SYLLABUS

Course Code	Course/Subject Name	Credits
CSDLO7032	Big Data Analytics	4

Course Objectives (CO):

1. To provide an overview of an exciting growing field of big data analytics.
2. To introduce programming skills to build simple solutions using big data technologies such as MapReduce and scripting for NoSQL, and the ability to write parallel algorithms for multiprocessor execution.
3. To teach the fundamental techniques and principles in achieving big data analytics with scalability and streaming capability.
4. To enable students to have skills that will help them to solve complex real-world problems in for decision support.
5. To provide an indication of the current research approaches that is likely to provide a basis for tomorrow's solutions.

Course Outcomes : Students should be able to -

1. Understand the key issues in big data management and its associated applications for business decisions and strategy.
1. Develop problem solving and critical thinking skills in fundamental enabling techniques like Hadoop, Mapreduce and NoSQL in big data analytics.
2. Collect, manage, store, query and analyze various forms of Big Data.
3. Interpret business models and scientific computing paradigms, and apply software tools for big data analytics.
4. Adapt adequate perspectives of big data analytics in various applications like recommender systems, social media applications etc.
5. Solve Complex real world problems in various applications like recommender systems, social media applications, health and medical systems, etc.

Pre-requisites : Some prior knowledge about Java programming, Basics of SQL, Data mining and machine learning methods would be beneficial.

Module	Detailed Contents	Hrs.
01	**Introduction to Big Data and Hadoop** 1.1 Introduction to Big Data, 1.2 Big Data characteristics, Types of Big Data, 1.3 Traditional vs. Big Data business approach, 1.4 Case Study of Big Data Solutions. 1.5 Concept of Hadoop 1.6 Core Hadoop Components; Hadoop Ecosystem **(Refer Chapters 1 and 2)**	06

02	**Hadoop HDFS and MapReduce** 2.1 Distributed File Systems : Physical Organization of Compute Nodes, Large-Scale File-System Organization. 2.2 MapReduce : The Map Tasks, Grouping by Key, The Reduce Tasks, Combiners, Details of MapReduce Execution, Coping With Node Failures. 2.3 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 2.4 Hadoop Limitations **(Refer Chapter 3)**	10
03	**NoSQL** 3.1 Introduction to NoSQL, NoSQL Business Drivers, 3.2 NoSQL Data Architecture Patterns: Key-value stores, Graph stores, Column family (Bigtable)stores, Document stores, Variations of NoSQL architectural patterns, NoSQL Case Study 3.3 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. **(Refer Chapter 4)**	06
04	**Mining Data Streams** 4.1 The Stream Data Model : A Data-Stream-Management System, Examples of Stream Sources, Stream Queries, Issues in Stream Processing. 4.2 Sampling Data techniques in a Stream 4.3 Filtering Streams : Bloom Filter with Analysis. 4.4 Counting Distinct Elements in a Stream, Count-Distinct Problem, Flajolet-Martin Algorithm, Combining Estimates, Space Requirements 4.5 Counting Frequent Items in a Stream, Sampling Methods for Streams, Frequent Itemsets in Decaying Windows. 4.6 Counting Ones in a Window: The Cost of Exact Counts, The Datar-Gionis-Indyk-Motwani Algorithm, Query Answering in the DGIM Algorithm, Decaying Windows. **(Refer Chapter 5)**	12
05	**Finding Similar Items and Clustering** 5.1 Distance Measures : Definition of a Distance Measure, Euclidean Distances, Jaccard Distance, Cosine Distance, Edit Distance, Hamming Distance. 5.2 CURE Algorithm, Stream-Computing , A Stream-Clustering Algorithm, Initializing & Merging Buckets, Answering Queries **(Refer Chapters 6 and 7)**	08
06	**Real-Time Big Data Models** 6.1 PageRank Overview, Efficient computation of PageRank : PageRank Iteration Using MapReduce, Use of Combiners to Consolidate the Result Vector. 6.2 A Model for Recommendation Systems, Content-Based Recommendations, Collaborative Filtering. 6.3 Social Networks as Graphs, Clustering of Social-Network Graphs, Direct Discovery of Communities in a social graph. **(Refer Chapters 8, 9 and 10)**	10

Module - 1

Syllabus :

Introduction to Big Data, Big Data characteristics, Types of Big Data, Traditional vs. Big Data business approach, Case Study of Big Data Solutions, Concept of Hadoop, Core Hadoop Components, Hadoop Ecosystem

Module - 2

Syllabus :

Distributed File Systems : Physical Organization of Compute Nodes, Large-Scale File-System Organization, MapReduce : The Map Tasks, Grouping by Key, The Reduce Tasks, Combiners, Details of MapReduce Execution, Coping With Node Failures, 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, Hadoop Limitations

Module - 3

Syllabus :

Introduction to NoSQL, NoSQL Business Drivers, NoSQL Data Architecture Patterns: Key-value stores, Graph stores, Column family (Bigtable)stores, Document stores, Variations of NoSQL architectural patterns; NoSQL Case Study, 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.

Module - 4

Syllabus :

The Stream Data Model: A Data-Stream-Management System, Examples of Stream Sources, Stream Queries, Issues in Stream Processing, Sampling Data techniques in a Stream, Filtering Streams: Bloom Filter with Analysis, Counting Distinct Elements in a Stream, Count-Distinct Problem, Flajolet-Martin Algorithm, Combining Estimates, Space Requirements, Counting Frequent Items in a Stream, Sampling Methods for Streams, Frequent Itemsets in Decaying Windows, Counting Ones in a Window: The Cost of Exact Counts, The Datar-Gionis-Indyk-Motwani Algorithm, Query Answering in the DGIM Algorithm, Decaying Windows.

Module - 5

Syllabus :

Distance Measures : Definition of a Distance Measure, Euclidean Distances, Jaccard Distance, Cosine Distance, Edit Distance, Hamming Distance., CURE Algorithm, Stream-Computing, A Stream-Clustering, Algorithm, Initializing & Merging Buckets, Answering Queries

Module - 6

Syllabus :

PageRank Overview, Efficient computation of PageRank : PageRank Iteration Using MapReduce, Use of Combiners to Consolidate the Result Vector, A Model for Recommendation Systems, Content-Based Recommendations, Collaborative Filtering, Social Networks as Graphs, Clustering of Social-Network Graphs, Direct Discovery of Communities in a social graph

Chapter 10 : Mining Social Network Graph 10-1 to 10-14

❑❑❑

CHAPTER 1

Module - 1

Introduction to Big Data

Syllabus

Introduction to Big Data, Big Data characteristics, Types of Big Data, Traditional vs. Big Data business approach, Case Study of Big Data Solutions

1.1 Introduction to Big Data Management

- We all are surrounded by huge data. People upload/download videos, audios, images from variety of devices. Sending text messages, multimedia messages, updating their Facebook, Whatsapp, Twitter status, comments, online shopping, online advertising etc. generates huge data.
- As a result of multiple processing machines have to generate and keep huge data too. Due to this exponential growth of data, Data analysis becomes very much required task for day to day operations.
- The term 'Big Data' means huge volume, high velocity and a variety of data.
- This big data is increasing tremendously day by day.
- Traditional data management systems and existing tools are facing difficulties to process such a Big Data.
- R is one of the main computing tools used in statistical education and research. It is also widely used for data analysis and numerical computing in other fields of scientific research.

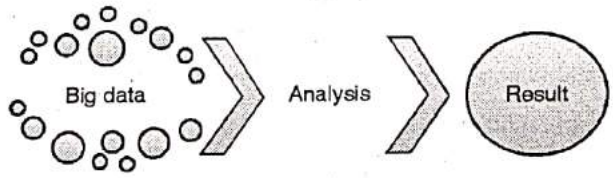


Fig. 1.1.1

1.2 Big Data

MU - May 17

Q What is Big Data ? (May 17, 2 Marks)

Q. Write a short note on Big Data.

- We all are surrounded by huge data. People upload/download videos, audios, images from variety of devices.
- Sending text messages, multimedia messages, updating their Facebook, Whatsapp, Twitter status, comments, online shopping, online advertising etc.
- Big data generates huge amount data.
- As a result machines have to generate and keep huge data too. Due to this exponential growth of data the analysis of that data becomes challenging and difficult.

- The term 'Big Data' means huge volume, high velocity and a variety of data. This big data is increasing tremendously day by day. Traditional data management systems and existing tools are facing difficulties to process such a Big Data.
- Big data is the most important technologies in modern world. It is really critical to store and manage it. Big is a collection of large datasets that cannot be processed using traditional computing techniques.
- Big Data includes huge volume, high velocity and extensible variety of data. The data in it may be structured data, Semi Structured data or unstructured data. Big data also involves various tools, techniques and frameworks.

1.3 Big Data Characteristics - Four Important V of Big Data

MU - May 16, Dec. 16, Dec. 17

Q. What are the three Vs of big data? **(May 16, Dec. 16, 2 Marks)**

Q. Describe any five characteristics of Big Data. **(Dec. 17, 5 Marks)**

Big data characteristics are as follows :

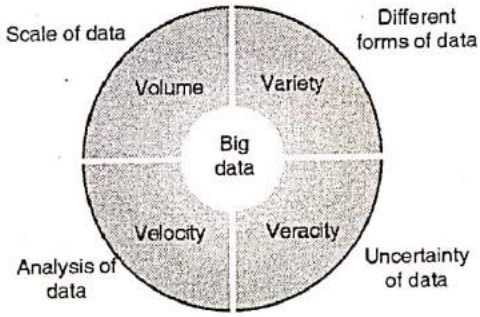


Fig. 1.3.1 : Big data characteristics

1. Volume

- Huge amount of data is generated during big data applications.
- The amount of data generated as well as storage volume is very big in size.

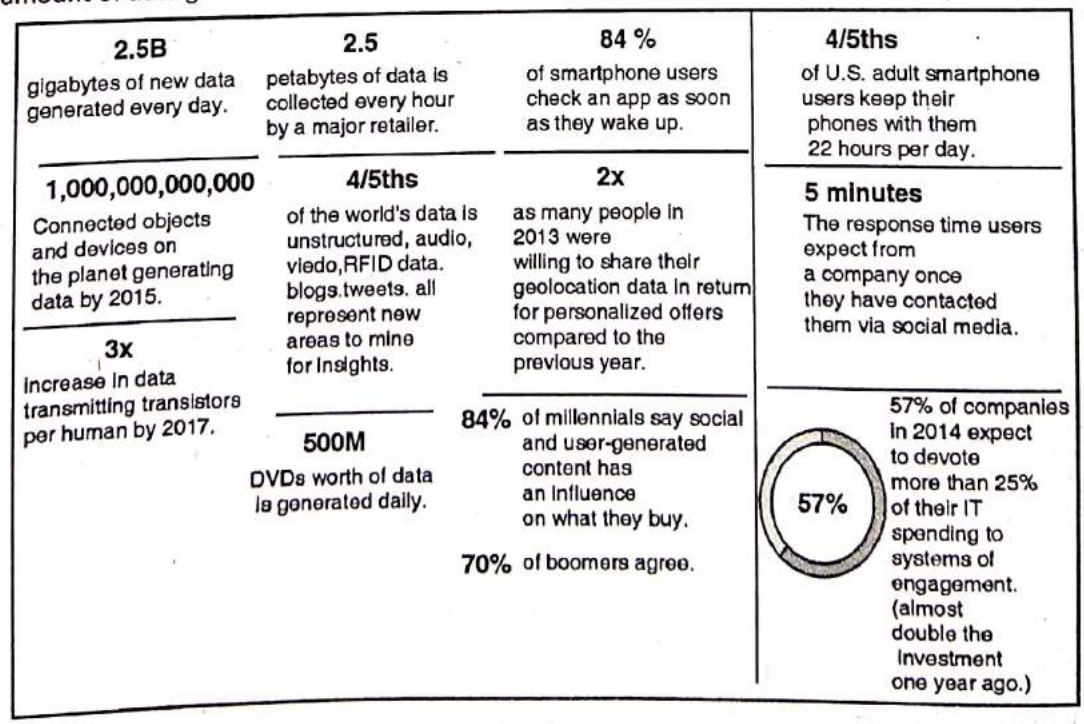


Fig. 1.3.2

2. Velocity

- For time critical applications the faster processing is very important. E.g. share marketing, video streaming
- The huge amount of data is generated and stored requires higher processing speed of processing data.
- The amount of digital data will be doubled in every 18 months and it repeats may be in less time in future.

3. Variety

The type and nature of data is having great variety.

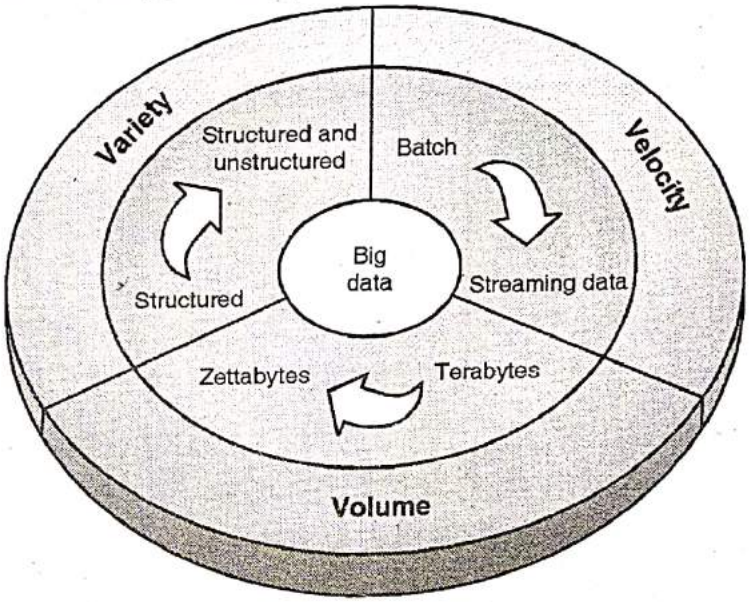


Fig. 1.3.3

4. Veracity

- The data captured is is not in certain format.
- Data captured can vary greatly.
- So accuracy of analysis depends on the veracity of the source data.

5. Additional characteristics

(a) Programmable

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

(b) Data driven

- The data driven approach is possible for scientists.
- As data collected is huge amounts.

(c) Multi Attributes

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

(d) Iterative

The more computing power can iterate on your models until you get them as per your own requirements.

1.4 Types of Big Data

Q. Write a short note on Types of Big Data.

1. Introduction

Hadoop, Hive, Pig, Cascading, Cascalog, mrjob, Caffeine, S4, MapR, Acunu, Flume, Kafka, Azkaban, Oozie, Greenplum

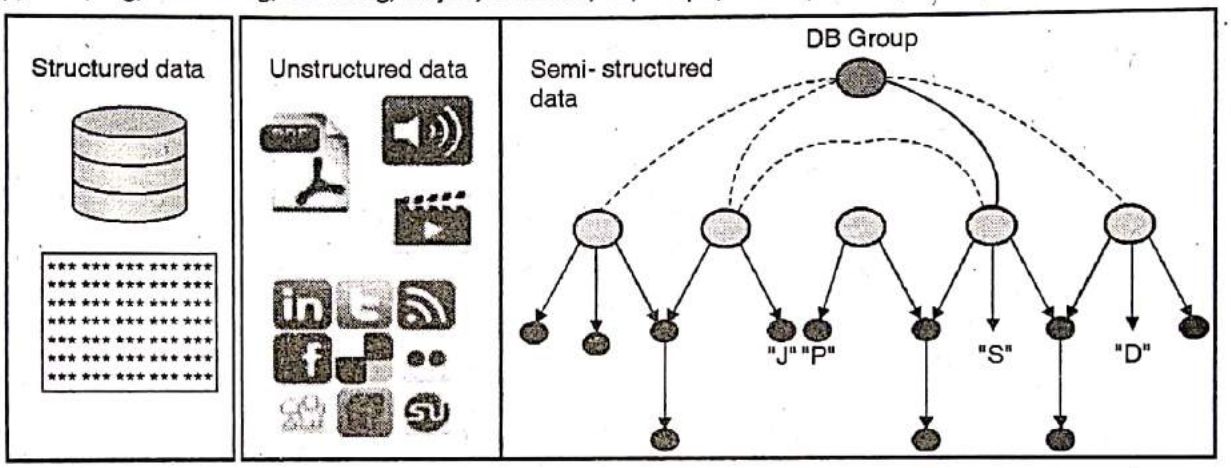


Fig. 1.4.1 : Types of big data

2. Structured data

- Structured data is generally data that has a definate length and format for big data.
- Like RDBMS tables have fixed number of colums and data can be increased by adding rows.

Example :

Structured data include marks data as numbers, dates or data like words and numbers. Structured data is very simple to dealing with, and easy to store in a database.

Sources of structured data

The data can be generated by human or it will be generated by machine.

(i) Human generated data

1. **Sensor data :** Radio frequency ID tags, medical devices, and Global Positioning System data.
2. **Web log data :** All kinds of data about their activity.
3. **Point-of-sale data :** Data associated with sales.
4. **Financial data :** Stock-trading data or banking transaction data.

(ii) Machine generated data

1. **Input data :** Survey data, responses sheets and so on.
2. **Click-stream data :** Data is generated every time you click a Link on a website.
3. **Gaming-related data :** Every move you make in a game can be recorded.

(iii) **Tools generates structured data**

(i) Data Marts (ii) RDBMS

(iii) Greenplum (iv) TeraData

3. Un-structured data

- Unstructured data is generally data collected in any available form without restricting them for any formats.
- Like audio, video data, Web blog data etc.

Example :

Unstructured data include video recording of CCTV surveillance.

Sources of unstructured data

The data can be generated by human or it will be generated by machine.

(i) **Human generated data**

1. **Satellite images :** This includes weather data or the data from satellite
2. **Scientific data :** This includes seismic imagery, weather forecasting data etc.
3. **Photographs and video :** This includes security, video etc.
4. **Radar or sonar data :** This includes vehicular, oceanographic data

(ii) Machine generated data

1. **Text data :** The documents, logs, survey results, and e-mails in company.
2. **Social media :** Generated from the social media platforms such as YouTube, Facebook etc.
3. **Mobile data :** This includes data such as text messages and location information etc.
4. **Website content :** Site data like YouTube, Flickr, or Instagram.

(iii) **Tools generates structured data**

1. Hadoop
2. HBase
3. Hive
4. Pig
5. Cloudera
6. MapR

4. Semi-structured data

- Along with structured and unstructured data, there's also a semi-structured data.
- Semi-structured data is information that doesn't reside in a RDBMS.
- It may organized in tree pattern which is easier to analyze in some cases.
- Examples of semi-structured data might include XML documents and NoSQL databases.

5. Hybrid data

- There are systems which will make use of both types of data to achieve competitive advantages.
- Structured data is offering simplicity whereas unstructured data will give lot of data about topic.

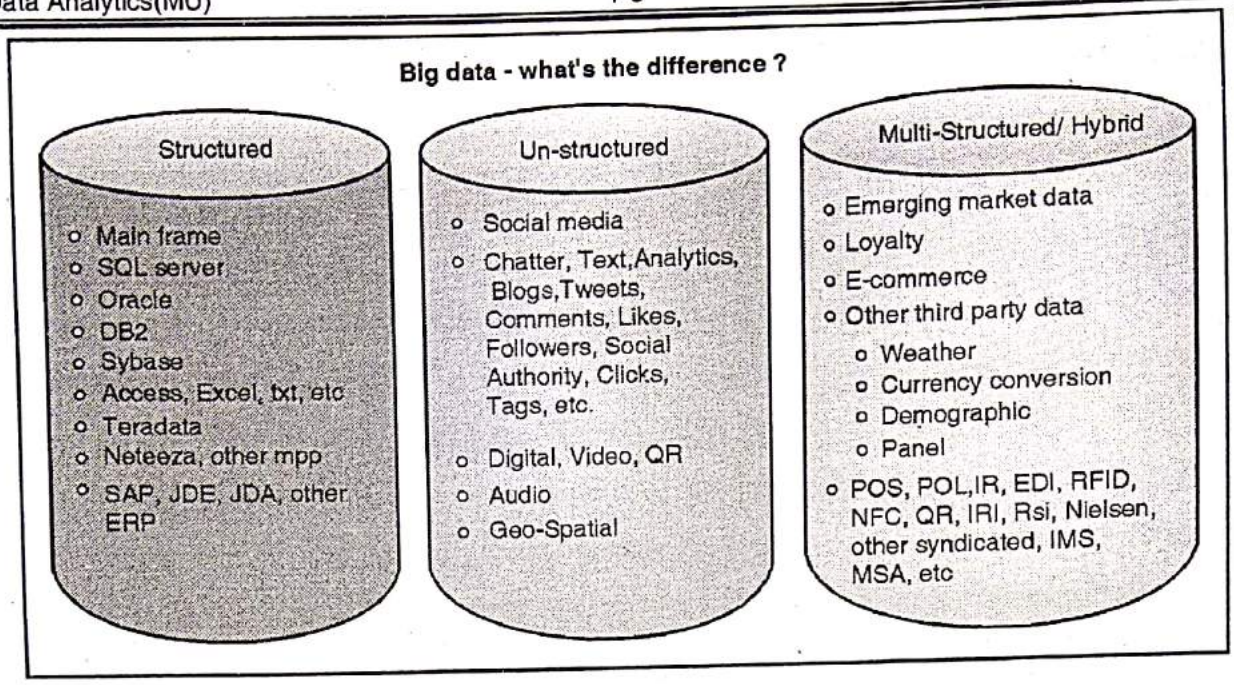


Fig. 1.4.2

1.5 Big Data vs. Traditional Data Business Approach

MU - Dec. 18

Q. Compare Traditional approach and traditional big data approach. **(Dec. 18, 3 Marks)**

The modern world is generating massive volumes of data at very fast rates. As a result, big data analytics is becoming a powerful tool for businesses looking to mining valuable data for competitive advantage.

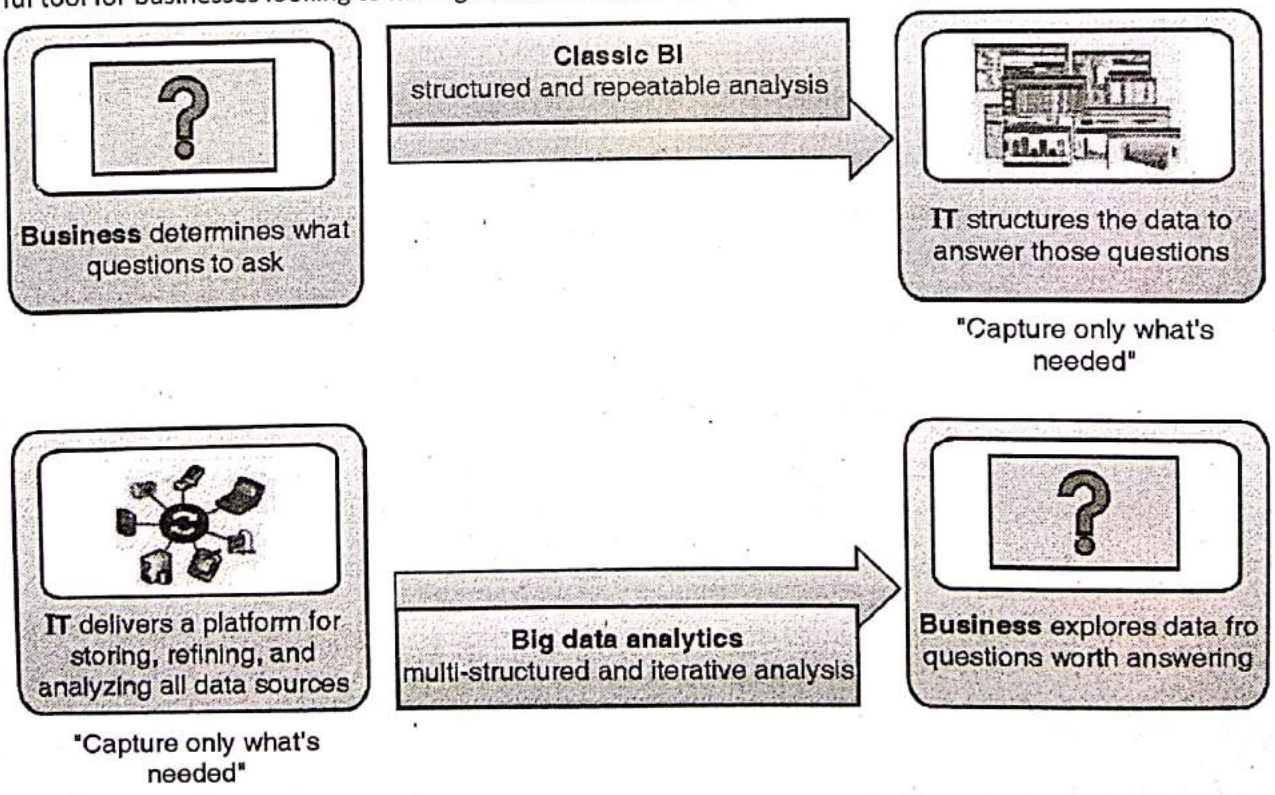


Fig. 1.5.1

1. Traditional business intelligence

- There are many systems distributed throughout the organization.
- The traditional data warehouse and business intelligent approach required extensive data analysis work with each of the systems and extensive transfer of data.
- Traditional Business Intelligence (BI) systems offer various levels and types of analyses on structured data but they are not designed to handle unstructured data.
- For these systems Big Data may creates a big problems due to data that flows in either structured or unstructured way.
- This makes them limited when it comes to delivering Big Data benefits.
- Many of the data sources are incomplete, do not use the same definitions, and not always available.
- Saving all the data from each system to a centralized location makes it unfeasible.

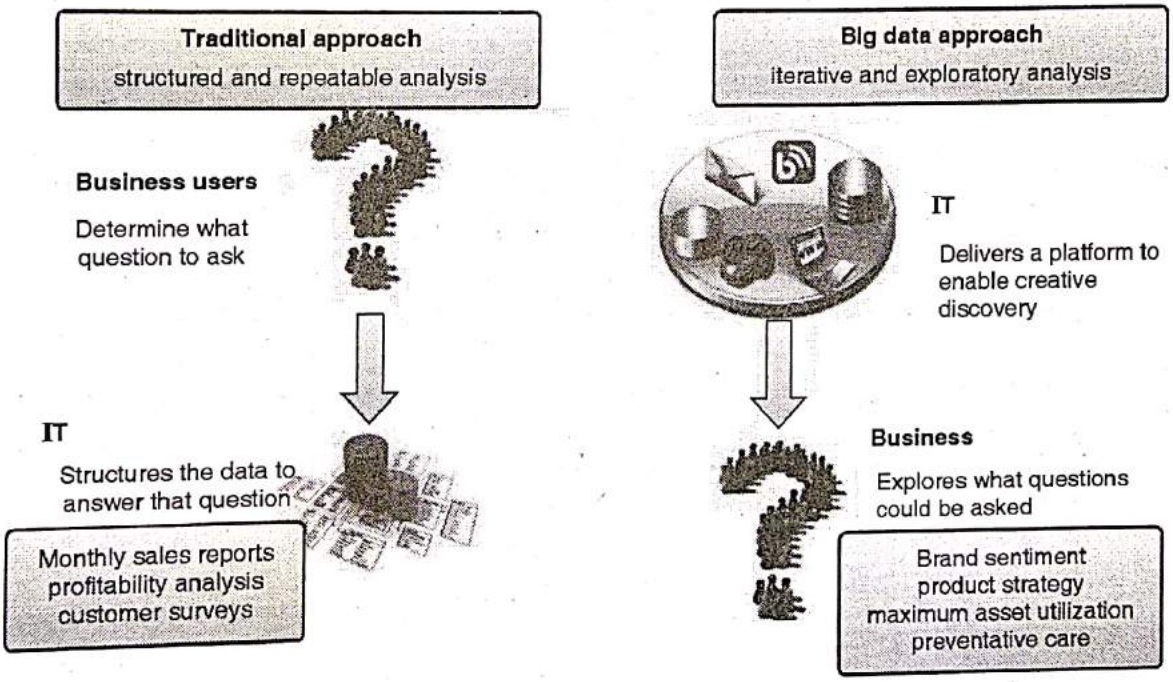


Fig. 1.5.2 : Traditional business intelligence

2. Big data analysis

- Big data means large or complex data set that traditional data processing applications may no be able to process efficientely.
- The Big data analytics involves data analysis, data capture, search, sharing, storage, transfer, visualization, querying and information security.
- The term generally only used for predictive analytics.

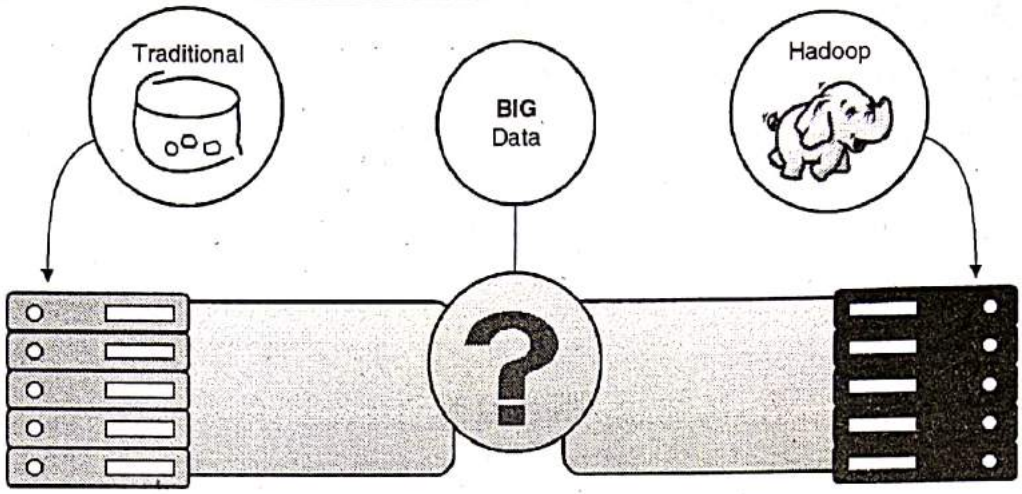


Fig. 1.5.3 : Big data analysis

- The efficiency of big data may lead to more confident decision making, and better decisions which can result in greater operational efficiency, cost reduction and reduced risk.
- Cloud-based platform can be used for the business world's big data problems.
- There can be some situations where running workloads on a traditional database may be the better solution.

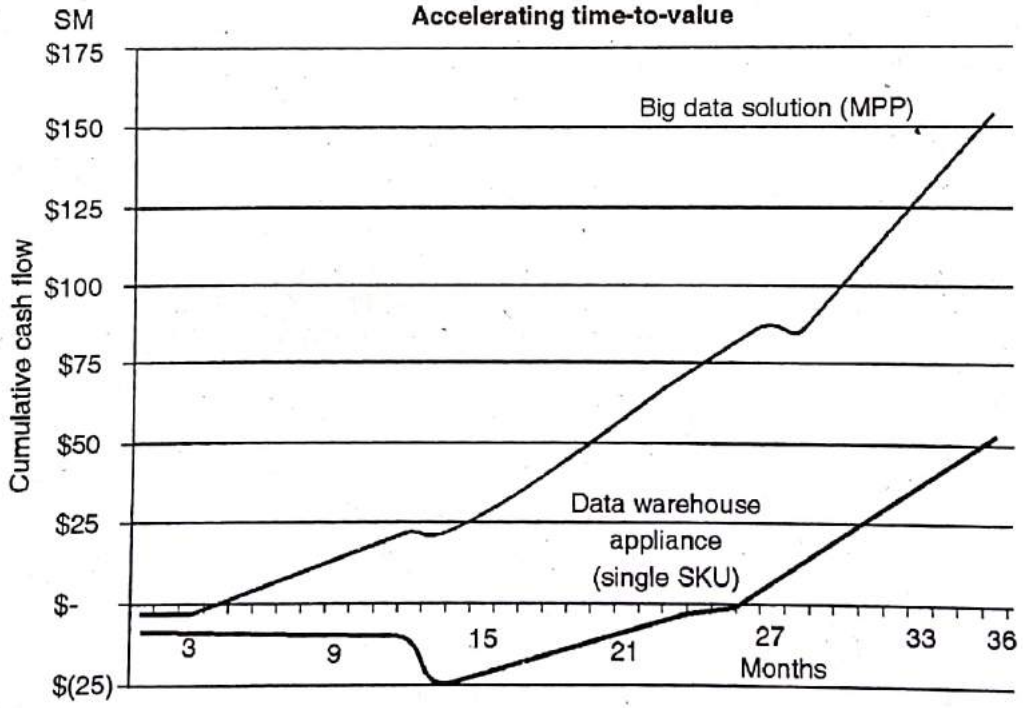


Fig. 1.5.4

3. Comparison of traditional and big data

Sr. No.	Dimension	Traditional	Big Data
1	Data source	Mainly internal.	Both inside and outside organization including traditional.
2	Data structure	Pre-defined structure.	Unstructured in nature.

Sr. No.	Dimension	Traditional	Big Data
3	Data relationship	By default, stable and interrelationship.	Unknown relationship.
4	Data location	Centralized.	Physically highly distributed.
5	Data analysis	After the complete build.	Intermediate analysis, as you go.
6	Data reporting	Mostly canned with limited and pre-defined interaction paths.	Reporting in all possible direction across the data in real time mode.
7	Cost factor	Specialized high end hardware and software.	Inexpensive commodity boxes in cluster mode.
8	CAP theorem	Consistency - Top priority.	Availability - Top priority.

4. Comparison between RDBMS and Hadoop

Sr. No.		Traditional RDBMS	Hadoop / Map Reduce
1	Data size	Gigabytes (Terabytes)	Petabytes (Hexabytes)
2	Access	Interactive and Batch	Batch – NOT interactive
3	Updates	Read/Write many times	Write once, Read many times
4	Structure	Static schema	Dynamic schema
5	Integrity	High (ACID)	Low
6	Scaling	Non linear	Linear
7	Query response time	Can be near immediate	Has latency (due to batch processing)

1.6 Tools used for Big Data

Q. Explain various tools used in Big Data.

1. **MapReduce**

   Hadoop, Hive, Pig, Cascading, Cascalog, mrjob, Caffeine, S4, MapR, Acunu, Flume, Kafka, Azkaban, Oozie, Greenplum

2. **Storage**

   S3, Hadoop Distributed File System

3. **Servers**

   EC2, Google App Engine, Elastic, Beanstalk, Heroku

4. **NoSQL**

   ZooKeeper Databases, MongoDB, Cassandra, Redis, BigTable, Hbase, Hypertable, Voldemort, Riak, CouchDB

5. **Processing**

   R, Yahoo! Pipes, Mechanical Turk, Solr/Lucene, ElasticSearch, Datameer, BigSheets, Tinkerpop

1.7 Data Infrastructure Requirements

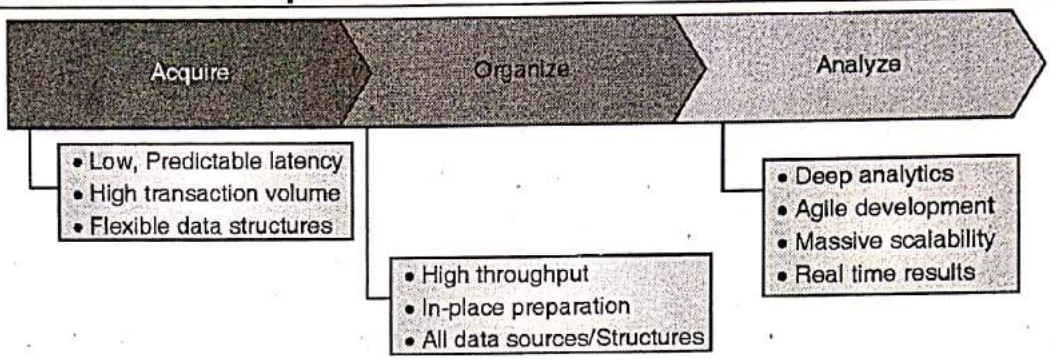


Fig. 1.7.1 : Data infrastructure requirement

1. **Acquiring data**

   High volume of data and transactions are the basic requirements of big data. Infrastructure should support the same. Flexible data structures should be used for the same. The amount of time required for this should be as less as possible.

2. **Organizing data**

   As the data may be structured, semi structured or unstructured it should be organized in a fast and efficient way.

3. **Analyzing data**

   Data analysis should be faster and efficient. It should support distributed computing.

1.8 Case Studies of Big Data Solutions

MU - May 16

Q. Give two examples of big data case studies. Indicate which vs are satisfied by these case studies. **(May 16, 4 Marks)**

There are many cases, in which big data solutions can be used effectively,

1. **Healthcare and public health industry**

   - The computing power of big data analytics enables us to predict disease, allow us to find new cures and better understand and predict disease patterns.
   - Like entire DNA strings can be decoded in minutes
   - Smart watches can be used to apply to predict symptoms for various diseases.
   - Big data techniques are already being used to monitor babies in a specialist premature and sick baby unit. By recording and analyzing every heart beat and breathing pattern of every baby, the unit was able to develop algorithms that can now predict infections 24 hours before any physical symptoms appear.
   - Big data analytics allow us to monitor and predict the developments of epidemics and disease. Integrating data from medical records with social media analytics enables us to monitor few diseases.

2. **Sports**

   - All sports popularly performing all analysis in big data analytics.
   - The Cricket and football matches you will observe many predictions which found correct maximum times.

- Example is IBM SlamTracker tool for tennis tournaments
- Also video analytics track the performance of every player in a cricket game, and sensor technology in sports equipment such as basket ball allows us to get feedback even using smart phones.
- Many elite sports teams also track athletes outside of the sporting environment using smart technology to track nutrition and sleep, as well as social media conversations.

3. Science and research

- Science and research is currently being transformed by the new possibilities big data.
- Experiments involve testing lot of possibilities of test cases and generate huge amounts of data.
- The many advanced lab uses computing powers of thousands of computers distributed across many data centers worldwide to analyze the data,
- It will help to leveraged performance areas of science and research.

4. Security enforcement

- Big data is applied for improving national security enforcement.
- The National Security Agency (NSA) in the U.S. uses big data analytics to foil terrorist plots.
- The big data techniques are used to detect and prevent cyber attacks.
- Police forces use big data tools to catch criminals and even predict criminal activity and credit card companies use big data use it to detect fraudulent transactions.

5. Financial trading

- Automated Trading and High-Frequency Trading (HFT) is a new area where big data can play a role.
- The big data algorithms can be used to make trading decisions.
- Today, the majority of equity trading now takes place via data algorithms that increasingly take into account signals from social media networks and news websites to make buy and sell decisions in split seconds.

Review Questions

Q. 1 Write a short note on Big Data.

Q. 2 Explain various applications of Big Data applications.

Q. 3 Give all characteristics of Big Data.

Q. 4 Explain three vs of Big Data.

Q. 5 Explain various types of big data in details.

Q. 6 Why to use Big Data over traditional business approach ?

Q. 7 Compare Traditional approach and traditional big data approach.

Q. 8 Explain various needs of Big Data.

Q. 9 Explain various tools used in Big Data.

Q. 10 Write a short note on :

(a) Types of Big Data

(b) Traditional vs Big Data business approach.

□□□

CHAPTER 2

Module - 1

Introduction to Hadoop

Syllabus

Concept of Hadoop, Core Hadoop Components, Hadoop Ecosystem

2.1 Hadoop

MU - May 17

Q. What is Hadoop? How Big Data andHadoop are linked? **(May 17, 4 Marks)**

Q. Write a short note on Hadoop.

- Hadoop is an open-source, big data storage and processing software framework. Hadoop stores and process big data in a distributed fashion on large clusters of commodity hardware. Massive data storage and faster processing are the two important aspects of Hadoop.

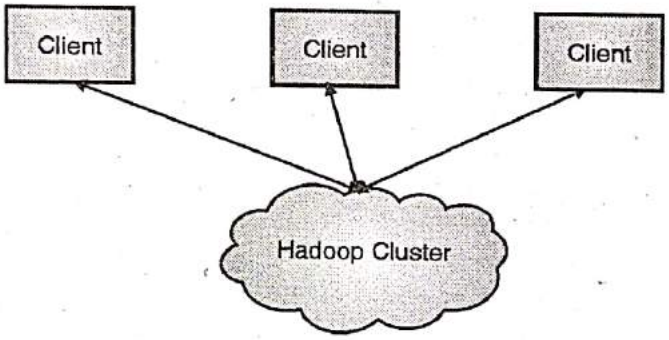


Fig. 2.1.1 : Hadoop cluster

- As shown in Fig. 2.1.1 Hadoop cluster is a set of commodity machines networked together in one location i.e. cloud.
- These cloud machines are then used for Data storage and processing. From individual clients users can submit their jobs to cluster. These clients may be present at some remote locations from the Hadoop cluster.
- Hadoop run applications on systems with thousands of nodes involving huge storage capabilities. As distributed file system is used by Hadoop data transfer rates among nodes are very faster.
- As thousands of machines are there in a cluster user can get uninterrupted service and node failure is not a big issue in Hadoop even if a large number of nodes become inoperative.
- Hadoop uses distributed storage and transfers code to data. This code is tiny and consumes less memory also.
- This code executes with data there itself. Thus the time to get data and again restore results is saved as the data is locally available. Thus interprocess communication time is saved which makes it faster processing.
- The redundancy of data is important feature of Hadoop due to which node failures are easily handled.
- In Hadoop user need not to worry about partitioning the data, data and task assignment to nodes, communication between nodes. As Hadoop handles it all, user can concentrate on data and operations on that data.

2.1.1 Hadoop - Features

1. Low cost

As Hadoop is an open-source framework, it is free. It uses commodity hardware to store and process huge data. Hence it is not much costly.

2. High computing power

Hadoop uses distributed computing model. Due to this task can be distributed amongst different nodes and can be processed quickly. Cluster have thousands of nodes which gives high computing capability to Hadoop.

3. Scalability

Nodes can be easily added and removed. Failed nodes can be easily detected. For all these activities very little administration is required.

4. Huge and flexible storage

Massive data storage is available due to thousands of nodes in the cluster. It supports both structured and unstructured data. No preprocessing is required on data before storing it.

5. Fault tolerance and data protection

If any node fails the tasks in hand are automatically redirected to other nodes. Multiple copies of all data are automatically stored. Due to this even if any node fails that data is available on some other nodes also.

2.1.2 Hadoop and Traditional RDBMS

Sr. No.	Hadoop	RDBMS
1.	Hadoop stores both structured and unstructured data.	RDBMS stores data in a structural way.
2.	SQL can be implemented on top of Hadoop as the execution engine.	SQL (Structured Query Language) is used.
3.	Scaling out is not that much expensive as machines can be added or removed with ease and little administration.	Scaling up (upgradation) is very expensive.
4.	Basic data unit is key/value pairs.	Basic data unit is relational tables.
5.	With MapReduce we can use scripts and codes to tell actual steps in processing the data.	With SQL we can state expected result and database engine derives it.
6.	Hadoop is designed for offline processing and analysis of large-scale data.	RDBMS is designed for online transactions.

2.2 Hadoop System Principles

1. Scaling out

In Traditional RDBMS it is quite difficult to add more hardware, software resources i.e. scale up. In Hadoop this can be easily done i.e. scale down.

2. Transfer code to data

In RDBMS generally data is moved to code and results are stored back. As data is moving there is always a security threat. In Hadoop small code is moved to data and it is executed there itself. Thus data is local. Thus Hadoop correlates preprocessors and storage.

3. Fault tolerance

Hadoop is designed to cope up with node failures. As large number of machines are there, a node failure is very common problem.

4. Abstraction of complexities

Hadoop provides proper interfaces between components for proper working.

5. Data protection and consistency

Hadoop handles system level challenges as it supports data consistency.

2.3 Hadoop Physical Architecture

MU - May 17, Dec. 18

Q. Explain Physical architecture of Hadoop. (May 17, Dec. 18, 4 Marks)

- Running Hadoop means running a set of resident programs. These resident programs are also known as daemons.
- These daemons may be running on the same server or on the different servers in the network.
- All these daemons have some specific functionality assigned to them. Let us see these daemons.

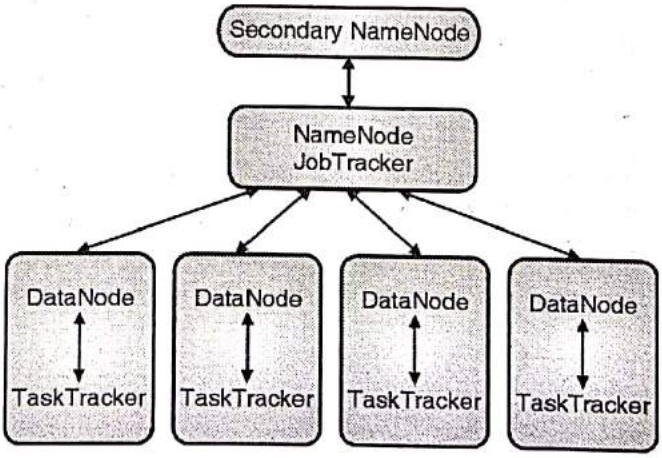


Fig. 2.3.1 : Hadoop cluster topology

NameNode

1. The NameNode is known as the master of HDFS.
2. DataNode is known as the slave of HDFS.
3. The NameNode has JobTracker which keeps track of files distributed to DataNodes.
4. NameNode directs DataNode regarding the low-level I/O tasks.
5. NameNode is the only single point of failure component.

DataNode

1. DataNode is known as the slave of HDFS.
2. The DataNode takes client block addresses from NameNodes.

3. Using this address client communicates directly with the DataNode.
4. For replication of data a DataNode may communicate with other DataNodes.
5. DataNode continually informs local change updates to NameNodes.
6. To create, move or delete blocks DataNode receives instructions from the local disk.

Secondary NameNode (SNN)

1. State monitoring of cluster HDFS is done by SNN.
2. Every cluster has one SNN.
3. SNN resides on its own machine also.
4. On the same server any other DataNode or TaskTracker daemons cannot run.
5. The SNN takes snapshots of the HDFS metadata at intervals by communicating constantly with NameNode.

JobTracker

1. JobTracker determines files to process, node assignments for different tasks, tasks monitoring etc.
2. Only one JobTracker daemon per Hadoop cluster is allowed.
3. JobTracker runs on a server as a master node of the cluster.

TaskTracker

1. Individual tasks assigned by JobTracker are executed by TaskTracker.
2. There is a single TaskTracker per slave node.
3. TaskTracker may handle multiple tasks parallelly by using multiple JVMs.
4. TaskTracker constantly communicates with the JobTracker. Within a specified amount of time if the TaskTracker fails to respond to JobTracker then it is assumed that the TaskTracker has crashed. Rescheduling of corresponding tasks are done to other nodes in the cluster.

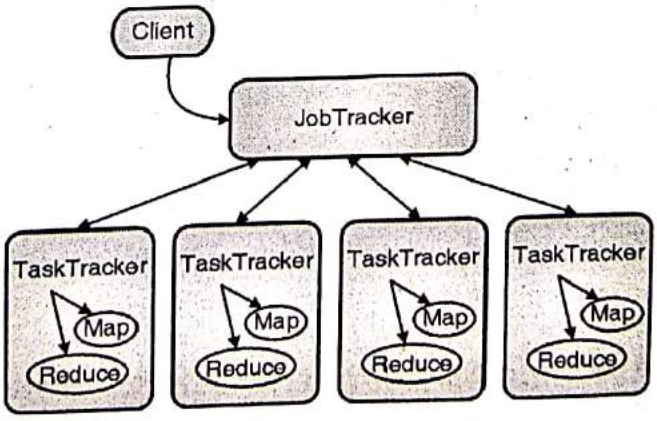


Fig. 2.3.2 : Jobtracker and tasktracker interaction

2.4 Hadoop Core Components

Q. Explain components of Core Hadoop.

- HDFS is a file system for Hadoop.
- It runs on clusters on commodity hardware.
  - HDFS - Hadoop distributed file system (storage)
  - Map reduce (processing)

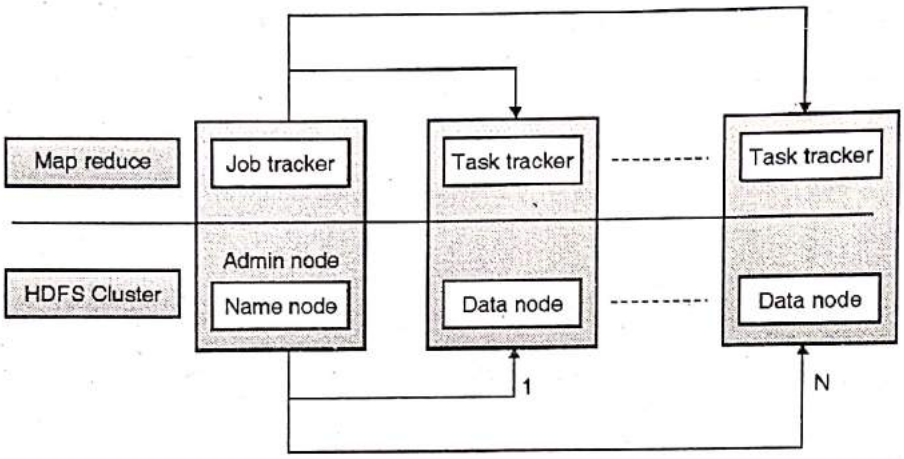


Fig. 2.4.1 : Hadoop core components

2.4.1 HDFS (Hadoop Distributed File System)

MU - Dec 17

Q. Describe the structure of HDFS in a hadoop ecosystem using a diagram. (Dec 17, 5 Marks)

- HDFS is a file system for Hadoop.
- It runs on clusters on commodity hardware.
- HDFS has following important characteristics :
  - Highly fault-tolerant
  - High throughput
  - Supports application with massive data sets
  - Streaming access to file system data
  - Can be built out of commodity hardware.

HDFS Architecture

- For distributed storage and distributed computation Hadoop uses a master/slave architecture. The distributed storage system in Hadoop is called as the Hadoop Distributed File System or HDFS. In HDFS a file is chopped into 64MB chunks and then stored, known as blocks.
- As previously discussed HDFS cluster has Master (NameNode) and Slave (DataNode) architecture. Name Node manages the namespace of the filesystem.

- In this namespace the information regarding file system tree, metadata for all the files and directories in that tree etc. is stored. For this it creates two files the namespace image and the edit log and stores information in it on consistent basis.
- A *client* interacts with HDFS by communicating with the Name Node and Datanodes. The user does not know about the assignment of Name Node and Data Node for functioning. i.e. Which NameNode and DataNodes are assigned or will be assigned.

1. NameNode

- The NameNode is known as the master of HDFS.
- DataNode is known as the slave of HDFS.
- The NameNode has Job Tracker which keeps track of files distributed to DataNodes.
- NameNode directs DataNode regarding the low-level I/O tasks.
- NameNode is the only single point of failure component.

2. DataNode

- DataNode is known as the slave of HDFS.
- The DataNode takes client block addresses from NameNodes.
- Using this address client communicates directly with the DataNode.
- For replication of data a DataNode may communicate with other DataNodes.
- DataNode continually informs local change updates to NameNodes.
- To create, move or delete blocks DataNode receives instructions from the local disk.

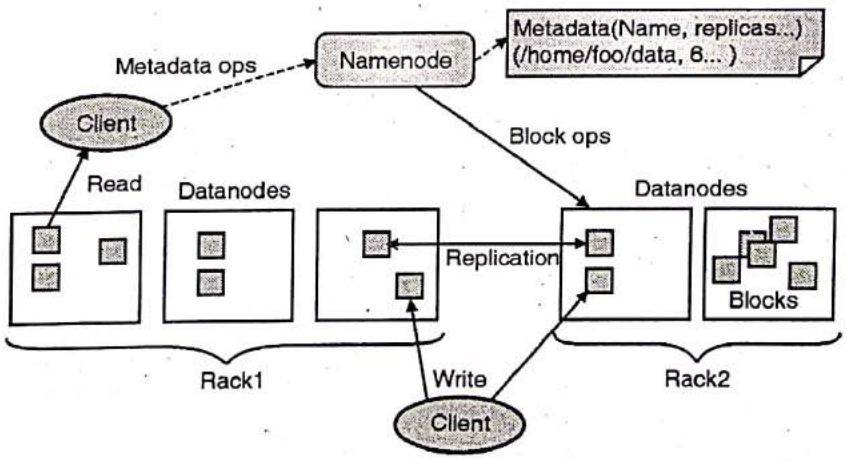


Fig. 2.4.2 : HDFS architecture

2.4.2 MapReduce

MU - May 17

Q. What is MapReduce? Explain How MapReduce work ? (May 17, 8 Marks)

- MapReduce is a software framework. In Mapreduce an application is broken down into number of small parts.

- These small parts are also as called fragments or blocks. These blocks then can be run on any node in the cluster.
- Data Processing is done by MapReduce. MapReduce scales and runs an application to different cluster machines.
- Required configuration changes for scaling and running for these applications are done by MapReduce itself. There are two primitives used for data processing by Mapreduce known as ***mappers*** and ***reducers***.
- Mapping and reducing are the two important phases for executing an application program. In the mapping phase MapReduce takes the input data, filters that input data and then transforms each data element to the mapper.
- In the reducing phase, the reducer processes all the outputs from the mapper, aggregates all the outputs and then provides a final result.
- MapReduce uses *lists* and *key/value pairs* for processing of data.

MapReduce core functions

1. **Read input**

   Divides input into small parts / blocks. These blocks then get assigned to a Map function

2. **Function mapping**

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

3. **Partition, compare and sort**

   - **Partition function :** With the given key and number of reducers it finds the correct reducer.
   - **Compare function :** Map intermediate outputs are sorted according to this compare function

4. **Function reducing**

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

5. **Write output**

   Gives file output.

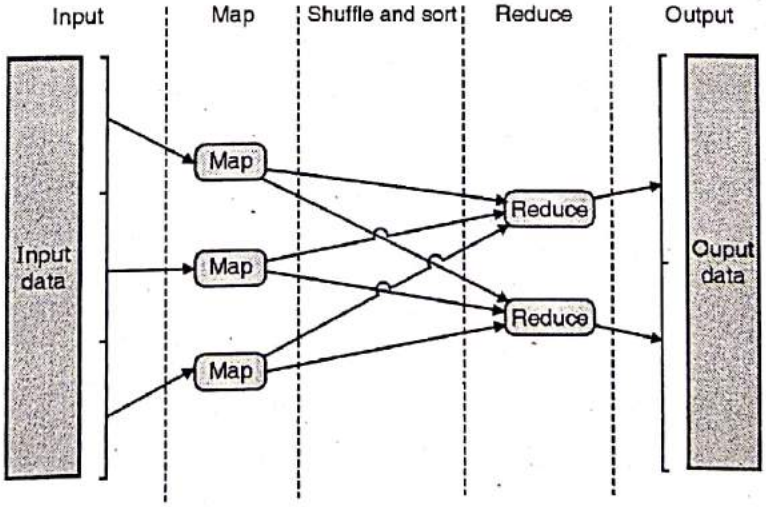


Fig. 2.4.3 : The general MapReduce dataflow

To understand how it works let us see one example,

File 1 : "Hello Sachin Hello Sumit"

File 2 : "Goodnight Sachin Goodnight Sumit"

Count occurrences of each word across different files.

Three operations will be there as follows,

(i) Map

Map1	Map2
< Hello, 1 >	< Goodnight, 1 >
< Sachin, 1 >	< Sachin, 1 >
< Hello, 1 >	< Goodnight, 1 >
< Sumit, 1 >	< Sumit, 1 >

(ii) Combine

Combine Map1	Combine Map2
< Sachin, 1 >	< Sachin, 1 >
< Sumit, 1 >	< Sumit, 1 >
< Hello, 2 >	< Goodnight, 2 >

(iii) Reduce

< Sachin, 2 >

< Sumit, 2 >

< Goodnight, 2 >

< Hello, 2 >

2.4.3 Hadoop - Limitation

MU - May 17, Dec.18

Q. State Limitations of Hadoop. **(May 17, Dec.18, 2 Marks)**

Q. What are the limitations of Hadoop?

- Hadoop can perform only batch processing and sequential access.
- Sequential access is time consuming.
- So a new technique is needed to get rid of this problem.

2.5 Hadoop - Ecosystem

MU - Dec. 16, May 17, May 18, Dec. 18

Q. Give Hadoop Ecosystem and briefly explain its components. **(Dec. 16, 10 Marks)**

Q. Explain Hadoop Ecosystem with core components. **(May 17, Dec. 18, 4 Marks)**

Q. What do you mean by the Hadoop Ecosystem?Describe any three components of a typical Hadoop Ecosystem. **(May 18, 10 Marks)**

Q. Explain Hadoop Ecosystem.

1. Introduction

- Hadoop can perform only batch processing and sequential access.
- Sequential access is time consuming.
- So a new technique is needed to get rid of this problem.
- The data in today's world is growing rapidly in size as well as scale up and shows no signs of slowing down.
- Statistics show that every year amount of data generated is more than previous years.
- The amount of unstructured data is much more than structured information stored in rows and columns.
- Big Data actually comes from complex, unstructured formats, everything from web sites, social media and email, to videos, presentations, etc.
- The pioneers in this field of data is Google, which designed scalable frameworks like MapReduce and Google File System.
- Apache open source has started with initiative by the name Hadoop, it is a framework that allows for the distributed processing of such large data sets across clusters of machines.

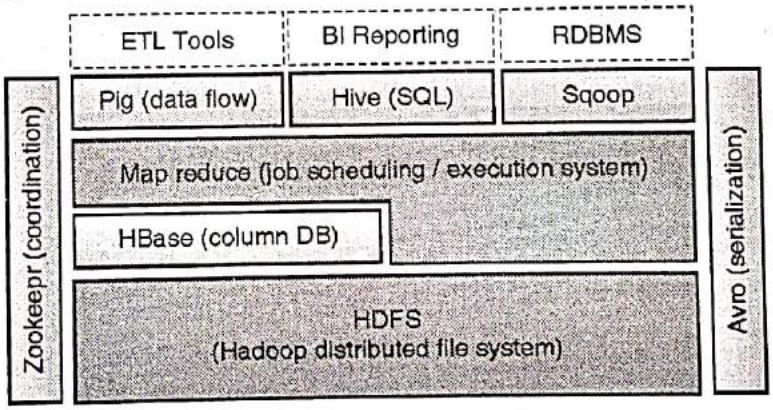


Fig. 2.5.1 : Hadoop ecosystem

2. Ecosystem

- Apache Hadoop, has 2 core projects,
  - Hadoop MapReduce
  - Hadoop Distributed File System
- Hadoop MapReduce is a programming model and software for writing applications which can process vast amounts of data in parallel on large clusters of computers.
- HDFS is the primary storage system, it creates multiple replicas of data blocks and distributes them on compute nodes throughout a cluster to enable reliable, extremely rapid computations.
- Other Hadoop-related projects are Chukwa, Hive, HBase, Mahout, Sqoop and ZooKeeper.

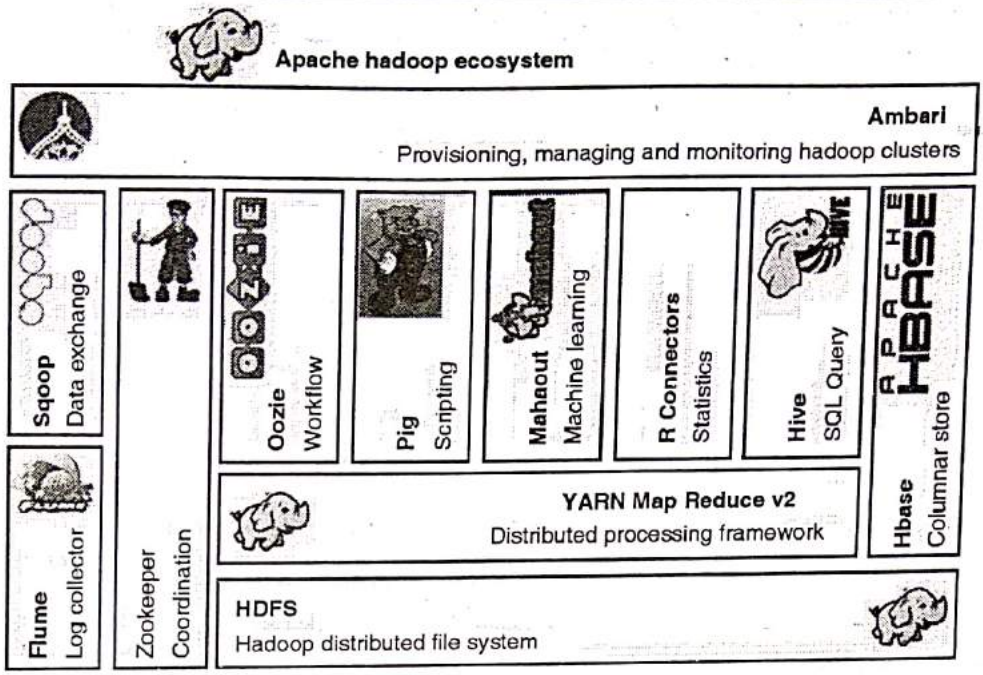


Fig. 2.5.2

2.6 ZooKeeper

1. ZooKeeper is a distributed, open-source coordination service for distributed applications used by Hadoop.
2. This system is a simple set of primitives that distributed applications can build upon to implement higher level services for synchronization, configuration maintenance, and groups and naming.

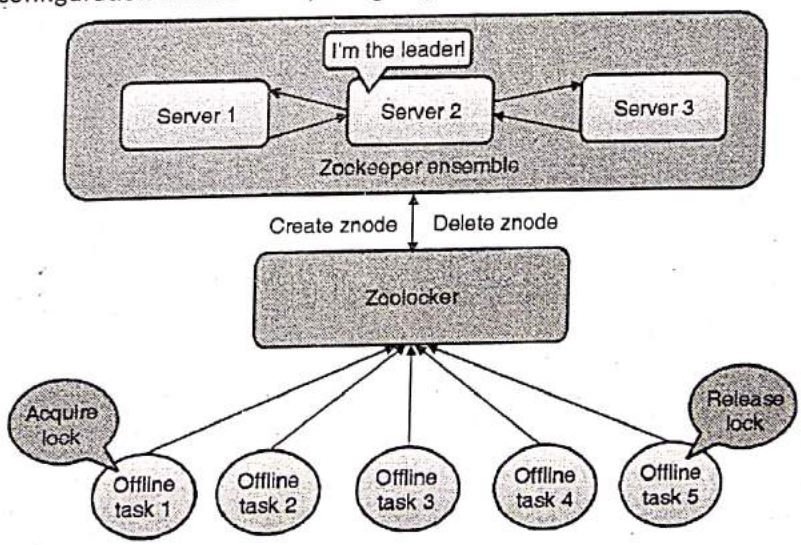
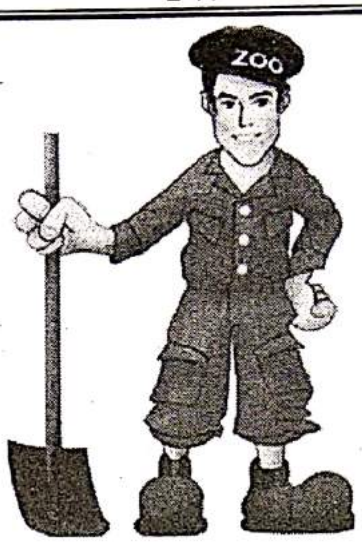


Fig. 2.6.1

3. This Coordination services are prone to errors such as race conditions and deadlock.
4. The main goal behind ZooKeeper is to use distributed applications.
5. ZooKeeper will allows distributed processes to coordinate with each other using shared hierarchical namespace organized as a standard file system.
6. The name space made up of of data registers called znodes, and these are similar to files and directories.
7. ZooKeeper data is kept in-memory, which means it can achieve high throughput and low latency.



2.7 HBase

- HBase is a distributed column-oriented database.
- HBase is hadoop application built on top of HDFS.
- HBase is suitable for huge datasets where real-time read/write random access is required.
- HBase is not a relational database. Hence does not support SQL.
- It is an open-source project and is horizontally scalable.
- Cassandra, couchDB, Dynamo and MongoDB are some other databases similar to HBase.
- Data can be entered in HDFS either directly or through HBase.
- Consistent read and writes, Automatic failure support is provided.
- It can be easily integrated with JAVA.
- Data is replicated across cluster. Useful when some node fails.

2.7.1 Comparison of HDFS and HBase

Sr. No.	HDFS	HBase
1.	HDFS is a distributed file system suitable for storing large files.	HBase is a database built on top of the HDFS.
2.	HDFS does not support fast individual record lookups.	HBase provides fast lookups for larger tables.
3.	It provides high latency batch processing.	Low latency random access.

2.7.2 Comparison of RDBMS and HBase

Sr. No.	RDBMS	HBase
1.	RDBMS uses schema. Data is stored according to its schema.	HBase is schema-less. Only column families are defined.
2.	Scaling is difficult.	Horizontally scalable.

Sr. No.	RDBMS	HBase
3.	RDBMS is transactional.	No transactions are there in HBase.
4.	It has normalized data.	It has de-normalized data.
5.	It is good for structured data.	It is good for semi-structured as well as structured data.
6.	It is row oriented database.	It is column oriented database.
7.	It is suitable for Online Transaction Process (OLTP).	It is suitable for Online Analytical Processing (OLAP).

2.7.3 HBase Architecture

- The Master performs administration, cluster management, region management, load balancing and failure handling.
- Region Server hosts and manages servers, region splitting, read/write request handling, client communication etc.
- Region contains Write Ahead Log (WAL). It may have multiple regions. Region is made up of Memstore and HFiles in which data is stored. Zookeeper is required to manage all the services.

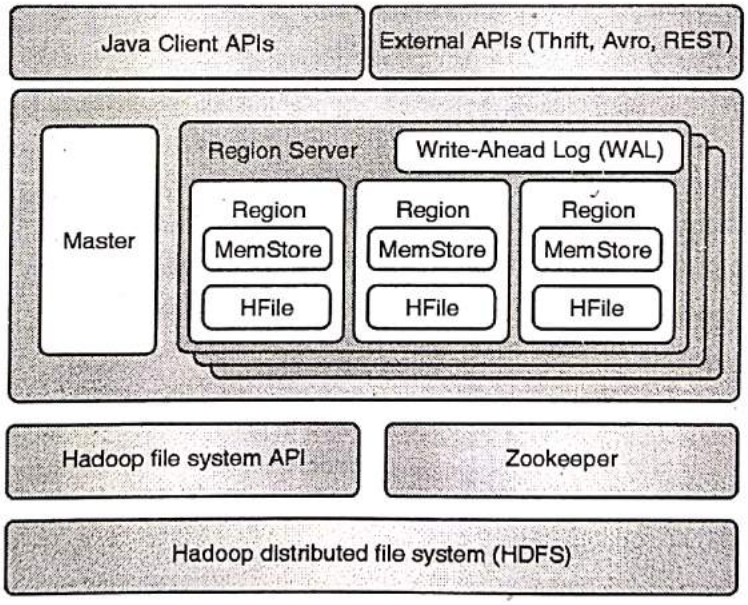


Fig. 2.7.1 : HBase database architecture

2.7.4 Region Splitting Methods

1. Pre splitting

Regions are created first and split points are assigned at the time of table creation. Initial set of region split points are to be used very carefully otherwise load distribution will be heterogeneous which may hamper clusters performance.

2. Auto splitting

This is by default action. It splits region when one of the stores crosses the max configured value.

3. Manual splitting

Split regions which are not uniformly loaded.

2.7.5 Region Assignment and Load Balancing

This information cannot be changed further as these are the standard procedures.

On startup

1. On startup Assignment Manager is invoked by Master.
2. From META the information about existing region assignments is taken by the AssignmentManager.
3. If the RegionServer is still online then the assignment is kept as it is.
4. If the RegionServer is not online then for region assignment the LoadBalancerFactory is invoked. The DefaultLoadBalancer will randomly assign the region to a RegionServer.
5. META is updated with this new RegionServer assignment. The RegionServer starts functioning upon region opening by the RegionServer.

When region server fails

1. Regions become unavailable when any RegionServer fails.
2. The Master finds which RegionServer is failed.
3. The region assignments done by that RegionServer then becomes invalid. The same process is followed for new region assignment as that of startup.

Region assignment upon load balancing

When there are no regions in transition, the cluster load is balanced by a load balancer by moving regions around. Thus redistributes the regions on the cluster. It is configured via hbase.balancer.period. The default value is 300000 (5 minutes).

2.7.6 HBase Data Model

- The Data Model in HBase is made of different logical components such as Tables, Rows, Column Families, Columns, Cells and Versions.
- It can handle semi-structured data that may be varied in terms of data type, size and columns. Thus partitioning and distributing data across the cluster is easier.

Row Key	Movies		Shows	
Screen	MovieName	Ticket	Time	Day
01	Harry Potter 1	200	6.00	Saturday
02	Harry Potter 2	250	3.00	Sunday

Column Families

Fig. 2.7.2 : HBase data model

1. Tables

Tables are stored as a logical collection of rows in Regions.

2. Rows

Each row is one instance of data. Each table row is identified by a rowkey. These rowkeys are unique and always treated as a byte[].

3. Column Families

Data in a row are grouped together as Column Families. These are stored in HFiles.

4. Columns

- A Column Family is made of one or more columns.
- A Column is accessed by, column family : columnname.
- There can be multiple Columns within a Column Family and Rows within a table can have varied number of Columns.

5. Cell

A Cell stores data as a combination of rowkey, Column Family and the Column (Column Qualifier).

6. Version

On the basis of timestamp different data versions are created. By default is the number of versions are 3 but it can be configured to some other value as well.

2.8 HIVE

- Hive is a data warehouse infrastructure tool.
- It processes structured data in HDFS. Hive structures data into tables, rows, columns and partitions.
- It resides on top of Hadoop.
- It is used to summarize big Data, analysis of big data.
- It is suitable for Online Analytical Application Processing.
- It supports ad hoc querries. It has its own SQL type language called HiveQL or HQL.
- SQL type scripts can be created for MapReduce operations using HIVE.
- Primitive datatypes like Integers, Floats, Doubles, and Strings are supported by HIVE.
- Associative Arrays, Lists, Structs etc. can be used.
- Serialize API and Deserialized API are used to store and retrieve data.
- HIVE is easy to scale and has faster processing.

2.8.1 Architecture of HIVE

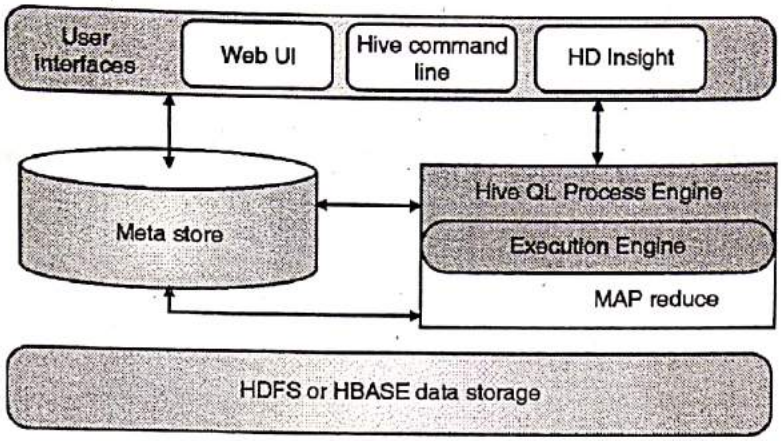


Fig. 2.8.1 : Hive architecture

This information cannot be changed further as these are the standard components.

1. **User interface**

Hive supports Hive Web UI, Hive command line, and Hive HD through which user can easily process queries.

2. **Meta store**

Hive stores Meta data, schema etc. in respective database servers known as metasores.

3. **HiveQL process engine**

HiveQL is used as querying language to get information from Metastore. It is an alternative to MapReduce Java program. HiveQL query can be written for MapReduce job.

4. **Execution engine**

Querry processing and result generation is the job of Execution engine. It is same as that of MapReduce results.

5. **HDFS or HBASE**

Hadoop distributed file system or HBASE are the data storage techniques to store data into file system.

2.8.2 Working of HIVE

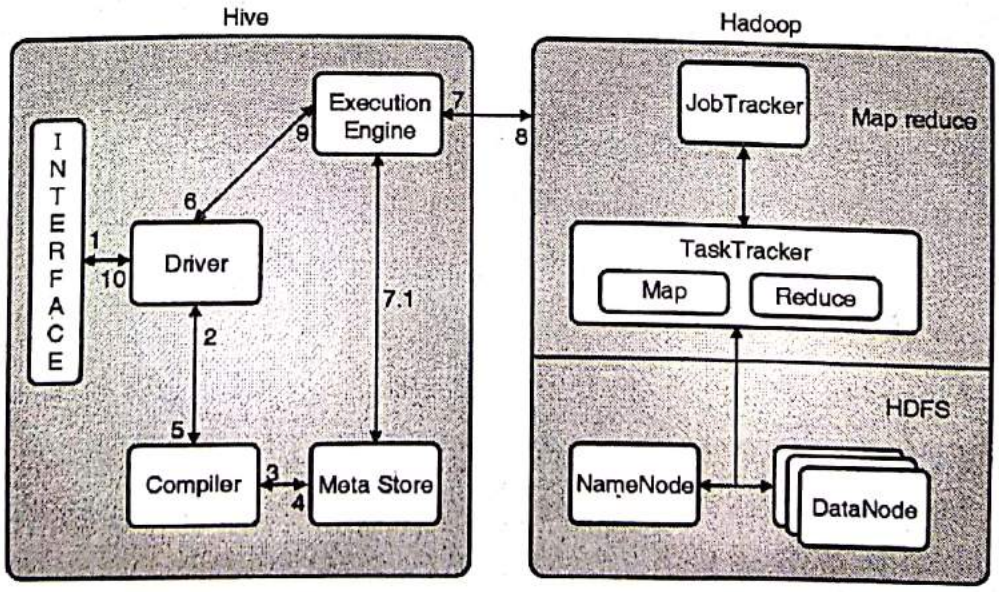


Fig. 2.8.2 : Hive and Hadoop communication

1. **Execute Query :** Command Line or Web UI sends query to JDBC or ODBC Driver to execute.
2. **Get Plan :** With the help of query compiler driver checks the syntax and requirement of query.
3. **Get Metadata :** The compiler sends metadata request to Metastore for getting data.
4. **Send Metadata :** Metastore sends the required metadata as a response to the compiler.
5. **Send Plan :** The compiler checks the requirement and resends the plan to the driver. Thus the parsing and compiling of a query is complete.
6. **Execute Plan :** The driver sends the execute plan to the execution engine.
7. **Execute Job :** The execution engine sends the job to JobTracker. JobTracker assigns it to TaskTracker.

7.1 **Metadata Operations :** The execution engine can execute metadata operations with Metastore.

8. **Fetch Result :** The execution engine receives the results from Data nodes.
9. **Send Results :** The execution engine sends those resultant values to the driver.
10. **Send Results :** The driver sends the results to Hive Interfaces.

2.8.3 HIVE Data Models

The Hive data models contain the following components,

1. Databases
2. Tables
3. Partitions
4. Buckets or clusters

Partitions

Table is divided into a smaller parts based on the value of a partition column. Then on these slices of data querries can be made for faster processing.

Buckets

Buckets give extra structure to the data that may be used for efficient queries. Different data required for querries joined together. Thus querries can be evaluated quickly.

Review Question

Q. 1 Write a short note on Hadoop.

Q. 2 What is Hadoop?

Q. 3 Explain components of Core Hadoop.

Q. 4 Explain Hadoop Ecosystem.

Q. 5 Explain physical architecture of Hadoop.

Q. 6 What are the limitations of Hadoop?

CHAPTER 3

Module - 2

Hadoop HDFS and MapReduce

Syllabus

Distributed File Systems : Physical Organization of Compute Nodes, Large-Scale File-System Organization, MapReduce : The Map Tasks, Grouping by Key, The Reduce Tasks, Combiners, Details of MapReduce Execution, Coping With Node Failures, 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, Hadoop Limitations

3.1 Distributed File Systems

- In the early days of computing, most of the computations were done on a standalone computer.
- A standalone computer is also called a compute node. It consists of a single processor along with its main memory, cache memory and a local disk.
- As the computations got more complex and the data started getting bigger, the need for more computation power was felt. This gave rise to the concept of parallel computing.
- But the problem with parallel computing is that it requires many processors and specialized hardware. This makes the whole parallel processing arrangement very costly.
- A cost-effective solution to this problem is to use the already existing standalone computers and combine their individual computing resources to create a big pool of computing resource and yet allow the compute nodes to operate more or less independently.
- This new paradigm of computing is known as cluster computing. Fig. 3.1.1 shows cluster of compute nodes in general.

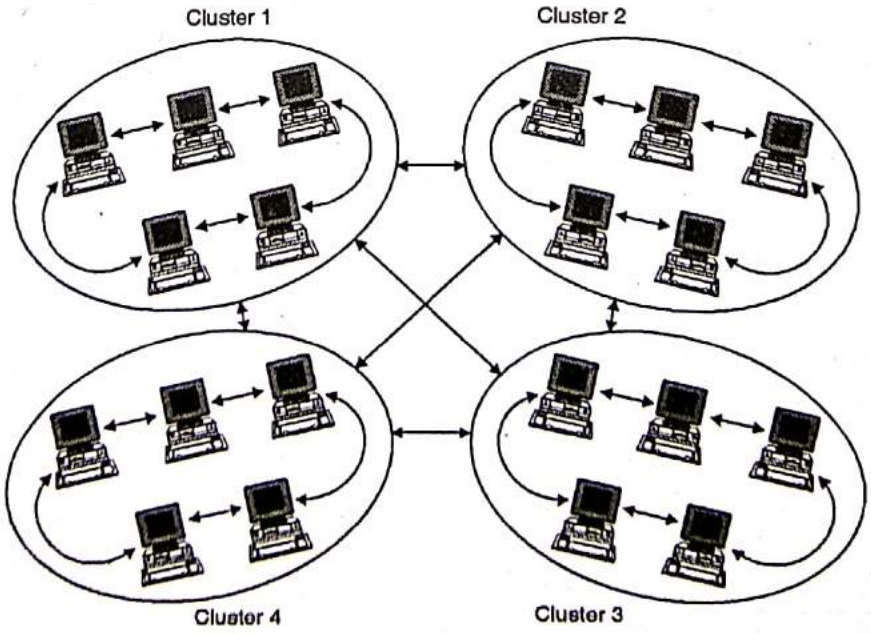


Fig. 3.1.1 : Cluster of compute nodes

3.1.1 Physical Organization of Compute Nodes

- The compute nodes are arranged in racks with each rack holding around 8 to 64 compute nodes as depicted in Fig. 3.1.2.

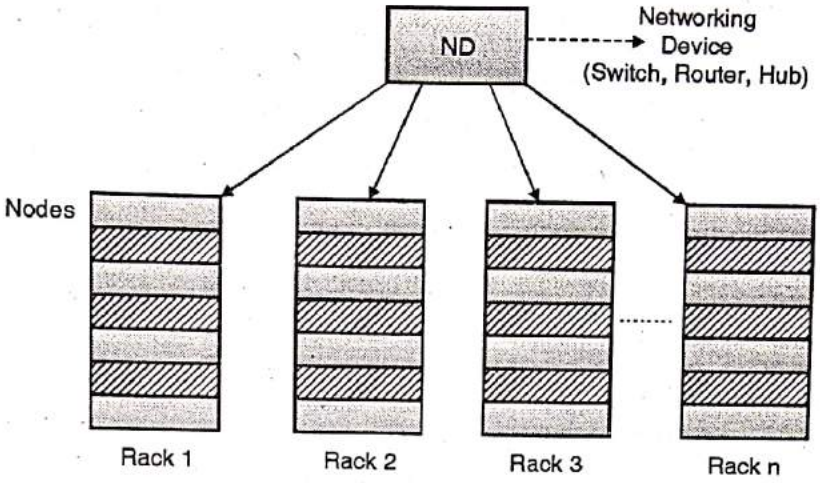


Fig. 3.1.2 : Compute nodes arranged in racks

- There are two levels of connections intra-rack and inter-rack. The compute nodes in a single rack are connected through a gigabit Ethernet and this is known as intra-rack connection. Additionally, the racks are connected to each other with another level of network or a switch which is known as the inter-rack connection.
- One major problem with this type of setup is that there are a lot of interconnected components and more the number of components, higher is the probability of failure. For example, single node failure or an entire rack failure.
- To make the system more robust against such type of failures the following steps are taken :
- Duplicate copies of files are stored at several compute nodes. This is done so that even if a compute node crashes, the file is not lost forever. This feature is known as "Data Replication". Fig. 3.1.3 shows data replication. Here, the data item D_1 is originally stored on Node 1 and a copy each is stored on Node 2 as well as Node 3. It means in total we have three copies of the same data item D_1. This is known as "Replication Factor" (RF).

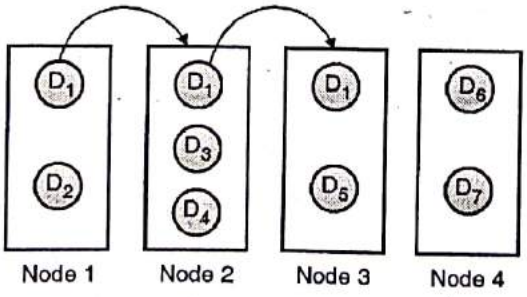


Fig. 3.1.3 : Data replication

- Computations are subdivided into tasks such that even if one task fails to complete execution, it may be restarted without affecting other tasks.

3.1.2 Large-Scale File-System Organization

- The conventional file systems present on standalone computers cannot take full advantage of cluster computing. For this reason a new type of file system is needed. This new file system is called Distributed File System or DFS.

- The types of files which are most suited to be used with DFS are :
  - Very large sized files having size in TBs or more.
  - Files with very less number of update operations compared to read and append operations.
- In DFS, files are divided into smaller units called "chunks". A chunk is usually of 64 MB in size. Each chunk is normally replicated and stored in three different compute nodes. It is also ensured that these three compute nodes are members of different racks so that in the event of rack failure at least one copy of the chunk is available.
- Both the chunk size and the replication factor can be adjusted by the user based on the demands of the application.
- All the chunks of a file and their locations are stored in a separate file called master node or name node. This file acts as an index to find the different chunks of a particular file. The master node is also replicated just like the individual chunks.
- The information about the master nodes and their replicas are stored in a directory. This directory in turn is replicated in a similar fashion and all the participants of the DFS are aware of the locations where the directory copies reside.
- There are many different implementations of the DFS described above such as :
  - Google File System (GFS),
  - Hadoop Distributed File System (HDFS),
  - Colossus, which is an improved version of GFS.

3.2 MapReduce

MU - Dec. 17, Dec. 18

Q. Explain concept of MapReduce using an example. **(Dec. 17, 5 Marks)**

Q. What is the MapReduce? Explain the role of combiner with the help of an example. **(Dec. 18, 10 Marks)**

- **MapReduce** can be used to write applications to process large amounts of data, in parallel, on large clusters of commodity hardware (a commodity hardware is nothing but the hardware which is easily available in the local market) in a reliable manner.
- **MapReduce** is a processing technique as well as a programming model for distributed computing based on java programming language or java framework.
- The MapReduce algorithm contains two important functions, namely Map and Reduce :
  - The Map tasks accept one or more chunks from a DFS and turn them into a sequence of key-value pairs. How the input data is converted into key-value pairs is determined by the code written by the user for the Map function.
  - A master controller collects and sorts the key-value pairs produced by the Map tasks. These sorted keys are then divided among the Reduce tasks. This distribution is done in such a way so that all the key-value pairs having the same key are assigned to the same Reduce task.
  - The Reduce tasks combine all of the values associated with a particular key. The code written by the user for the Reduce function determines how the combination is done.

3.2.1 The Map Tasks

- The input for a Map task is an element and the output is zero or more key-value pairs. An element could be anything such as a tuple or an entire document.
- Some useful assumptions are made which are:
  - An element is stored entirely in one chunk. That means one element cannot be stored across multiple chunks,
  - The types of keys and values both are arbitrary,
  - Keys need not be unique.
- Let us understand the MapReduce operations with an example. Let us suppose we are given a collection of documents and the task is to compute the counts of the number of times each word occurs in that collection.
- Here each document is an input element. One Map task will be assigned one or more chunks and that Map task will process all the documents in its assigned chunk(s).
- The output will be of the form:

  $(w_1, 1), (w_2, 1), (w_3, 1), \ldots, (w_n, 1)$

  Where $w_1, w_2, w_3, \ldots, w_n$ are the words in the document collection that is assigned to the Map task.
- If a particular word w appears c times, then the output will have c number of (w, 1) pairs.

3.2.2 Grouping by Key

- After the successful completion of all the Map tasks, the grouping of the key-value pairs is done by the master controller.
- The number of Reduce tasks r is set by the user in advance. The master controller uses a hash function that maps each key to the range 0 to r-1. In this step all the key-value pairs are segregated in r files according to the hash function output. These r files will be the input to the Reduce tasks.
- The master controller then performs the grouping by key procedure to produce a sequence of key/list-of-values pairs for each key k which are of the form $(k, [v_1, v_2, v_3, \ldots, v_n])$, where $(k, v_1), (k, v_2), (k, v_3), \ldots, (k, v_n)$ are the key-value pairs which were produced by all of the Map tasks.

3.2.3 The Reduce Tasks

- The input to a Reduce task is one or more keys and their list of associated values. And the output produced by the Reduce task is a sequence of zero or more key-value pairs which may be different from the key-value pairs produced by the Map tasks. But in most cases both the key-value pairs produced by the Reduce tasks and Map tasks are of the same type.
- In the final step the outputs produced by all of the Reduce tasks are combined in a single file.
- In our word count example, the Reduce tasks will add all the values for each key. The outputs will be of the form (w, s) pairs, where w is a word and sis the number of times it appears in the collection of documents.
- Fig. 3.2.1 shows the varlous MapReduce phases for the word frequency counting example.

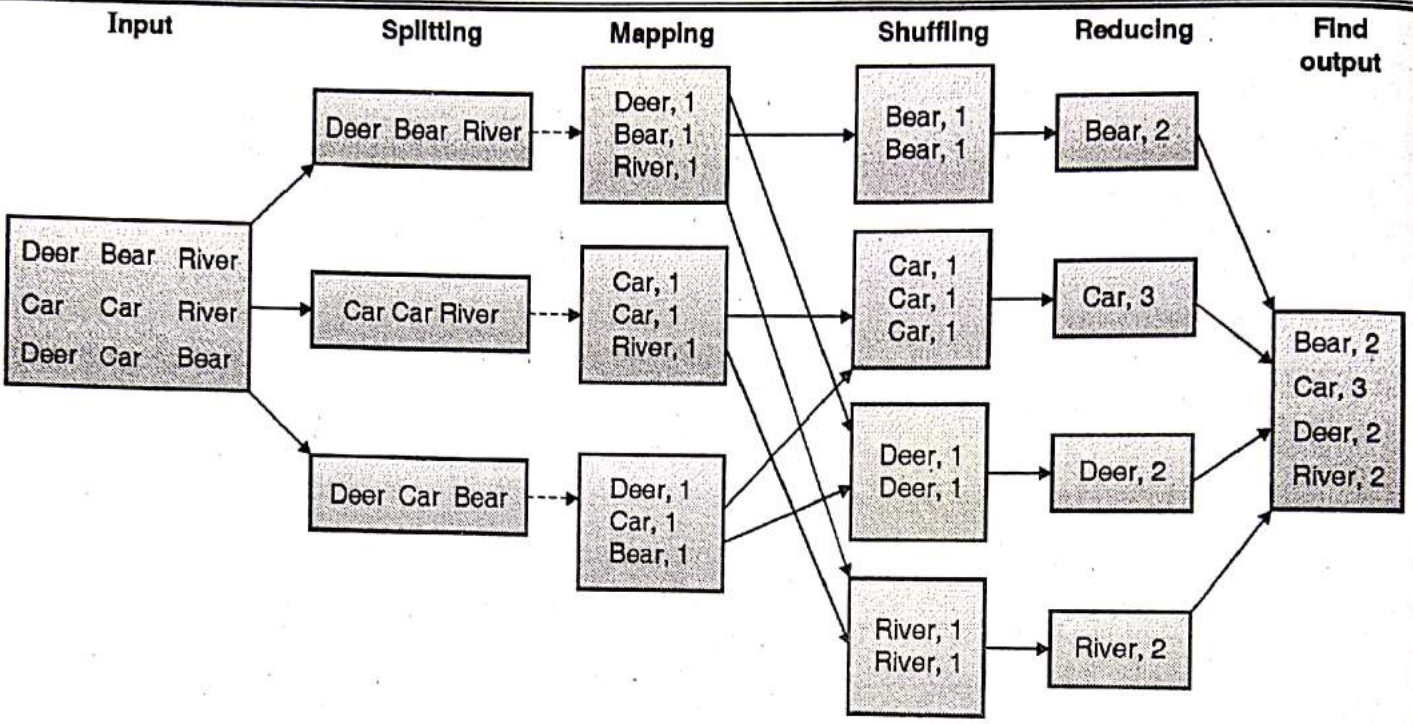


Fig. 3.2.1 : MapReduce process for word frequency count

3.2.4 Combiners

MU - May 17

Q. What are Combiners? When Should one use combiner in mapreduce job? **(May 17, 5 Marks)**

Q. What are combiners? Explain the position and significance of combiners.

- A combiner is a type of mediator between the mapper phase and the reducer phase. The use of combiners is totally optional. As a combiner sits between the mapper and the reducer, it accepts the output of map phase as an input and passes the key-value pairs to the reduce operation.
- Combiners are also known as semi-reducers as they reside before the reducer. A combiner is used when the Reduce function is commutative and associative. This means that the values can be combined in any order without affecting the final result. For example, addition is an operation which is both commutative as well as associative.
- In case of a huge dataset, when a MapReduce technique is applied, then the mapper phase generates an enormous amount of key-value pairs and these intermediate results need to be handed over to the reducer for the reduction process. But to transfer such a large number of key-value pairs from the mapper to the reducer, sufficiently large communication network bandwidth is required which generally leads to network congestion.
- So to avoid such congestion we can put some of the work of reducers in to the Map tasks. For example in a single Map task if c number of (w, 1) key-value pairs appear having the same key, they can be combined into a pair (w, c), where w is the key and c is the number of times the word w appears in the set of documents handled by this Map task.
- Fig. 3.2.2 shows position and working mechanism of combiner. Combiner reside between map phase and reduce phase which is responsible for making a group of values of same key which is received from mapper phase.

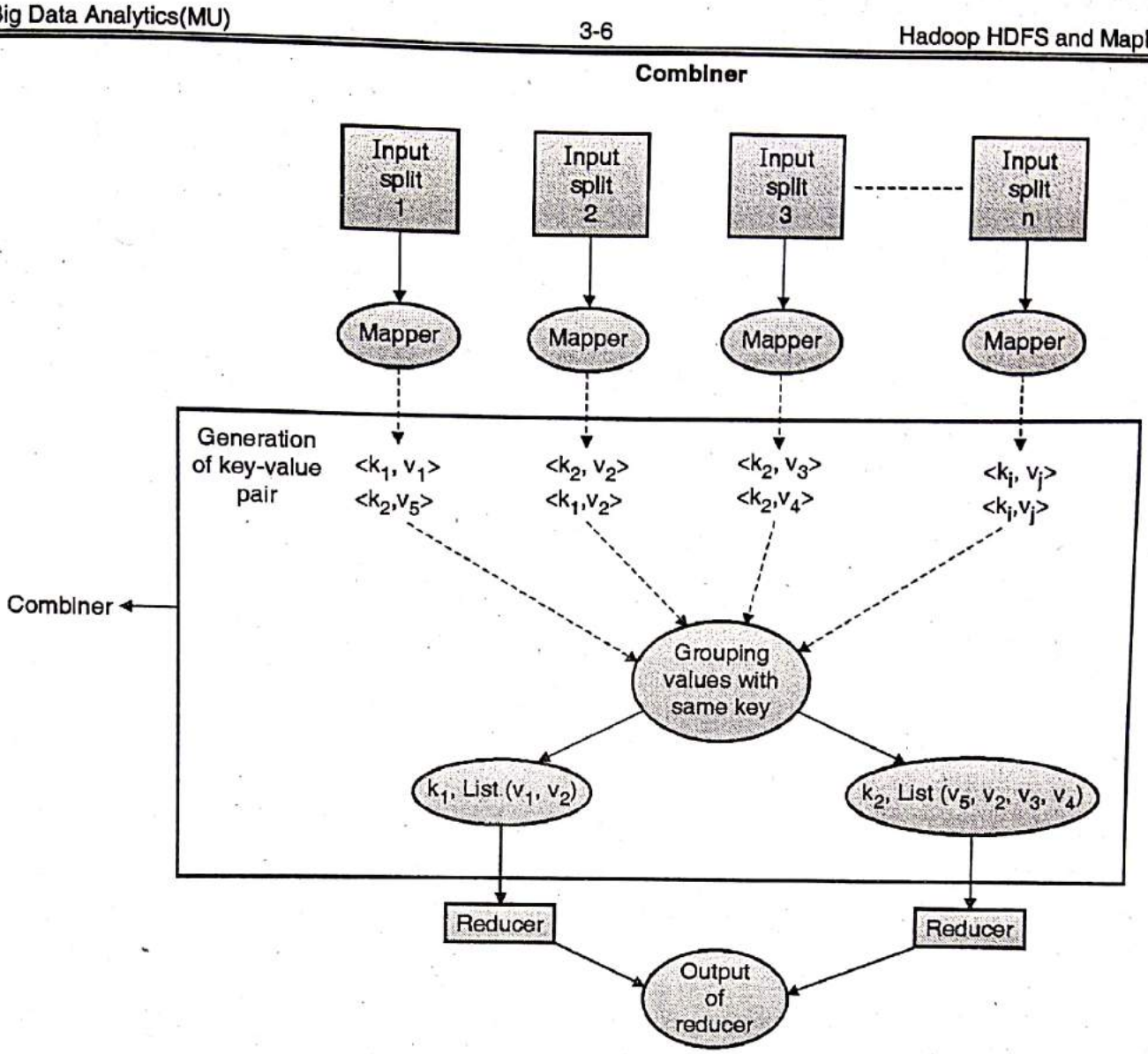


Fig. 3.2.2 : Position and Working Mechanism of Combiner

3.2.5 Details of MapReduce Execution

- In this section we will discuss in details how a MapReduce based program is executed. The user program first creates a Master controller process with the help of fork command using the library provided by the MapReduce system as depicted in Fig. 3.2.3.
- In addition to the Master process, the user program also forks a number of worker processes. These processes run on different compute nodes.
- The Master has the following responsibilities :
  - Creation of Map and Reduce tasks,
  - Assignment of Map and Reduce tasks to the Workers,
  - Keeping track of the Map and Reduce tasks (idle/executing/completed).
- A Worker process can be assigned either a Map task or a Reduce task but not both the tasks.

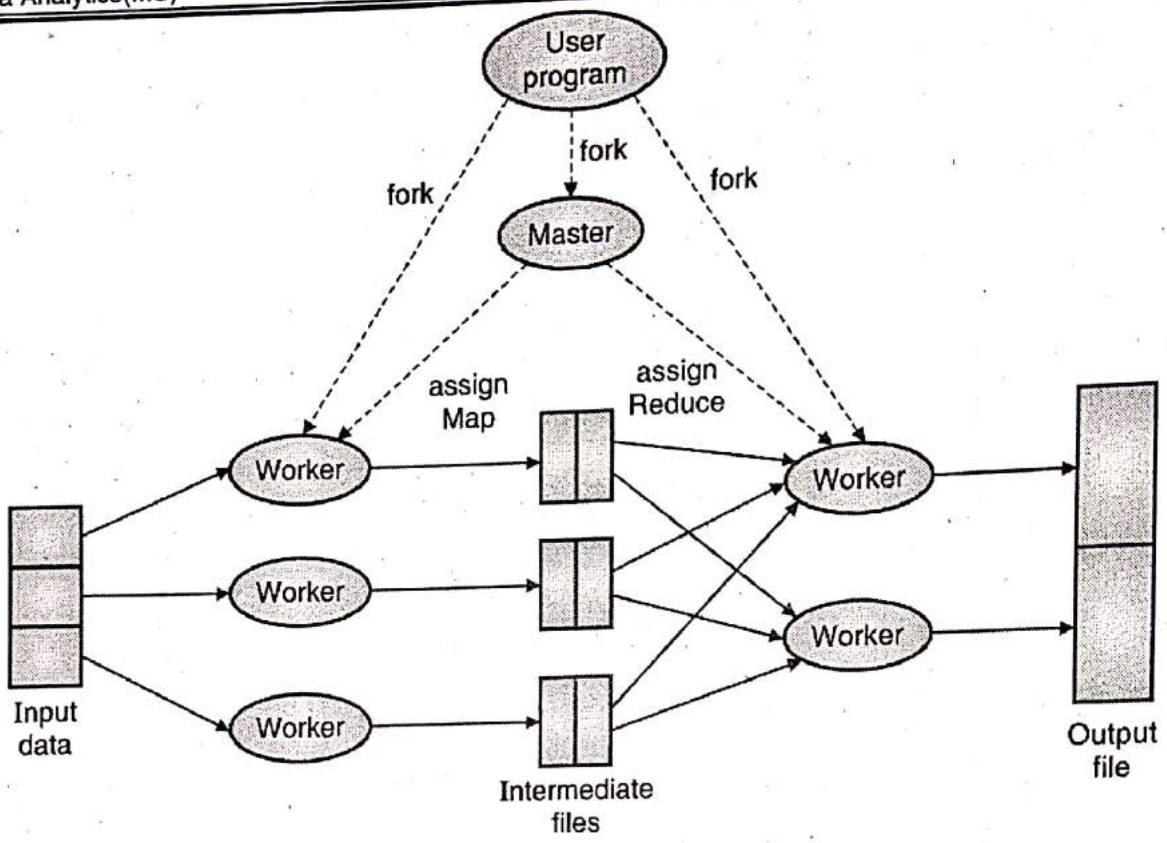


Fig. 3.2.3 : MapReduce program execution

- Every Map task creates one intermediate file in its local compute node for each of the Reduce tasks. So for example in the Fig. 3.2.3 there are two Reduce tasks, thus each of the Map tasks create two intermediate files which will be passed on as input to the Reduce tasks.
- All the Reduce tasks together will produce a single file as the final output of the MapReduce based program.

3.2.6 Coping with Node Failures

- There are three types of node failures :
  - Master node failure,
  - Map worker node failure,
  - Reduce worker node failure.
- If the Master node fails then the entire process has to be restarted. This is the worst kind of failure.
- If a Map worker node fails then the Master will assign the tasks to some other available worker node even if the task had completed.
- If a Reduce worker node fails then the tasks are simply rescheduled on some other Reduce worker later.

3.3 Algorithms using MapReduce

- MapReduce is not the solution for every problem. The distribute file system only makes sense for extremely large files which are very rarely updated.
- The main motivation behind the creation of Google's implementation of MapReduce was their PageRank algorithm which requires very large dimensional matrix-vector multiplications.
- In this section we will study a few algorithms which fit nicely into the MapReduce style of computing.

3.3.1 Matrix-Vector Multiplication by MapReduce

MU - May 16, May 19

Q. Explain matrix-vector multiplication algorithm by Map Reduce. (May 16, 10 Marks)

Q. Write pseudo code for Matrix vector Multiplication by MapReduce. Illustrate with an example showing all the steps. (May 19, 10 Marks)

- Let us consider a Matrix M of size n × n. Let m_{ij} denote the element in row i and column j. Let us also consider a vector v of length n and the j^{th} element of the vector is represented as v_j.
- The matrix-vector multiplication will produce another vector x whose ith element x_i is given by the formula:

$$x_i = \sum_j^n = 1\ m_{ij}v_j$$

- In real life applications such as Google's PageRank the dimensions of the matrices and vectors will be in trillions. Let us at first take the case where although the dimension n is large but it is also able to fit entirely in the main memory of the compute node.
- The Map task at each Map worker node works on a chunk of M and the entire v and produces the key-value pairs $(i, m_{ij}v_j)$. All the sum terms of the component x_i of the result vector of matrix-vector multiplication will be getting the same key i.
- The Reduce tasks sum all the values for a particular key and produce the result (i, x_i).
- In the case the where the vector v is too large to fit into the main memory of a compute node, an alternative approach as shown in Fig. 3.3.1 is taken. The matrix M is divided into vertical stripes and the vector v is divided into horizontal stripes having the following characteristics:
  - The number of stripes in M and the number of stripes in v must be equal,
  - The width of all the stripes in M must be equal,
  - The height of all the stripes in v must be equal.
  - The size of a stripe in v must be such that it can fit conveniently in main memory of a compute node.
- Now it is sufficient to multiply the jth stripe of M with the jth stripe of v. A chunk of a stripe of M and the corresponding entire stripe of v is assigned to each Map task and the calculations proceed as described earlier.

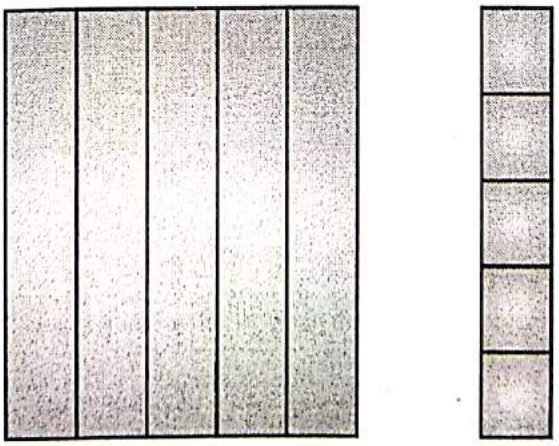


Fig. 3.3.1 : Division of matrix and vector into stripes

3.3.2 Relational-Algebra Operations

Q. What is relational Algebra? Explain different operation performed by Relational algebra.

- A relation is a table. A row of this table is called a tuple and the column headers are called attributes. The schema of a relation is the set of its attributes. A relation is represented by $R(A_1, A_2, \ldots, A_n)$, where R is the name of the relation and $A_1, A_2, \ldots, A_n$ are the attributes of R.
- Relational algebra defines the standard operations that can be performed on relations. In this section we are going to discuss the following relational algebra operations :
  - Selection,
  - Projection,
  - Union,
  - Intersection,
  - Difference.

1. Selection operation

- In case of a selection operation a constraint which is denoted by 'C' is applied on every tuple in the relation.
- Only those tuples which satisfy the specified constraint 'C' will be retrieved and shown by the system as output.
- Selection operation in the relational algebra is represented by $\sigma_C(R)$.

Where, $\sigma \rightarrow$ represents select operation

$C \rightarrow$ represents condition/constraint

$R \rightarrow$ represents the relation.

2. Projection operation

- Consider 'S' to be a subset of columns/attributes for a relation R.
- E.g. If a relation contains a total of10 columns then consider only first 5 columns for processing.
- From all the tuples/rows only the specified attributes/columns are retrieved and shown as output.
- The projection operation is represented in relational algebra as

$$\pi_S (R)$$

Where, $\pi \rightarrow$represents project operation

$S \rightarrow$ represents subset

$R \rightarrow$represents the relation

3. Union, Intersection and difference operations

- All the three operations operate on the rows of two different relations. The basic requirement is that both of these relations must be having the same schema.

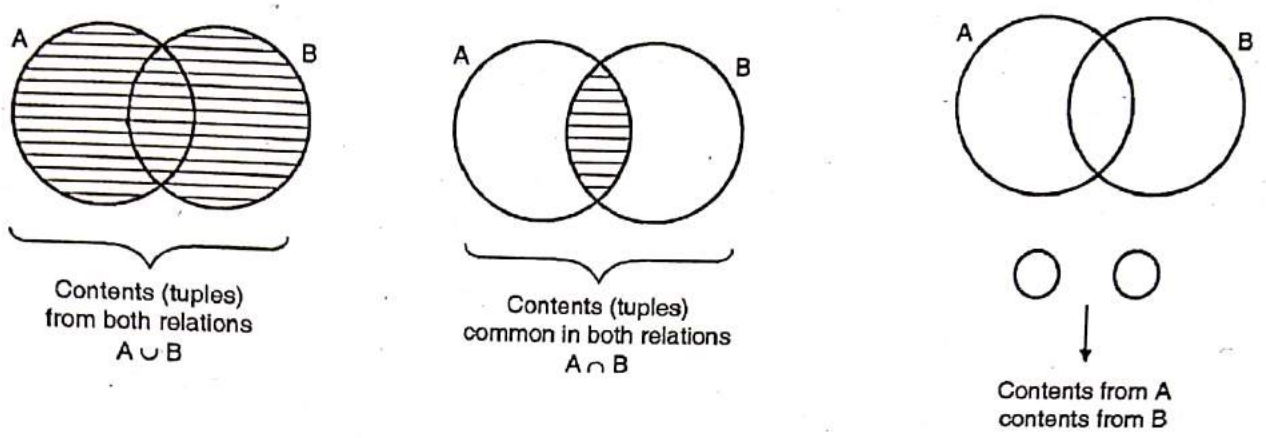


Fig. 3.3.2 : Union, Intersection and difference Venn diagrams

3.3.3 Computing Selections by MapReduce

- MapReduce is way too powerful for selection operations. Selections can be done completely either in the Map phase alone or in the Reduce phase alone. Here we shall discuss the former.

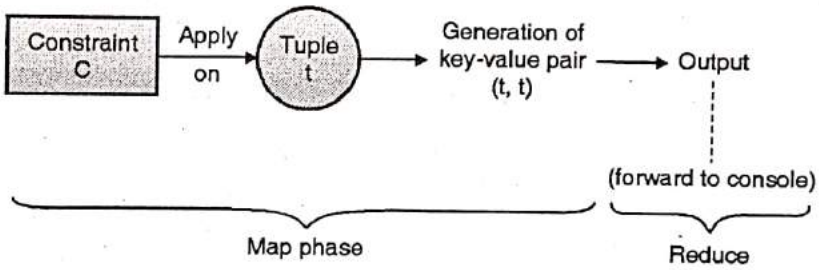


Fig. 3.3.3 : MapReduce implementation of selection operation

- In the Map task, the constraint C is applied on each tuple t of the relation.
- If C is satisfied by a tuple t, then the key-value pair (t, t) is produced for that particular tuple. Observe that here both the key as well as the value are the tuple t itself.
- If C is not satisfied by t, then the Map function will produce no output for that tuple t.
- As the processing is already finished in the Map function the Reduce function is the identity function. It will simply forward on the key-value pairs to the output for display.
- The output relation is obtained from either the key part or the value part as both are containing the tuple t.

3.3.4 Computing Projections by MapReduce

- In the Map task, from each tuple t in R the attributes not present in S are eliminated and a new tuple t' is constructed. The output of the Map tasks is the key-value pairs (t', t').
- The main job of the Reduce task is to eliminate the duplicate t's as the output of the projection operation cannot have duplicates.

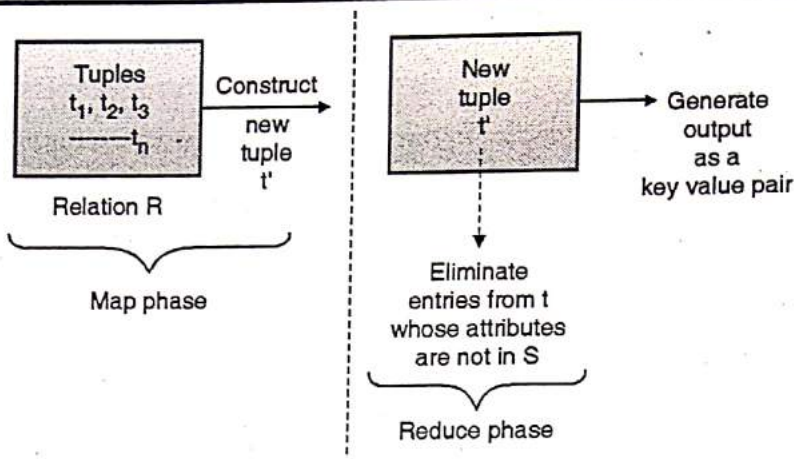


Fig. 3.3.4 : MapReduce implementation of projection operation

3.3.5 Union, Intersection and Difference by MapReduce

Q. Explain Union, Intersection and Difference operation with MapReduce techniques.

Union with MapReduce

- For the union operation R ∪ S, the two relations R and S must have the same schema. Those tuples which are present in either R or S or both must be present in the output.
- The only responsibility of the Map phase is of converting each tuple t into the key-value pair (t, t).
- The Reduce phase eliminates the duplicates just as in the case of projection operation. Here for a key t there can be either 1 value if it is present in only one of the relations or t can have 2 values if it is present in both the relations. In either case the output produced by the Reduce task will be (t, t).

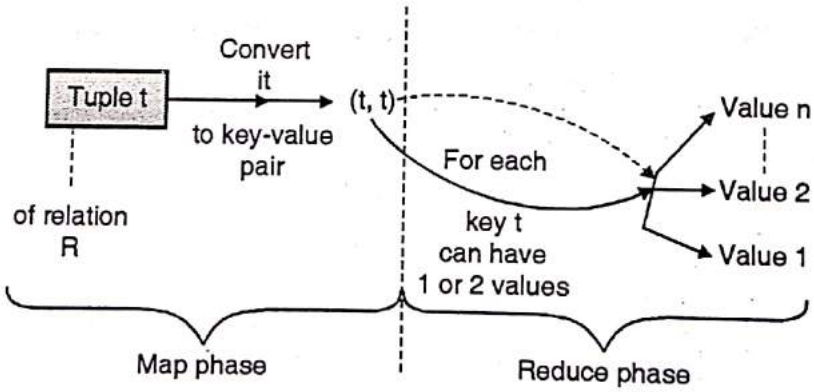


Fig. 3.3.5 : Union operation with MapReduce

Intersection with MapReduce

- For the intersection operation R ∩ S, both the relations R and S must have the same schema. Only those tuples which are present in both R and S should be present in the output.
- The responsibility of the Map phase is same as that of union operation i.e. conversion of a tuple 't' in a given relation 'R' into the key-value pair format (t, t).
- Reducer will produce the output (t, t) only if both R and S have the tuple t. This can be done by checking the number of values associated with the key t. If the key t has a list of two values [t, t], then the Reduce task will produce the output (t, t). If the key t has only one value [t], then the Reducer will produce nothing as output.

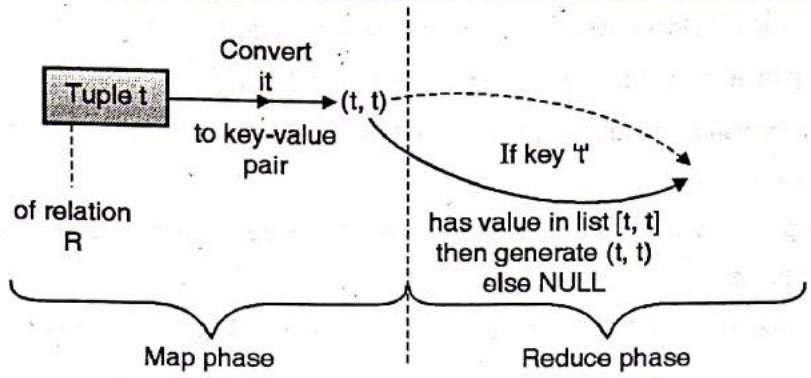


Fig. 3.3.6 : Intersection operation with MapReduce

Difference with MapReduce

- For the difference operation R – S, both the relations R and S must have the same schema. The tuples which are present only in R and not in S will be present in the output.
- The Map phase will produce the key-value pairs (t, R) for every tuple in R and (t, S) for every tuple in S.
- The Reduce phase will produce the output (t, t) only if the associated value of a key t is [R].

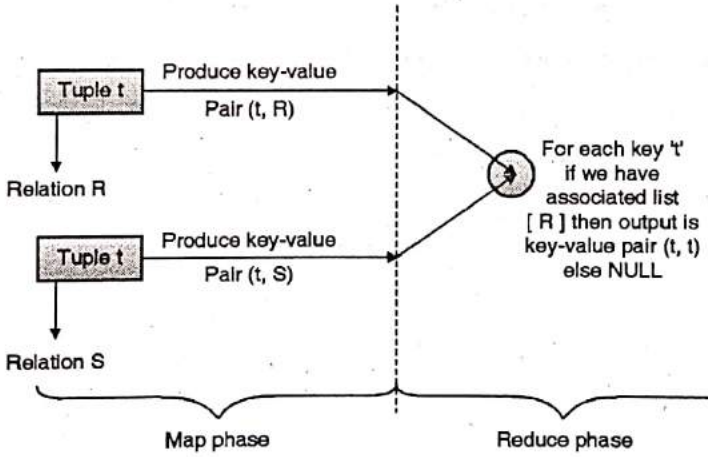


Fig. 3.3.7 : Difference operation with MapReduce

3.4 Hadoop Limitations

- Hadoop is a collection of open source projects created by Doug Cutting and Mike Cafarella in 2006. It was inspired by Google's MapReduce programming framework. Hadoop consist of the following core modules :
  - Hadoop Distributed File System (HDFS) which is the storage module and
  - MapReduce programming model which is the processing module.
- The various limitations of Hadoop are :
  - **Not fit for small files :** Hadoop was designed to work with big sized files and it cannot efficiently handle small files even if there are a huge number of small files. A file is considered small if its size is less than the HDFS block size which by default is 128 MB.
  - **Slow speed :** Hadoop works with massive amounts of distributed data which slows down the processing.

- **Not easy to use :** The developer needs to write the code for all the operations which makes it very difficult to use.
- **Security :** Hadoop does not support encryption which makes it vulnerable.
- **Real-time data processing not supported :** Hadoop is designed to support only batch processing and hence real-time processing feature is missing.
- **No iteration support :** Hadoop is not designed to support the feeding of the output of one stage to the input of the next stage of processing.
- **No caching :** Intermediate results are not cached and this brings down the performance.

Review Questions

Q. 1 Explain the large-scale file-system organization in detail.

Q. 2 Explain MapReduce framework with suitable example.

Q. 3 What are the different phases involved in MapReduce technique ? Explain with example.

Q. 4 What are combiners ? Explain the position and significance of combiners.

Q. 5 What is Relational Algebra ? Explain the different operations performed by Relational Algebra.

Q. 6 Explain selection and projection operation using MapReduce functionality.

Q. 7 Explain Union, Intersection and Difference operations with MapReduce techniques.

CHAPTER 4

Module - 3

NoSQL

Syllabus

Introduction to NoSQL, NoSQL Business Drivers, NoSQL Data Architecture Patterns: Key-value stores, Graph stores, Column family (Bigtable)stores, Document stores, Variations of NoSQL architectural patterns, NoSQL Case Study, 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.

4.1 NoSQL (What is NoSQL?)

MU - May 17

Q. What is NoSQL? **(May 17, 2 Marks)**

1. History

- The term NoSQL was first used by Carlo Strozzi in the year 1998.
- He mentioned this name for his Open Source Database system in which there was no provision of SQL Query interface.
- In the early 2009, at conference held in USA, NoSQL was comes into picture and actually comes in practice.

2. Overview

- NoSQL is a not a RDBMS (Relational Database Management System).
- NoSQL is specially designed for large amount of data stored in distributed environment.
- The important feature of NoSQL is, it is not bounded by table schema restrictions like RDBMS. It gives options to store some data even if there is no such column is present in table.
- NoSQL generally avoids join operations.

3. Need

- In real time, data requirements are changed a lot. Data is easily available with Facebook, Google+, Twitter and others.
- The data that includes user information, social graphs, geographic location data and other user-generated content.
- To make use of such abundant resources and data, it is necessary to work with a technology which can operate such data.
- SQL databases are not ideally designed to operate such data.
- NoSQL databases specially designed for operating huge amount of data.

4. Advantages

(i) Good resource scalability.

(ii) Lower operational cost.

(iii) Supports semi-structure data.

(iv) No static schema.

(v) Supports distributed computing.

(vi) Faster data processing.

(vii) No complicated relationships.

(viii) Relatively simple data models.

5. Disadvantages

(i) Not a defined standard.

(ii) Limited query capabilities.

6. Companies working with NoSQL

(i) Google (ii) Facebook

(iii) LinkedIn (iv) McGraw-Hill Education

4.2 NoSQL Basic Concepts

Theorem (Brewer's Theorem) for NoSQL

CAP theorem states three basic requirements of NoSQL databases to design a distributed architecture.

(a) Consistency

Database must remain consistent state like before, even after the execution of an operation.

(b) Availability

It indicates that NoSQL system is always available without any downtime.

(c) Partition Tolerance

This means that the system continues to function even the communication failure happens between servers i.e. if one server fails, other server will take over.

Note : It is very difficult to fulfill all the above requirements.

There are many combinations of NoSQL rules :

1. CA

- It is a single site cluster.
- All nodes are always in contact.
- Partitioning system can block the system.

2. **CP**

   Some data may not be accessible always still it may be consistent or accurate.

3. **AP**

   - System is available under partitioning.
   - Some part of the data may be inconsistent.

4. **BASE model**

   - Relational databases have some rules to decide behaviour of database transactions.
   - ACID model maintains the atomicity, consistency, isolation and durability of database transactions.
   - NoSQL turns the ACID model to the BASE model.

5. **BASE offers some guidelines**

   - Basic availability
   - Soft state
   - Eventual consistency

6. **Data storage**

   - NoSQL databases use the concept of a key / value store.
   - There are no schema restrictions for NoSQL database.
   - It simply stores values for each key and distributes them across the database, it offers efficient retrieval.

7. **Redundancy and Scalability**

   - To add redundancy to a database, we can add duplicate nodes and configure replication.
   - Scalability is simply a matter of adding additional nodes. There can be hash function designed to allocate data to server.

4.3 Case Study NoSQL (SQL vs NoSQL)

SQL databases are Relational Databases (RDBMS); whereas NoSQL database are non-relational database.

Data storage

- SQL databases stores data in a table whereas NoSQL databases stores data as document based, key-value pairs, graph databases or wide-column stores.
- SQL data is stored in form of tables with some rows.
- NoSQL data is stored as collection of key-value pair or documents or graph based data with no standard schema definitions.

Database schema

SQL databases have predefined schema which cannot be change very frequently, whereas NoSQL databases have dynamic schema which can be change any time for unstructured data.

Complex queries

- SQL databases provides standard platform for running complex query.
- NoSQL does not provide any standard environment for running complex queries.
- NoSQL are not as powerful as SQL query language.

Comparison between SQL and NoSQL

Sr. No.	SQL	NoSQL
1.	Full form is Structured Query Language.	Full form is Not Only SQL or Non-relational database.
2.	SQL is a declarative query language.	This is Not a declarative query language.
3.	SQL databases works on ACID properties, Atomicity Consistency Isolation Durability	NoSQL database follows the Brewers CAP theorem, Consistency Availability Partition Tolerance
4.	Structured and organized data	Unstructured and unreplicable data
5.	Relational Database is table based.	Key-Value pair storage, Column Store, Document Store, Graph databases.
6.	Data and its relationships are stored in separate tables.	No pre-defined schema.
7.	Tight consistency.	Eventual consistency rather than ACID property.
8.	**Examples :** MySQL Oracle MS SQL PostgreSQL SQLite DB2	**Examples :** MongoDB Big Table Neo4j Couch DB Cassandra HBase

4.4 Business Drivers of NoSQL

MU - May 17

Q. What are the business drivers for NoSQL? **(May 17, 4 Marks)**

1. The growth of big data

- Big Data is one of the main driving factor of NoSQL for business.
- The huge array of data collection acts as driving force for data growth.

2. Continuous availability of data

- The competition age demands less downtime for better company reputation.
- Hardware failures are possible but NoSQL database environments are built with a distributed architecture so there are no single points of failure.
- If one or more database servers goes down, the other nodes in the system are able to continue with operations without any loss of data.
- So, NoSQL database environments are able to provide continuous availability.

3. **Location independence**
   - It is ability to read and write to a database regardless of where that I/O operation is done.
   - The master/slave architectures and database sharding can sometimes meet the need for location independent read operations.
4. **Modern transactional capabilities**

   The transactions concept is changing and ACID transactions are no longer a requirement in database systems.
5. **Flexible data models**
   - NoSQL has more flexible data model as compared to others.
   - A NoSQL data model is schema-less data model not like RDBMS.
6. **Better architecture**
   - The NoSQL has more business oriented architecture for a particular application.
   - So, Organizations adopt a NoSQL platform that allows them to keep their very high volume data.
7. **Analytics and business intelligence**
   - A key driver of implementing a NoSQL database environment is the ability to mining data to derive insights that offers a competitive advantage.
   - Extracting meaningful business information from very high volumes of data is a very difficult task for relational database systems.
   - Modern NoSQL database systems deliver integrated data analytics and better understanding of complex data sets which facilitate flexible decision-making.

4.5 NoSQL Database Types

MU - May 16, Dec. 16, May 17, May 18, May 19

Q. What are the different data architecture patterns in NOSQL? Explain Graph Store and Column Family Store patterns with relevant examples. **(May 16, May 19, 10 Marks)**

Q. Explain different NoSQL data architecture patterns. **(Dec. 16, 10 Marks)**

Q. Discuss any two architectural patterns of NoSQL. **(May 17, 4 Marks)**

Q. Explain in detail any one NOSQL architecture pattern. Identify two applications that can use this pattern. **(May 18, 5 Marks)**

Different Architectural Patterns in NoSQL

- Key-Value databases examples : Riak,Redis, Memcached, BerkeleyDB, upscaledb, Amazon DynamoDB.
- Document databases examples : MongoDB, CouchDB, Terrastore, OrientDB , RavenDB
- Column family stores examples : Cassendra, HBase, HyperTable.
- Graph Databases examples : Neo4j, InfiniteGraph, FlockDB.

1. Key-value store databases

- This is very simple NoSQL database.
- It is specially designed for storing data as a schema free data.
- Such data is stored in a form of data along with indexed key.

Examples

- Cassandra
- Azure Table Storage (ATS)
- DyanmoDB

Fig. 4.5.1

Use Cases

This type is generally used when you need quick performance for basic Create-Read-Update-Delete operations and data is not connected.

Example

- Storing and retrieving session information for a Web pages.
- Storing user profiles and preferences
- Storing shopping cart data for ecommerce

Limitations

- It may not work well for complex queries attempting to connect multiple relations of data.
- If data contains lot of many-to-many relationships, a Key-Value store is likely to show poor performance.

2. Column store database

- Instead of storing data in relational tuples (table rows), it is stored in cells grouped in columns.
- It offers very high performance and a highly scalable architecture.

Fig. 4.5.2

Examples

(i) HBase (ii) Big Table (iii) Hyper Table

Use Cases

- Some common examples of Column-Family database include event logging and blogs like document databases, but the data would be stored in a different fashion.
- In logging, every application can write its own set of columns and have each row key formatted in such a way to promote easy lookup based on application and timestamp.
- Counters can be a unique use case. It is possible to design application that needs an easy way to count or increment as events occurs.

3. Document database

- Document databases works on concept of key-value stores where "documents" contains a lot of complex data.
- Every document contains a unique key, used to retrieve the document.
- Key is used for storing, retrieving and managing document-oriented information also known as semi-structured data.

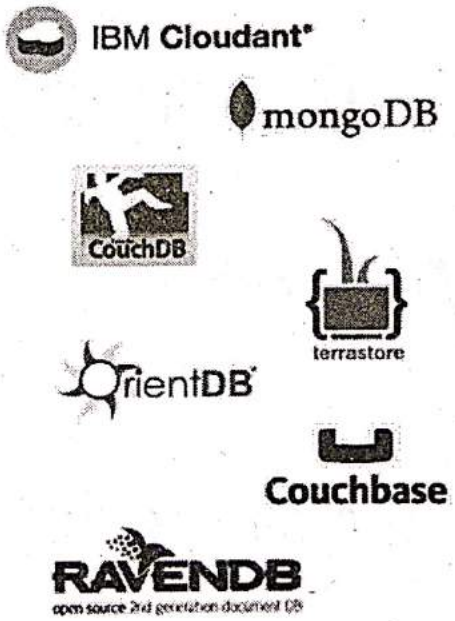


Fig. 4.5.3

Examples

(i) MongoDB (ii) Couch DB

Use Cases

- The example of such system would be event logging system for an application or online blogging.
- In online blogging user acts like a document; each post a document; and each comment, like, or action would be a document.

- All documents would contain information about the type of data, username, post content, or timestamp of document creation.

Limitations

- It's challenging for document store to handle a transaction that on multiple documents.
- Document databases may not be good if data is required in aggregation.

4. Graph database

Data is stored as a graph and their relationships are stored as a link between them whereas entity acts like a node.

Examples

(i) Neo4j (ii) Polyglot

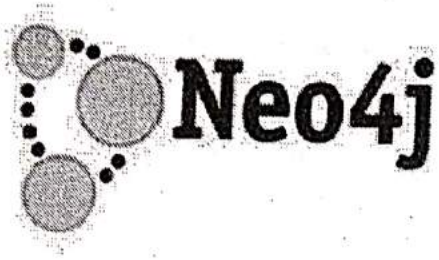


Fig. 4.5.4

Use Cases

- The very important and popular application would be social networking sites can benefit by quickly locating friends, friends of friends, likes, and so on.
- The Google Maps can help you to use graphs to easily model their data for finding close locations or building shortest routes for directions.
- Many recommendation systems makes effective use of this model.

Limitations

- Graph Databases may not be offering better choice over other NoSQL variations.
- If application needs to scale horizontally this may introduces poor performance.
- Not very efficient when it needs to update all nodes with a given parameter.

5. Comparison of NoSQL variations

Database model	Performance	Scalability	Flexibility
Key value store database	High	High	High
Column store database	High	High	Moderate
Document store database	High	Variable (High)	High
Graph database	Variable	Variable	High

4.6 Benefits of NoSQL

1. Big data analytics

- Big data is one of main feature promotes growth and popularity of NoSQL.
- NoSQL has good provision to handle such big data.

2. Better data availability

- NoSQL database works with distributed environments.
- NoSQL database environments should provide good availability across multiple data servers.
- NoSQL databases supply high performance.

3. Location independence

NoSQL data base can read and write database regardless of location of database operation.

4.7 Introduction to Big Data Management

- We all are surrounded by huge data. People upload/download videos, audios, images from variety of devices. Sending text messages, multimedia messages, updating their Facebook, WhatsApp, Twitter status, comments, online shopping, online advertising etc. generates huge data.
- As a result machines have to generate and keep huge data too. Due to this exponential growth of data the analysis of that data becomes challenging and difficult.
- The term 'Big Data' means huge volume, high velocity and a variety of data. This big data is increasing tremendously day by day.
- Traditional data management systems and existing tools are facing difficulties to process such a Big Data.
- R is one of the main computing tools used in statistical education and research. It is also widely used for data analysis and numerical computing in other fields of scientific research.

4.8 Big Data

- We all are surrounded by huge data. People upload/download videos, audios, images from variety of devices.
- Sending text messages, multimedia messages, updating their Facebook, WhatsApp, Twitter status, comments, online shopping, online advertising etc.
- Generates huge data. As a result machines have to generate and keep huge data too. Due to this exponential growth of data the analysis of that data becomes challenging and difficult.
- The term 'Big Data' means huge volume, high velocity and a variety of data. This big data is increasing tremendously day by day. Traditional data management systems and existing tools are facing difficulties to process such a Big Data.
- Big data is the most important technologies in modern world. It is really critical to store and manage it. Big is a collection of large datasets that cannot be processed using traditional computing techniques.
- Big Data includes huge volume, high velocity and extensible variety of data. The data in it may be structured data Semi Structured data or unstructured data. Big data also involves various tools, techniques and frameworks.

Four Important V of Big Data

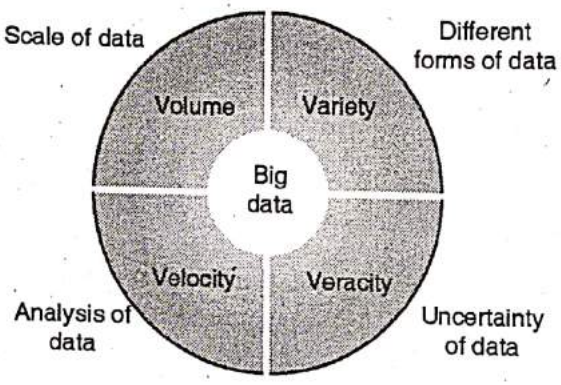


Fig. 4.8.1 : Four V of big data

1. **Volume**

   Huge amount of data is generated during big data applications.

2. **Velocity**

   For time critical applications the faster processing is very important. E.g. share marketing, video streaming

3. **Variety**

   The data may be Structured, Semi Structured or Unstructured.

4. **Veracity**

   Data is not certain. Data captured can vary greatly. So accuracy of analysis depends on the veracity of the source data.

4.8.1 Tools Used for Big Data

1. **Map Reduce**

   Hadoop, Hive, Pig, Cascading, Cascalog, mrjob, Caffeine, S4, MapR, Acunu, Flume, Kafka, Azkaban, Oozie, Greenplum

2. **Storage**

   S3, Hadoop Distributed File System

3. **Servers**

   EC2, Google App Engine, Elastic, Beanstalk, Heroku

4. **NoSQL**

   Zookeeper, MongoDB, Cassandra, Redis, Big Table, Hbase, Hyper table, Voldemort, Riak, Couch DB

5. **Processing**

   R, Yahoo! Pipes, Mechanical Turk, Solr/Lucene, ElasticSearch, Datameer, BigSheets, Tinkerpop

4.8.2 Understanding Types of Big Data Problems

1. **Acquiring data**

   High volume of data and transactions are the basic requirements of big data. Infrastructure should support the same. Flexible data structures should be used for the same. The amount of time required for this should be as less as possible.

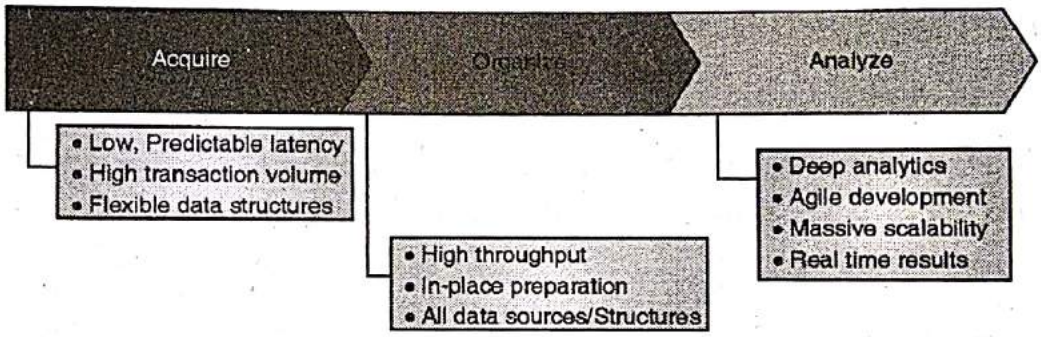


Fig. 4.8.2

2. **Organizing data**

   As the data may be structured, semi structured or unstructured it should be organized in a fast and efficient way.

3. **Analysing data**

   Data analysis should be faster and efficient. It should support distributed computing.

4.9 Four Ways of NoSQL to Operate Big Data Problems

1. **Key-value store databases**

   - This is very simple NoSQL database.
   - It is specially designed for storing data as a schema free data.
   - Such data is stored in a form of data along with indexed key.

Key : 1	ID : 123	First Name : Ganesh	
Key : 2	Email : abc@gmail.com	Location : Mumbai	Age : 37
Key : 3	Facebook ID : wea	Password : xxx	Name : Max

Fig. 4.9.1 : Example of unstructured data for user records

Working

- The schema-less format of a key value database is required for data storage needs.
- The key can be auto-generated while the value can be String.
- The key value uses a hash table in which there exists a unique key and a pointer to a particular item of data.
- A logical group of keys called as bucket, There can be identical keys in different buckets.
- It will improve performance because of the cache mechanisms that accompany the mappings.
- To read any value you need to know both the key and the bucket because the real key is a hash (Bucket+ Key).

Read Write values

- **Get(key) :** It will returns the value associated with key.
- **Put(key, value) :** It will associates the value with the key.
- **Multi-get(key1, key2, .., keyN) :** It will returns the list of values associated with the list of keys.
- **Delete(key) :** It will delete entry for the key from the data store.

2. Column store database

- Instead of storing data in relational tuples (table rows), it is stored in cells grouped in columns.
- It offers very high performance and a highly scalable architecture.

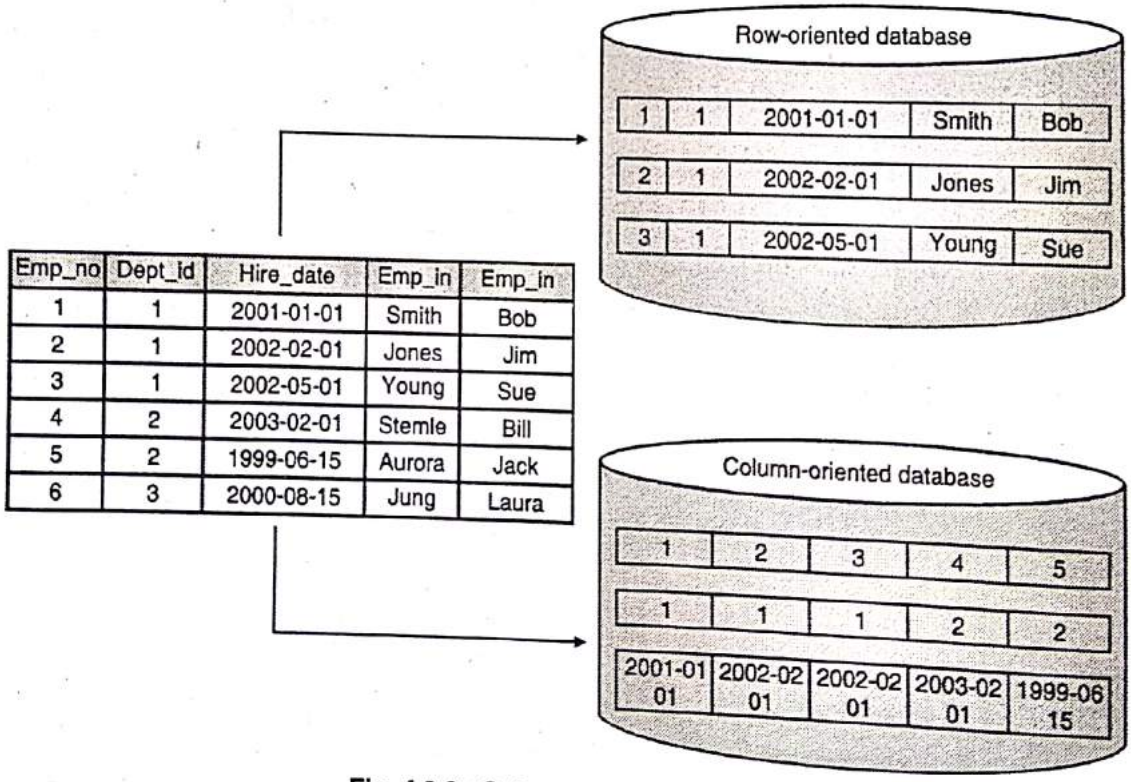


Fig. 4.9.2 : Column store database

Working

- In column-oriented NoSQL database, data is stored in cells grouped in columns of data rather than as rows of data.
- Columns are logically grouped into column families which are a virtually unlimited number of columns that can be

created at runtime.

- Read and write is done using columns.
- It offers fast search/ access and data aggregation.

Data Model

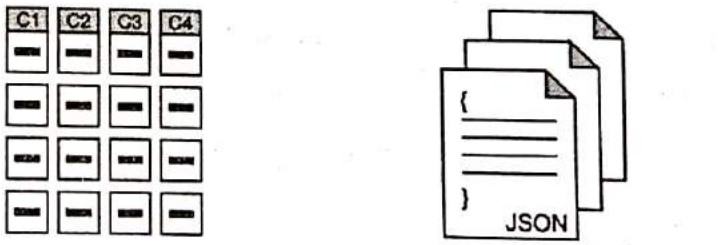
- **ColumnFamily :** Single structure that can group Columns and SuperColumns
- **Key :** The permanent name of the record having different numbers of columns.
- **Keyspace :** This defines name of the application.
- **Column :** It has an ordered list of elements with a name and a value defined.

Examples

(i) HBase (ii) Big Table (iii) Hyper Table

3. Document database

- Document databases works on concept of key-value stores where "documents" contains a lot of complex data.
- Every document contains a unique key, used to retrieve the document.
- Key is used for storing, retrieving and managing document-oriented information also known as semi-structured data.



Relational data model

Highly-structured table organization with rigidly-defined data formats and record structure.

Document data model

Connection of complex documents with arbitrary, nested data formats and varying "record" format.

Fig. 4.9.3 : Document database

Working

- This type of data is a collection of key value pairs is compressed as a document store quite similar to a key-value store, but the only difference is that the values stored is known as "documents" has some defined structure and encoding.
- The above example shows data values collected in Colum family.
- JSON and XML are common encoding as above.
- It is schemaless data makes easy for JSON to handle data.

Examples

(i) Mongo DB

(ii) Couch DB

4. Graph database

Data is stored as a graph and their relationships are stored as a link between them whereas entity acts like a node.

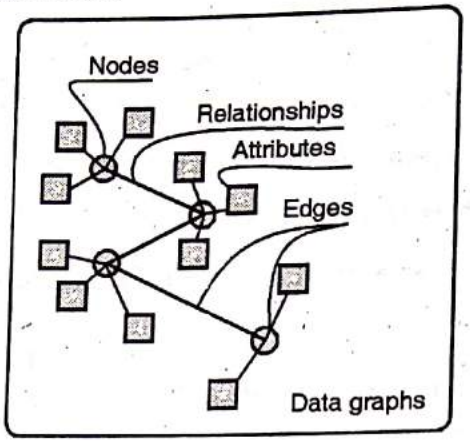


Fig. 4.9.4 : Graph database

Working

- In a Graph NoSQL Database, a flexible graphical representation is used with edges, nodes and properties which provide index-free adjacency.
- Data can be easily transformed from one model to the other using a Graph Base NoSQL database.
- These databases use edges and nodes to represent and store data.
- These nodes are organised by some relationships, which is represented by edges between the nodes.
- Both the nodes and the relationships have properties.

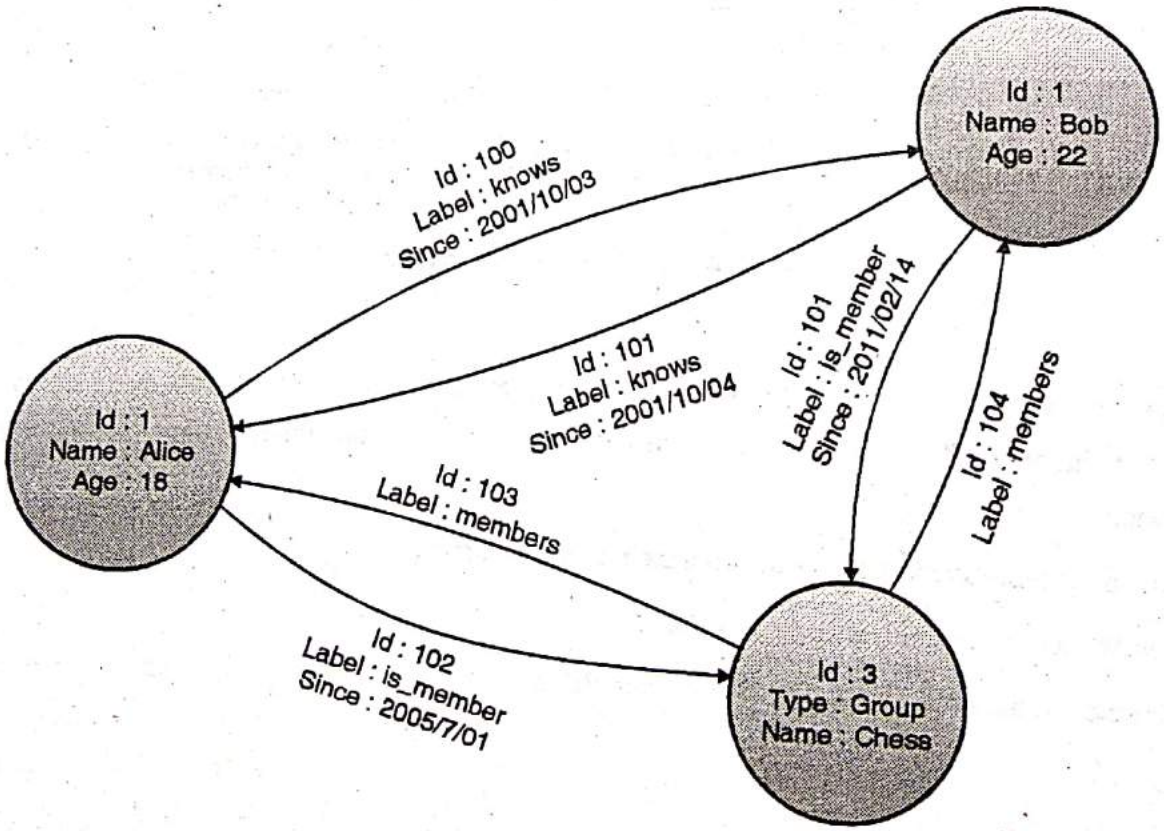


Fig. 4.9.5

Examples

(i) Neo4j (ii) Polyglot

4.10 Analyzing Big Data with a Shared-Nothing Architecture

Parallelism in databases represents one of the most successful instances of parallel computing system.

Types :

1. Shared Memory System (UNIX FS)
2. Shared Disk System (ORACLE RAC)
3. Shared Nothing System (HDFS)
4. Hierarchical System

4.10.1 Shared Memory System

(a) Architecture details

- Multiple CPUs are attached to a common global shared memory via interconnection network or communication bus.
- Shared memory architectures usually have large memory caches at each processor, so that referencing of the shared memory is avoided whenever possible.
- Moreover, caches need to be coherent. That means if a processor performs a write to a memory location, the data in that memory location should be either updated at or removed cached data.

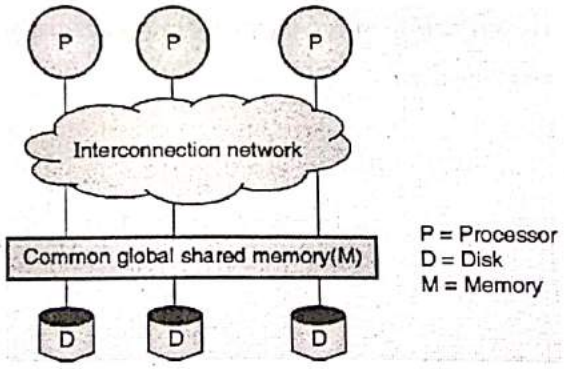


Fig. 4.10.1 : Shared memory system architecture

(b) Advantages

- Efficient communication between processors.
- Data can be accessed by any processor without being moved from one place to other.
- A processor can send messages to other processors much faster using memory writes.

(c) Disadvantages

- Bandwidth problem
- Not scalable beyond 32 or 64 processors, since the bus or interconnection network will get into a bottleneck.
- More number of processors can increase waiting time of processors.

4.10.2 Shared Disk System

(a) Architecture details

- Multiple processors can access all disk directly via inter communication network. But, every processor has local memory.
- Shared disk has two advantages over shared memory.
- Each processor has its own memory; the memory bus is not a bottleneck.
- System offers a simple way to provide a degree of **fault tolerance**.
- The systems built around this architecture are called **clusters**.

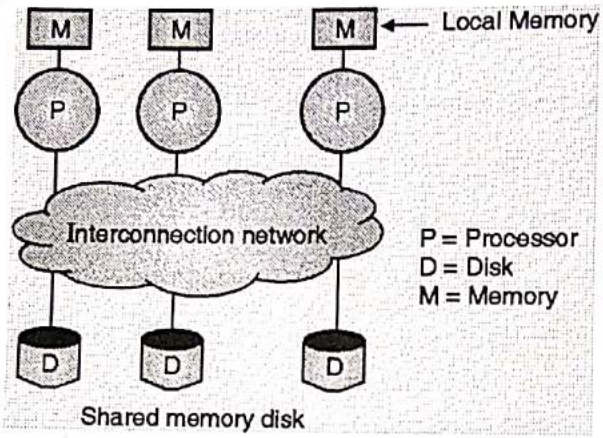


Fig. 4.10.2 : Shared memory disk architecture

(b) Advantages

- Each CPU or processor has its own local memory, so the memory bus will not face bottleneck.
- High degree of fault tolerance is achieved.
- **Fault tolerance** : If a processor (or its memory) fails, the other processor can take over its tasks, since the database is present on disks and are accessible to all processors.
- If one processor fails, other processors can take over its tasks, since database is on shared disk that can be accessible from all processors.

(c) Disadvantages

- Some Memory load is added to each processor.
- **Limited Scalability** : Not scalable beyond certain point. The shared-disk architecture faces this problem because large amounts of data are shipped through the interconnection network. So now the interconnection to the disk subsystem is a bottleneck.
- The basic problem with the shared-memory and shared-disk architecture is interference. As more CPUs are added, existing CPUs are slowed down because of the increased contention for memory accesses and network bandwidth.

(d) Applications

Digital Equipment Corporation : DEC cluster running Relational Databases were one of the early commercial user of shared disk database architecture. Now this is owned by Oracle.

Note : This observation has motivated the development of the shared nothing architecture, which is now widely considered to be the best architecture for large parallel database systems.

4.10.3 Shared Nothing Disk System

(a) Architecture details

- Each processor has its own local memory and local disk.
- A processor at one node may communicate with another processor using high speed communication network.
- Any terminal can act as a Node which functions as Server for data that is stored on local disk.
- Moreover, the interconnection networks for shared nothing systems are usually designed to be scalable, so that we can increase transmission capacity as more nodes are added to the network.

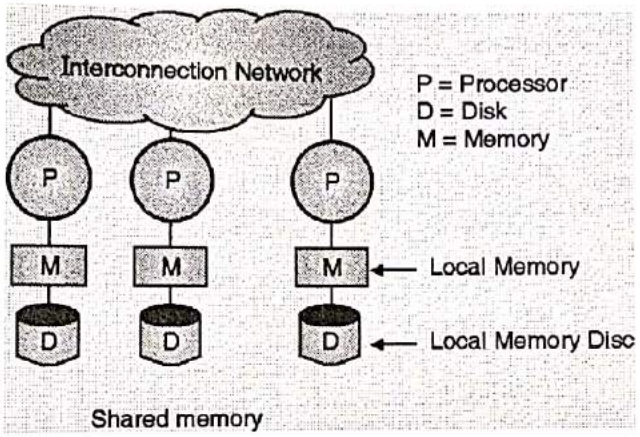


Fig. 4.10.3 : Shared nothing architecture

(b) Advantages

- In this type of architecture no need to go through all I/O, Only a single interconnection network queries which access non local disk can pass through the network.
- We can achieve High degree of parallelism. i.e. Number of CPU and disk can be connected as desired.
- Shared nothing architecture systems are more scalable and can easily support a large number of processors.

(c) Disadvantages

- Cost of communication and of non local disk access is higher than other two architectures since sending data involves software interaction at both ends.
- Requires rigid data partitioning.

(d) Applications

- The teradata database machine uses shared nothing database architecture.
- Grace and the Gamma research prototypes.

4.10.4 Hierarchical System

Architecture details

- The hierarchical architecture comes with combined characteristics of shared memory, shared disk and shred nothing architectures.
- At the top level, the system consists of nodes connected by an interconnection network and they do not share disks or memory with one another.
- This architecture is attempts to reduce the complexity of programming such systems yields to distributed virtual-memory architectures, where logically there is a single shared memory, the memory mapping hardware coupled with system software, allows each processor to view the disjoint memories as a single virtual memory.
- The hierarchical architecture is also referred to as **nonuniform memory architecture (NUMA).**

Hierarchical architecture

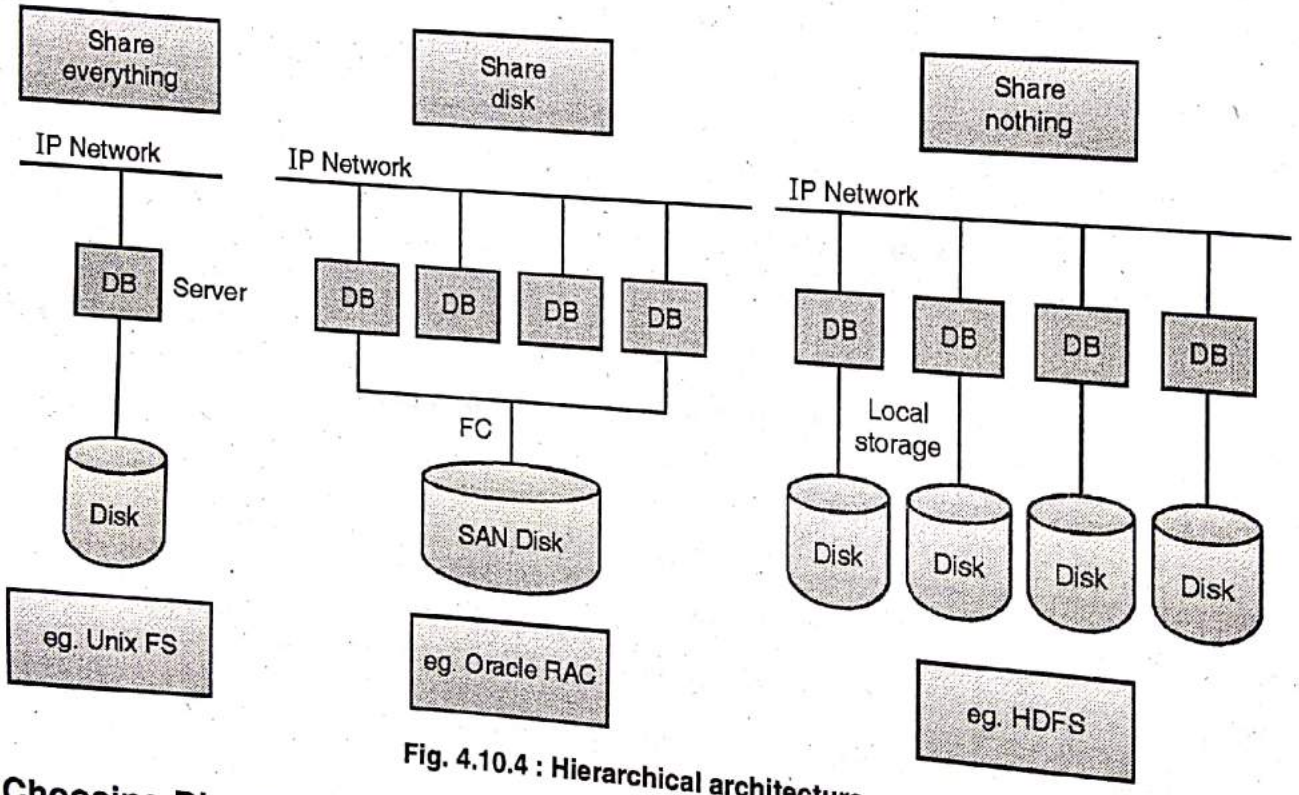


Fig. 4.10.4 : Hierarchical architecture

4.11 Choosing Distribution Models : Master-Slave versus Peer-to-Peer

Master Slave vs. Peer to Peer Model

- The data bases like HBase follow the master slave model In Master slave model one node out of all participating nodes will acts as the master i.e. the master node will responsible for governing all the decisions related to the job assignment, task assignment, data storage, data retrieval, data manipulation etc. The Master will delegate the jobs and tasks to the other working nodes in a given cluster. The main advantage is the working nodes cannot manipulate the things without permission of the master node.
- Fig. 4.11.1 shows Master slave model.

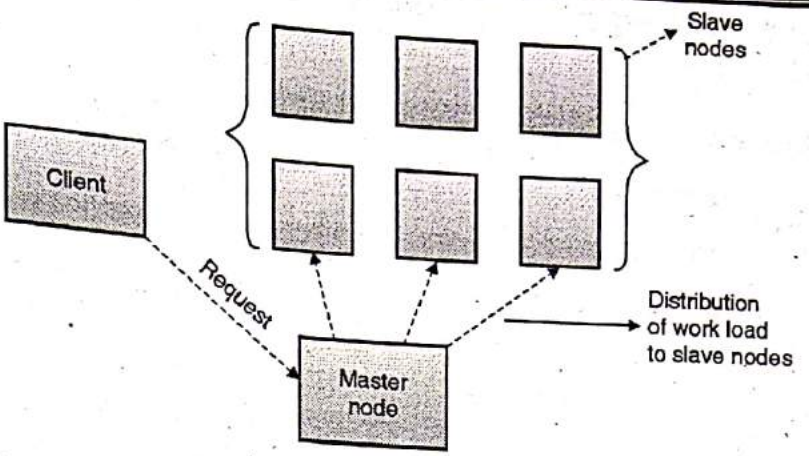


Fig. 4.11.1 : Master slave model

- The data bases like cassendra will follow the Peer- to-peer model. In peer to peer model all participating nodes has same rights. It means every node can perform all operations requested by the database user.
- These operations include Create, Read, Update, Delete as well as operations related to configuration of such databases.
- As all nodes has the same priority, so the requests from data base users will be received by any of the nodes irrespective of work load distribution.
- The peer to peer model will exhibit a data replication mechanism where data packets will be replicated for certain number of times to avoid the data loss if some part of the data base will get crashed.
- Fig. 4.11.2 shows Peer to Peer model.

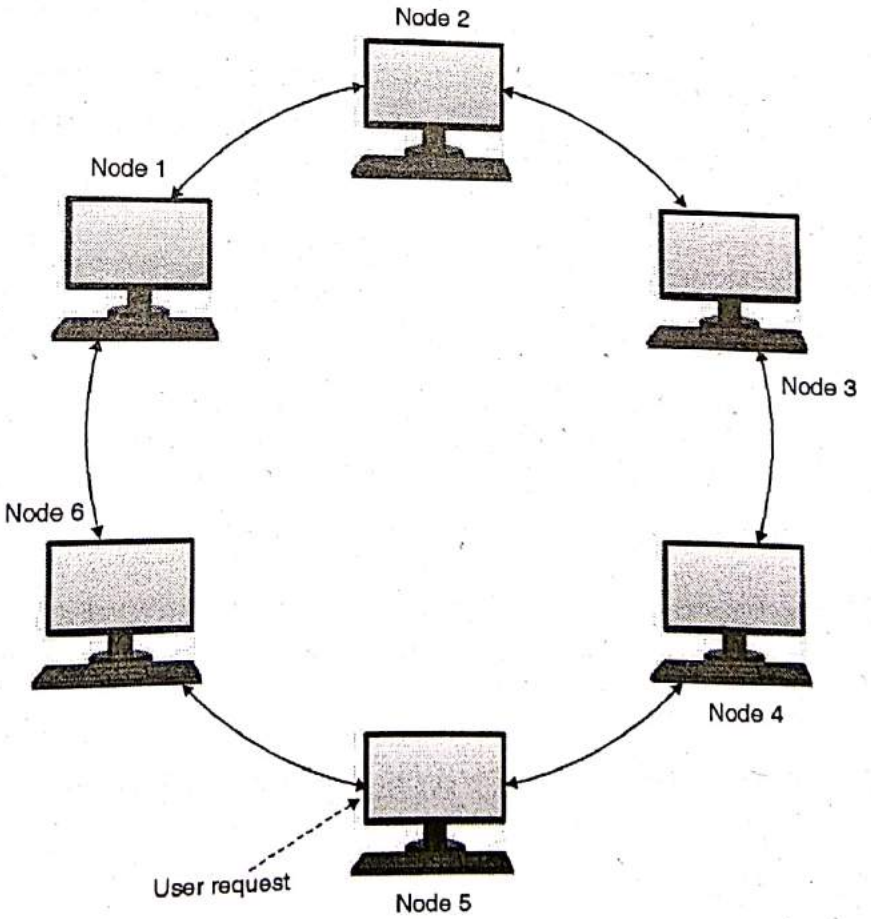


Fig. 4.11.2 : Peer to peer model

4.11.1 Big Data NoSQL Solutions

There are many NoSQL solutions to Bigdata problems as listed below,

1. Cassendra
2. DynamoDB
3. Neo4J
4. MongoDB
5. Hbase
6. Big Tables

4.11.1(A) Cassendra

1. Introduction

- Cassendra is a distributed storage system mainly designed for managing large amount of structured across multiple servers.
- Cassandra run with hundreds of servers and manages them.
- Cassandra provides a simple data model that supports dynamic control over data.
- Cassandra system was designed to run with economic hardware to handle large amount of data.
- For Example, Facebook runs on thousands of servers located in many data centres using Cassandra.
- The cassendra fall in the category of NOSQL data bases. In NOSQL the data bases doesn't have the schema i.e. a schema less data bases.
- The Main aim behind the development of cassesndra is to deal with the complexities involved in the Big Data which makes use of no. of nodes as well as it tries to avoid the critical part known as Single Point Failure.
- Cassendra has Data Replication as its major features, in which the data packet can duplicated no. of times. This duplication is generally limited to a pre defined threshold value. For Replication a Gossip Protocol is used.
- Nodes or work stations participating in the database will have equal priority, access privileges yet they all are connected to each other with dedicated communication links.
- When user of a data base requests to have read / write operation then any available node can receive this request and process it.

Basic building blocks of cassesndra

- **A node :** The Node represents the place where the data is actually stored.
- **The data center :** The Data center is the set of nodes which process same category of data.
- **The cluster :** A group of data centers is known as Cluster.
- **The commit log :** The Commit log is a maintenance tool which recovers the data if some failure occurs.

2. Types of Databases

- Besides Cassandra system there are few NOSQL databases very popular among users,
- Apache's HBase
- MongoDB

3. **Features of Cassandra**

(i) **Scalability**

Cassandra is highly scalable system; it also allow to add more hardware as per data requirement.

(ii) **24X7 Availability**

- Cassandra less chances of failure.
- It is always available for business applications.

(iii) **Good performance**

Cassandra system can be scaled up linearly, which means, it increases your outputs as you increase the number of nodes in the cluster.

(iv) **Good Data storage**

- Cassandra can store almost all possible data formats including: structured, semi-structured, and unstructured.
- It can manage all change to your data structures as per your need.

(v) **Data distribution**

Cassandra system provides flexible data distribution, as per need of data replication across multiple data centers.

(vi) **Transaction support**

- Cassandra supports properties like Atomicity, Consistency, Isolation, and Durability which is properties of transaction i.e.ACID.
- Faster write Operations
- Cassandra system was mainly designed for using low cost.
- It is designed faster write operations and can store huge amount of data, with very good read efficiency.

4. **History of Cassandra**

- Cassandra was developed at Facebook for searching facebook inbox.
- Cassandra system was open-sourced by Facebook in July 2008.
- Cassandra was accepted into Apache Incubator in 2009.

4.11.1(B) Dynamo DB

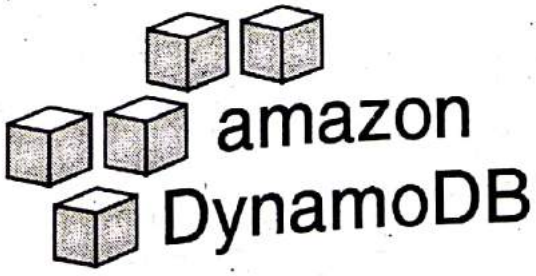


Fig. 4.11.3

1. **Data Model**

- Amazon's NOSQL database called DynamoDB in form of table is a collection of various items and each item is a collection of attributes.

- In a relational database, a table has a fixed schema of tables with primary key and list of its columns with data types.
- All tuples are of same schema.
- DynamoDB Data model contains,
  - Table
  - Items
  - Attributes
- DynamoDB requires only a primary key and does not require to define all of the attribute names and data types in advance.
- Each attribute of DynamoDB in an item is a name-value pair.

Primary Key

In order to create table we must specify primary key column name which identifies item set in table uniquely.

Partition Key

- A simple primary key has only one attribute known as the partition key.
- DynamoDB uses the partition key as input to hash function for internal use.

Partition Key and Sort Key

- A composite primary key is made of two attributes.
- The first attribute is the partition key, and the second attribute is the sort key.
- All items with the similar partition key are stored together, sorting is done in sorted order using sort key value.

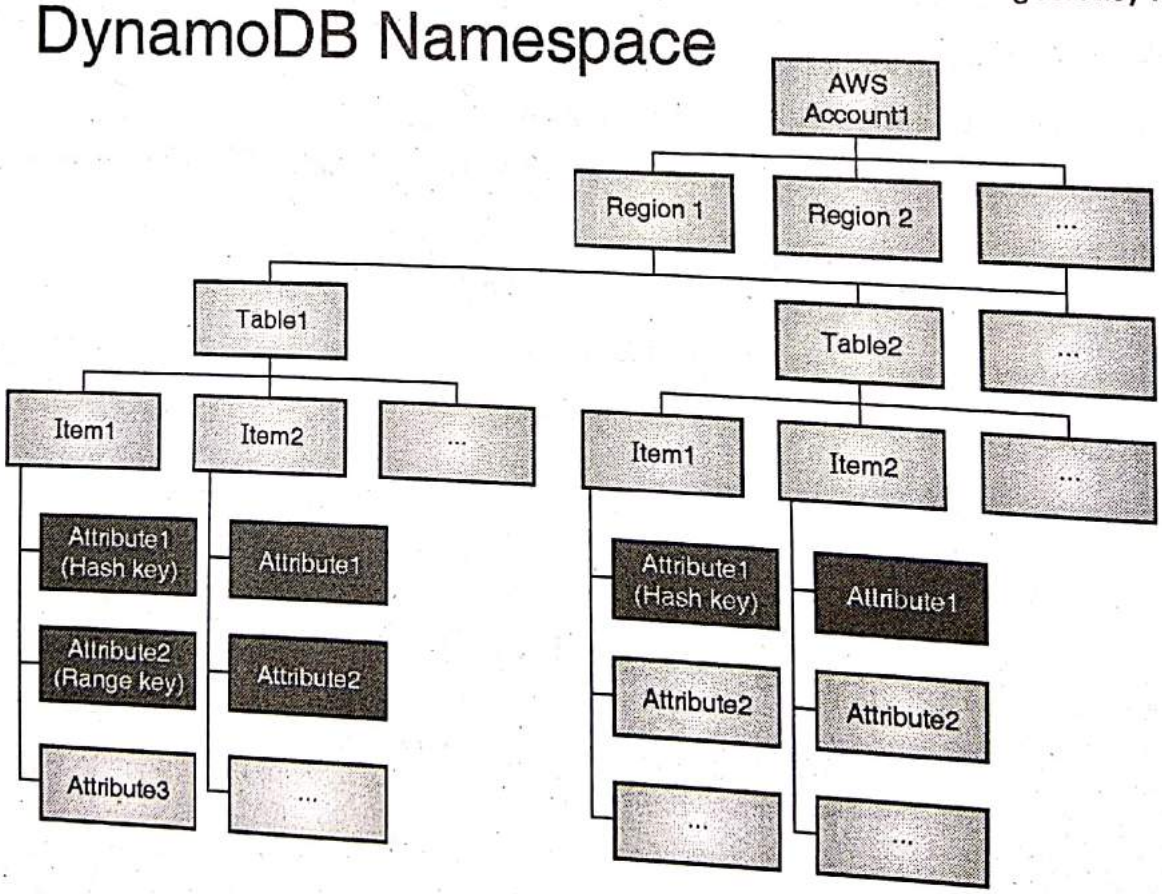


Fig. 4.11.4

2. Example

Consider a class of students with ID column as primary key,

StudentClass{ID,....}

```
{
 Id = 1011
 SubjectName = "Java"
 ISBN = "111-1111111111"
 Authors = [ "Author 1", "Author 2" ]
 Price = 150
 PageCount = 500
 Publication = TechMax
}

{
 Id = 1015
 SubjectName = "JDB"
 ISBN = "111-1111111111"
 Authors = ["Author 3"]
 Price = 1543
 PageCount = 5000
 Publication = TechMax
}
```

3. Data Type

- Amazon DynamoDB supports various data types
- **Scalar types :** Number, String, Binary, Boolean, and Null.
- **Document types :** List and Map.
- **Set types :** String Set, Number Set, and Binary Set.

4. CRUD Operations

(a) Table Operations

(i) CreateTable

- It is used to create new table on your account.
- To check status of table we can use DescribeTable command.

```
{
 "AttributeDefinitions": [
    {
```

```
        "AttributeName": "string",
        "AttributeType": "string"
    }
  ],


  "GlobalSecondaryIndexes": [
    {
      "IndexName": "string",
      "KeySchema": [
        {
          "AttributeName": "string",
          "KeyType": "string"
        }
      ],
      "Projection": {
        "NonKeyAttributes": [
          "string"
        ],
        "ProjectionType": "string"
      },
      "ProvisionedThroughput": {
        "ReadCapacityUnits": number,
        "WriteCapacityUnits": number
      }
    }
  ],
    "TableName": "string"
}
```

(ii) Readtables

The readtable operation used to read the tables which are created by create table command with the help o diagnostic tools such as list tables. Returns list of table names associated with the current account.

(iii) DeleteTable

The *DeleteTable* operation deletes a table and items associated with it.

(iv) UpdateTable

Modifies throughput, global secondary indexes for a given table.

(b) Item Operation

(i) BatchGetItem

The BatchGetItem operation returns the attributes of one or more items from tables.

(ii) BatchWriteItem

The BatchWriteItem operation deletes multiple items in tables.

(iii) DeleteItem

It will delete a single item with help of its primary key.

(iv) GetItem

GetItem operation returns a set of attributes for item with the given primary key.

(v) PutItem

It will create a new item, or replaces an old item with a new item.

(vi) UpdateItem

Edits an existing item's attributes to add new item to the table if it does not already exist.

(c) Others

(i) *Query*

A *Query* operation uses the primary key of a table to directly access items from that table.

(ii) *Scan*

The *Scan* operation returns one or more items and item attributes.

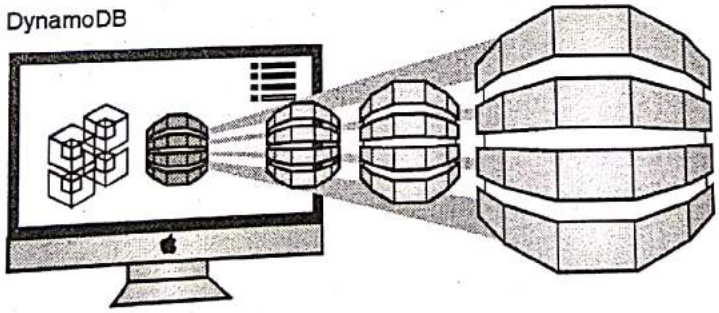


Fig. 4.11.5

5. Data Access

- Amazon DynamoDB is a web service uses HTTP and HTTPS as a transport layer services.
- JavaScript Object Notation (JSON) can be used as a message serialization format.
- Application code makes requests to the DynamoDB web service API.
- It is possible to use AWS Software Development Kits (SDKs).
- DynamoDB API libraries take care of request authentication, serialization, and connection management.

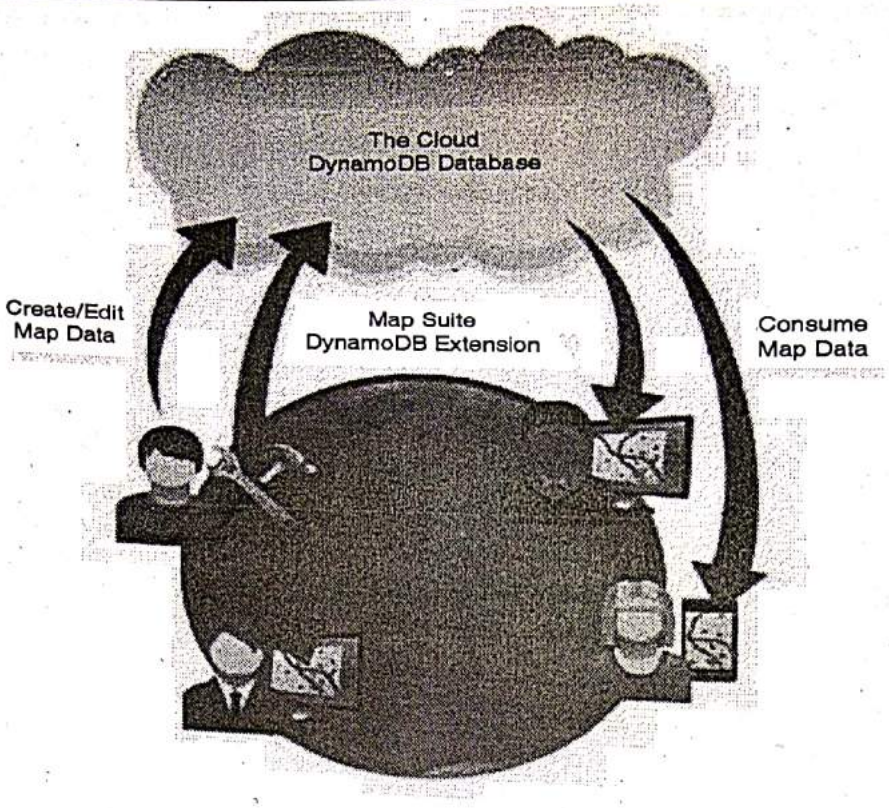


Fig. 4.11.6

6. Data Indexing

- In order to have efficient access to data in a table, Amazon DynamoDB creates and maintains indexes for the primary key attributes.
- Indexing allows applications to quickly retrieve data by specifying primary key.
- Applications may get good benefits from having one or more secondary keys available, for efficient data access with attributes other than the primary key.

Secondary Indexes

- In order to create table we must specify primary key column name which identifies item set in table uniquely.
- DynamoDB can create a table with a composite primary key it also supports one or more secondary indexes on table.
- A secondary index allows you to query data in the table using an alternate key like primary key.

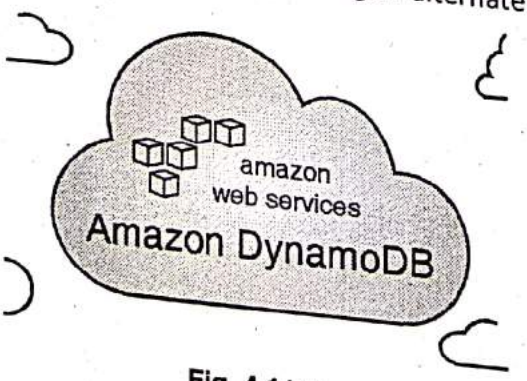


Fig. 4.11.7

Types of secondary indexes :

(i) Global Secondary Index

An index with a partition key and sort key, different from index on the table.

(ii) Local Secondary Index

An index that has the same partition and different sort key.

Review Question

Q. 1 Write a short note on Big Data.

Q. 2 Compare SQL and NoSQL databases.

Q. 3 Write a short note on Casssendra.

Q. 4 Write a short note on DynamoDB.

□□□

CHAPTER 5

Module - 4

Mining Data Streams

Syllabus

The Stream Data Model: A Data-Stream-Management System, Examples of Stream Sources, Stream Queries, Issues in Stream Processing, Sampling Data techniques in a Stream, Filtering Streams: Bloom Filter with Analysis, Counting Distinct Elements in a Stream, Count-Distinct Problem, Flajolet-Martin Algorithm, Combining Estimates, Space Requirements, Counting Frequent Items in a Stream, Sampling Methods for Streams, Frequent Itemsets in Decaying Windows, Counting Ones in a Window: The Cost of Exact Counts, The Datar-Gionis-Indyk-Motwani Algorithm, Query Answering in the DGIM Algorithm, Decaying Windows.

5.1 The Stream Data Model

Q. What is data stream? Explain data stream operations in the context of Big Data.

- A data stream is a flow of data which arrives at uncertain intervals of time.
- The data available from a stream is fundamentally different from the data stored in a conventional database in the sense that the data available in a database is complete data and can be processed at any time we want it to be processed. On the other hand, stream data is not completely available at any one point of time. Instead only some data is available with which we have to carry on our desired processing.
- Another problem with stream data is the storage of the entire data received from the stream. The rate of arrival is so rapid as well as the volume of data is so huge that it is not practically possible to save the entire data in a database. Rather it is necessary to summarize the stream data by taking appropriate samples.
- Generally, we have to perform the following fundamental operations to handle input data stream :
  (i) Gather information from the input data stream,
  (ii) Clear or filter the data,
  (iii) Apply standard modeling techniques,
  (iv) Deploy the generated solutions.

Diagrammatically we may represent the above four operations as shown in Fig. 5.1.1.

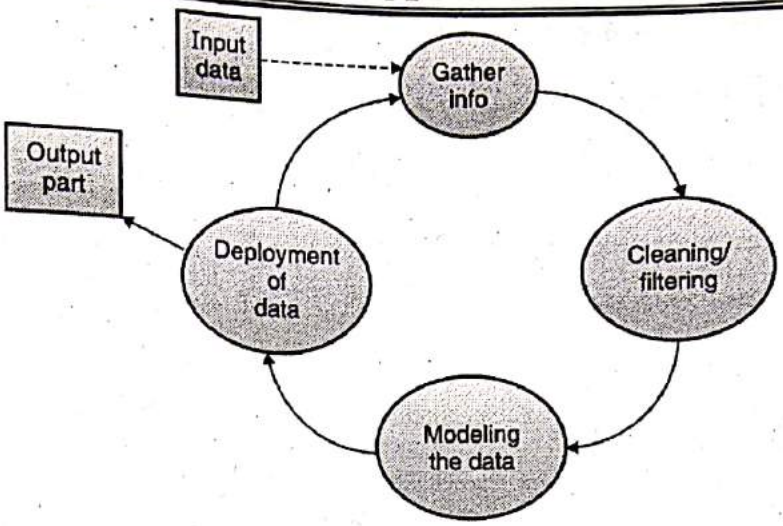


Fig. 5.1.1 : Stream data operations

- When data stream or data flow arrives at the compute node, then either it has to be stored in a database or processed immediately. Otherwise the data is lost forever.
- The major issues in this storage/processing task are :
  (i) The incoming rate of arrival of data is tremendous, and
  (ii) Different streams containing different categories of data items (images, text, alphanumeric, encoded, video data etc.) may be coming to the same compute node.
- It is practically not feasible to store the entire stream data as the storage will never be sufficient to accommodate it.
- Thus, it becomes necessary to summarize the incoming stream data in some way or other. Every algorithm dealing with stream data uses summarization. For this purpose, samples of the stream are taken and the stream is filtered to remove those portions which are not required for the computations.
- The next step is the estimation of all the different elements in the stream using just the samples obtained in the previous step. This drastically reduces the storage requirements.
- The summarization algorithms may be classified into the following two categories :
  (i) The first category involves algorithms described above which make use of good sampling techniques, efficient filtering (removal of irrelevant data items), and estimation of the stream elements.
  (ii) In the second category, algorithms are based on the concept of window. A window is nothing but the most recent n elements of the stream. Queries are then executed on the window considering it as a relation in a database. But the size and/or the number of windows may be large, so it is necessary to even summarize the windows.

5.1.1 A Data-Stream-Management System

MU - Dec. 16, May 17

Q. Explain abstract architecture of Data Stream Management system (DSMS). **(Dec. 16, 10 Marks)**

Q. What is a Data stream Management System? Explain with Block Diagram. **(May 17, 10 Marks)**

Q. Explain the data-stream-management system with neat diagram.

- A data-stream-management system is very similar to a conventional relational database-management system. Fig. 5.1.2 represents the organization of a data-stream-management system.

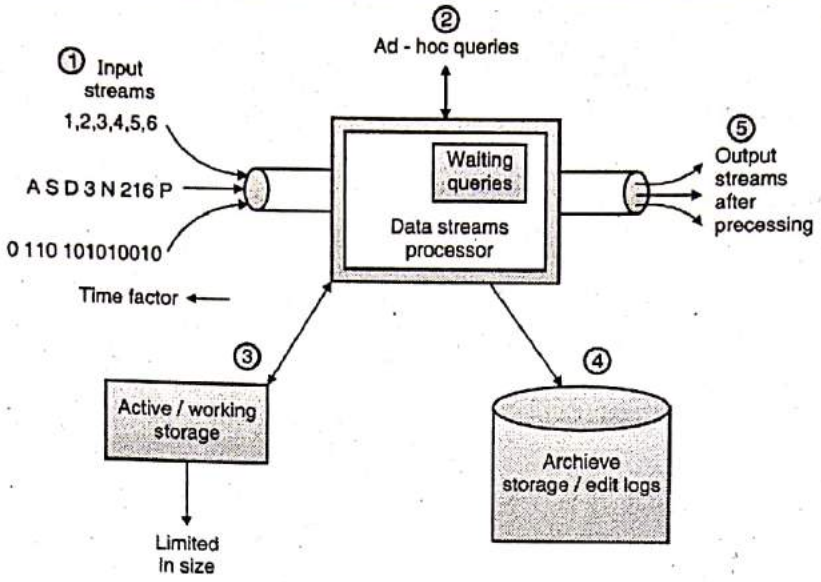


Fig. 5.1.2 : A data-stream-management system

– Let us understand the purpose of each of the components of the system :

1. **Input streams :** The input streams have the following characteristics:
   - There can be one or more number of input streams entering the system.
   - The streams can have different data types.
   - The rate of data flow of each stream may be different.
   - Within a stream the time interval between the arrival of data items may differ. For example, suppose the second data item arrives after 2 ms from the arrival of the first data item, then it is not necessary that the third data item will also arrive after 2 ms from the arrival of the second data item. It may arrive earlier or even later.
2. **Stream processor :** All types of processing such as sampling, cleaning, filtering, and querying on the input stream data are done here. Two types of queries are supported which are standing queries and ad-hoc queries. We shall discuss both the query types in details in the upcoming section.
3. **Working storage :** A limited memory such as a disk or main memory is used as the working storage for storing parts or summaries of streams so that queries can be executed. If faster processing is needed, main memory is used otherwise secondary storage disk is used. As the working storage is limited in size, it is not possible to store all the data received from all the streams.
4. **Archival storage :** The archival store is a large storage area in which the streams may be archived but execution of queries directly on the archival store is not supported. Also, the fetching of data from this store takes a lot of time as compared to the fetching of data from the working store.
5. **Output streams :** The output consists of the fully processed streams and the results of the execution of queries on the streams.

– The difference between a conventional database-management system and a data-stream-management system is that in case of the database-management system all of the data is available on the disk and the system can control the rate of data reads. On the other hand, in case of the data-stream-management system the rate of arrival of data is not in the control of the system and the system has to take care of the possibilities of data getting lost and take the necessary precautionary measures.

5.1.2 Examples of Stream Sources

Q. List and explain various data stream sources.

Following are some of the primary sources of stream data :

1. Sensor data

- Sensors are the devices which are responsible for reading and sending the measurements of various kinds of physical parameters such as temperature, wind speed, pressure, moisture content, humidity, pollution level, surface height, amount of light, etc.
- Consider a network of millions of sensors on the ocean surface to provide an early warning system for Tsunami by studying the behaviour of the ocean. This can save a lot of lives. But the amount of data received from all the sensors combined is simply huge.

2. Data streams related to Images

- High resolution image streams are relayed to the earth stations by the satellites which can amount to many terabytes of image data per day. Many such high-resolution images are released for the public from time to time by NASA as well as ISRO.
- Lower resolution image streams are produced by the CCTV cameras placed in and around important places and shopping centres. Now a days most of the public places and some of the private properties are under the surveillance of CCTV cameras 24×7.

3. Internet services and web services traffic

- The networking components such as switches and routers on the Internet receive streams of IP packets and route them to the proper destination. These devices are becoming smarter day by day by helping in avoiding congestion and detecting denial-of-service-attack, etc.
- Websites receive many different types of streams. Twitter receives millions of tweets, Google receives tens of millions of search queries, Facebook receives billions of likes and comments, etc. These streams can be studied to gather useful information such as the spread of diseases or the occurrence of some sudden event such as catastrophe.

5.1.3 Stream Queries

MU - May 17, Dec. 18

Q. With respect to data stream querying, give example of Ad-hoc queries. (May 17, Dec. 18, 2 Marks)

Q. What are stream queries? Explain different categories of stream queries.

In the previous section we have mentioned that the data-stream-management system supports two different types of queries:

1. Standing queries
2. Ad-hoc queries

Fig. 5.1.3 : Query types

1. Standing queries

- A standing query is a query which is stored in a designated place inside the stream processor. The standing queries are executed whenever the conditions for that particular query becomes true.
- For example, if we take the case of a temperature sensor then we might have the following standing queries in the stream processor :
  - Whenever the temperature exceeds 50 degrees centigrade, output an alert.
  - On arrival of a new temperature reading, produce the average of all the readings arrived so far starting from the beginning.
  - Output the maximum temperature ever recorded by the sensor, after every new reading arrival.

2. Ad-hoc queries

- An ad-hoc query is not predefined and is issued on the go at the current state of the streams. The nature of the ad-hoc queries cannot be determined in advance.
- To allow a wide range of arbitrary ad-hoc queries it is necessary to store a sliding window of all the streams in the working storage. A sliding window is nothing but the most recent elements in the stream. The number of elements to be accommodated in the sliding window has to be determined beforehand. As and when new elements arrive, the oldest ones will be removed from the window and hence the name sliding window.
- Instead of determining the size of the sliding window in advance we may also take another approach based on the unit of time. In this approach the sliding window may be designed to accommodate say all the stream data for an hour or a day or a month, etc.
- For example, a social networking website like Facebook may want to know the number of unique active users over the past one month.

5.1.4 Issues in Stream Processing

Q. Discuss different issues in data stream processing.

- The issues in stream processing mainly arise because of the following two basic reasons :
  1. The rapid rate of arrival of stream data, and
  2. The huge size of the data when all of the input streams are considered.
- Because of the rapid arrival of stream data, the processing speed also must match the arrival speed of the data. To achieve this, the entire stream processing algorithm must reside in main memory and there should be minimal secondary memory accesses. This is because the secondary memory accesses are many times slower than the main memory accesses.

- In the case of a slow stream it may be possible to process the data in that stream using a small portion of main memory. But if the number of such slow streams becomes large then again, the problem of shortage of main memory arises.
- Thus, one way to solve the issues related to stream processing is by having a large amount of main memory. But in real life compute nodes the amount of main memory available is usually limited which makes it unfeasible. So, we have to resort to some other way for solving the issues. One such way is to use the technique of sampling.

5.2 Sampling Data Techniques in a Stream

Q. What is sampling of data in a stream? How do we obtain representative sample ?

- A sample is a subset of a stream which adequately represents the entire stream. The answers to the queries on the sample can be considered as though they are the answers to the queries on the whole stream.
- Let us illustrate the concept of sampling with the example of a stream of search engine queries. A search engine may be interested in learning the behaviour of its users to provide more personalized search results or for showing relevant advertisements. Each search query can be considered as a tuple having the following three components: (user, query, time)

Obtaining a representative sample

- The first step is to decide what will constitute the sample for the problem in hand. For instance, in the search query stream we have the following two options :
  - o Take the sample of queries of each user, or
  - o Take the sample of users and include all the queries of the selected users.
- Option number 2 as a sample is statistically a better representation of the search query stream for answering the queries related to the behaviour of the search engine users.
- The next step is to decide what will be the size of the sample compared to the overall stream size. Here we will assume a sample size of 1/10th of the stream elements.
- When a new user comes into the system, a random integer between 0 and 9 is generated. If the number is 0, then the user's search query is added to the sample. If the number is greater than 0, then the user's search query is not added to the sample. A list of such users is maintained which shows which user's search query is to be included in the sample.
- When a new search query comes in the stream from any existing user, then the list of users is consulted to ascertain whether the search query from the user is to be included in the sample or not.
- For this method to work efficiently the list of users has to be kept in the main memory because otherwise disk access will be necessary to fetch the list which is a time-consuming task.
- But as the list of users grows it will become a problem to accommodate it into the main memory. One solution to this problem is to use a hash function as the random number generator. The hash function will map a user to a number between 0 to 9. If a user hashes to 0 then the search query of the user is added to the sample and otherwise it is not added.
- In general we can create a sample size of any rational fraction a/b by using a hash function which maps a user to a number between 0 and b-1 and the user's query is added to the sample if the hash value is less than a.

The general sampling problem

Q. Explain general sampling problem? What is effect on stream if we vary the sample size?

- Each tuple in the stream consists of n components out of which a subset of components called key on which the selection for the sample is based.
- For instance, in the search query example the key consists of only one component user out of the three components user, query and time. But it is not always necessary to consider only user as the key, we could even make query as the key or even the pair (user, query) as the key.
- So, to produce a sample of size a/b, we have to use a hash function which can map the key of a tuple to a value between 0 and b-1. The tuple will be added to the sample if the hash value is less than a.
- In case of a multi-component keys such as (user, query), the hash function has to do the extra work of combining the values of all the components in order to produce a single hash value.

Variation in sample size

- With the passage of time the sample size will keep on growing as new users as well as new search queries from existing users will be added to the sample. If we have a limit on the number of tuples that can be stored in the sample, we can follow the technique described below.
- We select such a hash function h which maps the keys to a very large number of values

$$0, 1, 2, \ldots, B-1.$$

We also initialize a threshold,

$$t = B-1$$

- A tuple from the stream is allowed into the sample if and only if

$$h(\text{key}) \leq t$$

- When the number of stored tuples reach the maximum limit, we can remove all those tuples whose key hash to the value B – 1 and modify the threshold as,

$$t = t-1$$

- We can even reduce t by more than 1 and remove all those tuples associated with the hash values higher than t.

5.3 Filtering Streams

Q. Explain the filtering process of data streams with suitable example.

- Filtering also known as selection is another important process which is applied to the streams. Through filtering only those tuples are accepted which satisfy the given criteria and other tuples are rejected.
- There are two categories of selection criteria :
  - The criteria based on any property of the tuple which can be calculated from the tuple itself. This is a case of easy filtering. For example, the selection criteria could be whether the tuple contains a query having the term "Big Data" in it.

- The criteria which involve the looking up of set membership. In this case the filtering becomes harder if the set is huge and cannot be stored in the main memory.

- Bloom filtering is a filtering technique which is used for eliminating or rejecting most of the tuples which do not satisfy the criteria.

Example of filtering

- Let us consider the example of spam email filtering. Let S be the set of safe and trusted email addresses. Assume that the size of S is one billion email addresses and the stream consist of the pairs (email address, email message).
- The set S cannot be accommodated in main memory because on average an email address is of minimum 20 bytes in size. So, to test for set membership in S, it becomes necessary to perform disk accesses. But as discussed earlier a disk access is many times slower than main memory access.
- We can do the spam filtering using only the main memory and no disk accesses with the help of Bloom filtering. In this technique the main memory is used as a bit array.
- Say we have 1 GB of main memory available for the filtering task. Since each byte consists of 8 bits, so 1 GB memory contains 8 billion bits. This means we have a bit array of 8 billion locations.
- We now need a hash function h which will map each email address in S to one of the 8 billion locations in the array. All those locations which get mapped from S are set to 1 and the other locations are set to 0.
- As there are 1 billion email addresses in S and 8 billion bits in main memory, so approximately 1/8th of the total available bits will be set to 1. The exact count of bits that are set to 1 will be less than 1/8th because more than one email address may hash to the same bit location.
- When a new stream element arrives, we simply need to hash it and check the contents of the bit location to which it is hashed. If the bit is 1, then the email is from a safe and trusted sender. On the other hand, if the bit is 0, then the email is a spam.

5.3.1 Bloom Filter with Analysis

MU - Dec. 17, Dec 18

Q. Explain the concept of a Bloom Filter using an example. (Dec. 17, 10 Marks)

Q. How Bloom filter is useful for big data analytics. Explain with one example. (Dec. 18, 10 Marks)

Q. What is Bloom filter? Explain Bloom filtering process with neat diagram.

The components of a Bloom filter are as follows :

- An array of n bits initialized to 0's.
- A set H of k hash functions each of which maps a key to one of the n bit locations :

  $h_1, h_2, ..., h_k$
- A set S consisting of m number of keys.
- Fig. 5.3.1 illustrates the block diagram of Bloom filter and its process.

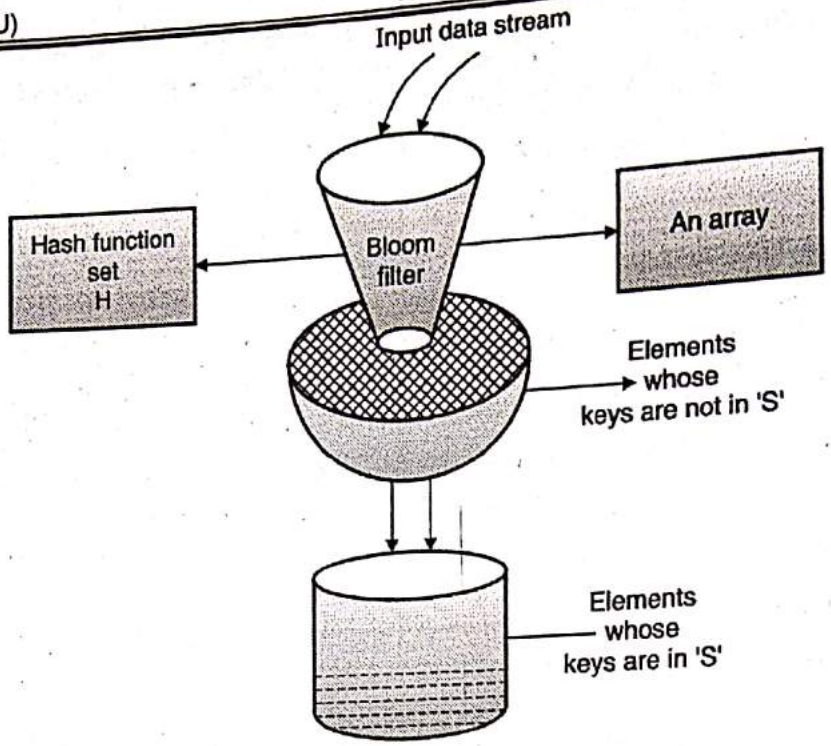


Fig. 5.3.1: Bloom filter

- Bloom filter accepts all those tuples whose keys are members of the set S and rejects all other tuples.
- Initially all the bit locations in the array are set to 0. Each key K is taken from S and one by one all of the k hash functions in H are applied on this key K. All bit locations produced by $h_i(K)$ are set to 1.

  $h_1(K), h_2(K), ..., h_k(K)$
- On arrival of a test tuple with key K, all the k different hash functions from H are once again applied on the test key K. If all the bit locations thus produced are 1, the tuple with the key K is accepted. If even a single bit location out of these turns out to be 0, then the tuple is rejected.

Analysis of bloom filtering

- The Bloom filter suffers from the problem of false positives. This means even if a key is not a member of S, there is a chance that it might get accepted by the Bloom filter.
- To find the probability of a false positive we need to use the model of a dart game in which darts are thrown at targets. Assume that there are x darts and y targets and the probability of any dart hitting any target is equally likely, then according to the theory of probability :
  - The probability of a given dart not hitting a given target is $\frac{x-1}{x}$
  - The probability of no dart hitting a given target is $\left(\frac{x-1}{x}\right)^y = \left(1-\frac{1}{x}\right)^{x\left(\frac{y}{x}\right)}$
- As we know, $(1-\epsilon)^{\frac{1}{\epsilon}} = \frac{1}{e}$ when ϵ is small. Thus, we can say that the probability of no dart hitting a given target is $\left(1-\frac{1}{x}\right)^{x\left(\frac{y}{x}\right)} = \left(\frac{1}{4}\right)^{y/x} = e^{-y/x}$

5.4 Counting Distinct Elements in a Stream

MU - Dec. 17

Q. Explain any one algorithm to count number of distinct elements in data stream. (Dec. 17, 10 Marks)

- Apart from sampling and filtering, one more simple and important processing on the stream data is that of counting the number of elements which are distinct in the given data stream.
- This also requires a large amount of main memory. But with the help of hashing and randomized algorithm we can achieve a good approximation of the count using only a small amount of memory.

5.4.1 Count - Distinct Problem

Q. Explain count distinct problem with suitable example.

- Let us illustrate the problem of count-distinct with the example of a website like Amazon or Facebook or Google which wants to know the number of monthly active unique users. These numbers are useful in preparing their server and other infrastructure for efficient load handling as well as for generation of advertising revenues.
- Here we assume that the elements of the stream belong to a universal set. The universal set for sites which have the login facility will be the set of all logins (usernames and passwords). Examples of such sites are Facebook, Twitter, Amazon, etc. But for sites like Google which do not require a login for searching, the universal set will be the set of all IP addresses.
- One way of performing the distinct user count will be to keep the entire list of elements (users) that have appeared on the stream in the main memory arranged in some efficient structure such as hash table or search tree.
- Whenever a new user visits the website, it is checked whether she is already there in the list or not. If she is not there in the list, she is added to the list. If she is already in the list, then no action is taken.
- This approach works fine till the number of users can easily fit the available main memory. If the number of users grows or there are a number of streams to be processed simultaneously, then it starts becoming a problem.
- This problem can be solved in a number of different ways :
  - By using a greater number of compute nodes but it increases the complexity of implementation.
  - By storing the list structure in secondary memory but it increases the time complexity by a huge factor.
- Thus, instead of trying to find the exact count of the distinct elements we can try to find an approximate count. This will ensure that the task is finished quickly using only a small portion of the main memory.

5.4.2 The Flajolet- Martin Algorithm

MU - Dec. 16, May 17

Q. Give problem in Flajolet-Martin (FM) algorithm to count distinct elements in a stream. (Dec. 16, 5 Marks)

Q. How to count distinct elements in a stream? Explain Flajolet-Ma1tin Algorithm. (May 17, 5 Marks)

Q. Explain Flajolet-Martin algorithm in detail.

- The Flajolet-Martin algorithm is used for estimating the number of unique elements in a stream in a single pass. The time complexity of this algorithm is O(n) and the space complexity is O(log m) where n is the number of elements in the stream and m is the number of unique elements.

- The major components of this algorithm are :
  - A collection of hash functions, and
  - A bit-string of length L such that $2^L > n$. A 64-bit string is sufficient for most cases.
- Each incoming element will be hashed using all the hash functions. Higher the number of distinct elements in the stream, higher will be the number of different hash values.
- On applying a hash function h on an element of the stream e, the hash value h(e) is produced. We convert h(e) into its equivalent binary bit-string. This bit string will end in some number of zeroes. For instance, the 5-bit string 11010 ends with 1 zero and 10001 ends with no zeroes.
- This count of zeroes is known as the tail length. If R denotes the maximum tail length of any element e encountered thus far in the stream, then the estimate for the number of unique elements in the stream is 2^R.
- Now to see that this estimate makes sense we have to use the following arguments using probability theory :
  - The probability of h(e) having a tail length of at least r is 2^{-r}.
  - The probability that none of the m distinct elements have tail length of at least r is $(1-2^{-r})^m$.
  - The above expression can also be written as $.((1-2^{-r})^{2r})^{m2-r}$
  - And finally, we can reduce it to e^{-m2-r} as $(1-2-r)^{2r} = e^{-1}$
- If $m >> 2^r$, the probability of finding a tail of length at least r approaches 1.
- If $m << 2^r$, the probability of finding a tail of length at least r approaches 0.
- So, we can conclude that the estimate of 2^R is neither going to be too low nor too high.
- Let us now understand the working of the algorithm with an example :

Stream: 5, 3, 9, 2, 7, 11

Hash function :

$$h(x) = 3x + 1 \bmod 32$$

$$h(5) = 3(5) + 1 \bmod 32 = 16 \bmod 32 = 16 = 10000$$

$$h(3) = 3(3) + 1 \bmod 32 = 10 \bmod 32 = 10 = 01010$$

$$h(9) = 3(9) + 1 \bmod 32 = 28 \bmod 32 = 28 = 11100$$

$$h(2) = 3(2) + 1 \bmod 32 = 7 \bmod 32 = 7 = 00111$$

$$h(7) = 3(7) + 1 \bmod 32 = 22 \bmod 32 = 22 = 10110$$

$$h(11) = 3(11) + 1 \bmod 32 = 34 \bmod 32 = 2 = 00010$$

Tail lengths: {4, 1, 2, 0, 1, 1}

$$R = \max(\text{Tail length}) = 4$$

$$\text{Estimate of } m = 2^R = 2^4 = 16.$$

- The above estimate is obtained from a single hash function. Similarly, the other hash functions will produce one estimate each. We will need a method of combining the estimates to arrive at the overall estimate.

5.4.3 Combining Estimates

Q. Explain the process of combining the Estimates.

- There are three approaches for combining the estimates from the different hash functions :
  - Average of the estimates, or
  - Median of the estimates, or
  - The combination of the above two.
- If we take the average of the estimates to arrive at the final estimate then it will be problematic in those cases where one or a few estimates are very large as compared to the rest of the others. Suppose the estimates from the various hash functions are 16, 25, 18, 32, 900, 23. The occurrence of 900 will take the average estimate to the higher side although most of the other estimates are not that high.
- The median is not affected by the problem described above. But a median will always be a power of 2. So, for example the estimate using a median will jump from $2^8 = 256$ to $2^9 = 512$, and there cannot be any estimate value in between. So, if the real value of m is say, 400, then neither 256 nor 512 is a good estimate.
- The solution to this is to use a combination of both the average and the median. The hash functions are divided into small groups. The estimates from the groups are averaged. Then the median of the averages is calculated which is the final estimate.
- Now even if a large value occurs in a group and makes its average large, the median of the averages will nullify its effect on the final estimate. The group size should be a small multiple of $\log_2 m$ so that any possible average value is obtained and this will ensure that we get a close estimate by using a sufficient number of hash functions.

5.4.4 Space Requirements

Q. Comment on space requirements in count distinct problem.

- We do not need to store the elements of the stream in main memory.
- The only data that needs to be stored in the main memory is the largest tail length computed so far by the hash function on the stream elements.
- So, there will be as many tail lengths as the number of hash functions and each tail length is nothing but an integer value.
- If there is only a single stream, millions of hash functions can be used on it. But a million hash functions are far more than what is necessary to arrive at a close estimate.
- Only when there are multiple streams to be processed simultaneously, we have to limit the number of hash functions per stream. Even in this case the time complexity of calculating the hash values is a bigger concern than the space constraint.

5.5 Counting Frequent Items in a Stream

Q. How frequent items in a stream are counted?

- There are two main differences between a stream and a file :
  - A stream has no end while every file ends at some point.

- The time of arrival of a stream element cannot be predicted in advance while the data in a file is already available.

- Moreover, the frequent items in a stream at some point of time may be different from the frequent items in the same stream at some other point of time.
- To continue our discussion, we need to understand the concept of an itemset. In the market-basket model of data we have two types of objects. One is items and the other is baskets. The set of items in a basket is called the itemset.
- In the next section we consider some of the sampling methods available for counting the frequent items in a stream. We will consider the stream elements as baskets of items.

5.5.1 Sampling Methods for Streams

- The simplest technique for estimating the frequent itemsets in a stream is to collect a few baskets and save them in a file. On this file we can run any frequent-itemsets algorithm. This algorithm will produce the estimate of the current frequent itemsets in the stream. The other stream elements which arrive during the execution of the algorithm can be stored in a separate file for processing later.
- After the completion of the first iteration we can run another iteration of the frequent-itemsets algorithm with :
  - A new file of baskets, or
  - The old file collected during the execution of the first iteration.
- We can go on repeating the above procedures for more iterations. This will result in a collection of frequent itemsets. If a particular itemset occurs in a fraction of the baskets that is lower than the threshold, it can be dropped from the collection.
- It should also be ensured that new frequent itemsets are added to the collection. For this to happen the following can be done :
  - A new segment of the baskets from the stream can be used as an input to the algorithm in some iteration.
  - Adding a few random itemsets in the collection and continuing the iterations.

5.5.2 Frequent Itemsets in Decaying Windows

- To use the concept of decaying windows for finding the frequent itemsets we need to keep the following points in mind :
  - The stream elements are not individual items, rather they are baskets of items. This means that many items appearing together is considered a single element here.
  - The target here is to find all of the frequent itemsets and not just limit ourselves to finding singleton itemsets. In other words, for each itemset received we need to initiate the count for not just that particular itemset but also all of its subsets. The problem here is that the number of subsets of even a not so big itemset may become unmanageable.
- To solve the first issue, we have to consider all the items appearing together as current items and do the following for each such item :
  - If the item has no current score, then its score is initialized to 1.
  - Otherwise, multiply the current score by the term (1 – c) and add 1 to the product. Here c is a small constant such as 10^{-9}.

- To solve the second issue, only those itemsets are scored whose all immediate proper subsets are already being scored.

5.6 Counting Ones in a Window

One of the most important operations performed on a stream is the operation of counting of the number of occurrences of a particular element in a stream. If we consider a binary stream and we have a window of size N bits on this stream, we may want to find the number of 1's in the last k bits of the window where $k \leq N$. Our main focus here is the case in which we cannot fit the entire window in the main memory.

5.6.1 The Cost of Exact Counts

- Again, there are two major approaches to the counting problem :
  1. Exact count
  2. Approximate count
- For the exact count approach, we need to store the entire N-bit window in the main memory. Otherwise it will not be possible to compute the exact count of the desired elements. Let us try to understand it with the following arguments.
- Let us suppose instead of storing the N-bit window in main memory, we store an n-bit representation of the N-bit window where N> n.
- Now, number of possible N-bit window sequences = 2^N.

  and, number of possible n-bit window representations = 2^n.

  Since, N > n, it implies $2^N > 2^n$.
- Clearly we can see that the number of representations are not sufficient to represent all possible window sequences. Hence by the pigeon-hole principle there exists at least two different window sequences p and q which are represented by the same window representation in main memory.
- As $p \neq q$, it means they must differ in at least one bit position. But since both of them are having the same representation, the answer to the query of the number of 1's in the last k bits is going to be the same for both p and q, which is clearly wrong.
- Thus, from the above discussion we can conclude that it is totally necessary to store the entire N-bit window in memory for exact counts.

5.6.2 The DGIM Algorithm (Datar – Gionis – Indyk - Motwani)

MU – May 16, May 18, Dec. 18, May 19

Q. Using an example bit stream explain the working of the DGIM algorithm to count number of 1's (Ones) in a data stream. **(May 16, May 18, 5/10 Marks)**

Q. Explain DGIM algorithm for counting ones in stream with example. **(Dec. 18, May 19, 10 Marks)**

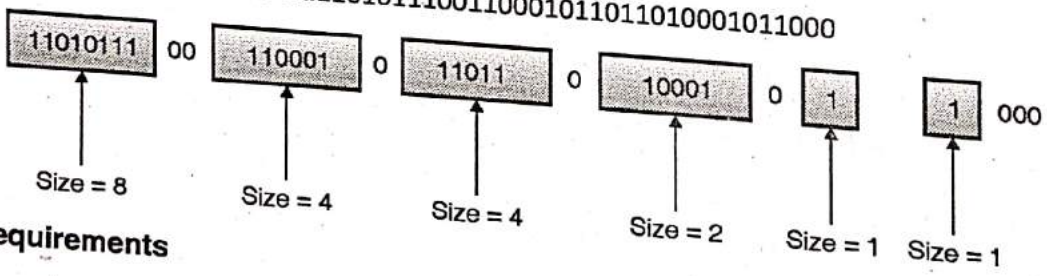
- In the simplest version of DGIM algorithm an N-bit window is represented using $O(\log^2 N)$ bits.
- The maximum error in the estimation of the number of 1's in the window in case of this algorithm is 50%.

- The two basic components of this algorithm are :
  - Timestamps, and
  - Buckets.
- Each bit arriving in the stream is assigned a timestamp. The timestamp is nothing but the sequence number of the bit in the order of arrival. So, the first bit is assigned timestamp 1, the second bit is assigned timestamp 2, the third bit is assigned timestamp 3 and each subsequent bit is assigned a timestamp one higher than the previous bit.
- The window is divided into buckets. Each bucket is described by :
  - The timestamp of the right most bit, and
  - The size of the bucket which is the number of 1's inside the bucket. The size must be a power of 2.

The following six conditions must be satisfied by the buckets :

1. The right end of every bucket must be occupied by a 1.
2. Every 1 should be inside some bucket.
3. A bit cannot be inside more than one bucket. In other words the buckets cannot overlap.
4. The number of buckets of a particular size can be either one or two. There will also be a limit on the maximum size of the bucket for a particular stream.
5. The size of a bucket is always a power of 2.
6. On moving from right to left the size of the buckets will increase.

Consider the following stream of bits : ...1101011100110001011011010001011000



Storage space requirements

The storage space required for the DGIM algorithm can be determined as follows :

1. A single bucket is represented using O(log N) bits.
2. Number of buckets is O (log N).
3. Total space required = O ($\log^2$ N).

5.6.3 Query Answering in the DGIM Algorithm

- If the query is to calculate the number of 1's in the last k bits of the window then :
  - Find the oldest timestamp bucket b which has at least one bit of the most recent k bits.
  - The estimate of the number of 1's = Half of the size of the oldest bucket + sizes of all the newer buckets.

- Let us apply the above mentioned steps to the buckets in the previous example. Suppose we want to estimate the number of 1's in the most recent 16 bits, then we observe that starting from the rightmost side the latest 16 bits fall inside both the size 1 buckets, one size 2 bucket and partially the size 4 bucket. This means in this case the oldest timestamp bucket is the size 4 bucket.
- Thus the estimate of the number of 1's in the latest 16 bits = (4/2) + 2 + 1 + 1 = 6. But the actual number of 1's is 7.

5.6.4 Decaying Windows

Q. What is Decaying Windows? Explain in detail.

- The problem with the sliding window of fixed size is that it does not take into account the older elements which are outside the window.
- A solution to this is the use of exponentially decaying windows which take into account all of the elements in the stream but assign different weightages to them. The recent elements are given more weightage compared to the older elements.
- This type of window is suitable for answering the queries on the most common recent elements. For example the most popular current movies, or the most popular items bought on Flipkart recently, or the top most current tweets, etc.
- Suppose the elements of a stream are :

  $e_1, e_2, e_3, \ldots, e_t$

  where e_1 is the oldest element and e_t is the most recent.

  Then the exponentially decaying window is the following sum :

  $$\sum_{i=0}^{t-1} e_{t-i} (1-c)^i$$

  where c is a small constant such as 10^{-9}.

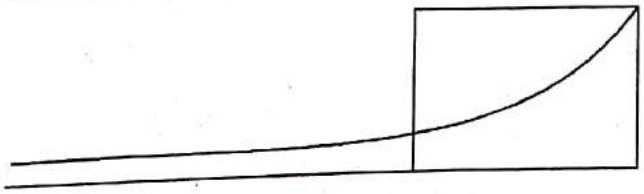


Fig. 5.6.1 : Fixed length window vs decaying window

- Fig. 5.6.1 shows the difference between a fixed length sliding window and an exponentially decaying window of equal weight. The rectangular box represents the fixed length window.
- When a new element e_{t+1} arrives in the stream, the following steps are taken :
  - Multiply the current sum by the term $(1-c)$, and
  - Add e_{t+1}.
- The major advantage of the decaying window is that we don't need to worry about the older elements going out of the window on arrival of new elements.

Review Questions

Q.1 What is data stream ? Explain data stream operations in the context of Big Data.

Q.2 What is stream Data Model ? Explain in detail.

Q.3 Explain the data-stream-management system with neat diagram.

Q. 4 List and explain various Data stream sources.

Q. 5 What are stream Queries ? Explain different Categories of stream Queries.

Q. 6 Discuss different issues in Data stream processing.

Q. 7 What is sampling of Data in a stream ? How do we obtain representative sample ?

Q. 8 Explain General Sampling problem. What is effect on stream if we vary the sample size?

Q. 9 Explain the filtering process of data streams with suitable example.

Q. 10 What is bloom filter ? Explain Bloom filtering process with neat diagram.

Q. 11 What is bloom filter ? Analyze the Bloom filter for performance.

Q. 12 Explain count distinct problem with suitable example.

Q. 13 Explain Flajolet-Martin algorithm in detail.

Q. 14 Explain the process of combining the Estimates. Also comment on space requirements.

Q .15 How frequent items in a stream are counted ?

Q. 16 What is cost of exact counts ?

Q. 17 Explain Datar-Gionis-Indyk-Motwani Algorithm in detail.

Q. 18 What is Decaying windows ? Explain in detail.

CHAPTER 6

Module - 5

Finding Similar Items

Syllabus

Distance Measures : Definition of a Distance Measure, Euclidean Distances, Jaccard Distance, Cosine Distance, Edit Distance, Hamming Distance.

6.1 Distance Measures

MU - May 16, Dec. 18

Q. Write a note on different distance measures that can be used to find similarity/ dissimilarity between data points in a big data set. **(May 16, 10 Marks)**

Q. What are distance measures? Brief any two distance measures. **(Dec. 18, 5 Marks)**

Q. What is a distance measure? Explain different criteria's regarding distance measures.

- A set of points is called a space. A space is necessary to define any distance measure. Let x and y be two points in the space, then a distance measure is defined as a function which takes the two points x and y as input, and produces the distance between the two points x and y as output. The distance function is denoted as :

  $d(x, y)$

- The output produced by the function d is a real number which satisfies the following axioms:

1. **Non-negativity :** The distance between any two points can never be negative.

   $d(x, y) \geq 0$

2. **Zero distance :** The distance between a point and itself is zero.

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

3. **Symmetry :** The distance from x to y is same as the distance from y to x.

   $d(x, y) = d(y, x)$

4. **Triangle inequality :** The direct distance between x and y is always smaller than or equal to the distance between x and y via another point z. In other words, distance measure is the length of the shortest path between two points x and y.

   $d(x, y) \leq d(x, z) + d(z, y)$

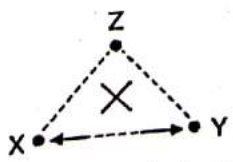


Fig. 6.1.1 : Triangle Inequality

In this section we shall discuss about the following distance measures in details :

(1) Euclidean Distance (2) Jaccard Distance

(3) Cosine Distance (4) Edit Distance

(5) Hamming Distance

6.1.1 Euclidean Distances

Q. What do you mean by Euclidean distance ? Explain with example.

- The Euclidean distance is the most popular out of all the different distance measures.
- The Euclidean distance is measured on the Euclidean space. If we consider an n-dimensional Euclidean space then each point in that space is a vector of n real numbers. For example, if we consider the two-dimensional Euclidean space then each point in the space is represented by (x_1, x_2) where x_1 and x_2 are real numbers.
- The most familiar Euclidean distance measure is known as the L_2– norm which in the n-dimensional space is defined as :

$$d([x_1, x_2,\ldots x_n], [y_1, y_2,\ldots y_n]) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

- For the two-dimensional space the L_2– norm will be :

$$d([x_1, x_2], [y_1, y_2]) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}$$

- We can easily verify all the distance axioms on the Euclidean distance :
  1. **Non-negativity :** The Euclidean distance can never be negative as all the sum terms $(x_i - y_i)$ are squared and the square of any number whether positive or negative is always positive. So the final result will either be zero or a positive number.
  2. **Zero distance :** In case of the Euclidean distance from a point to itself all the x_i's will be equal to the y_i's. This in turn will make all the sum terms $(x_i - y_i)$ equal to zero. So the final result will also be zero.
  3. **Symmetry :** $(x_i - y_i)^2$ will always be equal to $(y_i - x_i)^2$. So the Euclidean distance is always symmetric.
  4. **Triangle inequality :** In Euclidean space, the length of the side of a triangle is always less than or equal to the sum of the lengths of the other two sides.
- Some other distance measures that are used on the Euclidean space are :
  1. L_r– norm where r is a constant :

$$d([x_1, x_2,\ldots x_n], [y_1, y_2,\ldots y_n]) = \left(\sum_{i=1}^{n} |x_i - y_i|^r\right)^{\frac{1}{r}}$$

  2. L_1– norm which is commonly known as Manhattan distance :

$$d([x_1, x_2,\ldots x_n], [y_1, y_2,\ldots y_n]) = \sum_{i=1}^{n} |x_i - y_i|$$

  3. L_∞– norm which is defined as :

$$d([x_1, x_2,\ldots x_n], [y_1, y_2,\ldots y_n]) = \max (|x_i - y_i|) \; \forall \; i$$

Ex. 6.1.1 : Consider the two points (10, 4) and (6, 7) in the two-dimensional Euclidean space. Find the Euclidean distance between them.

Soln. :

(1) L_2– norm $= \sqrt{(10-6)^2 + (4-7)^2}$

$= \sqrt{4^2 + 3^2}$

$= \sqrt{16 + 9}$

$= \sqrt{25}$

$= 5$

(2) L_1 – norm $= |10-6| + |4-7|$

$= 4 + 3$

$= 7$

(3) L_∞– norm $= \max(|10-6|, |4-7|)$

$= \max(4, 3)$

$= 4$

6.1.2 Jaccard Distance

MU - May 18

Q. What do you mean by Jaccard Similarity? (May 18, 2 Marks)

- Jaccard distance is measured in the space of sets. Jaccard distance between two sets is defined as :

  $d(x, y) = 1 - SIM(x, y)$

  SIM(x, y) is the Jaccard similarity which measures the closeness of two sets. Jaccard similarity is given by the ratio of the size of the intersection and the size of the union of the sets x and y.

- We can verify the distance axioms on the Jaccard distance :

1. **Non-negativity :** The size of the intersection of two sets can never be more than the size of the union. This means the ratio SIM(x, y) will always be a value less than or equal to 1. Thus d(x, y) will never be negative.
2. **Zero distance :** If x = y, then $x \cup x = x \cap x = x$. In this case SIM(x, y) = x/x = 1. Hence, d(x, y) = 1 – 1 = 0. In other words the Jaccard distance between the same set and itself is zero.
3. **Symmetry :** As both union as well as intersection are symmetric $x \cup y = y \cup x$ and $x \cap y = y \cap x$, hence Jaccard distance is also symmetric d(x, y) = d(y, x).
4. **Triangle inequality :** Jaccard distance can also be considered as the probability that a random minhash function does not map both the sets x and y to the same value.

   $P[h(x) \neq h(y)] \leq P[h(x) \neq h(z)] + P[h(z) \neq h(y)]$

   Where h is the random minhash function.

6.1.3 Cosine Distance

Q. What is Cosine distance? Explain with suitable example.

- The cosine distance is measured in those spaces which have dimensions. Examples of such spaces are :
  1. Euclidean spaces in which the vector components are real numbers, and
  2. Discrete versions of Euclidean spaces in which the vector components are integers or Boolean (0 and 1).
- Cosine distance is the angle made by the two vectors from the origin to the two points in the space. The range of this angle is between 0 to 180 degrees.
- The steps involved in calculating the cosine distance given two vectors x and y are :
  1. Find the dot product x.y :

$$([x_1, x_2, \ldots x_n], [y_1, y_2, \ldots y_n]) = \sum_{i=1}^{n} x_i y_i$$

  2. Find the L_2– norms of both x and y,
  3. Divide the dot product x.y by the L_2– norms of x and y to get cos θ where θ is the angle between x and y.
  4. Finally, to get θ use the $\cos^{-1}$ function.
- Let us illustrate the concepts with an Ex. 6.1.2.
- The distance axioms on the Cosine distance may be verified as follows :
  1. **Non-negativity :** The range of the possible values is from 0 to 180 degrees. So there is no question of negative distance.
  2. **Zero distance :** If two vectors are the same direction, only then the angle between them will be zero.
  3. **Symmetry :** The angle between x and y will always be equal to the angle between y and x.
  4. **Triangle inequality :** The sum of the rotations from x to z and then z to y can never be less than the direct rotation from x to y.

Ex. 6.1.2 : Consider the following two vectors in the Euclidean space :
x = [1, 2, – 1], and y = [2, 1, 1].
Calculate the cosine distance between x and y.

Soln. :

Given x = [1,2, – 1] ;y = [2, 1, 1]

(i) $x \cdot y = [1\times2] + [2\times1] + [(-1)\times1]$

$= 2 + 2 + (-1) = 2 + 2 - 1$

$= 4 - 1 = 3$

(ii) L_2 norm for x $= \sqrt{(1)^2 + (2)^2 + (-1)^2} = \sqrt{6}$

L_2 norm for y $= \sqrt{(2)^2 + (1)^2 + (1)^2} = \sqrt{6}$

(iii) $\cos\theta = \dfrac{x,y}{(L_2 - \text{norm of x}), (L_2 - \text{norm of y})} = \dfrac{3}{\sqrt{6},\sqrt{6}} = \dfrac{3}{6} = \dfrac{1}{2}$

Now,

$$\text{Cos}^{-1} = 60°$$

∴ The angle between the two vectors x and y is 60°.

6.1.4 Edit Distance

Q. What is Edit distance? Explain with classical method.
Q. What is Edit distance? Explain with Longest Common Sequence (LCS) method.

- The points in the space for edit distance are strings.
- There are two different methods for defining and calculating edit distances :
  1. The classical method
  2. The Longest Common Subsequence (LCS) method

(I) Classical method

- The Edit distance between two strings x and y is the least number of edit operations required to convert x into y. Only two edit operations are allowed :
  1. Insertion of a single character, and
  2. Deletion of a single character.
- To illustrate let us take the following two strings :

x = JKLMN

y = JLOMNP

- For calculating the Edit distance between x and y we have to convert string x into string y using the edit operations of insertion and deletion.
- At first compare the character sequences in both the strings :

x	=	J	K	L	M	N	
y	=	J	L	O	M	N	P
		↓	↓	↓	↓	↓	↓
		①	②	③	④	⑤	⑥ → Positions

- Clearly, positions②,③and⑥are having different characters in x and y. So we need to make the necessary insertions and deletions at these three positions.

K	L	–
L	O	P
↓	↓	↓
②	③	⑥

- From position ② of string x, we have to delete the character K. The characters following K will be shifted one position to the left.

- After the first edit operation (deletion)the status of the string x is :

x =	J	L	M	N
	①	②	③	④

- Now the character O has to be inserted at position 3 i.e. after the character L and before the character M.
- After the second edit operation (insertion) the status of the string x is :

x =	J	L	O	M	N
	↓	↓	↓	↓	↓
	①	②	③	④	⑤

- In the final step, the character P has to be inserted in the string x at position 6.
- After the third and final edit operation (insertion) the status of the string x is :

X =	J	L	O	M	N	P
	↓	↓	↓	↓	↓	↓
	①	②	③	④	⑤	⑥

- So the Edit distance between x and y is :

$$d(x, y) = \text{Number of deletions} + \text{Number of insertions}$$
$$= 1 + 2 = 3$$

(ii) Longest Common Subsequence (LCS)

- The longest common subsequence of two strings x and y is a subsequence of maximum length which appears in both x and y in the same relative order, but not necessarily contiguous.
- Let us illustrate the concept of finding the Edit distance using LCS method with the same set of strings as in the previous method :

x =	J	K	L	M	N	
y =	J	L	O	M	N	P

- The longest common subsequence in x and y = JLMN.
- The formula of Edit distance using LCS is :

$$d(x, y) = \text{length of string } x + \text{length of string } y - 2 \times (\text{length of LCS})$$

Here,

$$\text{length of string } x = 5,$$
$$\text{length of string } y = 6,$$
$$\text{length of LCS} = 4,$$

So,

$$d(x, y) = 5 + 6 - 2 \times (4)$$
$$= 11 - 8$$
$$= 3$$

- The distance axioms on the Edit distance may be verified as follows :

1. **Non-negativity :** To convert one string into another string at least zero or more insertions and/or deletions are necessary. So, the Edit distance can never be negative.
2. **Zero distance :** Only in the case of two identical strings, the Edit distance will be zero.
3. **Symmetry :** The edit distance for converting string x into string y will be the same for converting string y into string x as the sequence of insertions and deletions can be reversed.
4. **Triangle inequality :** The sum of the number of edit operations required for converting string x into string z and then string z into string y can never be less than the number of edit operations required for converting the string x directly into the string y.

6.1.5 Hamming Distance

MU - Dec. 17

Q. Explain Hamming distance measure with an example. (Dec. 17, 5 Marks)

- Hamming distance is applicable in the space of vectors. Hamming distance between two vectors is the number of components in which they differ from each other.
- For example, let us consider the following two vectors :

x	=	1	0	0	0	1	1
y	=	1	1	1	0	1	0
		↓	↓	↓	↓	↓	↓
		①	②	③	④	⑤	⑥ → Positions

- The Hamming distance between the above two vectors is 3 because components at positions 2, 3 and 6 are different.
- The distance axioms on the Hamming distance may be verified as follows :

1. **Non-negativity :** Any two vectors will differ in at least zero or more component positions. So, the Hamming distance can never be negative.
2. **Zero distance :** Only in the case of two identical vectors, the Hamming distance will be zero.
3. **Symmetry :** The Hamming distance will be the same whether x is compared with y or y is compared with x.
4. **Triangle inequality :** The number of differences between x and z, plus the number of differences between z and y can never be less than the number of differences between x and y.

Review Questions

Q. 1 What is Jaccard similarity ?

Q. 2 What is distance Measure ? Explain different criteria's regarding distance measures.

Q. 3 What do you mean by Euclidean distance ? Explain with example.

Q. 4 What is Cosine distance ? Explain with suitable example.

Q. 5 Consider following are the two vectors in Euclidean space X = [1, 2, – 1] and Y = [2,1,1]. Calculate the cosine distance between X and Y.

Q. 6 What is Edit distance ? Explain with classical method.

Q. 7 What is Edit distance ? Explain with Longest Common Subsequence (LCS) method.

CHAPTER 7

Module - 5

Clustering

Syllabus

CURE Algorithm, Stream-Computing, A Stream-Clustering, Algorithm, Initializing & Merging Buckets, Answering Queries

7.1 Introduction

Q. What is clustering algorithm?

Cluster Using REpresentative i.e. CURE is very efficient data clustering algorithm for specifically large databases. CURE is robust to outliers.

Traditional clustering algorithm :

- In traditional clustering, it selects for any one point and it is only point considered as a cluster i.e. clusters centroid approach.
- Points in a cluster appear close to each other compared to other data points of any other clusters. It works in eclipse shape in better way.
- Drawback of traditional clustering algorithm is all-points approach makes algorithm highly sensitive to outliers and a minute change in position of data points.
- Cluster centroid and all points approach not work on arbitrary shape.

7.2 CURE Algorithm

MU - May 16, Dec. 16, May 18, May 19

Q. Clearly explain how the CURE algorithm can be used to cluster big data sets. (May 16, May 19, 10 Marks)

Q. Explain CURE algorithm for large scale clustering. (Dec. 16, 10 Marks)

Q. Explain the CURE algorithm for clustering large datasets. Please illustrate the algorithm using appropriate figures. (May 18, 10 Marks)

- CURE algorithm works better in spherical as well as non-spherical clusters.
- CURE : An efficient clustering algorithm for large database : sudipto Guha, Rajeev Rastogi, Kyuseok Shim.
- It prefers a set of points which are scattered as representative cluster than all-points or centroid approach.
- CURE uses random sampling and partitioning to speed up clustering.

7.2.1 Overview of CURE (Cluster Using REpresentative)

Data

↓

Make random sample

↓

Make partitioning of sample

↓

Partially cluster partitions

↓

Eliminate outliers

↓

Cluster partial clusters

↓

Label data in disk

7.2.2 Hierarchical Clustering Algorithm

Q. Write procedure of CURE cluster algorithm?

- A centroid-based point 'c' is chosen. All remaining scattered points are just at a fraction distance of α to get shrunk towards centroid.
- Such multiple scattered points help to discover in non spherical cluster i.e. elongated cluster.
- Hierarchical clustering algorithm uses such space which is linear to input size n.
- Worst-case time complexity is O (n^2 long n) and it may reduce to O(n^2) for lower dimensions.

CURE algorithm : CURE cluster procedure

- It is similar to hierarchical clustering approach. But it use sample point variant as cluster representative rather than every point in the cluster.
- First set a target sample number C. Then we try to select C well scattered sample points from cluster.
- The chosen scattered points are shrunk towards the centroid in a fraction of α where $0 \leq \alpha \leq 1$.

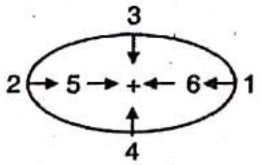


Fig. 7.2.1

- These points are used as representative of clusters and will be used as point in d_{min} cluster merging approach.
- After each merging, C sample points will be selected from original representative of previous clusters to represent new cluster.

- Cluster merging will be stopped until target K cluster is found.

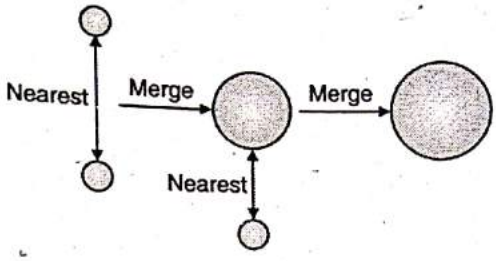


Fig. 7.2.2

7.2.2(A) Random Sampling and Partitioning Sample

MU - May 16

Q. Describe any two sampling techniques for big data with the help of examples. **(May 16, 10 Marks)**

Q. What is sampling? Explain random sampling and partition sampling.

- To reduce size of input to CURE's clustering algorithm random sampling is used in case of large data sets.
- Good clusters can be obtained by moderate size random samples, it provides tradeoff between efficiency and accuracy.
- Portioning sample reduces time required for execution because before final cluster made each partition get clustered whenever it is in pre-clustered data format at eliminated outliers.

7.2.2(B) Eliminate Outlier's and Data Labelling

Q. Write pseudo function of cluster algorithm.

Q. Write procedure for merging cluster in CURE.

- Outliers points are generally less than number in cluster.
- As random sample gets clustered, multiple representative points from each cluster are labelled with data set remainders.
- Clustering based on scattered point i.e. CURE approach found most efficient compared to centroid or all-points approach of traditional clustering algorithm.

Pseudo function of CURE (clustering algorithm)

```
Procedure cluster (s, k)
Begin
    T : =  build – kd – tree (s)
    Q : =  build – heap (s)
While size (Q) > k
do  {
    u : = extract – min (Q)
    v : = u – closest
    delete (Q, v)
    w : =  merge (u, v)
    delete – rep (T, u) ;
```

```
delete – rep (T, v) ;
insert – rep (T, w) ;
w – closest : = x
for each x ∈ Q
do {
    if dist (w, x) < dist (w, w – closest)
    w – closest : = x
    if x – closest is either u or v {
    if dist (x, x – closest) < dist (x, w)
    x – closest : = closest – cluster
    (T, x, dist (x, w))
else
x – closest : = w
relocate (Q, x)
 }
else if dist (x, x–closest) > dist (x, w){
x – closest : = w
relocate (Q, x)
 }
}
insert (Q, w)}
```

Procedure for merging clusters

```
Procedure merge (u, v)
being
w : = u U v
w. mean : = |u| u.mean + |v| v-mean/|u| + |v|
'tmpset : = ϕ
For i : = 1 to c do {
maxDist : = 0
for each point p in cluster w do {
if i = 1
min Dist : = dist (p, w, mean)
else
min Dist : = min {dist (p, q) : q ∈ tmpset}
if (min Dist > max Dist)
{ max Dist : = min Dist
Max point : = P
}
}

tmpset : = tmpset U {maxpoint}
}
```

```
For each point P in tempest do
w_rep := w_rep ∪ {p + α * (w-mean-p)}
return w
end
```

7.3 Stream Computing

Q. What is stream computing?

- Stream computing is useful in real time system like count of items placed on a conveyor belt.
- 1BM announced stream computing system in 2007, which runs 800 microprocessors and it enables to software applications to get split to task and rearrange data into answer.
- AT1 technologies derives stream computing with Graphical Processors (GPUs) working with high performance with low latency CPU to resolve computational issues.
- AT1 preferred stream computing to run application on GPU instead of CPU.

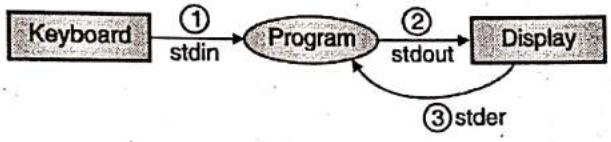


Fig. 7.3.1 : Standard stream for input, output and error

7.3.1 A Stream - Clustering Algorithm

Q. Explain BDMO algorithm.

- BDMO Algorithm has complex structures and it is designed in approach to give guaranteed performance even in worst case.
- BDMO designed by B. Bahcock, M. Datar, R. Motwani and L. OCallaghan.

Details of BDMO algorithm

(i) Stream of data are initially partitioned and later summarized with help of bucket size and bucket is a power of two.

(ii) Bucket size has few restrictions size of buckets are one or two of each size within a limit. Required bucket may start with sized or twice to previous for example bucket size required are 3, 6, 12, 24, 48 and so on.

(iii) Bucket size are restrained in some scenario, buckets mostly O (log N).

(iv) Bucket consists with contents like size, timestamp, number of points in cluster, centriod etc.

Few well – known algorithm for data stream clustering are :

(a) Small – Spaces algorithm
(b) BIRCH
(c) COBWEB
(d) C2ICM

7.4 Initializing and Merging Buckets

Q. What is bucket, how it is used for clustering?
Q. Explain in brief initializing and merging of bucket.

- A small size 'p' is chosen for bucket where p is power of 2. Timestamp of this bucket belongs to a timestamp of most recent points of bucket.
- Clustering of these points done by specific strategy. Method preferred for clustering at initial stage provide the centriod or clustroids, it becomes record for each cluster.

  Let,

  * 'p' be smallest bucket size.
  * Every p point, creates a new bucket, where bucket is time stamped along with cluster points.
  * Any bucket older than N is dropped
  * If number of buckets are 3 of size p

  p → merge oldest two
- Then propagated merge may be like (2_p, 4_p, ...).
- While merging buckets a new bucked created by review of sequence of buckets.
- If any bucket with more timestamp than N time unit prior to current time, at such scenario nothing will be in window of the bucket such bucket will be dropped.
- If we created p bucket then two of three oldest bucket will get merged. The newly merged bucket size nearly z_p, as we needed to merge buckets with increasing sizes.
- To merge two consecutive buckets we need size of bucket twice than size of 2 buckets going to merge. Timestamp of newly merged bucket is most recent timestamp from 2 consecutive buckets. By computing few parameters decision of cluster merging is taken.
- Let, k-means Euclidean. A cluster represent with number of points (n) and centriod (c).

  Put p = k, or larger – k-means clustering while creating bucket

  To merge, $n = n_1 + n_2$, $c = \frac{n_1 c_1 + n_2 c_2}{n_1 + n_2}$
- Let, a non Euclidean, a cluster represented using clusteroid and CSD. To choose new clusteroid while merging, k-points furthest are selected from clusteroids.

  $CSD_m(P) = CSD_1(P) + N_2(d^2(P, c_1) + d_2(c_1, c_2)) + CSD_2(c_2)$

7.5 Answering Queries

- Given m, choose the smallest set of bucket such that it covers the most recent m points. At most 2m points.
- Bucket construction and solution generation are the two steps used for quarry rewriting in a shared – variable bucket algorithm, one of the efficient approaches for answering queries.

Review Questions

Q. 1 What is clustering algorithm ?

Q. 2 What is CURE ?

Q. 3 Write procedure of CURE cluster algorithm.

Q. 4 What is sampling ? Explain random sampling and partition sampling.

Q. 5 Write pseudo function of cluster algorithm.

Q. 6 Write procedure for merging cluster in CURE.

Q. 7 What is stream computing ?

Q. 8 What is stdin, stdout, stddir ?

Q. 9 Explain BDMO algorithm.

Q. 10 What is bucket, how it is used for clustering ?

Q. 11 Explain in brief initializing and merging of bucket.

□□□

CHAPTER 8

Module - 6

Link Analysis

Syllabus

PageRank Overview, Efficient computation of PageRank : PageRank Iteration Using MapReduce, Use of Combiners to Consolidate the Result Vector

8.1 Page Rank Definition

MU - May 17

Q. Explain Page Rank with Example. (May 17, 2 Marks)

Q. What is Page Rank ? Explain the Inverted Index.

- Google™ is one of the giants in Information Technology. The major product of the Google i.e. search engines, dominate the all other web services.
- Before Google's search engine there were many search engines available but the algorithm they exhibit is not up to the mark. These search engines were worked equivalent to a "web-crawler".
- 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.
- This listing of different terms will be stored inside the specialized data structure known as an **"inverted index".**
- An inverted index data structure has listing of different non-redundant terms and it issues an individual pointer to all available sources to which given term is related.
- Fig. 8.1.1 shows inverted index functionality.

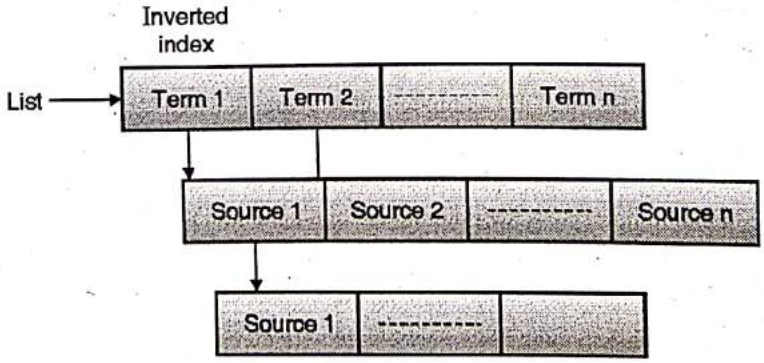


Fig. 8.1.1 : Inverted Index

- Every term from the inverted index will be extracted and analyzed for the usage of that term within the web page.

- Every term has some usage percentage within the given web page According to percentage of usage of terms in a web page it will be ranked. This will be achieved by firing a search query.

Example : If a given term 'x' is appeared in the Header of a web page then the page in which this term occurs proves to be much relevant rather than the term appeared in a paragraph text.

8.1.1 Importance of Page Ranks

Q. Explain Importance of Page Rank.

- In the Information Technology world, the storage and retrieval of the information becomes a crucial activity as Data generation from every sector increases exponentially.
- Every day if world will face new challenges for managing the different category web pages their arrangement by category, their ranking by search criteria etc.
- According to statistics in 1998 World Wide Web has around 150 million web pages and by today we have 1500 + million web pages.
- It is very much difficult to manage these huge number of web pages because every page contains following associated parameters :
  - (i) Number of terms involved
  - (ii) Category of web page
  - (iii) Topics involved in a given web page
  - (iv) Usage of involved topics by other web pages
  - (v) Quality of web pages etc.

Page Ranking : The term page Ranking can be defined as, "A classical method used to arrange the web pages according to its objective and the usage of terms involved in it on the world wide web by using any link data structure."

- The page Ranking mechanism was developed by Larry page and sergey Brin. This page-Ranking mechanism was a part of their research project which was started in 1995 and result into a functional prototype in 1998.
- After that, shortly they founded the Google.

8.1.2 Links in Page Ranking

MU - May 19

Q. List down the steps in modified Page Rank Algorithm to avoid spider trap with one example. **(May 19, 10 Marks)**

If we consider that, there are 150 million web pages exists in the some part of world wide web then all pages may have approximately 1.7 billion links to different web page.

Example

Suppose we have 3 pages A, B, C in a given domain of web sites. They have interconnection links between them as shown in Fig. 8.1.2

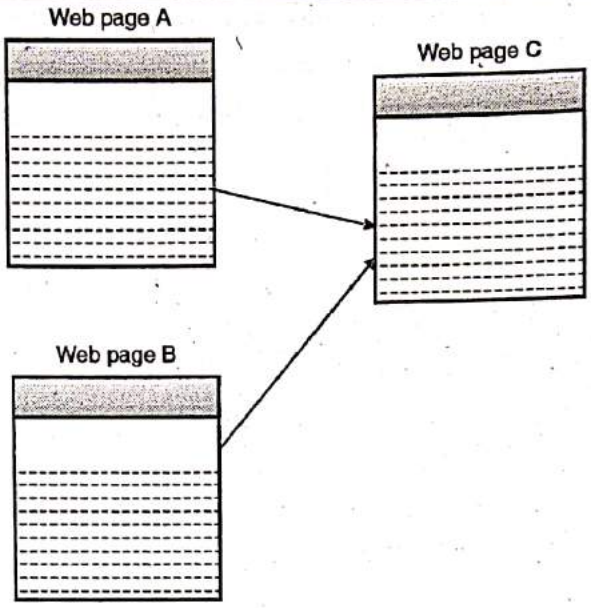


Fig. 8.1.2

The number of links exists between two or more web pages can be categorize as follows :

1. Back links
2. Forward links

1. Back links

With reference to Fig. 8.1.2 A and B are the Back links of web page 'C' i.e. Back link indicates given web page is referred by how many number of other web pages.

2. Forward link

- Forward link represents the fact that, how many web pages will be referred by a given web pages.
- Clearly, out of these two types of links back links are very important from Ranking of documents perspective.
- A web page which contains number of back links is said to be important web page and will get upper position in Ranking.
- A page Ranking in mathematical format can be represented as,

$$R(u) = c \sum_{V \in B_u} \frac{R(V)}{N_V}$$

Where,

μ : Represents the web page $B\mu$

N_V : It represents number of forward links of page v.

C : It represents the Normalizations factor to make

$$|R|_{L1} = 1 \; (|R|_{L1} = |R_1 + R_2 + \ldots R_n|)$$

- A world wide web can be considered as the 'Di-graph' i.e. Directed graph Any graph 'G' is composed of two fundamental components vertices and Edges.

G = (V, E)
- V → Vertices/nodes
- E → Edges

Here, vertices or Nodes can be mapped to pages.

- If we consider a small part of world wide web containing 4 web pages named as P_1, P_2, P_3, P_4.
- Every page i has Back links and forward links to other pages.
- Fig. 8.1.3 shows the above mentioned structure.

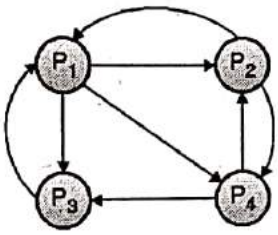


Fig. 8.1.3

- In Fig. 8.1.3 page P_1 has Forward links to page P_2, P_3 and P_4 respectively.
- Page P_2 has links to page 1 and page P_3.
- Page 3 has link to page 1 and
- Page 4 has links to page 2 and page 3.
- If a user starts surfing with page P_1 in above web page P_1 has links to page P_2, P_3 and P_4

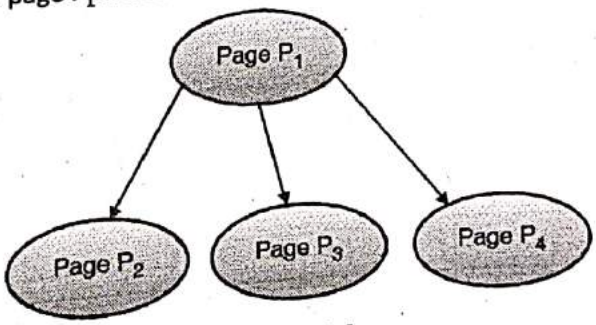


Fig. 8.1.4

- Probability that user will be at page P_2 / P_3 / P_4 is equal to 1/3.
- Probability that user will be at page 1 itself is '0'.
- Suppose user has chosen page P_2 then
- Probability that user will be at page P_1 is 1/2,
- Probability that user will be at page P_4 is 1/2.
- Probability that user will be at page P_2 or P_3 is 'D'.

These possibilities of web surfing by a given user can be represented using special structure known as "Transition Matrix".

In general, the transition Matrix 'M' is composed of 'n' pages 'n' rows and 'n' columns. Two pointer c and j will be to represent the current row and columns respectively.

Any given element can be represent as m_{ij}.

m_{ij} = 1/k if and only if the page at j^{th} column has k forward links.

- Additionally one of the forward links to same page itself.

∴ The transition Matrix for above web can be represented as,

$$M = \begin{array}{c} \\ A \\ B \\ C \\ D \end{array} \begin{array}{c} \begin{array}{cccc} A & B & C & D \end{array} \\ \begin{bmatrix} 0 & 1/2 & 1 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix} \end{array}$$

Matrix should be seen column wise Example 2

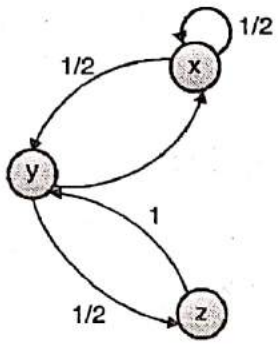


Fig. 8.1.5

∴ The transition Matrix for above graph can be represented as,

$$M = \begin{array}{c} \\ X \\ Y \\ Z \end{array} \begin{array}{c} \begin{array}{ccc} X & Y & Z \end{array} \\ \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 1 \\ 0 & 1/2 & 0 \end{bmatrix} \end{array}$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$$

$$\begin{bmatrix} 1/3 \\ 1/2 \\ 1/6 \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 1 \\ 0 & 1/2 & 0 \end{bmatrix} \begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix}$$ ...For first iteration

$$\begin{bmatrix} 5/12 \\ 1/3 \\ 1/4 \end{bmatrix} = \begin{bmatrix} 1/2 & 1/2 & 0 \\ 1/2 & 0 & 1 \\ 0 & 1/2 & 0 \end{bmatrix} \begin{bmatrix} 1/3 \\ 1/2 \\ 1/6 \end{bmatrix}$$ ...For second iteration

- Hence, with simplified page Rank algorithm a critical problem has evolved i.e. during each iteration, the loop accumulates the rank but never distributes rank to other pages.
- To identify the location at which the user will in near future one must have a probability with a specialized function known as "A page Rank".
- All Transition Matrixes will work on column vector or j^{th} component.
- Consider a user "xyz" want to surf the web consists of 'n' web pages. Every page in a web has equal probability that user will visit that page at next instance.

- Consider a vector V_0 as an initial vector component. The probability that the user will be at n^{th} page will be 1/n with same initial vector V_0.
- At next instance, user will at one of 'n' available pages. The probability distribution of this situation can be represented as, MV_0 on next instance it will be $M(M.V_0)$ and process continues.
- The probability that a user will be present at i^{th} node on given instance is equal to elements position into vectors value.

$$\text{Probability } (x_i) = \sum m_{ij} : V_j$$

Where,

(i) m_{ij} represents the probability of user movement at given instance from j^{th} location to i^{th} location.

(ii) V_j represents the probabilities that user is at j^{th} position for previous instance.

- Traditionally, this process is known as "Marker Principle"
- To have Markov distribution a system of graph should satisfy following constraints.

(i) Graph under consideration should be "strongly connected" i.e.

Every node is accessible other available nodes.

(ii) There should not be any dead ends.

8.1.3 Structure of the Web

Q. Explain Structure of Web? Explain Spider trap in detail.

- The web is nothing but the composition of number of individual independent nodes. The nodes can be also termed as workstations. We can imagine the web as set of different distributed systems together.
- In every distributed systems we have clusters, cluster means a group of similar objects for the clustering in the context of distributed systems, the term object can be replaced with node i.e. A cluster in a given distributed system is the collection of similar nodes.
- Additionally the term similarity can be determined by analyzing the concept such as :

(i) Nodes having same hardware configuration

(ii) Nodes having same operating system and other system and application software configuration.

- It is always recommended that, all the nodes in web should always be connected. This can be achieved theoretically but practically this is not the case always.
- Fig. 8.1.6 shows the part of World Wide Web where, every node is connected with other nodes.

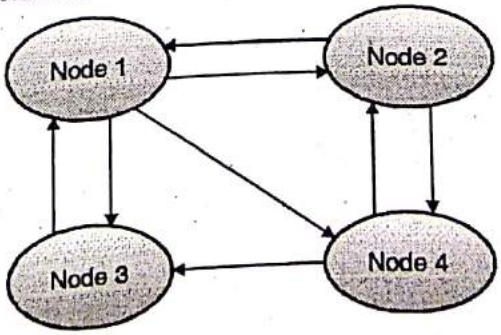


Fig. 8.1.6

– In practice any given web structure is composed of 4 types of components :

1. Strongly Connected Components (SCC)
2. In - components
3. Out components
4. Disconnected components

1. **A strongly connected components :** Is nothing but the components which are directly connected to each other for the data exchange and they also has forward and backward link to each other.

2. **In-components :** In-components are the integral part of where it exhibit the relation with SCC such that,

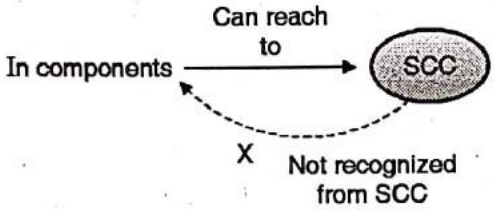


Fig. 8.1.7

3. **Out-components :** Out-components are the structures which shows following properties.

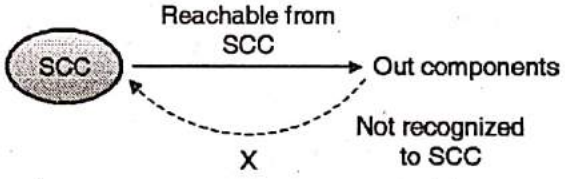


Fig. 8.1.8

The in-component and out components can have tendrils which represents in and out components.

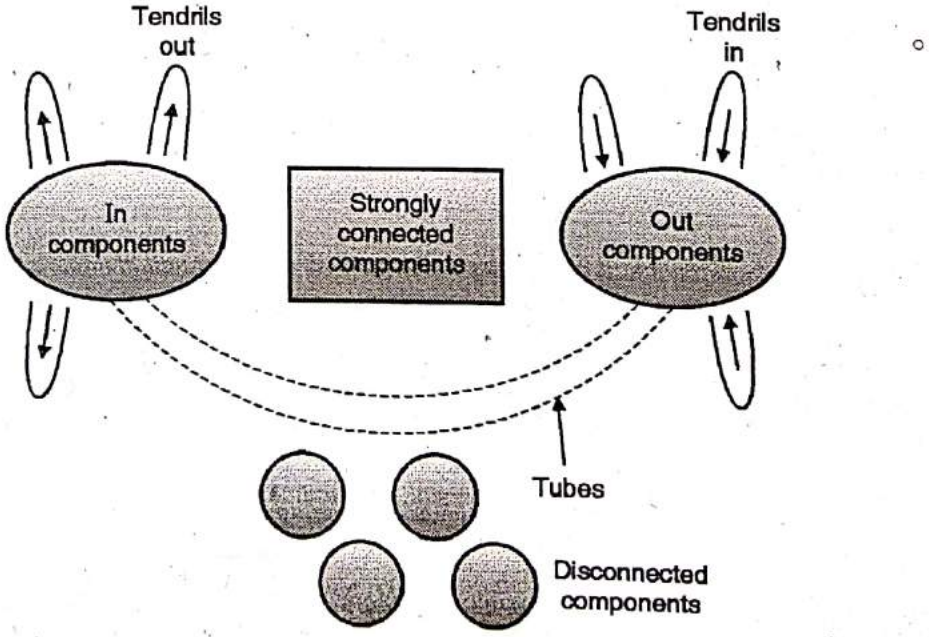


Fig. 8.1.9

In real time two major issues we will encountered :

(i) Dead end	(ii) Spider traps

(i) **Dead ends**

MU - Dec. 16

Q. Explain how dead ends are handled in Page Rank. (Dec. 16, 5 Marks)

- In a given part of web, if we encountered with a page which doesn't have links which are going out from that page or component.
- This will affect the transition matrix directly as, the column containing entry for dead end page will lead to sum = 0 instead of 1.
- The property of having sum = 1 for most of the columns in a given transition matrix is known as **'Stochasticity'** and if there are dead ends then some of the columns have '0' entries.
- Consider the Fig. 8.1.10.

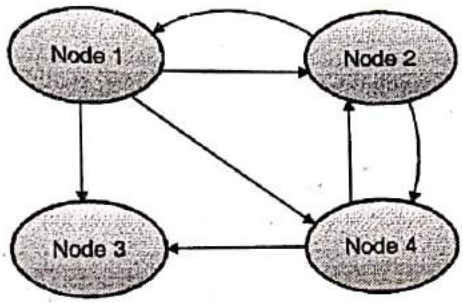


Fig. 8.1.10 : Node 3 is dead end

- By referring Fig. 8.1.10 we came to know that, Mode 3 is dead end and while searching if we encountered on the Node 3. i.e. at dead end then web surfing will struck at that page or node as there is not a single out link from Node 3. Hence, transition matrix for Fig. 8.1.10 is,

$$M = \begin{bmatrix} 0 & 1/2 & 0 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/2 & 1/2 & 0 & 0 \end{bmatrix}$$

Following are the ways to deal with dead ends :

1. **First approach to deal with dead ends we can delete that node by removing their incoming links.**
   - Disadvantage of this approach is it will introduce more dead ends which has to be solved with the same approach in recursive manner.
   - Though we delete the node but total page rank for a given graph to or web will be kept as it is the Nodes which are not available in graph G, but we can consider the set of other nodes which acts as predecessors for the calculation of page rank.
   - Additionally we can consider the successor nodes and have a division operation.
   - After this procedure, some nodes might be there which are not available in graph G but they are done with their predecessors calculations.
   - After some iterations all nodes has their page rank the order in which page ranks are calculated is exactly opposite to node deletion order.

- Suppose we have graph containing nodes and these nodes are arranged in following manner as shown in Fig. 8.1.11.

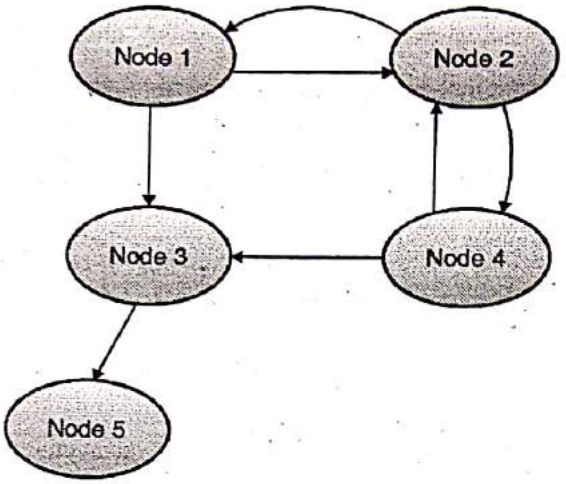


Fig. 8.1.11

- If we observe the Fig. 8.1.11 to calculate the page rank. We find that, Node 5 is the dead end as it doesn't have any forward links i.e. the links going out from Node 5.
- So hence, to avoid the dead ends, delete the Node 5 and its corresponding are coming from Node 3. So now the graph G becomes.

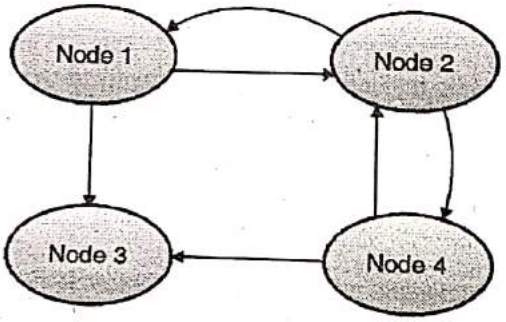


Fig. 8.1.12 : Graph a after deletion of node 5

- By observe the Fig. 8.1.12 we came to know that now 'node 3' is **'dead end'.**
- Now as we are avoiding the dead ends. Hence delete 'Node 3' and it is respective in coming edges. Now, Graph G becomes,

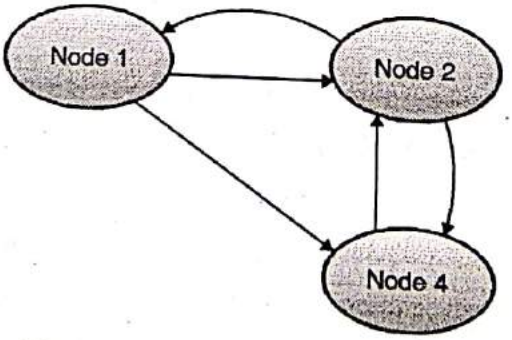


Fig. 8.1.13 : After deletion of node 3

- Now every node : Node 1, Node 2, and Node 3 has edges coming out i.e. forward links. Hence, is no dead end in graph 'G'.

The transition matrix for above graph will be,

$$M = \begin{bmatrix} 0 & 1/2 & 0 \\ 1/2 & 0 & 1 \\ 1/2 & 1/2 & 0 \end{bmatrix}$$

We can have component vector representation for above matrix as follows :

$$\begin{bmatrix} 1/3 \\ 1/3 \\ 1/3 \end{bmatrix} \text{ – (1) Iteration 1}$$

$$\begin{bmatrix} 1/6 \\ 3/6 \\ 2/6 \end{bmatrix} \text{ – (2) Iteration 2}$$

$$\begin{bmatrix} 3/12 \\ 5/12 \\ 4/12 \end{bmatrix} \text{ – (3) Iteration 3}$$

Final value for component vector will be, $\begin{bmatrix} 2/9 \\ 4/9 \\ 3/9 \end{bmatrix}$

Page rank for node 1 $= \frac{2}{9}$

Page rank for node 2 $= \frac{4}{9}$

Page rank for node 4 $= \frac{3}{9}$

- We have to calculate the page rank for Node 3 and Node 5 with the exact opposite order of node deletion. Here Node 1, Node 2, Node 4 are in the role of predecessors.
- Number of successor to Node 1 = 3, Hence, the contribution from Node 1 for calculating the page rank of Node 3 is 1/3
- For Node 5 it has 2 successors. Hence the contribution from Node 5 for calculating the page rank of node 3 is 1/2.

$$\text{Page rank of Node 3} = \left(\frac{1}{3} \times \frac{2}{9}\right) + \left(\frac{1}{2} \times \frac{3}{9}\right) = \frac{13}{54}$$

- For calculating the page rank of Node 5, Node 3 plays a crucial role. As Node 3 has, number of successors = 1 and node 1 has node 3 as its predecessor. Hence, we can conclude that Node 5 was page rank same as that of Node 3.
- As the aggregate of their page rank is greater than 1, so it doesn't indicate the distribution for a given user who is surfing through that web page. Still it highlights the importance of web page relatively.

2. **Another way to deal with dead ends is configure the process for a given user by having assumption that it is assumed to be moved through web known as "Taxation".**

- Taxation method points to other problem also which is known as "spider traps".

(ii) Spider traps

- Spider traps is nothing but set of web pages all of them containing the out links but they never going to link with any other page.

i.e. Spider Trap = Set of web pages with no dead ends but no edge going outside also (no forward link)

- Spider traps can be sowed in the web with or without intention. There can be multiple spider traps in real time in a given web page. Set but for demonstration purpose, consider Fig. 8.1.14 which shows the part of web containing only one spider trap.

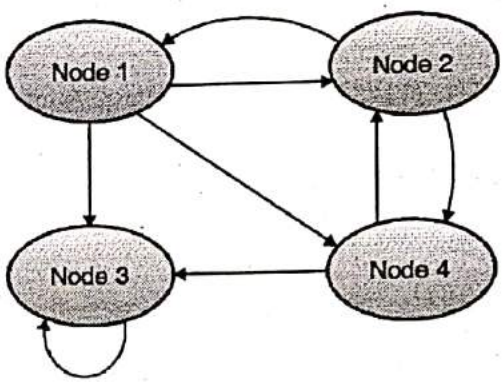


Fig. 8.1.14 : Graph with one spider - trap

The transition Matrix for the Fig. 8.1.14

$$M = \begin{bmatrix} 0 & 1/2 & 0 & 0 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 0 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{bmatrix}$$

- If we proceed further by the same method stated in previous section for calculating the page rank then the ultimate result that we get will be,

$$\begin{bmatrix} 1/4 \\ 1/4 \\ 1/4 \\ 1/4 \end{bmatrix} - (1) \text{ Iteration (1)}$$

$$\begin{bmatrix} 3/24 \\ 5/24 \\ 11/24 \\ 5/24 \end{bmatrix} - (2) \text{ Iteration (2)}$$

$$\begin{bmatrix} 5/48 \\ 7/48 \\ 29/48 \\ 7/48 \end{bmatrix} - (3) \text{ Iteration (3)}$$

$$\begin{bmatrix} 21/288 \\ 31/288 \\ 205/288 \\ 31/288 \end{bmatrix} - (4) \quad \text{Iteration (4)}$$

:

:

At last iteration we get,

$$\begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

- The highest page rank will be given to Node 3 as, there is no link which goes out from it, but if has the link to inside it. So, user is going to struck at Node 3. As it is represented that number of user are there at Node 3 so Node 3 has greater importance.
- To have a remedy to this problem we just configure the method of calculating the page rank by injecting a new concept known as 'teleporting' or more specifically a probability distribution of "teleporting" and we are not following the links going out from given node.
- To calculate the teleporting probability – calculate new component vector V_{new} for estimating page ranks such that,

$$V_{new} = \beta M \cdot V + (1 - \beta) \cdot e/n$$

Where,

β = Constant (Ranges from 0.8 to 0.9)

e = Aggregate vector of all vectors with value 1

n = Number of web pages/ nodes in a graph

- If we do not dead ends in the graph then,

Probability of introduction of New user to a given web = Probability of not to choose out-link for the current page by same user

- Another possibility is user will not be able to move to any page as $(1 - \beta)$ e/n term is independent of $\sum V$.

i.e. when we don't encountered with dead ends then,

$\sum V < 1$ but $\sum V = 0$.

8.1.4 Using Page Rank in a Search Engine

- The page rank calculation plays crucial role in deciding the overall efficiency of the search engine.
- A basic crawling is done to have page ranking to fetch the required information and the page.
- When a user submit some query or a request to the given search engine then at background a secret algorithm is triggered for the execution which fetches different web pages in some order which is based on a pre defined criteria.
- The user query generally in the form of single word or collection of words.

- For example, the most popular search engine Google has 250+ such predefined criteria to arrange the fetched web pages in some particular order.
- Every page on the web should possess minimum one word or one phrase in it. Same as that of user's search query.
- If the given web page doesn't contain any word or phrase then there is less probability that a page will have the highest page rank.
- In page ranking calculation, the place on the web page where the phrase is appeared is also matters e.g. (phrase appears in the header will have more importance than in footer, the phrase appeared in paragraph will have average importance)

8.2 Efficient Computation of Page Rank

Q. What is thrashing? How it affect the Page ranking Mechanism?

- In previous sections we have studied that, how to calculate the page rank of the given web page in a given web structure.
- The efficiency in such complex calculation is achieved as we have taken a small part of web i.e. for 4-5 nodes or pages only.
- But, if scale-out this small for concept to a real-time condition billions of web pages V, a matrix – vector multiplication we have to compute of order atleast 70-80 times till a component vector will stop changing its value.
- For such real time complexity the solution proposed is use of Mapreduce technique studied in Section 3.2, but such usage is not that straight forward, it has two handles to cross.
  (i) The most important parameter that how to represent the transition matrix for such huge number of web pages. If we try to represent the matrix for all available web pages which are under consideration then it is absolutely in efficient for performing the calculations. One way to handle this situation is to indicate the non-zero elements only.
  (ii) One more thing is if we go for an alternative to mapreduce functionality for performance and efficiency concerns then we may think for 'combiners' explained in Section 3.2.4.
- The combiners generally used to minimize the data more specifically an intermediate data result to be transferred to reducer task.

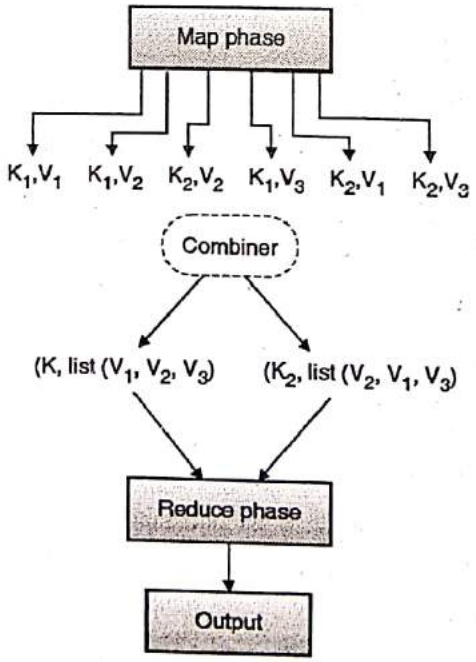


Fig. 8.2.1 : Use of combiners

- Also, the striping concept doesn't have much more impact to reduce the effect of "thrashing".

Thrashing

- All computations performed by the CPU itself irrespective of environment used i.e. Distributed Computing Environment (DCE) or standalone computing. Hence, the main task of processor / CPU is to execute the instructions and not the featuring of data from the secondary storage.
- If in some commonly occurred situation processor /CPU is busy in just fetching the data from secondary storage rather than executing the instruction then such situation is known as "Thrashing".

8.2.1 Representation of Transition Matrix

- As we know that, number of web pages that we are going to deal with are billions and number of links going out from a given web pages are 10 on an average.
- Entry '1' in billion pages is not zero. The best way to indicate the transition matrix is to have a list of different web page which has entries 'non-zero' with associated values.

The structure will look like,

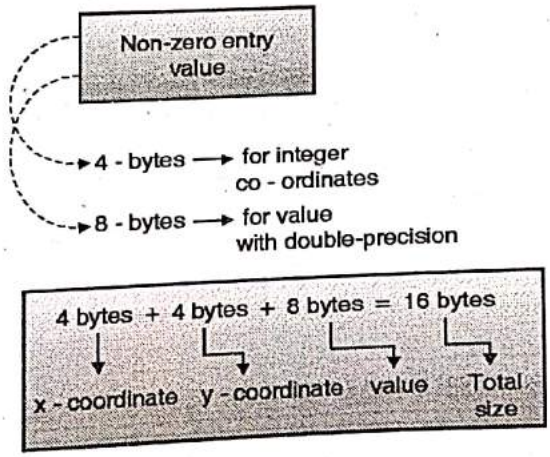


Fig. 8.2.2

- So, space required here has linear nature instead of quadratic.
- We can apply more compression by column wise representation of non zero entries i.e. 1/number of links going out from a given web page

A column is represented as ,

1 integer → to represent → out degree

1 integer → to represent → for every non-zero entry in that column

↓

yields a row number of entry location

8.2.2 Iterating Page Rank with MapReduce

Q. Explain Iterating Page Rank Process with MapReduce? Also comment on use of Combiners.

- A single pass of page rank calculation includes calculation two component vectors depicted by V and V_{new}.

$$V_{new} = \beta \cdot M \cdot V + (1 - \beta) \cdot e/n$$

Where, β = Constant (Ranges between 0.8 to 0.9)

e = components vector of entries 1

n = Number of web pages

M = Transition matrix

When 'n' has small value then V and V_{new} can be stored in primary memory or main memory for Map task.

- If in real – time V is big in size so that if can't be fit into main memory then we can go for striping method.

8.2.3 Use of Combiners to Aggregate the Result Vector

- The page ranking iteration with MapReduce task is not proven to be sufficient because
  - (i) If we want to add different terms V_{new} i.e. the i^{th} element of new resultant vector V, computed at Map phase

    This is equivalent to the usage of special structure "combiner" which actually combines the different values according to their key as shown in the Fig 8.2.3.
  - (ii) If we are not going to use MapReduce at all.
- Hence depending on the requirement situation and complexity of problem a method to be used should be decided.

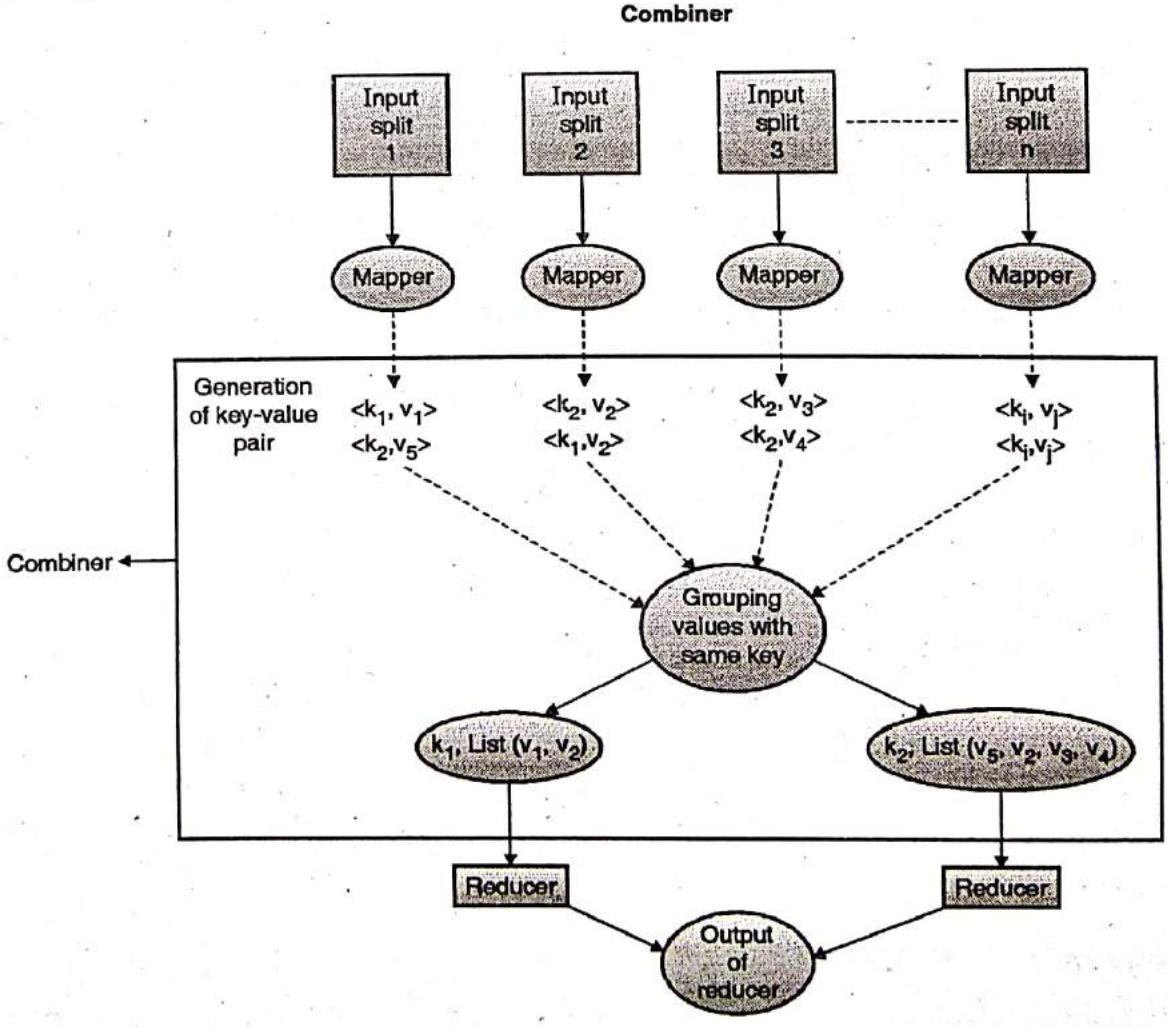


Fig. 8.2.3 : Combiner working mechanism

8.3 Link Spam

Q. What is Link Spam? Explain in Detail.

- When page rank question arises then the giants in information technology such as google will develop some solution to it. Additionally there are some security related issues such as "The spams".
- But we do have destructive minded people in society who will always try to affect the system by performing some malicious activities. Hence for page ranking calculations "spammers" are came into existence.
- Spammers have introduce the tools and techniques through which for any given page its page rank can be increase by a selected multiple such intentional rise in the value of page rank is known as a "link spam".
- For link spam spammers introduce the web pages itself for link spamming.

8.3.1 Spam Farm Architecture

Q. Explain Spam Farm Architecture in detail.

- The malicious web pages introduced by the spammers is known as spam farm. Fig. 8.3.1 shows the basic architecture of spam farm.

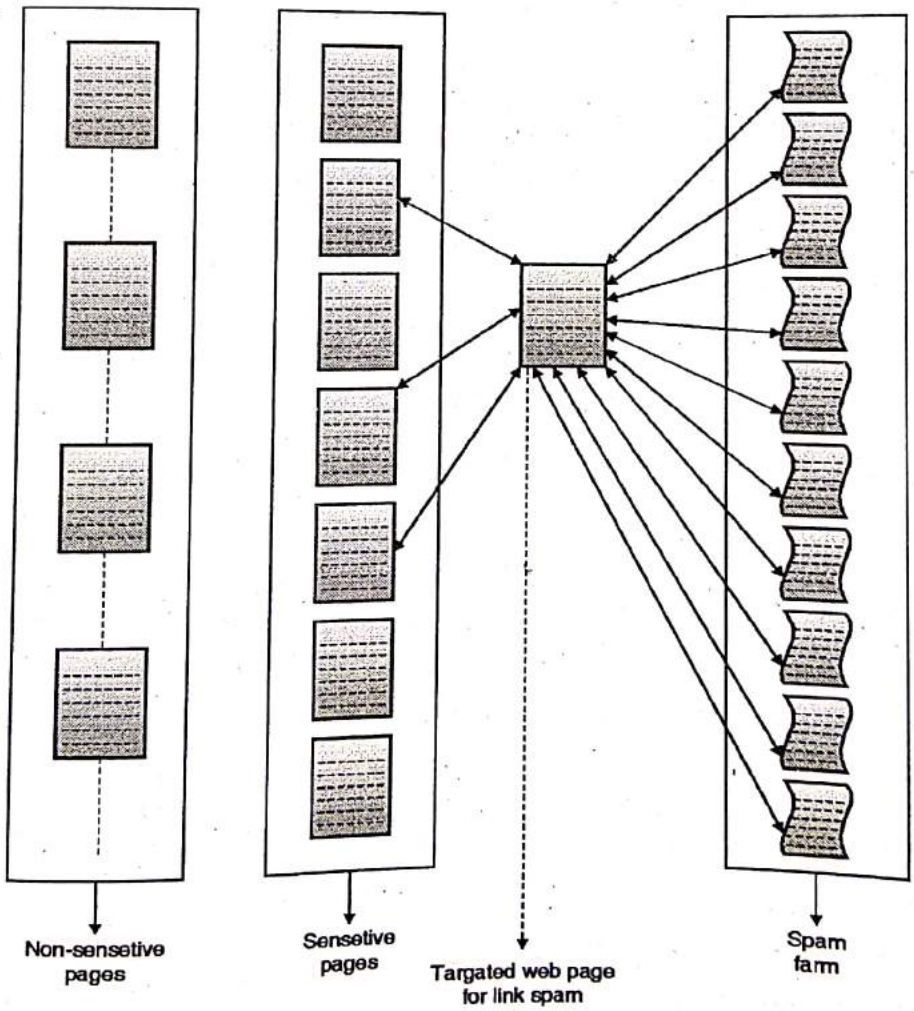


Fig. 8.3.1 : Basic architecture of spam farm

- If we consider the spammers perspective then Fig. 8.3.1 can be divided into 3 basic blocks

1.	Non-sensitive	2.	Sensitive
3.	Spam farm		

1. Non-sensitive

- These are the pages which are generally not accessible to the spammer for any spamming activity. As these pages are not accessible to spammers so, they will not affect by any activity performed by the spammer.
- Most of web pages in a given web structure will fall in this category.

2. Sensitive

- These are the web pages which are generally accessible to the spammers for any spamming related activity. As these pages are accessible to the spammers so, they will get affected easily by spamming activity performed by the spammer.
- The effect of spamming on these pages is generally indirect as these pages are not manipulated by the spammers directly.

3. Spam farm

The spam farm is the collection of malicious web pages which are used to increase the number of links pointed to and coming out from a given web page so, ultimately the page rank of a given page i.e. target web page will increase dramatically. There are other category web pages which supports the spamming activity by aggregating page ranking i.e. a part of term (1 - β).

8.3.2 Spam Farm Analysis

Q. Explain the Spam Farm Analysis in detail.

- In spamming activity we do page ranking calculation by considering sensitive, non sensitive and actual malicious spam farm pages.
- The basic page ranking calculation is done and by alternate method 'β' term is calculated which is also known as Taxation method.
- The term 'β' will depicts the fact that, how a part of page rank is segregated among the successor nodes for the next iteration. Actually β is the constant term ranges between 0.8 to 0.9 generally (0.85).
- We know that, there are some web pages who supports the spamming activity.
- So a page rank of one of such supporting page can be calculated with the help of following formula

$$Pr(s) = \beta \cdot Y / m + (1 - \beta) / n$$

Where,

Pr (s) → It represent the page rank of randomly selected supporting web page.

β → It is the constant range from 0.8 to 0.9.

Y → It represent page rank of target web page say 't'.

m → It represent number of web pages supporting spamming activity.

n → it represent total number of web pages in a given web structure.

The term β, Y is totally related to the target web page 't' calculated page rank will get segregated on all m's.

Hence a page rank represented by 'Y' for a given target web page 't' will composed of 3 incoming types :

(i) The number of links which are pointed to the target web page from the non-sensitive web pages represented by variable 'x'.

(ii) Page rank of supporting page with β multiples.

$\beta \cdot (Pr(s)$

Where, $Pr(d) = \beta \cdot Y / m\,t\,(1-\beta)/n$

∴ We can conclude that the page rank 'Y' of target web page 't' will be in the form of

$$Y = x + \beta \cdot m\left(\frac{\beta y}{m} + \frac{1-\beta}{n}\right)$$

$$= x + \beta^2 y + \beta \cdot (1-\beta) \times \frac{m}{n}$$

Here we can introduce a constant Q,

$$\therefore Y = \frac{x}{1-\beta^2} + Q \cdot \frac{m}{n}$$

$$Q = \frac{\beta(1-\beta)}{(1-\beta)^2}$$

$$= \frac{\beta}{(1+\beta)}$$

8.3.3 Dealing with Link Spam

- As we have seen the effect of link spam on the fundamental primary things related to page rank system.
- Link spam will disturb the page rank system completely hence to deal with the link spamming the different search engine thought of different solution, which will help in minimizing the effect of link spam.
- Basically there are two ways to deal with link spamming they are as follows :

(i) A traditional approach where the search engine algorithm will have top-view of whole scenario and eventually algorithm win find such link spams and remove them from the indexing structure.

But as soon as algorithm deletes the spammer web page the spammer will find the alternate way to do the link spamming.

(ii) A modern way to deal with the situation is, modify the procedure for calculation of page rank with reference to below grade link spams.

We will have two alternate procedures to do so :

(i) Trust ranking
(ii) Spam mass

(i) Trust ranking

- In trust ranking the system is going to trust on some of the web pages by assuming that those web pages are not the part of spam farm.

- Such set of web pages is termed as "topic".
- Consider a spam farm page want to increase a page rank of trusted web page. So, spam page can have a link to trusted page but that trusted page will not established a link to spam page.

(ii) Spam mass

- In spam as technique, the algorithm of page ranking will calculate the page rank for every web page also the part of the page rank (affected part) whose contributor is spam page will be analysed. This analysis is done with the help of comparison between normal page rank and page ranking obtained through trust ranking mechanism.
- This comparison can be achieved through following formula :

$$Pr\ (Sm) = \frac{Pr - Prtr}{Pr}$$

Where, $P_r(s_m)$ = page ranking by spam mass technique

P_r = page ranking by traditional method

P_rt_r = page ranking by trust ranking method

If $P_r(s_m) < 0$ i.e. negative

Or

$Pr\ (s_m) > 0$ but < 1 i.e. not close to 1 then that page is not a spam page else it is a spam page.

8.4 Hubs and Authorities

MU - May 17

Q. Explain Hubs and Authorities with neat diagram. (May 17, 5 Marks)

Q. What is Hubs and Authorities? Explain its significance.

- The hubs and authorities is an extension to the concept of page raking. Hubs and authorities will add more preciseness to the existing page rank mechanism.
- The ordinary, traditional page rank algorithm will calculate the page rank for all the web pages available in a given web structure. But user doesn't want to examine or view all of these web pages. He/she just want first 20 to 50 pages in an average case.
- Hence, the idea of hubs-and authorities will came into existence to have efficiency and reduce work load calculating page rank.
- In hubs and authorities page rank will be calculated for only those web pages who will fetch in resultant set of web pages for a given search query.
- It is also known as, hyperlink induced topic search abbreviated as HITS.
- The traditional page rank calculations have a single view for a given web page. But hubs and authorities algorithm will have two different shades of views for a given web page.

1. Some web page has importance as they will present signification information of given topic so these web pages are known as the authorities.
2. Some web pages has importance because they gives us the information of any randomly selected topic as well as they will direct us to other web pages to collect more information about the same. Such web pages known as hubs.

8.4.1 Formalizing Hubs and Authority

- As stated in earlier section hubs and authorities these are the two shades with which a web page can be viewed.
- So, we can allot 2 types of scores for a given web page.

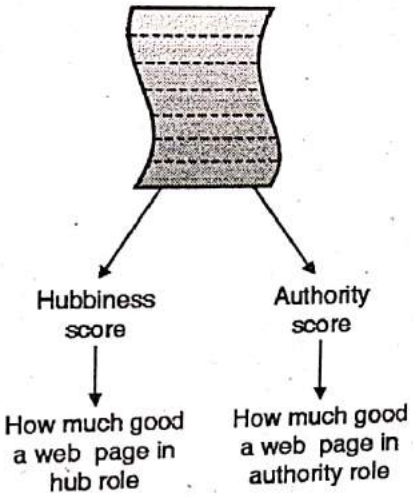


Fig. 8.4.1

h → represents hubbiness score

a → represents authority score

- j^{th} component of a web pages 'h' will give measure of Hubbiness of j^{th} page.
- j^{th} component of a web page 'a' will give measure of authority of j^{th} page.
- To have the notion of 'h' and 'a' consider link matrix 'LM' for web pages in a given web.
- Any element of LM can be represent as LMij

$\therefore$ LMij = 1 if a link is established from i^{th} page to j^{th} page.

LMij = 0 otherwise

The transpose of LM will change the result :

$\therefore$ LM_{ij}^{T} represent transpose of LMij

$\therefore$ LM_{ij}^{T} = 1 if a link is established from j^{th} page to i^{th} page

$\therefore$ LM_{ij}^{T} = 0 otherwise

- More that $LM^{T} = M$ where is a original transition matrix which maintain the record of No. of incoming and outgoing links.
- The difference between LM^{T} and M is,

$LM^{T} = 1$

and $M = \frac{1}{\text{No. of links going outside}}$ for that web page forgiven column.

Ex. 8.4.1 : Let the adjacency matrix for a graph of four vertices (n1 to n4) be as follows :

$$A = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Calculate the authority and hub scores for this graph using the HITS algorithm with k = 6, and identify the best authority and hub nodes.

MU - May 16. 10 Marks

Soln. :

$$\text{Given Matrix is} - \quad A = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Apply Transpose operation,

$$A^T = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

Now, Consider initial Hub score as 1

$$\begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} 1 \\ 1 \\ 2 \\ 4 \end{bmatrix} = \begin{bmatrix} \frac{1}{\sqrt{(1^2+1^2+2^2+4^2)}} \\ \frac{1}{\sqrt{(1^2+1^2+2^2+4^2)}} \\ \frac{2}{\sqrt{(1^2+1^2+2^2+4^2)}} \\ \frac{4}{\sqrt{(1^2+1^2+2^2+4^2)}} \end{bmatrix}$$

$$= \begin{bmatrix} 0.2132 \\ 0.2132 \\ 0.4264 \\ 0.8528 \end{bmatrix}$$

Iterate over K.

Review Questions

Q. 1 What is Page Rank ? Explain the Inverted Index ?

Q. 2 What is Page Rank? Explain Importance of Page Rank?

Q. 3 What are Links in Page Ranking? Explain in Detail.

Q. 4 What are links in Page Ranking? Explain Back Links and Forward Links with suitable example?

Q. 5 Explain the Structure of Web in the context of Link Analysis?

Q. 6 Explain Structure of Web? What is the significance of In-Components and Out Components?

Q.7 Discuss in detail Structure of web? Explain the Dead ends

Q.8 Explain Structure of Web? Explain Spider trap in detail.

Q.9 Explain the role of Page ranking in search engine?

Q.10 Explain the different modification suggested in efficient computation of Web pages.

Q.11 What is thrashing? How it affect the Page ranking Mechanism?

Q.12 Explain Iterating Page Rank Process with MapReduce? Also comment on use of Combiners.

Q.13 What is Link Spam? Explain in Detail.

Q.14 Explain Spam Farm Architecture in detail.

Q.15 What is Spam Farm explain with neat diagram? Also comment on Non-Sensetive, sensitive and spam farm.

Q.16 Explain the Spam Farm Analysis in detail.

Q.17 What is Link Spam? How to deal with Link Spam with Trust Ranking and Spam Mass.

Q.18 What is Hubs and Authorities? Explain its Significance.

□□□

CHAPTER 9

Module - 6

Recommendation Systems

Syllabus

A Model for Recommendation Systems, Content-Based Recommendations, Collaborative Filtering

9.1 Recommendation System

MU - Dec. 17

Q. What are recommendation systems? Clearly explain two applications for recommendation system. **(Dec. 17, 10 Marks)**

Q. What is recommendation system?

- It is vast widely used now-a-days. It is likely a subclass of information filtering system. It is used to give recommendations for books, games, news, movies, music, research articles, social tags etc.
- It is also useful for experts, financial services, life insurance, and social media like Twitter etc.
- Collaborative filtering and content-based filtering are the two approach used by recommendation system.
- Collaborative filtering uses user's past behaviour and apply some predication about user may like and accordingly post data.
- Content based filtering uses user's similar properties of data preferred by user.
- By using collaborative filtering and content based filtering a combine approach is developed i.e. Hybrid recommendation system.

9.1.1 The Utility Matrix

Q. Explain utility matrix with example.

- A recommendation system prefers the preference of a utility matrix. Users and item's these are entities used by recommendation system.
- Users have preference to data and these preferences must be observed.
- Every data itself is part of utility matrix as it belongs to some item category.

Example : A table representing users rating of apps on a scale 1 to 5, with 5 as highest rating Blank represents that user not replied on scale A1, A2 and A3 for Android 1, 2 and 3 i1, i2, i3 for iOS 1, 2, 3 users A, B and C gives rating.

	A1	A2	A3	i1	i2	i3
A	3		4		5	4
B		5				
C	3		4	4		4

Fig. 9.1.1 : A utility matrix representing ratings of apps on a scale 1to 5

- A typical users rating are very minute fraction of real scenario if we consider actual number of application of Android and iOS platform and number of users.
- It is observed in table for some apps there is less number of responses.
- The goal behind utility matrix is to make some predictions for blank spaces, these prediction are useful in recommendation system.
- As 'A' user gives rating 5 to i2 app so we have to take in account parameters of app i2 like its GUI, memory consumption, usability, music/effects if applicable etc.
- Similarly 'B' user gives rating 5 to A2 app so we have to take similar parameter in consideration. By judging both apps i2, A2 features and all we can put predication what can be further recommended to user A and B.
- From user "c" response though there is no use of full rating anywhere still it can be judge and predicted what kind feature based app user 'c' should be recommended.

9.1.2 Applications of Recommendation Systems

- Amazon. Com
- CDNOW. Com
- Quikr.com
- olx.com
- Drugstore.com
- eBay.com
- Moviefinder.com
- Reel.com and so many online good seller/buyer, trading website uses recommendation system.
- Product recommendation, Movie Recommendation, News Articles etc. are likely to be consolidate in a single place applications.

9.1.3 Taxonomy for Application Recommendation System

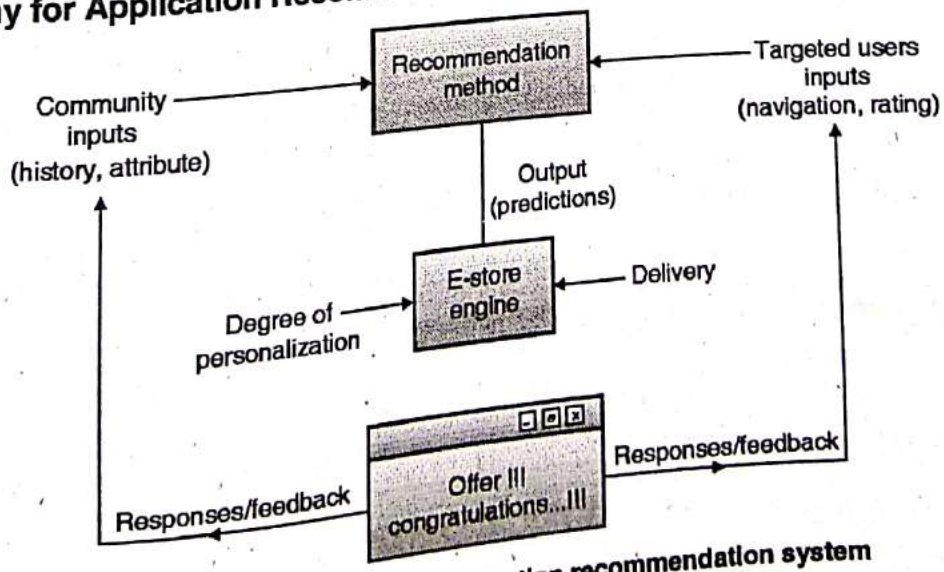


Fig. 9.1.2 : A taxonomy for application recommendation system

9.2 Content Based Recommendation

MU - May 19

Q. How recommendation is done based on properties of product? Explain with suitable example. (May 19, 10 Marks)

Q. Explain item profile of content based recommendation.

It focuses on items and user profiles in form of weighted lists. Profile are helpful to discover properties of items.

9.2.1 Item Profile

- An actor of drama or of movie is considered as an actor set, few viewers prefer drama or movie by their favourite actor(s).
- A set of teachers, some students prefer to be guided by few teacher(s) only.
- The year in which songs album release or made. Few viewers prefer old songs; some prefer to latest songs only, users sorting of songs based on year.
- So many classes are available which provides some data.
- Few domains has common feature for example a college and movie it has students, professors set and actors, directors set respectively. Certain ratio is maintained as many student and few professors in quantity while many actor works under one or two director guidance. Again 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.
- Music (song album) and a book has same value feature like songs writer/poet, year of release and author, publication year respectively.

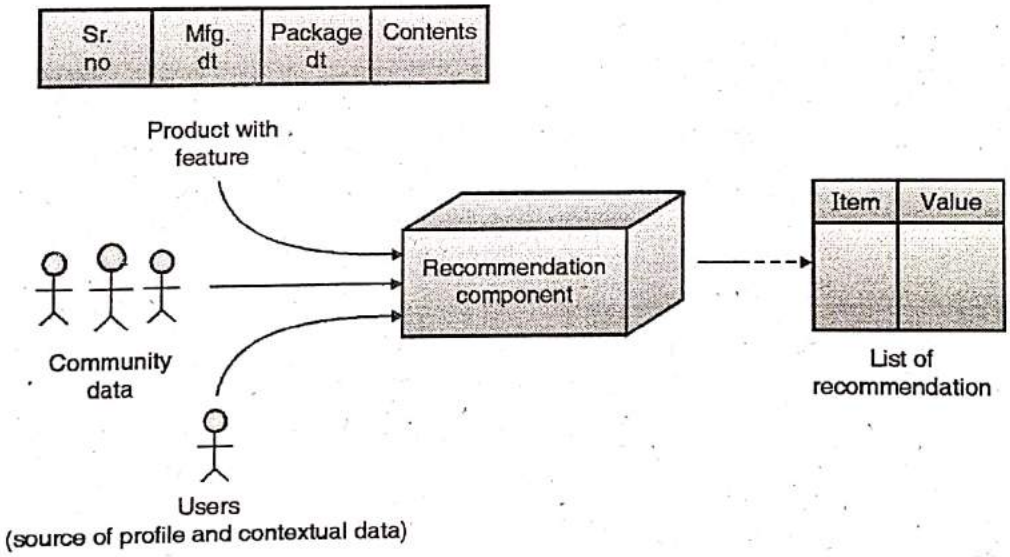


Fig. 9.2.1 : Recommendation system parameters

9.2.2 Discovering Features of Documents

MU - May 17

Q. How would you get the features of the document in a content -based system? (May 17, 3 Marks)

- Document collection and images are the two classes of items.
- We need to extract features from documents and images.

- There many kinds of document. Let say news articles in a newspaper. There are many articles in a newspaper but user reads very few of them. A recommendation system suggest for articles to a user supposed to be interested to read.
- Similarly there are so many websites and web pages, blogs could be recommended to a group of interested users.
- It will be more friendly to users if we can classify blogs accordingly to the topics.
- Unfortunately, document classes cannot provide available information features.
- A substitute is used for word identification on which characterize topic of document.
- We need to sort document by removing repeatedly used common words i.e. elimination of stop words. The words remain after elimination of stop words are proceed further to count their TF i.e. term frequency.
- IDF i.e. inverse Document frequency of each word in a document is observed and calculated.
- The high scoring words are the group of character which characterize document.
- Let, n words found in document with highest TF and IDF values of count. Then those n words are fixed to all similar documents.
- Words whose TF, IDF values of count founds threshold becomes a part of feature set.
- Such few words set represent a document. To know similarity between any two documents, there distance need to be measured between sets it can be done by either

(a) Jaccard Distance (b) Cosine Distance

9.2.3 Obtaining Item Features from Tags

- In case of published book database many features are available like Title, ISBN, Edition, Printing, Price, Compositor, Editor, Copyright etc., but user (reader) of book mainly concern with title and price tag mostly.
- We can get number of features of same items by tag items by entering phrase or search value range for price.
- By keeping tag option available, users can search at item on feature of tag value like price range for book, colour shade for cloth etc in online shopping.
- One problem in tag is to create tags and such enough tag awareness at user level.

9.2.4 Representing Item Profile

Q. How item profile is represented ?

- Some features are numerical for instance we might take the rating for mobile application based on Android OS platform to be a feature.
- This rating is a real number like 1, 2, 3, 4 and 5 stars by some users of it.
- By keeping options of only one component like 1 star or 5 stars will not make a sense to get possible average rating of
- By keeping 5 options in rating a good possible average for rating can be observed without lose of structure implicit in an application.
- By keeping 5 options in rating a good possible average for rating can be observed without lose of structure implicit in numbers.
- Numerical value based rating provides a single component of vector representing to items.
- Boolean value and other integer valued or real-valued parameter becomes components of vectors. Numerical features tell about similarity of items.

9.2.5 User Profiles

Q. What is a user profile in content based recommendation?

- Vectors are useful to describe items and user's preferences. Users and items relation can be plotted with the help of utility matrix.

  Example : Consider similar case like before but utility matrix has some nonblank entries that are rating in 1-5 range. Consider, user U gives responses with average rate of 3 there are three applications (Android OS based games) got ratings of 3, 4 and 5. Then user profile of U, the component for application will have value i.e. rated average of 3-3, 4-3 and 5-3 i.e. value of 1

- On other hand, user v gives average rating 4. So user v responses to application are 3, 5 and 2.
- The user profile for v has in the component for application, the average of 3-4, 5-4 and 2-4, i.e. value – 2/3.

9.2.6 Recommending Items to Users based on Content

- Between user's vector and item's vector cosine distance can be computed with help of profile vectors for users and items both.
- It is helpful to estimate degree to which user will prefer as an item (i.e. prediction for recommendation).
- If user's and response (like 1 to 5 scale for mobile apps) vectors cosine angle is large positive fraction. It means angle is close to 0 and hence there is very small consine distance between vectors.
- If user's and responses vector cosine angle is large negative fraction. It means angle is close to degree of 180 which is a maximum possible cosine distance.
- Cosine similarity function = for measuring cosine angle between two vector

$$\text{Cos } \theta = \frac{V_1 . V_2}{\| V_1 \| V_2 \|}$$

- In vector space model

$$V_d = [W_1 d, W_2 d, \ldots WNd]^T$$

Where,

$W_{t,d}$: TF* IDF weight of term 't' in 'd' document

TF : Term Frequency

IDF : inverse document frequency

9.2.7 Classification Algorithm

- It is used to know user's interest. By applying some function to new item we get some probability which user may like.
- Numeric values also help to know about degree of interest with some particular item.
- Few of techniques are listed as follows :

  (1) Decision Tree and Rule Induction

  (2) Nearest Neighbour Method

(3) Euclidean Distance Metric

(4) Cosine Similarity Function

Some other classification algorithm are :

(1) Relevance feedback and Rocchio's algorithm

(2) Linear classification

(3) Probabilistic methods

(4) Naive Bayes.

9.3 Collaborative Filtering

MU - Dec. 16, May 17

Q. Explain with example collaborative and content based filtering in a recommendation system. (Dec. 16, 10 Marks)

Q. Explain Collaborative Filtering based recommendation System. How it is different from content based recommendation systems ? (May 17, 10 Marks)

- Recommendation system in collaborative filtering becoming interesting as few domains are used move by research scholar and academician like human-computer interaction, information retrieval system and machine learning.
- Few famous recommender systems in some popular fields like Ringo-music, Bellcore-video recommender (movies), Jester-jokes etc.
- Collaborative filtering began to use in the early 1990s. Most widely used example of collaborative filtering and recommendation system is Amazon. com.
- To recommend among large set of values to users is very important. Recommendation must be get appreciated by user else effort taken for it were worthless.
- Netflix has 17,000 movies collection while Amozon.com has 4,10,000 title in its collection, so got proper selection of recommendation is necessary.
- Toolbox used for collaborative filtering becomes advanced with help of Bayesian interface, case-based reasoning method, information retrieval.
- Collaborating filtering deals with 'users' and items. A preference given by any user to an item is known as 'rating' and is represented by triplet value set of (User, Item, and Rating).
- Rating triplet of (users, items, rating) is used to create a sparx matrix and it is referred as rating matrix.
- 'Predict task' and 'recommend task' are used for evaluation and use of recommendation system.

Table 9.3.1 : Sample rating matrix con 5 star scales to apps

Apps(items) / Users	Whatsapp	Hangout	Telegram	Viber
User A	4	2	3	3
User B	3	3	5	3
User C	3	2	4	2

- Predict task tells about preference may given by a user or what user's likely preference to an item?
- Recommend task helpful to design n-items list for user's need. These n-items are not on basis of prediction preference because criteria to create recommendation may be different.

9.3.1 Measuring Similarity

Q. What is measuring similarity in collaborative filtering?

- Among values of utility matrix it is really a big question to measure similarity of items of users.

Table 9.3.2 : Utility matrix

Users \ Apps(items)	Whatsapp	Hangout	Telegram	Viber	Skype	Hike
User A	4			5	1	
User B	5	5	4			5
User C				2	4	
User D		3				

- Above utility matrix data is quite insufficient to put reliable conclusion. By considering values from A and C, they rated two apps in common but their ratings are diametrically very opposite.

9.3.2 Jaccard Distance

- In this sets of items rated are considered while values in matrix are ignored.

$$d_J(A,B) = 1 - J(A,B)$$
$$= \frac{|A \cup B| - |A \cap B|}{|A \cup B|}$$

- Alternatively Jacard distance can be given by,

$$A \Delta B = (A \cup B) - (A \cap B)$$

- For example, user A and User B have an intersection of size 1 and a union size is of 5, its Jaccord similarity be 1/5 and Jacard distance be 4/5.
- User A and User C have Jaccard similarity 2/4 and Jaccard distance is same i.e. ½ .
- So, comparatively A and C are closer than A and B.
- User A and User C has very less matching choice of apps but user A and user B both rated nearly similar to one app i.e. Whatsapp.

9.3.3 Cosine Distance

- If user doesn't give any rating from 1 to 5 to any app then it is considered as a 0 (zero)

- Cosine angle between User A and User B is,

$$\frac{4 \times 5}{\sqrt{4^2+5^2+1^2}\sqrt{5^2+5^2+4^2}} = 0.380$$

- Cosine angle between user A and user C is,

$$\frac{5 \times 2 + 1 \times 4}{\sqrt{4^2+5^2+1^2}\sqrt{2^2+4^2+5^2}} = 0.322$$

- A is closer to B compare to C as longer cosine value implies a smaller angle.

9.3.4 Rounding the Data

- Rounding data by assigning one value to higher rating and assign NULL to lower values.
- For example, in our utility matrix few apps having ratings like 3, 4 and 5 will consider it as "1" and those having ratings like 2 and 1 will consider it as unrated keep it NULL.
- This approach will gives correct conclusion as Jaccard distance between A and B is 3/4 and A and C its 1, since C appears further from A compared to B.
- This approach will give correct conclusion and it can be verified by applying cosine distance that matrix will be.

Table 9.3.3 : Utility matrix values (ratings) 3, 4, 5 are replace by 1 and 2 and 1 value (ratings) are kept unrated (NULL)

Apps(items) / Users	WhatsApp	Hangout	Telegram	Viber	Skype	Hike
User A	1			1		
User B	1	1	1			
User C					1	1
User D		1				

9.3.5 Normalizing Rating

Low rating get convert into negative while high rating get converted into positive as it is subtracted from average rating, this is known as Rating Normalization.

9.4 Pros and Cons in Recommendation System

Q. Write any two pros and cons for Collaborative filtering and content-based filtering.

9.4.1 Collaborative Filtering

Pros

i) No knowledge engineering efforts needed.
ii) Serendipity in results.
iii) Continuous learning for market process.

Cons

(i) Rating feedback is required.

(ii) New items and users faces to cold start.

9.4.2 Content-based Filtering

Pros

(i) No. community requirement.

(ii) Items can be compared among themselves.

Cons

(i) Need of content description.

(ii) New users face cold start.

Review Questions

Q. 1 What is recommendation system?

Q. 2 Enlist application of recommendation system and taxonomy for application recommendation system.

Q. 3 Explain utility matrix with example.

Q. 4 Explain item profile of content based recommendation.

Q. 5 How item profile is represented ?

Q. 6 What is a user profile in content based recommendation ?

Q. 7 Explain in collaborative filtering.

Q. 8 What is measuring similarity in collaborative filtering ?

Q. 9 What is Jaccard distance and cosine distance in collaborative filtering ?

Q. 10 Explain rounding the data and normalized rating.

Q. 11 Write any two pros and cons for Collaborative filtering and content-based filtering.

□□□

CHAPTER 10

Module - 6

Mining Social Network Graph

Syllabus

Social Networks as Graphs, Clustering of Social-Network Graphs, Direct Discovery of Communities in a social graph

10.1 Introduction

Q. What is sociogram and Barbasi-Albert algorithm ?

- Social network idea came into theory and research in 1980s by Ferdinand Tonnis and Emile Durkhiem. Social network is bind with domain like social links, social group.
- Major work started in 1930s in various areas like mathematics, anthropology, psychology etc. I.L. Moreno provides foundation for social network as provided a Moreno's sociogram which represent social links related with a person.
- Moreno's sociogram example : Name the girl with whom you would like to go to industrial visit tour.

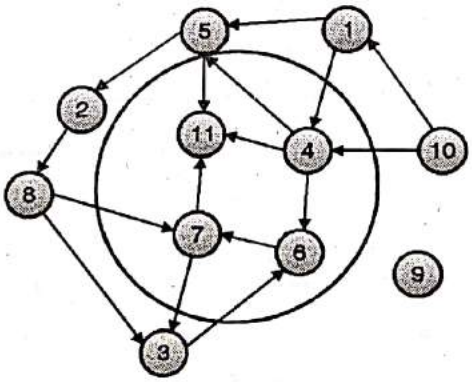


Fig. 10.1.1 : Moreno's sociogram with industrial visit tour

- Sociogram gives interpersonal relationship among members participated in group. Sociogram present choice in number.

$$C = \frac{\text{Number of mutual choices}}{\text{Number of possible mutual choices in the group}}$$

- The Barabasi-Albert (BA) model provides a network which initially connected with M_o nodes of network.

$$P_i = \frac{k_i}{\sum_j k_j}$$

Where, K_i - Degree of node i

j - All per existing node

- The new nodes gives preference to get attach with heavily linked nodes.

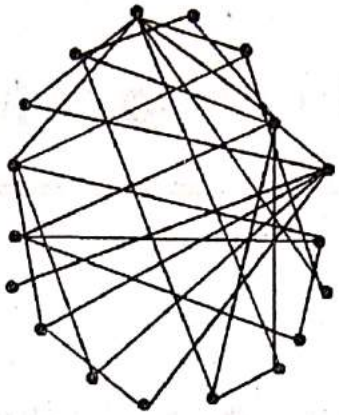


Fig. 10.1.2 : Barabasi algorithm model shows steps of growth of network ($M_0 = M = 2$)

- BA model is used to generate random scale free network. Scale-free network used in most of popular domain like the internet, World Wide Web, citation network and few social networks.
- Social network deals with large-scale data. After analyzing large data a huge set of information can be achieved.
- Linkedin, Facebook are vast widely used and very popular examples for social network. As we can find friends over the network with 1st, 2nd, 3rd connection or mutual friends (i.e. friends of friend) in Linkedin and Facebook respectively.
- Google+ is one of social network which gives link nodes in groups categories like Friends, Family, Acquaintances following Featured on Google+ etc.
- Social network is huge platform to analyze data and obtain information. Further will see efficient algorithm to discover different graphs properties.

10.2 Social Network as Graphs

MU - Dec. 16

Q. Write a note on Social Network Graphs. (Dec. 16, 10 Marks)

Q. How can a social network treated as graph?

- In general a graph is collection of set of edges (e) and set of vertices (V). If there is an edge exists between any two nodes of graph then that node relates with each other.
- Graphs are categories by many parameters like ordered pairs of nodes, unordered pairs of nodes.
- Some edge has direction, weight. Relationship among graph is explained with help of an adjacency matrix.
- Small network can be easily managed to construct a graph, it is quite impossible with huge/wide network.
- Summary statistics and performance metrics are useful for design of graph for a large network.
- Network and graphs can be elaborate with the help of few parameters like diameter i.e. largest distance between any two nodes, centrality degree distribution.
- Social website like Facebook uses undirected social graph for friends while directed graph used in social website like Twitter, Google+ (plus). Twitter gives connection like 1st , 2nd , 3rd and Google classify linked connection in friends, family, Acquaintances, Following etc.

10.2.1 Parameters Used in Graph (Social Network)

Q. Explain the graph parameters listed below :
(i) Degree (ii) Geodesic distance (iii) Density.
Q. How degree, closeness, between's centrality is measured?

Every node is distinct in a network and it is part of graph by set of links. Some general parameter consider for any social network as graph are :

(a) **Degree :** Number of adjacent nodes (considering both out degree and in-degree). Degree of node n_i denoted by $d(n_i)$.

(b) **Geodesic Distance :** Actual distance between two node n_i and n_j, expressed by d (i, j).

(c) **Density :** It gives correctness of a graph, it is useful to count closeness of network.

(d) **Centrality :** It tells about degree centrality i.e. nodes appearance in the centre of network centrality has types.

Example :

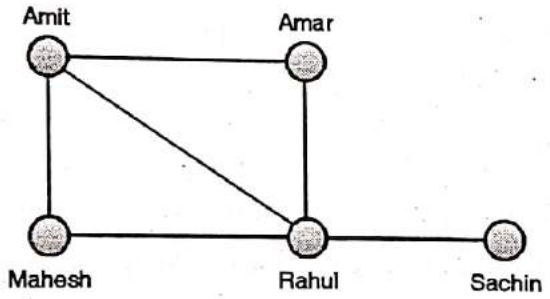


Fig. 10.2.1

Degree of each nodes are as follows :

Nodes	Degree
Amit	3
Amar	2
Mahesh	2
Rahul	4
Sachin	1

Density of undirected graph is 0.6.

Geodesic Distances between two nodes is as follows :

	Amit	Amar	Mahesh	Rahul	Sachin
Amit	–	1	1	1	2
Amar	1	–	2	1	2
Mahesh	1	2	–	1	2
Rahul	1	1	1	–	1
Sachin	2	2	2	1	–

Degree of centrality

$$C'_D(n_i) = \frac{d(n_i)}{(g-1)}$$

Closeness centrality

$$C'_C(n_i) = C'_C(n_i)(g-1)$$

Between's centrality

$$C_B(n_i) = C_B(n_i) / \left[\frac{(g-1)(g-2)}{2}\right]$$

$$C_B(n_i) = \sum_{j<k} g_{jk}(n_i) / g_{jk}$$

g_{jk} = The number of geodesics connecting jk

$g_{jk}(n_i)$ = The number that actor i is on.

10.2.2 Varieties of Social Network

Q. What is social network? Explain any one type in details.

10.2.2(A) Collaborative Network

- A network where each node has some value and as it gets connected with another node its values get changed.
- A tennis player has some records on his name in single. There are some other records on his name associated with another player name in doubles.
- A node may have different values depending on its connection with neighbouring node.
- Several kinds of data are available having two or more common networks.

10.2.2(B) Email Network

- When a node represents an Email account it is a single node. Every node of an e-mail is in link with at least one e-mail account (i.e. sender mail ID and receiver mail ID).
- Sometimes email are send from one side and sometime e-mail are send from both side in such scenario edges are supposed weak and strong respectively.

10.2.2(C) Telephone Network

- These nodes consist with values like phone numbers which gives it a distinct value.
- As a call is placed between two user nodes get some additional values like time of call period of communication etc.
- In telephone network edge gets weight by number of calls mode by it to other. Network assign edges with the way they contact each other like frequently, rarely, never get connected.

10.3 Clustering of Social Network Graphs

- Cluster gives data into subsets of related or linked objects. Custer coefficient gives degree to which various nodes of a graph tend to duster together.

- Graphs are used to represent data by few clustering algorithms. Clusters can be generated on basis of graph based properties.

10.3.1 Distance Measure for Social-Network Graphs

- Measuring a distance is an essential task for applying clustering technique on any graph. Few graph edge has label, it represents distance measure. Some edge of graph may be unlabeled.
- The distance d(x, y) = 0 if there is an existence or presence of an edge i.e. nodes appear close as there is an edge. The distance d(x, y) = 1 means no edge or nodes appear distant.
- ⊥ and ∞ can be used to represent values for an existing edge.

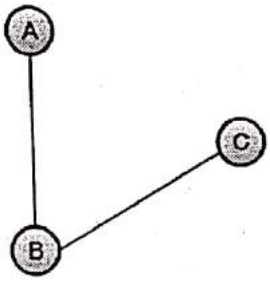


Fig. 10.3.1 : Example for triangle inequality

- '0 and ⊥' or '⊥ and ∞' these are not true 2-valued distance measure. Uses of these values violate triangle inequality when there are edge and nodes combination as shown in Fig. 10.3.1
- In above example, there is edge (A,B) and edge (B, C) but there is no any edge between node A and node C i.e. edge (A.C)
- Above example can be valued by assigning value ⊥ to distance of an existing edge and 1.5 to distance of missing edge

10.3.2 Applying Standard Cluster Method

Q. Explain following clustering algorithm in short :
(a) Hierarchical (b) K-means (c) K-medoid d) Fuzzy C-means.

- Clustering is popular technique to find patters from large dataset domain. It is useful in visualization of data and hypothesis generation.

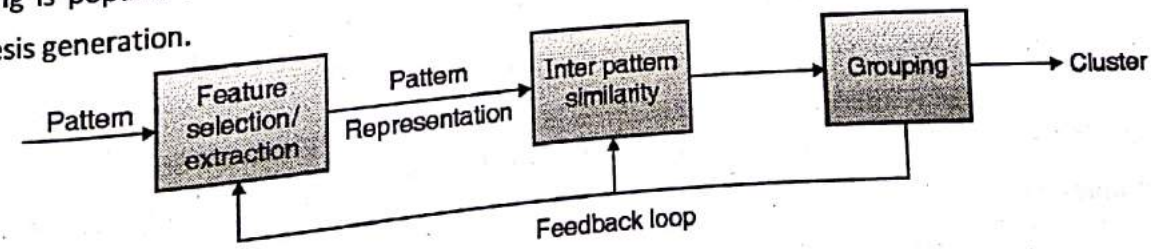


Fig. 10.3.2 : Overview of clustering

- Various clustering algorithms are as follows :

(A) Hierarchical
(B) K-means
(C) K-medoid
(D) Fuzzy C-means

(A) Hierarchical clustering

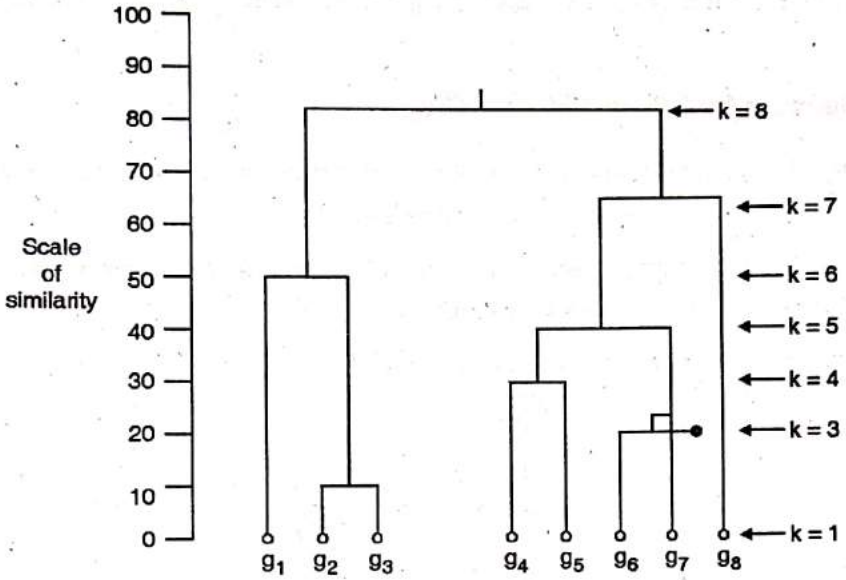


Fig. 10.3.3 : Hierarchical Clustering example

There are two types of hierarchical clustering :

(i) Agglomerative (bottom-up)

(ii) Divise (top-down)

(i) Agglomerative (bottom-up)

It starts with each document assuming as a single cluster, ever almost all documents are belonging to one cluster.

(ii) Divise (top-down)

It start with all document which are part of a single and same cluster. Every node generates a cluster for its own.

(B) K-means clustering

- It is one of the unsupervised clustering algorithm.
- Number of cluster represented by 'k' it is an input to algorithm.
- It is basically an iterative in nature, it work on numerical data and it is easy to for implementation.
- Bayesian Information Criterion (BIC) of Minimum Description Length (MDL) can be used to estimate K ('K' is a user input)
- It is easy to work with any distance measure with K-medoids, K-medoids is general version of K-means algorithm.

(C) K-medoid clustering

- It work with quantitative variable types and numerical.
- In both categorical variables and outliers Euclidean distances do not work in better way.
- It is more intensive.
- Compare to K-means, it is computationally costlier.
- It is applied for categorical data and when data points are not available (i.e. pair wise distances available only).

(D) Fuzzy C-means clustering (FCM)

- It is unsupervised and it always converges.
- It allows one piece of data which is part of two or more clusters.
- It is used frequently in pattern recognition.

10.3.3 Betweenness

- To find communities among social networks some specialized technique are developed as there are few problems with standard clustering methods.
- Betweenness is shortest path available between two nodes. For example an edge (x, y) is betweenness of node a and b such that the edge (x, y) lies on shortest path between a and b.
- a and b are two different communities where edge (x, y) lies somewhere as shortest path between a and b.

10.3.4 The Girvan - Newman Algorithm

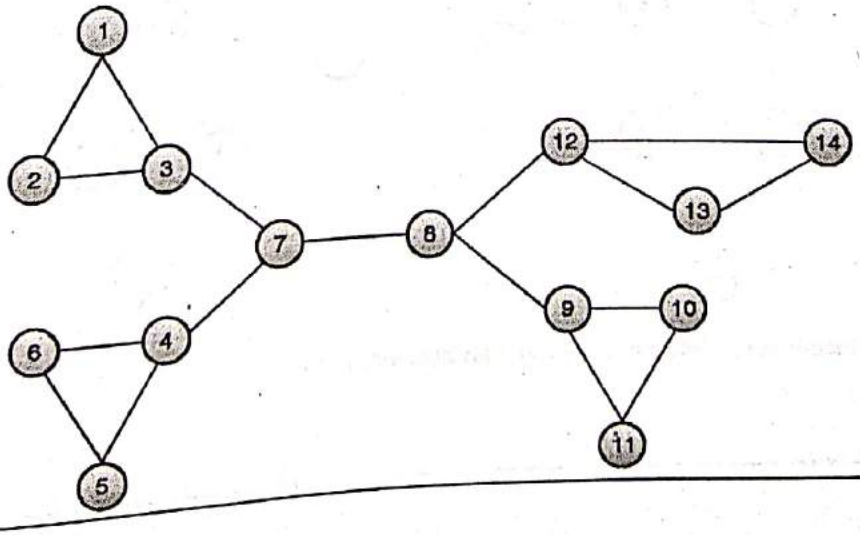
MU - May 19

Q. Explain Girvan-Newman algorithm to mine Social Graphs. **(May 19, 10 Marks)**

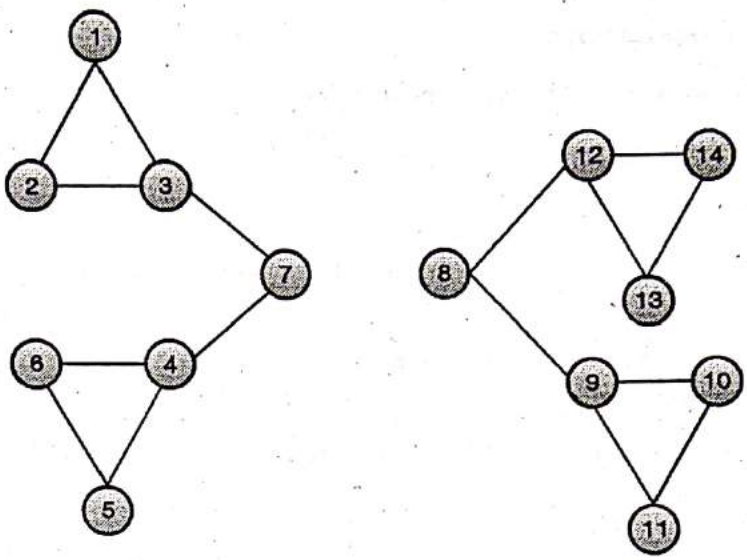
Q. Explain Girvan-Newman algorithm in details with help of suitable example.

- It is published in 2002 by Michelle Girvan and mark Newman for :
  - Community detection.
  - To measure edge - betweenness among all existing edges.
  - To remove edge having large valued betweenness.
  - To option optimized modular function.
- Girvan Newman algorithm checks for edge betweenness centrality and vertex betweenness centrality.
- Vertex betweenness centrality is total number of shortest path that pass through each vertex on the network.
- 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.
- Edge betweenness centrality is number of shortest path which pass through given edge.

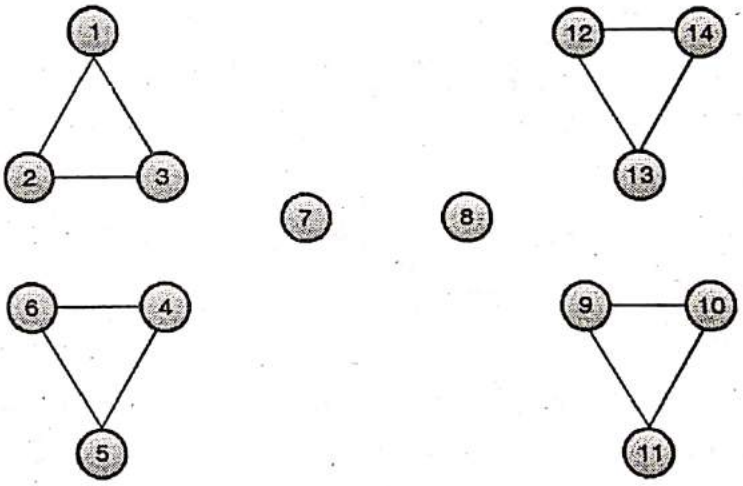
Example : Successive delete edges of high betweenness.



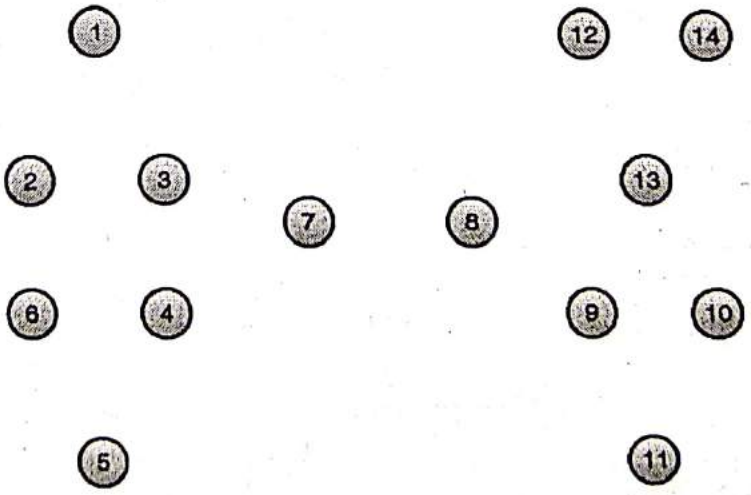
Step 1



Step 2



Step 3



– Standard process to successively deleting edges of high betweenness.

Step 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.

Step 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. Keep repeatation of tasks by recalculating all betweenness and removing the edge or edges having highest betweenness.

10.3.5 Using Betweenness to Find Communities

- It is an approach to find most shortest path within a graph which connect two vertex.
- It is process of systematically removal of edges, edges having highest betweenness are preferred to remove first, process is continued till graph is broken into suitable count of connected components.

Example :

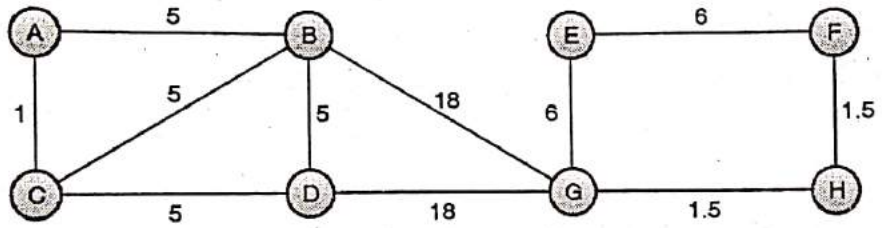


Fig. 10.3.4 : Betweenness score for graph example

- Between C and B there are two path, so edge (A, B), (B, D), (A, C) and (C, D) get credited by half a shortest path.
- Clearly, edge (D, G) and (B, G) has highest betweenness, so it will get removed first, it will generate components namely {A, B, C, D} and {E, F, G, H}.
- By keeping removal with highest betweenness next removal are with score 6 i.e. (E.G) and (E, F). Later, removal with score 5 i.e. (A, B), (B, D) and (C, D) .
- Finally graph remains as :

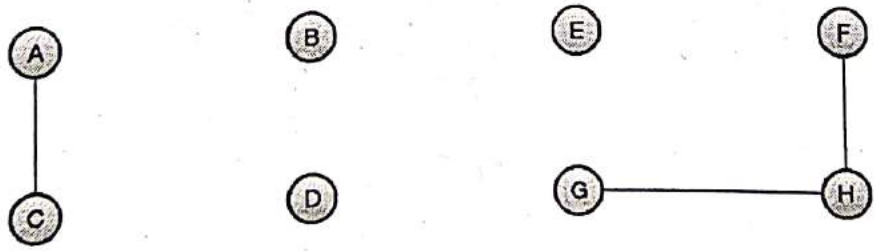


Fig. 10.3.5 : All edges with betweenness 5 and more are removed

- 'Communities' implies that A and C more close to each other than to B and D. In short B and D are "traitor" to community {A, B, C, D} because they have friend G outside the community.
- Similarly G is "traitor" to group {E, F, G, H} and only F, G and H remain connected.

10.4 Direct Discovery of Communities

MU - Dec. 17

Q. Explain any one algorithm for finding communities in a social graph. (Dec. 17, 6 Marks)

Q. Explain communities discovery.

Discovery of communities deals with large number edges search from a graph.

Finding cliques

- Cliques can be defined as a set of nodes having edges between any two of vertices.
- To find clique is quite difficult task. To find largest set of vertices where any two vertices needs to be connected within a graph is known as maximum clique.

10.4.1 Bipirate Graph

- It is graph having vertices which can be partitioned into two disjoint sets suppose set V and set U. Both V and U sets are not necessary of having same size.
- A graph is said to be bipirate if and only if it does not posses a cycle of an odd length.

Example :

Suppose we have 5 engines and 5 mechanics where each mechanic has different skills and can handle different engine by vertices in U. An edge between two vertices to shows that the mechanics has necessary skill to operate the engine which it is linked. By determining maximum matching we can maximize the number of engines being operated by workforce.

10.4.2 Complete Bipirate Graph

- A graph $K_{m, n}$ is said to be complete bipirate graph as its vertex set partitioned into two subsets of m and n vertices respectively.
- Two vertices are connected if they belong to different subsets.

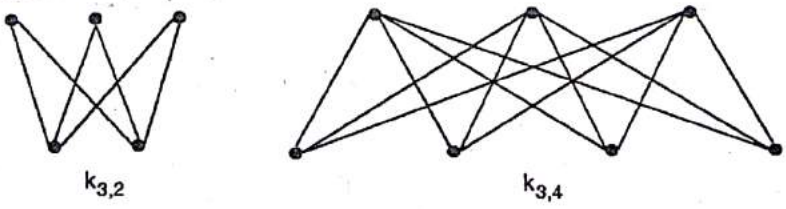


Fig. 10.4.1

10.5 Simrank

Q. What is simrank ? Explain in brief.

- It is one of the approach of analyze a social network graphs.
- Graphs consists of various types of nodes, simrank is useful to calculate the similarity from same type nodes.
- Simrank is useful for random walkers on a social graph while starting with a particular node.
- Simrank needs calculation and it is done at every starting node for limited sizes graph.

10.5.1 Random Walker on Social Network

- Social network graph is mostly undirected and web graph founds directed. Random walker of social graph can meet to any number of neighboring node of it.

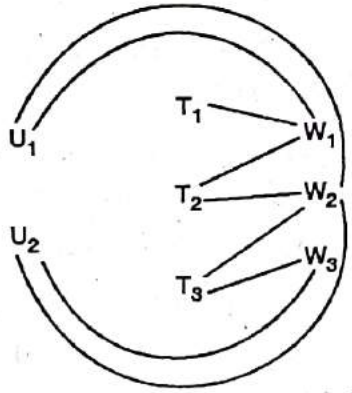


Fig. 10.5.1 : A tripartite graph example for random walker social network

- Suppose, example as shown in Fig. 10.5.1 U-users, T-tags, W-web pages.
- At very first step walker will go for U_1 or W_1. If walker prefer to go W_1 then in next attempt it will visit to T_1 or T_2. If walker prefer to visit U_1 then in next attempt it will meet either T_1, T_2 or T_3.
- T_1 and T_2 both tag placed on webpage W_1 and reachable by user U_1 and U_2.
- By keeping trend of visiting start at any node traversal can visit all node of a_1 network. Irrespective of start node walker can visit all node so it is known as random walker on social network.

10.5.2 Random Walks with Restart

- Random node visiting walker may stop at some node in random.
- To know when a random walker may stop can be calculate with help of probabilities putting in a matrix of transition.
- Suppose, M is transition matrix of graph G entering at row a column b of M is 1/k if node b of graph having degree K and one of the adjacent node is a else entry is 0 (zero).

Example :

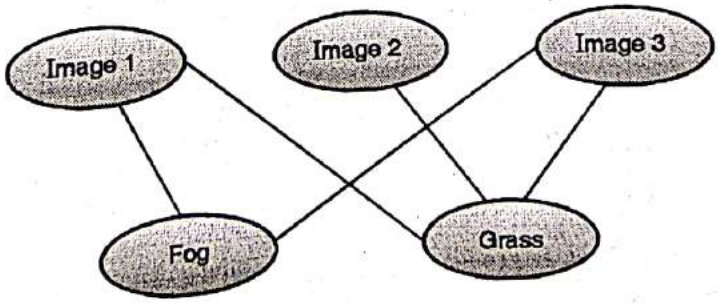


Fig. 10.5.2 : A simple bipartite social graph

- In Fig. 10.5.2 network consists of three images, and two tags "Fog" and "Grass" Image1 and Image 3 has two tags and image 2 has only one tag i.e. "Grass".
- Image 1 and Image 3 are comparably more similar than Image 2 and random walker with restart at Image 1 probably support that intention after applying analysis to it.

- Nodes can be kept in order like Image 1, Image 2, Image 3, fog, grass. The transaction matrix for graph will be like.

$$\begin{bmatrix} 0 & 0 & 0 & 1/2 & 1/3 \\ 0 & 0 & 0 & 0 & 1/3 \\ 0 & 0 & 0 & 1/2 & 1/3 \\ 1/2 & 0 & 1/2 & 0 & 0 \\ 1/2 & 1 & 1/2 & 0 & 0 \end{bmatrix}$$

- The fifth column of node "Grass" which is connected to each of image node. If therefore it has some degree like 3 then non-zero entries to node "Grass" column must have to be 1/3.
- The image nodes correspond to first three rows and first three columns of transaction matrix so entry 1/3 appears in the first three rows of column 5. Since "fog" node does not have an edge to either itself or "Grass" node.
- Let, β be probability of random walker, so $1-\beta$ is probability the walker will teleport to initial node N. e_N is column vector that has 1 in the row for node N otherwise its 0 (zero).

10.6 Counting Triangles using MapReduce

Q. What is counting triangles using MapReduce? Enlist its application.

- In era of Big data, approx 2.5 quintillion byte data increasing per day. In 2004, Google introduced Map Reduce used in search engine.
- Map Reduce used for processing and to generate large data sets. Map function gives data processed by a pair of key set while Reduce function used to merge those data values.
- Map and reduce function likewise function introduced in Lisp also.

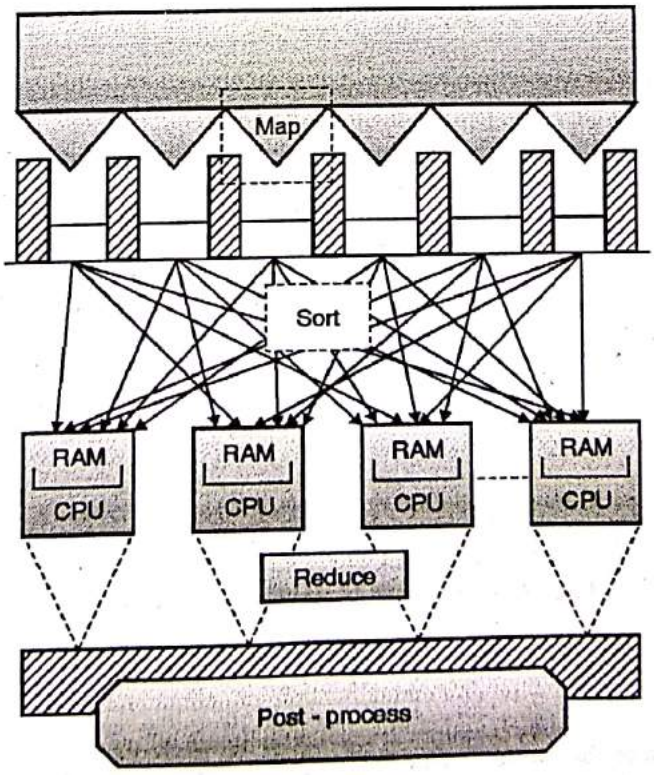


Fig. 10.6.1 : Map-Reduce process general overview

- Map-Reduce process does with help of three stages :
  - Mapping
  - Shuffle
  - Reducing
- Counting of triangle is helpful to know community around any node within social network; it helps to know 'clustering co-efficient'.
- Let, a graph G = (V.E.) a simple undirected and unweighted graph.

$$\text{Let, } n = |v|$$
$$m = |E|$$
$$T(v) = \text{set of neighbors of } v$$
$$= \{ W \in V \mid (v, w) \in E \}$$
$$dv = |T_{(v)}|$$

Cluster co-efficient (cc(v)) for a node where $v \in V$ is,

$$\binom{dv}{2} cc(v) = |\{(u, w) \in E \mid u \in T(v), w \in T(v)|$$

Above expression gives cluster coefficient for node v

- Map Reduce is used in page ranking. It is also useful in :
  - (i) Web access log states,
  - (ii) Inverted index construction
  - (iii) Document clustering
  - (iv) Statistical machine translation
  - (v) Machine learning
  - (vi) Web link-graph reversal
  - (vii) Distributed sorting
  - (viii) Distributed pattern based search
  - (ix) Machine translation

Review Questions

Q.1 What is sociogram and Barbasi-Albert algorithm?

Q.2 How can a social network treated as graph ?

Q.3 Explain the graph parameters listed below :
  - (i) Degree
  - (ii) Geosestic distance
  - (iii) Density

Q. 4 How degree, closeness, between's centraliy is measured ?

Q. 5 What is social network ? Explain any one type in details.

Q. 6 Explain following clustering algorithm in short :

(a) Hierarchical

(b) K-means

(c) K-medoid

(d) Fuzzy C-means

Q. 7 Explain Girvan-Newman algorithm in details with help of suitable example.

Q. 8 What is betweenness ? Explain use of betweenness to find communities.

Q. 9 Explain bipirate graph and complete bipirate graph.

Q. 10 What is simrank ? Explain in brief.

Q. 11 What is MapReduce ? Enlist its application ?

□□