SPPU New Syllabus

A Book Of

RECENT TRENDS IN IT

For B.B.A.(CA) : Semester - VI

[Course Code CA - 601 : Credits = 3 + 1 = 04]

CBCS Pattern

As Per New Syllabus, Effective from June 2021

Dr. Pallawi Bulakh

M.Sc. M.Phil., Ph.D., NET with JRF
Asst. Professor
Department of Computer Science
Modern College of Arts, Science and Commerce
Ganeshkhind, Pune-16
Member - BOS, Computer Application
Savitribai Phule Pune University

Mrs. Meenal Kaustubh Jabde

M.Sc. Computer Science (Pursuing Ph.D.)
Asst. Professor
Modern College of Arts, Science & Commerce
Ganeshkhind, Pune 16

Price ₹ 150.00

NIRALI PRAKASHAN
ADVANCEMENT OF KNOWLEDGE

N6187

RECENT TRENDS IN IT

ISBN 978-93-5451-363-3

First Edition : January 2022

Published By :
NIRALI PRAKASHAN
Abhyudaya Nagar, 1312, Shivaji Nagar,
Off J.M. Road, Pune – 411005
Tel : (020) 25512336/37/39
Email : niralipune@pragationline.com

Polyplate

Printed By :
YOGIRAJ PRINTERS AND BINDERS
Survey No. 10/1A, Ghule Industrial Estate
Nanded Gaon Road
Nanded, Pune - 411041

DISTRIBUTION CENTRES

PUNE

Nirali Prakashan
(For orders outside Pune)
S. No. 28/27, Dhayari Narhe Road, Near Asian College
Pune 411041, Maharashtra
Tel : (020) 24690204; Mobile : 9657703143
Email : bookorder@pragationline.com

Nirali Prakashan
(For orders within Pune)
119, Budhwar Peth, Jogeshwari Mandir Lane
Pune 411002, Maharashtra
Tel : (020) 2445 2044; Mobile : 9657703145
Email : niralilocal@pragationline.com

MUMBAI
Nirali Prakashan
Rasdhara Co-op. Hsg. Society Ltd., 'D' Wing Ground Floor, 385 S.V.P. Road
Girgaum, Mumbai 400004, Maharashtra
Mobile : 7045821020, Tel : (022) 2385 6339 / 2386 9976
Email : niralimumbai@pragationline.com

DISTRIBUTION BRANCHES

DELHI
Nirali Prakashan
Room No. 2 Ground Floor
4575/15 Omkar Tower, Agarwal Road
Darya Ganj, New Delhi 110002
Mobile : 9555778814/9818561840
Email : delhi@niralibooks.com

BENGALURU
Nirali Prakashan
Maitri Ground Floor, Jaya Apartments,
No. 99, 6th Cross, 6th Main,
Malleswaram, Bengaluru 560003
Karnataka; Mob : 9686821074
Email : bengaluru@niralibooks.com

NAGPUR
Nirali Prakashan
Above Maratha Mandir, Shop No. 3,
First Floor, Rani Jhanshi Square,
Sitabuldi Nagpur 440012 (MAH)
Tel : (0712) 254 7129
Email : nagpur@niralibooks.com

KOLHAPUR
Nirali Prakashan
New Mahadvar Road, Kedar Plaza,
1st Floor Opp. IDBI Bank
Kolhapur 416 012 Maharashtra
Mob : 9850046155
Email : kolhapur@niralibooks.com

JALGAON
Nirali Prakashan
34, V. V. Golani Market, Navi Peth,
Jalgaon 425001, Maharashtra
Tel : (0257) 222 0395
Mob : 94234 91860
Email : jalgaon@niralibooks.com

SOLAPUR
Nirali Prakashan
R-158/2, Avanti Nagar, Near Golden
Gate, Pune Naka Chowk
Solapur 413001, Maharashtra
Mobile 9890918687
Email : solapur@niralibooks.com

marketing@pragationline.com | www.pragationline.com

Also find us on www.facebook.com/niralibooks

Preface ...

We take this opportunity to present this book entitled as **"Recent Trends in IT"** students of Sixth Semester – BBA (Computer Applications). The object of this boo present the subject matter in a most concise and simple manner. The book is written according to the New Syllabus (CBCS Pattern-2019).

The book has its own unique features. It brings out the subject in a very simple an manner for easy and comprehensive understanding of the basic concepts, its intri procedures and practices. This book will help the readers to have a broader vi Recent Trends in IT. The language used in this book is easy and will help students to in their vocabulary of Technical terms and understand the matter in a better and happier

We sincerely thank Shri. Dineshbhai Furia and Shri. Jignesh Furia of Nirali Prakash the confidence reposed in us and giving us this opportunity to reach out to the stud BBA(Computer Application) studies.

We have given our best inputs for this book. Any suggestions towards the improv of this book and sincere comments are most welcome on niralipune@pragationline.com

Au

■■■

Syllabus ...

1. Introduction to Recent Trends **[Lectures 2]**

1.1 Artificial Intelligence

1.2 Data Warehouse

1.3 Data Mining

1.4 Spark

2. Artificial Intelligence **[Lectures 8]**

2.1 Introduction and Concept of AI

2.2 Applications of AI

2.3 Artificial Intelligence, Intelligent Systems, Knowledge – Based Systems, AI Techniques

2.4 Early Work in AI and Related Fields

2.5 Defining AI Problems as a State Space Search

2.6 Search and Control Strategies

2.7 Problem Characteristics

2.8 AI Problem: Water Jug Problem, Tower of Hanoi, Missionaries and Cannibal Problem

3. AI Search Techniques **[Lectures 8]**

3.1 Blind Search Techniques: BFS, DFS, DLS, Iterative deepening Search, Bidirectional Search, and Uniform Cost Search

3.2 Heuristic Search Techniques: Generate and Test, Hill Climbing, Best First Search, Constraint Satisfaction, Mean - End Analysis, A*, AO*

4. Data Warehousing **[Lectures 8]**

4.1 Introduction to Data Warehouse

4.2 Structure of Data Warehouse

4.3 Advantages and Uses of Data Warehouse

4.4 Architecture of Data Warehouse

4.5 Multidimensional Data Model

4.6 OLAP Vs. OLTP

4.7 OLAP Operations

4.8 Types of OLAP Servers: ROLAP Versus MOLAP Versus HOLAP

5. Data Mining **[Lectures 12]**

5.1 Introduction to Data Mining

5.2 Data Mining Task

5.3 Data Mining Issues

5.4 Data Mining Versus Knowledge Discovery in Databases

5.5 Data Mining Verification vs. Discovery

5.6 Data Pre-processing - Need, Data Cleaning, Data Integration and Transformation, Data Reduction

5.7 Accuracy Measures: Precision, Recall, F-measure, Confusion Matrix, Cross-Validation, Bootstrap

5.8 Data Mining Techniques

5.9 Frequent Item-sets and Association Rule Mining: Apriori Algorithm, FP Tree Algorithm

5.10 Graph Mining: Frequent Sub-graph Mining

5.11 Software for Data Mining: R, Weka, Sample Applications of Data Mining

5.12 Introduction to Text Mining, Web Mining, Spatial Mining, Temporal Mining

6. Spark **[Lectures 10]**

6.1 Introduction to Apache Spark

6.2 Spark Installation

6.3 Apache Spark Architecture

6.4 Components of Spark

6.5 Spark RDDs

6.6 RDD Operations: Transformation and Actions

6.7 Spark SQL and Data Frames

6.8 Introduction to Kafka for Spark Streaming

■■■

Contents ...

■■■

1... Introduction to Recent Trends

Learning Objectives ...

- To get familiar with the features of Artificial Intelligence.
- To understand the concept of Data Mining and Data Warehousing.
- To introduce the concept of Spark.

1.1 INTRODUCTION

- Artificial Intelligence is not a new word and not a new technology for researchers. This technology is much older than you would imagine. Even there are the myths of Mechanical men in Ancient Greek and Egyptian Myths.
- The word "Artificial Intelligence" was first adopted by American Computer scientist John McCarthy at the Dartmouth Conference. For the first time, AI was coined as an academic field.

1.2 ARTIFICIAL INTELLIGENCE

- Artificial Intelligence (AI), as we know it today, is not a new field. Nowadays, we can say that AI is all around us. Artificial intelligence is progressing rapidly into diverse areas in modern society. AI can be used in several areas such as research in the medical field or creating innovative technology, for instance, autonomous vehicles and many more. The definitions have also changed in the course of time, due to the rapid developments. Definitions that is popular as *"imitating intelligent human behavior,"* which is self-explanatory.
- Artificial Intelligence emerged as an increasingly impactful discipline in science and technology.
- Artificial Intelligence is a technology that replicates human intelligence in a machine. If a machine can learn, think, and execute just as a human brain would do, in short, the intelligence that is exhibited by a machine is known as artificial intelligence.

- Artificial Intelligence helps a machine or system in thinking, rationalizing, learning, reasoning, problem-solving, planning, decision making and language processing. Hence, when a machine can take decisions and perform tasks on its own, it is called an intelligent machine.

1.2.1 Advantages of AI

1. **Reduction in Human Error:** Humans make mistakes from time to time, but computers don't if they programmed properly. With AI, the decisions are taken from the previously gathered information applying a certain set of algorithms.
2. **Available 24×7:** An average human will work for 6-8 hours a day excluding the breaks. But by using AI we can make machines work 24×7 without any breaks.
3. **Digital Assistance:** Highly advanced organizations use digital assistants to interact with users which save the need of human resources. The digital assistant is also used in many websites to provide things that the user wants. We can chat with them about what we are looking for. Some chatbots are designed in such a way that it becomes hard to determine that we are chatting with a chatbot or a human being.
4. **Faster Decisions:** While making a decision humans will analyze many factors both emotionally and practically, but AI-powered machines work on what is programmed and deliver the results in a faster way.

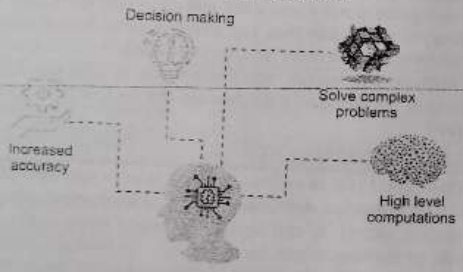


Fig. 1.1: Advantages of AI

1.2.2 Disadvantages of AI

1. **Making Human Idle:** AI is making humans idle with its applications automating most of the work. Humans tend to get addicted to these inventions which can cause problems for future generations.
2. **Unemployment:** As AI is replacing most of the repetitive tasks and other work with robots, human interference is becoming less, which will cause a major problem in the employment standards. Every organization is looking to replace the minimum qualified individuals with AI robots which can do similar work with more efficiency.

3. **High Costs of Creation:** As AI is updating every day the hardware and software need to get updated with time to meet the latest requirements. Machines need repairing and maintenance which need plenty of costs. Its creation requires huge costs as they are very complex machines.
4. **No Emotions:** There is no doubt that machines are much better when it comes to working efficiently but they cannot replace the human connection that makes the team. Machines cannot develop a bond with humans, which is an essential attribute when it comes to Team Management.

1.3 DATA WAREHOUSE

- **Data Warehouse**, also known as DWH is a system that is used for reporting and data analysis.
- The term Data Warehouse was first coined by Bill Inmon.
- Data Warehouse is a concept which supports decision support systems where a large amount of data is merged.
- A data warehouse is a repository which is at the top of multiple databases.
- It can be defined as a process for collecting and managing data from varied sources to provide meaningful business insights.
- A data warehouse is used to analyze the data, to generate reports based on that analysis. The data used for analysis or reporting is heterogeneous data collected from multiple sources.
- The main purpose of data warehousing is to combine such data for the purpose of analyzing and reporting. The data is then used for the strategic planning and decision making for the organizations.

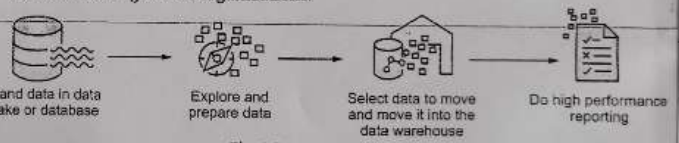


Fig. 1.2: Data warehousing

- Different tools are used for data warehouse in such as Amazon Redshift, Microsoft Azure, Snowflake, Micro Focus Vertica, Teradata, MarkLogic.

1.3.1 Advantages of Data Warehousing

1. Data Warehouse allows business users to quickly access critical data from some sources all in one place.
2. Data Warehouse stores a large amount of historical data that helps users to analyze different time periods and trends to make future predictions.
3. Data Warehouse helps to reduce total turnaround time for analysis and reporting.
4. Data Warehouse provides consistent information on various cross-functional activities.
5. Data Warehouse helps to integrate many sources of data to reduce stress on the production system.

1.3.2 Disadvantages of Data Warehousing

1. Not an ideal option for unstructured data.
2. Difficult to make changes in data types and ranges, data source schema, indexes, and queries.
3. The data warehouse may seem easy, but it is too complex for the average users.

1.3.3 Applications

- Listed below are the applications of Data warehouses across numerous industry backgrounds.

1. **Transportation Industry:** In the transportation industry, data warehouses record customer data enabling traders to experiment with target marketing where the marketing campaigns are designed by keeping customer requirements in mind.
2. **Services Sector:** Data Warehouses find themselves to be of use in the service sector for maintenance of financial records, revenue patterns, customer profiling, resource management, and human resources.
3. **Manufacturing and Distribution Industry:** A manufacturing organization has to take several make-or-buy decisions which can influence the future of the sector, which is why they utilize high-end OLAP tools as a part of data warehouses to predict market changes, analyze current business trends, detect warning conditions, view marketing developments, and ultimately take better decisions.
4. **Healthcare:** In the Healthcare sector, all of their financial, clinical, and employee records are fed to warehouses as it helps them to strategize and predict outcomes, track and analyze their service feedback, generate patient reports, share data with tie-in insurance companies, medical aid services, etc.
5. **Government and Education:** The government uses data warehouses to maintain and analyze tax records, health policy records and their respective providers, and also their entire criminal law database is connected to the state's data warehouse. Criminal activity is predicted from the patterns and trends, results of the analysis of historical data associated with past criminals.

   Universities use warehouses for extracting of information used for the proposal of research grants, understanding their student demographics, and human resource management. The entire financial department of most universities depends on data warehouses, inclusive of the Financial Aid department.
6. **Retailing:** Retailers are the mediators between wholesalers and end customers, and that's why it is necessary for them to maintain the records of both parties. For helping them store data in an organized manner, the application of data warehousing comes into the frame.

7. **Banking:** Data warehouse used in the banking industry for analyzing consumer data, market trends, government regulations and reports. It is more importantly used for financial decision making.

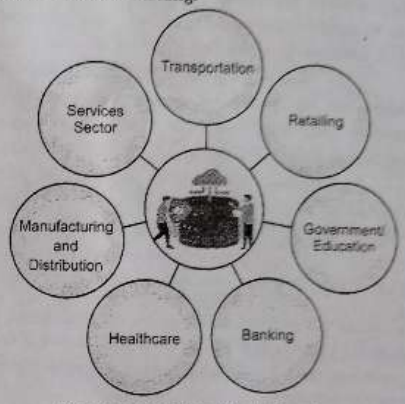


Fig. 1.3: Applications of Data Warehouse

1.4 DATA MINING

- Large amount of data is available in different industries and organizations. The availability of this huge data is of no use unless it is converted into valuable information. Otherwise, we are sinking in data, but starving for knowledge. The solution to this problem is data mining which is the separation of useful information from the huge amount of data that is available.
- Data mining is defined as: "Data mining, also known as Knowledge Discovery in Data (KDD), is the process of uncovering patterns and other valuable information from large data sets".
- Its foundation comprises three intertwined scientific disciplines: statistics (the numeric study of data relationships), artificial intelligence (human-like intelligence displayed by software and/or machines) and machine learning (algorithms that can learn from data to make predictions).
- It is mining knowledge from data i.e., process of extracting hidden information from a large data set.
- Data Mining deals with discovery of hidden knowledge, unexpected patterns, and new rules from large data sets.
- Data mining uses data from various data sources and that data need to be integrated preprocessed before data mining can be done.

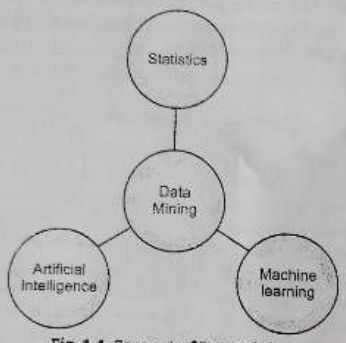


Fig. 1.4: Concept of Data Mining

- Different visualization techniques like graphical, icon based, hierarchical geometric, pixel based, and hybrid are used for representing output of data.
- Different tools are used for data mining like Rapid Miner, Oracle Data Mining, Python, Kaggle and Rattle.

1.4.1 Advantages of Data Mining

1. The data mining helps financial institutions and banks to identify probable defaulters and hence will help them whether to issue credit card, loan etc. or not. This is done based on past transactions, user behavior and data patterns.
2. The data mining based methods are cost effective and efficient compared to other statistical data applications.
3. It has been used in many different areas or domains viz. bioinformatics, medicine, genetics, education, agriculture, law enforcement, e-marketing, electrical power engineering etc. For example, in genetics it helps in predicting risk of diseases based on DNA sequence of individuals.

1.4.2 Disadvantages of Data Mining

1. The information obtained based on data mining by companies can be misused against a group of people.
2. The data mining techniques are not 100% accurate and may cause serious consequences in certain conditions.
3. Different data mining tools work in different manners due to different algorithms employed in their design. Hence the selection of the right data mining tool is a tedious and cumbersome task as one needs to obtain knowledge of algorithms, features etc. of various available tools.

1.4.3 Applications of Data Mining

- Data mining is used by many organizations to improve the customer base. They focus on customer behavioral patterns, market analysis, profit areas and product improvement. The essential areas where data mining is used are as follows:
  - **(a) Education:** Educational data mining deals with developing the methods to discover the knowledge from the education field. It is used to find out project students' areas of interests, future learning capacities and other aspects. Educational institutions can apply different data mining techniques and take appropriate/accurate decisions based on the outcome of the mining process. Also, the analysis of slow and fast learners and accordingly their teaching pattern can be determined.
  - **(b) Health and Medicine:** Data mining can effectively be used in health care systems. During Covid-19 pandemic, the predictions of the Covid-19 waves and the volume of patients was done using data mining. In Genetics also, data mining helps in determining the sequence of the genes and future trends.
  - **(c) Market Analysis:** Market analysis is based on a particular pattern of purchase followed by customers. These patterns help the shop owner to understand the buying pattern of customers and accordingly useful decisions can be implemented so as to increase the profit of the store. Also, the market analysis helps to find out the different methodologies to retain the existing customers and gain new ones.
  - **(d) Fraud Detection:** A fraud detection system helps in finding out the pattern of fraud, its potential attackers/criminal detection and possible solutions using different data mining algorithms. These data mining methods provide timely and efficient solutions for detection and prevention of the frauds. Intrusion and lie detection can also be addressed by these mechanisms.

1.5 SPARK

- Spark is a general purpose distributed data processing engine that is suitable for use in a wide range of circumstances.
- Spark is one of Hadoop's sub-projects developed in 2009 in UC Berkeley's AMP Lab by Matei Zaharia. It was open sourced in 2010 under a BSD license. It was donated to Apache software foundation in 2013, and now Apache Spark has become a top-level Apache project from Feb. 2014.
- Spark is an accessible, intense, powerful, and proficient Big Data tool for handling different enormous information challenges.
- It is an open source, wide range data processing engine. That reveals development API's, which also qualifies data workers to accomplish streaming, machine learning or SQL workloads which demand repeated access to data sets. However, Spark can perform batch processing and stream processing. Batch processing refers to the processing of the previously collected job in a single batch. Whereas stream processing means to deal with Spark streaming data.
- Moreover, it is designed in such a way that it integrates with all the Big Data tools. Like spark can access any Hadoop data source, also can run on Hadoop clusters.

- Spark supports the Java, Scala, Python, and R languages.
- It incorporates libraries with composable APIs for Machine Learning (MLlib), SQL for interactive queries (Spark SQL), Sream processing (Structured Streaming) for interacting with real-time data, and Graph processing (GraphX).

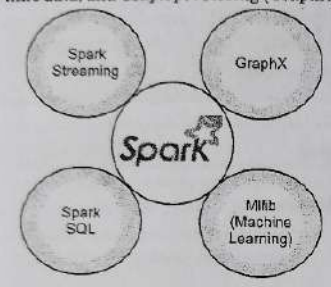


Fig. 1.5: Spark Components

1.5.1 Advantages of Spark

1. When it comes to Big Data, processing speed always matters. Spark is wildly popular with data scientists because of its speed. Spark is 100x faster than Hadoop for large scale data processing. Spark uses an in-memory (RAM) computing system whereas Hadoop uses local memory space to store data. Spark can handle multiple petabytes of clustered data of more than 8000 nodes at a time.
2. Spark carries easy-to-use APIs for operating on large datasets. It offers over 80 high-level operators that make it easy to build parallel apps.
3. Spark not only supports 'MAP' and 'reduce'. It also supports Machine learning (ML), Graph algorithms, Streaming data, SQL queries, etc.
4. Spark can handle many analytics challenges because of its low-latency in-memory data processing capability. It has well-built libraries for graph analytics algorithms and machine learning.

1.5.2 Disadvantages of Spark

1. In the case of Apache Spark, you need to optimize the code manually since it doesn't have any automatic code optimization process. This will turn into a disadvantage when all the other technologies and platforms are moving towards automation.
2. Apache Spark doesn't come with its own file management system. It depends on some other platforms like Hadoop or other cloud-based platforms.
3. There are fewer algorithms present in the case of Apache Spark Machine Learning Spark MLlib. It delays in terms of several available algorithms.

1.5.3 Applications of Spark

1. **Machine Learning:** Machine learning approaches become more feasible and increasingly accurate due to enhancement in the volume of data. As spark is capable of storing data in memory and can run repeated queries quickly, it makes it easy to work on machine learning algorithms.
2. **Data Integration:** The data generated by systems are not consistent enough to combine for analysis. To fetch consistent data from systems we can use processes like Extract, Transform, and Load (ETL). Spark is used to reduce the cost and time required for this ETL process.
3. **Processing Streaming Data:** It is always difficult to handle the real-time generated data such as log files. Spark is capable enough to operate streams of data and refuses potentially fraudulent operations.
4. **Interactive Analytics:** Spark is able to generate the respond rapidly. So, instead of running pre-defined queries, we can handle the data interactively.

Summary

- A branch of Computer Science named Artificial Intelligence pursues creating computers or machines as intelligent as human beings.
- The data warehouse works as a central repository for where information is coming from one or more data sources.
- Data Mining is defined as the procedure of extracting information from huge sets of data.
- Spark is a lightning-fast cluster computing technology, designed for fast computation.

Check Your Understanding

1. What is Artificial intelligence?
   (a) Putting your intelligence into Computer.
   (b) Programming with your own intelligence.
   (c) Making a Machine Intelligent.
   (d) Playing a Game.
2. Spark was initially started by _____ at UC Berkeley AMPLab in 2009.
   (a) Mahek Zaharia (b) Matei Zaharia
   (c) Doug Cutting (d) Stonebraker
3. Data Mining is also called _____.
   (a) Data Processing (b) Data Discovery
   (c) Knowledge Discovery in Data (d) Knowledge Processing

4. _____ is a subject-oriented, integrated, time-variant, nonvolatile collection of data in support of management decisions.
   (a) Data Warehousing (b) Data Mining
   (c) Web Mining (d) Text Mining
5. Spark is packaged with higher level libraries, including support for _____ queries.
   (a) C (b) C++
   (c) SQL (d) All of the above
6. The important aspect of the data warehouse environment is that data found within the data warehouse is _____.
   (a) Subject-oriented (b) Time-variant
   (c) Integrated (d) All of the above
7. Which is not data mining tool?
   (a) Rapid Miner (b) Oracle Data Mining
   (c) Kaggle (d) JavaScript
8. A data warehouse is which of the following?
   (a) Can be updated by end users.
   (b) Contains numerous naming conventions and formats.
   (c) Organized around important subject areas.
   (d) Contains only current data.
9. A goal of data mining includes which of the following?
   (a) To explain some observed event or condition.
   (b) To confirm that data exists.
   (c) To analyze data for expected relationships.
   (d) To create a new data warehouse.
10. The term Data Warehouse was first coined by _____.
    (a) Roger Mougalas (b) Arthur Samuel
    (c) Bill Inmon (d) Herbert Simon

Answers

1. (c)	2. (b)	3. (c)	4. (a)	5. (c)	6. (d)	7. (d)	8. (c)	9. (a)	10. (c)

Practice Questions

Q.I Answer the following questions in short.

1. What is Data Mining?
2. What is Data Warehousing?
3. Explain Spark.
4. What is the application of Artificial Intelligence in healthcare industry?
5. Which are components of Spark?

Q.II Answer the following questions.

1. Explain Advantages and Disadvantages of Data Mining.
2. Describe Advantages and Disadvantages of Data Warehousing.
3. What are the Advantages and Disadvantages of Artificial Intelligence?
4. What is Spark? Discuss its Advantages and Disadvantages.
5. Write applications of Data Mining.

Q.III Define the terms.

1. Data Mining
2. Data Warehousing
3. Spark
4. Artificial Intelligence

2... Artificial Intelligence

Learning Objectives ...

- To introduce concept of AI along with its applications and techniques.
- To study about how to represent an AI problem in state space search.
- To know about the basics of Search and Control strategies.
- To get knowledge about different Problem characteristics.

2.1 INTRODUCTION AND CONCEPT OF AI

What is AI?

- *"AI is the study of how to make computers do things which at the moment, people do better".*

OR

- *"AI is the science and engineering of making machines intelligent".*

OR

- *"Artificial intelligence (AI) is an area of computer science that involves building smart machines that are able to perform tasks which usually require human intelligence".*

OR

- According to the father of Artificial Intelligence, John McCarthy, it is *"The science and engineering of making intelligent machines, especially intelligent computer programs".*
- There are many definitions around, but most of them can be classified into the following four categories:
  1. Systems that think like humans.
  2. Systems that act like humans.
  3. Systems that think rationally.
  4. Systems that act rationally.
- To understand AI, we should know other related terms such as intelligence, knowledge, data, learning etc.

Philosophy of AI:

- The development of AI started with the intention of creating similar intelligence in machines that we find in humans.
  - **Intelligence:** Ability to learn, understand and think in a logical way about things.
  - **Knowledge:** Knowledge is awareness or understanding of someone or something, such as facts, information, descriptions or skills, which is acquired through experience or education.
  - **Data:** Information in raw or unorganized form.

2.2 APPLICATIONS OF AI

- Artificial Intelligence is progressing rapidly into diverse areas in modern society. AI can be used in several areas such as research in the medical field or creating innovative technology, for instance, autonomous vehicles and many more as shown in Fig. 2.1.

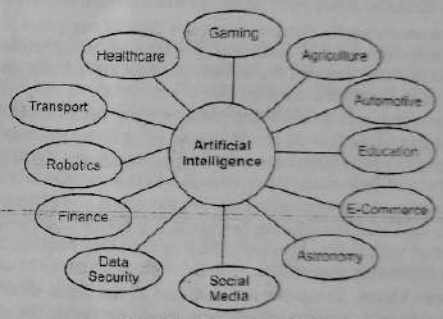


Fig. 2.1: Applications of AI

- The fields which are closely related with AI include Gaming, Robotics, Cognitive Science, Speech Recognition, NLP, Expert System, Machine Learning and Handwriting.
- Let us see these fields in detail:
  1. **Gaming:** AI plays a crucial role in strategic games such as Chess, Poker, Tic-Tac-Toe, etc., where machines can think of a large number of possible positions based on heuristic knowledge.
  2. **Robotics:** Robotics is a branch of AI, which is composed of Electrical Engineering, Mechanical Engineering and Computer Science for designing, construction and application of robots.
  3. **Cognitive Science:** It is the interdisciplinary and scientific study of human behavior and intelligence, with a focus on how information is perceived, processed and transformed.

4. **Speech Recognition:** Speech recognition is a technology that is empowered by AI to add convenience to its users. This new technology has the power to convert voice messages to text. And it also has the ability to recognize an individual based on their voice command.
5. **Natural Language Processing(NLP):** NLP is a subfield of AI which is a method of communicating with an intelligent system using a natural language such as English and Hindi etc. Major emphases of natural language processing include speech recognition, natural language understanding, and natural language generation.
6. **Expert System:** An Expert System is an intelligent computer program that can perform special and difficult tasks in some fields at the level of human experts. An expert system is typically designed to provide capabilities like those of a human expert when performing a task. Moreover, it can be used to drive vehicles, provide financial forecasts or do things those human experts do.
7. **Machine Learning:** Machine learning is a field of computer science that aims to teach computers how to learn and act without being explicitly programmed. Machine learning involves the construction of algorithms that familiarize their models to improve the ability to make predictions.
8. **Face Detection and Recognition:** Using virtual filters on our face when taking pictures and using face ID for unlocking our phones are two applications of AI. The former incorporates face detection meaning any human face is identified. The latter uses face recognition through which a specific face is recognized.
9. **E-Payments:** AI can be used to improve the speed and efficiency of the payment process. Artificial Intelligence has made it possible to deposit cheques from the comfort of your home. AI is proficient in decoding handwriting, making online cheque processing practicable.
10. **Computer Vision:** Computer vision is one of the fields of artificial intelligence that trains and enables computers to understand the visual world. Face recognition programs used by banks, government, etc. Handwriting recognition, electronics and manufacturing inspection, photo interpretation, baggage inspection, reverse engineering to automatically construct a 3D geometric model are examples of Computer Vision.

2.3 ARTIFICIAL INTELLIGENCE, INTELLIGENT SYSTEMS, KNOWLEDGE-BASED SYSTEMS, AI TECHNIQUES

2.3.1 Intelligent Systems

- An intelligent system which is a synonym to the narrow AI, studies agents and machines that perform automation of a well-defined task using statistical and machine learning algorithms based on related domain knowledge.
- Intelligent systems have emerged in information technology as a type of system derived from successful applications of artificial intelligence.

- An intelligent system is a machine with an embedded, Internet-connected computer that has the capacity to gather and analyze data and communicate with other systems.
- Other criteria for intelligent systems include the capacity to learn from experience, security, connectivity, the ability to adapt according to current data and the capacity for remote monitoring and management.

2.3.2 Knowledge-based Systems

- A Knowledge-based Systems (KBS) is one of the major and important family members of AI groups. With the advancement in computing facilities and other resources, attention is now turning to more demanding tasks which require intelligence. A KBS can act as an expert on demand, anytime and anywhere.

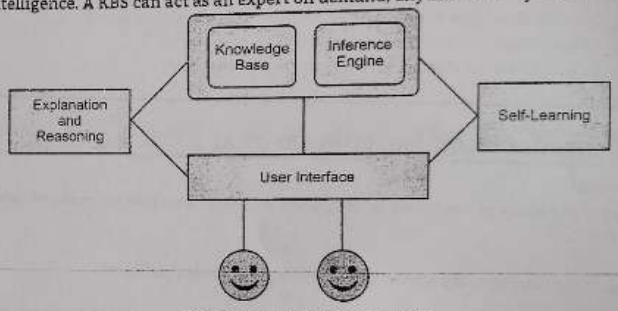


Fig. 2.2: General Structure of KBS

- It is a computer-based system that uses and generates knowledge from data, information and knowledge. These systems can understand the information being processed and can decide based on it.

Objectives of KBS:

1. Provides a high level of intelligence.
2. Provides a high amount of knowledge in different areas.
3. Solves social problems in a better way than the traditional computer-based information systems.
4. Offers significant software productivity improvement.
5. Significantly reduces the cost and time to develop computerized systems.

2.3.3 AI Techniques

- AI problems span a very broad field. They are hard but there are some techniques to solve these problems.
- AI Techniques depict how we represent, manipulate and reason with knowledge in order to solve problems.

- One of the few hard and fast results to come out of the first three decades of AI research is that intelligence requires knowledge.
- Some of the less desirable or suitable properties that knowledge possesses are:
  1. It is voluminous i.e. huge, or large to handle.
  2. It is hard to characterize or differentiate accurately.
  3. It is constantly or continuously changing.
  4. The way of organization of the data and its corresponding way of use is different.
- So, by examining above properties we are forced to conclude that-AI technique is a method to organize and use the knowledge efficiently in such a way that:
  1. The knowledge captures generalizations. We can group the situations that share some important properties without additional requirement of memory and updates. So, we usually call something without this property as "data" rather than knowledge.
  2. It can be understood by people who provide it because for many programs the bulk of data can be generated automatically.
  3. It can be easily modified to correct the errors and to reflect the changes in the world.
  4. Though it is not totally accurate or complete, it should be useful in many situations.
- By keeping above constraints in mind, AI techniques must be designed. There is some degree of independence between problems and problem-solving techniques. It is possible to solve AI problems without using AI techniques and it is possible to apply AI techniques to non-AI problems.
- Three important AI Techniques are as follows-
  1. **Search:** When no more direct approach is available as well as a framework in which any direct technique is available then search provides a way of solving problems. A search program finds a solution for a problem by trying various sequences of actions or operators until a solution is found.
  2. **Use of Knowledge:** The use of knowledge provides a way of solving complicated problems by manipulating the structures of the objects that are concerned.
  3. **Abstraction:** Abstraction finds a way of separating important features and notifications from the unimportant ones that would otherwise confuse any process.

2.4 EARLY WORK IN AI AND RELATED FIELDS

- **Year 1943**: The first work which is now recognized as AI was done by Warren McCulloch and Walter pits in 1943. They proposed a model of artificial neurons.
- **Year 1949**: Donald Hebb demonstrated an updating rule for modifying the connection strength between neurons. His rule is now called Hebbian learning.

- **Year 1950**: Alan Turing who was an English mathematician and pioneered Machine learning in 1950. Alan Turing published "Computing Machinery and Intelligence" in which he proposed a test. The test can check the machine's ability to exhibit intelligent behavior equivalent to human intelligence, called a Turing test.
- **Year 1955**: Allen Newell and Herbert A. Simon created the "first artificial intelligence program" Which was named as "Logic Theorist". This program has proved 38 of 52 Mathematics theorems and found new and more elegant proofs for some theorems.
- **Year 1956**: The word "Artificial Intelligence" first adopted by American Computer scientist John McCarthy at the Dartmouth Conference. For the first time, AI was coined as an academic field.
- **Year 1966**: The researchers emphasized developing algorithms which can solve mathematical problems. Joseph Weizenbaum created the first chatbot in 1966, which was named ELIZA.
- **Year 1972**: The first intelligent humanoid robot was built in Japan which was named as WABOT-1.
- The duration between the years **1974 to 1980** was the first AI winter duration. AI winter refers to the time where computer scientists dealt with a severe shortage of funding from the government for AI research. During AI winters, an interest of publicity on artificial intelligence was decreased.
- **Year 1980:** After AI winter duration, AI came back with "Expert System". Expert systems were programmed that emulate the decision making ability of a human expert.
- In 1980, the first national conference of the American Association of Artificial Intelligence was held at Stanford University.
- **Year 1987 to 1993**: The duration between the years 1987 to 1993 was the second AI Winter duration.
- Again, Investors and government stopped funding for AI research due to high cost but not efficient result. The expert system such as XCON was very cost effective.
- **Year 1997**: In the year 1997, IBM Deep Blue beats world chess champion Gary Kasparov and became the first computer to beat a world chess champion.
- **Year 2002**: AI entered the home in the form of Roomba, a vacuum cleaner.
- **Year 2006**: AI came in the Business world till the year 2006. Companies like Facebook, Twitter, and Netflix also started using AI.
- **Year 2011**: In the year 2011, Watson is a question-answering computer system capable of answering questions posed in natural language.
- **Year 2012**: Google has launched an Android app feature "Google now", which was able to provide information to the user as a prediction.
- **Year 2014**: In the year 2014, Chatbot "Eugene Goostman" won a competition in the infamous "Turing test."

- **Year 2018**: The "Project Debater" from IBM debated on complex topics with two master debaters and performed extremely well.

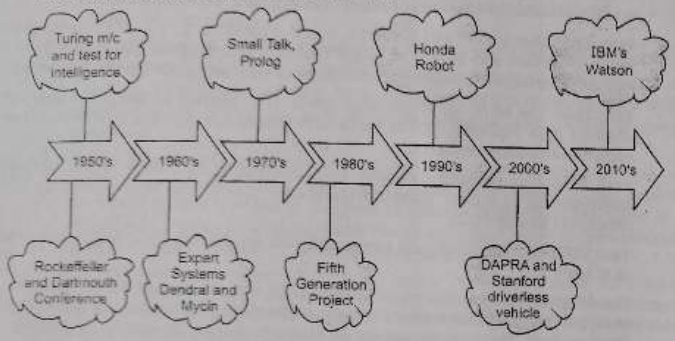


Fig. 2.3: History of AI

Sub-fields of Artificial Intelligence:

- AI now consists of many sub-fields, using a variety of techniques, such as:
  - **Machine Learning** – e.g. decision tree learning, version space learning.
  - **Neural Networks** – e.g. brain modelling, time series prediction, classification.
  - **Evolutionary Computation** – Evolutionary algorithms are motivated by biological evolution, and use mechanisms that replicate the evolutionary concepts of reproduction, mutation, recombination and selection. Evolutionary computation techniques can produce highly improved solutions in a wide range of problem settings. e.g. genetic algorithms, genetic programming.
  - **Computer Vision** – e.g. object recognition, image understanding.
  - **Robotics** – e.g. intelligent control, autonomous exploration.
  - **Expert Systems** – An Expert System is a program that is designed to solve the problems which requires human expertise or experience. e.g. decision support systems, teaching systems.
  - **Speech Processing** – e.g. speech recognition and production.
  - **Natural Language Processing** –e.g. machine translation.
  - **Planning** – Planning refers to the process of choosing/computing the correct sequence of steps to solve a given problem. e.g. scheduling, game playing.

2.5 DEFINING AI PROBLEMS AS A STATE SPACE SEARCH

2.5.1 A State Space

- A state space represents a problem in terms of states and operators that change states. A state space consists of:
  - A representation of the states the system can be in. For example, in a board game, the board represents the current state of the game.
  - A set of operators that can change one state into another state. In a board game, the operators are the legal moves from any given state. Often the operators are represented as programs that change a state representation to represent the new state.
  - An initial state.
  - A set of final states; some of these may be desirable, others undesirable. This set is often represented implicitly by a program that detects terminal states.

2.5.2 Definitions of Problem

- A problem is defined by its 'elements' and their 'relations'. To provide a formal description of a problem, we need to do the following:
  (a) Define a state space that contains all the possible configurations of the relevant objects, including some impossible ones.
  (b) Specify one or more states that describe possible situations, from which the problem-solving process may start. These states are called **initial states.**
  (c) Specify one or more states that would be an acceptable solution to the problem. These states are called **goal states.**

2.5.3 Problem Space

- A 'problem space' is an abstract space.
- A problem space encompasses all valid states that can be generated by the application of any combination of operators on any combination of objects.
- The problem space may contain one or more solutions. A solution is a combination of operations and objects that achieve the goals.

2.5.4 Problem Solving

- Problem solving is a process of generating solutions from observed data or given data.
- Problem solving has been the key area of concern for Artificial Intelligence.
- It is however not always possible to use direct methods (i.e., go directly from data to solution). Instead, problem solving often needs to use indirect or model-based methods.
- A 'problem' is characterized by:
  1. A set of goals.
  2. A set of objects.
  3. A set of operations.

- These could be well-defined and may evolve during problem solving.
- Problem solving may be characterized as a systematic search through a range of possible actions to reach some predefined goal or solution.
- To solve the problem of playing a game, we require the rules of the game and targets for winning as well as representing positions in the game. The opening position can be defined as the initial state and a winning position as a goal state. Moves from the initial state to other states leading to the goal state follows legally. The representation of games leads to a state space representation, and it is common for well-organized games with some structure.
- In short, to solve an AI problem, follow the steps as:
  1. Define the problem precisely i.e. specify the problem space, the operators for moving within the space and the initial and final state.
  2. Select one or more techniques for representing knowledge and for problem-solving and apply it to the problem.

2.5.5 Production System

- These systems were proposed by Emil Post in 1943 which is also known as Inferential Systems, Rule-Based System, or simply Production System.
- For describing and performing the search operation in AI programs it is useful to structure them. The process of solving the problem can be modeled as production System. If one adopts a system with production rules and a rule-interpreter, then the system known as Production System.
- Production System consists of:
  - Set of rules which are defined by the left side and right side of the system. The left side contains a set of things to watch for (condition), and the right side contains the things to do (action).
  - One or more knowledge databases that contain relevant information are for the given problem. Where some parts of the database may be permanent, others may temporary and only exist during the solution of the current problem.
  - A control strategy which determines the order of applying the rules to the database and offers a way of deciding any conflicts that can arise when various rules match at once.
  - The computational system called as Rule Applier which implements the control strategy and applies the rules.
- A production system in AI is a type of cognitive architecture that defines specific actions as per certain rules. The rules represent the declarative knowledge of a machine to respond according to different conditions.
- Today, many expert and automation methodologies rely on the rules of production systems. Below Fig. 2.4 shows the basic architecture of production systems in AI.

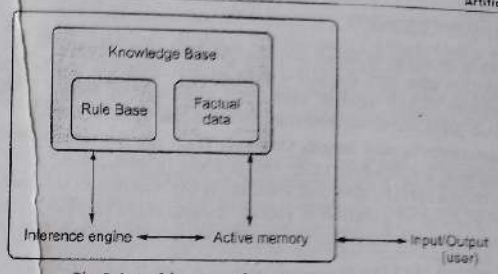


Fig. 2.4: Architecture of Production System in AI

2.6 SEARCH AND CONTROL STRATEGIES

2.6.1 Search Strategy

- The word 'search' refers to the search for a solution in a problem space.
- Search proceeds with different types of search control strategies. A strategy is defined by picking the order in which the nodes expand.
- So far, we have not given much attention to the question of how to decide which rule to apply next during the process of searching for a solution to a problem. This question arises when more than one rule will have its left side match the current state.
- In search method or technique, firstly select one option and leave other option. If this option is our final goal, then stop the search else we continue selecting, testing, and expanding until either a solution is found or no more states to be expanded.
- The depth-first search and breadth-first search are the two common search strategies.

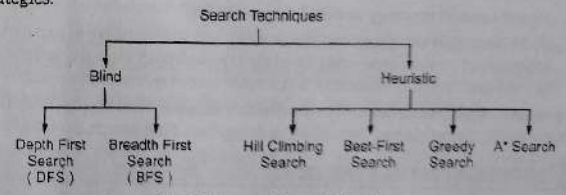


Fig. 2.5: Types of Search Techniques

- Basically there are two types of searches:
  - **Uninformed or Blind Search or Brute Force:** The uninformed or blind or brute-force search is the search methodology having no additional information about states beyond that provided in the problem definition. In this type of search

there is no order in which the solution paths are considered. So, blind search algorithm uses only the initial state, search operators and test for a solution. Examples of this kind of search are Breadth-First Search (BFS), Depth-First Search (DFS), Depth-Limited search (DLS) & Bidirectional Search.

- **Heuristic/Informed Search Strategy:** These are the search techniques where additional information about the problem is provided in order to guide the search in a specific direction. Heuristic is a rule of thumb that leads to a solution but provides no gurantee of success. Examples of this kind of search are Best-First Search, Hill Climbing, Constraint Satisfaction, Problem Reduction, etc.

2.6.2 Control Strategy

- Control strategies are also called as Search strategies. These are adopted for applying the rules and searching the problem solution in search space.
- To solve the problem, we must use a good control strategy which ultimately takes us to the goal state.
- The control strategy is responsible for obtaining the solution of the problem. Hence, if the wrong control strategy is applied, it may be possible that a solution is never obtained, even if it exists.

Features of good control strategies:

- The features of good control strategies are as follows:
  - **A good control strategy is that it causes motion.**
    Consider the water jug problem of the previous section. If we implemented a simple control strategy of starting each time at the top of the list of the rules and choosing the first applicable one, we would never solve the problem and indefinitely fill the 4-liter jug.
    So, control strategies that do not cause motion will never lead to a solution.
  - **A good control strategy should be systematic.**
    If the control strategy is not systematic, we may explore a particular useless sequence of operators several times before we finally find a solution. We should not use search space randomly, but it must be covered systematically.
- **Examples:** Widely used Control Strategies are Breadth-First Search(BFS), Depth-First Search(DFS), Generate and Test, Hill-Climbing, Best-first search, Problem Reduction.

2.7 PROBLEM CHARACTERISTICS

- A problem may have different aspects of representation and explanation. To choose the most appropriate method for a particular problem, it is necessary to analyze the problem along several key dimensions.

Features of a Problem:

- Some of the main key features of a problem are given below:
  1. Is the problem decomposable into a set of sub problems?
  2. Can the solution steps be ignored or undone?

  3. Is the problem universally predictable?
  4. Is the desired solution a state of the world or a path to a state?
  5. Is the knowledge base consistent?
  6. What is the role of Knowledge?
  7. Will the solution of the problem require interaction between the computer and the problem solver?
- Let's have a look one by one characteristic.

1. Is the problem decomposable?

- A very large and composite problem can be easily solved if it can be broken into smaller problems and recursion could be used. Suppose we want to solve problem

Example: $\int x^2 + 3x + \sin^2 x \cos^2 x \, dx$

- This can be done by breaking it into three smaller problems and solving each by applying specific rules. Adding the results, the complete solution is obtained.

2. Can the solution steps be ignored or undone?

- Problems fall under three classes ignorable, recoverable and irrecoverable. This classification is with reference to the steps of the solution to a problem.

(i) **Ignorable problems:**

Example: In theorem proving, solution steps can be ignored.

In the theorem proving, we may later find that it is of no help. We can still proceed further since nothing is lost by this redundant step. This is an example of ignorable solutions steps.

(ii) **Recoverable problems:**

Example: In 8-Puzzle, solution steps can be undone.

In the 8-Puzzle problem, tray and arrange it in specified order. While moving from the start state towards goal state, we may make some stupid move and consider theorem proving. We may proceed by first proving the lemma. But we may backtrack and undo the unwanted move. This only involves additional steps, and the solution steps are recoverable.

(iii) **Irrecoverable problems:**

Example: In the game of Chess, solution steps can't be undone.

In the game of Chess, if a wrong move is made, it can neither be ignored nor be recovered. The thing to do is to make the best use of the current situation and proceed. This is an example of an irrecoverable solution steps.

3. Is the problem universally predictable?

- Problems can be classified into those with certain outcomes (8-Puzzle and Water Jug problems) and those with uncertain outcomes (playing cards).
- In certain outcome problems, planning could be done to generate a sequence of operators that guarantees to lead to a solution. Planning helps to avoid unwanted solution steps.

- For uncertain outcome problems, planning can at best generate a sequence of operators that has a good probability of leading to a solution. The uncertain outcome problems do not guarantee a solution and it is often very expensive since the number of solutions paths to be explored increases exponentially with the number of points at which the outcome cannot be predicted.
- Thus, one of the hardest types of problems to solve is the irrecoverable, uncertain outcome problems (Example: Playing cards).

4. **Is a good solution absolute or relative? Is the solution a state or a path?**
- There are two categories of problems: simple and complex problems.
  - In simple problems, like water jug and 8 puzzle problems, we are satisfied with the solution, we are aware of the solution path taken, whereas in the complex problem not just any solution is acceptable.
  - We want the best, like that of traveling salesman problem, where it is the shortest path. In any path problems, by heuristic methods we obtain a solution and we do not explore alternatives. For the best-path problems all possible paths are explored using an exhaustive search until the best path is obtained.

5. **Is the solution a state or a path?**
- Consider the water jug problem, the final state we get i.e. (2, 0) for this type of the problem path is important which takes to that state. So, the solution of the water jug problem must be a sequence of operations that produces the final state i.e., problems whose solution is path to state.
- Now, consider the word "Bank", which has two interpretations.

(a) **Bank:** Financial Institution

(b) **Bank:** Side of river

- Here, we should decide the meaning which is suitable for the whole sentence.
- But to solve the problem of finding the interpretation we need to produce only the interpretation itself. No record of the processing by which interpretation was found is necessary. So, this problem solution is a state of the world.
- By these two examples, we can say that some problems have solutions in states and some are having path to that state.

6. **Is the knowledge base consistent?**
- In some problems the knowledge base is consistent and in some it is not. For example, consider the case when a Boolean expression is evaluated. The knowledge base now contains theorems and laws of Boolean Algebra which are always true.
- On the contrary, consider a knowledge base that contains facts about production and cost. These keep varying with time. Hence many reasoning schemes that work well in consistent domains are not appropriate in inconsistent domains.

7. **What is the role of Knowledge?**
- Though one could have unlimited computing power, the size of the knowledge base available for solving the problem does matter in arriving at a good solution.

- For example, the game of playing chess, just the rules for determining legal moves and some simple control mechanisms are sufficient to arrive at a solution. But additional knowledge about good strategy and tactics could help to constrain the search and speed up the execution of the program. The solution would be then realistic.
- Consider the case of predicting the political trend. This would require an enormous amount of knowledge even to be able to recognize a solution, leave alone the best.
- These two examples illustrate the difference between problems for which a lot of knowledge is important only to constrain the search for a solution and those for which a lot of knowledge is required even to be able to recognize a solution.

Examples: 1. Playing chess, 2. Newspaper understanding.

8. **Will the solution of the problem require interaction between the computer and the problem solver?**
- The problems can again be categorized under two heads:
  1. Solitary in which the computer will be given a problem description and will produce an answer, with no intermediate communication and with the demand for an explanation of the reasoning process. Simple Theorem Proving falls under this category. By giving the basic rules and laws, the theorem could be proved, if one exists.

     Example: Theorem Proving (give basic rules and laws to computer)
  2. Conversational in which there will be intermediate communication between a person and the computer, either to provide additional assistance to the computer or to provide additional informed information to the user, or both problems such as Medical diagnosis fall under this category. In this problem, people will be unwilling to accept the decision of the program if they cannot follow its reasoning.

     Example: Problems such as Medical Diagnosis.

2.8 AI PROBLEM: WATER JUG PROBLEM, TOWER OF HANOI, MISSIONARIES AND CANNIBAL PROBLEM

2.8.1 State Space Representation of Water Jug Problem

Definition:

- In this problem, we use two jugs called four liter and three liters; four holds a maximum of four liters of water and three a maximum of three liters of water. There is a pump that can be used to fill the jugs with water. How can we get two liters of water in the four-liter jug?

State Space Representation:

- The state space is a set of prearranged pairs giving the number of liters of water in the pair of jugs at any time, i.e., (x, y) where,

  x = 0, 1, 2, 3 or 4.

  y = 0, 1, 2 or 3.

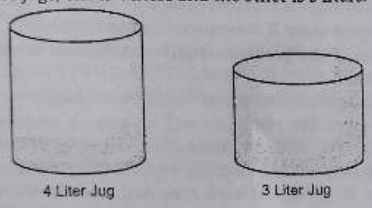
Where,

x - Represents the number of liters of water in the 4-liter jug.

y - Represents the number of liters of water in the 3-liter jug.

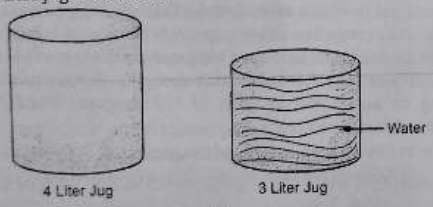
- The start state is (0, 0) and the goal state is (2, n), where n may be any, but it is limited to three holding from 0 to 3 liters of water or empty.
- x and y show the name and numerical number shows the amount of water in jugs for solving the water jug problem.
- Following fig. 2.5 shows how to solve the problem.

Step 1: We have two jugs; one is 4 liters and the other is 3 liters.



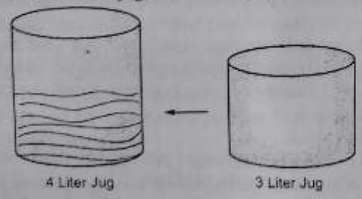
(a)

Step 2: Fill a 3 liter jug with water.



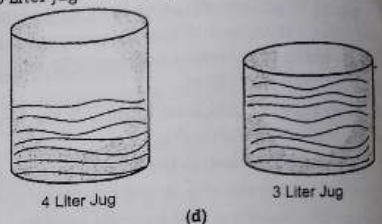
(b)

Step 3: Pour water from a 3 liter jug into a 4 liter jug.



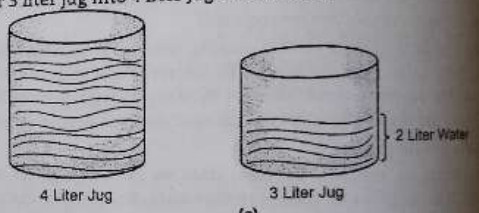
(c)

Step 4: Fill again 3 Liter jug with water.



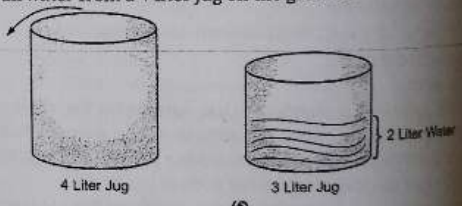
(d)

Step 5: Pour 3 liter jug into 4 liter jug till it is filled.



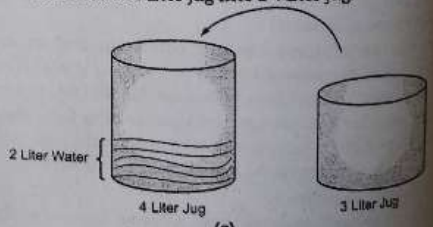
(e)

Step 6: Pour all water from a 4 liter jug on the ground.



(f)

Step 7: Pour the water from a 3 liter jug into a 4 liter jug.



(g)

Fig. 2.5: Water Jug Problem

- So, the result of the water jug problem is as shown above.
- Table 2.1 shows Production Rules of the Water Jug Problem. In this table, the left side shows rules or conditions that must be satisfied before the operator described by the rule can be applied.
- The major production rules for solving this problem are shown below:

Table 2.1: Production Rules for Water Jug Problem

Sr. No.	Current State	Next State	Descriptions
1.	(x, y) if x < 4	(4, y)	Fill the 4 liter jug
2.	(x, y) if y < 3	(x, 3)	Fill the 3 liter jug
3.	(x, y) if x > 0	(x – d, y)	Pour some water out of the 4 liter jug
4.	(x, y) if y > 0	(x, y – d)	Pour some water out of the 3 liter jug
5.	(x, y) if y > 0	(0, y)	Empty the 4 liter jug
6.	(x, y) if y > 0	(x, 0)	Empty the 3 liter jug on the ground
7.	(x, y) if x + y >= 4 and y > 0	(4, y – (4 – x))	Pour water from the 3 liter jug into the 4 liter jug until the 4 liter jug is full.
8.	(x, y) if x + y >= 3 and x > 0	(x – (3 – y), 3)	Pour water from the 4 liter jug into the 3 liter jug until the 3 liter jug is full.
9.	(x, y) if x + y <= 4 and y > 0	(x + y, 0)	Pour all the water from the 3 liter jug into the 4 liter jug.
10.	(x, y) if x + y <= 3 and x > 0	(0, x + y)	Pour all the water from the 4 liter jug into the 3 liter jug
11.	(0, 2)	(2, 0)	Pour the 2 gallons from 3 liter jug into the 4 liter jug.
12.	(2, y)	(0, y)	Empty the 2 gallons in the 3 liter jug on the ground

- For this problem, there are several sequences of operators that solve the problem, one such sequence is shown in Table 2.2.

Table 2.2: One solution to the Water Jug Problem

Sr. No.	4 liter jug contents	3 liter jug contents	Rule Applied
1.	0	0	–
2.	0	3	2
3.	3	0	9
4.	3	3	2
5.	4	2	7
6.	0	2	5 or 12
7.	2	0	9 or 11

- The problem is solved by using the production rules in combination with an appropriate control strategy, moving through the problem space until a path from an initial state to a goal state is found.
- To provide a formal description of a problem we must do the following:
  1. Define a state space that contains all the possible configuration of the relevant objects.
  2. Specify one or more states within that space that describe possible situations from which the problem-solving process starts called the initial state.
  3. Specify one or more states that would be acceptable as a solution to the problem which is called the goal state.
  4. Set of rules that describe the actions available.
- The problem can be solved by using the rules, in combination with an appropriate control strategy.

2.8.2 State Space Representation of Tower of Hanoi

Definition:

- Tower of Hanoi is a mathematical game puzzle where we have three pillars and n numbers of disks.

Rules of game:

- This game has following rules :
  1. Only one disk will move at a time.
  2. The larger disk should always be on the bottom and the smaller disk on top of it (Even during intermediate move).
  3. Move only the uppermost disk.
  4. All disks move to the destination pillar from the source pillar.
- Three pillars say A, B and C and 3 disks are given in the problem as shown in Fig. 2.6. Initially all disks on pillar A. Finally, we want all 3 disks on pillar C using pillar B as shown in Fig. 2.7.

Initial State:

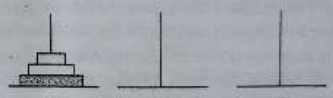


Fig. 2.6

Final State:

Fig. 2.7

Solution:

Disk 1 moved from A to C
Disk 2 moved from A to B
Disk 1 moved from C to B
Disk 3 moved from A to C
Disk 1 moved from B to A
Disk 2 moved from B to C
Disk 1 moved from A to C

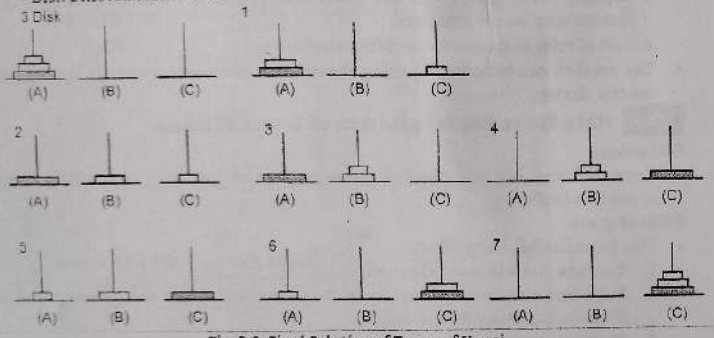


Fig. 2.8: Final Solution of Tower of Hanoi

2.8.3 State Space Representation of Missionary Cannibal Problem

The Missionaries and Cannibals Problem Statement:

- *"Three missionaries and three cannibals are present at one side of a river and need to cross the river. There is only one boat available. At any point of time, the number of cannibals should not outnumber the number of missionaries at that bank. It is also known that only two persons can occupy the boat available at a time".*
- The objective of the solution is to find the sequence of their transfer from one bank of river to another using the boat sailing through the river satisfying these constraints.

Production Rules:

- We can form various production rules.
- Let Missionary be denoted by 'M' and Cannibal, by 'C'. These rules are described below:

Rule 1: (0, M) : One missionary sailing the boat from bank-1 to bank-2
Rule 2: (M, 0) : One missionary sailing the boat from bank-2 to bank-1
Rule 3: (M, M) : Two missionaries sailing the boat from bank-1 to bank-2

Rule 4: (M, M) : Two missionaries sailing the boat from bank-2 to bank-1
Rule 5: (M, C) : One missionary and one Cannibal sailing the boat from bank-1 to bank-2
Rule 6: (C, M) : One missionary and one Cannibal sailing the boat from bank-2 to bank-1
Rule 7: (C, C) : Two Cannibals sailing the boat from bank-1 to bank-2
Rule 8: (C, C) : Two Cannibals sailing the boat from bank-2 to bank-1
Rule 9: (0, C) : One Cannibal sailing the boat from bank-1 to bank-2
Rule 10: (C, 0) : One Cannibal sailing the boat from bank-2 to bank-1

- All or some of these production rules will have to be used in a particular sequence to find the solution of the problem. The rules applied and their sequence is presented in the following Table 2.3.

Table 2.3: Missionaries and Cannibals problem-Rules applied and their sequence

After application of Rule	Persons in the river bank 1	Persons in the river bank 2	Boat Position
Start state	M, M, M, C, C, C	0	bank 1
5	M, M, C, C	M, C	bank 2
2	M, M, C, C, M	C	bank 1
7	M, M, M	C, C, C	bank 2
10	M, M, M, C	C, C	bank 1
3	M, C	C, C, M, M	bank 2
6	M, C, C, M	C, M	bank 1
3	C, C	C, M, M, M	bank 2
10	C, C, C	M, M, M	bank 1
7	C	M, M, M, C, C	bank 2
10	C, C	M, M, M, C	bank 1
7	0	M, M, M, C, C, C	bank 2

Summary

- "Artificial Intelligence (AI) is an area of computer science that involves building smart machines that are able to perform tasks which usually require human intelligence.
- The fields which are closely related with AI include Gaming, Robotics, Cognitive Science, Speech Recognition, NLP, Expert System, Machine Learning and Handwriting.
- An intelligent system is a machine with an embedded, Internet-connected computer that has the capacity to gather and analyze data and communicate with other systems.

- KBS is a computer-based system that uses and generates knowledge from data, information and knowledge.
- Solution to any problem is the collection of such different states and set of operations. This collection of states is termed as State Space. Each of these states is achieved using the application of operations to the previous state.

Check Your Understanding

1. _____ provides the frameworks into which more direct methods for solving sub-parts of a problem can be embedded.
   (a) Search (b) Problem
   (c) State (d) State Space
2. A _____ is a representation of problem elements at a given moment.
   (a) Search (b) Problem
   (c) State (d) State Space
3. A production system consists of _____.
   (i) A set of rules.
   (ii) One or more databases.
   (iii) A Control Strategy
   (a) (i) and (ii) only (b) (ii) and (iii) only
   (c) (i) and (iii) only (d) All (i), (ii) and (iii)
4. _____ helps us to decide which rule to apply next during the process of searching for a solution to a problem.
   (a) Control strategies (b) Production system
   (c) Problem (d) State space
5. _____ is the computational system that implements the control strategy and applies the rules.
   (a) A set of rules (b) A control strategy
   (c) One or more knowledge (d) A rule applier
6. Which of the following are the benefits of the production system?
   (i) Production systems provide an excellent tool for structuring AI programs.
   (ii) The individual rules can be added, removed, or modified independently.
   (iii) The production rules are expressed in a natural form.
   (a) (i) and (ii) only (b) (ii) and (iii) only
   (c) (i) and (iii) only (d) All (i), (ii) and (iii)
7. State whether the following statements about the state space are True.
   (i) A state space forms a graph in which the nodes are states and the arch between nodes is actions.
   (ii) In state space, a path is a sequence of states connected by a sequence of actions.
   (a) (i) only (b) (ii) only
   (c) Both (i) and (ii) (d) None of the above

8. State whether the following statements about defining the problem are True or False.
   (i) A problem will define a state space that contains all the possible configurations of relevant objects.
   (ii) A problem will specify a set of rules that describe the actions available.
   (a) (i) True, (ii) False (b) (i) False, (ii) True
   (c) (i) True, (ii) True (d) (i) False, (ii) False
9. The application/applications of Artificial Intelligence is/are _____.
   (a) Expert Systems (b) Gaming
   (c) Vision Systems (d) All of the above
10. Production rules are important in the _____ Problem.
    (a) Water Jug (b) Towers of Hanoi
    (c) Missionary and Cannibal (d) Producers and Consumers

Answers

1. (a)	2. (c)	3. (d)	4. (a)	5. (d)	6. (d)	7. (c)	8. (c)	9. (d)	10. (a)

Practice Questions

Q.I Answer the following questions in short.

1. Define Search strategy.
2. What is AI? Explain with its applications.
3. State the things required to be considered when we want to build an AI system that is used to solve a particular problem.
4. Explain History of AI.
5. Which are techniques of AI?

Q.II Answer the following questions.

1. State 4 components using which problem can be well formulated.
2. Give state space representation for "Water Jug Problem".
3. Explain the production system in detail.
4. Discuss problem characteristics in detail.
5. Give state space representation for "Missionary Cannibal Problem".
6. Give state space representation for "Tower of Hanoi Problem".

Q.III Define the terms.

1. Problem Space
2. Ignorable Problem
3. Recoverable Problem
4. Irrecoverable Problem
5. Expert System

■■■

3... AI Search Techniques

Learning Objectives ...

- To know about different Uninformed Search strategies.
- To know about the Informed strategies.
- To know about the different Heuristic Search Techniques.
- To apply search algorithms to real-world problems.

3.1 INTRODUCTION

- Search is one of the operational tasks that characterize AI programs best. To perform prescribed functions almost every AI program depends on a search procedure. Typically, problems are defined in terms of states, and solutions correspond to a goal state. Different search techniques are available that we will describe here.

3.2 BLIND SEARCH TECHNIQUES

- Uninformed search or Blind search is the search methodology having no additional information about states beyond that provided in the problem definitions. In this search total search space is looked for solutions.
- The different types of search algorithms are as follows:
  - Breadth First Search
  - Depth First Search

3.2.1 BFS (Breadth-first Search)

- Breadth First searches are performed by exploring all nodes at a given depth before proceeding to the next level. This means that all immediate children of nodes are explored before any children's children are considered.
- Construct a tree with the initial state as its root. Generate all its successors by applying all the rules that are appropriate. Fig. 3.1 shows how the tree looks at this point. Now for each leaf node, generate all its successors by applying appropriate rules. The tree at this point is shown in Fig. 3.2. Continue this process until some rule produces a goal state. This process is called Breadth First Search.

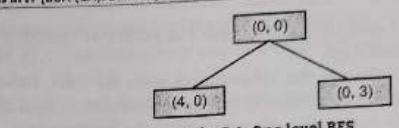


Fig. 3.1: One level BFS

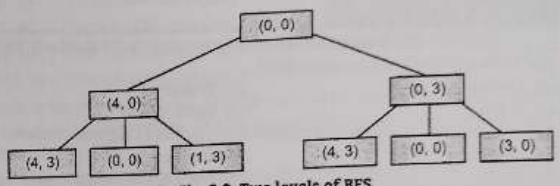


Fig. 3.2: Two levels of BFS

Algorithm: Breadth-first Search

1. Create a variable called NODE-LIST and set it to the initial state.
2. Until goal state is found or NODE-LIST is empty:
   (a) Remove the first element from NODE-LIST and call it E. If NODE-LIST was empty, quit.
   (b) For each way that each rule can match the state described in E do:
      (i) Apply the rule to generate a new state.
      (ii) If the new state is a goal state, quit and return to this state.
      (iii) Otherwise, add the new state to the end of NODE-LIST.

Advantages of Breadth-First Search:

1. BFS will not get trapped explaining a blind path which happens in depth first search.
2. If there is a solution then Breadth-First Search is guaranteed to find it out.
3. If there are multiple solutions then Breadth-First Search can find minimal solution i.e. one that requires the minimum number of steps, will be found.

Disadvantage of Breadth First Search:

1. High storage requirement exponential with tree depth. A BFS on a binary tree generally requires more memory than a DFS.
2. When the search space is large the search performance will be poor compared to other heuristic searches.

3.2.2 Depth First Search

- Depth first searches are performed by going downward into a tree as early as possible.

- Consider a single branch of the tree until it produces a solution or until a decision to terminate the path is made. It makes sense to terminate a path if it reaches a dead end, produces a previous state or becomes longer than some limit, in such cases backtracking occurs. To overcome such backtracking is known as Depth First Search.
- Following Fig. 3.3 shows Depth First Search tree for Water Jug Problem.

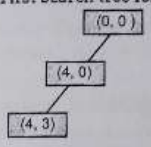


Fig. 3.3: Depth First Search Tree

Algorithm of Depth First Search:

1. If the initial state is a goal state, quit and return success.
2. Otherwise, do the following until success or failure is signaled:
   (a) Generate a successor, E, of the initial state. If there are no more successors, signal failure.
   (b) Call depth first search with E as the initial state.
   (c) If success is returned, signal success. Otherwise continue in this loop.

Advantages of Depth First Search:

1. Depth First Search requires less memory since only the nodes on the current path are stored.
2. If Depth First Search finds a solution without examining much of the search space at all. This is particularly significant if many acceptable solutions exist. Depth First Search can stop when one of them is found.
3. To solve simple problems like water jug problem, we are using those control strategies that cause motion and are systematic which will lead to the final state. But for solving complex problems, we need efficient control structure.

Disadvantages of Depth First Search:

1. May find a sub-optimal solution (one that is deeper or more costly than the best solution).
2. Incomplete without a depth bound. It may not find a solution even if one exists.

3.2.3 Depth Limited Search (DLS)

- Depth limited search is the new search algorithm for uninformed search. The unbounded tree problem happens to appear in the depth first search algorithm, and it can be fixed by imposing a boundary or a limit to the depth of the search domain.
- The Depth Limited Search (DLS) method is almost equal to Depth First Search (DFS). But DLS can work on the infinite state space problem because it bounds the depth of the search tree with a predetermined limit L. Nodes at this depth limit are treated as if they had no successors.

Failure conditions of DLS:

- Depth limited search can be terminated with two conditions of failure:
  - **Standard failure value:** It indicates that the problem does not have any solution.
  - **Cutoff failure value:** It defines no solution for the problem within a given depth limit.

Advantages of Depth Limited Search:

1. Depth limited search is better than DFS and requires less time and memory space.
2. DFS assures that the solution will be found if it exists infinite time.
3. There are applications of DLS in graph theory particularly similar to the DFS.
4. To struggle the disadvantages of DFS, we add a limit to the depth, and our search strategy performs recursively down the search tree.

Disadvantages of Depth Limited Search:

1. The depth limit is compulsory for this algorithm to execute.
2. The goal node may not exist in the depth limit set earlier, which will push the user to iterate further adding execution time.
3. The goal node will not be found if it does not exist in the desired limit.

3.2.4 Iterative Deepening Search

- The iterative deepening algorithm is a combination of DFS and BFS algorithms. This search algorithm finds out the best depth limit and does it by gradually increasing the limit until a goal is found.
- This algorithm performs Depth First Search up to a certain "depth limit", and it keeps increasing the depth limit after each iteration until the goal node is found.
- This Search algorithm combines the benefits of Breadth First Search's fast search and Depth First Search's memory efficiency.
- The iterative search algorithm is useful for uninformed search when the search space is large, and depth of the goal node is unknown.

Advantage:

1. It combines the benefits of BFS and DFS search algorithms in terms of fast search and memory efficiency.

Disadvantage:

1. The main drawback of IDDFS is that it repeats all the work of the previous phase.

Example:

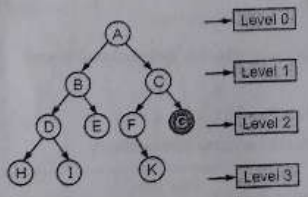


Fig. 3.4: Iterative Deepening Search

- First Iteration → A
- Second Iteration → A, B, C
- Third Iteration → A, B, D, E, C, F, G
- Fourth Iteration → A, B, D, H, I, E, C, F, K, G
- In the fourth iteration, the algorithm will find the goal node.

3.2.5 Bidirectional Search

- Bidirectional search algorithm runs two simultaneous searches, one from initial state called as forward-search and other from goal node called as backward-search, to find the goal node.
- Bidirectional search replaces one single search graph with two small subgraphs in which one starts the search from an initial vertex and other starts from goal vertex. The search stops when these two graphs intersect each other.
- Bidirectional search can use search techniques such as BFS, DFS, DLS, etc.
- In the below search tree, a bidirectional search algorithm is applied. This algorithm divides one graph/tree into two sub-graphs. It starts traversing from node 1 in the forward direction and starts from goal node 16 in the backward direction.
- The algorithm terminates at node 9 where two searches meet.

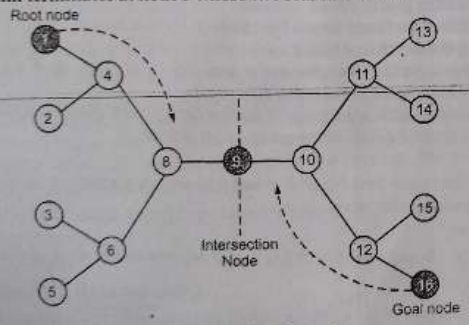


Fig. 3.5: Bidirectional Search

Advantages:

1. Bidirectional search is fast.
2. Bidirectional search requires less memory.

Disadvantages:

1. Implementation of the bidirectional search tree is difficult.
2. In bidirectional search, one should know the goal state in advance.

3.2.6 Uniform Cost Search

- Uniform Cost Search is a searching algorithm used for traversing a weighted tree graph. This algorithm comes into play when a different cost is available for ea edge.
- The primary goal of the Uniform Cost Search is to find a path to the goal node wh has the lowest cumulative cost. Uniform Cost search expands nodes according their path costs from the root node. It can be used to solve any graph/tree where t optimal cost is in demand.
- Uniform Cost Search algorithm is implemented by the priority queue. It giv maximum priority to the lowest cumulative cost. Uniform Cost Search is equivale to the BFS algorithm if the path cost of all edges is the same.

Algorithm:

1. Insert root node into the queue.
2. Repeat till queue is not empty:
   (a) Remove the next element with the highest priority from the queue.
   (b) If the node is a destination node, then print the cost and the path and exit. Else insert all the children of removed elements into the queue with th cumulative cost as their priorities.

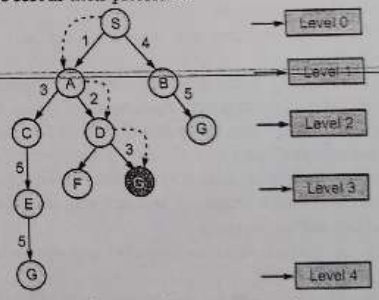


Fig. 3.6: Uniform Cost Search

Advantages:

1. Uniform Cost Search is optimal because at every state the path with the least c is chosen.

Disadvantages:

1. It does not care about the number of steps involved in searching and is o concerned about path cost. Due to which this algorithm may be stuck in infinite loop.

3.3 HEURISTIC SEARCH TECHNIQUES

- These are the search techniques where additional information about the problem is provided in order to guide the search in a specific direction.
- When we need to solve hard problems, it often becomes necessary to compromise with requirements of mobility and systematicity. We need to construct a control structure that is no longer guaranteed to find the best answer but almost always find the good answer.
- To achieve that, we introduce the concept of Heuristic. A heuristic comprises of a technique that improves the efficiency of a search process by sacrificing claims of completeness.
- Using good heuristics we can get good solutions to hard problems.
- We will study the following search techniques.
  - Generate-and-test
  - Simple Hill Climbing
  - Best First Search
  - Constraint Satisfaction
  - Means End Analysis
  - A* and AO*

3.3.1 Generate-and-Test

- The generate-and-test is the simplest approach among all the algorithms. Usually it helps in finding out the solution but not always.
- Generate-and-Test Search is a heuristic search technique based on Depth-First Search with Backtracking which guarantees to find a solution if done systematically and there exists a solution. In this technique, all the solutions are generated and tested for the best solution. It ensures that the best solution is checked against all possible generated solutions.
- This algorithm works in two modules:

  (a) Generator Module: It creates the possible solution.

  (b) Tester Module: It tests or evaluates each of the proposed solution either accepting or rejecting the solution.
- Here, an action may stop when one acceptable solution is found or action may continue until all possible solutions are found.

Algorithm of Generate-and-Test:

- The algorithm is as follows:

  Step 1 : Generate a possible solution. For example, generating a particular point in the problem space or generating a path from a start state.

  Step 2 : Test to see if this is an actual solution by comparing the chosen point or the endpoint of the chosen path to the set of acceptable goal states.

  Step 3 : If a solution is found, quit. Otherwise go to Step 1.

- Diagrammatic representation is as shown in Fig. 3.7.

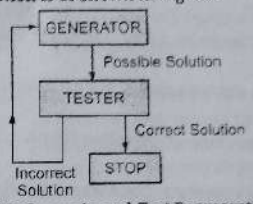


Fig. 3.7: Generate-and-Test Representation

- This algorithm is basically a DFS procedure only as complete solutions must be created before testing. When it is used in systematic form then it is simply an exhaustive search of the problem space. It is often called the ***British Museum method*** as it can operate by generating solutions randomly. However, in this method there will be no guarantee that a solution will ever be found. So, we need heuristic to sharpen the method.

3.3.2 Simple Hill Climbing

- Hill Climbing is a form of heuristic search algorithm which is used in solving optimization related problems in Artificial Intelligence domain.
- Simple Hill Climbing is the simplest form of the Hill Climbing Algorithm. It is so called because of the way the nodes are selected for expansion. In the search path at each point, the successor node that appears to lead most quickly to the top of the hill is selected for exploration. Hill Climbing is a variation of the DFS [Generate-and-Test] algorithm in which feedback is used to decide in which direction to move in search space.
- Generally, in DFS the test function answers in terms of yes or no but in Hill Climbing the test function provides a **heuristic function** f[n,g] a function of the nodes n and/or goals g, which gives us an estimate how close a given state is to a goal state.
- This is a local search method in which we consider a state space landscape as shown in Fig. 3.8.

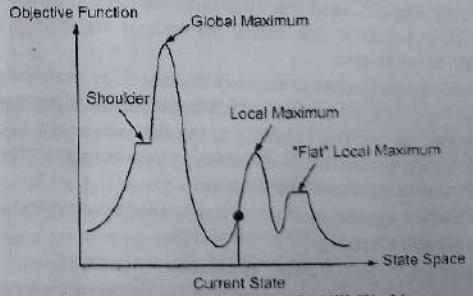


Fig. 3.8: State Space landscape for Hill Climbing

- State space diagram is a graphical representation of the set of states our search algorithm can reach vs the value of our Objective Function (the function which we wish to maximize).
  - **X-axis:** Denotes the state space i.e. states or configuration our algorithm may reach.
  - **Y-axis:** Denotes the values of objective function corresponding to a particular state.
- The best solution will be that state space where the objective function has maximum value (Global Maximum).

Different regions in the State Space Diagram:

1. **Local Maximum:** It is a state which is better than its neighboring state however there exists a state which is better than it (Global Maximum). This state is better because here the value of the objective function is higher than its neighbors.
2. **Global Maximum:** It is the best possible state in the state space diagram. This is because in this state, the objective function has the highest value.
3. **Plateau/flat local maximum:** It is a flat region of state space where neighboring states have the same value.
4. **Ridge:** It is a region which is higher than its neighbors but itself has a slope. It is a special kind of local maximum.
5. **Current State:** The region of state space diagram where we are currently present during the search.
6. **Shoulder:** It is a plateau that has an uphill edge.

- The landscape has both location defined by the state and elevation defined by the value of heuristic cost function or objective function.
- The aim is to find global minimum i.e. lowest valley, if the elevation corresponds to the objective function then the aim is to find global maximum.
- Local Search algorithms explore this landscape. A complete local search algorithm always finds a goal if one exists. An optimal algorithm always finds a global minimum or maximum.
- Hill Climbing search tries to improve the search by modifying the current state as shown in Fig. 3.8 by arrow. So, a search tree is generated using a heuristic function.
- As explained earlier Hill Climbing is like DFS where the most promising child is selected for expansion. When the children have been generated, alternative choices are evaluated using a heuristic function.
- The path which appears most promising has been chosen next.

Algorithm for Hill Climbing:

- The algorithm for Hill Climbing as follows:

Step 1 : Evaluate the initial state. If it is a goal state then stop and return success. Otherwise, make the initial state the current state.

Step 2 : Loop until the solution state is found or there are no new operators present which can be applied to the current state.

(a) Select a state that has not been yet applied to the current state and apply it to produce a new state.

(b) Perform these to evaluate new state.

(i) If the current state is a goal state, then stop and return to success.

(ii) If it is better than the current state, then make it the current state and proceed further.

(iii) If it is not better than the current state, then continue in the loop until a solution is found.

Step 3 : Exit

Steepest Ascent Hill climbing Algorithm:

- It first examines all the neighboring nodes and then selects the node closest to the solution state as the next node. It is called as Steepest Ascent Hill Climbing or gradient search.
- The algorithm for Steepest Ascent Hill Climbing as follows:

Step 1 : Evaluate the initial state, if it is goal state then return success and stop, else make the current state as the initial state.

Step 2 : Loop until a solution is found or the current state does not change.

(a) Let SUCC be a state such that any successor of the current state will be better than it.

(b) For each operator that applies to the current state:

(i) Apply the new operator and generate a new state.

(ii) Evaluate the new state.

(iii) If it is a goal state, then return it and quit, else compare it to the SUCC.

(iv) If it is better than SUCC, then set a new state as SUCC.

(v) If the SUCC is better than the current state, then set the current state to SUCC.

Step 3 : Exit

The Problems or Disadvantages in Hill Climbing:

1. **Local Maximum:** It is a state which is better than all of its neighbors but it is not better than some other states which are farther away. At local maxima all moves appear to make things worse as shown in Fig. 3.9.

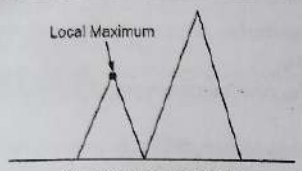


Fig. 3.9: Local Maximum

Solution to the above problem:

- One possible solution is backtracking. We can backtrack to the earlier node and try to go in different directions to attain the global peak.

2. **Plateau:** It is a flat area of the search space in which all neighboring states [nodes] have the same value. Actually plateau is an area of the state space landscape where the evaluation function is flat as shown in Fig. 3.10.

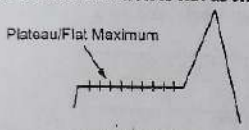


Fig. 3.10: Plateau

Solution to the above problem:

- A big jump-in some direction can be done in order to get to a new section of a search space. This method is recommended as in plateau all neighboring points have the same value.
- Another solution is to apply small steps several times in the same direction which depends upon the available rules.

3. **Ridges:** It is a special kind of local maximum. It is an area of the search space which is higher than surrounding areas and that itself has a slope as shown in Fig. 3.11.

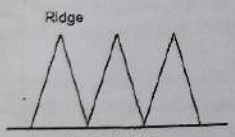


Fig. 3.11: Ridge

We can not travel the ridge by single moves as the orientation of the high region compared to the set of available moves makes it possible.

Solution to the above problem:

- Trying different paths at the same time is a solution.
- Bidirectional search can be useful here.

3.3.3 Best-First Search

- Best-First Search is a way of combining the advantages of both Depth and Breadth First Search.
- We will call a graph an **OR - graph**, since each of its branches represents alternative problem solving path. The Best-First Search selects the most promising of the nodes we have generated so far. This can be achieved by applying appropriate **Heuristic function** to each of them. At any point in the search process best first moves forward from the most promising of all the nodes generated so far.
- The Best-First Process is illustrated in Fig. 3.12 where numbers by the nodes may be regarded as estimates of the distance or cost to reach the goal node.

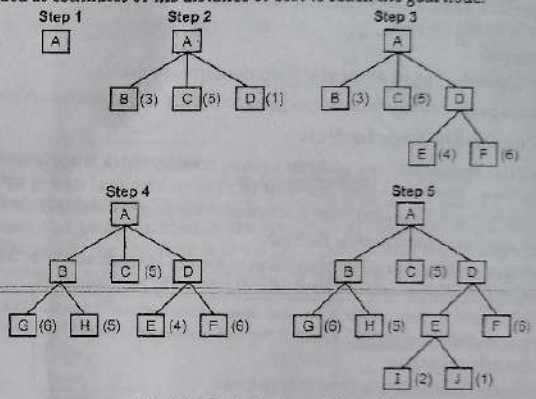


Fig. 3.12: Best-First Search Example

Heuristic function:

$f(n) = h(n)$

where,

h(n) - estimated straight line distance from node n to **goal.**

- To implement the graph search procedure, we will need to use two lists of nodes:
  - **OPEN:** Nodes that have been generated but have not been visited yet.
  - **CLOSED:** Nodes that have been already visited.
- **Best-First Search Algorithm:**

  Step 1 : Place the starting node into the OPEN list.

  Step 2 : If the OPEN list is empty, Stop and return failure.

  Step 3 : Remove the node n, from the OPEN list which has the lowest value of h(n), and places it in the CLOSED list.

Step 4 : Expand the node n, and generate the successors of node n.

Step 5 : Check each successor of node n, and find whether any node is a goal node or not. If any successor node is a goal node, then return success and terminate the search, else proceed to Step 6.

Step 6 : For each successor node, the algorithm checks for evaluation function f(n), and then checks if the node has been in either OPEN or CLOSED list. If the node has not been in both lists, then add it to the OPEN list.

Step 7 : Return to Step 2.

Advantage:

1. Best-first search can switch between BFS and DFS by taking the advantages of both the algorithms.

Disadvantages:

1. It considers the cost of the goal from the current state.
2. Some paths can continue to look good according to the heuristic function.

3.3.4 Constraint Satisfaction

- Constraint satisfaction is a problem solving method which is applicable to a variety of problems. There are many problems in AI in which a goal state is not specified in the problem and it requires to be discovered according to some specific constraint.
- For example, Cryptarithmetic Problem.
- Constraint satisfaction process operates in a space of constraint sets. The initial state contains the constraints that are originally given in the problem.
- A goal state is any state that has been constrained enough. For example in Cryptarithmetic problems, enough means that each letter has been assigned a unique numeric value.
- Constraint satisfaction is a two-step process:
  1. Constraints are discovered and propagated as far as possible.
  2. If there is still no solution, then search begins with adding new constraints and so forth.

Constraint Satisfaction Algorithm:

1. Propagate available constraints.
   (a) First set OPEN list of all objects that must be assigned values in a complete solution.
   (b) Repeat until inconsistency or all objects are assigned valid values.
      (i) Select an OB object from OPEN and strengthen as much as possible the set of constraints that apply to the object.
      (ii) If a set of constraints are different from the previous set then add to OPEN all objects [OB] that share any of these constraints.
      (iii) Remove OB from OPEN.

2. If the union of constraints discovered above defines a solution then quit a returns the solution.
3. If the union of the constraints discovered above defines a contradiction, t return failure.
4. If neither of the above occurs, then it is necessary to make a guess at someth in order to proceed. To do this, loop until a solution is found or all poss solutions have been eliminated:
   (a) Select an object whose value is not yet determined and select a way strengthening the constraints on that object.
   (b) Recursively invokes constraint satisfaction with the current set constraints augmented by the strengthening constraint just selected.

- To see how this algorithm works, consider the Cryptarithmetic problem as show Fig. 3.13.

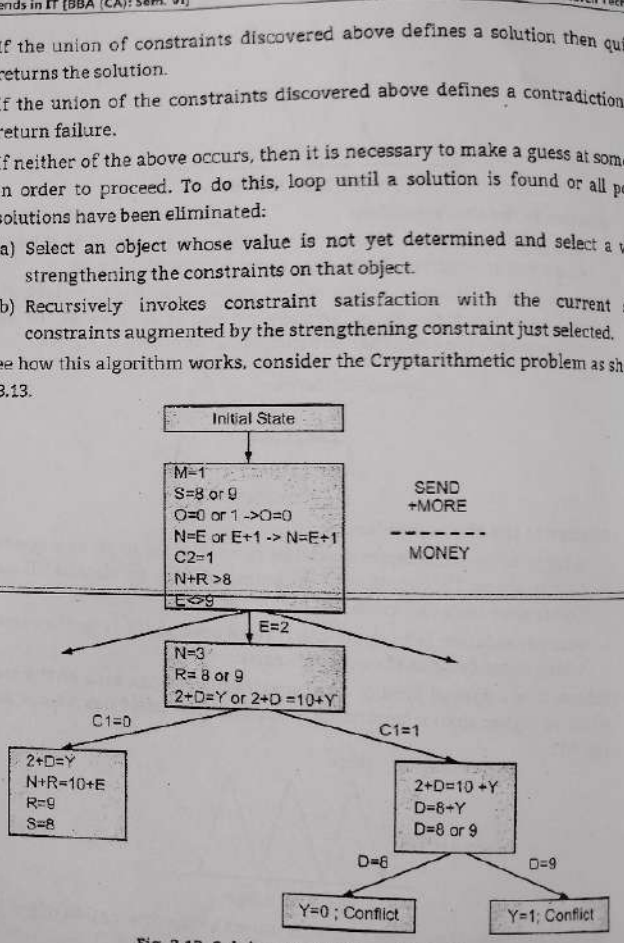


Fig. 3.13: Solving Cryptarithmetic Problem

- The goal state is a problem state in which all letters have been assigned digits in a way that no two digits have the same value i.e. all initial constraints are sati C1, C2, C3 represent carry bits.

3.3.5 Means End Analysis (MEA)

- The purpose of Means End Analysis is to identify a procedure that causes a transition from the current state to a goal state or at least to an intermediate state that is closer to the goal state.
- Means end analysis is a technique used to solve problems in AI which combines forward and backward strategies to solve complex problems. Using these mixed strategies, complex problems can be solved first, followed by smaller ones.
- First of all, the system evaluates the differences between the current state and the goal state. It then decides the best action to be undertaken to reach the End goal.
- The following Fig. 3.14 shows how the target goal is divided into Sub-goals that are then linked with executable actions.

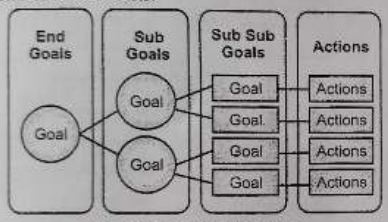


Fig. 3.14: Goals Dividation

Algorithm for **Means End Analysis:**

Step 1 : Compare CURRENT to GOAL, if there are no differences between both then return Success and Exit.

Step 2 : Else, select the most significant difference and reduce it by doing the following steps until the success or failure occurs.

(a) Select a new operator O which is applicable for the current difference, and if there is no such operator, then signal failure.

(b) Attempt to apply operator O to CURRENT. Make a description of two states.

(i) **O-Start**, a state in which O?s preconditions are satisfied.

(ii) **O-RESULT**, the state that would result if O were applied in O-start.

(c) If **(First-Part <------ MEA (CURRENT, O-Start)** And **(LAST-Part <----- MEA (O-Result, GOAL)**, are successful, then signal Success and return the result of combining First-Part, O, and Last-Part.

Example of problem solving in Means End Analysis:

- To solve the above problem, we will first find the differences between initial states and goal states, and for each difference, we will generate a new state and will apply the operators.

- The following Fig. 3.15 shows the initial state and the goal state.

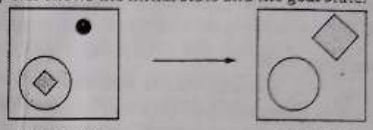


Fig. 3.15: Initial State and Goal State

Operators for the problem:

1. **Delete Operator:** In the initial state, the dot symbol at upper right corner which does not present in the final state. The dot symbol can be removed by applying the *delete operator* as shown in Fig. 3.16.

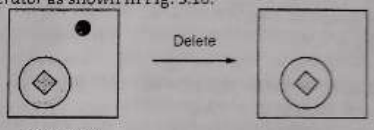


Fig. 3.16: Delete Operator

2. **Move Operator:** In the next step, we will compare the new state with the goal state. The green diamond in the new state is inside the circle while the green diamond in the end state is at the top right corner. We will move this diamond symbol to the right position by applying the *move operator* as shown in Fig. 3.17.

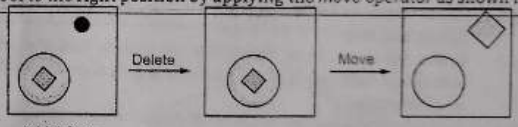


Fig. 3.17: Move Operator

3. **Expand Operator:** The new state generated in step 2 in which we find that the diamond symbol is smaller than the one in the end state. We can increase the size of this symbol by applying the *expand operator* as shown in Fig. 3.18.

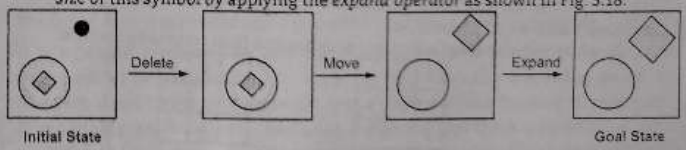


Fig. 3.18: Expand Operator

After applying the above three operators, we will find that the state in step 3 is the same as the goal state, which means that the problem has been solved.

3.3.6 A* and AO*

3.3.6.1 Introduction to A*

- The A* Algorithm is a specialization of best-first search. It provides general guidelines with which to estimate goal distances for general search graphs.
- At each node along a path to the goal, the A* algorithm generates all successor nodes and computes an estimate of the distance i.e. cost from the start node to a goal node through each of the successors. Then it chooses the successor with the shortest estimated distance for expansion. The successors for this node are generated, their distances estimated, and the process continues until a goal is found or the search ends in failure.
- The form of the heuristic function for A* is,

  $f(n) = g(n) + h(n)$

  Where,

  $g(n)$ = the cost or the distance from start node to node n.

  $h(n)$ = the cost from node n to a goal node.
- A* Algorithm involves maintaining two lists: OPEN and CLOSED.
  - OPEN contains those nodes that have been evaluated by the heuristic function but have not been expanded into successors yet.
  - CLOSED contains those nodes that have already been visited.

A* Algorithm:

Step 1 : Place the starting node in the OPEN list.

Step 2 : Check if the OPEN list is empty or not, if the list is empty then return failure and stop.

Step 3 : Select the node from the OPEN list which has the smallest value of evaluation function (g + h), if node n is goal node then return success and stop, otherwise.

Step 4 : Expand node n and generate all of its successors, and put n into the closed list. For each successor n', check whether n' is already in the OPEN or CLOSED list, if not then compute the evaluation function for n' and place it into the Open list.

Step 5 : Else if node n' is already in OPEN and CLOSED, then it should be attached to the back pointer which reflects the lowest g(n') value.

Problem Based On A* Algorithm:

- Given an initial state of a 8 puzzle problem and final state to be reached:

2	8	3
1	6	4
7		5

Initial State

1	2	3
8		4
7	6	5

Final State

- Find the most cost-effective path to reach the final state from initial state using A* Algorithm.

  Consider g(n) = Depth of node and h(n) = Number of misplaced tiles.

Solution:

- A* Algorithm maintains a tree of paths originating at the initial state.
- It extends those paths one edge at a time.
- It continues until the final state is reached as shown in Fig. 3.19.

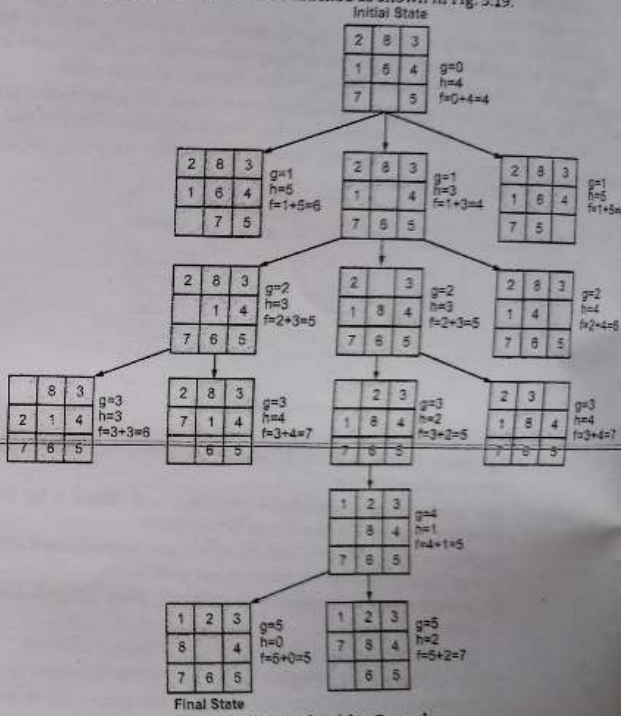


Fig. 3.19: A* Algorithm Example

Advantages:

1. It is complete and optimal.
2. It is the best one from other techniques. It is used to solve very complex problems.
3. It is optimally efficient, i.e. there is no other optimal algorithm guaranteed to expand fewer nodes than A*.

Disadvantages:

1. This algorithm is complete if the branching factor is finite and every action has fixed cost.
2. The speed execution of A* search is highly dependent on the accuracy of the heuristic algorithm that is used to compute h(n).

3.3.6.2 Introduction to AO* or AND-OR graphs

- When a problem is divided into sub problems, where each sub problem can be solved separately and a combination of these will be a solution, AND-OR graphs or AND-OR trees are used for representing the solution. The decomposition of the problem generates AND arcs. One AND may point to any number of successor nodes. All these must be solved so that the arc will rise to many arcs, indicating several possible solutions. Hence, the graph is known as AND-OR instead of AND. Fig. 3.20 shows an AND-OR graph.

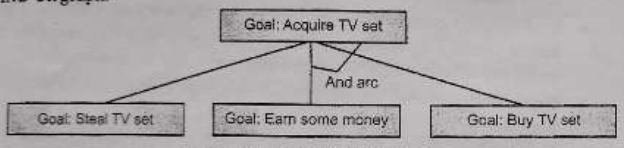


Fig. 3.20: A Simple AND-OR graph

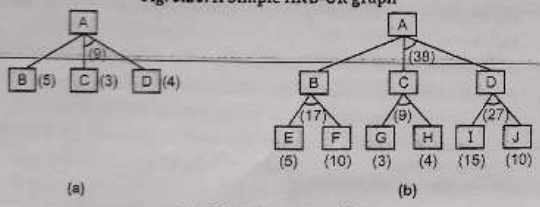


Fig. 3.21: AND-OR graph

- In Fig. 3.21, the top node A has been expanded producing two areas one leading to B and leading to C-D. The numbers at each node represent the value of f' at that node (cost of getting to the goal'state from the current state). It is assumed that every operation (i.e. applying a rule) has unit cost, i.e., each arc with a single successor will have a cost of 1 and each of its components. With the available information till now, it appears that C is the most promising node to expand since its f' = 3, the lowest but going through B would be better since to use C we must also use D' and the cost would be 9 (3 + 4 + 1 + 1). Through B it would be 6(5 + 1).
- Thus the choice of the next node to expand depends not only on a value but also on whether that node is part of the current best path from the initial mode. Fig. 3.21 makes this clearer. In this figure, the node G appears to be the most promising node,

with the least f' value. But G is not on the current beat path, since to use G we must use GH with a cost of 9 and again this demands that arcs be used (with a cost of 27). The path from A through B, E-F is better with a total cost of (17 + 1 = 18). Thus we can see that to search an AND-OR graph, the following three things must be done.

1. Traverse the graph starting at the initial node and following the current best path, and accumulate the set of nodes that are on the path and have not yet been expanded.
2. Pick one of these unexpanded nodes and expand it. Add its successors to the graph and computer f' (cost of the remaining distance) for each of them.
3. Change the f' estimate of the newly expanded node to reflect the new information produced by its successors. Propagate this change backward through the graph. Decide which of the current best paths.

- The propagation of revised cost estimation backward is in the tree is not necessary in A* algorithm. This is because in AO* algorithm expanded nodes are re-examined so that the current best path can be selected. The working of AO* algorithm is illustrated in following Fig. 3.22.

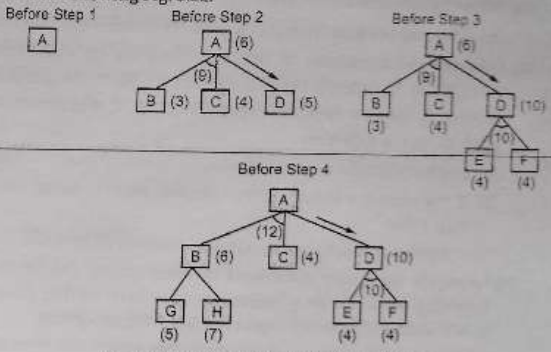


Fig. 3.22: The operation of problem reduction

- The initial node is expanded and D is marked initially as a promising node.
- D is expanded producing an AND arc E-F. f' value of D is updated to 10. Going backwards we can see that the AND arc B-C is better. It is now marked as the current best path. B and C have to be expanded next. This process continues until a solution is found or all paths have led to dead ends, indicating that there is no solution. An A* algorithm the path from one node to the other is always that of the lowest cost and it is independent of the paths through other nodes.

- The Depth first search and Breadth first search given earlier for OR trees or graphs can be easily adopted by AND-OR graphs. The main difference lies in the way termination conditions are determined, since all goals following an AND node must be realized; whereas a single goal node following an OR node will do.
- So for this purpose we are using AO* algorithm. Like A* algorithm here we will use two arrays and one heuristic function.
  - **OPEN:** It contains the nodes that have been traversed but yet not been marked solvable or unsolvable.
  - **CLOSE:** It contains the nodes that already have been processed.

AO* Algorithm:

1. Let G consists only of the node representing the initial state called this node INTT. Compute h' (INIT).
2. Until INIT is labeled SOLVED or h' (INIT) becomes greater than FUTILITY, repeat the following procedure.
   (i) Trace the marked arcs from INIT and select an unbounded node NODE.
   (ii) Generate the successors of NODE if there are no successors then assign FUTILITY as h' (NODE). This means that NODE is not solvable. If there are successors then for each one called SUCCESSOR, that is not also an ancestor of NODE do the following:
   (a) Add SUCCESSOR to graph G.
   (b) If the successor is not a terminal node, mark it solved and assign zero to its h' value.
   (c) If successor is not a terminal node, compute its h' value.
   (iii) Propagate the newly discovered information up the graph by doing the following. Let S be a set of nodes that have been marked SOLVED. Initialize S to NODE. Until S is empty repeat the following procedure:
   (a) Select a node from S call if CURRENT and remove it from S.
   (b) Compute h' of each of the arcs emerging from CURRENT, Assign minimum h' to CURRENT.
   (c) Mark the minimum cost path as the best out of CURRENT.
   (d) Mark CURRENT SOLVED if all of the nodes connected to it through the new marked are have been labeled SOLVED.
   (e) If CURRENT has been marked SOLVED or its h' has just changed, its new status must be propagated backwards up the graph. Hence all the ancestors of CURRENT are added to S.

Summary

- In this lesson, we have discussed the most common methods of proble representation in AI are:
  1. State Space Representation.
  2. Problem Reduction.
- State Space Representation is highly beneficial in AI because they provide possible states, operators and the goals. In case of problem reduction, a comple problem is broken down or decomposed into a set of primitive sub-problem solutions for these primitive sub-problems are easily obtained.
- Search is a characteristic of almost all AI problems. Search strategies can b compared by their time and space complexities. It is important to determine th complexity of a given strategy before investing too much programming effo since many search problems are traceable.
- Breadth-First searches are performed by exploring all nodes at a given dept before proceeding to the next level. This means that all immediate children nodes are explored before any children's children are considered.
- Depth first searches are performed by going downward into a tree as early a possible.
- In case of brute search (Uninformed Search or Blind Search) , nodes in the space are explored mechanically until a goal is found, a time limit has been reached, failure occurs.
- Examples of brute force search are breadth-first search and depth first searc
- In case of Heuristic Search (Informed Search) cost or another function is used t select the most promising path at each point in the search.
- Heuristics evolution functions are used in the best first strategy to find good solution paths.

Check Your Understanding

1. Which data structure is used to implement BFS?
   (a) Stack (b) Queue
   (c) Linked List (d) Priority Queue
2. What is a heuristic function?
   (a) A function to solve mathematical problems.
   (b) A function which takes parameters of type string and returns an integ value.
   (c) A function whose return type is nothing.
   (d) A function that maps from problem state descriptions to measures desirability.

3. Which is true regarding BFS (Breadth First Search)?
   (a) BFS will get trapped exploring a single path.
   (b) The entire tree so far been generated must be stored in BFS.
   (c) BFS is not guaranteed to find a solution if exists.
   (d) BFS is nothing but Binary First Search.
4. What is the problem space of Means-End analysis?
   (a) An initial state and one or more goal states.
   (b) One or more initial states and one goal state.
   (c) One or more initial states and one or more goal state.
   (d) One initial state and one goal state.
5. Which search strategy is also called as Blind Search?
   (a) Uninformed search (b) Informed search
   (c) Simple reflex search (d) All of the mentioned
6. Which search is implemented with an empty first-in-first-out queue?
   (a) Depth first search (b) Breadth-first search
   (c) Bidirectional search (d) None of the mentioned
7. What is the other name of informed search strategy?
   (a) Simple search (b) Heuristic search
   (c) Online search (d) None of the mentioned
8. Which function will select the lowest expansion node at first for evaluation?
   (a) Greedy best-first search (b) Best-first Search
   (c) Depth First Search (d) None of the mentioned
9. Which search is complete and optimal when h(n) is consistent?
   (a) Best-first Search
   (b) Depth first Search
   (c) Both Best-first and Depth first Search
   (d) A* Search
10. ____ are mathematical problems defined as a set of objects whose state must satisfy a number of constraints or limitations.
    (a) Constraints Satisfaction Problems
    (b) Uninformed Search Problems
    (c) Local Search Problems
    (d) All of the mentioned
11. Which of the Following problems can be modeled as CSP?
    (a) 8-Puzzle Problem (b) 8-Queen Problem
    (c) Map Coloring Problem (d) All of the mentioned

Answers

1. (b)	2. (c)	3. (b)	4. (a)	5. (a)	6. (b)	7. (b)	8. (b)	9. (d)	10. (a)	11. (d)

Practice Questions

Q.I Answer the following questions in short.

1. What are the two advantages of Breadth-First Search?
2. What are the two advantages of Depth First Search?
3. Write down the algorithm of Best-First Search Algorithm.
4. Write Hill Climbing Algorithm.
5. What are the disadvantages of Hill Climbing?
6. Write down the algorithm of Generate and Test.
7. What are the advantages and disadvantages of Iterative Deepening Search?
8. What are the advantages and disadvantages of Bidirectional Search?
9. State in which two failure conditions where depth limited search terminated?
10. What are the advantages and disadvantages of Uniform Cost Search?

Q.II Answer the following questions.

1. Write down the algorithm of Breadth-First Search with its advantages.
2. Write down the algorithm of Depth First Search with its advantages.
3. Explain A* Algorithm with example.
4. Explain AO* Algorithm.
5. Explain Means End Analysis with an example.

Q.III Define the terms.

1. Local Maximum
2. Plateau
3. Ridge
4. Heuristic Search
5. Depth-limited Search

4... Data Warehousing

Learning Objectives ...

- To introduce concept of data warehouse.
- To know about architecture of data warehouse.
- To get information of OLAP and OLTP servers
- To study about Multidimensional Data Model.
- To learn various types of OLAP servers.

4.1 INTRODUCTION TO DATA WAREHOUSE

- As we know that there is tremendous growth in generation of data. Traditional database systems have a drawback of handling homogeneous data. Moreover, large organizations need a reliable solution to handle this huge data that too with heterogeneous format. Also, the system handling data should be powerful enough to manipulate, access and analyze this huge data with an effective system. The solution to this was provided in 1980 by William Inmon by using the term data warehouse.
- Data warehouse is a concept which supports decision support systems where a large amount of data is merged together.
- A Data Warehouse (DW) is a repository which is at the top of multiple databases. It can be defined as a process for collecting and managing data from varied sources to provide meaningful business insights.
- The formal definition of a data warehouse by W. H. Inmon is given below:
  - *"A data warehouse is a subject-oriented (it can be used to analyze a particular area or domain), integrated (it integrates multiple data sources in a single identification), time-variant (historical data is stored in a data warehouse, whereas in a transaction processing system recent data is stored) and non-volatile (once the data is moved to a warehouse, it doesn't change) collection of data in support of management's decision-making process."*

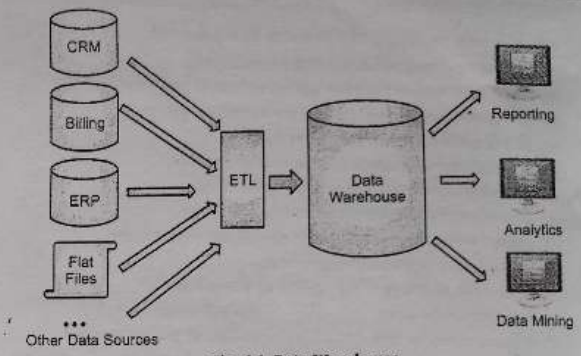


Fig. 4.1: Data Warehouse

- A data warehouse is used to analyze the data, to generate reports based on that analysis.
- Here, the data used for analysis or reporting is heterogeneous data collected from varied/multiple sources. The main purpose of data warehousing is to combine such data for the purpose of analysing and reporting. The data is then used for the strategic planning and decision making for the organizations.
- Data warehousing is different from normal data processing and querying.

Table 4.1: Difference between Database and Data Warehousing

Sr. No.	Database	Data Warehousing
1.	It is collection of interrelated and homogeneous data stored in rows and columns.	It is central storage of data from multiple data sources.
2.	It is used for manipulation of data.	It is used for analysis and reporting from the data.
3.	It is more of application-oriented.	It is subject-oriented.
4.	It uses online transaction processing system.	It uses an online analytical processing system.
5.	It is based on the relational model.	It is based on data modelling techniques.

4.2 STRUCTURE OF DATA WAREHOUSE

- A database schema is the logical structure of the database. This structure contains the information and entails about the name and other details of all the records associated with that database. It also includes the relationships and association of the data amongst themselves.

- In case of a traditional database system, database schema describes the details where as in the case of data warehouse, need to maintain the schema as well. The database uses a relational model whereas in case of data warehouse, it uses star and snowflakes schema.
- The schema design in the data warehouse is based on fact tables and dimension tables. The fact tables contain data corresponding to any business process. It stores quantitative information for analysis. A dimension table stores data about how the data in fact table is being analyzed. They facilitate the fact table in gathering different dimensions on the measures which are to be taken.
- The **star schema and snowflake schema** are two ways to structure a data warehouse.

4.2.1 Star Schema

- Star Schema is the most common schema in data warehouses. This is widely used to design the data warehouse. The basic architecture of star schema includes one fact table and many dimension tables. The advantage of star schema is that it is very efficient in handling the queries.

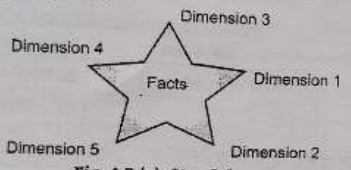


Fig. 4.2 (a): Star Schema

- It is called a star schema because the diagram looks like a star, with points radiating from a center. Star schema contains one fact table associated with many dimension tables. The center of the star contains fact table and the points of the star are the dimension tables.

Fact table:

- The fact table contains primary information of the data warehouse. The fact table has two types of columns having foreign keys to dimension tables and measures which contain numeric facts.
- The fact tables are in 3NF form.
- Typical fact tables store data about specific events such as sales event, holiday event.

Dimension Table:

- Dimensional tables provide detailed attribute data, records found in the fact table.
- The primary key which is present in each dimension is related to a foreign key which is present in the fact table.
- The dimension tables are in de-normalized form. Every dimension in the star schema should be represented by the only one-dimensional table.
- The dimension table should be joined to a fact table.
- Typical dimension tables store data about geographic region (markets, cities), times, clients, products.

Example of Star Schema:

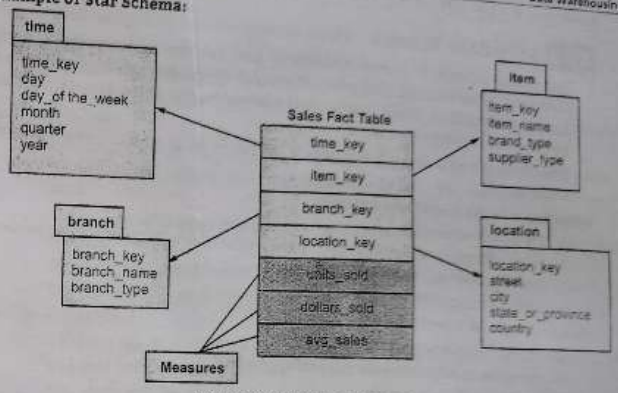


Fig. 4.2 (b): Example of Star Schema

- We are creating a schema which includes the sales data of a company. Sales are intended along following dimensions: time, item, branch, and location.
- The schema contains a central fact table for sales that includes keys to each of the four dimensions, along with measures: dollar-sold, units-sold and avg sales. The capacity of the fact table is reduced by the generation of dimension identifiers such as time_key and item_key via the system.
- Only a single table reproduces each dimension, and each table contains a group of attributes as it is shown in the star schema. The location dimension table includes the attribute set {location_key, street, city, state and country}. This restriction may introduce some redundancy. For example, two cities can be of same state and country, so entries for such cities in the location dimension table will create redundancy among the state and country attributes.

Advantages of Star Schema:

1. Its performance is good because simple queries are used.
2. It contains single dimension tables.
3. It has simple structure, so easy to understand.
4. It has less number of foreign keys and hence shorter query execution time.

Disadvantages of Star Schema:

1. It has redundant data and hence difficult to maintain/change.
2. There are data integrity issues.
3. Many-to-Many relationships within business entities are not supported.

4.2.2 Snowflake Schema

- Snowflake Schema is a modification of star schema where some dimensional hierarchy is normalized into a set of smaller dimension tables, forming a shape similar to snowflake. 'Snow flaking' or the normalization of the dimension tables can be done in many different ways.
- Snowflake schema is an arrangement of tables in a multidimensional database system.

Advantages to the Snowflake Schema:

1. Data is structured.
2. Data integrity is maintained. Data redundancy is completely removed by creating new dimension tables.
3. Less disk space is utilized.

Disadvantages of Snowflake Schema:

1. It requires more complex queries.
2. Complex queries decrease the performance.

Example of Snowflake Schema:

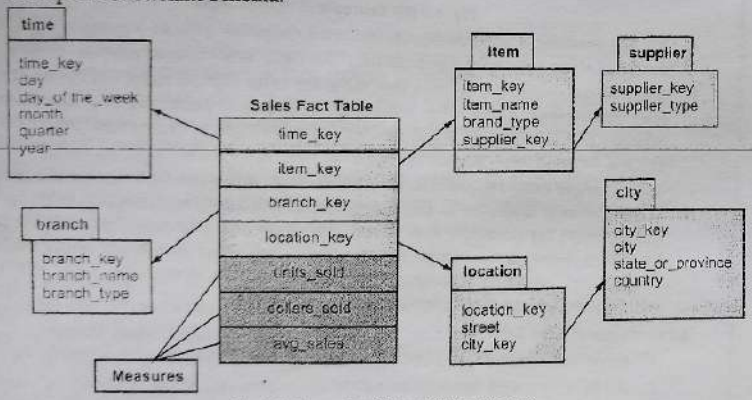


Fig. 4.3: Example of Snowflake Schema

- In this example, the sales fact table is identical to that of the star schema, but the main difference is in the definition of dimension tables.
- The single dimension table for the item in the star schema is normalized in the snowflake schema, results in creation of new item and supplier tables.
- For instance, the item dimension table consists of the attributes item_key, item_name, brand, type, and supplier_key, where supplier_key is connected to the supplier dimension table, which holds supplier_key and supplier_type information.

- Similarly, the location dimension table involves the attributes location_key, street, and city_key, and city_key is linked to city dimension table containing the city, state and country attribute.

4.2.3 Difference between Star Schema and Snowflakes Schema

Table 4.2: Star schema Vs Snowflake Schema

Sr. No.	Star Schema	Snowflake Schema
1.	Star schema is relational schema which follows the concept of facts and dimensions.	Snowflake schema is an extension of the star schema.
2.	Data redundancy is high.	Low data redundancy.
3.	Response time is fast.	It is less fast than star schema.
4.	Tables in database are not normalized.	Data is normalised.
5.	Single dimensional data is used.	Multidimensional data is used.
6.	Top-down approach is used.	Bottom-up approach is used.
7.	Simple in design.	Complex in design.

4.3 ADVANTAGES AND USES OF DATA WAREHOUSE

4.3.1 Advantages of Data Warehouse

- Data Warehouses are an introductory data infrastructure that provides the following advantages:

1. **Improved control of data:** Information in the data warehouse is under the control of data warehouse users so that, even if the source system data is removed over time. The information in the warehouse can be stored safely for extended periods of time.
2. **Better retrieval of data:** Data warehouses are separate from operational systems. They provide retrieval of data without slowing down operational systems.
3. **Increased productivity of corporate decision makers:** Data warehousing improves the productivity of corporate decision makers by creating an integrated database of consistent, subject-oriented, historical data. It integrates data from multiple incompatible systems into a form that provides one consistent view of the organization. By transforming data into meaningful information, a data warehouse allows business managers to perform more functional, accurate, and consistent analysis.

4. **More cost-effective decision making:** Data warehousing keeps all data in one place. It does not need much IT support. There is less of a need for outside industry information, which is expensive and difficult to integrate.
5. **Better enterprise intelligence:** It helps to provide better enterprise intelligence and enhanced customer service.
6. **Potential high Returns on Investment:** Return on Investment (ROI) refers to the amount of increased revenue. Implementations of data warehouses and complementary business intelligence systems have enabled business to generate higher amounts of revenue and provide significant cost savings.
7. **Provides competitive advantage:** The competitive advantage is gained by allowing decision-makers access to data that can reveal previously unavailable, unknown, and untapped information on such as customers, trends, and demands.

4.3.2 Uses of Data Warehouse

- Here are some major uses/applications of data warehouses across different industries/institutes:
- **Banking:**
  - Identify the potential risk of default and manage and control collections.
  - Performance analysis of each product, service, interchange, and exchange rates.
  - Track performance of accounts and user data.
  - Provide feedback to bankers regarding customer relationships and profitability.
- **Education:**
  - Store and analyze information about faculty and students.
  - Maintain student portals to facilitate student activities.
  - Extract information for research grants and evaluate student demographics.
  - Integrate information from different sources into a single repository for analysis and strategic decision making.
- **Insurance:**
  - Analyze data patterns and customer trends – Maintain records of all internal and external sources, including existing participants.
  - Design customized offers and promotions for customers.
  - Predict and analyze changes in the industry.
- **Government:**
  - Maintain and analyze tax records, health policy records, and their respective providers.
  - Prediction of criminal activities from patterns and trends.
  - Searching terrorist profile.
  - Threat assessment and fraud detection.

- **Finance:**
  - Evaluation of customer expenses trends.
  - Maintain transparency in transactions.
  - Predict/spot defaulters and act accordingly.
  - Analyze and forecast different aspects of business, stock, and bond performance.
- **Healthcare:**
  - Generate patient, employee, and financial records. This would include patient's personal information, financial transactions with the hospital, and insurance data.
  - Share data with other entities like insurance companies, NGOs, and Medical aid services.
  - Use data mining to identify patient trends.
  - Provide feedback to physicians on procedures and tests.
- **Construction:**
  - Generate the data of every purchase made during the construction timeline, the wages of contractual employees.
  - This data will be recorded in a data warehouse and later used for business visualizations to estimate the overall spending of the company on a single construction site.

4.4 ARCHITECTURE OF DATA WAREHOUSE

- The architecture shows collection of data, the storage of data, and the querying and data analysis support.
- Data warehouse architecture defines the arrangement of data in different databases. As the data must be organized and cleansed to be valuable, a modern data warehouse structure identifies the most effective technique of extracting information from raw data.
- Each data warehouse is different, but all are characterized by standard vital components.

4.4.1 Basic Components of Data Warehousing

- The basic components of data warehousing includes:

(a) **Data Migration:** Data migration is a one-time process of transferring internal data from one storage system to another: it may include preparing, extracting, and, if necessary, transforming the data.

(b) **The Warehouse Database:** The central component of a data warehousing architecture is a database that stocks all enterprise data and makes it manageable for reporting.

(c) **Access Tools:** These tools fall into four main categories: query and reporting tools, application development tools, online analytical processing tools, and data mining tools.

4.4.2 Properties for Data Warehouse System Architecture

- Architecture properties are important for a data warehouse system; these are as follows:
  - **Separation:** Analytical and transactional processing should be kept separately as much as possible.
  - **Scalability:** Hardware and software architectures should be easy to upgrade. Because the data volume and the number of users' requirements progressively increase.
  - **Security:** Monitoring accesses is necessary because of the strategic data stored in data warehouses.
  - **Extensibility:** Without redesigning the whole system, the architecture should be able to host new applications and technologies.
  - **Ability to administer:** Data warehouse management must be easy.

4.4.3 ETL Tools

- The data is extracted from the operational system. This extracted data must be reformatted, integrated, cleaned, and summarized before loading into a warehouse. The data which is not required gets removed in the conversion process. This process is done with the various tools available for transformation, cleaning, summarization and loading. These tools are called as **ETL tools** (Extract, Transform and Load).
- The main responsibilities of ETL tools include:
  - To remove the identity specifications of the data according to the rules.
  - To remove unwanted data from the operational database before loading it to the data warehouse.
  - To find and replace common names and definitions as the data is arriving from multiple destinations.
  - To summarize the data.
  - To recover the data with default values if there are missing values.
  - To remove the redundancy in data.
- This migration process is similar to that needed for data mining applications, but that data mining applications are not necessarily performed on summarized or business-wide data.

4.4.4 Types of Data warehouse Architectures

- There are mainly three types of Data warehouse Architectures:

1. **Single-tier Architecture:**

- The purpose of a single-tier is to minimize the amount of data stored. This goal is to remove data redundancy. This is a basic architecture of data warehouse which is not frequently used today.

2. **Two-tier Architecture:**

- Two-tier architecture separates physically available sources and data warehouse. This architecture is not expandable. This is also not supporting a large number of end-users. It has connectivity problems because of network limitations.

3. **Three-tier Architecture:**

- This is the most widely used architecture.
- Typical data warehouse architecture is three-tier architecture as follows:
  - (i) **Bottom Tier:** Bottom tier consists of an actual **data-warehouse server**. It includes summary data and metadata repository. **Summary data** is replication of detailed information which stores a summary of the data present in the data warehouse. **Metadata** is data about data. It is the card index describing how information is structured within the data warehouse. Metadata is used to map data sources to the common view of information within the data warehouse and used to automate the production of summary tables. It also used to direct a query to the appropriate data source.
  - (ii) **Middle Tier:** The middle tier is an **OLAP server**. It is implemented using a Relational OLAP model or Multidimensional OLAP.
  - (iii) **Top Tier:** Top tier consists of **front end tools** that are used for querying the database which is present in the data warehouse to get the information for analysis.

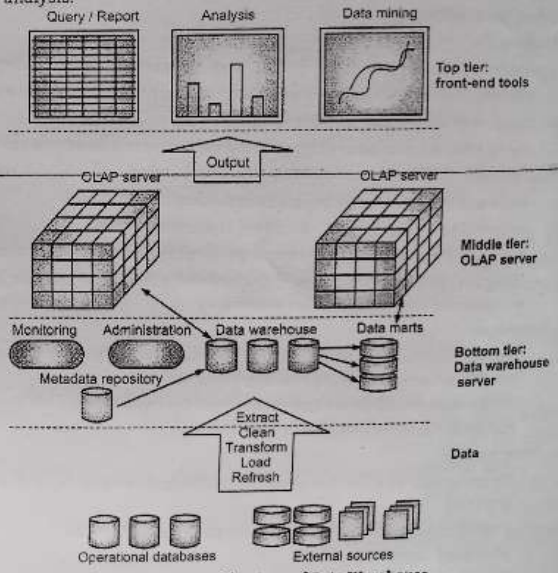


Fig. 4.4: Architecture of Data Warehouse

4.5 MULTIDIMENSIONAL DATA MODEL

- A multidimensional model views data in the form of a data-cube. Mostly, data warehousing supports two or three-dimensional cubes.
- A data cube enables data to be modelled and viewed in multiple dimensions. It is defined by dimensions and facts.
- Dimensions are organizations about which an entity needs to hold information. For example, dimensions allow storing to keep track of items such as monthly item purchases and branches and positions in the store sales record.
- Data warehouses and Online Analytical Processing (OLAP) tools are based on a multidimensional data model. OLAP in data warehousing enables users to view data from different angles and dimensions.

4.6 OLAP VS. OLTP

- Operational Database Management Systems also called as OLTP (Online Transactions Processing Databases), are used to manage dynamic data in real-time whereas Data Warehousing/Online Analytical Processing (OLAP) are used for large-scale processing and analysis of data.
- Following table shows the differences between OLTP and OLAP.

Table 4.3: Difference between OLTP and OLAP

Sr. No.	OLTP	OLAP
1.	Online Transaction Processing system.	Online Analytical processing system.
2.	It manages transaction oriented applications.	It manages the reports to multi-dimensional analytical queries.
3.	It is an online database modifying system.	It is an online query answering system.
4.	Its basic focus is on manipulating the database.	Its main focus is to analyze and extract the data for strategic decision making.
5.	The queries are short and simple.	The queries are long and complex.
6.	The modeling of OLTP is industry oriented.	The design of OLAP is subject or domain specific.
7.	The main purpose is to control day to day transactions in the database.	Its main purpose is to find the hidden data and support decision making.
8.	Less Number of data accessed.	Large Number of data accessed.
9.	Relational databases are created for online Transactional Processing (OLTP).	Data Warehouse designed for online Analytical Processing (OLAP).

4.7 OLAP OPERATIONS

- OLAP operations are done on multidimensional data. This multidimensional data is organized in various dimensions. Every dimension includes multiple levels of abstraction. So, there are various OLAP operations to demonstrate these views.
- OLAP operations are based on a multidimensional view of data. Here is the list of OLAP operations:
  - Roll-up
  - Drill-down
  - Slice and dice
  - Pivot (rotate)

1. Roll-up Operation:

- Roll-up operation performs aggregation on a data cube by either climbing up a hierarchy for a dimension or by reducing the dimensions. When roll-up is performed, some dimensions are reduced from the data.

Example 1: Consider the following example,

Location	Medal
Delhi	5
New York	2
Patiala	3
Los Angeles	5

- Delhi, New York, Patiala and Los Angeles won 5, 2, 3 and 5 medals respectively. So in this example, we roll upon Location from Cities to Countries.

Roll-up Operation:

Location	Medal
India	8
America	7

Example 2:

- Roll-up operation is performed by climbing up a concept hierarchy for the dimension location.
- Initially the concept hierarchy was "street < city < province < country".
- On rolling up, the data is aggregated by ascending the location hierarchy from the level of the city to the level of the country.
- The data is grouped into cities rather than countries.
- When roll-up operation is performed, one or more dimensions from the data cube are removed.

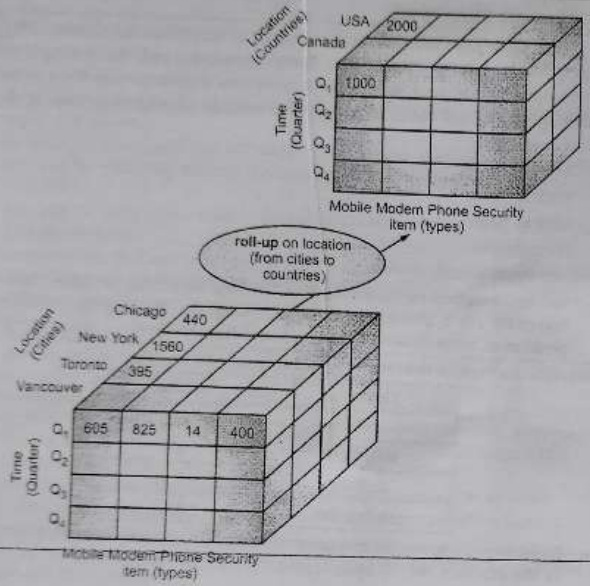


Fig. 4.5 (a): Roll-up Operation

2. **Drill-down Operation:**
   - The drill-down operation (also called roll-down) is the reverse operation of **roll-up**. It is performed by either of the following ways:
     - By stepping down a concept hierarchy for a dimension.
     - By introducing a new dimension.

Example 3:

Location	Medal
India	8
America	7

Location	Medal
Delhi	5
New York	2
Patiala	3
Los Angeles	5

Example 4:

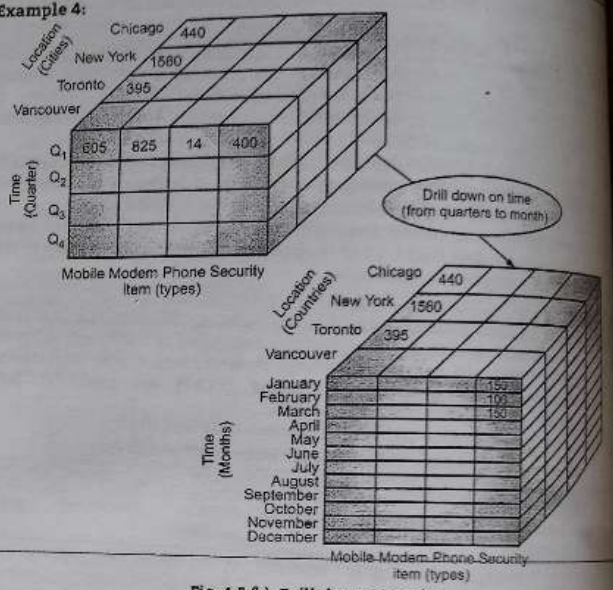


Fig. 4.5 (b): Drill-down operation

- Drill-down operation is performed by stepping down a concept hierarchy for the dimension time.
- Initially the concept hierarchy was "day < month < quarter < year".
- On drilling down, the time dimension is descended from the level of quarter to the level of month.
- When drill-down operation is performed, one or more dimensions from the data cube are added.
- It navigates the data from less detailed data to highly detailed data.

3. **Slice and Dice Operation:**
   - The slice operation performs a selection on one dimension of the given cube resulting in a sub-cube. It reduces the dimensionality of the cubes.

Example 5:

- For example, if we want to make a selection where Medal = 5.

Location	Medal
Delhi	5
Los Angeles	5

- The dice operation defines a sub-cube by performing a selection on two or more dimensions. For example, if we want to make a selection where Medal = 3 or Location = New York.

Location	Medal
Patiala	3
New York	2

Example 6: The slice operation selects one particular dimension from a given cube and provides a new sub-cube. The following diagram shows how does slice operation work.

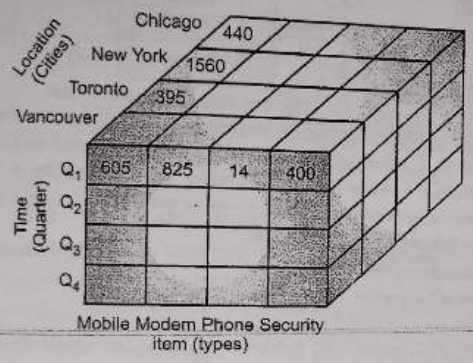


Fig. 4.5 (c): Slice Operation

- Dice operation selects two or more dimensions from a given cube and provides a new sub-cube. Consider the following diagram that shows the dice operation.

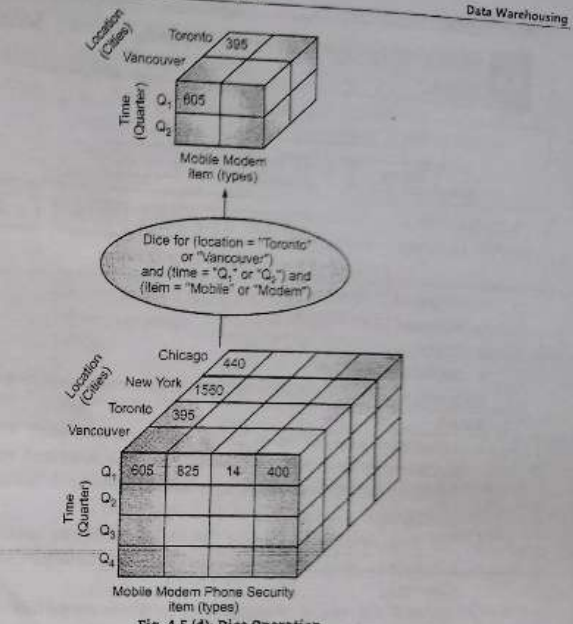


Fig. 4.5 (d): Dice Operation

- The dice operation on the cube based on the following selection criteria involves three dimensions:
  - (location = "Toronto" or "Vancouver")
  - (time = "Q1" or "Q2")
  - (item = " Mobile" or "Modem")

4. Pivot (Rotate) Operation:

- The pivot operation is also known as rotation. It rotates the data axis in view in order to provide an alternative presentation of data. Consider the following diagram that shows the pivot operation.

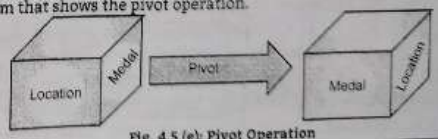


Fig. 4.5 (e): Pivot Operation

4.8 TYPES OF OLAP SERVERS: ROLAP VERSUS MOLAP VERSUS HOLAP

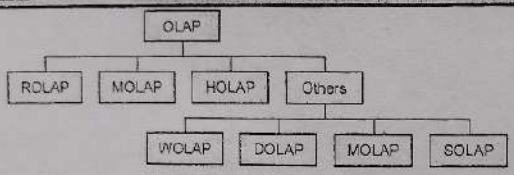


Fig. 4.6: Types of OLAP Server

- Three types of OLAP servers are:
  1. Relational OLAP (ROLAP)
  2. Multidimensional OLAP (MOLAP)
  3. Hybrid OLAP (HOLAP)
- These data models differ mainly in terms of data storage and technique.

1. **ROLAP:**
   - ROLAP stands for Relational Online Analytical Processing. ROLAP servers act as mediators between back end server and user front end tools. The relational databases handle warehouse data and OLAP acts as middleware to provide missing data.
   - The main advantage of using ROLAP is that it can handle large amount of information as RDBMS comes with lots of functionalities and ROLAP works on top of that.
   - The limitation of ROLAP is that it can be slower the performance, if the data size is bigger. Each ROLAP operation is based on SQL query; so as there are limitations on SQL, the ROLAP can not be used.

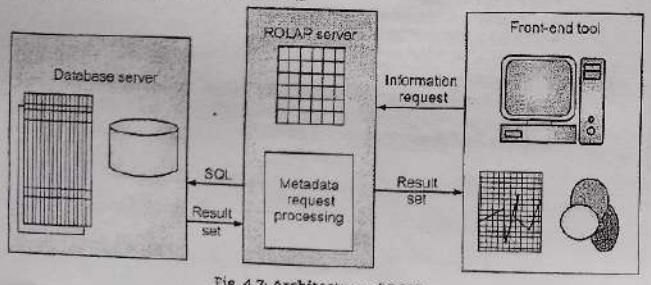


Fig. 4.7: Architecture of ROLP

2. **MOLAP:**
   - MOLAP stands for Multidimensional OLAP.

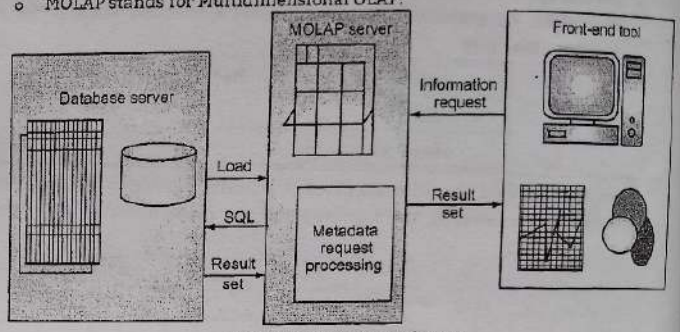


Fig. 4.8: Architecture of MOLP

   - MOLAP system is based on multidimensional data model and operations. Data is stored in multidimensional arrays and positional techniques are used to access them. Here, data is summarized and stored in a multidimensional cube. Relational model is not used to store the data.
   - It has following components:
     - Database server.
     - MOLAP server.
     - Front-end tool.
   - MOLAP is extensively used in applications where iterative and comprehensive time-series analyses of trends are done.
   - This can be very useful for organizations with performance-sensitive multi-dimensional analysis requirements and those have built or are in the process of building a data warehouse architecture that contains multiple subject areas.
   - The advantages of MOLAP include high performance and efficiency in performing complex operations/calculations.
   - The disadvantages include lack of handling large volumes of data.

3. **HOLAP:**
   - HOLAP stands for Hybrid OLAP, an application using both relational and multidimensional techniques.
   - HOLAP includes the best features of **MOLAP** and **ROLAP** into a single architecture. HOLAP provides benefits of both MOLAP and ROLAP. HOLAP offers greater scalability than ROLAP and faster computation than MOLAP.
   - HOLAP stores a large amount of data and it provides fast access at all levels of aggregation.

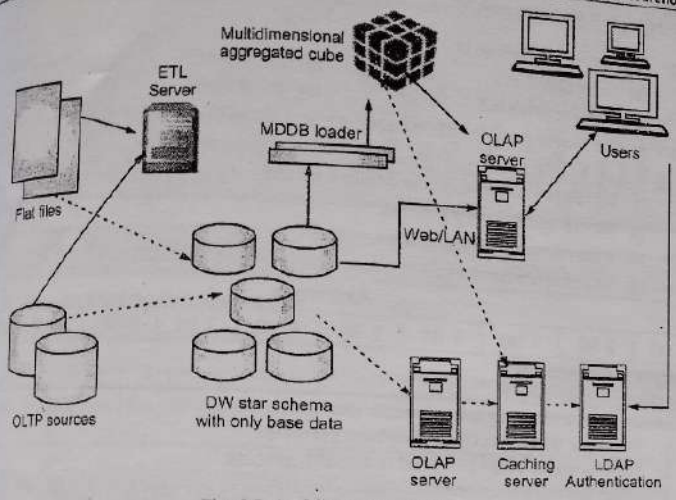


Fig. 4.9: Architecture of HOLP

- The main advantage of HOLAP balances the disk space requirement, as it stores the aggregate information on the OLAP server and the detail record remains in the relational database. So no duplicate copy of the detail record is maintained.
- HOLAP architecture is very complicated because it supports both MOLAP and ROLAP servers.

Other types of OLAP Server:

- Other types of OLAP include:
  - **Web OLAP (WOLAP):** WOLAP refers to an OLAP application that can be accessed through a web browser.
  - **Desktop OLAP (DOLAP):** DOLAP stands for desktop analytical processing. In comparison to other OLAP applications, functionality is limited. It is less expensive.
  - **Mobile OLAP (MOLAP):** The user is working and accessing data via mobile devices.
  - **Spatial OLAP (SOLAP):** SOLAP combines the capabilities of Geographic Information Systems (GIS) and OLAP into a single user interface. This allows for the quick and easy exploration of data stored in a spatial database.

Table 4.4: Difference between ROLAP, MOLAP, and HOLAP

Basis of Comparison	ROLAP	MOLAP	HOLAP
Meaning	Relational Online Analytical Processing	Multi-Dimensional Online Analytical Processing	Hybrid Online Analytical Processing
Data Storage	It stores data in a relational database.	It stores data in a multi-dimensional database.	It stores data in a relational database
Technique	It uses Structured Query Language (SQL).	It utilizes the Sparse Matrix technique.	It uses a combination of SQL and Sparse Matrix technique.
Volume of data	It processes huge data.	It can process a limited volume of data.	It can process huge volumes of data.
Data Arrangement	It arranges data in rows and columns (tables).	It arranges data in data cubes.	There is a multi-dimensional arrangement of data
Designed View	The multi-dimensional view is dynamic.	The multi-dimensional view is static.	The multi-dimensional view is dynamic.

Summary

- "Data Warehouse" was first coined by Bill Inmon in 1990.
- A Data Warehouse (DW) is a collection of technologies aimed at enabling the knowledge worker (executive, manager, and analyst) to make better and faster decisions.
- A data warehouse can provide a central repository for large amounts of historical, integrated, and consistent data.
- A data warehouse is a subject-oriented, integrated, time-varying, non-volatile collection of data in support of the management's decision making process.
- The entire process of getting data into the data warehouse is called Extract, Transform and Load (ETL) process. Architecture, and users can access data directly from the various source systems through the data warehouse.

- Two-level (two-layer) architecture of data warehouse highlights the separation of physically available sources and data warehouses.
- Generally a data warehouses adopts a three-tier (layer/level) architecture consist of the top, middle and bottom tier.
- OLAP Operations are done on multidimensional data Roll-up these operations are Roll-up, Drill-down, Slice and dice and Pivot (rotate).
- A database schema is the logical structure of the database.
- There are two types of schemas in data warehouse: Star schema and Snowflake schema.

Check Your Understanding

1. OLAP stands for _____.
   (a) Online Analytical Processing (b) Online Analysis Processing
   (c) Online Transaction Processing (d) Online Aggregate Processing
2. Data that can be modeled as dimension attributes and measure attributes are called _____ data.
   (a) Multidimensional (b) Single dimensional
   (c) Measured (d) Dimensional
3. What do data warehouses support?
   (a) OLAP (b) OLTP
   (c) OLAP and OLTP (d) Operational databases
4. _____ is a subject-oriented, integrated, time-variant, nonvolatile collection of data in support of management decisions.
   (a) Data Mining (b) Data Warehousing
   (c) Web Mining (d) Text Mining
5. The data Warehouse is_____.
   (a) read only (b) write only
   (c) read write only (d) None of the mentioned
6. The important aspect of the data warehouse environment is that data found within the data warehouse is_____.
   (a) subject-oriented (b) time-variant
   (c) integrated (d) All of the above
7. _____ describes the data contained in the data warehouse.
   (a) Relational data (b) Operational data
   (c) Metadata (d) Informational data
8. Which of these is not an access tool in data warehousing?
   (a) OLAP tools (b) Data mining tools
   (c) Reporting tools (d) Source system tools

9. The Data Warehouse is _____.
   (a) Integrated (b) Non-volatile
   (c) Subject oriented (d) All of these
10. A data warehouse is said to contain a 'time-varying' collection of data because _____.
    (a) Its contents vary automatically with time.
    (b) Its life-span is very limited.
    (c) It contains historical data.
    (d) Its contents have explicit time-stamp.

Answers

1. (a)	2. (a)	3. (a)	4. (b)	5. (a)	6. (d)	7. (c)	8. (d)	9. (d)	10. (c)

Practice Questions

Q.I Answer the following questions in short.

1. What are OLTP and OLAP database systems?
2. List the major steps involved in the ETL process.
3. What is a data warehouse?
4. What are the benefits of building an enterprise data warehouse?
5. List major components of any data warehouse system.
6. List the characteristics of OLAP systems.

Q.II Answer the following questions.

1. What is the major difference between the star schema and the snowflake schema?
2. List some differences between an OLTP system and a data warehouse system.
3. Describe the applications of a data warehouse.
4. What are the major differences between OLTP and OLAP system?
5. Explain the star scheme technique of modelling a data warehouse.
6. Explain a multidimensional view and a data cube.

Q.III Define the terms.

1. Star schema
2. Snowflake schema
3. OLAP Cube
4. ETL tools
5. Fact table
6. MOLAP
7. SOLAP
8. DOLAP
9. Metadata

5... Data Mining

Learning Objectives ...

- To understand the concept and need of Data Mining.
- To study the KDD process in detail.
- To know the techniques and knowledge representation techniques in Data Mining.
- To understand the real world applications of data mining.

5.1 INTRODUCTION TO DATA MINING

- Today, we live in a world where millions and trillions of data is generated daily. With the evolution of WWW (World Wide Web), there has been tremendous growth in information content, information processing and information retrieval.
- Information retrieval is also a major need as it is the basis for future prediction, analysis and decision making. Mining refers to the extraction of valuable things. Data mining, in turn, refers to the study of collecting, cleaning, processing and analyzing the data and to retrieve meaningful information from huge data.
- Data Mining is analysis of data and use of software techniques and statistical methods to find patterns in data.
- Data Mining deals with discovery of hidden knowledge, unexpected patterns and new rules from large data sets.
- Data Mining is the use of algorithms to extract the information and patterns derived by the KDD process.
- It is also known as the process of extracting hidden information from a large data set i.e. mining knowledge from data.

5.2 DATA MINING TASK

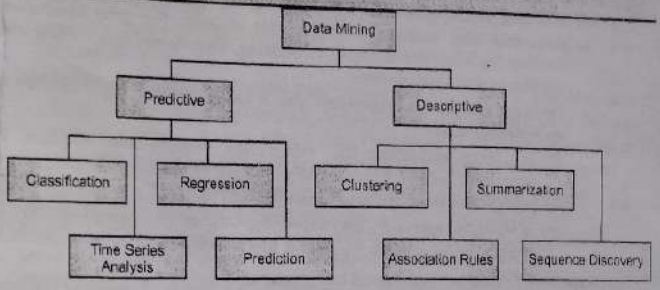


Fig. 5.1: Data Mining Tasks

- Data mining tasks can be categorized into two main types: Predictive and Descriptive.

(A) Predictive Data Mining:

- Predictive data mining tasks include the prediction based on the available data set in hand. These tasks give the model based on data and predict the future trends related to that data or unknown values that may be of interest for the future.
- The example of predictive tasks includes the prediction of future value of gold according to the current market trend. Also, prediction of high or low value of a share in the share market based on its previous growth is also a predictive data mining task.
- Predictive data mining includes Classification, Regression, Prediction and Time Series Analysis. Let's see the details of these tasks.

1. **Classification:** The dictionary meaning of classification is to classify, i.e. to categories or create groups of the data items according to particular criteria. In data mining, classification can be defined as arrangement of data items or making groups of data items based on the data points or observed values. The output of classification is a method that will decide the class of an object based on its attributes.

   Examples of classification are:

   (a) To find potential customers for a new product.

   (b) To find the probable list of customer who are likely to apply for the credit card, based on previous data.

2. **Regression:** Regression can be defined as a data mining technique that is generally used for the purpose of predicting a range of continuous values (which can also be called "numeric values") in a specific data set.

- It is used to map data items to a real valued variable. Regression is very frequently used in business and market analysis. The main application involves financial prediction or forecasting, Environmental modeling and analyzing trends and patterns.
- There are two types of regression. One is Linear regression and another is Multiple regression.
  (i) In **Linear regression**, the relationship between two variables is established using a linear equation to observe the data. The output is a straight line which has only line dependent variables.
  (ii) In **Multiple regression**, the relationship between two or more variables is established to predict the output and a single continuous dependent variable.

3. **Time Series Analysis:** Time series analysis is the process of recording the data point at specific time intervals. This data is then used to predict the future values based on the data points recorded. Time series analysis can produce very important information for a business if used efficiently. The example of time series includes weather record, economic indicators, stock market analysis, workload projections etc.
4. **Prediction:** Prediction is a classification task. Prediction discovers the relationship between dependent variables and relationship between independent variables. It can also be viewed as estimation. The prediction is based on the data in hand and predictions or future trends of a phenomenon can be predicted using some predictive algorithms. The best example of prediction is the profit that could be gained out of sale. Predication is the technique of identifying the unavailable numerical data for a new process. Prediction applications include flooding, speech recognition, machine learning, and pattern recognition.

Descriptive Data Mining:

Descriptive data mining tasks include the analysis of available data patterns or models to find out new interesting and significant information based on available data set.

The example of descriptive data mining tasks includes the interchange in places of the super market according to the purchase pattern of the customers.

Descriptive data mining includes Clustering, Summarization, Association Rules and Sequence Discovery. Let's discuss these tasks one by one.

Clustering:

- Clustering or Cluster Analysis is the method where the data points are grouped together according to their characteristics. The data points in the same group (called a cluster) are more similar (in some sense) to each other than to those in other groups (clusters). Clustering can also be referred to as *unsupervised learning*.

- Clustering can be used to find out the specific class of customer in the market.
- Clustering of similar character genes can give a new insight. Clustering can be used in outlier detection.

2. **Summarization:**
   - Summarization is the process of finding the list of the generated data. The process of Summarization divides the data into subsets with descriptions. Summarization is also called **Characterization or Generalization**. It extracts or derives representative information about the database. This may be accomplished by actually retrieving portions of the data.
3. **Association Rules:**
   - Association rules find out the correlation among the data. Association rules find out a specific type of association between the data items. These associations are used to identify the frequency occurrence in the pattern and accordingly the strategies for business are changed or modified.
   - An example of association rule is that in supermarket, generally, bread and butter are kept side by side or milk and bread are kept near to each other so that the person who buys milk will surely have the association of buying the bread as well. This phenomenon is also known as **Market Basket Analysis**
4. **Sequence Discovery:**
   - Sequence discovery, or Sequential Pattern Mining, is a data mining technique that discovers statistically relevant patterns in sequential data. This mining program evaluates certain criteria, such as occurrence frequency, duration, or values in a set of sequences to find interesting hidden patterns. For example, most people who purchase CD players may be found to purchase CDs within one week.
   - Other typical examples include customer shopping sequences (First buy computer, then CD-ROM, and then digital camera, within 3 months), Web clickstreams, Biological sequences, Sequences of events in Science and Engineering, and in Natural and Social developments.

5.3 DATA MINING ISSUES

- Data mining issues should be always considered by all data miners and data mining algorithms before going for data mining. Following issues were faced while doing data mining.

1. **Human Interaction:** When a data mining task is to be undertaken, the goal is not clear. Users as well as the technical expert are unaware of the results. There is a need for a proper interface between the domain expert and users. The queries are formed by the experts based on the user's demand.
2. **Overfitting:** Overfitting is a statistical error. When a model is generated for a particular data set, it is supposed that the same model should accommodate future data sets as well. But overfitting occurs when the generated model is well

suited for the training data set and it is not suited for the test data set or future data set. Overfitting can be reduced by increasing the training data set and by reducing model complexity.

3. **Outliers:** When a model is derived, there are some values of data that do not fit in the model. These values are significantly different from the normal values, or they do not fit in any cluster. These values are called outliers. They can also be called as exceptions in the model derived. If a model includes outliers, it may not behave well for other significant values.
4. **Interpretation of the results:** Interpretation of the results obtained by data mining is a very crucial task. This interpretation is beyond only explanation of the results. This task requires expert analysis and interpretation. Hence, interpretation of the results is an issue in data mining.
5. **Visualization of the results:** Visualization of the results is useful to understand and quickly view the output of the different database algorithms.
6. **Large data sets:** Data Mining models are generally designed to test the small data sets. But, when these models are applied to large data sets i.e. data sets with larger size then these models either fail or they wobble. There are many such models that work very well for the normal data sets but are inefficient in handling large data sets. The large data set issue can be handled with sampling and parallelization.
7. **High Dimensionality:** Dimensionality of the database refers to the different attributes that are present in the database. High dimensionality in a database leads to more number of attributes leading to confusion of choosing the attributes for the particular task. An increase in the number of attributes increases the complexity and effectiveness of the algorithm. The solution to High Dimensionality is to reduce the number of attributes.
8. **Multimedia Data:** Many users demand the mining tasks for graphical, video or audio data. The multimedia data can be an issue in data mining as traditionally data mining tasks are designed for numeric or alphanumeric data.
9. **Missing Data:** Sometimes the data is incomplete or missing. During the KDD process, this data may be filled with nearest estimates. These estimates may give false or invalid results creating problems.
10. **Irrelevant data:** Some data used in the mining task may not be relevant to the actual mining tasks. This data may either lead to invalid results or may vary the results.
11. **Noisy Data:** The data which has no meaning is called noisy data. These values need to be corrected or replaced with meaningful data.
12. **Changing Data:** The databases used for the mining tasks are subject to change. The algorithms run on the database at a particular time may show different results if the database is dynamic. This issue can be solved by running the data mining algorithm of the changed database completely.

13. **Integration:** The normal database querying results in the output as a traditional data processing system whereas KDD process gives the results which are unknown to the user. There is no union in the KDD process and the traditional Query processing. The integration of these two will certainly give more profitable results.
14. **Application:** The output of the Data mining process results in unknown facts about the data. It is a big challenge to use this data for the purpose of decision making. This task is more crucial than developing or applying algorithms to databases. Proper application or use of mining tasks will produce better results.

5.4 DATA MINING VERSUS KNOWLEDGE DISCOVERY IN DATABASES

- Let us first understand about Knowledge Discovery in Databases, and then we will see the difference between Data Mining and Knowledge Discovery in Databases.

5.4.1 Knowledge Discovery in Databases

- The KDD process (Knowledge Discovery in Database process) is the process of digging or finding the truth laid in databases which is not known yet and the things which are previously not discovered. The patterns, the stuff that has not been detected yet can be found with the KDD process. This extraction of unknown stuff through the KDD process is useful in automating summarization, pattern recognition and finding out the truth from facts and figures.
- The KDD process is divided into following steps:

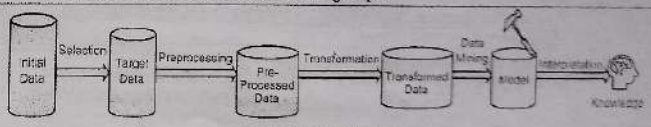


Fig. 5.2: KDD Process

1. **Selection:** The data which is to be mined may not be necessarily from a single source. The data may have many heterogeneous origins. This data needs to be obtained from various data sources and files. The data selection is based on your mining goal. Data relevant to the mining task is selected from various sources.
2. **Pre-processing:** Pre-processing involves cleaning of the data and integration of the data. The data selected for mining purposes may have some incorrect, irrelevant values which lead to unwanted results. Some values may be missing or erroneous. Also, when data is collected from heterogeneous sources, it may involve varying data types and metrics. So, this data needs to be cleaned and integrated for noise elimination and inconsistency.

3. **Transformation:** Data transformation is the process of converting the data into the format which is suitable for processing. Here, data is created in the form which is required by the data mining process.
4. **Data Mining:** The Data Mining process leads towards using methods, techniques to extract the pattern present in the data. The process involves transformation of relevant data records into patterns using classification. This step involves application of various data mining algorithms to the transformed data. This process generates the desired results for which the whole KDD process is undertaken.
5. **Visualization/Interpretation:** This is the last step in the KDD process. In this step, the data is presented to the user in the form of reports, tables or graphs. The presentation of the data to the users directly affects the usefulness of the results.

Table 5.1: Data Mining Vs Knowledge Discovery in Databases

Sr. No.	Data mining	Knowledge Discovery in Databases
1.	Data Mining is application of a specific algorithm in order to extract patterns from data.	KDD (Knowledge Discovery in Databases) is a field of computer science, which includes the tools and theories to help humans in extracting useful and previously unknown information (i.e. knowledge) from large collections of data.
2.	Data Mining is only the application of a specific algorithm based on the overall goal of the KDD process.	KDD consists of several steps, and Data Mining is one of them.
3.	Data mining process visualizes the data for generation of the results.	The KDD process involves data cleaning, data integration. Also it creates a common repository for all resources such as data warehouses.

5.5 DATA MINING VERIFICATION VS. DISCOVERY

Table 5.2: Difference between Verification and Discovery in Data Mining

Sr. No.	Verification	Discovery
1.	It takes a hypothesis from the user and tests the validity of it against the data.	Knowledge discovery is the concept of analyzing large amount of data and gathering out relevant information leading to the knowledge discovery process for extracting meaningful rules, patterns and models from data.

contd. ...

2.	The emphasis is with the user who is responsible for formulating the hypotheses and issuing the query on the data to affirm or negate the hypothesis.	The discovery model differs in its emphasis in that it is the system automatically discovering important information hidden in the data
3.	No new information is created in the retrieval process.	The discovery or data mining tools aim to tell many facts about the data.
4.	The search process here is iterative in that the output is reviewed, a new set of questions or hypothesis formulated to refine the search and the whole process repeated.	The data is shifted in search of frequently occurring patterns, trends and generalizations about the data without intervention or guidance from the user.

5.6 DATA PRE-PROCESSING – NEED, DATA CLEANING, DATA INTEGRATION AND TRANSFORMATION, DATA REDUCTION

5.6.1 Need of Data Pre-Processing

- We know that a tremendous amount of data is generated daily in today's world of web. The huge size of the data makes it vulnerable to changes. The resultant data may be incomplete, noisy and inconsistent. This weakens the quality of the data which is used for data mining purposes leads to invalid results. The process of cleaning the data and making it useful for the process of mining is called Data Pre-processing.

Pre-processing Process:

- The pre-processing process consists of many steps. Pre-processing can be performed manually or automatically. Let's discuss the steps involved in Data Pre-processing.

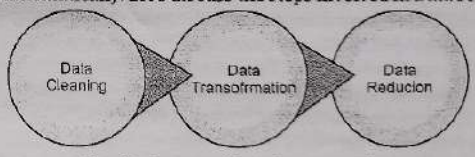


Fig. 5.3: Process of Data Pre-processing

5.6.2 Data Cleaning

- The first step in data pre-processing is data cleaning. It is also known as scrubbing. Data cleaning includes handling missing data and noisy data.
  - **(a) Missing data:** Missing data is the case wherein some of the attributes or attribute data is missing or the data is not normalized. This situation can be handled by either ignoring the values or filling the missing value.

(b) **Noisy data:** This is data with error or data which has no meaning at all. This type of data can either lead to invalid results or can create the problem to the process of mining itself. The problem of noisy data can be solved with binning methods, regression and clustering.

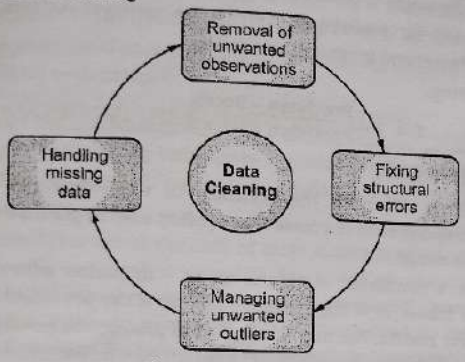


Fig. 5.4: Data Cleaning

5.6.3 Data Integration and Transformation

- Data integration is the process of combining data from disparate sources into a meaningful and valuable data set for the purpose of analysis.
- In this step, a logical data source is prepared. This is done by collecting and integrating data from multiple sources like databases, legacy systems, flat files, data cubes etc.

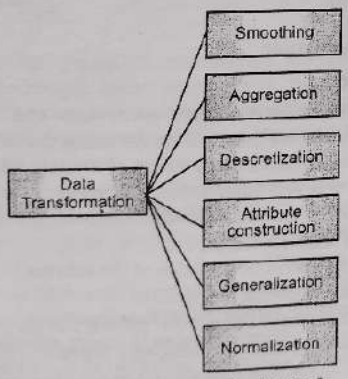


Fig. 5.5: Data Transformation Methods

- Data used for data mining is the data which comes from various heterogeneous platforms. This unstructured and structured data needs to be combined for smooth processing of data mining. This homogeneous data is then analyzed to find out the patterns. The benefits of data transformation include the improved quality of the data. The user can get maximum value out of the available data. Transformation also helps in improving the performance of the queries. There are specialized tools for data transformation.
- The various data transformation methods include:
  - (a) **Smoothing:** This is the process of removing the unnecessary data and cleaning the data so as to improve the functionality of the data.
  - (b) **Aggregation:** This is the process of collecting the data from heterogeneous platforms and converting it to a uniform format. This improves the quality of the data.
  - (c) **Discretization:** Large data sets are complex to handle. Discretization is the process of breaking up the data in small intervals. These chunks are continuous chunks and these are supported by all the existing frameworks.
  - (d) **Attribute construction:** To improve the efficiency in the mining process, some new attributes are generated from existing data sets.
  - (e) **Generalization:** This is the process of converting low level attributes to high level attributes using hierarchy.
  - (f) **Normalization:** In the process of Normalization, attributes are scaled within a specified range.

5.6.4 Data Reduction

- Data reduction is a process that reduced the volume of original data and represents it in a much smaller volume.

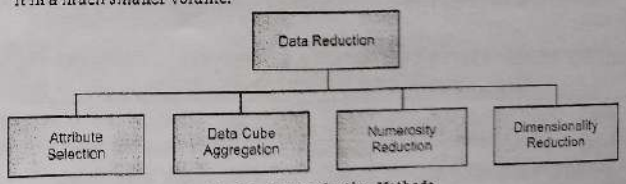


Fig. 5.6: Data Reduction Methods

- The various data reduction methods include:
  - (a) **Attribute Selection:** When data is collected from various sources, it may contain duplicate attributes. Some of the attributes are irrelevant. The Attribute Selection method is used to remove such redundant and unnecessary attributes from the data set. This process results in an improved data set.

(b) **Data Cube Aggregation:** In this reduction method, aggregation property is applied on selected data sets so as to get the data in a much simpler format.

(c) **Numerosity Reduction:** In this reduction method, actual data is substituted with a mathematical model of the data.

(d) **Dimensionality Reduction:** In this reduction method, duplicate attributes are removed to reduce the data size.

5.7 ACCURACY MEASURES: PRECISION, RECALL, F-MEASURE, CONFUSION MATRIX, CROSS-VALIDATION, BOOTSTRAP

- In this section, we present the performance of the classifiers using different metrics: Accuracy, Precision, Recall and F-measure. These metrics are used to provide a better overview of the model performance. The accuracy measure by itself is not a perfect measure if the data set is not balanced. Precision and recall are better measures in the case of imbalanced data sets.

5.7.1 Accuracy Measures

- The accuracy of a classifier is given as the percentage of total correct predictions divided by the total number of instances.
- The information system consists of a number of different documents. And the various operations are done on these documents to retrieve useful information.
- The information is retrieved using queries. The similarity between the query and the retrieved document is calculated. This similarity measure is a set membership function describing the likelihood of the document that the retrieved document is relevant to users query.

5.7.2 Precision and Recall

- The effectiveness of the system in processing a query is measured by precision and recall.
- Precision and Recall are calculated by,

$$\text{Precision} = \frac{[\text{Relevant and Retrieved}]}{[\text{Retrieved}]}$$

$$\text{Recall} = \frac{[\text{Relevant and Retrieved}]}{[\text{Relevant}]}$$

- Precision is used to answers the question: "Are all documents retrieved one that I am interested in?" in short, precision is the fraction of relevant instances among the retrieved instances.
- Recall answers "Have all relevant documents been retrieved?" Recall is the fraction of relevant instances that were retrieved. Both precision and recall are therefore based on relevance.

5.7.3 F-Measure

- F-measure or F-score is a measure of accuracy of a model on a data set. A measure that combines precision and recall. It is the harmonic mean of precision and recall, the traditional F-measure or balanced F-score.
- The F-score is used for evaluating information retrieval systems such as search engines, an also in natural language processing.
- It is calculated using,

$$F = 2 - \frac{\text{Precision} - \text{Recall}}{\text{Precision} + \text{Recall}}$$

5.7.4 Confusion Matrix

- A Confusion Matrix describes the accuracy of the solution to a classification problem. A confusion matrix is a table that is often used to describe the performance of a classification model.
- Given m classes, a confusion matrix is an m × m matrix where row represents the **actual truth labels**, and the column represents **the predicted labels**. It is also known as the error matrix. The matrix consists of predictions result in a summarized form, which has a total number of correct predictions and incorrect predictions.
- The matrix looks like as below table:

Table 5.3: Confusion Matrix

	Actual Positive	Actual Negative
Predictive positive	True positive (TP)	False Positive (FP)
Predictive Negative	False Negative (FN)	True Negative (TN)

- Accuracy is calculated by,

$$\text{Accuracy} = \frac{TP + TN}{\text{Total Population}}$$

5.7.5 Cross Validation

- Cross Validation is a standard tool used to develop and fine-tune data mining models. In Cross Validation, we train our model using the subset of the data set and then evaluate using the complementary subset of the data set.
- The steps involved in Cross Validation are as follows:
  1. Reserve some portion of sample data set.
  2. Using the rest data set, train the model.
  3. Test the model using the reserve portion of the data set.
- Cross Validation has the following applications:
  - Validating the robustness of a particular mining model.
  - Evaluating multiple models from a single statement.
  - Building multiple models and then identifying the best model based on statistics.

5.7.6 Bootstrap

- The bootstrap method involves iteratively re-sampling a data set with replacement. The bootstrap method samples the given training tuples uniformly with replacement. That is, each time a tuple is selected, it is equally likely to be selected again and re-added to the training set. There are several bootstrap methods.
- The major application of bootstrapping includes repeated sampling methods to build a more confident measurement.

5.8 DATA MINING TECHNIQUES

- There are many data mining techniques organizations can use to turn raw data into actionable insights. Some of them are given below:

1. **Statistical techniques:**

- Statistical techniques are at the core of most analytics involved in the data mining process. The different analytics models are based on statistical concepts, which output numerical values that are applicable to specific business objectives. For example, neural networks use complex statistics based on different weights and measures to determine if a picture is a dog or a cat in image recognition systems.

2. **Classification:**

- This technique is used to obtain important and actual information about data and metadata.
- It is considered to be a complex data method among other data mining techniques. Information is classified into different classes.
- For example, credit customers can be classified according to three risk categories: "low", "medium", or "high".

3. **Clustering:**

- In this technique, the pieces of information are grouped according to their similarities. This technique helps to recognize the differences and similarities between the data. For example, different groups of customers are clustered together to find similarities and dissimilarities between the parts of information about them.

4. **Regression:**

- This data mining tool is designed to identify and analyze the interactions between different variables. It's used for identification of the probability of a particular variable from other variables' existence. This method is also known as predictive power.
- Regression analysis is also used to predict the future value of a specific entity (the given feature could be either linear or nonlinear). Regression techniques are quite advantageous, due to the power of neural networks which is a unique method that emulates the neural signals in the brain. Ultimately the goal of regression is to show the links between two pieces of information in one set.

5. **Association:**

- This mining data technique is used to find an association between two or more events or properties. It drills down to an underlying model in the database systems.

6. **Outer detection (Outlier analysis):**

- This type of data mining technique relates to the observation of data items in the data set, which do not match an expected pattern or expected behavior. This technique may be used in various domains like intrusion, detection, fraud detection, etc. It is also known as Outlier Analysis or Outlier Mining.

7. **Prediction:**

- Prediction is considered to be an essential data mining technique. It used a combination of other data mining techniques such as clustering, classification, etc. To predict a future event, it analyzes instances or past events in the right sequence.

8. **Sequential patterns:**

- This technique of data mining helps to discover or recognize similar patterns in transaction data over some time. For example, it comprises of finding interesting subsequence in a set of sequences, where the stake of a sequence can be measured in terms of different criteria like length, occurrence frequency, etc.

9. **Decision trees:**

- Decision trees are a specific type of predictive model that lets organizations effectively mine data. This type of data mining tool is used quite often as it's the simplest for understanding. At the root of such decision trees, there is a simple question with many possible answers. Based on the responses, we can get the final answer to the central question. For example, we can attempt to respond to the following question: Should we play cricket today?

5.9 FREQUENT ITEMSETS AND ASSOCIATION RULE MINING: APRIORI ALGORITHM, FP TREE ALGORITHM

- One of the important data mining techniques is Association rules. Association Mining is the method for finding frequent items in the data set. In association mining, the associations and correlations between item sets in transactional and relational databases are found. In other words, association mining finds out which itemset appear together in a transaction. Itemset are the group of items that appear together in a transaction or record.
- The most common model of association rule mining is the Market-basket model. The market-basket model of data is used to describe a common form of many to many relationships between two kinds of objects.

5.9.1 Frequent Itemset

- The common method to find out the association rule in the mining task is to break up the problem into two parts:
  1. Find the large itemset.
  2. Generate rules from the frequent itemset.
- So, we can define the Large/Frequent Itemset as,

  "A Large/Frequent itemset is an itemset whose number of occurrences is above the threshold."

Large Itemset Algorithm:

Input:

```
D // Database of transactions
I // Items
L // Large itemsets
S // Support
α // Confidence
```

Output:

```
R // Association Rules satisfying s and α
```

ARGen algorithm:

```
1. R = ϕ;
2. for each l ∈ L do
3. for each x ⊂ l such that x ≠ ϕ do
4. if support(l) >= α then
5. support(x)
6. R=R U { x ==> (l - x)};
```

Example of Large Itemset (ARGen Algorithm):

L = {{Biscuit}, {Bread}, {Milk}, {Butter}, {Bread, Butter}}

Consider l = {Bread, Peanut Butter}

There are two non-empty subsets of l: {Bread} and {Butter}

With first non-empty subset,

$$\frac{\text{Support }(\{\text{Bread, Butter}\})}{\text{Support }(\{\text{Bread}\})} = \frac{60}{80} = 0.75$$

- The above example states that confidence of association rule is `Bread=>Butter` is 75% which is greater than so it is valid association rule added to R.

5.9.2 Association Rule Mining

- Use of data mining is to extract useful patterns from data irrespective of type of data. Finding patterns is nothing but finding relationships among data. These relationships help users to create groups of data items.
- The kind of purchasing items in group denotes some kind of relationships among them. This relationship in data mining is called as association rule.
- These association rules are often used in marketing, advertising, inventory applications and in retail business.
- It is also used in fault prediction in telecommunication networks. Association rules are used to show the relationships between data items. Association rules are used to detect common usage in purchasing items.
- The existence of item in a transaction is best captured by a likelihood measure or a probability.
- For example, consider patient's medical data set which consists of patient symptoms and illness that a patient suffered from. When association rule is applied to this data set doctor can find out any correlation among the symptoms and illness.
- Following example will give a clear idea about association rule mining.
- **Example 1:** A Grocery store keeps record of items purchased by customer at bill counter. The manager receives transaction report summary. This summary contains different types of items and their quantity. Such transaction report summary will be generated periodically. Manager observed that the number of times Butter is purchased at the same number of times Bread is purchased. The percentage of Butter and Bread is 100%. It is also found that 33.3% of the time Butter is purchased, Jelly is also purchased. However, Butter exists in only about 50% of the overall transactions.
- The database should be in the form of tuples where association rule are to be found. Each tuple is the list of items purchased at one time.
- The support of an item (or set of items) is the percentage of transactions in which that item (or items) occurs.
- Consider Table which shows sample transaction database of a Grocery shop:

Table 5.4: Sample Database for Example 1

Transaction	Items
t_1	Bread, Jelly, Butter
t_2	Bread, Butter
t_3	Bread, Milk, Butter
t_4	Biscuit, Bread
t_5	Biscuit, Milk

For above Example 1, Support will be calculated as follows:

Table 5.5: Count of Support for Example 1

Set	Support	Set	Support
Biscuit	40	Biscuit, Bread, Milk	0
Bread	80	Biscuit, Bread, Butter	0
Jelly	20	Biscuit, Jelly, Milk	0
Milk	40	Biscuit, Jelly, Butter	0
Butter	60	Biscuit, Milk, Butter	0
Biscuit, Bread	20	Bread, Jelly, Milk	0
Biscuit, Jelly	0	Bread, Jelly, Butter	20
Biscuit, Milk	20	Bread, Milk, Butter	20
Biscuit, Butter	0	Jelly, Milk, Butter	0
Bread, Jelly	20	Biscuit, Bread, Jelly, Milk	0
Bread, Milk	20	Biscuit, Bread, Jelly, Butter	0
Bread, Butter	60	Biscuit, Bread, Milk, Butter	0
Jelly, Milk	0	Beer, Jelly, Milk, Butter	0
Jelly, Butter	20	Bread, Jelly, Milk, Butter	0
Milk, Butter	20	Biscuit, Bread, Jelly, Milk, Butter	0
Biscuit, Bread, Jelly	0		

Definition: Given a set of items $I = \{I_1, I_2, \ldots, I_n\}$ and a database of transactions $D = \{t_1, t_2, \ldots, t_n\}$ where, $t_i = \{I_{i1}, I_{i2}, \ldots, I_{ik}\}$ and $I_{ij} \in I$, an association rule is an implication of the form $X \Rightarrow Y$ where $X, Y \subset I$ are sets of items called item sets and $X \cap Y = \phi$.

- In simple words, if $\{i_1, i_2, \ldots, i_k\} \rightarrow j$ means: if a basket contains all of $i_1, \ldots, i_k$ then it is likely to contain j.
- The support(s) for an association rule $X \Rightarrow Y$ is the percentage of transactions in the database that contain $X \cup Y$.
- The confidence or strength(α) for an association rule $X \Rightarrow Y$ is the ratio of the number of transactions that contain $X \cup Y$ to the number of transactions that contain X.

Example 2: Example of rule, support, confidence and lift.

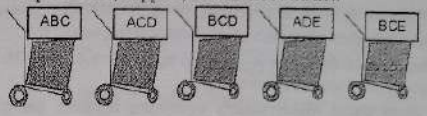


Fig. 5.7: Market basket analysis for Example 2

Table 5.6: Rule, Support and Confidence for Example 2

Rule	Support	Confidence	Lift
$A \Rightarrow D$	2/5	2/3	10/9
$C \Rightarrow A$	2/5	2/4	5/6
$A \Rightarrow C$	2/5	2/3	5/6
$B\&C \Rightarrow D$	1/5	1/3	5/6

Example 3: Basket contains the items as follows:

$B_1 = \{a, b, c\}$ $B_2 = \{a, b, j\}$

$B_3 = \{a, b\}$ $B_4 = \{c, j\}$

$B_5 = \{a, b, c\}$ $B_6 = \{a, b, c, j\}$

$B_7 = \{c, b, j\}$ $B_8 = \{b, c\}$

Using association rule we can say $\{a, b\} \rightarrow c$

Confidence of rule: $Conf(X \Rightarrow Y) = supp(X \cup Y)/supp(X)$.

- For above example, Confidence = 2/4 = 50%.
- Selection of association rules is based on support and confidence. Confidence measures strength of the rule. Support measures frequency of its occurrence in database. Generally large confidence values and smaller support is used. For example, look at table 5.5 No Bread $\Rightarrow$ Butter it has α = 75% (this rule holds 75% of time i.e. 3/4 times Bread occurs as Butter).
- Jelly $\Rightarrow$ Milk (this rule does not hold because they were not purchased together). So, an advertising agency will advertise for bread and butter by keeping some discount on any one of them to increase the sale of that particular product.
- Efficiency of association rule algorithms depends on number of times database is scanned and maximum number counted itemsets.

5.9.3 Apriori Algorithm

- This is one of the well-known best algorithms for generating association rules.
- It is powerful algorithm for mining frequent itemsets for Boolean association rules.
- Name of the algorithm is based on the priority it uses i.e. Apriori.
  - Apriori property based on fact that it uses prior knowledge of frequent itemset properties.

- This property uses iterative approach as level wise search.
- At any level k itemsets are used to explore (k + 1) itemsets.
- At first step, whole database is scanned and count of each individual item is found. Assume minimum support.
- Consider those items which satisfy minimum support. Set of such frequent itemset is found.
- The resulting set is denoted as L_1 (Level 1).
- L_1 is used to find L_2 (L_2 is the frequent itemsets 2).
- L_2 is used to find L_3 and the process continues till no more frequent k-itemsets can be found.
- Every time database has to be scanned for find L_k frequent itemset.

Apriori Property (Large Itemset Property):

- Any subset of a large itemset must be large.

  OR
- All nonempty subsets of a frequent itemset must also be frequent.

- Apriori employs an iterative approach known as level-wise search, where k-itemsets are used to explore k + 1-itemsets.
  - Initially, scan DB once to get frequent 1-itemset.
  - Generate length (k + 1) candidate itemsets from length k frequent itemsets.
  - Test the candidates against DB.
  - Terminate when no frequent or candidate set can be generated.
- **Apriori Pruning Principle:** If there is any itemset which is infrequent, its superset should not be generated / tested.

Method:

L_k denotes the set of frequent k-itemsets : Large itemset.

C_k is the superset of L_k : Candidate for Large itemset.

- Apriori Algorithm is a Two-step process is followed consisting of **join** and **prune** actions to generate L_k from L_{k-1}.
  - **Join Step:** Apriori assumes that items within a transaction or itemset are sorted in lexicographic order.

    The Candidate set C_k is generated by taking the join $L_{k-1} \times L_{k-1}$, where members of L_{k-1} are joinable if their first k–2 items are in common. This ensures that no duplicates are generated.
  - **Prune step:** To reduce the size of C_k, Apriori property is used as follows:
    - Any (k–1)-itemset that is not frequent cannot be a subset of a frequent k-itemset. Hence if any (k–1)-subset of a candidate k-itemset is not in L_{k-1}, the candidate cannot be frequent and can be removed from C_k.
    - The count of each candidate in C_k is used to determine L_k (minimum support count).

Algorithm Apriori_generate(L_k):

1. for each itemset l_1 in L_k
2. for each itemset l_2 in L_k
3. If k – 1 elements in l_1 and l_2 are equal
   // If $l_1[1] = l_2[1]$ and $l_1[2] = l_2[2]$ and ... $l_1[k-1] = l_2[k-1]$ and $l_1[k] < l_2[k]$
4. $C = l_1 \times l_2$
5. add C to C_{k+1}
6. for each k subset s of c
7. if s does not belong to L_k then
8. delete c
9. break

The Apriori Algorithm:

C_k: Candidate itemset of size k

L_k: frequent itemset of size k

L_1 = {frequent items};

1. for($k = 1$; $L_k \neq \emptyset$; k++) do
2. begin
3. C_{k+1} = Apriori_generate(L_k)
   // candidates generated from L_k;
4. for each transaction t in database do
5. increment the count of all candidates in C_{k+1}
6. that are contained in t
7. L_{k+1} = candidates in C_{k+1} with min_support
8. end
9. return $\cup_k L_k$;

Example 4: Consider the database shown in Table 5.7 where supmin=2. Apply Apriori algorithm and find frequent itemsets.

Table 5.7: Sample Database for Example 11

Tid	Items
10	A, B, E
20	B, E
30	B, C
40	A, B, D
50	A, C
60	B, C
70	A, C
80	A, B, C, E
90	A, B, C

Solution:

Tid	Items
20	B, E
30	B, C
40	A, B, D
50	A, C
60	B, C

C_1 — 1st scan →

Itemset	sup
{A}	6
{B}	7
{C}	6
{D}	1
{E}	3

L_1 →

Itemset	sup
{A}	6
{B}	7
{C}	6
{E}	3

L_1

Itemset	sup
{A}	6
{B}	7
{C}	6
{E}	3

$L_1 \times L_1$ →

C_2

Itemset	sup
{A, B}	4
{A, C}	4
{A, E}	2
{B, C}	4
{B, E}	3
{C, E}	1

2nd scan →

L_2

Itemset	sup
{A, B}	4
{A, C}	4
{A, E}	2
{B, C}	4
{B, E}	3

L_2

Itemset	sup
{A, B}	4
{A, C}	4
{A, E}	2
{B, C}	4
{B, E}	3

$L_2 \times L_2$ →

C3 = {{A, B, C}, {A, B, E}, {A, C, E}, {B, C, E}}
The 2 – item subsets of {A, B, C} are {A, B}, {B, C}, {A, C} which are all in L2
The 2 – item subsets of {A, B, E} are {A, B}, {B, E}, {A, E} which are all in L2
The 2 – item subsets of {A, C, E} are {A, C}, {C, E} and {A, E}, {C, E} are not in L2
Remove {A, C, E}
The 2 – item subsets of {B, C, E} are {B, C}, {C, E} and {B, E}, {C, E} are not in L2
Remove {B, C, E}

C_3

Itemset
{A, B, C}
{A, B, E}
{A, C, E}
{B, C, E}

3rd scan →

L_3

Itemset	sup
{A, B, C}	2
{A, B, E}	2

Itemset
{A, B, C}
{A, B, E}

$L_3 \times L_3$ →

$C_4 = \{\{A, B, C, E\}\}$
The 3 – item subsets of {A, B, C, E} are {A, B, C}, {B, C, E}, {A, C, E} and {A, B, E},
{B, C, E} and {A, C, E} are not in L_3
Remove {A, B, C, E}
Thus, C_4 is empty and algorithm terminates.

(Note: In above example, sup = support count)

Time Complexity of Algorithm:

- $O(2^d)$ where d: Total number of unique items in the transaction data set.

Advantages of Apriori Algorithm:

1. This algorithm uses breadth-first search.
2. Easy to implement.
3. Uses large itemset (Apriori) property.

Disadvantages of Apriori Algorithm:

1. This algorithm scans the database multiple times for generating candidate sets.
2. Apriori algorithm requires huge memory space as they deal with a large number of candidate itemset generations. So, space complexity is high.
3. Apriori algorithm execution time is more wasted in producing candidates every time.

- There are different variations of Apriori algorithm to remove its drawbacks. Few of them are Partition algorithm, Sampling algorithm, Dynamic Itemset Counting (DIC) and Direct Hashing and Pruning (DHP).

5.9.4 FP-tree Algorithm

- FP-tree (Frequent Pattern tree) is an algorithm for mining frequent itemsets from a database by using association rules. It's an alternative to the apriori algorithm. A frequent pattern is generated without the need for candidate generation.
- FP growth algorithm represents the database in the form of a tree called a frequent pattern tree or FP tree.
- Frequent Pattern Tree is a tree-like structure that is made with the initial itemset of the database. The purpose of the FP tree is to mine the most frequent pattern. Each node of the FP tree represents an item of the itemset.

Input:

- *D*, a transaction database;
- *min_sup*, the minimum support count threshold.

Output: The complete set of frequent patterns.

Method:

1. The FP-tree is constructed in the following steps:
   (a) Scan the transaction database D once. Collect F, the set of frequent items, and their support counts. Sort F in support count descending order as L, the list of frequent items.
   (b) Create the root of an FP-tree, and label it as "null". For each transaction *Trans* in D do the following. Select and sort the frequent items in *Trans* according to the order of L. Let the sorted frequent item list in *Trans* be [p|P], where p is the first element and P is the remaining list. Call *insert_tree([p|P], T)*, which is performed as follows. If T has a child N such that *N.item-name = p.item-name*, then increment N's count by 1; else create a new node N, and let its count be 1, its parent link be linked to T, and its node-link to the nodes with the same *item-name* via the node-link structure. If P is nonempty, call *insert_tree (P, N)* recursively.
2. The FP-tree is mined by calling FP growth (FP tree, null), which is implemented as follows:

```
procedure FP_growth(Tree, α)
1.  if Tree contains a single path P then
2.      for each combination (denoted as β) of the nodes in the path P
3.      generate pattern β ∪ α with support_count = minimum support count of
    nodes in β;
4.  else for each ai in the header of Tree {
5.      generate pattern β = ai ∪ α with support_count = ai.support_count;
6.      construct β' s conditional pattern base and then β's conditional FP_tree
    Treeβ;
7.      if Treeβ ≠ Ø then
8.              call FP_growth(Treeβ, β;) }
```

- **FP growth**: Mine frequent itemset using an FP-tree by pattern fragment growth.

Example:

- Let's consider the example of FP growth algorithm:

```
['Milk', 'Onion', 'Nutmeg', 'Kidney Beans', 'Eggs', 'Yogurt'],
['Dill', 'Onion', 'Nutmeg', 'Kidney Beans', 'Eggs', 'Yogurt'],
['Milk', 'Apple', 'Kidney Beans', 'Eggs'],
['Milk', 'Unicorn', 'Corn', 'Kidney Beans', 'Yogurt'],
['Corn', 'Onion', 'Onion', 'Ice cream', 'Eggs']
```

Step 1 : Each item has a frequency of occurring; the number of occurrences will increase the item's support. Here we are talking minimum support 3 or 0.6 of any item for the whole data set.

Step 2 : We can see in the transactions that kidney beans are occurring in every transaction, so, at the end of the structure, its support will be 5, but for the first transaction, its support will be 1. So in this step, we reset the item's order by their support degree for the whole data set.

```
['Kidney Beans', 'Eggs', 'Yogurt', 'Onion', 'Milk',],
['Kidney Beans', 'Eggs', 'Yogurt', 'Onion'],
['Kidney Beans', 'Eggs', 'Milk'],
['Kidney Beans', 'Yogurt', 'Milk'],
['Eggs', 'Onion']
```

As we have discussed support degree will be 3 or 0.6 for this data set

Step 3 : Insert the first transaction of the data in a chart like in the picture:

5.10 GRAPH MINING: FREQUENT SUB-GRAPH MINING

Graph Mining:

- *Graph Mining* is the set of tools and techniques used to:
  (a) analyze the properties of real-world graphs.
  (b) predict how the structure and properties of a given graph might affect some application.
  (c) develop models that can generate realistic graphs that match the patterns found in real-world graphs of interest.

Frequent Sub-graph Mining:

- It is the process of finding graph structures that occur in a significant number of times among a set of graphs.

Example: In the example given, frequent sub-graph mining is the process of finding the occurrences of O-H bonds in the structure.

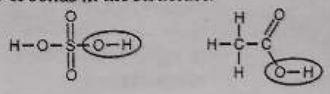


Fig. 5.8: Frequent Sub-Graph Mining

- Other applications of the sub-graph mining are:
  - Finding common biological pathways among species.
  - Recurring patterns of human interaction during an epidemic.
  - Highlighting similar data to reveal the data set as a whole.

5.11 SOFTWARE FOR DATA MINING: R, WEKA, SAMPLE APPLICATIONS OF DATA MINING

- A Data mining tool is a software application that is used to discover patterns and trends from large sets of data and transform those data into more refined information. It helps you to identify unsuspected relationships amongst the data for business growth. It also allows you to analyze, simulate, plan and predict data using a single platform.
- Data Mining is the set of techniques that utilize specific algorithms, statistical analysis, artificial intelligence, and database systems to analyze data from different dimensions and perspectives. Data Mining tools have the objective of discovering patterns/trends/groupings among large sets of data and transforming data into more refined information.
- It is a framework, such as RStudio or Tableau that allows you to perform different types of data mining analysis.
- We can perform various algorithms such as clustering or classification on your data set and visualize the results itself. It is a framework that provides us better insights for our data and the phenomenon that data represents. Such a framework is called a data mining tool.

5.11.1 R

- R is an open-source programming tool developed by Bell Laboratories. R is a **programming language** and an environment for statistical computing and graphics.
- It is compatible with UNIX platforms, FreeBSD, Linux, macOS, and Windows operating systems. R is popular for data mining as it is used to run a variety of statistical analysis, such as time-series analysis, clustering, and linear and non-linear modelling.
- R also supplies excellent data mining packages. Overall, R also offers graphical facilities for data analysis. The applications of R also include statistical computing, analytics, and machine learning tasks.

5.11.2 Weka

- Weka is a collection of machine learning algorithms for data mining tasks. It is open-source software that provides tools for data pre-processing, implementation of several Machine Learning algorithms. The algorithms can either be applied directly to a data set or called from your own Java code.
- Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes.
- Weka is comprehensive software that lets you pre-process the big data, apply different machine learning algorithms on big data and compare various outputs. This software makes it easy to work with big data and train a machine using machine learning algorithms

5.11.3 Sample Applications of Data Mining

- Data mining is used by many organizations to improve the customer base. They focus on customer behavioral patterns, market analysis, profit areas and product improvement. The essential areas where data mining is used are as follows:
  - (a) **Education:** Educational data mining deals with developing the methods to discover the knowledge from the education field. It is used to find out/project students' areas of interest, future learning capacities and other aspects. Educational institutions can apply different data mining techniques and take appropriate/accurate decisions based on the outcome of the mining process. Also, the analysis of slow and fast learners and accordingly their teaching pattern can be determined.
  - (b) **Health and Medicine:** Data mining can be effectively used in health care systems. During Covid-19 pandemic, the predictions of the Covid-19 waves and the volume of patients was done using data mining. In Genetics, data mining also helps in determining the sequence of the genes and future trends.
  - (c) **Market Analysis:** Market analysis is based on a particular pattern of purchase followed by customers. These patterns help the shop owner to understand the buying pattern of customers and accordingly useful decisions can be implemented so as to increase the profit of the store. Also, the market analysis helps to find out the different methodologies to retain the existing customers and gain new ones.
  - (d) **Fraud Detection:** A fraud detection system helps in finding out the pattern of fraud, its potential attackers/criminal detection and possible solutions using different data mining algorithms. These data mining methods provide timely and efficient solutions for detection and prevention of the frauds. Intrusion and lie detection can also be addressed by these mechanisms.

5.12 INTRODUCTION TO TEXT MINING, WEB MINING, SPATIAL MINING, TEMPORAL MINING

5.12.1 Text Mining

- Text mining (also known as text analysis), is the process of transforming unstructured text into structured data for analysis. Text mining uses Natural Language Processing (NLP), allowing machines to understand the human language and process it automatically. Then By applying advanced analytical techniques, such as Naïve Bayes, Support Vector Machines (SVM), and other deep learning algorithms, companies are able to explore and discover hidden relationships within their unstructured data.

- In the era of information, text is one of the most common data types within databases. Depending on the database, this data can be organized as:
  - **Structured data:** This data is standardized into a tabular format with numerous rows and columns. Structured data can include inputs such as names, addresses, and phone numbers.
  - **Unstructured data:** This data does not have a predefined data format. It can include text from sources, like social media or product reviews, or rich media formats like video and audio files.
  - **Semi-structured data:** As the name suggests, this data is a mixture of structured and unstructured data formats. Examples of semi-structured data include XML, JSON.
- The applications of text mining includes the risk management, maintenance, healthcare and spam filtering

5.12.2 Web Mining

- As tremendous amount of data is being generated daily on the web, the mining of this data is very essential. Web mining refers to the mining of data related to World Wide Web. This data contains the actual data present on web as well as the data related to web. Web data can be classified into following categories:
  - Content of actual web page.
  - Inter-page structure containing actual linkage structure between web pages.
  - Intra-page structure containing HTML or XML code.
  - Web page access log.
  - User profiles.

Tasks of Web Mining:

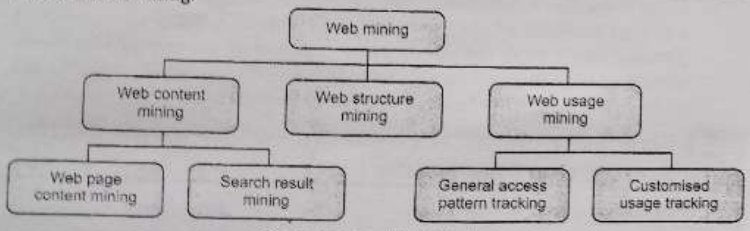


Fig. 5.9: Tasks of Web Mining

- Web mining tasks can be divided into number of tasks. Above figure shows the different tasks included in web mining.
  1. **Web Content Mining**: Web content mining is the process of examining the contents of a web page as a result of web searching activity. The contents include web pages as well as graphics data. Web content mining is categorized into web page content mining and search result mining.

  2. **Web Structure Mining:** Web structure mining is the process of discovering structure information from the web. It deals with creating a model of the web organizations. This can be used to classify the web pages or to create similarity measures between the documents. The structure of the web graph consists of web pages as nodes, and hyperlinks as edges connecting related pages. Structure mining shows the structured summary of a particular website. It identifies relationships between web pages linked by information or direct link connection.
  3. **Web Usage Mining:** Web usage mining is used for mining the web log records. In other words, the information of the web page based on the access data of is mined. It helps to discover the user access patterns of web pages. Web server keeps a tract/log of every web page. After that, based on this data record, the potential patterns are found to capture the customers.
- Web server registers a web log entry for every web page.
- Analysis of similarities in web log records can be useful to identify the potential customers for e-commerce companies.
- Web usage mining is further divided into General Access Pattern Tracking and Customized Usage Tracking.

Application of Web Mining:

- Web mining has various applications due to various uses of the web. Following is the list of some applications of web mining:
  - Marketing and conversion tool.
  - Data analysis on website and application accomplishment.
  - Audience behavior analysis.
  - Advertising and campaign accomplishment analysis.
  - Testing and analysis of a site.

5.12.3 Spatial Mining

- Spatial data are the data about objects that are located in a physical space. This includes the data related to space and including maps. Spatial mining is the process of application of data mining to spatial data. In spatial mining, geographic or spatial information is used to produce the results.
- In Spatial mining the extraction of knowledge, spatial relationships, or other interesting patterns stored in spatial databases is done. The application of spatial mining is for learning spatial records, discovering spatial relationships and relationships among spatial and non-spatial records, constructing spatial knowledge bases, reorganizing spatial databases, and optimizing spatial queries. The spatial mining is also applicable in the areas of GIS systems, Geology, Environmental Science and Robotics.
- Many data mining techniques are directly applicable to spatial data but there are some new techniques and tools that are applicable specifically to the spatial data.

- Spatial data is more complex as compared to the non-spatial data. Hence there are specialized operations and data structures used to access special data. Spatial queries like spatial join, range query, a nearest neighbor query and a distance scan and spatial data structures like quad tree, R-tree, K-d Tree are used to store the spatial data.

5.12.4 Temporal Mining

- A **temporal** database stores data relating to time instances. Temporal Data Mining is a single step in the process of Knowledge Discovery in Temporal Databases that enumerates structures (temporal patterns or models) over the temporal data, and any algorithm that enumerates temporal patterns from, or fits models to, temporal data is a Temporal Data Mining Algorithm.
- Temporal Data Mining often involves processing time series, typically sequences of data, which measure values of the same attribute at a sequence of different time points.
- Temporal data mining tasks include:
  - Temporal data characterization and comparison.
  - Temporal clustering analysis.
  - Temporal classification.
  - Temporal association rules.
  - Temporal pattern analysis.
  - Temporal prediction and trend analysis.

Summary

- Data Mining deals with discovery of hidden knowledge, unexpected patterns, and new rules from large data sets.
- Following issues were faced while doing data mining: Human Interaction, Overfitting, Outliers, Interpretation of result, Visualization of result, large data sets, High Dimensionality, Multimedia Data, Missing data, Irrelevant data, Noisy data, Changing Data, Integration, Application.
- Knowledge Discovery in Databases (KDD) is the process of finding useful information and patterns in data.
- The KDD process is divided into steps: Selection, Pre-processing, Transformation, Data Mining, Measuring, Interpretation/Visualization.
- Data mining tasks are Predictive and Descriptive.
- Predictive tasks include classification, regression, time series analysis and prediction.
- Descriptive tasks include clustering, summarization, association rules and sequence discovery.

- Knowledge representation methods are graphical, geometric, icon based, pixel based and hierarchical and hybrid.
- Applications of data mining include education, health, finance and fraud detection.
- Data pre-processing is the process of cleaning the data and making it useful for the process of mining.
- The steps in data pre-processing are Data cleaning, Data transformation and Data reduction.
- Data reduction is a process that reduced the volume of original data and represents it in a much smaller volume.
- R and Weka are popular data mining tools.
- Spatial mining is the process of application of data mining to spatial data.
- Temporal Data Mining is a single step in the process of Knowledge Discovery in Temporal Databases.

Check Your Understanding

1. Data mining is also called _____.
   (a) Data Processing (b) Data Discovery
   (c) Knowledge discovery (d) Knowledge Processing.
2. Which of the following is an essential process in which the intelligent methods are applied to extract data patterns?
   (a) Warehousing (b) Data Mining
   (c) Text Mining (d) Data Selection
3. What are the functions of Data Mining?
   (a) Association and correctional analysis classification.
   (b) Prediction and characterization.
   (c) Cluster analysis and Evolution analysis.
   (d) All the above
4. Which of the following statements is correct about data mining?
   (a) It can be referred to as the procedure of mining knowledge from data.
   (b) Data mining can be defined as the procedure of extracting information from a set of the data.
   (c) The procedure of data mining also involves several other processes like data cleaning, data transformation, and data integration.
   (d) All of the above.
5. _____ is not data mining functionality?
   (a) Clustering and Analysis (b) Selection and Interpretation
   (c) Classification and Regression (d) Characterization and Discrimination

6. _____ is the output of KDD.
   (a) Query (b) Useful Information
   (c) Data (d) information
7. The analysis performed to uncover interesting statistical correlations between associated-attribute-value pairs is called _____.
   (a) Mining of Association (b) Mining of Clusters
   (c) Mining of Correlation (d) None of the above
8. What does Apriori algorithm do?
   (a) It mines all frequent patterns through pruning rules with lesser support.
   (b) It mines all frequent patterns through pruning rules with higher support.
   (c) Both (a) and (b).
   (d) None of these.
9. What does FP growth algorithm do?
   (a) It mines all frequent patterns through pruning rules with lesser support.
   (b) It mines all frequent patterns through pruning rules with higher suppor.t
   (c) It mines all frequent patterns by constructing a FP tree.
   (d) All of these.
10. What techniques can be used to improve the efficiency of Apriori algorithm?
    (a) Hash Based Techniques (b) Transaction Reduction
    (c) Partitioning (d) All of these
11. A Decision Tree is a _____ model.
    (a) Predictive (b) Descriptive
    (c) Deterministic (d) Process

Answers

1. (c)	2. (b)	3. (d)	4. (d)	5. (b)	6. (b)	7. (c)	8. (a)	9. (a)	10. (d)	11. (a)

Practice Questions

Q.I Answer the following questions in short.

1. Define Data Mining.
2. List out steps of the KDD process.
3. What are the types of data?
4. Compare descriptive and predictive data mining.
5. What is prediction?
6. Why do we need data pre-processing?
7. What is Data integration?
8. List out the tasks of data mining.
9. What is Graph mining?

Q.II Answer the following questions.

1. What is Data Cleaning? Describe various methods of Data Cleaning.
2. Discuss about the major issues in Data Mining.
3. Explain various accuracy measures in the data mining.
4. Describe techniques of Data Mining?
5. Differentiate between Data Verification and Discovery.
6. Explain FP-tree Algorithm.
7. Write the difference between Data Mining and Knowledge Discovery databases.

Q.III Define the terms.

1. Web mining
2. Weka
3. F-measure
4. Time Series Analysis
5. Data Cleaning
6. Linear regression.

6... Spark

Learning Objectives ...

- To learn about Spark Installation.
- To get information of Apache Spark Architecture.
- To know about Components of Spark.
- To study Spark RDDs and RDD Operations-Transformation and Actions.
- To get knowledge about Spark SQL and Data Frames.
- To know about Kafka for Spark Streaming.

6.1 INTRODUCTION

- As the data size in the industries is growing day by day, organizations are using Hadoop to analyze the data sets. The reason behind using Hadoop is that it is based on simple programming model i.e. MapReduce.
- Also, one more reason is that Hadoop enables scalable, feasible, flexible and fault tolerant solutions which are cost effective as well. In this process, the main concern is to maintain the speed of processing large data sets. Therefore, Apache introduced Spark for speeding up the Hadoop computational computing software process.

6.2 INTRODUCTION TO SPARK

- Apache Spark is an open-source, distributed processing system used for big data workloads. It is cluster computing designed for fast computation.
- It utilizes in-memory caching, and optimized query execution for fast analytic queries against data of any size.
- Spark is a general engine for large-data processing. It is based on Hadoop MapReduce and it extends the MapReduce model to efficiently use it for more types of computations, which includes interactive queries and stream processing.

6.2.1 Features of Apache Spark

- Apache Spark has the following features:
  - **Speed**: The main feature of Spark is its in-memory cluster computing that increases the processing speed of an application. Spark helps to run an application in Hadoop cluster, up to 100 times faster in memory, and 10 times faster when running on disk. This is possible by reducing the number of read/write operations to disk. It stores the intermediate processing data in memory.
  - **Multiple language support:** Spark supports multiple languages. It provides various APIs written in Java, Scala, Python and R.
  - **Multiple platform support:** Spark will run on multiple platforms while not moving the processing speed. It runs on Hadoop, Kubernetes, Mesos, Standalone, and even within the Cloud.
  - **Advanced Analytics**: Spark not only supports 'Map' and 'reduce'. It also supports SQL queries, Streaming data, Machine Learning (ML), and Graph algorithms.

6.2.2 Installation of Spark

- Apache Spark is a framework used in cluster computing environments for analyzing big data. This platform became widely popular due to its ease of use and the improved data processing speeds over Hadoop. It is advisable to install Spark on Linux based systems.
- The prerequisite for Spark installation is that your system should be updated with Java and Scala.
- Following are the steps to install Spark:

Step 1 : Verifying Java installation

- Java version can be verified with the command `$java-version`. If Java is installed in your system then it will show the version of the installed Java and if it is not installed then it is advisable to install the Java on to the system.

Step 2 : Verifying Scala installation

- The Scala installation on to the system can be verified with the command `$scala-version`. If Scala is installed on to the system then it will show you the version of the installed Scala and if it is not install then it is advisable to install it onto your system.

Step 3 : Downloading Spark

- Download the latest version of Spark by visiting the following link `https://spark.apache.org/downloads.html`.
- After downloading it, you will find the Spark tar file in the download folder. After that, follow the steps given below for installing Spark.

Step 4 : Extracting Spark tar

- The following command for extracting the spark tar file.

```
$ tar xvf spark-1.3.1-bin-hadoop2.6.tgz
```

Step 5 : Moving Spark software files

- The following commands for moving the Spark software files to respective directory (/usr/local/spark).

```
$ su -
Password:

# cd /home/Hadoop/Downloads/
# mv spark-1.3.1-bin-hadoop2.6 /usr/local/spark
# exit
```

Step 6 : Setting up the environment for Spark

- Add the following line to ~/.bashrc file. It means adding the location, where the spark software files are located to the PATH variable.

```
export PATH=$PATH:/usr/local/spark/bin
```

- Use the following command for sourcing the ~/.bashrc file.

```
$ source ~/.bashrc
```

Step 7 : Verifying the Spark Installation

- Write the following command for opening Spark shell.

```
$spark-shell
```

- If spark is installed successfully then you will find the following output.

```
Spark assembly has been built with Hive, including Datanucleus jars on classpath
Using Spark's default log4j profile: org/apache/spark/log4j-defaults.properties
15/06/04 15:25:22 INFO SecurityManager: Changing view acls to: hadoop
15/06/04 15:25:22 INFOSecurityManager: Changing modify acls to: hadoop
15/06/04 15:25:22 INFO SecurityManager: SecurityManager: authentication disabled;
uiacls disabled; users with view permissions: Set(hadoop); users with modify permissions: Set(hadoop)
15/06/04 15:25:22 INFO HttpServer: Starting HTTP Server
15/06/04 15:25:23 INFO Utils: Successfully started service 'HTTP class server' on port 43292.
Welcome to
```



```
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /___/ .__/\_,_/_/ /_/\_\   version 1.4.0
      /_/
```

```
Using Scala version 2.10.4 (Java HotSpot(TM) 64-Bit Server VM, Java 1.7.0_71)
Type in expressions to have them evaluated.
Spark context available as sc
scala>
```

6.3 APACHE SPARK ARCHITECTURE

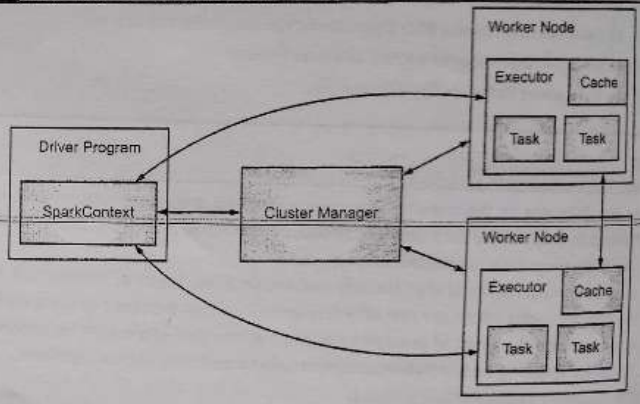


Fig. 6.1: Apache Spark Architecture

- There are several useful things to note about this architecture. These are as follows:
  1. Each application gets its own executor processes, which stay up for the duration of the whole application and run tasks in multiple threads. This has the benefit of isolating applications from each other, on both the scheduling side (each driver schedules its own tasks) and executor side (tasks from different applications run in different JVMs). It means that data cannot be shared across different Spark applications (instances of SparkContext) without writing it to an external storage system.

2. Spark is uncertain to the underlying cluster manager. As long as it can acquire executor processes, and these communicate with each other, it is relatively easy to run it even on a cluster manager that also supports other applications (e.g. Mesos/YARN/Kubernetes).
3. The driver program must listen for and accept incoming connections from its executors throughout its lifetime. As such, the driver program must be network addressable from the worker nodes.
4. Because the driver schedules tasks on the cluster, it should be run close to the worker nodes, preferably on the same local area network. If you'd like to send requests to the cluster remotely, it is better to open an RPC to the driver and have it submit operations from nearby than to run a driver far away from the worker nodes.

6.4 COMPONENTS OF SPARK

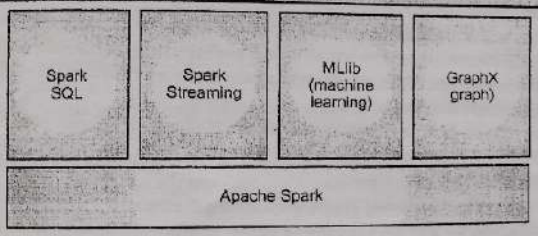


Fig. 6.2: Components of Spark

- The main components of Apache spark are as shown in the figure:

1. **Apache Spark:**
   - Spark Core is the underlying general execution engine for spark platform. All the other functionality is built upon.
   - It provides in-Memory computing and referencing datasets in external storage systems.
2. **Spark SQL:**
   - Spark SQL is a component above Spark Core. It contains a new data abstraction called SchemaRDD.
   - SchemaRDD provides support for structured and semi-structured data. It supports many sources of data including Hive tablets, Parquet, JSON.
3. **Spark Streaming:**
   - Spark Streaming leverages Spark Core's fast scheduling capability to perform streaming analytics. It ingests data in mini-batches and performs RDD (Resilient Distributed Datasets) transformations on those mini-batches of data.

   - Spark streaming permits ascendible, high-throughput, fault-tolerant stream process of live knowledge streams. Spark can access data from a source like a Flume, TCP socket. It will operate different algorithms in which it receives the data in a file system, database and live dashboard.
   - Spark uses **Micro-batching** for real-time streaming. Micro-batching is a technique that permits a method or a task to treat a stream as a sequence of little batches of information. Hence, spark streaming groups the live data into small batches. It delivers it to the batch system for processing. The functionality of this module is:
     (a) Enables processing of live streams of data like log files generated by production web services.
     (b) The API's defined in this module are quite similar to spark core RDD API's.
4. **MLlib (Machine Learning Library):**
   - MLlib is a distributed machine learning framework above Spark because of the distributed memory-based Spark architecture. According to benchmarks, it is done by the MLlib developers against the Alternating Least Squares (ALS) implementations.
   - Spark MLlib is nine times as fast as the Hadoop disk-based version of Apache Mahout (before Mahout gained a Spark interface).
5. **GraphX:**
   - GraphX is a distributed graph-processing framework on top of Spark.
   - It provides an API for expressing graph computation that can model the user-defined graphs by using the Pregel abstraction API.
   - It also provides an optimized runtime for this abstraction.

6.5 SPARK RDDs

Resilient Distributed Datasets (RDD):

- RDD is a fundamental data structure of Apache Spark. It is an immutable collection of objects which computes on the different node of the cluster.
- Decomposing the name RDD:
  - **Resilient:** i.e. fault-tolerant with the help of RDD lineage graph and so able to re-compute missing or damaged partitions due to node failures.
  - **Distributed:** Since Data resides on multiple nodes.
  - **Dataset:** It represents records of the data you work with. The user can load the data set externally which can be either JSON file, CSV file, text file or database via JDBC with no specific data structure.
- Formally, an RDD is a read-only, partitioned collection of records. RDD is a fault-tolerant collection of elements that can be operated in parallel.
- Spark makes use of the concept of RDD to achieve faster and efficient MapReduce operations.
- RDDs can contain any type of Python, Java, or Scala objects, including user-defined classes.

2. Spark is uncertain to the underlying cluster manager. As long as it can acquire executor processes, and these communicate with each other, it is relatively easy to run it even on a cluster manager that also supports other applications (e.g. Mesos/YARN/Kubernetes).
3. The driver program must listen for and accept incoming connections from its executors throughout its lifetime. As such, the driver program must be network addressable from the worker nodes.
4. Because the driver schedules tasks on the cluster, it should be run close to the worker nodes, preferably on the same local area network. If you'd like to send requests to the cluster remotely, it is better to open an RPC to the driver and have it submit operations from nearby than to run a driver far away from the worker nodes.

6.4 COMPONENTS OF SPARK

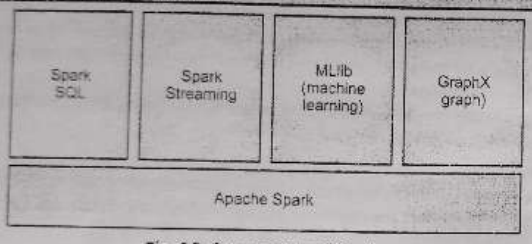


Fig. 6.2: Components of Spark

- The main components of Apache spark are as shown in the figure:

1. **Apache Spark:**
   - Spark Core is the underlying general execution engine for spark platform. All the other functionality is built upon.
   - It provides in-Memory computing and referencing datasets in external storage systems.
2. **Spark SQL:**
   - Spark SQL is a component above Spark Core. It contains a new data abstraction called SchemaRDD.
   - SchemaRDD provides support for structured and semi-structured data. It supports many sources of data including Hive tablets, Parquet, JSON.
3. **Spark Streaming:**
   - Spark Streaming leverages Spark Core's fast scheduling capability to perform streaming analytics. It ingests data in mini-batches and performs RDD (Resilient Distributed Datasets) transformations on those mini-batches of data.

   - Spark streaming permits ascendible, high-throughput, fault-tolerant stream process of live knowledge streams. Spark can access data from a source like a Flume, TCP socket. It will operate different algorithms in which it receives the data in a file system, database and live dashboard.
   - Spark uses **Micro-batching** for real-time streaming. Micro-batching is a technique that permits a method or a task to treat a stream as a sequence of little batches of information. Hence, spark streaming groups the live data into small batches. It delivers it to the batch system for processing. The functionality of this module is:
     (a) Enables processing of live streams of data like log files generated by production web services.
     (b) The API's defined in this module are quite similar to spark core RDD API's.
4. **MLlib (Machine Learning Library):**
   - MLlib is a distributed machine learning framework above Spark because of the distributed memory-based Spark architecture. According to benchmarks, it is done by the MLlib developers against the Alternating Least Squares (ALS) implementations.
   - Spark MLlib is nine times as fast as the Hadoop disk-based version of Apache Mahout (before Mahout gained a Spark interface).
5. **GraphX:**
   - GraphX is a distributed graph-processing framework on top of Spark.
   - It provides an API for expressing graph computation that can model the user-defined graphs by using the Pregel abstraction API.
   - It also provides an optimized runtime for this abstraction.

6.5 SPARK RDDs

Resilient Distributed Datasets (RDD):

- RDD is a fundamental data structure of Apache Spark. It is an immutable collection of objects which computes on the different node of the cluster.
- Decomposing the name RDD:
  - **Resilient:** i.e. fault-tolerant with the help of RDD lineage graph and so able to re-compute missing or damaged partitions due to node failures.
  - **Distributed:** Since Data resides on multiple nodes.
  - **Dataset:** It represents records of the data you work with. The user can load the data set externally which can be either JSON file, CSV file, text file or database via JDBC with no specific data structure.
- Formally, an RDD is a read-only, partitioned collection of records. RDD is a fault-tolerant collection of elements that can be operated in parallel.
- Spark makes use of the concept of RDD to achieve faster and efficient MapReduce operations.
- RDDs can contain any type of Python, Java, or Scala objects, including user-defined classes.

Features of RDD:

- Spark RDD has the following features:
  1. **Immutability:** RDD is immutable. You cannot change the state of RDD. If you want to change the state of RDD, you need to create a copy of the existing RDD and perform your required operations. Hence, the required RDD can be retrieved at any time.
  2. **Fault-tolerant:** Once you perform any operations in an existing RDD, a new copy of that RDD is created, and the operations are performed on the newly created RDD. Thus, any lost data can be recovered easily and recreated. This feature makes Spark RDD fault-tolerant.
  3. **Partitioning:** Each and every dataset in Spark RDD is logically partitioned across many servers so that they can be computed on different nodes of the clusters.
  4. **Persistence:** Intermediate results generated by RDD are stored to make the computation easy. It makes the process optimized.
  5. **Lazy Evaluation:** Transformations in RDDs are implemented using lazy operations. In lazy evaluation, the results are not computed immediately. It will generate the results, only when the action is triggered. Thus, the performance of the program is increased.
  6. **In-memory Computation:** Spark uses in-memory computation as a way to speed up the total processing time. In the in-memory computation, the data is kept in RAM (random access memory) instead of the slower disk drives. This is very helpful as it reduces the cost of memory and increase the computation power.

Methods to create RDDs:

- There are two ways to create RDDs :
  1. **Parallelizing collection (Parallelized):** Parallelizing an existing collection in your driver program.
  2. **Referencing External Dataset:** Referencing a dataset in an external storage system, such as a shared file system, HDFS, HBase, or any data source offering a Hadoop Input format.

Need of RDD in Spark:

- The key motivations behind the concept of RDD are:
  - Iterative algorithms.
  - Interactive data mining tools.
  - DSM (Distributed Shared Memory) is a very general abstraction, but this generality makes it harder to implement in an efficient and fault tolerant manner on commodity clusters. Here the need of RDD comes into the picture.
  - In distributed computing systems data is stored in intermediate stable distributed store such as HDFS or Amazon S3. This makes the computation of a job slower since it involves many I/O operations, replications, and serializations in the process.

6.6 RDD OPERATIONS: TRANSFORMATIONS AND ACTIONS

- RDD provides two types of operations: Transformations and Actions.
  1. **Transformations:** These operations create a new data set from the existing dataset. The transformations are considered lazy as they only computed when an action requires a result to be returned to the driver program.

Table 6.1: Operations in Transformation and its description

Sr. No.	Operations in Transformation	Description
1.	`map(func)`	It returns a new distributed dataset formed by passing each element of the source through a function `func`.
2.	`filter(func)`	It returns a new dataset formed by selecting those elements of the source on which `func` returns true.
3.	`flatMap(func)`	Here, each input item can be mapped to zero or more output items, so `func` should return a sequence rather than a single item.
4.	`mapPartitions(func)`	It is similar to map, but runs separately on each partition (block) of the RDD, so `func` must be of type `Iterator<T> => Iterator<U>` when running on an RDD of type T.
5.	`mapPartitionsWithIndex(func)`	It is similar to `mapPartitions` that provides `func` with an integer value representing the index of the partition, so func must be of type `(Int, Iterator<T>) => Iterator<U>` when running on an RDD of type T.
6.	`sample(withReplacement, fraction, seed)`	It samples the fraction of the data, with or without replacement, using a given random number generator seed.
7.	`union(otherDataset)`	It returns a new dataset that contains the union of the elements in the source dataset and the argument.
8.	`intersection(otherDataset)`	It returns a new RDD that contains the intersection of elements in the source dataset and the argument.
9.	`distinct([numPartitions]))`	It returns a new dataset that contains the distinct elements of the source dataset.

contd. ...

10.	`groupByKey([numPartitions])`	It returns a dataset of `(K, Iterable)` pairs when called on a dataset of `(K, V)` pairs.
11.	`reduceByKey(func, [numPartitions])`	When called on a dataset of `(K, V)` pairs, returns a dataset of `(K, V)` pairs where the values for each key are aggregated using the given reduce function `func`, which must be of type `(V,V) => V`.
12.	`aggregateByKey(zeroValue)(seqOp, combOp, [numPartitions])`	When called on a dataset of (K, V) pairs, returns a dataset of (K, U) pairs where the values for each key are aggregated using the given combine functions and a neutral "zero" value.
13.	`sortByKey([ascending], [numPartitions])`	It returns a dataset of key-value pairs sorted by keys in ascending or descending order, as specified in the boolean ascending argument.
14.	`join(otherDataset, [numPartitions])`	When called on datasets of type (K, V) and (K, W), returns a dataset of (K, (V, W)) pairs with all pairs of elements for each key. Outer joins are supported through `leftOuterJoin`, `rightOuterJoin`, and `fullOuterJoin`.
15.	`cogroup(otherDataset, [numPartitions])`	When called on datasets of type (K, V) and (K, W), it returns a dataset of `(K, (Iterable, Iterable))` tuples. This operation is also called *groupWith*.
16.	`cartesian(otherDataset)`	When called on datasets of types T and U returns a dataset of (T, U) pairs (all pairs of elements).
17.	`pipe(command, [envVars])`	Pipe each partition of the RDD through a shell command, e.g. a Perl or bash script.
18.	`coalesce(numPartitions)`	It decreases the number of partitions in the RDD to numPartitions.
19.	`repartition(numPartitions)`	It reshuffles the data in the RDD randomly to create either more or fewer partitions and balance it across them.
20.	`repartitionAndSortWithinPartitions(partitioner)`	It repartitions the RDD according to the given `partitioner` and, within each resulting partition, sorts records by their keys.

2. **Actions:** It returns a value to the driver program after running a computation on the dataset. The Transformations in Apache Spark create RDDs from each other but to work on actual Dataset, and then we perform Action operations. These operations give non-RDD values as results that are stored on drivers or to the external storage system.

Table 6.2: Operations in Actions and its description

Sr. No.	Operations in Action	Description
1.	`reduce(func)`	It aggregates the elements of the dataset using a function `func` (which takes two arguments and returns one). The function should be commutative and associative so that it can be computed correctly in parallel.
2.	`collect()`	It returns all the elements of the dataset as an array at the driver program. This is usually useful after a filter or other operation that returns a sufficiently small subset of the data.
3.	`count()`	It returns the number of elements in the dataset.
4.	`first()`	It returns the first element of the dataset (similar to `take(1)`).
5.	`take(n)`	It returns an array with the first n elements of the dataset.
6.	`takeSample(withReplacement, num, [seed])`	It returns an array with a random sample of num elements of the dataset, with or without replacement, optionally pre-specifying a random number generator seed.
7	`takeOrdered(n, [ordering])`	It returns the first n elements of the RDD using either their natural order or a custom comparator.
8.	`saveAsTextFile(path)`	It is used to write the elements of the dataset as a text file (or set of text files) in a given directory in the local filesystem, HDFS or any other Hadoop-supported file system. Spark calls `toString` on each element to convert it to a line of text in the file.

contd. ...

9.	saveAsSequenceFile(path) (Java and Scala)	It is used to write the elements of the dataset as a Hadoop SequenceFile in a given path in the local filesystem, HDFS or any other Hadoop-supported file system.
10.	saveAsObjectFile(path) (Java and Scala)	It is used to write the elements of the dataset in a simple format using Java serialization, which can then be loaded using SparkContext.objectFile().
11.	countByKey()	It is only available on RDDs of type (K, V). Thus, it returns a hashmap of (K, Int) pairs with the count of each key.
12.	foreach(func)	It runs a function func on each element of the dataset for side effects such as updating an Accumulator or interacting with external storage systems.

6.7 SPARK SQL AND DATA FRAMES

6.7.1 Spark SQL

- Spark SQL is a Spark module for structured data processing. Internally, Spark SQL uses this extra information to perform extra optimizations.
- There are several ways to interact with Spark SQL including SQL and the Dataset API. Spark SQL runs on top of the Spark Core. It allows developers to import relational data from Hive tables and Parquet files, run SQL queries over imported data and existing RDDs and easily writes RDDs out to Hive tables or Parquet files.
- Spark SQL introduces an extensible optimizer called *Catalyst* as it helps in supporting a wide range of data sources and algorithms in Big data.

Uses of Spark SQL:

- One use of Spark SQL is to execute SQL queries.
- Spark SQL can also be used to read data from an existing Hive installation.
- When running SQL from within another programming language the results will be returned as a Dataset/Data Frame. You can also interact with the SQL interface using the command-line or over JDBC/ODBC.

6.7.2 Data Frames

- A Data Frame is a distributed collection of data, which is organized into named columns. Data frames can be compared to relational databases.
- A Data Frame can be constructed from an array of different sources such as Hive tables, Structured Data files, external databases, or existing RDDs.

Characteristic of Data Frame:

- It has ability to process the data in the size of Kilobytes to Petabytes on a single node cluster to large cluster.
- It supports different data formats (Avro, CSV, Elastic search, and Cassandra) and storage systems (HDFS, HIVE tables, MySQL etc.).
- It has state of art optimization and code generation through the Spark SQL Catalyst optimizer (tree transformation framework).
- It can be easily integrated with all Big Data tools and frameworks via Spark-Core.
- It provides API for Python, Java, Scala, and R Programming.

6.8 INTRODUCTION TO KAFKA FOR SPARK STREAMING

- As we all know that, a tremendous amount of data is being generated every day. To handle and make use of this data is tedious work. The main challenges faced while handling and using this data is: collection of the data and secondly, analysis of the collected data.
- To overcome these challenges, a fast messaging system is required. The messaging system in this context is a system which is responsible for transferring the data from one application to another application. The main focus of the messaging system is on data instead of focusing on how to share the data. Messaging systems are of two types one is Point to Point and second one is Publish-subscribe.
- **Apache Kafka** is a framework implementation of a software bus using stream-processing. It is developed in Scala and Java. Kafka aims to provide a unified, high-throughput, low-latency platform for handling real-time data feeds. It is fast, scalable and distributed by design.
- Apache Kafka is a distributed publish - subscribe messaging system and a robust queue that can handle a high volume of data and enables you to pass messages from one end-point to another. Kafka is suitable for both offline and online message consumption. Kafka messages are persisted on the disk and replicated within the cluster to prevent data loss.
- Kafka is built on top of the ZooKeeper synchronization service. It integrates very well with Apache Storm and Spark for real-time streaming data analysis.

Architecture:

- Kafka stores key-value messages that come from arbitrarily many processes called producers. The data can be partitioned into different "partitions" within different "topics". Within a partition, messages are strictly ordered by their offsets (the position of a message within a partition), and indexed and stored together with a timestamp. Other processes called "consumers" can read messages from partitions. For stream processing, Kafka offers the Streams API that allows writing Java applications that consume data from Kafka and write results back to Kafka.

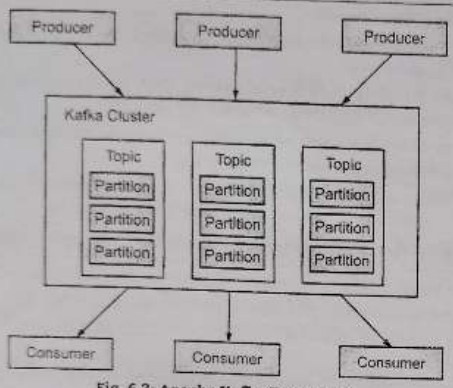


Fig. 6.3: Apache Kafka Framework

Benefits of Kafka:

- **Reliability:** Kafka is distributed, partitioned, replicated and fault tolerance.
- **Scalability:** Kafka messaging system scales easily without down time.
- **Durability:** Kafka uses Distributed commit log which means messages persist on disk as fast as possible, hence it is durable.
- **Performance:** Kafka has high throughput for both publishing and subscribing messages. It maintains stable performance even though many TB messages are stored.

Summary

- Apache Spark is an open-source, distributed processing system used for big data workloads. It is cluster computing designed for fast computation.
- The prerequisite for Spark installation is that your system should be updated with Java and Scala.
- Spark Core is the underlying general execution engine for spark platform. All the other functionality is built upon.
- RDD is the fundamental data structure of Apache Spark which is an immutable collection of objects which computes on the different node of the cluster.
- RDD provides two types of operations:
  - **Transformations** create a new data set from the existing dataset. The transformations are considered lazy as they only computed when an action requires a result to be returned to the driver program.
  - **Actions** return a value to the driver program after running a computation on the dataset.

- Spark SQL is a Spark module for structured data processing. Internally, Spark SQL uses this extra information to perform extra optimizations.
- Apache Kafka is a framework implementation of a software bus using stream-processing. It is developed in Scala and Java.

Check Your Understanding

1. _____ is a component on top of Spark Core.
   (a) Spark Streaming (b) Spark SQL
   (c) RDDs (d) All of the mentioned
2. Spark SQL provides a domain-specific language to manipulate _____ in Scala, Java, or Python.
   (a) Spark Streaming (b) Spark SQL
   (c) RDDs (d) All of the mentioned
3. _____ leverages Spark Core fast scheduling capability to perform streaming analytics.
   (a) MLlib (b) Spark Streaming
   (c) GraphX (d) RDDs
4. _____ is a distributed machine learning framework on top of Spark.
   (a) MLlib (b) Spark Streaming
   (c) GraphX (d) RDDs
5. _____ is a distributed graph processing framework on top of Spark.
   (a) MLlib (b) Spark Streaming
   (c) GraphX (d) All of the mentioned
6. Spark architecture is _____ times as fast as Hadoop disk-based Apache Mahout and even scales better than Vowpal Wabbit.
   (a) 10 (b) 20
   (c) 50 (d) 100
7. Users can easily run Spark on top of Amazon's _____.
   (a) Infosphere (b) EC2
   (c) EMR (d) None of the mentioned
8. Which of the following languages is not supported by Spark?
   (a) Java (b) Pascal
   (c) Scala (d) Python
9. Spark is packaged with higher level libraries, including support for _____ queries.
   (a) SQL (b) C
   (c) C++ (d) None of the mentioned

10. Spark powers a stack of high-level tools including Spark SQL, MLlib for _____.
    (a) regression models (b) statistics
    (c) machine learning (d) reproductive research
11. What are the benefits of Apache Kafka over the traditional technique?
    (a) Fast (b) Scalable
    (c) Durable (d) All of the above
12. Spark is best suited for _____ data.
    (a) Real-time (b) Virtual
    (c) Structured (d) All of the above
13. When was Apache Spark developed?
    (a) 2007 (b) 2008
    (c) 2009 (d) 2010

Answers

1. (b)	2. (c)	3. (b)	4. (a)	5. (c)	6. (a)	7. (b)	8. (b)	9. (a)	10. (c)
11. (d)	12. (a)	13. (c)							

Practice Questions

Q.I Answer the following questions in short.

1. What is spark?
2. What are the features of spark?
3. What is RDD?
4. What are the data formats supported by Spark?
5. Do you need to install Spark on all nodes of the YARN cluster?
6. Define functions of SparkCore.
7. Name the components of Spark Ecosystem.
8. List the functions of Spark SQL.

Q.II Answer the following questions.

1. How is Apache Spark different from MapReduce?
2. Explain the working of Spark with the help of its architecture.
3. What is SchemaRDD in Spark RDD?
4. Explain the concept of Resilient Distributed Dataset (RDD).
5. How do we create RDDs in Spark?
6. What is Executor Memory in a Spark application?
7. Discuss the architecture of Kafka.

Q.III Define the terms.

1. Spark SQL
2. Spark Streaming
3. Data frame
4. Transformation
5. Action

Bibliography

- https://www.guru99.com
- https://en.wikipedia.org/wiki
- https://www.dataschool.io
- https://www.javatpoint.com
- https://togaware.com/papers
- https://www.sciencedirect.com/topics
- https://spark.apache.org/docs
- https://data-flair.training
- https://www.tutorialspoint.com

■■■