T.Y.B.C.A. SCIENCE : SEMESTER-V

NEW SYLLABUS
CBCS PATTERN

ARTIFICIAL INTELLIGENCE

MEENAL JABDE
CHARUSHILA PATIL

NIRALI
PRAKASHAN
ADVANCEMENT OF KNOWLEDGE

SPPU New Syllabus

A Book Of

Artificial Intelligence

For B.C.A. (Science) : Semester - V

[Course Code BCA - 354 : Credits-02]

CBCS Pattern

As Per New Syllabus, Effective from June 2021

Mrs. Meenal Kaustubh Jabde
M.Sc. Computer Science
Asst. Professor, Modern College of Arts
Science & Commerce,
Ganeshkhind, Pune 16

Mrs. Charushila Patil
NET, SET, Ph.D (Computer Science) Pursuing
Asst. Professor Computer Science Department
Pratibha College of Commerce and Computer Studies
Chinchwad, Pune-19

Price ₹ 110.00

NIRALI PRAKASHAN
ADVANCEMENT OF KNOWLEDGE

N5874

ARITIFICIAL INTELLIGENCE **ISBN 978-93-5451-003-8**

First Edition : August 2021

Published By :
NIRALI PRAKASHAN
Abhyudaya Pragati, 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 an opportunity to present this book entitled as "Artificial Intelligence" to the students of **B.C.A. (Science), Semester - V** as per the New Syllabus, (CBCS Pattern 2019).

The book covers theory of Introduction to Artificial Intelligence, Problems, Problem Spaces and search, Searching Algorithms, Knowledge Representation, Slot and Filler Structures, Recent Trends in AI and Applications.

A special words of thank to Shri. Dineshbhai Furia, Mr. Jignesh Furia for showing full faith in us to write this text book.

We also thank Mr. Akbar Shaikh, Ms. Chaitali Takale, Mr. Ravindra Walodare, Mr. Sachin Shinde, Mr. Ashok Bodke, Mr. Moshin Sayyed and Mr. Nitin Thorat of M/s Nirali Prakashan for their excellent co-operation.

Although every care has been taken to check mistakes and misprints, any errors, omission and suggestions from teachers and students for the improvement of this text book shall be most welcome.

Authors

■■■

Syllabus ...

1. Introduction to Artificial Intelligence **[4 Hrs.]**

1.1 Introduction to AI
1.2 Comparison of AI, Machine Learning, Deep Learning
1.3 Applications of AI
1.4 AI Techniques
1.5 Intelligent Agents, Agents and Environments, Structure of Agents

2. Problems, Problem Spaces and search **[5 Hrs.]**

2.1 Defining Problem as a State Space Search
2.2 Production System
2.3 Problem Characteristics
2.4 Search and Control Strategies
2.5 Problems – Water Jug problem, Missionary Cannibal Problem, Block Words Problem, Monkey and Banana problem

3. Searching Algorithms **[6 Hrs.]**

3.1 Uninformed Search Algorithms/Blind Search Techniques
  3.1.1 Breadth-First Search
  3.1.2 Depth-First Search
3.2 Informed (Heuristic) Search Techniques
  3.2.1 Generate-and-Test
  3.2.2 Simple Hill Climbing
  3.2.3 Best First Search
  3.2.4 Constraint Satisfaction
  3.2.5 Means End Analysis
  3.2.6 A* and AO*

4. Knowledge Representation **[7 Hrs.]**

4.1 Definition of Knowledge
4.2 Types of knowledge (Procedural and Declarative Knowledge)
4.3 Approaches to Knowledge Representation
4.4 Knowledge Representation using Propositional and Predicate Logic
4.5 Conversion to Clause Form
4.6 Resolution in Propositional Logic
4.7 Resolution in Predicate Logic

5. Slot and Filler Structures **[4 Hrs.]**

5.1 Weak Structures (Semantic Networks and Frame)
5.2 Strong Structures (Conceptual Dependencies and Script)

6. Recent Trends in AI and Applications **[4 Hrs.]**

6.1 Introduction to Machine Learning
6.2 Types of Learning, (Supervised, Unsupervised and Reinforcement Learning)
6.3 Predictive Analytics (Weather Forecasting)
6.4 AI-Powered Chatbots (SBI Card Chatbot (ILA))

■■■

Contents ...

■■■

1...

Introduction to Artificial Intelligence

Learning Objectives ...

- To know about the basics of Artificial Intelligence.
- To introduce the concept of Machine Learning and Deep Learning.
- To understand Application of AI.
- To learn different AI Techniques.

1.1 INTRODUCTION TO ARTIFICIAL INTELLIGENCE

- Artificial intelligence consists of two words, artificial which means "man-made" and intelligence means "power of thinking", i.e. a man-made thinking power.
- 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. John McCarthy is father of AI.
- According to the father of AI, it is a science and engineering of making intelligent machines and computer programs.
- Today, Artificial Intelligence (AI) is not a new field. Nowadays, we can say that AI is all around us. 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 are popular as "imitating intelligent human behaviour," which is self-explanatory.

1.1.1 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. These terms are defined as follows:
  - **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.

1.1.2 Need of AI?

- It has become important to understand more about Artificial Intelligence and why do we need AI in our lives.
- Artificial intelligence (AI) has already started playing a major role in our lives.
- Understanding the role of AI in our lives can throw light on its need in society, businesses, and regular day-to-day life. Human efficiency, activity, and capabilities are highly improvised and augmented when coupled with intelligent machines.
- The AI-driven Industrial revolution will bring an impact that no other revolution has brought to date.

1.1.3 Advantages of AI

1. **Reduction in Human Error:** Humans make mistakes from time to time. Computers, however, do not make these mistakes if they are programmed properly. With Artificial intelligence, the decisions are taken from the previously gathered information applying certain set of algorithms.
2. **Available 24×7:** An Average human will work for 6-8 hours a day excluding the breaks. Humans are built in such a way to get some time out for refreshing themselves and get ready for a new day of work and they even have weekly offs to stay intact with their work life and personal life. But using AI we can make machines work 24×7 without any breaks and they won't even get bored unlike humans.

3. **Digital Assistance:** Some of the 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.

1.1.4 Disadvantages of AI

1. **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.
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. **Making Humans Lazy:** AI is making humans lazy with its applications automating most of the work. Humans tend to get addicted to these inventions which can cause a problem to future generations.
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.

1.2 COMPARISON OF AI, MACHINE LEARNING, DEEP LEARNING

Machine Learning:

- A subset of AI that includes statistical techniques that enables machines to improve at tasks with experience.
- Machine learning systems can quickly apply knowledge from large datasets to excel at speech recognition, face recognition, translation and many other tasks.

Deep Learning:

- Deep learning is an aspect of machine learning. Furthermore, it is a form of machine learning that applies neural networks.
- Deep learning algorithms solve the same problem using deep neural networks, a type of software architecture inspired by the human brain.

- AI Systems frequently integrate artificial intelligence, machine learning, and deep learning to create a sophisticated intelligence machine that will perform given human functions well.

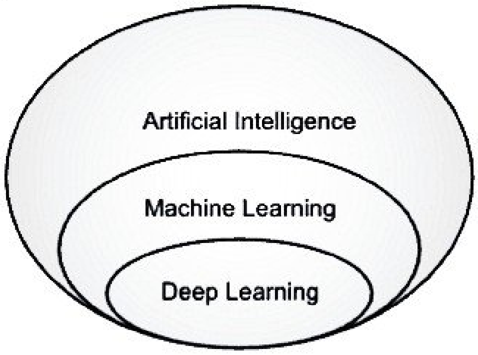

Fig. 1.1: AI Vs. Machine Learning Vs. Deep Learning

Table 1.1: Comparison of AI, Machine Learning, Deep Learning

Artificial Intelligence	Machine Learning	Deep Learning
Artificial Intelligence came around the 1950s.	Machine Learning was invented around the 1960s.	Deep Learning came around the 1970s.
Artificial intelligence (AI) refers to the simulation of human intelligence in machines that are programmed to think like humans and mimic their actions.	Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed.	Deep Learning is artificial neural networks to solve complex problems.
AI is a subset of Data Science.	Machine Learning is a subset of AI and Data science.	Deep Learning is a subset of AI, Deep Learning and Data Science.
AI focuses on building machines that can think like humans.	Machine Learning aims to learn through data to solve the problem.	Deep Learning aims to build neural networks that automatically discover patterns for feature detection.
AI completely deals with structured, semi-structured data.	Machine Learning deals with structured and semi-structured data.	Deep learning deals with structured and unstructured data.

1.3 APPLICATIONS OF AI

- The fields which are closely related with AI include Gaming, Robotics, Cognitive Science, Speech Recognition, Natural Language Processing (NLP), Expert System and Machine Learning.

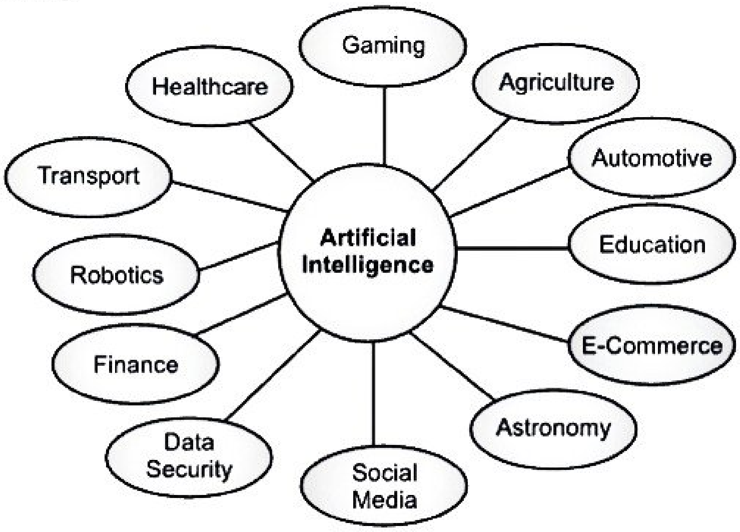

Fig. 1.2: Applications of AI

- Some of the fields are:
    - o **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.
    - o **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.
    - o **Cognitive Science**: It is the interdisciplinary study of human behaviour and intelligence, with a focus on how information is perceived, processed, and transformed.
    - o **Speech Recognition**: Speech recognition is one such 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 could recognize an individual based on their voice command.
    - o **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.
    - o **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 similar to 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.

- **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 adapt their models to improve their ability to make predictions.
- **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 that are now part of our daily lives. The former incorporates face detection meaning any human face is identified. The latter uses face recognition through which a specific face is recognised.
- **E-Payments:** Having to run to the bank for every transaction can be a hectic errand. Good news! Banks are now leveraging artificial intelligence to facilitate customers by simplifying payment processes. Artificial intelligence has made it possible to deposit cheque from the comfort of your home. AI is proficient in deciphering handwriting, making online cheque processing practicable.
- **Computer Vision:** Face recognition programs in use by banks, government, etc. Handwriting recognition, electronics and manufacturing inspection, photo interpretation, baggage inspection, reverse engineering to automatically construct a 3D geometric model.

1.4 AI TECHNIQUES

- AI problems span a very broad spectrum. They are hard but there are some techniques to solve these 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 the 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.
- 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.

1.5 INTELLIGENT AGENTS, AGENTS AND ENVIRONMENTS, STRUCTURE OF AGENTS

1.5.1 Intelligent Agents

- "An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors".
- An agent in Artificial Intelligence is one that takes decisions based on rewards and punishments.
- An intelligent agent is a program that can make decisions or perform a service based on its environment, user input and experiences. These programs can be used to autonomously gather information on a regular, programmed schedule or when prompted by the user in real time.
- A human agent has sensory organs to get information (percepts) from the world (environment) and has muscles (effectors) to take actions in response to the percepts. Robots, the physical instantiation of agents, have got sensors like infrared range finders and cameras to gather information. In this case the effectors are the motors.

1.5.2 Agents and Environments

- AI can be thought of as intelligent "agents" that interact with environment.
- Agent and Environment are two pillars in Artificial Intelligence. We can say that broadly agents are the solution and environment is the problem.
- In general, intelligent agents of all types (including rats, people, as well as AI programs) interact with their environments through: Perception and Action.
  1. **Perception** is the process of transforming something from the environment into internal representations (memories, beliefs, etc.).

2. **Action** done when the agent, by doing something, changes the environment.

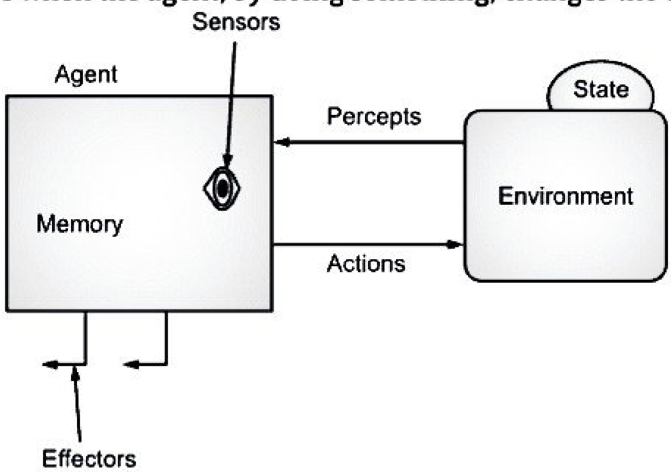

Fig. 1.3: AI Agent and Environment

- Above Fig 1.3 shows, the interaction of the Agent with the Environment is through Sensors and Effectors.
- Sensors are what agents use to get things from their environment to perform their perception. The trading agent may use this information to make trades based on the information they perceive from their sensors to influence the market to their advantage.
- Consider the example of a chatbot which is a virtual assistant. When it reads and understands the meaning of a user's messages, it is called perception. And when it replies to the user after analyzing the user's message, it is called the action.
- An agent can be:
  - o **Human-Agent:** A human agent has eyes, ears, and other organs which work for sensors and hand, legs, and vocal tract work for actuators.
  - o **Robotic Agent:** A robotic agent can have cameras, infrared range finder, NLP for sensors and various motors for actuators.
  - o **Software Agent:** Software agent can have keystrokes, file contents as sensory input and act on those inputs and display output on the screen.
- Some examples of Agent and Environment are as shown in Table 1.2:

Table 1.2: Examples of Agent and Environment

Agent	Environment
Robot	Room
Vehicle	Road
Machine	Working Field

- Hence, the world around us is full of agents such as thermostat, cell phone, camera, and even we are also agents.

- Few terms that is related to Agent and Environment are:
  - **Perception**: What agent sees in the environment?
  - **Perception history**: It is the history of perception which comes in a specific period.
  - **Actuators**: A mechanism that puts something into action.
  - **Effectors**: Agent's organs (hands and legs) that becomes active.

1.5.3 Structure of Agents

- Agent Program is a function that implements the agent mapping from percepts to actions. There exists a variety of basic agent program designs, reflecting the kind of information made explicit and used in the decision process. The designs vary in efficiency, compactness, and flexibility. The appropriate design of the agent program depends on the nature of the environment.
- An intelligent agent is a combination of Agent Program and Architecture.
- The architecture makes the precepts from the sensors available to the agent program. Runs the program and feeds the program's action choices to the effectors as they are generated.

```
Agent = Architecture + Program
```

- Before we design an agent program, we must have a pretty good idea of the possible percepts and actions, what goals or performance measure the agent is supposed to achieve, and what sort of environment it will operate in.

Table 1.3: Examples of Different Agent

Agent Type	Percepts	Actions	Goals	Environment
Medical diagnosis system	Symptoms, findings, patient's answers	Questions, tests, treatments	Healthy patient, minimize costs	Patient, hospital
Satellite image analysis system	Pixels of varying intensity, color	Print a categorization of scene	Correct categorization	Images from orbiting satellite
Part-picking robot	Pixels of varying intensity	Pick-up parts and sort into bins	Place parts in correct bins	Conveyor belt with parts
Refinery controller	Temperature, pressure readings	Open, close valves; adjust temperature	Maximize purity, yield, safety	Refinery
Interactive English tutor	Typed words	Print exercises, suggestions, corrections	Maximize student's score on test	Set of students

Summary

- According to the father of Artificial Intelligence, John McCarthy, it is "The science and engineering of making intelligent machines, especially intelligent computer programs".
- An AI system is composed of an agent and its environment. The agents act in their environment. The environment may contain other agents.
- AI programs are developed to perform specific tasks, that are being utilized for a wide range of activities including medical diagnosis, electronic trading platforms, robot control and remote sensing.
- AI has been used to develop and advance numerous fields and industries, including finance, healthcare, education, transportation, and more.
- Artificial intelligence (AI), machine learning and deep learning are three terms often used interchangeably to describe software that behaves intelligently.
- Deep learning is a subset of machine learning, and machine learning is a subset of AI, which is an umbrella term for any computer program that does something smart.
- Through different AI techniques we can solve hard problems easily.

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. Who is the Father of AI"?
   (a) Fisher Ada (b) Alan Turing
   (c) John McCarthy (d) Allen Newell
3. The application/applications of Artificial Intelligence is/are ______.
   (a) Expert Systems (b) Gaming
   (c) Vision Systems (d) All of the above
4. If a robot is able to change its own trajectory as per the external conditions, then the robot is considered as the ______.
   (a) Mobile (b) Non-Servo
   (c) Open Loop (d) Intelligent

5. AI Technique is a manner to organize and use the knowledge efficiently in such a way that ______.
   (a) It should be perceivable by the people who provide it
   (b) It should be easily modifiable to correct errors
   (c) Both (a) and (b)
   (d) None of above
6. Which of the following is not an application of AI?
   (a) Intelligent Robots (b) Handwriting Recognition
   (c) Speech Recognition (d) Content mining
7. What is Machine learning?
   (a) The autonomous acquisition of knowledge through the use of computer programs
   (b) The autonomous acquisition of knowledge through the use of manual programs
   (c) The selective acquisition of knowledge through the use of computer programs
   (d) The selective acquisition of knowledge through the use of manual programs
8. An AI system is composed of?
   (a) Agent (b) Environment
   (c) Both (a) and (b) (d) Sensors
9. Agent's structure can be viewed as________.
   (a) Architecture (b) Agent Program
   (c) Architecture + Agent Program (d) None of the Above
10. Which instruments are used for perceiving and acting upon the environment?
   (a) Sensors and Actuators (b) Sensors
   (c) Perceiver (d) Perceiver and Sensor

Answers

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

Practice Questions

Q.I Answer the following questions in short.

1. What is artificial intelligence?
2. What are intelligent agents?
3. Give some examples of AI in real life.
4. Why do we need Artificial Intelligence?
5. How is machine learning related to AI?
6. A major thrust of AI is in the development of computer functions associated with human intelligence. State True or False.

7. A robot is a mechanical device equipped with simulated human senses and the capability of acting on its own. State True or False.
8. Artificial intelligence (AI) is the science of making humans imitate computer thinking and behaviour. State True or False.

Q.II Answer the following questions.

1. How Artificial intelligence, Machine Learning, and Deep Learning differ from each other?
2. Give some real-world applications of AI.
3. What are techniques of AI?
4. What are the less desirable properties of knowledge?
5. Explain the structure of an AI Agent.
6. What are the different types of Agent?
7. What are the advantages of AI?
8. What are disadvantages of AI?
9. Explain need of AI.

Q.III Define the terms.

1. Data
2. Knowledge
3. AI
4. Intelligent AI
5. Machine learning
6. Deep learning
7. Perception
8. Actuators

■■■

2...

Problems, Problem Spaces, and Search

Learning Objectives ...

- To know about how to represent a Problem in State Space Search.
- To know about the basics of Search and Control Strategies.
- To understand the concept of the Production System.
- To know about different Problem Characteristics.

2.1 DEFINING PROBLEM AS A STATE SPACE SEARCH

- **Problem**: The term problem is used in a situation when the desired objective is not achieved. Reaching to an objective from initial situation is unknown initially and it consists of sequence of intermediate states.
- A state space representation allows for the formal definition of a problem which makes the movement from initial state to the goal state easily.
- Problem solving is a process of generating solutions for a given problem. To solve a particular problem, need to build a system. For building a new system, we need to do four things:
  1. Define the problem precisely that includes specifications about initial situations and final situations constitute acceptable solutions to the problem.
  2. Analyze the problem. A few important features which have an impact on the appropriateness of various possible techniques for solving the problem.
  3. Isolate and represent the task knowledge that is necessary to solve the problem.
  4. Choose the best problem-solving techniques and apply it to the problem.
- State space search is a process used in the field of computer science, including artificial intelligence, in which successive configurations or states of an instance are considered, with the goal of finding a solution with a desired property.

- In AI we use machine to solve a problem, once a solution is planned and fed to it. That is done using state space diagram. So, the problem is defined in terms of state. Each state is the abstraction of all available information.
- 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.
- During the problem solving an operator is applied to a state to move it to the next state. The process is going on till the final state is achieved. This approach of generating a solution is called **State Space Method.**
- The basic idea behind state space search is that we can solve the given problem by checking the steps considering the fact that they might lead us towards solution. Each action takes a step ahead to a different state.

2.1.1 Water Jug Problem

- In this problem, you are given two jugs called 4 liter and 3 liter; 4 holds a maximum of four liters of water and 3 a maximum of three liters of water. Neither has any measuring marker on it. There is a pump that can be used to fill the jugs with water. How can we get two liters of water in the 4 liter jug?
- 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).
- Such that,

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

  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 number, but it is limited to three holding from 0 to 3 liters of water or empty.
- We have assumed that we can fill jug with pump, we can pour water out of jug onto the ground. Also, we can pour water from one jug to another.
- We use control structure that loops through simple cycle in which some rules whose left side matches the current state is chosen.

Following Steps shows how we can solve the problem:

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

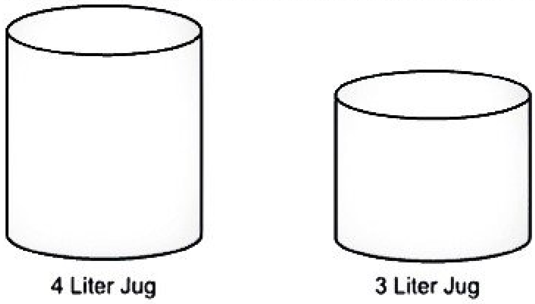

Fig. 2.1

Step 2 : Fill a 3 liter jug with water.

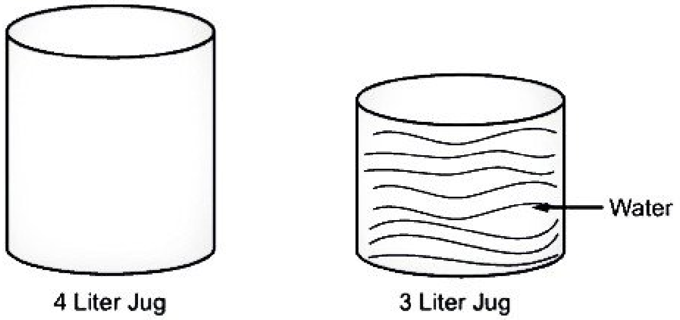

Fig. 2.2

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

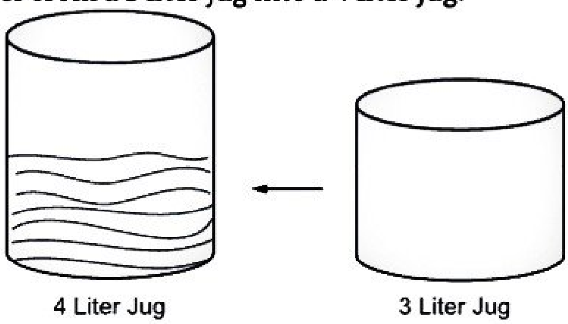

Fig. 2.3

Step 4 : Fill again 3 liter jug with water.

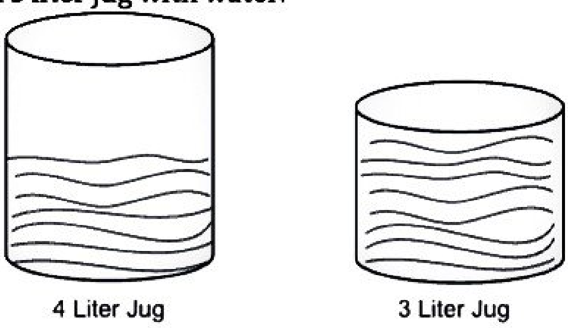

Fig. 2.4

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

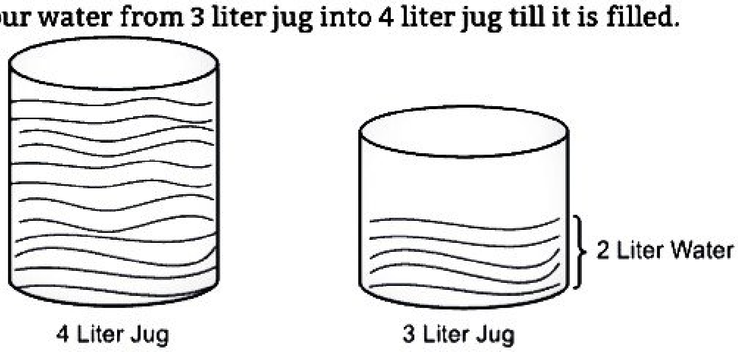

Fig. 2.5

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

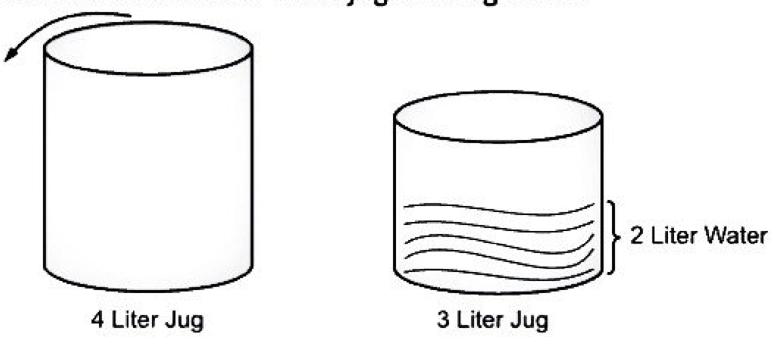

Fig. 2.6

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

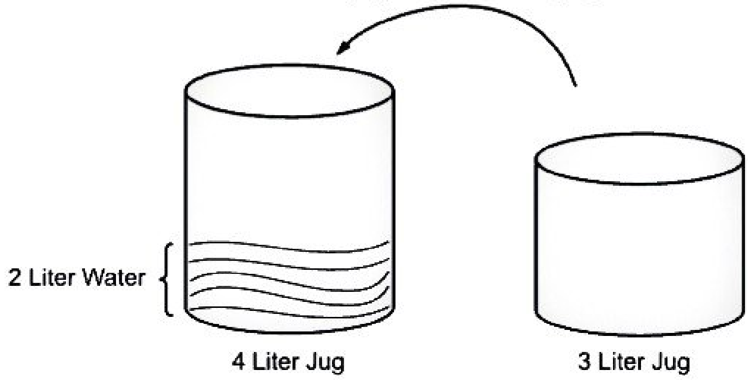

Fig. 2.7: Result of Water Jug Problem

- To solve the problem scientifically, we should convert informal problem statement into formal problem description.
- 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.

contd. ...

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 liter water from 3 liter jug into the 4 liter jug.
12.	(2, y)	(0, y)	Empty the 2 liter water from 4 liter jug on the ground.

- For water jug problem there are several sequences of operators that solve the problem, one such sequence is shown in Table. 2.2. Here operator includes information about what must be true in the world before the action can take place, and how the world is changed by the action.

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 that is used for selecting rules to apply during problem solving, 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 which contains all possible configurations of the relevant objects.
  2. Specify one or more states which is called as *Initial state*, within that space which describe the situation from which problem-solving process may start.

3. Specify one or more states which is called as *Goal State* that would be acceptable as solutions to the problem.
4. Specify a set of rules that describes the action available.

- Thus, the problem can be solved by using rules with proper control strategy.

2.2 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 is 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. 2.8 shows the basic architecture of production systems in AI.

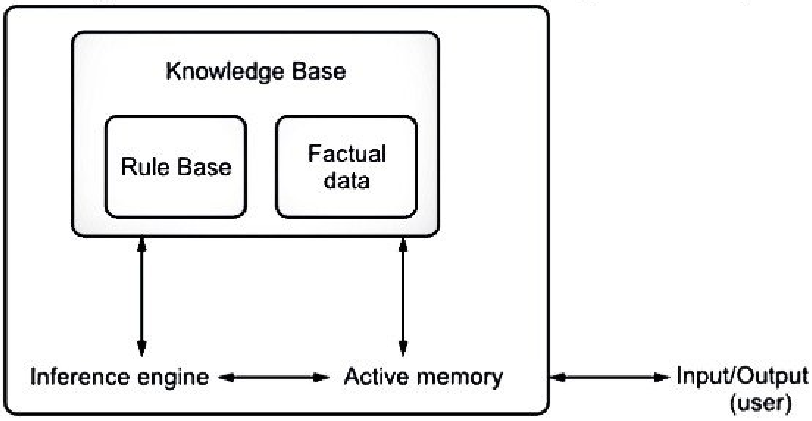

Fig. 2.8: Architecture of Production System in AI

2.3 PROBLEM CHARACTERISTICS

- A problem may have different aspects of representation and explanation. In order to choose the most appropriate method for a particular problem, it is necessary to analyze the problem along several key dimensions.
- Some of the main key features of a problem are given below:
  - Is the problem decomposable into a set of sub problems?
  - Can the solution steps be ignored or undone?
  - Is the problem universally predictable?
  - Is a good solution absolute or relative?
  - Is the desired solution a state of the world or a path to a state?
  - What is the role of Knowledge?
  - Will the solution of the problem require interaction between the computer and the person?
- Let's have a look one by one:

1. **Is the problem decomposable?**
   - A very large and complex problem can be easily solved if it can be divided into smaller problems and recursion could be used. Suppose we want to solve problems.

     Example: $\int x2 + 3x + \sin 2x \cos 2x \, dx$

     This can be done by breaking it into smaller problems and solving each by applying specific rules. And then adding the results to make complete solution.

2. **Can the solution steps be ignored or undone?**
   - The important classes of problems are ignorable, recoverable, and irrecoverable. This classification is with reference to the steps of the solution to a problem.
     - **(i) Ignorable problems:** In which solution steps can be ignored. Ignorable problems can be solved using a simple control structure that never backtracks. Example: Theorem proving.
     - **(ii) Recoverable problems:** In which solution steps can be undone. The recoverability plays an important role in determining the complexity of the control structure necessary for the problem's solution. Example: 8 puzzle.
     - **(iii) Irrecoverable problems:** In which solution steps can't be undone. 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. Example: Chess.

3. **Is the problem universally predictable?**
   - Problems can be categorized into those with definite results (eight puzzle and water jug problems) and those with indefinite results (playing cards).
   - In definite result 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 indefinite result problems, planning can generate a best sequence of operators that has a good possibility of leading to a solution. It won't give the guarantee to solution, and it is 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 problems. Example: Playing cards.

4. Is a good solution absolute or relative?

- There are two categories of problems: simple and complex problems.
- In simple problems, means water jug and 8 puzzle problems, we are satisfied with the solution, we are aware of the solution path taken.
- In complex problem, not just any solution is acceptable. We want the best, like that of traveling salesman problem, we have found not just a solution but the best path. In any path problems, by heuristic methods we obtain a solution and we do not explore alternatives.

5. Is the solution a state or a path?

- In 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 sequence of operations that produces the final state is the solution of water jug problem.
- Now, consider the word "right", which has two interpretations.
  (a) Right - Direction
  (b) Right - Correct
- Here, we should decide the meaning which is suitable by viewing the whole sentence.
- To solve the problem of finding the meaning, no record of processing is necessary. We need to consider only interpretation. 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. What is the role of Knowledge?

- Arriving at a good solution, the size of the knowledge base available matters though one could have unlimited computing power.
- 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 then be realistic.
- Now, consider the case of predicting the political trend. This would require a massive amount of knowledge even to be able to recognize a solution.

- o These two examples illustrate the difference between problems for which a lot of knowledge is important only to make 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.

7. **Will the solution of the problem require interaction between the computer and the person?**
   - o The program requires intermediate interaction with people for additional inputs and to provide reassurance to the user.
   - o There are 2 types of problems:
     1. Solitary, in which there is no intermediate communication and no 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 is to provide either additional assistance to the computer or to provide additional information to the user. Example: Problems such as medical diagnosis.

Problem Classification:

- There are several broad classes into which the problems fall. These classes can each be associated with generic control strategy that is appropriate for solving the problems.
- Following table 2.3 shows analysis of some problems with respect to the seven problem characteristics.

Table 2.3: Problem Classification

Sr. No.	Characteristics → Name of problem ↓	1 Is the problem decom-posable	2 Can Solu-tion step ignored/ undone	3 Is Universe predict-table	4 Is a solution absolute or relative	5 Is a solu-tion state or path	6 What is role of knowledge	7 Does require human inter-action
1.	Chess	No	No	No	Absolute	Path	Need of knowledge and rules and legal moves	No
2.	Water Jug Problem	No	Yes	Yes	Absolute	Path	No need of knowledge	Yes
3.	8 - Puzzle	No	Yes	Yes	Absolute	Path	Knowledge for condition	No
4.	Travelling Salesperson Problem	No	Yes	Yes	Absolute	Path	Knowledge for condition	No
5.	Tower of Hanoi	No	Yes	Yes	Absolute	Path	Knowledge	No

							of recursion	

2.4 SEARCH AND CONTROL STRATEGIES

- Control strategies are also called as Search strategies. These are adopted for applying the rules and searching the problem solution in search space. It is already mentioned earlier that 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.
- The features of good control strategies are as follows:
  (a) A good control strategy is 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.
  (b) 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. Example: BFS, DFS.
- 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. To solve the problem, we have to use a good control strategy which ultimately takes us to the goal state.
- In search method or technique, firstly select one option and leave other option. If this option is our final goal, then stop else we continue selecting, testing, and expanding until either a solution is found or no more states to be expanded. This will be determined by Search method. There are different types of search algorithms as shown in Fig. 2.9.

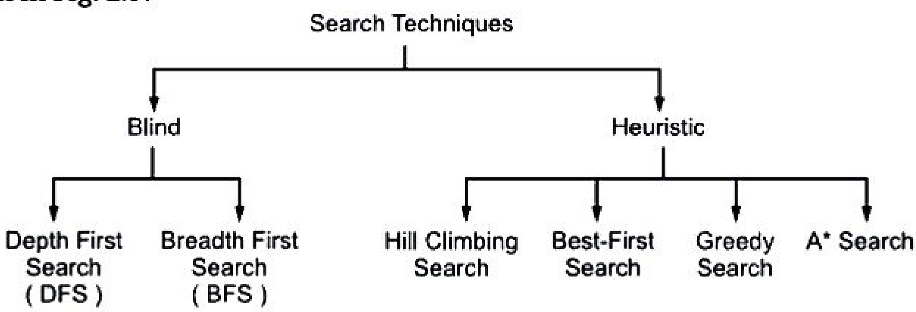

Fig. 2.9: Types of Search Techniques

- Basically there are two types of searches:
  1. **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) and Bidirectional Search.
  2. **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.

2.5 PROBLEMS

2.5.1 Water Jug problem

- Already discussed in point 2.1.1.

2.5.2 Missionary Cannibal Problem

- "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 side. 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 side of river to another using the boat sailing through the river satisfying these constraints.
- 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 side-1 to side-2.

Rule 2 : (M, 0) : One missionary sailing the boat from side-2 to side-1.

Rule 3 : (M, M) : Two missionaries sailing the boat from side-1 to side-2.

Rule 4 : (M, M) : Two missionaries sailing the boat from side-2 to side-1.

Rule 5 : (M, C) : One missionary and one Cannibal sailing the boat from side-1 to side-2.

Rule 6 : (C, M) : One missionary and one Cannibal sailing the boat from side-2 to side-1.

Rule 7 : (C, C) : Two Cannibals sailing the boat from side-1 to side-2.

Rule 8 : (C, C) : Two Cannibals sailing the boat from side-2 to side-1.

Rule 9 : (0, C) : One Cannibal sailing the boat from side-1 to side-2.

Rule 10 : (C, 0) : One Cannibal sailing the boat from side-2 to side-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 2.4.

Table 2.4: Applied Rules and their Sequence in Missionaries and Cannibals Problem

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

2.5.3 Block Words Problem

- "There are some cubic blocks, out of which, each is placed either on the table or on other block forming a particular configuration. We have to move the blocks to arrange them to form some other given configuration by applying a minimum number of moves. The requirement of moving a block is that only the top block from a group can be shifted in a move. It can be placed either on the table or on top of any other block."
- Initial state and Goal state are shown in following Fig. 2.10.

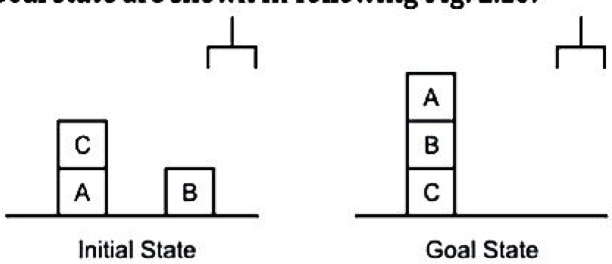

Fig. 2.10: Start State and Goal State

Sequence of Actions:

1. Grab C
2. Pick up C
3. Place on table C
4. Grab B
5. Pickup B
6. Stack B on C
7. Grab A
8. Pick up A
9. Stack A on B

Start State- ON [C, A]

Goal State- ON [B, C] and ON [A, B]

- Following operators are available:
    - CLEAR(x)(block x has nothing on it) → ON [x, Table]
    - CLEAR (x) and CLEAR (y) → ON [x,y] (put x on y)
- Execution plan for acquiring goal state is as shown in following Fig. 2.11.

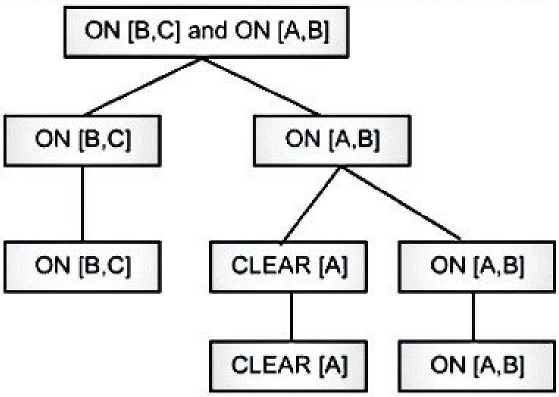

Fig. 2.11: Execution Tree Structure

2.5.4 Monkey and Banana Problem

- "A monkey and a bunch of bananas are present in a room. The bananas are hanging from the ceiling. The monkey cannot reach the bananas directly. However, in the room there is one chair and a stick. The monkey can reach the banana standing on the chair. We have to find the sequence of events by which monkeys can reach the bananas."

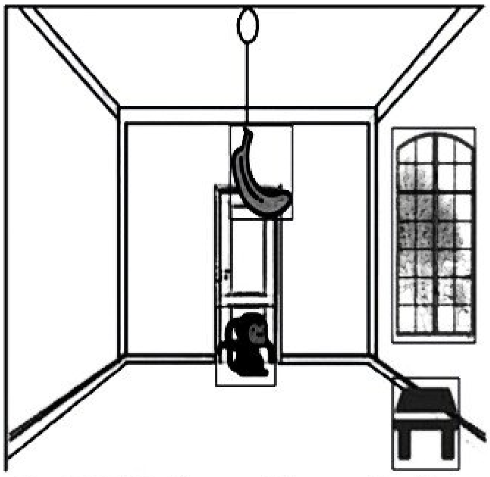

Fig. 2.12: Monkey and Banana Problem

- **Initial State:** Monkey on ground with empty hand.
- **Final/Goal State:** Monkey eating Bananas.
- **Actions**:
  - Climb chair or get off
  - Grab banana
  - Wave Stick
  - Eat Banana

- If the monkey is clever enough and if he performs the action as per following tree structure shown in Fig. 2.13, he can get his lunch.

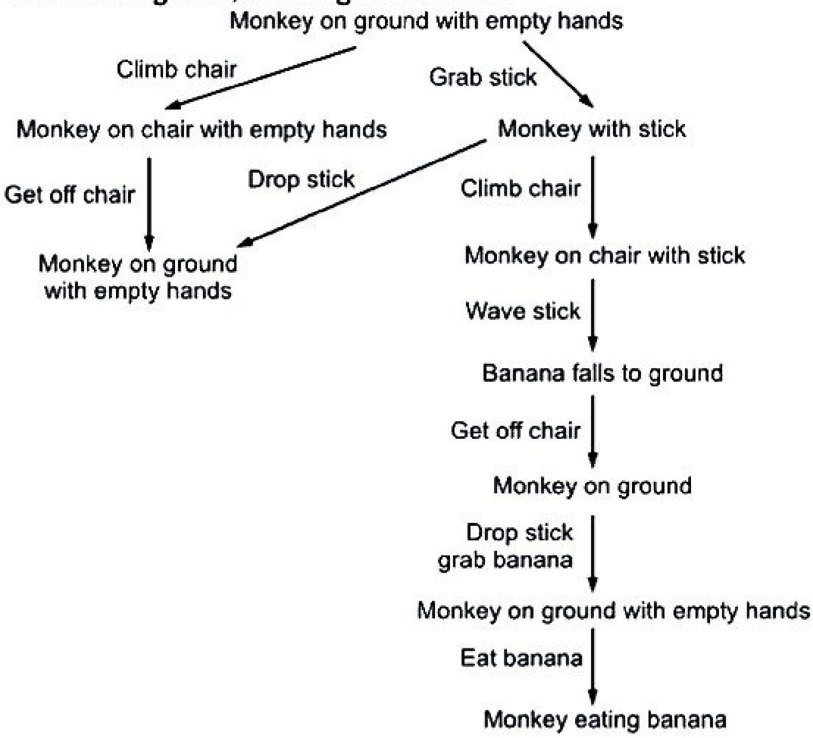

Fig. 2.13: Tree Structure for Monkey and Banana Problem

Summary

- The term problem is used in a situation when the desired objective is not achieved.
- A state space representation allows for the formal definition of a problem which makes the movement from initial state to the goal state easily.
- State space search is a process used in the field of computer science, including artificial intelligence, in which successive configurations or states of an instance are considered, with the goal of finding a goal state with a desired property.
- The problem is solved by using the production rules in combination with an appropriate control strategy that is used for selecting rules to apply during problem solving, moving through the problem space until a path from an initial state to a goal state is found.
- Production System was proposed by Emil Post in 1943 which is also known as Inferential Systems or Rule-based system.
- Production systems provide appropriate structures for performing and describing search processes.
- 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.
- Some of the main characteristics of a problem are: Is the problem decomposable into a set of sub problems?, Can the solution steps be ignored or undone?, Is the problem universally predictable?, Is the desired solution a state of the world or a path to a state?, What is the role of Knowledge?, Will the solution of the problem require interaction between the computer and the person?
- Control strategies which are also called as Search strategies are adopted for applying the rules and searching the problem solution in search space.
- The features of good control strategies are: A good control strategy is causes motion, A good control strategy should be systematic.
- Basically there are two types of searches: 1. Uninformed or Blind Search or Brute Force, 2. Heuristic / Informed Search Strategy.
- A heuristic comprises a technique that improves the efficiency of a search process by sacrificing claims of completeness.

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 arches between nodes are 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. Which search method takes less memory?
   (a) Depth-First Search (b) Breadth-First search
   (c) Linear Search (d) Optimal search
9. Which data structure is used to implement BFS?
   (a) Stack (b) Queue
   (c) Linked List (d) Priority Queue
10. 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

Answers

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

Practice Questions

Q.I Answer the following questions in short.

1. Define Search strategy.
2. State the things required to be considered when we want to build an AI system that is used to solve a particular problem.
3. What is Heuristic search?
4. A good control strategy is that it causes motion and should be systematic. State True or False.
5. Heuristic Search is used to solve simple problems. State True or False.

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 "Monkey Banana problem".
7. Give state space representation for "Block world Problem".

Q.III Define the terms.

1. Heuristic Search
2. Production System
3. Ignorable problem
4. Recoverable Problem
5. Irrecoverable Problem

■■■

3... Searching Algorithms

Learning Objectives ...

- To impart knowledge about different Uninformed Search Algorithms.
- To give understanding about the Informed strategies.
- To understand wide variety of Algorithms.
- To apply Algorithm to real-world problem.

3.1 UNINFORMED SEARCH ALGORITHMS/ BLIND SEARCH TECHNIQUES

- 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 term of states, and solutions correspond to a goal state. Different search techniques are available that we will describe here.
- 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

3.1.1 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.

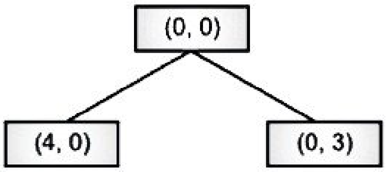

Fig. 3.1: One level BFS

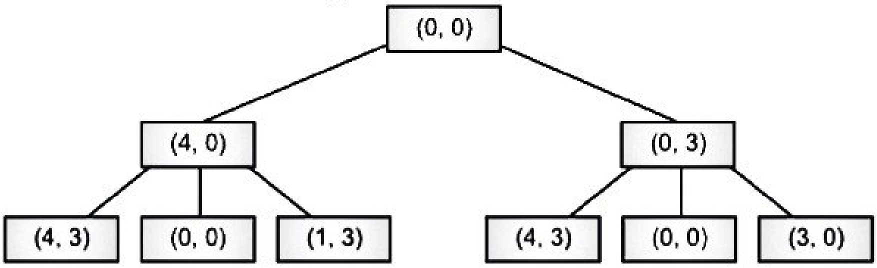

Fig. 3.2: Two levels of BFS

Algorithm: Breadth-First Search

Step 1: Create a variable called NODE -LIST and set it to the initial state.

Step 2: Until goal state is found, or NODE-LIST is empty, do:

(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 alley 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.

Disadvantages of Breadth First Search:

1. High storage requirement: exponential with tree depth

3.1.2 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 yields 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. 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.

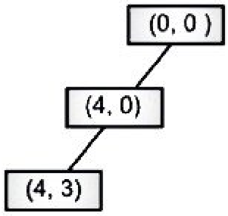

Fig. 3.3: Depth First Search Tree

Algorithm: Depth-First Search

Step 1: If the initial state is a goal state, quit and return success.

Step 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.

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, may not find a solution even if one exists.

3.2 INFORMED (HEURISTIC) SEARCH TECHNIQUES

- For complex problems, the traditional algorithms, presented above, are unable to find the solution within some practical time and space limits. Consequently, many special techniques are developed, using heuristic functions.
- 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 and 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.

Characteristics of Heuristic Search:

- Heuristics are knowledge about domain, which help search and reasoning in its domain.

- Heuristic search incorporates domain knowledge to improve efficiency over blind search.
- Heuristic is a function that, when applied to a state, returns value as estimated merit of state, with respect to goal. Heuristics might (for reasons) underestimate or overestimate the merit of a state with respect to goal. Heuristics that underestimate are desirable and called admissible.
- Heuristic evaluation function estimates likelihood of given state leading to goal state.
- Heuristic search function estimates cost from current state to goal, presuming function is efficient.

3.2.1 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:

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. 3.4.

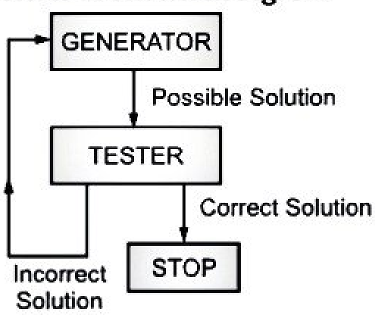

Fig. 3.4: Generate-and-Test Representation

- 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 as **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.

3.2.2 Simple Hill Climbing

- It is called Hill Climbing 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. 3.5.

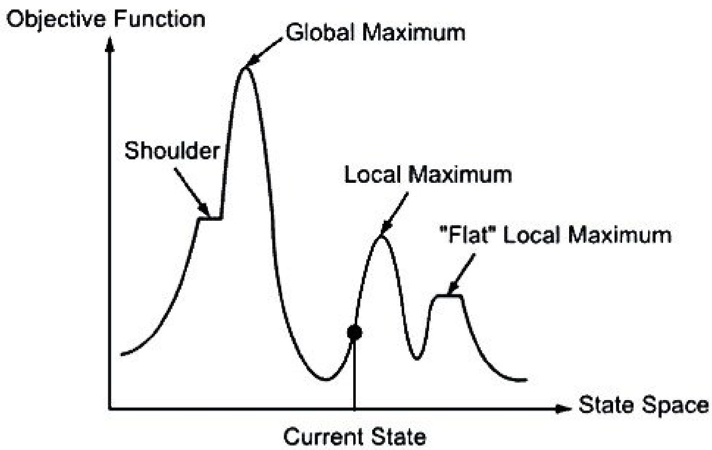

Fig. 3.5: State Space landscape for Hill Climbing

- 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 that is higher than its neighbours 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. 3.5 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.

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 o 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:

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

  (d) Evaluate the new state.

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

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

  (g) 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 neighbours 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. 3.6.

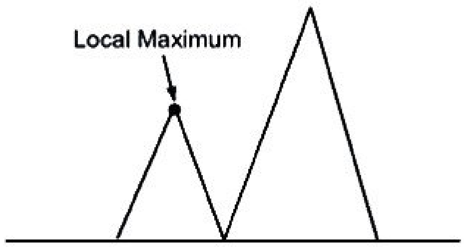

Fig. 3.6: Local Maximum

Solution to the above problem:

- One possible solution is backtracking. We can back rack 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 neighbouring 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. 3.7.

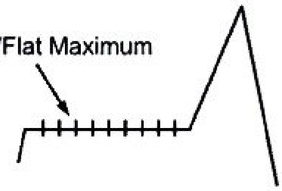

Fig. 3.7: Plateau

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 neighbouring 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. 3.8.

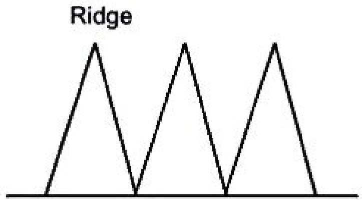

Fig. 3.8: Ridge

- 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.

3.2.3 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 an 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. 3.9 where numbers by the nodes may be regarded as estimates of the distance or cost to reach the goal node.
- **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.

Best First Search - Example

- Heuristics here is the estimate of cost of getting to the solution (goal) from a given node.

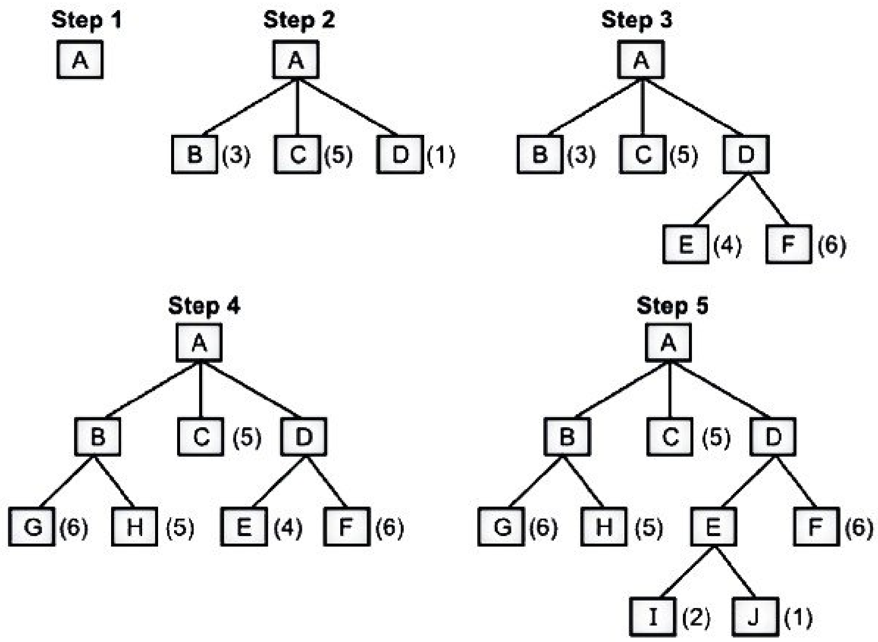

Fig. 3.9: Best First Search Example

Advantage of Best First Search:

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

Disadvantages of Best First Search:

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.

3.2.4 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. Eample: 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:

Step 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.

Step 2: If the union of constraints discovered above defines a solution then quit and return the solution.

Step 3: If the union of the constraints discovered above defines a contradiction, then return failure.

Step 4: If neither of the above occurs, then it is necessary to make a guess at something in order to proceed. To do this, loop until a solution is found or all possible solutions have been eliminated:

(a) Select an object whose value is not yet determined and select a way of strengthening the constraints on that object.

(b) Recursively invokes constraint satisfaction with the current set of constraints augmented by the strengthening constraint just selected.

- To see how this algorithm works, consider the cryptarithmetic problem as shown in Fig. 3.10.

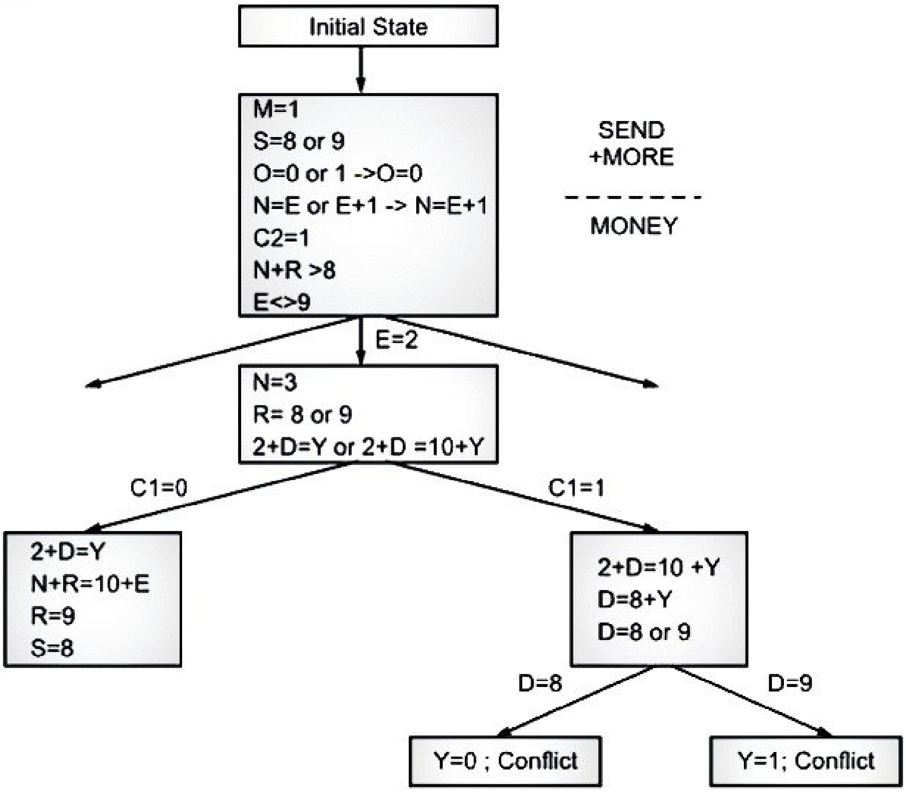

Fig. 3.10 Solving Cryptarithmetic Problem

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

3.2.5 Means End Analysis

- 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. Then it decides the best action to be undertaken to reach the end goal.
- The following Fig. 3.11 shows how the target goal is divided into sub-goals that are then linked with executable actions.

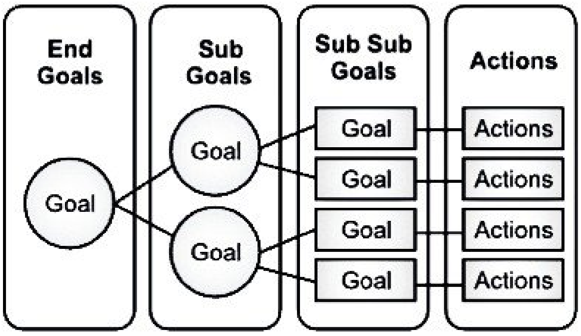

Fig. 3.11: Goals Dividation

Algorithm for Means-Ends 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.

- 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:

- The following Fig. 3.12 shows the Initial state and the Goal state.

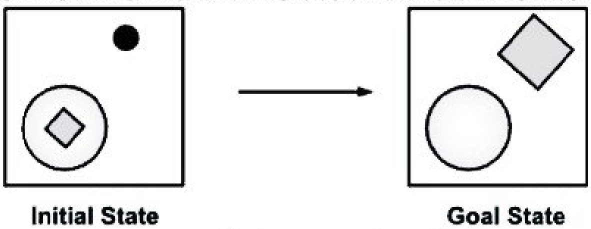

Fig. 3.12: Initial state and Goal State

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. 3.13.

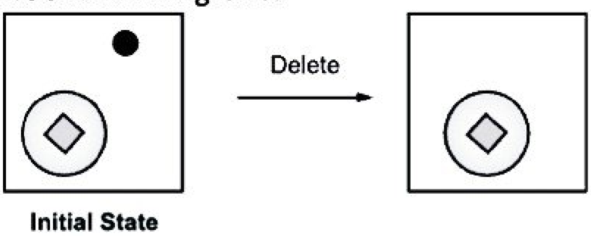

Fig. 3.13: Delete Operator

2. **Move operator**: We will then compare the new state with the goal state. The black diamond in the new state is inside the circle while the black 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. 3.14.

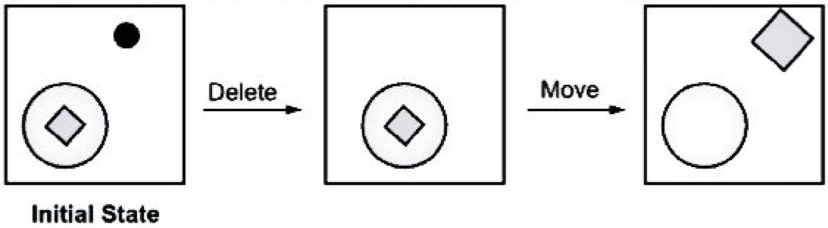

Fig. 3.14: Move Operator

3. **Expand operator**: The new state generated in second step, 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. 3.15.

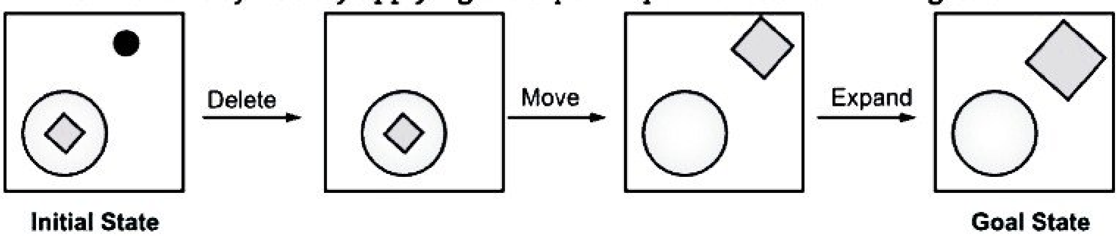

Fig. 3.15: Expand Operator

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

3.2.6 A* and AO*

3.2.6.1 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.
    1. OPEN contains those nodes that have been evaluated by the heuristic function but have not been expanded into successors yet.
    2. 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 go to Step 4.

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

Fig. 3.16: 8 - Puzzle Problem

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

  Consider,

  g(n) = Depth of node

  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. 3.17.

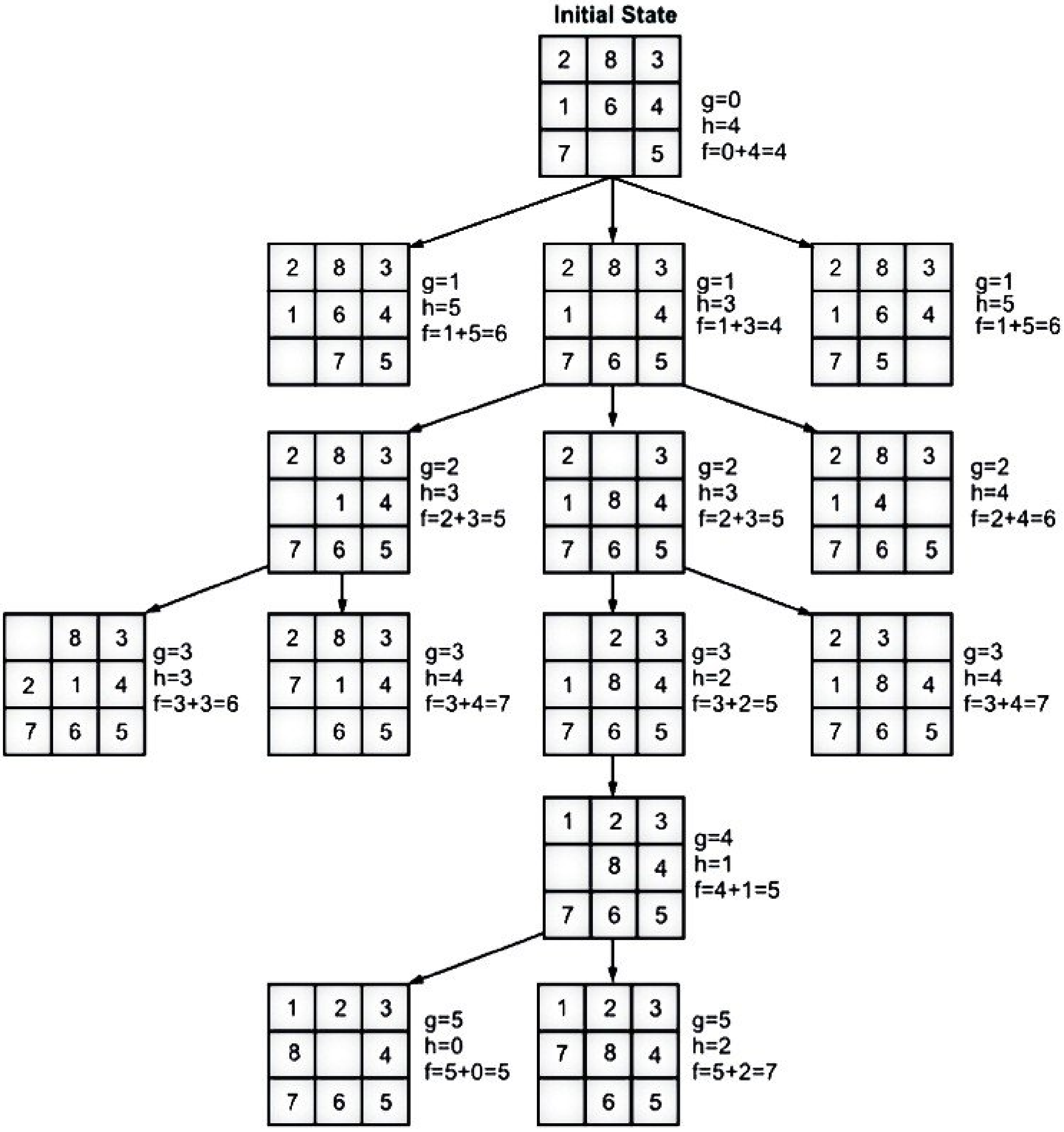

Fig. 3.17: A* Algorithm Example

Advantages of A*:

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 of A*:

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).

3.2.6.2 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 are 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 3.18 shows an AND - OR graph.

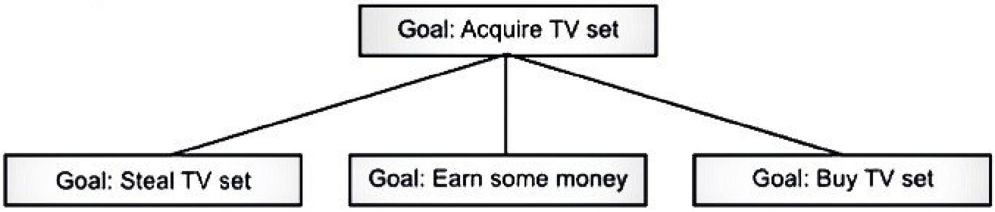

Fig. 3.18: A simple AND-OR graph

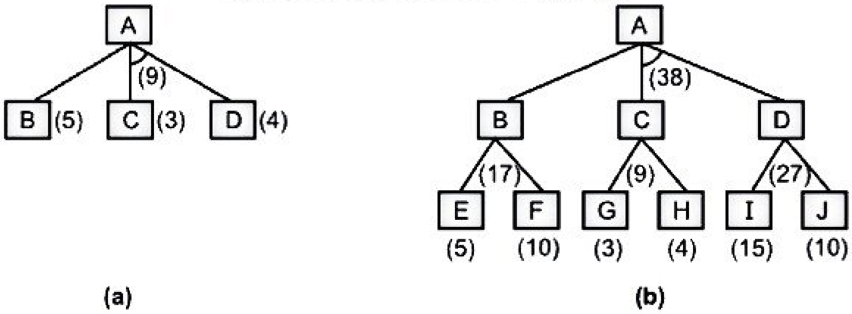

Fig. 3.19: AND-OR graph

- In Fig. 3.19 (a), the top node A has been expanded producing two paths 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. 3.19 (b) makes this clearer. In Fig. 3.19 (b) 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 G-H 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).

- **The following three things keep in mind while traversing the graph:**
  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 the following Fig. 3.20.

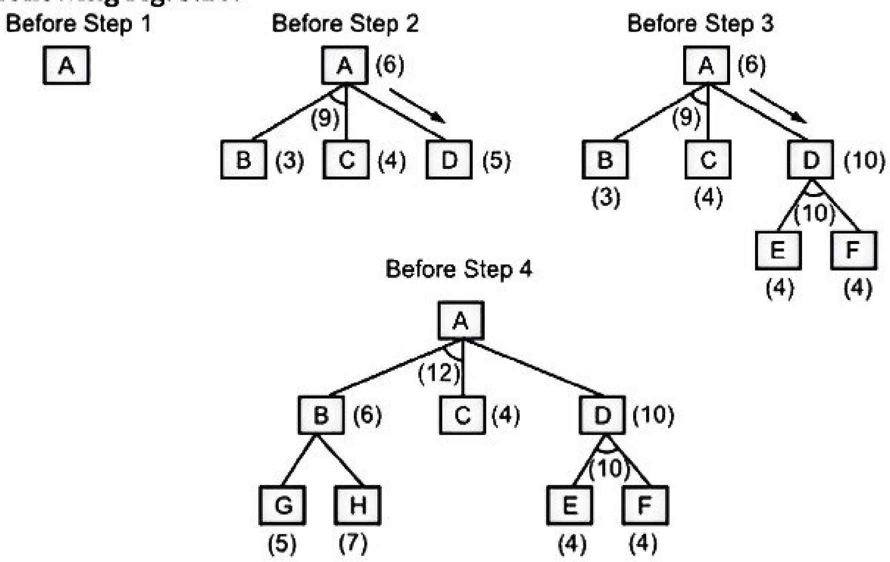

Fig. 3.20: The operation of problem reduction

- 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. In 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.

1. **OPEN:** It contains the nodes that has been traversed but yet not been marked solvable or unsolvable.
2. **CLOSE:** It contains the nodes that have already been processed.

AO* Algorithm:

Step 1: Let G consists only of the node representing the initial state called this node INTT. Compute h' (INIT).

Step 2: Until INIT is labeled SOLVED or hi (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:

Lets 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 a s 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

- 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.
- Search is a characteristic of almost all AI problems. Search strategies can be compared by their time and space complexities. It is important to determine the complexity of a given strategy before investing too much programming effort, since many search problems are traceable.
- 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.
- Depth first searches are performed by going downward into a tree as early as 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, or failure occurs. Examples of brute force search are breadth first search and depth first search.
- In case of Heuristic Search (Informed Search) cost or another function is used to 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.
- For complex problems, the traditional algorithms, presented above, are unable to find the solution within some practical time and space limits. Consequently, many special techniques are developed, using heuristic functions.

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 integer value
   (c) A function whose return type is nothing
   (d) A function that maps from problem state descriptions to measures of 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 & 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. (d)	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.

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

■■■

4... Knowledge Representation

Learning Objectives ...

- To know about the basics of Knowledge Representation.
- To express the knowledge about the world in a Computer-Tractable form.
- To know the basics of Propositional and Predicate Logic.
- To introduce the concept of Resolution.

4.1 INTRODUCTION

- Knowledge representation is the key through which we can make the future AI system much smarter than compared to what they are today.
- Humans excel in comprehending, reasoning, and interpreting information. Humans have knowledge about things and use that knowledge to accomplish various activities in the real world. However, knowledge representation and reasoning deal with how robots achieve all of these things. As a result, the following is a description of knowledge representation:
  - Knowledge representation and reasoning (KR, KRR) are aspects of artificial intelligence that deal with how AI agents think and how their thinking influences their behavior.
  - It's in charge of encoding information about the real world in a way that a computer can comprehend and use it to solve difficult real-world problems like diagnosing a medical condition or conversing with humans in natural language.
  - It's also a technique of describing how artificial intelligence can represent knowledge. Knowledge representation is more than just storing data in a database; it also allows an intelligent machine to learn from its knowledge and experiences in order to act intelligently, like a person.

4.1.1 Definition of Knowledge

- The information about a domain that can be used to solve problems in that area is known as **Knowledge**. The act, fact or state of knowing is a set of facts and principles gathered by humankind.

- Knowledge is represented in AI systems in the following way.
  - **Objects:** Objects are nothing more than the truthful truths. Such facts can be common knowledge or universal truths, such as "The Sun Rises in the East", "Cars have wheels" or any other fact that remains true in any situation.
  - **Events:** Things that happen in the real world are called events. Events are defined as anything that occurs in real time. It is a crucial component because it is the first thing to consider in knowledge representation. The wars, famines, achievements, advancement of societies, etc., are an example of this knowledge.
  - **Performance:** This refers to how well the knowledge has been acquired and how well it can be applied to machines.
  - **Meta-Knowledge:** This refers to information that has previously been obtained, whether by a human brain or a machine.
  - **Knowledge base:** This is the primary component of the agent's knowledge acquisition.
  - **Facts:** It is the knowledge of the factual description of the world.

4.2 TYPES OF KNOWLEDGE

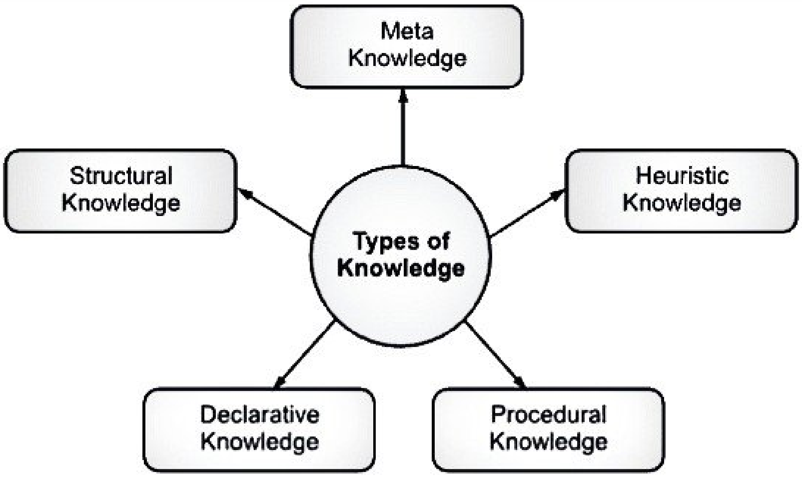

Fig. 4.1: Types of Knowledge

- Primarily, there are five types of knowledge in AI systems. The knowledge types are as follows:
  1. **Declarative Knowledge:** Declarative knowledge is the ability to understand something. It includes concepts of facts, instances, objects declared as a statement. It is also called descriptive knowledge and expressed in declarative sentences. It is simpler than procedural language. Example: Student database.
  2. **Structural Knowledge:** It describes the relationship between instances and description.
  3. **Procedural Knowledge:** It deals with the plans, procedures, strategies and rules required for a particular system to work efficiently. It is also known as imperative

knowledge. Procedural knowledge is a type of knowledge which is responsible for knowing how to do something. For example, the steps applied to solve any real world problem are known as procedural knowledge. It can be directly applied to any task. Procedural knowledge depends on the task on which it can be applied.

4. **Meta Knowledge:** It is the knowledge consisting of the higher-level data of other types of knowledge data.
5. **Heuristic Knowledge:** The knowledge provided by experts of certain domains, subjects, disciplines, and fields is known as Heuristic knowledge, which they have been obtained after years of experience. This helps in taking the best approach to particular problems and making decisions.

4.3 APPROACHES TO KNOWLEDGE REPRESENTATION

- A good system for representing knowledge in a certain area should have the four properties listed below:
  - **Representational Accuracy:** A knowledge representation system should be able to represent any type of knowledge that is necessary.
  - **Inferential Adequacy:** The Knowledge Representation System should be able to manipulate representational structures in order to generate new knowledge that corresponds to the existing structure.
  - **Inferential Efficiency:** The ability to store relevant guides to direct the inferential knowledge mechanism in the most productive paths.
  - **Acquisitional Efficiency:** The ability to learn new information quickly and easily via automated approaches.

4.4 KNOWLEDGE REPRESENTATION USING PROPOSITIONAL AND PREDICATE LOGIC

Methods of Knowledge Representation:

- There are mainly four methods of knowledge representation which are given as follows:
  1. Logical Representation
  2. Semantic Network Representation
  3. Frame Representation
  4. Production Rules

1. **Logical Representation:**
   - Logical representation is a language with some concrete rules which deals with propositions and has no ambiguity in representation. It consists of precisely defined syntax and semantics which supports the sound inference.
   - Syntaxes are the rules which decide how we can construct legal sentences in the logic.
   - Semantics are the rules by which we can interpret the sentence in the logic.

- Logical representation can be categorised into mainly two logics:
  (a) Propositional Logics
  (b) Predicate Logics

2. **Semantic Networks:**
   - This method allows you to represent knowledge in the form of a graphical network. This network consists of nodes representing objects and arcs representing their relationships.
   - This representation consists of mainly two types of relations:
     (a) IS-A relation (Inheritance)
     (b) Kind-of-relation
   - This form of representation is more natural than logical. It is simple to understand however suffers from being computationally expensive and do not have the equivalent of quantifiers found in the logical representation.

3. **Frames:**
   - These methods are the AI data structure which divides knowledge into substructures by representing stereotypes situations. It consists of a collection of slots and slot values. These slots may be of any type and sizes. Slots have names and values which are called **Facets.**

4. **Production Rules:**
   - Production rules system consist of **(condition, action)** pairs which mean, "If condition then action". It consists of the set of production rules, working memory and the recognize-act-cycle.

4.4.1 Propositional Logic

- The simplest kind of logic is Propositional Logic (PL), in which all statements are made up of propositions. The term "proposition" refers to a declarative statement that can be true or false. It is a method of expressing knowledge in logical and mathematical terms.
- Propositional logic, also known as sentential logic and statement logic.

  Example:

  (a) It is rainy season.
  (b) The Sun sets at East. (False proposition)
  (c) 5 + 6 = 7 (False proposition)
  (d) 6 is even number.

Fundamental propositional logic facts:

- The following are some fundamental propositional logic facts:
  - Propositional Logic operates with 0 and 1, so it is also known as Boolean logic.
  - In propositional logic, symbolic variables are used to express the logic, and any symbol can be used to represent a proposition, such as A, B, C, P, Q, R, and so on.

- Propositions can be true or false, but not both at the same time.
- An object, relations or functions, and logical connectives are main parts of propositional logic.
- Logical operators are another name for these connectives.
- The fundamental components of propositional logic are propositions and connectives.
- A connective is a logical operator that joins two sentences together.
- A tautology, also known as a valid sentence, is a proposition formula that is always true.
- Contradiction is a proposition formula that is always false.
- The term "proposition formula" refers to a formula that has both true and false values.
- Statements like "Where is Ram"," "Where are you," and "What is your opinion" are not propositions because they are questions, orders, or opinions.

Syntax of Propositional logic is as follows:

- The allowed sentences for knowledge representation are defined by the syntax of propositional logic.
- Propositions are divided into two categories:
  1. Atomic Propositions
  2. Compound Propositions

1. **Atomic Propositions:**
   - Atomic propositions are the simple propositions. It consists of a single proposition symbol. These are the sentences which must be either true or false.

     Example:

     (a) 3 + 2 = 5, which is both an atomic proposition and a factual fact.

     (b) "The Sun is cool" is likewise a proposition.

2. **Compound propositions**:
   - These propositions are created by combining simpler or atomic propositions and connecting them with parenthesis and logical connectives.

     Example:

     (a) "It is winter season and climate is very cold."

     (b) "Mohan is a physician with a clinic in Mumbai."

Logical Connectives:

- Logical connectives are used to link two simpler ideas or to logically represent a statement. With the use of logical connectives we can form compound assertions.

- There are five primary connectives which are listed below:

1. **Negation:**
   - A statement like $\neg P$ is referred to as a negation of P. There are two types of literals: Positive and Negative Literals.
2. **Conjunction:**
   - A conjunction is sentence that contains a $\wedge$ connective, such as $P \wedge Q$.

     For example, "Raghav is a bright and diligent young man".

     It can be written as,

     P = Raghav is a bright young man.

     Q = Raghav is a diligent young man = $P \wedge Q$.
3. **Disjunction:**
   - A sentence which has $\vee$ connective, such as $P \vee Q$, is called disjunction, where P and Q are the propositions.

     For example, "Sonali is a doctor or an engineer".

     Here P = Sonali is a Doctor. Q = Sonali is an Engineer, so we can write it as $P \vee Q$.
4. **Implication:**
   - A sentence such as $P \rightarrow Q$ is called an implication. If-then rules are another name for implications.

     For example, "If it is summer, then Climate is Hot".

     Here P = It is summer, and Q = Climate is Hot, so it is represented as $P \rightarrow Q$.
5. **Biconditional:**
   - A sentence such as $P \Leftrightarrow Q$ is a Biconditional sentence.

     For example, "If It is raining then you should use umbrella."

     It can be written as $P \Leftrightarrow Q$ where,

     P = It is raining, Q = you should use umbrella.

- Following is the summarized table for Propositional Logic Connectives:

Table 4.1: Propositional Logic Connective Symbols

Connective symbols	Word	Technical term	Example
$\wedge$	AND	Conjunction	$A \wedge B$
$\vee$	OR	Disjunction	$A \vee B$
$\rightarrow$	Implies	Implication	$A \rightarrow B$
$\Leftrightarrow$	If and only if	Biconditional	$A \Leftrightarrow B$
$\neg$ or ~	Not	Negation	$\neg A$ or $\neg B$

Limitations of Propositional logic:

- Propositional Logic does not allow us to conclude the truth of ALL, SOME or NONE statements.

  Examples:

  (a) All the women are hardworking.

  (b) Some people are honest.

- Propositional logic has limited expressive power.
- In propositional logic, we cannot describe statements in terms of their properties or logical relationships.

4.4.2 Predicate Logic

- We have seen how to represent statements using propositional logic. Unfortunately, we can only represent facts that are either true or false in propositional logic. To represent complicated phrases or natural language statements, PL is insufficient. The expressive power of propositional logic is quite restricted.
- Predicate logic is a formal language in which propositions are expressed in terms of predicates, variables and quantifiers.
- Predicates are a fundamental concept in mathematical logic. Predicates express similar kinds of propositions involving its arguments.
- Consider the sentence below, which we can't represent with PL logic.

  Examples:

  - "Some persons are intelligent"
  - "Kapil enjoys cricket"

- As PL logic is insufficient to represent the above statements, we needed to use stronger logic, such as first-order logic.

First-Order Logic (FOL):

- In artificial intelligence, first-order logic is another method of knowledge representtation. It's a variant of propositional logic.
- FOL has enough expressiveness to convey natural language statements effectively.
- Predicate logic or First-order Predicate Logic (FOPL) are other names for first-order logic. First-order logic is a powerful language that makes it easier to build information about objects and to represent relationships between them.
- First-order logic (like natural language) assumes not only that the world includes facts, as propositional logic does, but also that the world has the following things:
  - **Objects**: A, B, persons, numbers, colours, battles, theories, squares, pits, and so on.
  - **Relationships:** They can be unary, such as red, round, or nearby, or n-any, such as sister of, brother of, has colour, or comes between.
  - **Function:** Mother of, best buddy, second inning of, end of.

- As a natural language, First-order logic also has two main parts:
  (a) Syntax
  (b) Semantics

Syntax of First-Order logic:

- The syntax of FOL determines which collection of symbols is a logical expression in first-order logic. The basic syntactic elements of first-order logic are symbols. We write statements in short-hand notation in FOL.

Basic Elements of First-order logic:

- Following are the basic elements of FOL syntax:

Table 4.2: Basic elements of FOL

Constant	1, 2, A, Juhi, Pune, dog,....
Variables	x, y, z, a, b,....
Predicates	Sister, Father, >,,....
Function	sqrt, Left Leg Of, ...
Connectives	$\wedge$ $\vee$, $\neg$, $\Rightarrow$ $\Leftrightarrow$
Equality	==
Quantifier	$\forall$, $\exists$

Types of FOL Sentences:

1. **Atomic sentences:**

- Atomic sentences are the most basic sentences of first-order logic. These sentences are formed from a predicate symbol followed by list of terms.
- We can represent atomic sentences as Predicate (term1, term2, ..., term n).

  Example:

  Ram and Shyam are friends: $\Rightarrow$ friends(Ram, Shyam).

  Peacock is a bird: $\Rightarrow$ bird (Peacock).

2. **Complex Sentences:**

- Complex sentences are made by combining atomic sentences using connectives.
- First-order logic statements can be divided into two parts:
  1. **Subject:** Subject is the main part of the statement.
  2. **Predicate:** A predicate can be defined as a relation, which binds two atoms together in a statement.
- Consider the statement: "x is a flower", it consists of two parts, the first part x is the subject of the statement and second part "is a flower," is known as a predicate.

$$\underbrace{x}_{\text{Subject}} \quad \underbrace{\text{is a flower}}_{\text{Predicate}}$$

Fig. 4.2: Parts of FOL Statement

Quantifiers in First-Order Logic:

- A quantifier is a language element which generates quantification and quantification specifies the quantity of specimen in the universe of discourse.
- These are the symbols that permit to determine or identify the range and scope of the variable in the logical expression.
- There are two types of Quantifier:
  1. Universal Quantifier (for all, everyone, everything)
  2. Existential Quantifier (for some, at least one).

1. Universal Quantifier:

- Universal quantifier is a symbol of logical representation. It specifies that the statement within its range is true for everything or every instance of a particular thing.
- The Universal quantifier is represented by a symbol $\forall$, which look like an inverted A.
- If x is a variable, then $\forall x$ is read as:
  - For all x
  - For each x
  - For every x.

  Example: All man read books.

  Let a variable x which refers to man so it can be represented as below:

  $\forall$ x : man(x) → read (x, books).

  It will be read as: There are all x where x is a man who read books.

2. Existential Quantifier:

- Existential quantifiers are the type of quantifiers. This quantifier express that the statement within its scope is true for at least one instance of something.
- It is denoted by the logical operator $\exists$, which look like as inverted E. When it is used with a predicate variable then it is called as an existential quantifier.
- If x is a variable, then existential quantifier will be $\exists x$ or $\exists(x)$. And it will be read as:
  - There exists a 'x.'
  - For some 'x.'
  - For at least one 'x.'

Example:

Some boys are honest.

$\exists$x: boys(x) $\wedge$ Honest (x)

It will be read as: There are some x where x is a boy who is honest.

Points to remember:

- The main connective for universal quantifier $\forall$ is implication →.
- The main connective for existential quantifier $\exists$ is and $\wedge$.

Properties of Quantifiers:

- In universal quantifier, $\forall x \ \forall y$ is similar to $\forall y \ \forall x$.
- In Existential quantifier, $\exists x \ \exists y$ is similar to $\exists y \ \exists x$.
- $\exists x \forall y$ is not similar to $\forall y \exists x$.

Some Examples of FOL using quantifier:

1. **All Parrots fly.**

   In this example, the predicate is "fly (Parrots)". And since there are all Parrots who fly so it will be represented as follows:

   ∀x : Parrots(x) → fly(x).

2. **Every Student respects his Teacher.**

   In this example, the predicate is "respect(x, y)", where x = Student, and y = Teacher. Since there is every man so will use ∀, and it will be represented as follows:

   ∀x :Student(x) → respects (x, Teacher).

3. **Some girls play Chess.**

   In this example, the predicate is "play(x, y)", where x = girls, and y = Chess. Since there are some girls so we will use ∃, and it will be represented as:

   ∃ x girls(x) → play(x, Chess).

4. **Not all students like both Computers and Marathi.**

   In this question, the predicate is "like(x, y)", where x = student, and y = subject. Since there are not all students, so we will use ∀ with negation, so following representation for this:

   ¬∀(x): [student(x) → like(x, Computers) ∧ like(x, Marathi)].

5. **Only one student passed in Science:**

   In this question, the predicate is "passed(x, y)", where x = student, and y = subject. Since there is only one student who passed in Science, so we will use following representation for this:

   ∃ (x): [student(x) → passed (x, Science) ∧ ∀ (y) [¬(x == y) ∧ student(y) → ¬ passed (x, Science)].

4.5 CONVERSION TO CLAUSE FORM

- Clause Normal Form (CNF) is a sub-language of 1st order logic. A clause is an expression of the form L1 |...| Lm where each Li is a literal. Clauses are denoted by uppercase letters with a superscript |, e.g., C|.
- There are satisfiability preserving transformations from 1st order logic to CNF, i.e., if set of (1st order) formulae are satisfiable (or consistent), then their CNF is satisfiable.
- Conversely, if the CNF of a set of formulae is unsatisfiable, then the formulae are unsatisfiable. This is then useful for showing logical consequence. Computationally, CNF is easier to work with, and is the form used by the resolution inference rule.

Algorithm for Conversion to Clause Form:

- Suppose we know that "all Romans who know Marcus either hate Caesar or think that anyone who hates anyone is crazy".
- This should be represented as in wff(Well-Formed Formulas) as:

  $\forall x\text{: [Roman(x)} \wedge \text{know(x, Marcus)]} \cdot \text{[hate(x, Caesar)} \vee (\forall y\text{:} \exists z\text{:hate(y,z)} \cdot \text{thinkcrazy(x,y))]}$

Step 1 : Elimination of if-then (→ operator):

- Eliminate → using the fact that a → b is equivalent to ¬ a∨b. Performing this transformation on the wff above listed yields,

  $\forall x\text{:[Roman(x)} \wedge \text{know(x,Marcus)]} \vee \text{[hate(x,Caesar)} \vee (\forall y\text{:} \neg(\exists z\text{:hate(y,z)} \vee \text{thinkcrazy(x,y))]}$

Step 2 : Reduction of the scope of negation:

- Reduce the scope of negation to a single term using the fact that $\neg(\neg p) = p$.
- De Morgan's Law says that,

  $\neg(a \wedge b) = \neg a \vee \neg b$

  $\neg(a \vee b) = \neg a \wedge \neg b$

  and the standard correspondences between quantifiers

  $[\neg\forall x\text{:P(x)} = \exists x\text{: } \neg\text{P(x) and } \neg\exists\text{(x): P(x)} = \forall x\text{: } \neg\text{P(x)]}$

  performing this transformation on wff from Step 1,

  $\forall x\text{:[} \neg \text{Roman(x)} \vee \neg\text{know(x,Marcus)]} \vee \text{[hate(x,Caesar)} \vee (\forall y\text{:}\forall z\text{: } \neg\text{hate(y,z)} \vee \text{thingcrazy(x,y))]}$

Step 3 : Renaming the variable within the scope of quantifiers

- Standardize variables so that each quantifier binds a unique variable since variables are just dummy names. This process cannot affect the truth value of the wff.
- For example the formula:

  $\forall x\text{: P(x)} \vee \forall x\text{:Q(x)}$

  Converted to

  $\forall x\text{: P(x) } \forall y\text{: Q(y)}$

Step 4 : Moving of quantifiers in the front of the expression

- Move all quantifiers to the left of the formula without changing their relative order performing this operation on Step 2 we get,

  $\forall x\text{: } \forall y\text{:}\forall z\text{:[} \neg \text{Roman(x)} \vee \neg \text{know(x,Marcus)]} \vee \text{[hate(x,Caesar)} \vee (\neg \text{hate(y,z)} \vee \text{thingcrazy(x,y))]}$
- This is known as **'Prenex normal form'**.

Step 5 : Eliminate existential quantifiers(∃):

- A formula that contains as existentially quantifiable variable assets that there is a value that can be substituted for the variable that makes the formula true. We can eliminate the quantifier by substituting for the variable a reference to a function that produces the desired value.

- Since we do not necessarily know how to produce the value, we must create a new function name for every such replacement. We make no assertion about the functions except that they must exist.
- For example,

  `∃y: President(y)`

  can be transformed into formula as

  `President(S1)`

  where S1 is the function with no argument that somehow produces a vales that satisfies President.
- If existential quantifiers occur within the scope of universal quantifiers then the value that satisfies the predicate may depend on the value of the universally quantified variable.
- For example,

  `∀x: ∃y: father-of(y, x)`

  The value of y that satisfies father-of depends on the particular value of x.
- Thus we must generate functions with the same number of arguments as the number of universal quantifiers in whose scope the expression occurs.
- This example transformed into,

  `∀x: father-of(S2(x),x))`
- This general function is called "**Skolem function**". Sometime ones with no arguments are called '**Skolem constants**'.

Step 6 : Drop the prefix:

`[¬ Roman(x) ∨ ¬know(x,Marcus)] ∨[hate(x,Caesar) ∨(¬hate(y,z) ∨ thingcrazy(x,y))]`

Step 7 : Convert the matrix into a conjunction of disjoints.

- In case of our example, since there are no and's it is only necessary to exploit the associative property of or.

  `¬ Roman(x) ∨ ¬ know(x,Marcus) ∨hate(x,Caesar) ∨ ¬ hate(y,z) ∨ thingcrazy(x,y)`

Step 8 : Create a separate clause corresponding to each conjunct.

- In order for wff to be true, all the clauses that are generated from it must be true.

Step 9 : Standardize the variables in the set of clauses generated in step 8.

- By this, we mean rename the variable so that no two clauses make reference to the same variable. In making this transformation, we rely on the fact that,

  `(∀x: P(x) ∧ Q(x))= ∀x: P(x) ∧∀x:Q(x)`

4.6 RESOLUTION IN PROPOSITIONAL LOGIC

Resolution:

- Resolution is a theorem proving technique that proceeds by building refutation proofs, i.e. proofs by contradictions. It was invented by a Mathematician John Alan Robinson in the year 1965.
- Resolution is used, if there are various statements are given, and we need to prove a conclusion of those statements. Unification is a key concept in proofs by resolutions. Resolution is a single inference rule which can efficiently operate on the conjunctive normal form or clausal form.

Algorithm for Resolution in Propositional Logic:

Step 1 : Convert all propositions to the clause form.

Step 2 : Negate P and convert the result to clause form. Add it to the set of clauses obtained in Step 1.

Step 3 : Repeat until either a contradiction is found or no program can be made.

(a) Select two clauses call these as parent clause.

(b) Resolve them together. The resulting clause called "Resolvent" will be the disjunction of all of the literals of both the parent clauses with the following exception: If there are any pair of literal L and ¬L such that one of the parent clauses contain L and other contains ¬L then select one such pair and eliminate both L and ¬L from the resolvent.

(c) If the resolvent is the empty clause, then a contraction has been found. If it is not then add it to the set of clauses available to the procedure.

Table 4.3: Examples for Propositional Logic

Given Axioms	Converted to clause form
P	Q
$(P \wedge Q) \rightarrow R$	$\neg P \vee \neg Q \vee R$
$(S \vee T) \rightarrow Q$	$\neg S \vee Q$
	$\neg T \vee Q$
T	T

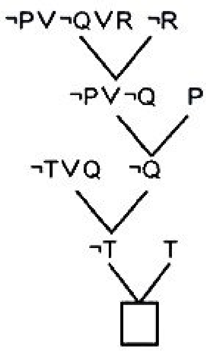

Fig. 4.3

4.7 RESOLUTION IN PREDICATE LOGIC

- **Steps for Resolution:**
  1. Conversion of facts into first-order logic.
  2. Convert FOL statements into CNF.
  3. Negate the statement which needs to prove (proof by contradiction).
  4. Draw resolution graph (unification).
- To better understand all the above steps, we will take an example in which we will apply resolution.

  Example:

  (a) Ram likes all kind of food.

  (b) Grapes and vegetables are food.

  (c) Anything anyone eats and not killed is food.

  (d) Shyam eats Almonds and still alive.

  (e) Sunil eats everything that Anil eats.

  Prove by resolution that:

  (f) Ram likes almonds.

Step 1: Conversion of Facts into FOL

- In the first step we will convert all the given statements into its first order logic.
  1. `∀x: ¬ food(x) → likes(Ram, x)`
  2. `food(Grapes) ∧ food(vegetables).`
  3. `∀x ∀y : eats(x, y) ∧ ¬ killed(x) → food(y)`
  4. `eats (Shyam, Almonds) ∧ alive(Shyam)`
  5. `∀x : eats(Shyam, x) → eats(Sunil, x)`
  6. `∀x: ¬ killed(x) → alive(x)`
  7. `∀x : alive(x) → ¬ killed(x)`
  8. `likes (Ram, Almonds).`

Step 2: Conversion of FOL into CNF

- In First order logic resolution, it is required to convert the FOL into CNF as CNF form makes easier for resolution proofs.

(i) Eliminate all implication (→) and rewrite:
  1. `∀x ¬ food(x) V likes(Ram, x)`
  2. `food(Grapes) ∧ food(vegetables)`
  3. `∀x ∀y ¬ [eats(x, y) ∧ ¬ killed(x)] V food(y)`
  4. `eats (Shyam, Almonds) ∧ alive(Shyam)`
  5. `∀x ¬ eats(Shyam, x) V eats(Sunil, x)`
  6. `∀x¬ [¬ killed(x) ] V alive(x)`
  7. `∀x ¬ alive(x) V ¬ killed(x)`
  8. `likes(Ram, Almonds).`

(ii) Move negation (¬) inwards and rewrite:

1. ∀x ¬ food(x) V likes(Ram, x)
2. food(Grapes) Λ food(vegetables)
3. ∀x ∀y ¬ eats(x, y) V killed(x) V food(y)
4. eats (Shyam, Almond) Λ alive(Shyam)
5. ∀x ¬ eats(Shyam, x) V eats(Sunil, x)
6. ∀x ¬killed(x)] V alive(x)
7. ∀x ¬ alive(x) V ¬ killed(x)
8. likes(Ram, Almond).

(iii) Rename variables or standardize variables:

1. ∀x ¬ food(x) V likes(Ram, x)
2. food(Grapes) Λ food(vegetables)
3. ∀y ∀z ¬ eats(y, z) V killed(y) V food(z)
4. eats (Shyam, Almond) Λ alive(Shyam)
5. ∀w¬ eats(Shyam, w) V eats(Sunil, w)
6. ∀g ¬killed(g)] V alive(g)
7. ∀k ¬ alive(k) V ¬ killed(k)
8. likes(Ram, Almond).

(iv) Eliminate existential instantiation quantifier by elimination:

- In this step, we will eliminate existential quantifier ∃, and this process is known as Solemnization. But in this example problem since there is no existential quantifier so all the statements will remain same.

(v) Drop Universal quantifiers:

- In this step, we will drop all universal quantifier since all the statements are not implicitly quantified so we don't need it.

1. ¬ food(x) V likes(Ram, x)
2. food(Grapes)
3. food(vegetables)
4. ¬ eats(y, z) V killed(y) V food(z)
5. eats (Shyam, Almonds)
6. alive(Shyam)
7. ¬ eats(Shyam, w) V eats(Sunil, w)
8. killed(g) V alive(g)
9. ¬ alive(k) V ¬ killed(k)
10. likes(Ram, Almonds).

(vi) Distribute conjunction ∧ over disjunction ¬.

- This step will not make any change in this problem.

Step 3: Negate the statement to be proved

- In this statement, we will apply negation to the conclusion statements, which will be written as ¬`likes(Ram, Almonds)`.

Step 4: Draw Resolution graph

- Now in this step, we will solve the problem by resolution tree using substitution. For the above problem, it will be given as follows:

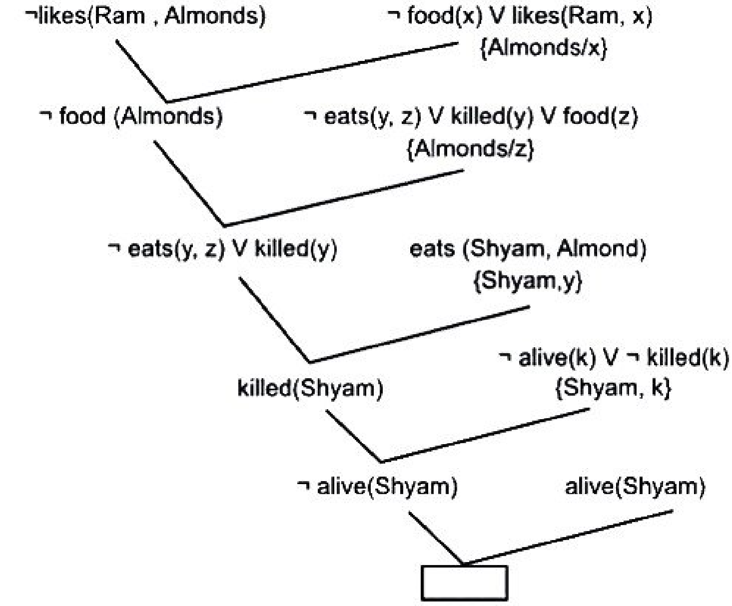

Fig. 4.4: Resolution Graph

- Hence, the negation of the conclusion has been proved as a complete contradiction with the given set of statements.

Summary

- Knowledge representation and reasoning (KR, KRR) are the part of Artificial intelligence which concerned with AI agents thinking and how thinking contributes to intelligent behavior of agents.
- There main two types of knowledge one is Procedural Knowledge and other is Declarative Knowledge.
- There are 4 main approaches to knowledge representations such as Representational accuracy, Inferential adequacy, Inferential efficiency, Acquisitional (learning) efficiency.
- Propositional logic (PL) is the simplest form of logic where all the statements are made by propositions. It is a technique of knowledge representation in logical and mathematical form.
- We cannot represent relations like ALL, some, or none with propositional logic. Propositional logic has limited expressive power.

- First-order logic is another way of knowledge representation in artificial intelligence. It is an extension to propositional logic.
- Resolution is used to prove conclusions.

Check Your Understanding

1. Which is not a property of representation of knowledge?
   (a) Representational Verification (b) Representational Adequacy
   (c) Inferential Adequacy (d) Inferential Efficiency
2. Which of the following option is used to build complex sentences in knowledge representation?
   (a) Symbols (b) Connectives
   (c) Quantifiers (d) None of these
3. In artificial Intelligence, knowledge can be represented as ______.
   (i) Predicate Logic (ii) Propositional Logic
   (iii) Compound Logic (iv) Machine Logic
   (a) Both (i) and (ii) (b) Only (i) and (iii)
   (c) Only (iv) (d) Only (ii)
4. For propositional Logic, which statement is false?
   (a) Can have answer other than true or false
   (b) Each sentence is declarative sentence
   (c) It is a knowledge representation technique in AI.
   (d) None of the above
5. First order logic Statements contains______.
   (a) Predicate and Preposition (b) Subject and an Object
   (c) Predicate and subject (d) None of these
6. The knowledge representation models are based on in which of the following?
   (i) Logic
   (ii) Rules
   (iii) Frames
   (a) (i) and (ii) (b) (i) and (iii)
   (c) (ii) and (iii) (d) All of the above
7. A knowledge representation system should have which of the following properties?
   (i) Representation Adequacy
   (ii) Inferential Adequacy
   (iii) Inferential Efficiency
   (a) (i) and (ii) (b) (ii) and (iii)
   (c) (i) and (iii) (d) All (i), (ii) and (iii)

8. ______ is the ability to represent all kinds of knowledge that are needed in that domain.
   (a) Representation Adequacy (b) Inferential Adequacy
   (c) Inferential Efficiency (d) Acquisitional Efficiency
9. ______ is the ability to manipulate the representational structures to derive new structures corresponding to new knowledge inferred from old.
   (a) Representation Adequacy (b) Inferential Adequacy
   (c) Inferential Efficiency (d) Acquisitional Efficiency
10. ______ is the ability to acquire new knowledge using automatic methods wherever possible rather than reliance on human intervention.
   (a) Representation Adequacy (b) Inferential Adequacy
   (c) Inferential Efficiency (d) Acquisitional Efficiency
11. The problem solving requires which of the following:
   (i) Formal knowledge representation
   (ii) Conversion of informal knowledge to formal knowledge
   (iii) Conversion of formal knowledge to informal knowledge
   (a) (i) and (ii) (b) (ii) and (iii)
   (c) (i) and (iii) (d) All (i), (ii) and (iii)
12. ______ is the ability to incorporate additional information into the knowledge structure that can be used to focus the attention of the inference mechanisms in the most promising direction.
   (a) Representation Adequacy (b) Inferential Adequacy
   (c) Inferential Efficiency (d) Acquisitional Efficiency
13. Which of the following statements correctly define knowledge representation in AI?
   (a) It is the way in which facts and information are stored in the storage system of the agent.
   (b) It is the way in which we feed the knowledge in machine understandable form.
   (c) We modify the knowledge and convert it into the format which is acceptable by the machine.
   (d) All of the above

Answers

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

Practice Questions

Q.I Answer the following questions in short.

1. Define Knowledge.
2. Explain Propositional Logic with example.
3. Explain Predicate Logic with example.
4. What is Resolution? Explain with example.
5. Which are types of knowledge are there in AI?
6. What are the different approaches to knowledge representation?
7. Identify which of the following statements are propositions-
   (a) France is a country.
   (b) 2024 will be a leap year.
   (c) Sun rises in the west.
   (d) Are you tired?

Q.II Answer the following questions.

1. How does predicate logic help with knowledge representation?
2. Consider the following 3 FOPL statements. Using resolution, prove FIDO WILL DIE
   (i) $\forall x{:}\mathrm{dog}(x) \rightarrow \mathrm{animal}(x)$
   (ii) Dog(FIDO)
   (iii) $\forall y{:}\mathrm{animal}(y) \rightarrow \mathrm{die}(y)$
3. Consider the following statements:
   (i) All philosophers are Indian.
   (ii) All Indians are happy.
   (iii) Either Aryabhatta or C.V.Raman is a Philosopher.
   (iv) C.V.Raman is not a Philosopher.
   Represent the above information in wff and prove that "Aryabhatta is happy".
4. Translate the following statements to FOPL.
   (i) There is a mushroom that is purple and poisonous.
   (ii) There is a bunny who is cute.
5. Consider the following axioms:
   (i) All hounds howl at night.
   (ii) Anyone who has any cats will not have any mice.
   (iii) Light sleepers do not have anything which howls at night.
   (iv) Ram has either a cat or a hound.
   (Conclusion) If Ram is a light sleeper, then Ram does not have any mice. Convert the above axioms to wff in First Order Predicate Logic and transform each clause to conjunctive normal form.

6. Consider the following axioms:
   (i) Every child loves Santa.
   (ii) Everyone who loves Santa loves any reindeer.
   (iii) Rudolph is a reindeer, and Rudolph has a red nose.
   (iv) Anything which has a red nose is weird or is a clown.
   (v) No reindeer is a clown.
   (vi) Scrooge does not love anything which is weird.
   (Conclusion) Scrooge is not a child.
   Represent these axioms in predicate calculus; skolemize as necessary and convert each formula to clause form. Prove the unsatisfiability of the set of clauses by resolution.
7. Consider the following axioms:
   (i) Anyone whom Mary loves is a football star.
   (ii) Any student who does not pass does not play.
   (iii) John is a student.
   (iv) Any student who does not study does not pass.
   (v) Anyone who does not play is not a football star.
   (Conclusion) If John does not study, then Mary does not love John.
   Represent these axioms in predicate calculus; skolemize as necessary and convert each formula to clause form. Prove the unsatisfiability of the set of clauses by resolution.
8. Explain the algorithm for Resolution in Propositional Logic.

Q.III Define the terms.

1. Declarative Knowledge
2. Propositional Logic
3. First-Order Logic
4. Clause Normal Form
5. Resolution

■■■

5... Slot and Filler Structures

Learning Objectives ...

- To learn about Weak slot and Filler Structures (Semantic Networks and Frame)
- To study Strong Structures (Conceptual Dependencies and Script)

5.1 WEAK STRUCTURES (SEMANTIC NETWORKS AND FRAME)

Introduction to Weak Slot and Filler Structure:

- Slot and Filler Structure support property inheritance. The knowledge in slot and filler systems consists of structures as a set of entities and their attributes. This structure is called a weak slot and filler structure.
- A slot is an attribute value pair in its simplest form.
- A filler is a value that a slot can take could be a numeric, string (or any data type) value or a pointer to another slot.
- A weak slot and filler structure does not consider the content of the representation. This structure also known as "Knowledge Poor" Structure.

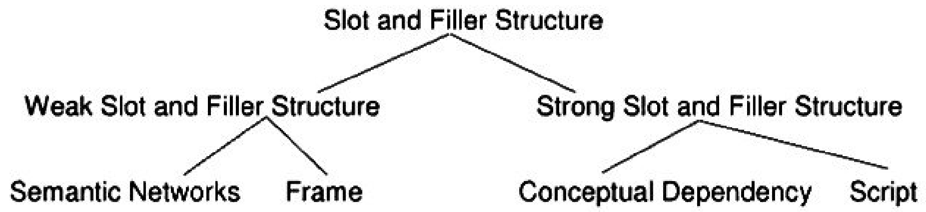

Fig. 5.1: Classification of Slot and Filler Structure

Types of Weak Slot and Filler Structure:

1. Semantic Networks
2. Frame

5.1.1 Semantic Networks

- The easiest form of knowledge representation is semantic network. Semantic network is an alternative to predicate logic as a form of knowledge representation. It uses visual representation of knowledge in the form of Graph.

- In semantic network, information is represented as set of nodes representing objects in real world, set of labeled arcs represent relationships among the nodes. It uses an instance of relationship.

Steps to create semantic network:

1. Create new nodes to represent new objects either contained or alluded to in the knowledge.
2. Relate information to nodes and fill up slots.

Inference in Semantic Net:

- The main purpose of inference mechanism is to follow the links between the nodes. There are two methods for this purpose:
  1. **Inheritance:** Inheritance is used to represent the links/relationship between objects. The *is-a* and *instance* relationship provide a mechanism to implement inheritance.
  2. **Intersection search:** Intersection search includes representing and finding the relationships between objects by spreading activation from each of two nodes and seeing where the activation meets by assigning a special tag to each visited node using semantic network.

Example of Semantic Network:

- The following example contains both *is-a* and *instance* as well as domain specific relation as team and uniform color.

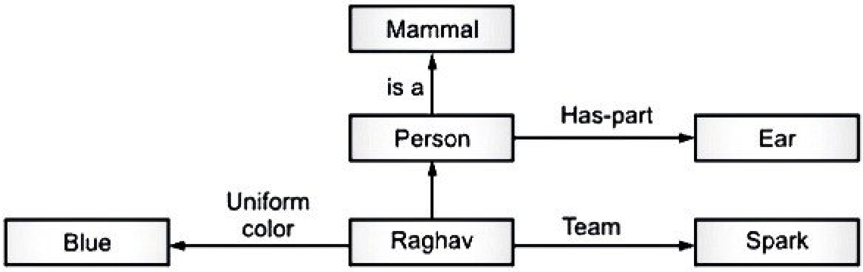

Fig. 5.2: Example of Semantic Network

Representing non-binary predicates:

1. Unary form predicate can be represented as follows:

   For example, Man (Hitler) can be converted into: instance (Hitler, Man).
2. Other articles can be represented as, Score(India, Australia, 4-1).
3. Three or more place predicates can be converted to binary form as follows:
   - Create new object representing the entire predicate.
   - Introduce binary predicates to describe relation to this new object.
   - One complex example representing sentence **Ram gives fruits to Meena.**

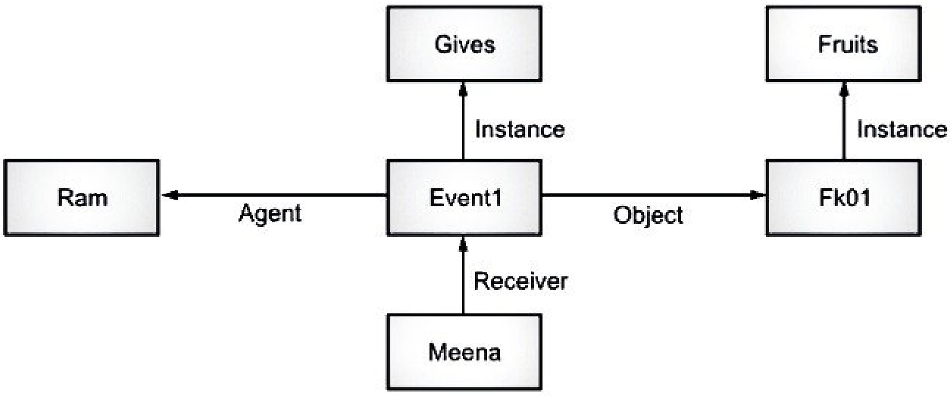

Fig. 5.3: Example of Binary Predicate

Partitioned Semantic Nets:

- Suppose we want to represent simple quantified expressions in semantic nets. One way to do this is to partition the semantic net into a hierarchical set of spaces, each of which corresponds to the scope of one or more variables.
- For example, consider the following sentence:
  Batsmen hit the ball can be represented using semantic net as,

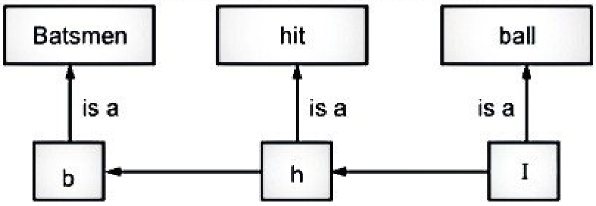

Fig. 5.4: Example of Semantic Net

- The nodes Batsmen, hit and ball represent the classes of Batsmen, hitting, ball respectively. While b, h, l represent a particular Batsman, hitting, particular ball. Therefore, it is easy to represent the above fact using semantic net with no partitioning.
- But if we want to represent the fact-Every batsman has hit a ball which can be represented in logic as

  $\forall x\text{: Batsman}(x) \rightarrow \exists y\text{: ball}(y) \wedge \text{hit}(x,y)$
- Above fact can be presented using partitioning semantic net as:

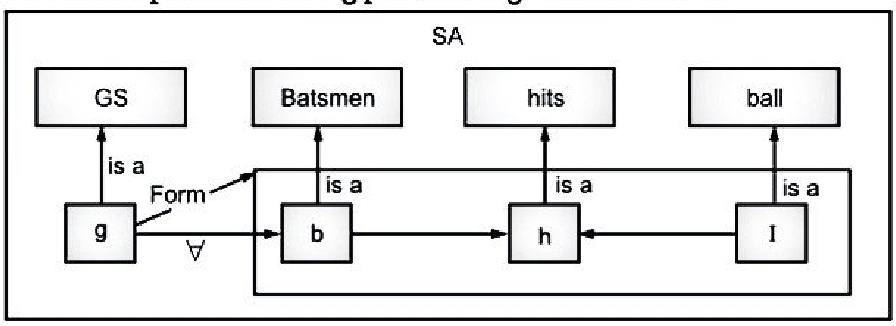

Fig. 5.5: Example of Partitioning Semantic Net

- In the above diagram, **GS** is a special class, of general statement. The node **g** is an instance of GS about the world (i.e. those with universal quantifiers). Every element of GS has at least two attributes:
  1. A **form** that states which relation is being asserted.
  2. One or more (for all) ($\forall$) or There exists($\exists$) connections: These represent universally quantifiable variables in such statement. For example x, y.
- In above example, variable b can stand for any element of the class Batsmen, and other variables h and l are understood existentially quantified.

  Example 2: Ram scored 88% . Manish scored less percentage than Ram.

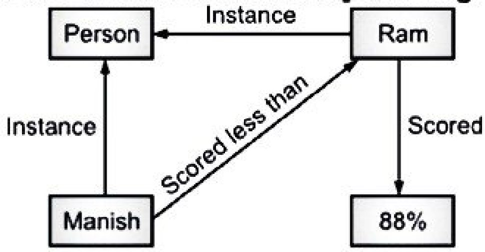

Fig. 5.6: Semantic Net for Example 2

Example 3: Ramesh fixes the chair by fevicol.

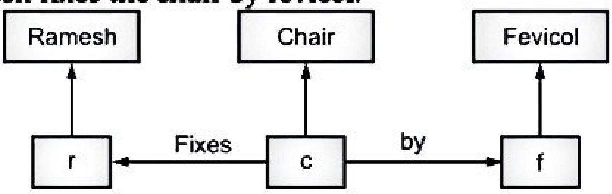

Fig. 5.7: Semantic Net for Example 3

5.1.2 Frames

- Frame is an extension to semantic network. A frame is a record like structure which consists of a collection of attributes and its values to describe an entity in the world.
- Frames are the AI data structure which divides knowledge into substructures by representing stereotypes situations. It consists of a collection of slots and slot values. These slots may be of any type and sizes.
- **Facets:** The various aspects such as names and values of a slot are known as Facets. Facets are features of frames which enable us to put constraints on the frames. Example: IF-NEEDED facts are called when data of any particular slot is needed. A frame may consist of any number of slots, and a slot may include any number of facets. Facets may have any number of values. A frame is also known as slot-filter knowledge representation in artificial intelligence.
- Frames are derived from semantic networks and later evolved into our modern-day classes and objects. A single frame is not much useful. Frames system consists of a collection of frames which are connected. In the frame, knowledge about an object or event can be stored together in the knowledge base.
- The frame is a type of technology which is widely used in various applications including Natural language processing and machine visions.

Example 1: Let us take an example of a frame for a 'book' entity.

Table 5.1: Frame representation for Example 1

Slots	Fillers
Title	Artificial Intelligence
Stream	Computer Science
Author	Rich Knight
Edition	Fourth Edition
Year	1990
Pages	100

Example 2: Let's suppose we are taking an entity-Ajay. Ajay is Teacher as a profession, and his age is 50; he lives in city Pune, and the State is Maharashtra. So following table shows the frame representation for this:

Table 5.2: Frame representation for Example 2

Slots	Fillers
Name	Ajay
Profession	Teacher
Age	50
City	Pune
State	Maharashtra

Example 3: Let's suppose we are taking an entity-Sachin. Sachin is Cricket player, and his age is 30; he lives in city Pune, and the State is Maharashtra. So following table shows the frame representation for this:

Table 5.3: Frame representation for Example 3

Slots	Fillers
Name	Sachin
Profession	Cricket Player
Age	30
City	Pune
State	Maharashtra

Advantages of frame representation:

1. The frame knowledge representation makes the programming easier by grouping the related data.
2. The frame representation is comparably flexible and used by many applications in AI.
3. It is very easy to add slots for new attributes and relations.
4. It is easy to include default data and to search for missing values.
5. Frame representation is easy to understand and visualize.

Disadvantages of frame representation:

1. In frame system inference mechanism is not be easily processed.
2. Inference mechanism cannot be smoothly proceeded by frame representation.
3. Frame representation has a much generalized approach.

5.2 STRONG STRUCTURES (CONCEPTUAL DEPENDENCIES AND SCRIPT)

Strong Slot and Filler Structure:

- Strong Slot and Filler Structure typically,
  - Represent links between objects according to more *rigid* rules.
  - Specific notions of what types of object and relations between them are provided.
  - Represent knowledge about common situations.

Types of strong slot and filler structure:

- There are following types of strong slot and filler structure:
  1. Conceptual Dependency (CD)
  2. Scripts
  3. Cyc

5.2.1 Conceptual Dependency (CD)

- Conceptual Dependency originally developed to represent knowledge acquired from natural language input.
- It is independent of the language in which the sentences were originally stated.
- CD representations of a sentence are built out of primitives that are not words belonging to the language but are conceptual. These primitives are combined to form the meanings of the words. As an example consider the event represented by the sentence.

Goals of CD:

- The goals of this theory are:
  - To help in the drawing of inference from sentences.
  - To be independent of the words used in the original input.
  - That is to say: For any two (or more) sentences that are identical in meaning there should be only one representation of that meaning.

CD provides:

- A structure into which nodes representing information can be placed.
- A specific set of primitives.
- At a given level of granularity.

- Sentences are represented as a series of diagrams showing actions using both abstract and real physical situations.
  - The agent and the objects are represented.
  - The actions are built up from a set of primitive acts which can be modified by tense.

Examples of Primitive Acts:

- **ATRANS:** Transfer of an abstract relationship. e.g. give.
- **PTRANS:** Transfer of the physical location of an object. e.g. go.

- **PROPEL**: Application of a physical force to an object. e.g. push.
- **MTRANS**: Transfer of mental information. e.g. tell.
- **MBUILD**: Construct new information from old. e.g. decide.
- **SPEAK**: Utter a sound. e.g. say.
- **ATTEND**: Focus a sense on a stimulus. e.g. listen, watch.
- **MOVE**: Movement of a body part by owner. e.g. punch, kick.
- **GRASP**: Actor grasping an object. e.g. clutch.
- **INGEST**: Actor ingesting an object. e.g. eat.
- **EXPEL**: Actor getting rid of an object from body. e.g. cry.

Primitive conceptual categories:

- Six primitive conceptual categories provide *building blocks* which are the set of allowable dependencies in the concepts in a sentence:
  1. **PP**: Real world objects.
  2. **ACT**: Real world actions.
  3. **PA**: Attributes of objects.
  4. **AA**: Attributes of actions.
  5. **T**: Times.
  6. **LOC**: Locations.
- Consider the example: **I gave a book to a man.**

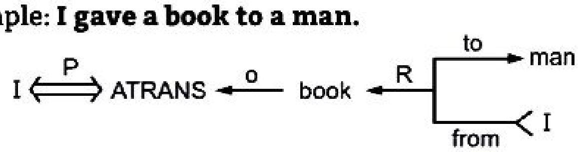

Fig. 5.8: Example of Conceptual Dependency

- Arrows indicate the direction of dependency. Letters above arrows indicate certain relationships as:
  - **o**: Object.
  - **R**: Recipient-donor.
  - **I**: Instrument e.g. eat with a spoon.
  - **D**: Destination e.g. going home.
- Double arrows (⇔) indicate *two-way* links between the actor (PP) and action (ACT).
- The actions are built from the set of primitive acts.
  - These can be modified by tense etc.
  - The use of tense and mood in describing events is extremely important and the following are the modifiers:
    - **p:** past
    - **f:** future
    - **t:** transition
    - t_s: start transition
    - t_f: finished transition
    - **k:** continuing

- ?: interrogative
- /: negative
- **delta:** timeless
- **c:** conditional

- The absence of any modifier implies the present tense.

Examples of CD are:

1. Ved ate ice-cream with spoon.

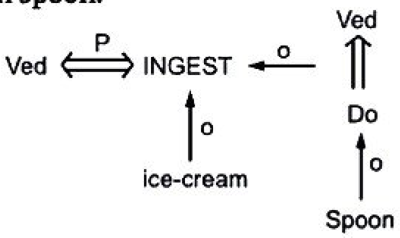

Fig. 5.9: Conceptual dependency for Example 1

2. Tushar sold his car to Vaibhav.

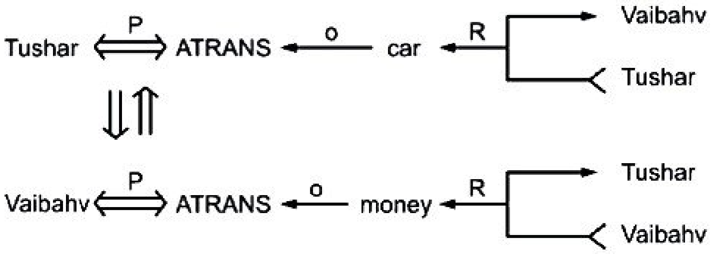

Fig. 5.10: Conceptual dependency for Example 2

3. Radha threw a ball to Mohan.

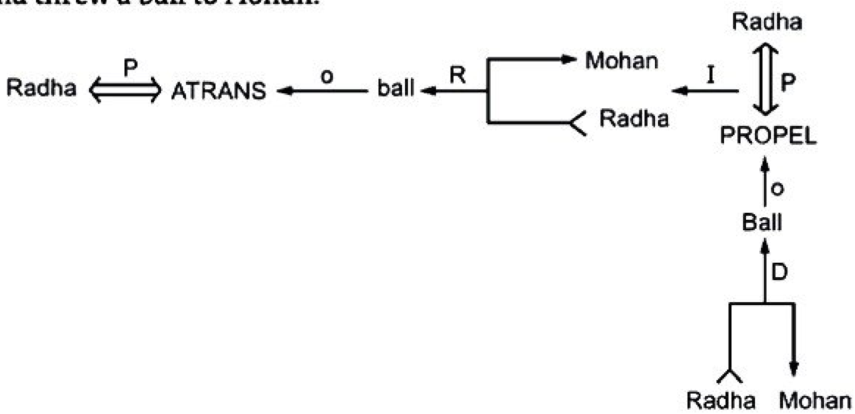

Fig. 5.11: Conceptual dependency for Example 3

4. Ramesh went to Delhi.

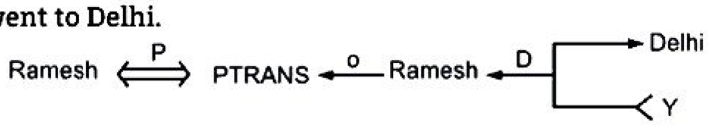

Fig. 5.12: Conceptual dependency for Example 4

5. Seeta cried.

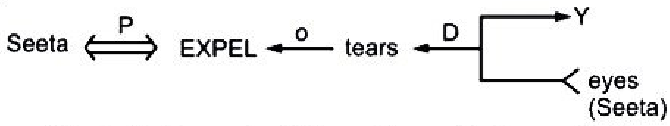

Fig. 5.13: Conceptual dependency for Example 5

6. John killed Mary by throwing a rock at her.

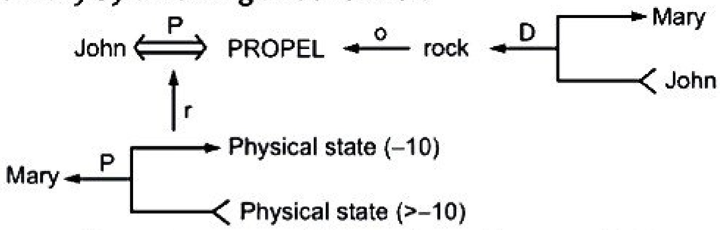

Fig. 5.14: Conceptual dependency for Example 6

Advantages of Conceptual dependency:

1. Using these primitives involves fewer inference rules.
2. Many inference rules are already represented in CD structure.
3. The holes in the initial structure help to focus on the points still to be established.

Disadvantages of Conceptual dependency:

1. Knowledge must be decomposed into fairly low level primitives.
2. Impossible or difficult to find correct set of primitives.
3. A lot of inference may be still required.
4. Representations can be complex even for relatively simple actions. For example, Consider, 'Andy bet Charles seven pounds that Wales would win the Rugby World Cup.'
5. Such Complex representations require a lot of storage.

Applications of CD:

1. **MARGIE**(Meaning Analysis, Response Generation and Inference on English): model natural language understanding.
2. **SAM**(Script Applier Mechanism): Scripts to understand stories.
3. **PAM**(Plan Applier Mechanism): Scripts to understand stories.

5.2.2 Script

- A script is a structure that describes a stereotyped sequence of events in a particular context. Also it is a set of circumstances which could be expected to follow on from one another.
- A script consists of a set of slots; some information is always associated with each slot. It is similar to a thought sequence or a chain of situations which could be anticipated. It could be considered to consist of a number of slots or frames but with more specialized roles.

- Scripts are beneficial because:
  - Events tend to occur in known runs or patterns.
  - Causal relationships between events exist.
  - Entry conditions exist which allow an event to take place.
  - Prerequisites exist upon events taking place. For example, when a student progresses through a degree scheme or when a purchaser buys a house.

Components of a Script:

- The components of a script include:
  - **Entry Conditions**: These must be satisfied before events in the script can occur.
  - **Results**: Conditions that will be true after events in script occur.
  - **Props**: Slots representing objects involved in events.
  - **Roles**: Persons involved in the events.
  - **Track**: Variations on the script. Different tracks may share components of the same script.
  - **Scenes**: The sequence of *events* that occur. *Events* are represented in *conceptual dependency* form.

  Examples: Scripts are useful in describing certain situations. Some examples are given below:

Example 1: Script for robbing a bank.

- This might involve:
  - Getting a gun.
  - Hold up a bank.
  - Escape with the money.
- Here the **Props** might be:
  - Gun, *G*.
  - Loot, *L*.
  - Bag, *B*
  - Get away car, *C*.
- The **Roles** might be:
  - Robber, *S*.
  - Cashier, *M*.
  - Bank Manager, *O*.
  - Policeman, *P*.
- The **Entry Conditions** might be:
  - *S* is poor.
  - *S* is destitute.

- The **Results** might be:
  - *S* has more money.
  - *O* is angry.
  - *M* is in a state of shock.
  - *P* is shot.

  There are 3 scenes: Obtaining the gun, Robbing the bank and then Get away.
- The full Script could be described as follows:

<table>
<tr><th>Script: ROBBERY</th><th>Track: Successful Snatch</th></tr>
<tr><td>Props:
• G = Gun
• L = Loot
• B = Bag
• C = Get away car</td><td>Roles:
• R = Robber
• M = Cashier
• O = Bank Manager
• P = Policeman</td></tr>
<tr><td>Entry conditions:
• R is poor
• R is destitute</td><td>Results:
• R has more money
• O is angry
• M is in a state of shock
• P is shot</td></tr>
<tr><td colspan="2">Scene 1: Getting a gun
• R PTRANS R into Gun Shop
• R MBUILD R choice of G
• R MTRANS choice
• R ATRANS buys G
(go to scene 2)</td></tr>
<tr><td colspan="2">Scene 2: Holding up the bank
• R PTRANS R into bank
• R ATTEND eyes M, O and P
• P MOVE R to M position
• R GRASP G
• R MOVE G to point to M
• R MTRANS "Give me the money or ELSE" to M
• R MTRANS "Hold it Hands Up" to R
• R PROPEL shoots G
• P INGEST bullet from G
• M ATRANS L to M
• M ATRANS L puts in bag B
• M PTRANS exit
• O ATRANS raises the alarm
(go to Scene 3)</td></tr>
<tr><td colspan="2">Scene 3: The getaway
• M PTRANS C</td></tr>
</table>

Example 2: Script for watching cricket Match.

Script : Watching a match	Various Scenes
Track: Cricket match **Props:** • Tickets • Seat • Match **Roles:** • Person (who wants to Watch a match) – P • Booking Clerk – BC • Security personal – SP • Ticket Checker - TC **Entry Conditions:** • P wants to watch match • P has a money **Re results:** • P saw a match • P has less money • P is happy (if his team has won) or not (if his team has lost) or some other problem at stadium	**Scene 1: Going to a stadium** • P PTRANS P to the stadium • P ATTEND eyes to ticket counter **Scene 2: Buying ticket** • P PTRANS P to ticket counter • P MTRANS ticket requirement to BC • P MTRANS stand information to BC • BC ATRANS ticket to P **Scene 3: Going inside stadium and sitting on a seat** • P PTRANS P into Stadium • SP MOVE security check P • TC ATTEND eyes on ticket POSS by P • TC MOVE to tear part of ticket • TC MTRANS (showed seat) to P • P PTRANS P to seat • P MOVE P to sitting position **Scene 4: Watching a match** • P ATTEND eyes on match • P MBUILD (moments) from the match **Scene 5: Exiting** • P PTRANS P out of Stadium

Example 3: Restaurant Script.

<table>
<tr><th>Script : RESTAURANT</th><th>Various Scenes</th></tr>
<tr><td rowspan="4">Track: Coffee shop
Props:
• Tables
• Menu
• F = Food
• Check
• Money
Roles:
• S = Customer
• W = Waiter
• C = Cook
• M = Cashier
• O = Owner

Entry Conditions:
• S is hungry
• S has money

Results:
• S has less money
• O has more money
• S is not hungry
• S is pleased (optional)</td>
<td>Scene 1: Entering
• S PTRANS S into restaurant
• S ATTEND eyes to tables
• S MBUILD where to sit
• S PTRANS S to table
• S MOVE S to sitting position</td></tr>
<tr><td>Scene 2: Ordering
(Menu on table) (W bring menu) (S asks for menu)
S PTRANS menu to S
S MTRANS signal to W
W PTRANS W to table
S MTRANS 'need menu' to W
W PTRANS W to menu

W PTRANS W to table
W ATRANS menu to S

S MTRANS W to table
*S MBUILD choice of F
S MTRANS signal to W
W PTRANS W to table
S MTRANS 'I want F' to W

W PTRANS W to C
W MTRANS (ATRANS F) to C

C MTRANS 'no F' to W
W PTRANS W to S
W MTRANS 'no F' to S
(go back to *)

C Do (Prepare F script)
to Scene 3

(go to Scene 4 at no pay path)</td></tr>
<tr><td>Scene 3: Eating
• C ATRANS F to W
• W ATRANS F to S
• S INGEST F
(Option: Return to Scene 2 to order more
Otherwise, go to Scene 4)</td></tr>
<tr><td>Scene 4: Exiting
S MTRANS S to W
(W ATRANS check to S)

W MOVE (Write check)
W PTRANS W to S
W ATRANS check to S
S ATRANS tip to W
S PTRANS S to M
S ATRANS money to M
(No pay path) S PTRANS S to out of restaurant</td></tr>
</table>

Important points to remember about Script:

- If a particular script is to be applied it must be activated and the activating depends on its significance.
- If a topic is mentioned in passing then a pointer to that script could be held.
- If the topic is important then the script should be opened.
- The danger lies in having too many active scripts much as one might have too many windows open on the screen or too many recursive calls in a program.
- Provided events follow a known trail we can use scripts to represent the actions involved and use them to answer detailed questions.
- Different trails may be allowed for different outcomes of Scripts (e.g. The bank robbery goes wrong).

Advantages:

1. Ability to predict events.
2. A single coherent interpretation may be build up from a collection of observations.

Disadvantages:

1. Less general than frames.
2. May not be suitable to represent all kinds of knowledge.

5.2.3 Cyc

- **Cyc** is a long-term artificial intelligence project that attempts to assemble a comprehensive ontology and database of everyday common-sense knowledge.
- The *Cyc* project began in July 1984 by Douglas Lenat.
- Its initial goal is to capture knowledge from a hundred randomly selected articles in the En*CYC*lopedia Britannica.
- Both Implicit and Explicit knowledge encoded.
- Importance given on study of underlying information (assumed by the authors but not needed to tell to the readers).
- The original version of CycL was a frame language, but the modern version is not. Rather, it is a declarative language.
- Cyc has been used in commercial web-search systems (e.g. HotBot) and in question-answering systems, most recently in a purely deductive system for answering AP chemistry questions, developed in collaboration with Vulcan, Inc
- Cyc allows multiple inheritance (multiple is-a parents): for example, Intangible Stuff has Intangible Object and Stuff as parents. It uses Collection for higher order types.

CYC Motivations:

- Why build large knowledge bases?
  1. **Brittleness**: Specialized knowledge bases are *brittle*. Hard to encode new situations and non-graceful degradation in performance. Commonsense based knowledge bases should have a firmer foundation.

2. **Form and Content**: Knowledge representation may not be suitable for AI. Commonsense strategies could point out where difficulties in content may affect the form.
3. **Shared Knowledge**: Should allow greater communication among systems with common bases and assumptions.

Structure of Cyc:

- Cyc uses a declarative language called CycL based on first-order logic.
- CycL was written in Lisp and has a similar syntactical appearance.
- The main lexical component is a set of concepts, also called constants, which begin with `#$`. These constants can be specific objects, collections of objects, or relations between objects.
- For example, `(#$isa #$DouglasLenat #$AIResearcher)` means "Douglas Lenat is an AI researcher." This is an example of a fact in the Knowledge Base.

  Example: Suppose we read that *Wellington learned of Napoleon's death.*

  Then we (humans) can conclude *Napoleon never new that Wellington had died.*

How do we do this?

- We require special implicit knowledge or common sense such as:
  - We only die once.
  - You stay dead.
  - You cannot learn of anything when dead.
  - Time cannot go backwards.

Summary

- There are two types of Slot and filler structures: Weak slot and filler structure and Strong slot and filler structure.
- The knowledge in slot and filler systems consists of structures as a set of entities and their attributes.
- Slot is an attribute and filler is value associated with attribute.
- Weak slot and filler structure consist of two types: semantic network and frame.
- Semantic network is a graphical representation of information consists of node and labelled arc.
- A frame is a record like structure which consists of a collection of attributes and its values to describe an entity.
- There are main two types of strong slot and filler structure: Conceptual Dependency and Script.
- Conceptual Dependency is used to help in the drawing of inference from sentences.
- A script is a structure that describes a stereotyped sequence of events in a particular context.
- CYC is determined attempt to form a very large knowledge base aimed at capturing commonsense reasoning.

Check Your Understanding

1. What is the frame?
   (a) A way of representing knowledge
   (b) Data Structure
   (c) Data Type
   (d) None of the mentioned
2. Which of the following elements constitutes the frame structure?
   (a) Facts or Data
   (b) Procedures and default values
   (c) Frame names
   (d) Frame reference in hierarchy
3. The basic inference mechanism in semantic network in which knowledge is represented as ______ is to follow the links between the nodes.
   (a) Frames
   (b) filler
   (c) slots
   (d) blocks
4. "There exists two ways to infer using semantic networks in which knowledge is represented as Frames"
   (a) Intersection Search
   (b) Inheritance Search
   (c) Multiheritance search
   (d) Network search
5. The two basic axioms of CD are ______.
   (a) For any two sentences that are identical in meaning, regardless of language, there should be only one representation.
   (b) Any information in a sentence that is implicit must be made explicit in the representation of the meaning of that sentence.
   (c) Both (a) & (b)
   (d) None of the above
6. Basic idea of ______ is to break the network into spaces that consist of groups of nodes and arcs and regard each space as a node.
   (a) Partitioned Semantic Networks
   (b) Simple semantic Network
   (c) Graph
   (d) None of the above
7. A ______ is a structure that prescribes a set of circumstances that could be expected to follow on from one another.
   (a) Semantic Network
   (b) Script
   (c) Frame
   (d) CD

8. ______ refers to a transfer of possession – the abstract transfer of possession from one person to another, as in give or a buy.
   (a) ATRANS (b) PTRANS
   (c) MTRANS (d) EXPEL
9. ______ refers to the transmission of an IDEA – some conceptualization is transmitted from one head to another (or within the same head).
   (a) ATRANS (b) PTRANS
   (c) MTRANS (d) EXPEL
10. ______ refers to a person taking something outside his or her body: spitting, crying, sweating, etc.
   (a) ATRANS (b) PTRANS
   (c) MTRANS (d) EXPEL
11. Which of the following is an extension of the semantic network?
   (a) Expert Systems (b) Rule Based Expert Systems
   (c) Decision Tree Based networks (d) Partitioned Networks
12. Graph used to represent semantic network is ________.
   (a) Undirected graph (b) Directed graph
   (c) Directed Acyclic Graph (DAG) (d) Directed complete graph

Answers

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

Practice Questions

Q.I Answer the following questions in short.

1. Which are components of Script?
2. Represent the following sentence into the appropriate semantic network diagram. "Isha loaned the book to Ria".
3. What does PTRANS Primitive act indicate in a conceptual dependency representation?
4. Construct semantic net representation for:
   (i) Doctor (marcus), Black smith (marcus).
   (ii) Priya gave the Pink flowered vase to her teacher.
5. What does PROPEL Primitive act indicate in a conceptual dependency representation?
6. Represent using semantic net:
   (i) Every girl likes ice cream.
   (ii) I own black color car.

7. Write a scripts for:
    (i) Watching cricket match
    (ii) Going to market
    (iii) Restaurant

Q.II Answer the following questions.

1. Explain all primitive acts in Conceptual dependency.
2. Write short note on semantic network.
3. Compare script and frame using example.
4. Write short note on scripts.
5. Write short note on frames.
6. Write advantages and disadvantages of frame representation.

Q.III Define the terms.

1. Slot
2. Filler
3. Facets
4. Cyc
5. Weak slot

■■■

6... Recent Trends in AI and Applications

Learning Objectives ...

- To introduce concept of Machine Learning.
- To know about types of Learning, (Supervised, Unsupervised and Reinforcement Learning).
- To get information of Predictive Analytics (Weather Forecasting).
- To study about AI-Powered Chatbots (SBI card Chatbot (ILA)).

6.1 INTRODUCTION TO MACHINE LEARNING

- In the actual world, we are surrounded by individuals who can learn anything from their experiences due to its capacity to learn, and we have computers or machines that follow our commands. But, like a human, can a machine learn from past experiences or data? So this is where Machine Learning comes in picture.
- Machine Learning (ML) is a subfield of Artificial Intelligence (AI) that focuses on the development of algorithms that allow a computer to learn on its own from data and previous experiences.
- Arthur Samuel popularized the term "Machine Learning" in 1959.

Definition:

- "Machine learning allows a machine to learn from data, improve performance based on experiences, and try to predict things without having to be explicitly programmed. A machine can learn if it can improve its performance by gaining new data."
- Machine learning algorithms create a mathematical model with the help of sample historical data, referred to as training data that helps in making predictions or decisions without being explicitly programmed. In order to create predictive models, machine learning combines Computer science and Statistics.
- Machine learning is the process of creating or employing algorithms that learn from historical data. The more information we supply, the better our performance will be.

What is Machine Learning and how does it work?

- A Machine Learning system learns from previous data constructs prediction models and predicts the result whenever fresh data is received.
- Learning is used within the field of data analytics to devise complex models and algorithms that lend themselves to prediction. In commercial use, this is known as Predictive Analytics.
- The accuracy of predicted output is dependent on the amount of data, as a large amount of data helps to build a better model that predicts the output more accurately. Through learning from previous relationships and trends in the data set, these analytical models enable researchers, data scientists, engineers, and analysts to "create dependable, repeatable judgments and results" and uncover "hidden insights" (input).
- If we have a complex situation for which we need to make predictions, rather than writing code for it, we may just input the data to generic algorithms, and the machine will develop the logic based on the data and forecast the outcome. Machine learning has shifted our perspective on the issue.
- The following block diagram shows how the Machine Learning algorithm works:

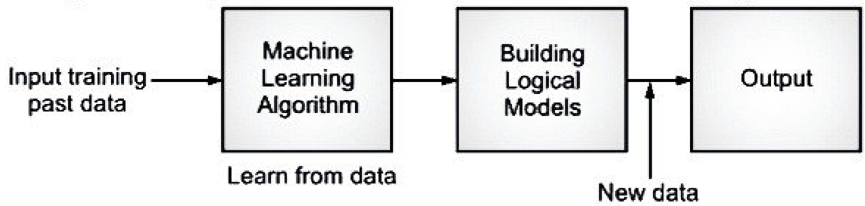

Fig. 6.1: Working of Machine Learning Algorithm

Characteristics:

- Machine Learning has the following characteristics:
  - o Machine learning makes use of data to find patterns in a dataset.
  - o It can learn from previous data and improve on its own.
  - o It is a technology that is based on data.
  - o Machine learning and data mining are very similar in that they both deal with large amounts of data.

Need for Machine Learning:

- Machine learning is becoming increasingly important. Machine learning is required because it is capable of performing tasks that are too complex for a human to perform directly. As humans, we have some limits in that we cannot manually access vast amounts of data, needing the use of computer systems, which brings us to machine learning.
- We can train machine learning algorithms by giving them with a large amount of data and allowing them to examine the data autonomously, build models, and predict the desired output. The cost function can be used to determine the performance of the machine learning algorithm, which is dependent on the amount of data. We can save both time and money with the help of machine learning.

- Machine learning's importance can be simply appreciated by looking at its applications. Machine learning is being employed in self-driving cars, cyber fraud detection, facial recognition, and Facebook friend suggestion, among other things. Various prominent corporations, such as Netflix and Amazon, have developed machine learning models that monitor customer interest and recommend products based on that information.

Importance Machine Learning:

- The following are some main points that show Machine Learning's importance:
  - Rapid increment in the production of data.
  - Complicated problems those are tough for a human to solve.
  - Making decisions in a variety of fields including finance.
  - Extracting meaningful information from data and uncovering hidden patterns.

6.2 TYPES OF MACHINE LEARNING

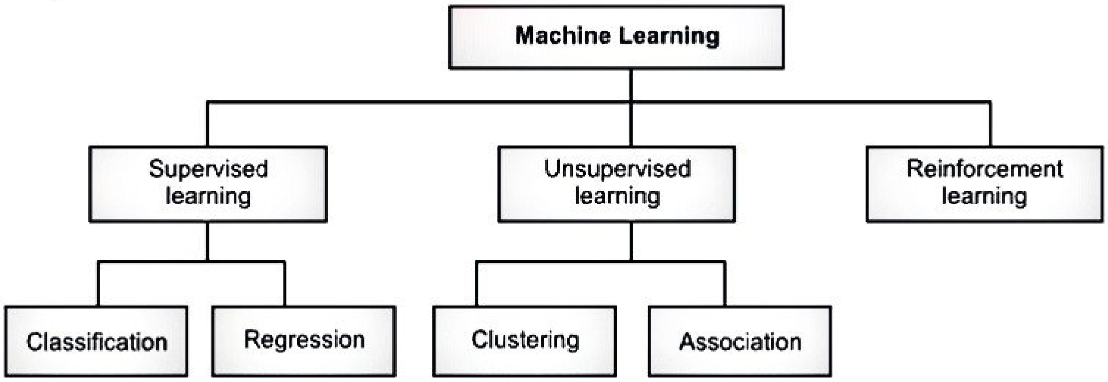

Fig. 6.2: Types of Machine Learning

- Machine learning implementations are divided into three groups based on the type of learning "signal" or "response" provided to a learning system, as follows:

1. **Supervised learning:**
   - Supervised learning, also known as supervised machine learning. It is defined as when an algorithm learns from example data and associated target responses, which might be numeric values or string labels such as classes or tags, in order to predict the correct response when presented with new instances.
   - Example: This method is comparable to human learning under the guidance of a teacher. The teacher gives good examples for the student to memorize and then the student uses these specific instances to infer general norm.
   - Another example is text classification problems. In this set of problems, the goal is to predict the class label of a given piece of text. One particularly popular topic in text classification is to predict the sentiment of a piece of text, like a tweet or a product review.

2. **Unsupervised learning:**
   - Unsupervised learning occurs when an algorithm learns from simple samples with no accompanying response, allowing the algorithm to discover data patterns on its own. This sort of technique restructures the data into new features that may indicate a class or a new set of uncorrelated values. They are quite valuable in supplying people with fresh useful inputs to supervised machine learning algorithms as well as insights into the meaning of data.
   - It parallels the methods that humans use to determine that certain things or events belong to the same class, such as observing the degree of resemblance between objects, as a type of learning.

   Examples:
   - **Recommender systems:** This type of learning is used in several recommendation systems seen on the internet in the form of marketing automation.
   - **Customer segmentation:** Understanding different customer groups around which to build marketing or other business strategies.
   - **Genetics:** Clustering DNA patterns to analyse evolutionary biology.
3. **Reinforcement learning:**
   - Reinforcement learning (RL) is the training of machine learning models to make a sequence of decisions. It occurs when you present the algorithm with examples that don't have labels. However, you can give positive or negative feedback to an example depending on the solution the algorithm proposes. This falls under the category of Reinforcement learning, which is linked to applications in which the algorithm must make decisions (rather than just being descriptive, as in unsupervised learning), and the decisions have consequences. It's similar to learning by trial and error in the human world.
   - Errors aid learning since they come with a cost (money, effort, regret, sorrow, and so on) teaching you that some actions are less likely to succeed than others.

Components of Reinforcement Learning:

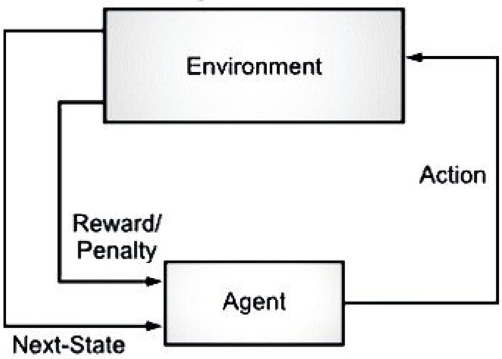

Fig. 6.3: Components of Reinforcement Learning

Block Diagram of Reinforcement Learning:

1. **States:** The observation, the agent does on the environment after performing an action.
2. **Action:** An action that the agent performs on the environment based on its observation.
3. **Reward/Penalty:** The feedback the agent receives based on the action it performed. If the feedback is positive, it receives a reward and if the feedback is negative, it receives a punishment/Penalty.

- There is an agent and an environment. The environment gives the agent a state. The agent chooses an action and receives a reward from the environment along with the new state. This learning process continues until the goal is achieved or some other condition is met.

  Examples:

  - In gaming when computers learn to play video games on their own, an application provides the algorithm with scenarios such as the gamer being trapped in a maze while avoiding an enemy. The application lets the algorithm about the outcomes of its activities. The algorithm learns while attempting to avoid and survive what it discovers to be dangerous.
  - You can see how the Google DeepMind business produced a reinforcement learning program that plays old Atari video games. When you watch the video, note how the program starts out slow and unskilled but improves with training over time until it becomes a champion.
  - In industry reinforcement, learning-based robots are used to perform various tasks. Apart from the fact that these robots are more efficient than human beings, they can also perform tasks that would be dangerous for people.
  - In NLP(Natural Language Processing), RL can be used in text summarization, question answering, and machine translation.

4. **Semi-supervised learning:**
   - Semi-supervised learning occurs when a training set contains some (typically many) of the desired outputs but no complete training signal. This theory has a specific situation known as Transduction, in which the whole collection of problem cases is known at learning time except for a portion of the targets.
   - Another categorization of machine learning tasks arises when one considers the desired output of a machine-learned system:
     (a) Classification: When inputs are divided into two or more classes the learner must produce a model that assigns unseen inputs to one or more (multi-label classification) of these classes. This is typically tackled in a supervised way. Spam filtering is an example of classification, where the inputs are email (or other) messages and the classes are "spam" and "not spam".
     (b) Regression: This is also a supervised problem. It is a case when the outputs are continuous rather than discrete.

(c) **Clustering:** When a set of inputs is to be divided into groups unlike in classification, the groups are not known beforehand, making this typically an unsupervised task.

- AI can be described as an area of computer science that simulates human intelligence in machines. It's about smart algorithms making decisions based on the available data.
- AI and ML are closely related, but these terms aren't interchangeable. ML actually falls under the umbrella of AI. It demands that machines carry out tasks in the same way that humans do.

6.3 TRENDS OF AI TECHNOLOGIES

Following are some trends of AI technologies:

- **AI Customer Support and Assistance:** Each business should attempt to offer an enjoyable customer experience. Satisfying existing customers helps businesses market new products and services. AI enables firms to improve their customer service by offering better response time and interaction. Artificial system assistance includes sales tasks and customer services.
- **Data Access Enabling Ubiquity:** Reliable and accurate information helps businesses shift to AI-powered automated decision making. It has cut operational cost, streamlined processes, and improved the research capabilities of many organizations.
- **Predictive Analytics:** Predictive analytics will help businesses use real data to prepare for outcomes and behaviors thus being more proactive.
- **Enhanced Customization:** Businesses need to offer relevant and personalized services to remain relevant and widen their client base.
- **Real-Time Marketing Activities:** Instant data on current marketing decisions is part of real-time marketing. It depends on relevant trends and customer feedback to prepare strategies.
- **AI-Powered Chatbots:** Many businesses use chatbots to market products and make payments. They are efficient in offering exemplary customer service.

6.4 PREDICTIVE ANALYTICS (WEATHER FORCASTING)

- **Predictive analytics** uses historical data to predict future events. Typically, historical **data** is used to build a mathematical model that captures important trends. That predictive model is then used on current data to predict what will happen next, or to suggest actions to take for optimal outcomes.
- In other words, **Predictive analytics** refers to using historical data, machine learning, and artificial intelligence to predict what will happen in the future. This historical data is fed into a mathematical model that considers key trends and patterns in the data. The model is then applied to current data to predict what will happen next.

Basic tips for writing Case study:

1. Read and examine the case thoroughly. Take notes, highlight relevant facts, underline key problems.
2. Focus your Analysis. Identify two to five key problems.
3. Uncover possible solutions/changes needed.
4. Select the best Solution.

Case Study: Predictive Analytics and Data Science to keep an eye on the Weather

- Weather companies want to do their best so that consumers and businesses can have reliable forecasts – not just for the day but in advance of it. Accuracy matters to weather services such as the Weather Channel and AccuWeather and predictive analytics, along with Data Science, is allowing them to get more precise weather reports than ever before. We all like to discuss what is going on outside, and we want to know if enough snow is expected for good skiing on an upcoming vacation, whether it will be pouring on a special occasion, or if the expectation of heavy storms with high winds requires a business to activate disaster recovery plans.
- In fact, a multi-year study of forecast accuracy – found that AccuWeather was the most accurate provider for wind and precipitation forecasts and was a co-leader in accuracy for temperature forecasts. For temperature forecasts, AccuWeather was the most accurate for high temperature forecasts, while The Weather Channel was the most accurate for low temperature forecasts.
- AccuWeather's clients include WABC in New York City, which has been using the weather service for nearly fifty years. Commenting on the report, AccuWeather founder, president and chairman Joel N. Myers noted that AccuWeather forecasts are much more localised to individual addresses and that it provides highly detailed forecasts both in location and time. "AccuWeather's forecasts also extend farther into the future than any other source, with 90-day forecasts available for all locations on the planet," he said.
- As you might expect, data and data analytics rule in generating weather forecast success. The study mentioned above was conducted by ForecastWatch.com, a service of Intellovations, LLC. ForecastWatch defines itself as "the meteorologists' source" for unbiased data and analysis to improve forecast quality. The service collects weather forecasts and verifies them against actual observations for 86 countries, compiles forecasts and information from 1,350 locations, and stores 800 million historical forecasts in its proprietary database (so far). Weather companies and other businesses can use this for accurate analysis based on business parameters, or they can gain direct access to the database to create customizable data sets.
- Between an organization's own data and that supplied by ForecastWatch, weather services and other clients have a wealth of data on hand to measure forecast accuracy

and quality. The data on hand serves as the base for conducting analytics for a number of use cases, including making strategic decisions such as which weather tools to invest in, says Eric Floehr, founder and CEO of ForecastWatch.

- "Whenever an analyst or data scientist creates a model, they feed raw data into a machine learning system like a neural network or some linear regression system," Floehr says. "They're trying to take all the input data and find the best way to combine it all to generate predictions that will be as close as possible to what actually happens."
- Weather companies and other businesses can layer ForecastWatch's data on top of their own in its cloud and run their models.

Who Benefits from Weather Watches?

- Weather companies were the first industry to use ForecastWatch's solution to measure forecast accuracy and quality. But its base of customers has been expanding to businesses in any field that needs highly reliable data for weather forecast predictive analytics.
- In one example provided by Floehr, a large transportation company has improved its decision matrix by leveraging smart weather data for accurate analysis of upcoming weather events. "It has to do certain things to ensure the safety of their transports based on different weather conditions," he says.
- Another interesting example has to do with how sports arenas and team facility owners can take advantage of marrying their own data with ForecastWatch's historical weather data to help build better business models. Gaining a better understanding of how weather likely will affect fan attendance – based on ForecastWatch's historical weather data combined with their own attendance models – gives them the tools they need to predict and make decisions around how the arena should operate on certain days.
- Previously, attendance models may have included data about a team's record or the day of week or the time of a game, but without taking historical weather data into account, accuracy was hampered. Historical data about weather conditions can provide additional perspective on whether attendance will be strong for a certain game.
- "Consumers are making decisions based not just on the weather at the time of the event but even days before that," Floehr says. "With our almost billion historical weather forecasts, we can supply data in an easily used form for machine learning and AI so that [clients] can train their models effectively."
- Then, they have more insight into attendance expectations and are able to plan in advance everything from how many ushers they'll need for a game to how many hot dogs to order from a supplier, he says.
- A national grid energy utility that ForecastWatch works with uses its historical weather information as part of its data sets to create models to handle people and

resource placement when there is going to be a potentially impactful event – flooding followed by severe winds, for instance. "When the ground is saturated, it's a lot easier for trees to topple over, and with a high wind event after that, trees have the potential to bring down power lines," Floehr explains. Now the utility can be more accurate in preparing responses to impactful weather events.

Python's Place in All This

- "The ForecastWatch service is 99 percent Python and open source", Floehr says. Python is pretty much the preeminent language of meteorology as an upward level language of manipulating and dealing with the output of models, he explains. At American Meteorological Society conferences, attendees can take short Python courses.
- "Python is used extensively in the meteorology community as it will handle large amounts of data, analyze it and basically manipulate that data to meet needs," he says.
- When ForecastWatch started up in the early 2000s, a lot of observation stations for weather were still using modems and communicating their results at the end of the day. But now, not only is there so much more data, it's coming at everyone faster, too.
- "It's almost instantaneous. You get data and more of it in real time, and that has enabled lot of amazing things," Floehr says – including, of course, the company's services. But there's still room for many businesses to understand that historical weather data can help them with their tactical and strategic decision-making. He says "Data is the new currency. More and more of it exists and so more and more decisions can be made using it".

(***Note:** Case study taken from https://www.dataversity.net/)

6.5 AI-POWERED CHATBOTS (SBI CARD CHATBOT (ILA))

- An AI chatbot is a program within a website or app that simulates human conversations using NLP (natural language processing). A **chatbot** is programmed to work independently from a human operator. It can answer questions formulated to it in natural language and respond like a real person. It provides responses based on a combination of predefined scripts and machine learning applications.
- SBI Cards' ILA is a one-of-its-kind solution that is helping the company drive revenue growth.

Case Study: How is India's 2nd Largest Credit Card Company using a Chatbot to drive Revenue Growth?

- In 2018, SBI Cards, one of India's largest credit card companies, decided to collaborate with Senseforth.ai to build ILA, an AI-powered agent, which not only addresses customer queries but helps consumers discover the right product and make informed decisions.

Features:

1. Trained to handle over 200 types of questions and their variations.
2. Handles 42 types of transactions.
3. Addresses over 170,000 user queries each day from ~8,000 unique users.
4. Accuracy of over 97 percent.
5. Integrated with multiple CRMs and databases.

Available on:

Website | Mobile App

Language Support:

English

Summary:

- Since its launch in 2018, SBI Cards has been using ILA as first-line customer support and has addressed over 23 million queries till date. In addition to this over 25,000 service requests have been booked through ILA. Adoption of this Conversational AI solution has also helped the credit card company acquire over 130,000 leads so far, resulting in top-line revenue growth of thousands of dollars.
- In the future, ILA will support more Indian languages and perform more tasks than it currently handles.

About SBI Card:

- SBI Cards & Payment Services Ltd. is a payment solutions provider and the second largest credit card issuer in India. SBI Card was launched in October 1998 by the State Bank of India, India's largest bank, and GE Capital. SBI Card is headquartered in Gurgaon, India, and has branches in over 100 cities across the country.

  (***Note:** Case study taken from https://www.senseforth.ai/)

Check Your Understanding

1. What is Machine learning?
   (a) The autonomous acquisition of knowledge through the use of computer programs.
   (b) The autonomous acquisition of knowledge through the use of manual programs.
   (c) The selective acquisition of knowledge through the use of computer programs.
   (d) The selective acquisition of knowledge through the use of manual program.
2. Which of the following is not a type of data?
   (a) Ordinal (b) Cardinal
   (c) Nominal (d) Categorical
3. Automated vehicle is an example of ______.
   (a) Unsupervised learning (b) Supervised learning
   (c) Active learning (d) Reinforcement learning

4. In which of the following learning, the teacher returns reward and punishment to learner?
   (a) Unsupervised learning
   (b) Supervised learning
   (c) Active learning
   (d) Reinforcement learning
5. Targeted marketing, Recommended Systems, and Customer Segmentation are applications in which of the following?
   (a) Supervised Learning: Classification
   (b) Unsupervised Learning: Clustering
   (c) Unsupervised Learning: Regression
   (d) Reinforcement learning
6. Which of the following is NOT supervised learning?
   (a) Supervised Learning
   (b) Unsupervised Learning
   (c) Semi-supervised Learning
   (d) Half-Supervised Learning
7. ______ machine learning deals with unlabeled data to find patterns from data.
   (a) Supervised
   (b) Unsupervised
   (c) Semi-supervised
   (d) Reinforcement
8. ______ means learning process is controlled by supervisor or teacher.
   (a) Supervised learning
   (b) Unsupervised learning
   (c) Semi-supervised learning
   (d) Reinforcement learning
9. Which of the following is a type of unsupervised learning?
   (a) Classification
   (b) Regression
   (c) Association Rule
   (d) Decision Tree
10. Machine learning is the field of AI which consist of Learning algorithms that ______.
   (a) At executing some task
   (b) Overtime with experience
   (c) Improve their performance
   (d) All of the above

Answers

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

Practice Questions

Q.I Answer the following questions in short.

1. What is AI chatbot?
2. What is the need of machine learning?
3. What is predictive analysis?
4. Write example of unsupervised learning.
5. List the application of machine learning.

Q.II Answer the following questions.

1. What is Machine Learning? Explains its type in brief.
2. What is Supervised and Unsupervised Learning? Explain with examples.
3. Explain Reinforcement learning.
4. Write a case study on use of the chatbot in the industry?
5. Differentiate between Supervised, Unsupervised and Reinforcement learning.
6. What is Predictive analytics? Explain with suitable case study.

Q.III Define the terms.

1. Regression
2. Classification
3. Clustering
4. Chatbot

■■■

Bibliography

Websites:

1. https://www.google.co.in/books/edition/Artificial_Intelligence/DDNHzcN6jasC?hl=en&gbpv=1&dq=types%20of%20control%20strategies%20in%20ai&pg=PA31&printsec=frontcover&bsq=types%20of%20control%20strategies%20in%20a
2. http://artificialintelligence-notes.blogspot.com/2010/07/problem-characteristics-heuristic.html
3. https://www.javatpoint.com/search-algorithms-in-ai
4. https://www.cet.edu.in/noticefiles/271_AI%20Lect%20Notes.pdf]
5. https://link.springer.com/content/pdf/10.1007%2F978-3-642-21004-4_6.pdf
6. https://www.javatpoint.com/propositional-logic-in-artificial-intelligence#:~:text=Propositional%20logic%20(PL)%20is%20the,in%20logical%20and%20mathematical%20form.
7. https://users.cs.cf.ac.uk/Dave.Marshall/AI2/node68.html
8. https://www.amirajcollege.in/wp-content/uploads/2020/06/2180703-chapter-8-weak-slot-and-filler-structure.pdf
9. http://www.cs.nott.ac.uk/~pszrq/files/5FAIKR.pdf
10. https://www.tutorialspoint.com/machine_learning_with_python/machine_learning_with_python_types_of_learning.htm
11. https://www.google.com/search?rlz=1C1CHBD_enIN904IN905&sxsrf=ALeKk00Q2m3rKgkBK5T2RZeGQkEbVbkF4g:1629097978712&q=Types+of+learning+in+AI+javatpoint&sa=X&ved=2ahUKEwj20szy_rTyAhVsyzgGHSAzCw0Q1QIwG3oECC8QAQ
12. https://www.google.com/search?rlz=1C1CHBD_enIN904IN905&sxsrf=ALeKk00Q2m3rKgkBK5T2RZeGQkEbVbkF4g:1629097978712&q=Types+of+Learning+in+AI+geeksforgeeks&sa=X&ved=2ahUKEwj20szy_rTyAhVsyzgGHSAzCw0Q1QIwGXoECDQQAQ
13. www.javatpoint.com

Books:

1. Artificial Intelligence, Tata McGraw Hill, Elaine Rich and Kevin Knight
2. Computational Intelligence, Eberhart, Elsevie
3. Artificial Intelligence: A New Synthesis, Nilsson, Elsevier
4. Introduction to Artificial Intelligence and Expert System, Dan Patterson, Prentice Hall of India Pvt. Ltd
5. Artificial Intelligence: A Modern Approach, Russel & Norvig, Pearson Education
6. Introduction to Machine Learning, Ethem Alpaydin, PHI

For Case Studies

1. https://www.oracle.com/in/chatbots/what-is-a-chatbot/
2. https://www.dataversity.net/case-study-predictive-analytics-and-data-science-keep-an-eye-onthe-weather/
3. https://www.senseforth.ai/conversational-ai-case-studies/SBI-Cards/

■■■