be a predicate involving one or more features of an item.
6. To classify an item, we start at the root, and apply the predicate at the root to the item.
7. If the predicate is true, go to the left child, and if it is false, go to the right child.
8. Then repeat the same process at the node visited, until a leaf is reached.
9. That leaf classifies the item as liked or not.
10. Construction of a decision tree requires selection of a predicate for each interior node.
11. There are many ways of picking the best predicate, but they all try to arrange that one of the children gets all or most of the positive examples in the training set (i.e, the items that the given user likes, in our case) and the other child gets all or most of the negative examples (the items this user does not like)
12. Once we have selected a predicate for a node N, we divide the items into the two groups: those that satisfy the predicate and those that do not.
13. For each group, we again find the predicate that best separates the positive and negative examples in that group.
14. These predicates are assigned to the children of N.
15. This process of dividing the examples and building children can proceed to any number of levels.
16. We can stop, and create a leaf, if the group of items for a node is homogeneous; i.e., they are all positive or all negative examples.
17. However, we may wish to stop and create a leaf with the majority decision for a group, even if the group contains both positive and negative examples.

## Example:

1. Suppose our items are news articles, and features are the highTF.IDF words (keywords) in those documents.
2. Further suppose there is a user U who likes articles about baseball, except articles about the New York Yankees.
3. The row of the utility matrix for U has 1 if U has read the article and is blank if not.
4. We shall take the 1’s as “like” and the blanks as “doesn’t like.”
5. Predicates will be Boolean expressions of keywords.
6. Since U generally likes baseball, we might find that the best predicate for the root is “homerun” OR (“batter” AND “pitcher”).
7. Items that satisfy the predicate will tend to be positive examples (articles with 1 in the row for U in the utility matrix), and items that fail to satisfy the predicate will tend to be negative examples (blanks in the utility-matrix row for U).

8. Figure 6.3 shows the root as well as the rest of the decision tree.
9. Suppose that the group of articles that do not satisfy the predicate includes sufficiently few positive examples that we can conclude all of these items are in the “don’t-like” class.
10. We may then put a leaf with decision “don’t like” as the right child of the root.
11. However, the articles that satisfy the predicate includes a number of articles that user U doesn’t like; these are the articles that mention the Yankees.
12. Thus, at the left child of the root, we build another predicate.
13. We might find that the predicate “Yankees” OR “Jeter” OR “Teixeira” is the best possible indicator of an article about baseball and about the Yankees.
14. Thus, we see in Fig. 6.3 the left child of the root, which applies this predicate.
15. Both children of this node are leaves, since we may suppose that the items satisfying this predicate are predominantly negative and those not satisfying it are predominantly positive.
16. Unfortunately, classifiers of all types tend to take a long time to construct.
17. For instance, if we wish to use decision trees, we need one tree per user.
18. Constructing a tree not only requires that we look at all the item profiles, but we have to consider many different predicates, which could involve complex combinations of features.
19. Thus, this approach tends to be used only for relatively small problem sizes.

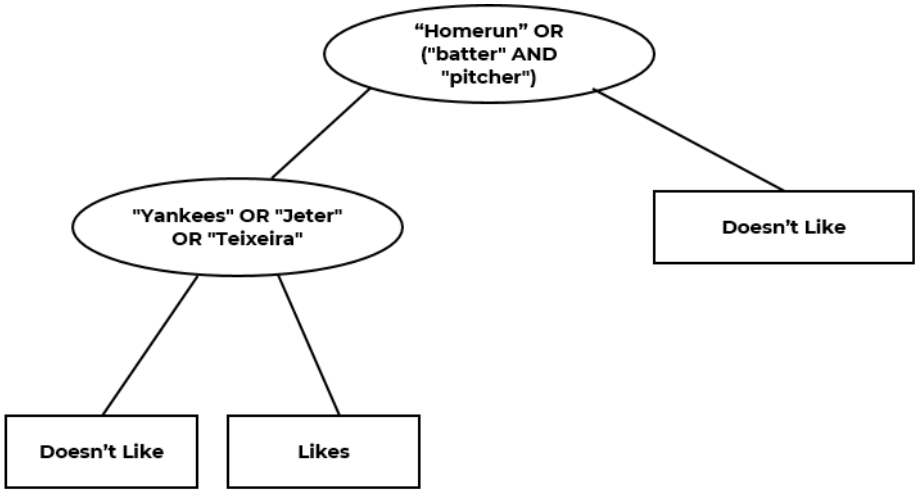
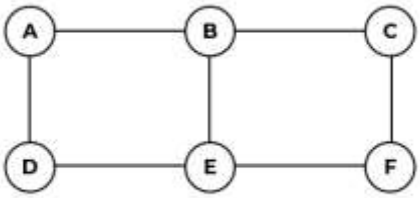



**Figure 6.3: Example of decision tree**

**Q4. Describe Girwan — Newman Algorithm. For the following graph show how the Girvan Newman algorithm finds the different communities**




**Ans:** **[10M – DEC19]**

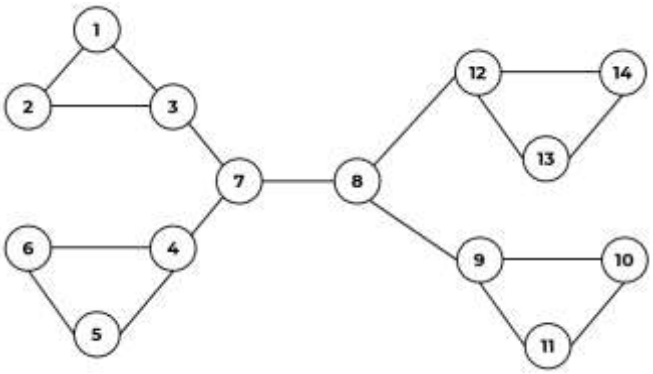
## GIRVAN NEWMAN ALGORITHM:

1. Girvan–Newman algorithm is a **hierarchical method used to detect communities in complex systems**.
2. It is published in 2002 by Michelle Girvan and mark Newman.
3. It is used for:
   a. Community detection.
   b. To measure edge - betweenness among all existing edges.
   c. To remove edge having large valued betweenness.
   d. To option optimized modular function.
4. It also checks for edge betweenness centrality and vertex betweenness centrality.
5. Vertex betweenness centrality is total number of shortest path that pass through each vertex on the network.
6. If any ambiguity found with above (vertex betweenness centrality) then every path is adjusted to equal weight 1/N among all N paths between two vertices.
7. Edge betweenness centrality is number of shortest path which pass through given edge.
8. In order to find out between edges, we need to calculate shortest paths from going through each of the edges.
9. For the removal of each edge, the calculation of edge betweenness is O(EN); therefore, this algorithm's time complexity is $O(E^{2N})$.

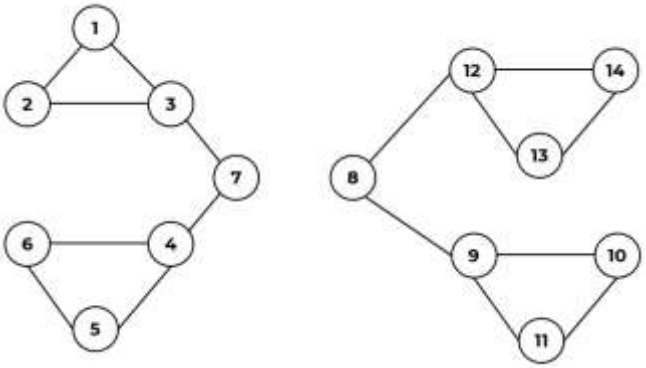
## STEPS:

1. Find edge with highest betweenness of multiple edges of highest betweenness if there is a tie and remove those edges from graph. It may affect to graph to get separate into multiple components. If so, this is first level of regions in the portioning of graph.
2. Now, recalculate all betweenness and again remove the edge or edges of highest betweenness. It will break few existing component into smaller, if so, these are regions nested within larger region.
3. Keep repeatation of tasks by recalculating all betweenness and removing the edge or edges having highest betweenness.

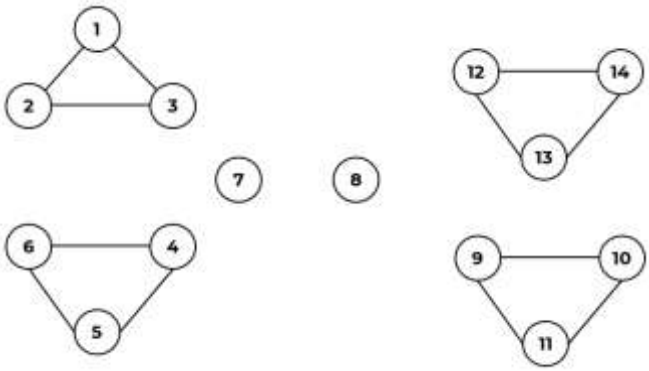
**EXAMPLE:**



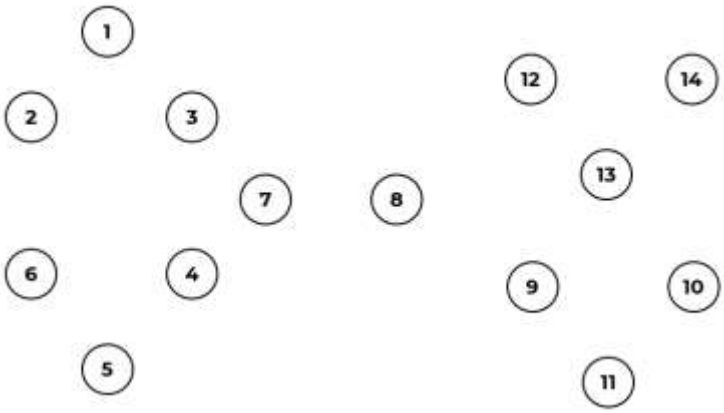

**Step 1:**




**Step 2:**



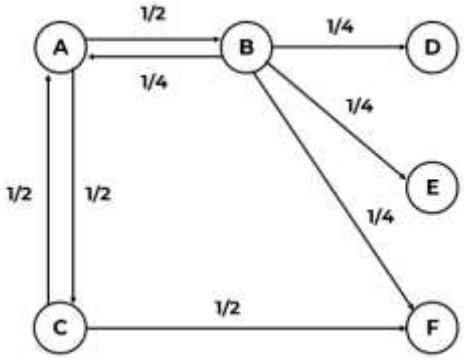

**Step 3:**



**Q5. Compute the page rank of each page after running the PageRank algorithm for two iterations with teleportation factor Beta (β) value = 0.8**

**Ans:** **[10M – DEC19]**

**Weights assigned to every edge:**




**Utility Matrix:**

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| **A** | 0 | ¼ | ½ | X | X | X |
| **B** | ½ | 0 | 0 | X | X | X |
| **C** | ½ | 0 | 0 | X | X | X |
| **D** | 0 | ¼ | 0 | X | X | X |
| **E** | 0 | ¼ | 0 | X | X | X |
| **F** | 0 | ¼ | 1/2 | X | X | X |

Here: D, E, F are dead ends and denoted as X or – in matrix

As teleportation factor, β = 0.8, we create an eigen vector V = $\begin{bmatrix} 0.8 \\ 0.8 \\ 0.8 \\ 0.8 \\ 0.8 \\ 0.8 \end{bmatrix}$

So, to calculate page rank, we need to carry out 2 iterations, as mentioned in the sum.

Let's say matrix A = $\begin{bmatrix} 0 & 1/4 & 1/2 & - & - & - \\ 1/2 & 0 & 0 & - & - & - \\ 1/2 & 0 & 0 & - & - & - \\ 0 & 1/4 & 0 & - & - & - \\ 0 & 1/4 & 0 & - & - & - \\ 0 & 1/4 & 1/2 & - & - & - \end{bmatrix}$

**<u>Iteration I:</u>**

We compute A

$A^1.V = \begin{bmatrix} 0.6 \\ 0.4 \\ 0.4 \\ 0.2 \\ 0.2 \\ 0.6 \end{bmatrix}$

**Iteration II:**

$A^2.V = A . (A^1.V)$

$$= \begin{bmatrix} 0.4 \\ 0.2 \\ 0.2 \\ 0.05 \\ 0.05 \\ 0.4 \end{bmatrix}$$

Therefore, the page rank computed is:

A =0.4, B = 0.2, C = 0.2, D= 0.05, E = 0.05, F = 0.4

## -- EXTRA QUESTIONS --

**Q1. What is Page Rank? Explain the Inverted Index**

**Ans:** **[P | Medium]**

**PAGE RANK:**

1. PageRank (PR) is an algorithm used by Google Search to rank websites in their search engine results.
2. PageRank was named after Larry Page, one of the founders of Google.
3. It is a way of measuring the importance of website pages.
4. It works by counting the number and quality of links to a page to determine a rough estimate of how important the website is.
5. The underlying assumption is that more important websites are likely to receive more links from other websites.
6. Search engine work equivalent to “Web Crawler”
7. Web Crawler is the web component whose responsibility is to identify and list down the different terms found on every web page encountered by it.
8. This listing of different terms will be stored inside the specialized data structure known as an **“Inverted Index”**
9. Figure 6.4 shows inversed index functionality.
10. Every term from the inverted index will be extracted and analyzed for the usage of that term within the web page.

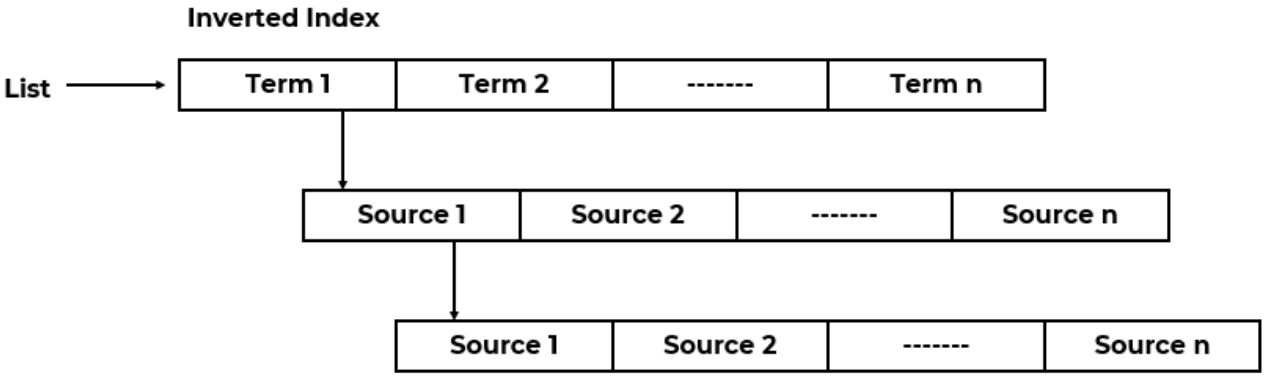



**Figure 6.4: Inversed index functionality**

**LINKS IN PAGE RANKNG:**

1. Assume we have 3 pages A, B C in a given domain of web sites.
2. As shown in figure 6.5 they have interconnection links between them.
3. Links can be **back links and forward links**.
4. Back links indicates given web page is referred by how many number of other web pages.
5. As shown in figure 6.5, A and B are back links of web page C
6. Forward links indicates how many web pages will be referred by given web page.
7. A web page which contains number of back links is said to be important web page and will get upper position in Ranking.

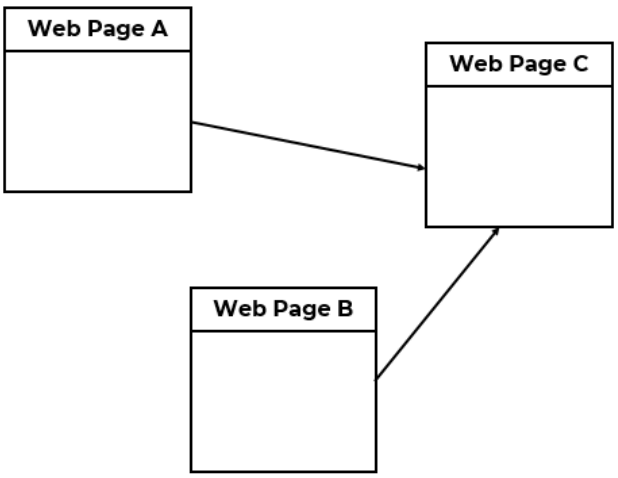



**Figure 6.5**

8. In the general case, the PageRank value for any page u can be expressed as:

$$PR(u) = \sum_{v \in B_u} \frac{PR(v)}{L(v)}$$

9. i.e. the PageRank value for a page u is dependent on the PageRank values for each page v contained in the set Bu (the set containing all pages linking to page u), divided by the number L(v) of links from page v.

---

**Q2. Define content based recommendation system. How it can be used to provide recommendations to users.**

**Ans:** **[P | Medium]**

**CONTENT BASED RECOMMENDATION:**

1. A content based recommender works with data that the user provides, either explicitly (rating) or implicitly (clicking on a link).
2. Based on that data, a user profile is generated, which is then used to make suggestions to the user.
3. As the user provides more inputs or takes actions on the recommendations, the engine becomes more and more accurate.
4. Item profile in content based systems focuses on items and user profiles in form of weighted lists.
5. Profile are helpful to discover properties of items.
6. Consider the below examples:
   a. Some students prefer to be guided by few teachers only.
   b. Some Viewers prefer drama or movie by their favorite actors only.
   c. Few Viewers prefer old songs on other hand few viewers may prefer new songs only depending upon users sorting of songs based on year.
7. In general, there are so many classes which provides such data.
8. Few domains has common features for example a college and movie it has students, professors set and actors, director set respectively.
9. Certain ratio is maintained in such cases like every college and movie has year wise datasets as movie released in a year by director and actor and college has passing student every year etc.
10. Music song album and book has same value feature like songs writers/poet, year of release and publication year etc.

11. Consider the figure 6.6 which shows recommendation system parameters.

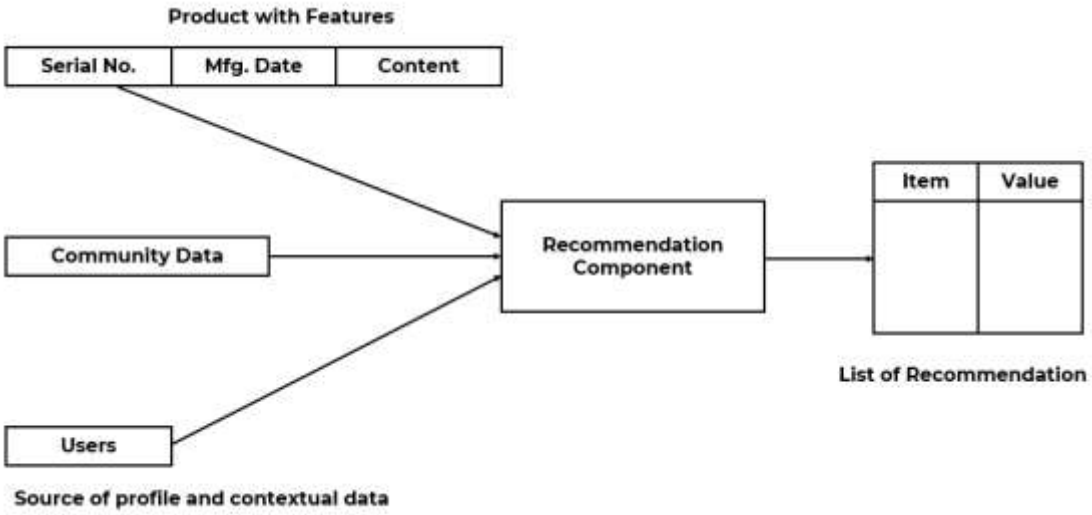



**Figure 6.6: Recommendation system parameters**

**Q3. What are different recommender systems? Explain any one with example.**

**Ans:** **[P | Medium]**

## RECOMMENDATION SYSTEM:

1. Recommendation system is widely used now-a-days.
2. It is sub class of information filtering system.
3. It is used to provide recommendation for games, movies, music, books, social tags, articles etc.
4. It is useful for experts, financial services, life insurance and social media based organization.
5. There are two types of recommendation systems:
   a) Collaborative filtering.
   b) Content based filtering.

## COLLABORATIVE FILTERING SYSTEMS:

1. It uses community data from peer groups for recommendation.
2. These exhibits all those things that are popular among the peers.
3. It uses user's past behaviour and apply some predication about user may like and accordingly post data.
4. These filtering systems recommend items based on **similarity measure between users and/or items.**
5. Here user profile and contextual parameters along with the community data are used by the recommender systems to personalize the recommendation list.
6. This is the most prominent approach in **e-commerce website**.
7. The basic assumption for collaborative filtering is:
   a. User gives ratings to item in the catalog.
   b. Customer who had similar taste in past will have similar taste in future.
   c. Users who agreed in their subjective evaluations in the past will agree in the future too.
   d. To find out similarity we can use Pearson's correlation co-efficient as:

$$sim(a,b) = \sum_p ep \frac{(ra,p-ra)(rb,p-rb)}{\sqrt{\sum_p ep(ra,p-ra)^2,(rb,p-rb)^2}}$$

Where A, b = users

Ra, p = rating of user 'a' for item 'p'

P = Set of items rated by both a and b

e. We can use this formula for prediction as :

$$pred(a,p) = ra + \frac{\sum_b ensim(a,b) \times (\eta b,p-rb)}{\sum_b ensim(a,b)}$$

**Advantages:**

1. Continuous learning for market process.
2. No knowledge engineering efforts needed.

**Disadvantages:**

1. Rating & feedback is required.
2. New items and users faces to cold start.

---

**Q4. What are social network graphs? How does clustering of social network graphs work?**

**Ans:** **[P | Medium]**

**SOCIAL NETWORK GRAPH:**

1. Social networks are naturally modeled as graphs, which we sometimes refer to as a social graph.
2. The entities are the nodes, and an edge connects two nodes if the nodes are related by the relationship that characterizes the network.
3. If there is a degree associated with the relationship, this degree is represented by labeling the edges.
4. Often, social graphs are undirected, as for the Facebook friend's graph.
5. But they can be directed graphs, as for example the graphs of followers on Twitter or Google+.
6. Consider the example of small social network as shown in figure 6.7

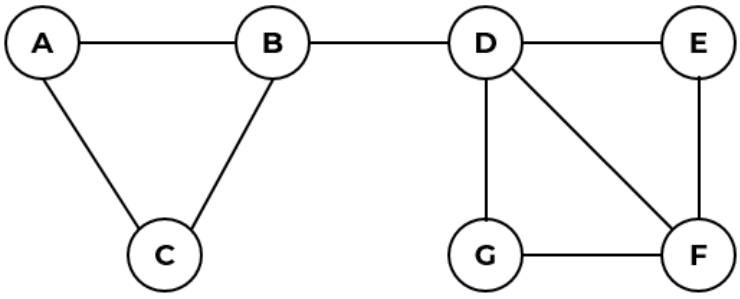



**Figure 6.7: Example of social network graph**

7. The entities are the nodes A through G.
8. The relationship, which we might think of as "friends," is represented by the edges.
9. For instance, B is friends with A, C, and D.

## CLUSTERING OF SOCIAL-NETWORK GRAPHS:

Clustering of the graph is considered as a way to identify communities. Clustering of graphs involves following steps:

### I) Distance Measures for Social-Network Graphs:

1. If we want to apply standard clustering techniques to a social-network graph, our first step would be to define a distance measure.
2. When the edges of the graph have labels, these labels might be usable as a distance measure, depending on what they represented.
3. But when the edges are unlabeled, as in a "friends" graph, there is not much we can do to define a suitable distance.
4. Our first instinct is to assume that nodes are close if they have an edge between them and distant if not.
5. Thus, we could say that the distance d(x, y) is 0 if there is an edge (x, y) and 1 if there is no such edge.
6. We could use any other two values, such as 1 and ∞, as long as the distance is closer when there is an edge.

### II) Applying Standard Clustering Methods:

1. There are two general approaches to clustering: **hierarchical (agglomerative) and point-assignment.**
2. Hierarchical clustering of a social-network graph starts by combining some two nodes that are connected by an edge.
3. Successively, edges that are not between two nodes of the same cluster would be chosen randomly to combine the clusters to which their two nodes belong.
4. The choices would be random, because all distances represented by an edge are the same.
5. In Point-assignment approach to clustering social networks, all edges are at the same distance will introduce a number of random factors that will lead to some nodes being assigned to the wrong cluster.

### III) Betweenness:

1. Since there are problems with standard clustering methods, several specialized clustering techniques have been developed to find communities in social networks.
2. The simplest one is based on finding the edges that are least likely to be inside the community.
3. Define the betweenness of an edge (a, b) to be the number of pairs of nodes x and y such that the edge (a, b) lies on the shortest path between x and y.
4. To be more precise, since there can be several shortest paths between x and y, edge (a, b) is credited with the fraction of those shortest paths that include the edge (a, b).
5. As in golf, a high score is bad.
6. It suggests that the edge (a, b) runs between two different communities; that is, a and b do not belong to the same community.

### IV) The Girvan-Newman Algorithm:

1. Girvan–Newman algorithm is a **hierarchical method used to detect communities in complex systems**.
2. It is published in 2002 by Michelle Girvan and mark Newman.

3. It is used for:
   a. Community detection.
   b. To measure edge - betweenness among all existing edges.
   c. To remove edge having large valued betweenness.
   d. To option optimized modular function.
4. It also checks for edge betweenness centrality and vertex betweenness centrality.

**V) Using betweenness to find communities:**

1. The betweenness scores for the edges of a graph behave something like a distance measure on the nodes of the graph.
2. It is not exactly a distance measure, because it is not defined for pairs of nodes that are unconnected by an edge, and might not satisfy the triangle inequality even when defined.
3. However, we can cluster by taking the edges in order of increasing betweenness and add them to the graph one at a time.
4. At each step, the connected components of the graph form some clusters.
5. The higher the betweenness we allow, the more edges we get, and the larger the clusters become.
6. More commonly, this idea is expressed as a process of edge removal.
7. Start with the graph and all its edges; then remove edges with the highest betweenness, until the graph has broken into a suitable number of connected components.

**DEC-2019** **[3 Hours] – [80 Marks]**

Q1. (a) Explain Edit distance measure with an example. [05]

**Ans: [Chapter No. 05 | Page No. 40]**

(b) When it comes to big data how NoSQL scores over RDBMS. [05]

**Ans: [Chapter No. 03 | Page No. 25]**

(c) Give difference between Traditional data management and analytics approach Versus Big data Approach [05]

**Ans: [Chapter No. 01 | Page No. 01]**

(d) Give Applications of Social Network Mining [05]

**Ans: [Chapter No. 06 | Page No. 48]**

Q2. (a) What is Hadoop? Describe HDFS architecture with diagram. [10]

**Ans: [Chapter No. 02 | Page No. 17]**

(b) Explain with block diagram architecture of Data Stream Management System. [10]

**Ans: [Chapter No. 04 | Page No. 33]**

Q3. (a) What is the use of Recommender System? How is classification algorithm used in recommendation system? [10]

**Ans: [Chapter No. 06 | Page No. 50]**

(b) Explain the following terms with diagram [10]

1) Hubs and Authorities

2) Structure of the Web

**Ans: [Chapter No. 06 | Page No. 48]**

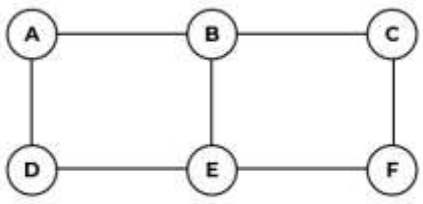
Q4. (a) What do you mean by Counting Distinct Elements in a stream. Illustrate with an example working of a Flajolet — Martin Algorithm used to count number of distinct elements. [10]

**Ans: [Chapter No. 04 | Page No. 34]**

(b) Explain different ways by which big data problems are handled by NoSQL. [10]

**Ans: [Chapter No. 03 | Page No. 26]**

Q5. (a) Describe Girwan — Newman Algorithm. For the following graph show how the Girvan Newman algorithm finds the different communities. [10]

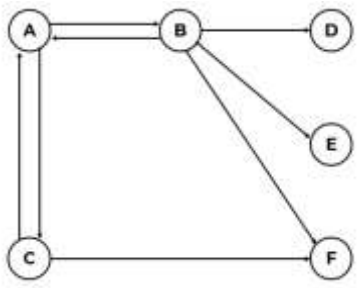



**Ans: [Chapter No. 06 | Page No. 53]**

(b) What is the role of JobTracker and TaskTracker in MapReduce. Illustrate Map Reduce execution pipeline with Word count example. [10]

**Ans: [Chapter No. 02 | Page No. 19]**

Q6. (a) Compute the page rank of each page after running the PageRank algorithm for two iterations with teleportation factor Beta (β) value = 0.8 [10]




**Ans: [Chapter No. 06 | Page No. 55]**

(b) What are the challenges in clustering of Data streams. Explain stream clustering algorithm in detail. [10]

**Ans: [Chapter No. 05 | Page No. 41]**

---

**Note: We have tried to cover almost every important question(s) listed in syllabus. If you feel any other question is important and it is not cover in this solution then do mail the question on support@BackkBenchers.com or Whatsapp us on +91-9930038388 / +91-7507531198**