PNS SCHOOL OF ENGINEERING & TECHNOLOGY

Nishamani Vihar, Marshaghai, Kendrapara

LECTURE NOTES

ON

ARTIFICIAL INTELLIGENCE & MACHINE LEARNING

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

6TH SEMESTER

PREPARED BY

MR. BISWARANJAN SWAIN

LECTURER IN COMPUTER SCIENCE & ENGINEERING

Syllabus

Unit 1. Introduction to AI

1.1 Definition of AI, History of AI
1.2 Goals and Applications of AI
1.3 Intelligent agent
1.4 Computer vision
1.5 Natural Language Processing
1.6 Turing test
1.7 Problem solving in Games

Unit 2: Introduction to Search Algorithm

2.1 Search, Search space, Search Tree
2.2 Categories and Types of Search
2.3 Heuristic Algorithm vrs Solution Guaranteed Algorithm
2.4 Local search and Optimal problem(Hill climbing, BFS,A*,AO*)
2.5 Adversarial Search
2.6 AI and Game Playing

Unit 3: Knowledge Representation and Reasoning

3.1What to represent, Knowledge
3.2 Properties of Knowledge Representation System, Approaches
3.3 Knowledge Representation
3.4 Reasoning and Types of reasoning

Unit 4: Machine Learning

4.1 Machine Learning
4.2 Statistical or Unsupervised Learning
4.3 ML Properties
4.4 Reinforcement Learning
4.5 Decision Tree

Unit 5: Pattern Recognition

5.1 Introduction to Pattern recognition
5.2 Design Principles of Pattern recognition system
5.3 Statistical Pattern recognition System
5.4 Machine Perception
5.5 Line Finding and Interception
5.6 Object Identification

Unit 6: Expert System

6.1 Introduction to Expert system
6.2 Basic Architecture
6.3 Type of Problem Solved by Expert system
6.4 Features of an Expert System
6.5 Expert System Architectures
6.6 Expert System Tools
6.7 Existing Expert Systems
6.8 Applications of Expert System Technology

UNIT-1

Introduction to AI

1.1 Definition of AI

Artificial intelligence (AI) refers to computer systems capable of performing complex tasks that historically only a human could do, such as reasoning, making decisions, or solving problems.

Artificial intelligence is the simulation of human intelligence processes by machines, especially computer systems.

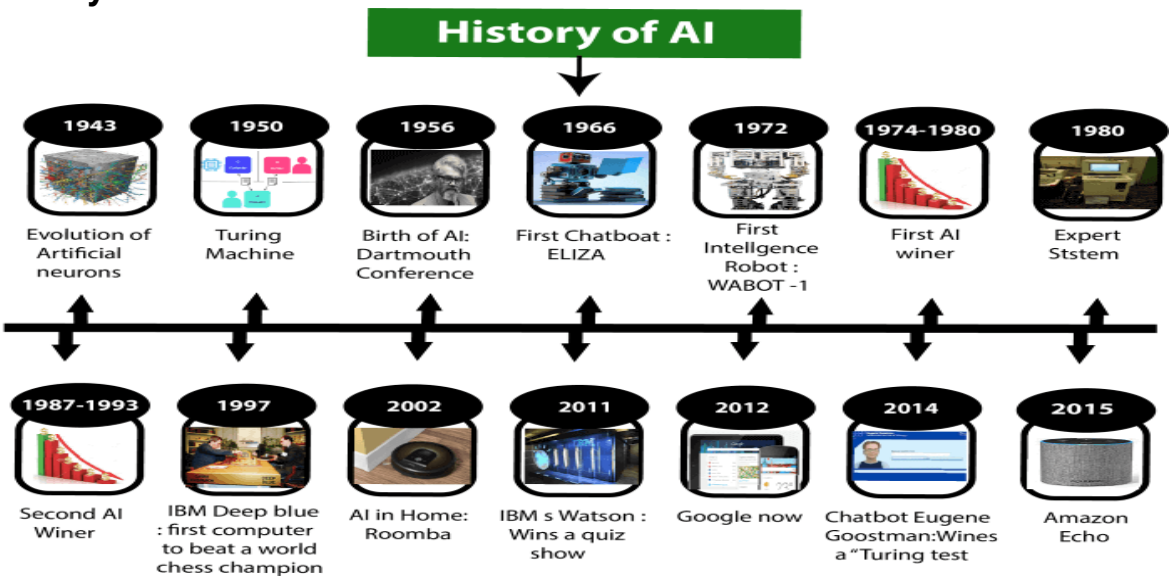
How does AI work?

AI requires a foundation of specialized hardware and software for writing and training machine learning algorithms. Languages used for AI are, Python, R, Java, C++ and Julia etc.

AI systems work by using large amounts of labeled training data, analyzing the data for correlations and patterns, and using these patterns to make predictions about future states. AI programming focuses on cognitive skills that include the following:

- **Learning.** This aspect of AI programming focuses on acquiring data and creating rules for how to turn it into actionable information. The rules, which are called *algorithms*.
- **Reasoning.** It focuses on choosing the right algorithm to reach a desired outcome.
- **Self-correction.** It designed to continually fine-tune algorithms and ensure they provide the most accurate results possible.
- **Creativity.** It uses neural networks, rules-based systems, statistical methods and other AI techniques to generate new images, new text, new music and new ideas.

History of AI



Artificial Intelligence is not a new word and not a new technology for researchers.
Maturation of Artificial Intelligence (1943-1952)

- It all started in the year 1943 by a proposed model of **artificial neurons**.
- In the year 1950, Alan Turing, English mathematician published **"Computing Machinery and Intelligence"** in which he proposed a test. The test can check the machine's ability to exhibit intelligent behaviour equivalent to human intelligence, called a **Turing test**.
- **In the year 1955,** Allen Newell and Herbert A. Simon created the "first artificial intelligence program" which was named as **"Logic Theorist"**. This program had proved 38 of 52 Mathematics theorems, and find new and more proofs for some theorems.
- **Year 1956:** The word "Artificial Intelligence" first adopted by American Computer scientist John McCarthy at the Dartmouth Conference. For the first time, AI coined as an academic field.

The golden years-Early enthusiasm (1956-1974)

- The first chatbot was created in 1966, named as ELIZA.
- The first intelligent humanoid robot was built in the year 1972 in Japan named as WABOT-1.
- In the year 1997, IBM Deep Blue beats world chess champion, Gary Kasparov, and became the first computer to beat a world chess champion.
- **Year 2002:** for the first time, AI entered the home in the form of Roomba, a vacuum cleaner.

Advances in **deep learning**, **natural language processing (NLP)**, and **reinforcement learning** led to AI models like:

- **2014**: GANs (Generative Adversarial Networks) revolutionized image generation.
- **2017**: The **Transformer** model, introduced by Google, became the foundation for modern NLP.
- **2020s**: Large Language Models (LLMs) like **GPT-3, ChatGPT**, and **GPT-4** demonstrated near-human text generation.

1.2 Goals and Applications of AI

The research goal of artificial intelligence is to create technology that allows computers and machines to work intelligently. AI aims to create systems that can **think, learn, and act** like humans—or even beyond human capabilities. Its goals can be categorized into different levels, from narrow applications to long-term advancements.

Narrow AI (Weak AI) – Solving Specific Tasks

☑**Automation** – Reducing human effort by automating repetitive tasks (e.g., chatbots, self-checkout systems).

☑**Decision Support** – Assisting humans in making better decisions (e.g., medical diagnosis, stock market analysis).

☑**Pattern Recognition** – Identifying patterns in data for insights (e.g., fraud detection, recommendation systems).

☑**Natural Language Processing (NLP)** – Enabling machines to understand and generate human language (e.g., ChatGPT, Google Translate).
☑**Computer Vision** – Helping machines interpret and analyze images and videos (e.g., facial recognition, self-driving cars).

Reasoning & Problem Solving – Developing AI that can solve problems across multiple domains like a human.
Common Sense Understanding – Teaching AI to understand the world intuitively and adapt to different contexts.
Creativity & Innovation – Enabling AI to generate new ideas, art, and scientific discoveries.
Self-Learning & Adaptability – Creating AI that continuously learns and improves on its own, like humans do.

3. Super AI – Beyond Human Intelligence (Future Goal)
Self-Awareness & Consciousness – Building AI with self-awareness and emotions (a topic of debate).
Autonomous Decision-Making – Developing AI that can make independent, ethical decisions.
Solving Humanity's Greatest Challenges – AI contributing to curing diseases, preventing climate change, and advancing space exploration.

Application of AI

Artificial Intelligence has various applications in today's society. Following are some sectors which have the application of Artificial Intelligence:

1. AI in Astronomy

- Artificial Intelligence can be very useful to solve complex universe problems. AI technology can be helpful for understanding the universe such as how it works, origin, etc.

2. AI in Healthcare

- Healthcare Industries are applying AI to make a better and faster diagnosis than humans. AI can help doctors with diagnoses and can inform when patients are worsening so that medical help can reach to the patient before hospitalization.

3. AI in Gaming

- AI can be used for gaming purpose. The AI machines can play strategic games like chess, where the machine needs to think of a large number of possible places.

4. AI in Finance

- The finance industry is implementing automation, chatbot, adaptive intelligence, algorithm trading, and machine learning into financial processes.

5. AI in Data Security

- The security of data is crucial for every company and cyber-attacks are growing very rapidly in the digital world. AI can be used to make your data more safe and secure. Some examples such as AEG bot, AI2 Platform,are used to determine software bug and cyber-attacks in a better way.

6. AI in Social Media

- Social Media sites such as Facebook, Twitter, and Snapchat contain billions of user profiles, which need to be stored and managed in a very efficient way. AI can organize and manage massive amounts of data. AI can analyze lots of data to identify the latest trends, hashtag, and requirement of different users.

7. AI in Travel & Transport

- AI is becoming highly demanding for travel industries. AI is capable of doing various travel related works such as from making travel arrangement to suggesting the hotels, flights, and best routes to the customers. Travel industries are using AI-powered chatbots which can make human-like interaction with customers for better and fast response.

8. AI in Automotive Industry

- Some Automotive industries are using AI to provide virtual assistant to their user for better performance. Such as Tesla has introduced TeslaBot, an intelligent virtual assistant.
- Various Industries are currently working for developing self-driven cars which can make your journey more safe and secure.

9. AI in Robotics:

- Artificial Intelligence has a remarkable role in Robotics. Usually, general robots are programmed such that they can perform some repetitive task, but with the help of AI, we can create intelligent robots which can perform tasks with their own experiences without pre-programmed.
- Humanoid Robots are best examples for AI in robotics, recently the intelligent Humanoid robot named as Erica and Sophia has been developed which can talk and behave like humans.

10. AI in Entertainment

- We are currently using some AI based applications in our daily life with some entertainment services such as Netflix or Amazon. With the help of ML/AI algorithms, these services show the recommendations for programs or shows.

11. AI in Agriculture

- Agriculture is an area which requires various resources, labor, money, and time for best result. Now a day's agriculture is becoming digital, and AI is emerging in this field. Agriculture is applying AI as agriculture robotics, solid and crop monitoring, predictive analysis. AI in agriculture can be very helpful for farmers.

12. AI in E-commerce

- AI is providing a competitive edge to the e-commerce industry, and it is becoming more demanding in the e-commerce business. AI is helping shoppers to discover associated products with recommended size, color, or even brand.

13. AI in education:

- AI can automate grading so that the tutor can have more time to teach. AI chatbot can communicate with students as a teaching assistant.
- AI in the future can be work as a personal virtual tutor for students, which will be accessible easily at any time and any place.

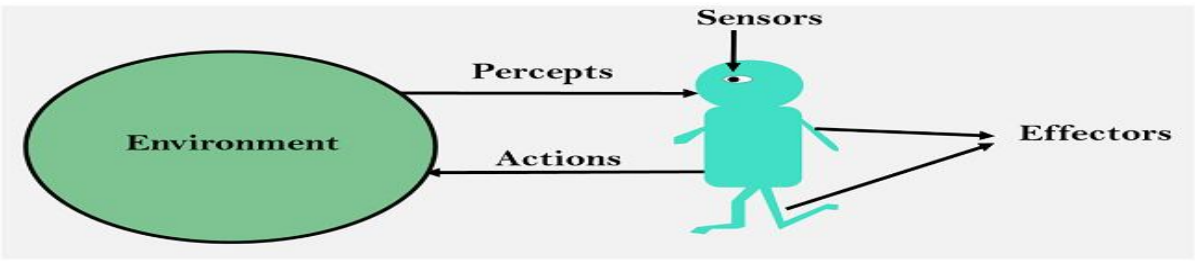
1.3 Intelligent agent

An intelligent agent is an autonomous entity which act upon an environment using sensors, actuators and effectors for achieving goals. An Agent runs in the cycle of **perceiving**, **thinking**, and **acting**.

Sensor: Sensor is a device which detects the change in the environment and sends the information to other electronic devices. An agent observes its environment through sensors.

Actuators: Actuators are the component of machines that converts energy into motion. The actuators are only responsible for moving and controlling a system. An actuator can be an electric motor, gears, rails, etc.

Effectors: Effectors are the devices which affect the environment. Effectors can be legs, wheels, arms, fingers, wings, fins, and display screen.



Following are the main four rules for an AI agent:

- **Rule 1:** An AI agent must have the ability to perceive the environment.
- **Rule 2:** The observation must be used to make decisions.

- **Rule 3:** Decision should result in an action.
- **Rule 4:** The action taken by an AI agent must be a rational action.

PEAS Representation

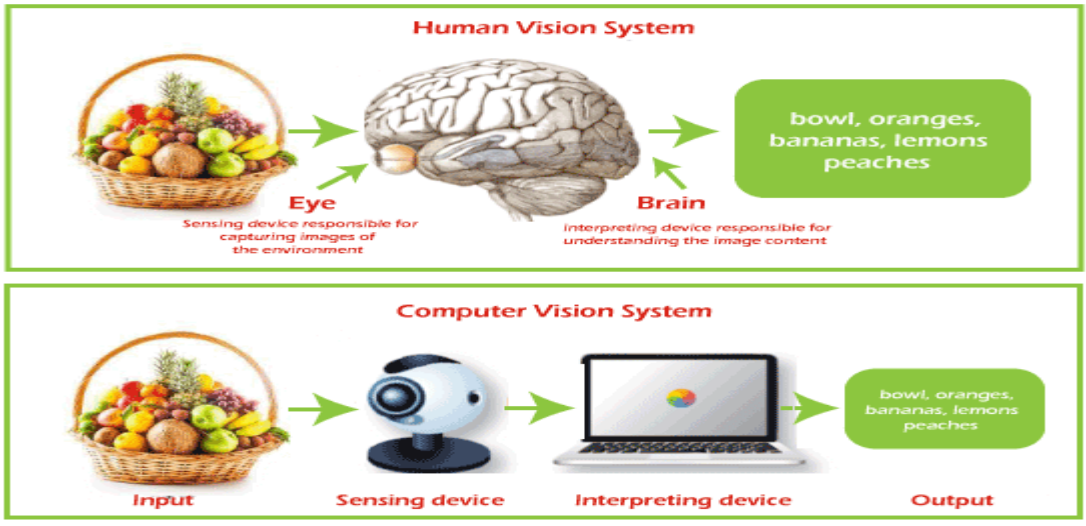
PEAS is a type of model on which an AI agent works upon. It is made up of four words:

- **P:** Performance measure
- **E:** Environment
- **A:** Actuators
- **S:** Sensors

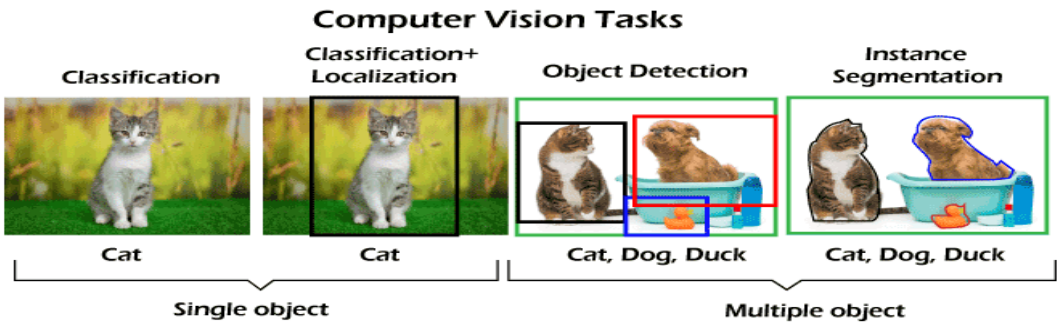
Here **performance measure** is the objective for the success of an agent's behaviour.
Environment is that to which the agent belongs and perceives that environment through the sensors.
Let's suppose a self-driving car then PEAS representation will be:
Performance: Safety, time, legal drive, comfort
Environment: Roads, other vehicles, road signs, pedestrian
Actuators: Steering, accelerator, brake, signal, horn
Sensors: Camera, GPS, speedometer, odometer, accelerometer, sonar.

1.4 Computer vision

Computer vision makes computer systems capable of extracting meaningful information from visual data like videos and images. It also helps to take appropriate actions and make recommendations based on the extracted information.



Firstly, a vast amount of visual labelled data is provided to machines to train it. This labelled data enables the machine to analyze different patterns in all the data points and can relate to those labels. E.g., suppose we provide visual data of millions of dog images. In that case, the computer learns from this data, analyzes each photo, shape, the distance between each shape, color, etc., and hence identifies patterns similar to dogs and generates a model. As a result, this computer vision model can now accurately detect whether the image contains a dog or not for each input image.



Tasks associated with Computer Vision are

- **Object classification: Object classification is a** computer vision technique/task used to classify an image, such as whether an image contains a dog, a person's face, or a banana. The class of an object can be accurately predicted, present in an image with image classification.
- **Object Identification/detection:** It uses image classification to identify and locate the objects in an image or video. With such detection and identification technique, the system can count objects in a given image or scene and determine their accurate location and labelling.
  For example, in a given image, one dog, one cat, and one duck can be easily detected and classified using the object detection technique.
- **Object Verification:** The system processes videos, finds the objects based on search criteria, and tracks their movement.
- **Object Landmark Detection:** The system defines the key points for the given object in the image data.
- **Image Segmentation:** Image segmentation not only detects the classes in an image as image classification; instead, it classifies each pixel of an image to specify what objects it has. It tries to determine the role of each pixel in the image.
- **Object Recognition:** In this, the system recognizes the object's location with respect to the image.

1.5 Natural Language Processing

It is a field of Artificial Intelligence (AI) and Computer Science that is concerned with the interactions between computers and humans in natural language. The goal of NLP is to develop algorithms and models that enable computers to understand, interpret, generate, and manipulate human languages.

Common Natural Language Processing (NLP) Task:

- **Text and speech processing:** This includes Speech recognition, text-&-speech processing, encoding(i.e converting speech or text to machine-readable language), etc.
- **Text classification:** This includes Sentiment Analysis in which the machine can analyze the qualities, emotions, and sarcasm from text and also classify it accordingly.
- **Language generation:** This includes tasks such as machine translation, summary writing, essay writing, etc. which aim to produce coherent and fluent text.

- **Language interaction:** This includes tasks such as dialogue systems, voice assistants, and chatbots, which aim to enable natural communication between humans and computers.

Components of NLP

There are two components of NLP as given −

Natural Language Understanding (NLU)

Understanding involves the following tasks −

- Mapping the given input in natural language into useful representations.
- Analyzing different aspects of the language.

Natural Language Generation (NLG)

It is the process of producing meaningful phrases and sentences in the form of natural language from some internal representation.

The NLU is harder than NLG.

Difficulties in NLU

NL has an extremely rich form and structure.

It is very ambiguous. There can be different levels of ambiguity −

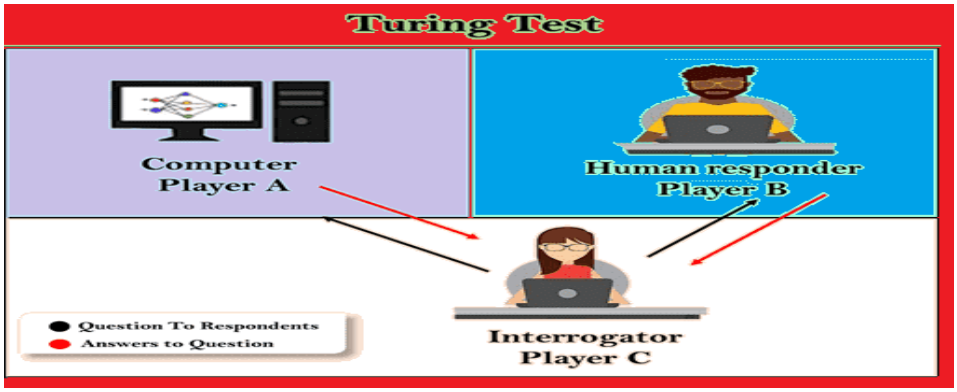
- **Lexical ambiguity** − It is at very primitive level such as word-level.
- For example, treating the word “board” as noun or verb?
- **Syntax Level ambiguity** − A sentence can be parsed in different ways.
- For example, “He lifted the beetle with red cap.” − Did he use cap to lift the beetle or he lifted a beetle that had red cap?
- **Referential ambiguity** − Referring to something using pronouns. For example, Rima went to Gauri. She said, “I am tired.” − Exactly who is tired?
- One input can mean different meanings.
- Many inputs can mean the same thing.

1.6 Turing test

Turing Test was introduced by Alan Turing in his 1950 paper, "Computing Machinery and Intelligence," which considered the question, "Can Machine think?" It is used to check whether a machine can think like a human or not, this test is known as the Turing Test. In this test, Turing proposed that the computer can be said to be an intelligent if it can mimic human response under specific conditions.

The basic idea of the Turing Test is simple: a human judge engages in a text-based conversation with both a human and a machine, and then decides which of the two they believe to be a human. If the judge is unable to distinguish between the human and the machine based on the conversation, then the machine is said to have passed the Turing Test.

The Turing Test is widely used as a benchmark for evaluating the progress of artificial intelligence research, and has inspired numerous studies and experiments aimed at developing machines that can pass the test.



There are three players in this game: one is a computer, another is a human responder, and the third is a human interrogator who is separated from the other two players and whose task is to determine which of the two is a machine.

Because all players communicate via keyboard and screen, the outcome is unaffected by the machine's capacity to transform words into speech.

The exam result is determined not by the number of correct answers, but by how closely the responses resemble those of a human. The computer is allowed to do whatever it can to force the interrogator to make a false identification.

The questions and answers can be like:

Interrogator: Are you a computer?

Player A (Computer): No

Interrogator: Multiply two large numbers such as (256896489*456725896)

Player A: Long pause and give the wrong answer.

If an interrogator is unable to distinguish between a machine and a human in this game, the computer passes the test, and the machine is said to be intelligent and capable of thinking like a human.

1.7 Problem solving in Games

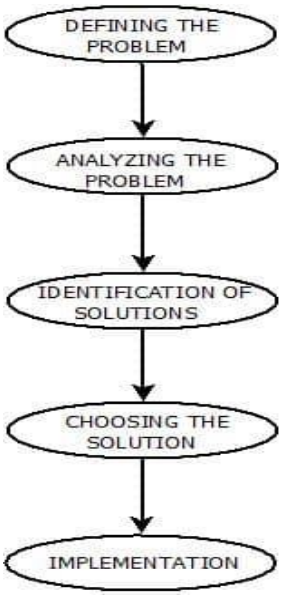
Problem solving can be done by building an artificially intelligent system to solve that particular problem. First the problem statement is defined and then the solution is generated by keeping the conditions in mind.

Some of the most popularly used problem solving with the help of artificial intelligence are:

1. Chess.
2. Sudoku
3. Travelling Salesman Problem.
4. Tower of Hanoi Problem.
5. Water-Jug Problem.
6. N-Queen Problem.

Steps : Solve Problem Using Artificial Intelligence

- The process of solving a problem consists of five steps. These are:



Problem Solving in Artificial Intelligence

1. Defining The Problem: The definition of the problem must be included precisely. It should contain the possible initial as well as final situations which should result in acceptable solution.
2. Analyzing The Problem: Analyzing the problem and its requirement must be done as few features can have immense impact on the resulting solution.
3. Identification Of Solutions: This phase generates reasonable amount of solutions to the given problem in a particular range.
4. Choosing a Solution: From all the identified solutions, the best solution is chosen basis on the results produced by respective solutions.

Implementation: After choosing the best solution, its implementation is done.

UNIT-2

Introduction to Search Algorithms

2.1 Search, Search space, Search Tree

AI uses search strategies or algorithms to solve a specific problem and provide the best result. There are various problem-solving search algorithms. Some single-player games such as tile games, Sudoku, crossword, etc.

Search Algorithm Terminologies:

- **Search:** Searching is a step by step procedure of locating a solution to a problem by systematically looking at nodes in a search tree or search space until a goal node is found.
- **Search Space:** Search space represents a set of possible solutions, which a system may have.
- **Start State:** It is a state from where agent begins **the search**.
- **Search tree:** A tree representation of search problem is called Search tree. The root of the search corresponds to the initial state.

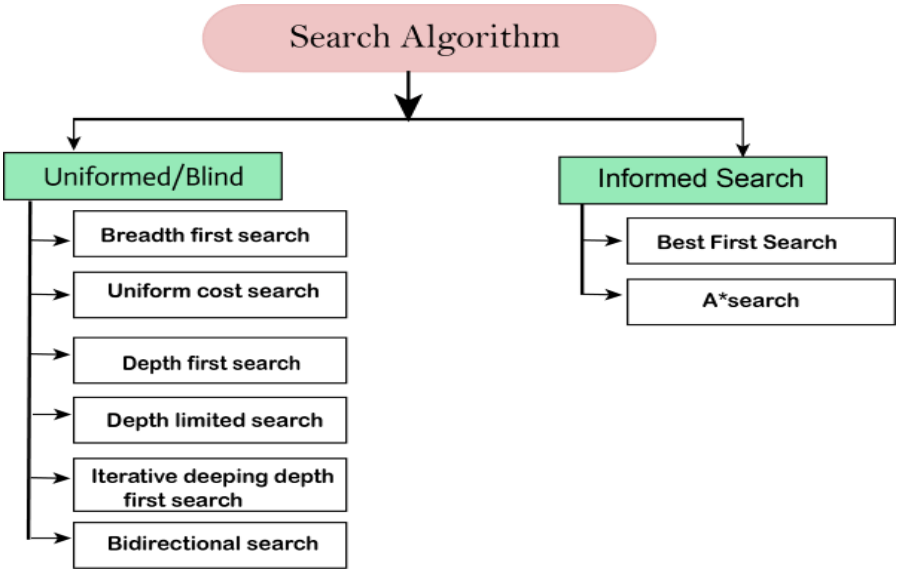
Properties of Search Algorithms:

Following are the four essential properties of search algorithms to compare the efficiency of these algorithms:

- **Completeness:** A search algorithm is said to be complete if it guarantees to return a solution if at least any solution exists for any random input.
- **Optimality:** If a solution found for an algorithm is guaranteed to be the best solution (lowest path cost) among all other solutions, then such a solution for is said to be an optimal solution.
- **Time Complexity:** Time complexity is a measure of time for an algorithm to complete its task.
- **Space Complexity:** It is the maximum storage space required at any point during the search, as the complexity of the problem.

2.2 Categories and Types of Search

Based on the search problems we can classify the search algorithms into uninformed (Blind search) search and informed search (Heuristic search) algorithms.



Uninformed/Blind Search:

The uninformed search does not contain any domain knowledge such as closeness, the location of the goal. It operates in a brute-force way as it only includes information about how to traverse the tree and how to identify leaf and goal nodes. It examines each node of the tree until it achieves the goal node.

It can be divided into 6 main types:

1. Breadth-first Search
2. Depth-first Search
3. Depth-limited Search
4. Iterative deepening depth-first search
5. Uniform cost search
6. Bidirectional Search

1. Breadth-first Search:

- Breadth-first search is the most common search strategy for traversing a tree or graph. This algorithm searches breadth wise in a tree or graph, so it is called breadth-first search.
- BFS algorithm starts searching from the root node of the tree and expands all successor node at the current level before moving to nodes of next level.
- The breadth-first search algorithm is an example of a general-graph search algorithm.
- Breadth-first search implemented using FIFO queue data structure.

Advantages:

- BFS will provide a solution if any solution exists.
- If there are more than one solutions for a given problem, then BFS will provide the minimal solution which requires the least number of steps.

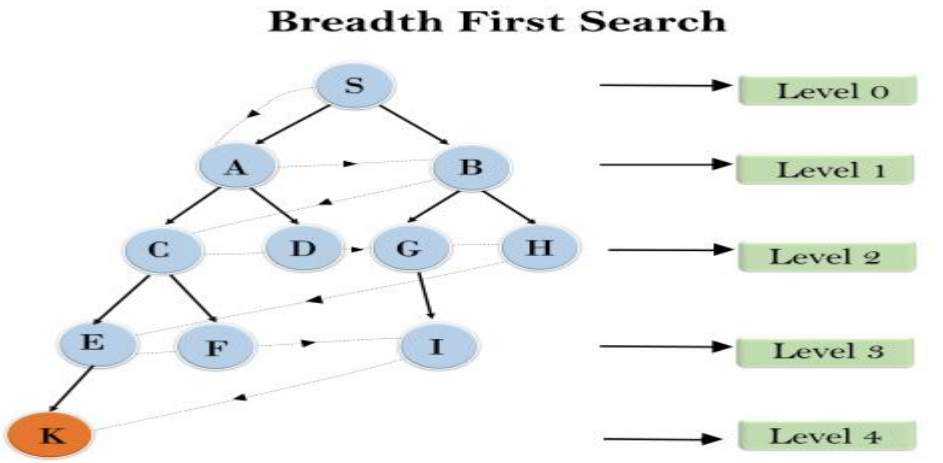
Disadvantages:

- It requires lots of memory since each level of the tree must be saved into memory to expand the next level.
- BFS needs lots of time if the solution is far away from the root node.

Example:

In the below tree structure, we have shown the traversing of the tree using BFS algorithm from the root node S to goal node K. BFS search algorithm traverse in layers, so it will follow the path which is shown by the dotted arrow, and the traversed path will be:

1. S---> A--->B---->C--->D---->G--->H--->E---->F---->I---->K



Time Complexity: Time Complexity of BFS algorithm can be obtained by the number of nodes traversed in BFS until the shallowest Node. Where the d= depth of shallowest solution and b is a node at every state.

$T(b) = 1+b^2+b^3+.....+ b^d= O(b^d)$

Space Complexity: Space complexity of BFS algorithm is given by the Memory size of frontier which is $O(b^d)$.

Completeness: BFS is complete, which means if the shallowest goal node is at some finite depth, then BFS will find a solution.

Optimality: BFS is optimal if path cost is a non-decreasing function of the depth of the node.

2. Depth-first Search

- Depth-first search is a recursive algorithm for traversing a tree or graph data structure.
- It is called the depth-first search because it starts from the root node and follows each path to its greatest depth node before moving to the next path.
- DFS uses a stack data structure for its implementation.
- The process of the DFS algorithm is similar to the BFS algorithm.

Note: Backtracking is an algorithm technique for finding all possible solutions using recursion.

Advantage:

- DFS requires very less memory as it only needs to store a stack of the nodes on the path from root node to the current node.

- It takes less time to reach to the goal node than BFS algorithm (if it traverses in the right path).

Disadvantage:

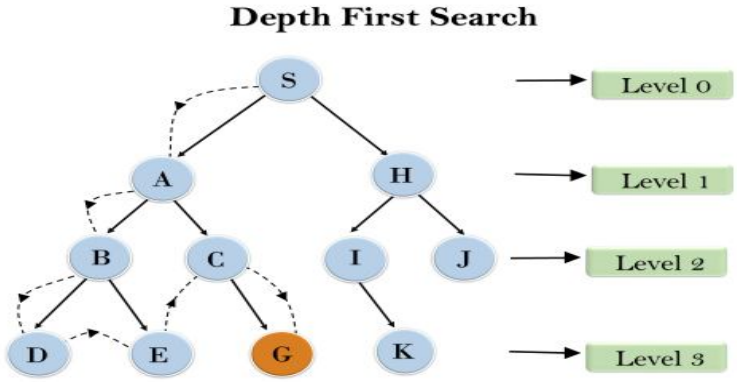
- There is the possibility that many states keep re-occurring, and there is no guarantee of finding the solution.
- DFS algorithm goes for deep down searching and sometime it may go to the infinite loop.

Example:

In the below search tree, we have shown the flow of depth-first search, and it will follow the order as:

Root node--->Left node ----> right node.

It will start searching from root node S, and traverse A, then B, then D and E, after traversing E, it will backtrack the tree as E has no other successor and still goal node is not found. After backtracking it will traverse node C and then G, and here it will terminate as it found goal node.



Completeness: DFS search algorithm is complete within finite state space as it will expand every node within a limited search tree.

Time Complexity: Time complexity of DFS will be equivalent to the node traversed by the algorithm. It is given by:

$T(n)= 1+ n^2+ n^3 +.....+ n^m=O(n^m)$

Where, m= maximum depth of any node and this can be much larger than d (Shallowest solution depth)

Space Complexity: DFS algorithm needs to store only single path from the root node, hence space complexity of DFS is equivalent to the size of the fringe set, which is **O(bm)**.

Optimal: DFS search algorithm is non-optimal, as it may generate a large number of steps or high cost to reach to the goal node.

2.3 Heuristic Algorithm vs Solution Guaranteed Algorithm

The informed search algorithm is more useful for large search space. Informed search algorithm uses the idea of heuristic, so it is also called Heuristic search.

Heuristics function: Heuristic is a function which is used in Informed Search, and it finds the most promising path. It takes the current state of the agent as its input and produces the estimation of how close agent is from the goal. The heuristic method, however, might not always give the best solution, but it guaranteed to find a good solution in reasonable time. Heuristic function estimates how close a state is to the goal. It is represented by h(n), and it calculates the cost of an optimal path between the pair of states. The value of the heuristic function is always positive.

Admissibility of the heuristic function is given as:

1. h(n) <= h*(n)

Here h(n) is heuristic cost, and h*(n) is the estimated cost. Hence heuristic cost should be less than or equal to the estimated cost.

Pure Heuristic Search:

Pure heuristic search is the simplest form of heuristic search algorithms. It expands nodes based on their heuristic value h(n). It maintains two lists, OPEN and CLOSED list. In the CLOSED list, it places those nodes which have already expanded and in the OPEN list, it places nodes which have yet not been expanded.

On each iteration, each node n with the lowest heuristic value is expanded and generates all its successors and n is placed to the closed list. The algorithm continues unit a goal state is found.

In the informed search we will discuss two main algorithms which are given below:

- **Best First Search Algorithm(Greedy search)**
- **A* Search Algorithm**

1.) Best-first Search Algorithm (Greedy Search):

Greedy best-first search algorithm always selects the path which appears best at that moment. It is the combination of depth-first search and breadth-first search algorithms. It uses the heuristic function and search. Best-first search allows us to take the advantages of both algorithms. With the help of best-first search, at each step, we can choose the most promising node. In the best first search algorithm, we expand the node which is closest to the goal node and the closest cost is estimated by heuristic function, i.e.

1. f(n)= g(n).

Were, h(n)= estimated cost from node n to the goal.

The greedy best first algorithm is implemented by the priority queue.

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 goal node, then return success and terminate the search, else proceed to Step 6.
- **Step 6:** For each successor node, algorithm checks for evaluation function f(n), and then check if the node has been in either OPEN or CLOSED list. If the node has not been in both list, then add it to the OPEN list.
- **Step 7:** Return to Step 2.

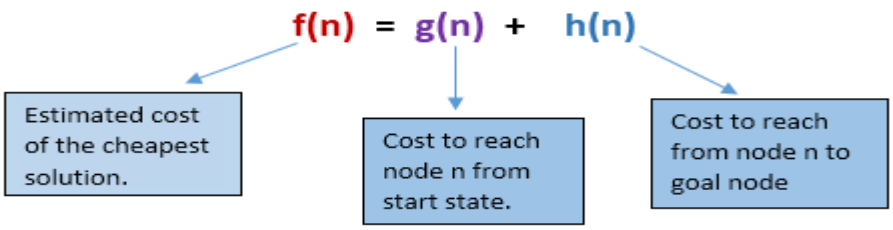
Advantages:

- Best first search can switch between BFS and DFS by gaining the advantages of both the algorithms.
- This algorithm is more efficient than BFS and DFS algorithms.

Disadvantages:

- It can behave as an unguided depth-first search in the worst case scenario.
- It can get stuck in a loop as DFS.
- This algorithm is not optimal.
- 2.) A* Search Algorithm:
- A* search is the most commonly known form of best-first search. It uses heuristic function h(n), and cost to reach the node n from the start state g(n). It has combined features of UCS and greedy best-first search, by which it solve the problem efficiently. A* search algorithm finds the shortest path through the search space using the heuristic function. This search algorithm expands less search tree and provides optimal result faster. A* algorithm is similar to UCS except that it uses g(n)+h(n) instead of g(n).

- In A* search algorithm, we use search heuristic as well as the cost to reach the node. Hence we can combine both costs as following, and this sum is called as a **fitness number**.



2.4 Local search and Optimal problem (Hill climbing, BFS,A*,AO*)

Local search is a type of Artificial Intelligence (AI) algorithm used to solve optimisation problems. It is also known as simulated annealing or hill-climbing and involves searching for the best solution in each region using greedy search techniques.

Hill Climbing Algorithm

- Hill climbing algorithm is a local search algorithm which continuously moves in the direction of increasing elevation/value to find the peak of the mountain or best solution to the problem. It terminates when it reaches a peak value where no neighbor has a higher value.
- Hill climbing algorithm is a technique which is used for optimizing the mathematical problems. One of the widely discussed examples of Hill climbing algorithm is Traveling-salesman Problem in which we need to minimize the distance traveled by the salesman.
- It is also called greedy local search as it only looks to its good immediate neighbor state and not beyond that.
- A node of hill climbing algorithm has two components which are state and value.
- Hill Climbing is mostly used when a good heuristic is available.
- In this algorithm, we don't need to maintain and handle the search tree or graph as it only keeps a single current state.

Features of Hill Climbing:

Following are some main features of Hill Climbing Algorithm:

- **Generate and Test variant:** Hill Climbing is the variant of Generate and Test method. The Generate and Test method produce feedback which helps to decide which direction to move in the search space.

- **Greedy approach:** Hill-climbing algorithm search moves in the direction which optimizes the cost.
- **No backtracking:** It does not backtrack the search space, as it does not remember the previous states.

Breadth First Search Algorithm

Breadth first search algorithm is an algorithm used to find the shortest path from one node to another in a graph. It begins by searching through the nodes closest to the starting node, and then expands outward to the more distant nodes. The algorithm is often used in tree traversal and network routing algorithms.

Working of Breadth First Search Algorithm

The breadth first search algorithm is a method used to traverse a tree or graph. In this algorithm, the nodes are visited in the order of their distance from the root node. The root node is expanded first, then the immediate child nodes of the root node are expanded, and so on. This process continues until all the nodes in the tree or graph have been visited.

The main advantage of the breadth first search algorithm is that it is simple to implement and can be easily understood. Moreover, this algorithm can be used to find the shortest path between two nodes in a graph.

Advantages of Breadth First Search Algorithm

There are many advantages to using the Breadth First Search algorithm:

- The main advantage is that it is very simple to implement and understand.
- It is guaranteed to find the shortest path from the starting point to the goal.
- Breadth First Search tends to find paths with fewer steps than other algorithms, such as Depth First Search.
- Breadth First Search can be easily parallelized, which means that it can take advantage of multiple processors to speed up the search.

Disadvantages of Breadth First Search Algorithm

There are a few disadvantages to the Breadth First Search algorithm:

- It can be very memory intensive since it needs to keep track of all the nodes in the search tree.

- It can be slow since it expands all the nodes at each level before moving on to the next level.
- It can sometimes find sub-optimal solutions since it doesn't explore all possible paths through the search tree.

Applications of Breadth First Search Algorithm

The breadth first search algorithm is a powerful tool that can be used to solve various problems. In this article, we will explore some of the potential applications of this algorithm.

1. Graph traversal: Breadth first search can be used to traverse a graph. This can be useful for tasks such as finding the shortest path between two nodes or determining if a graph is connected.

2. Network routing: Breadth first search can be used to find the shortest path between two nodes in a network. This can be useful for tasks such as routing traffic or finding the quickest route between two points.

3. Pattern matching: Breadth first search can be used to match patterns in data. This can be useful for tasks such as searching for a specific string in a text file or finding all instances of a particular word in a document.

A* Algorithm

Let's briefly understand what the A* algorithm is:

- A* is a widely used path-finding algorithm in computer science and is particularly useful for applications like route planning in maps, robotics, and games.
- It is an informed search algorithm meaning it uses a heuristic to estimate the cost of reaching the goal from any given node.
- A* algorithm is like a GPS for computers. It is widely used in many areas of computer science, especially in things like video games where characters need to find the best path to a destination.
- A* is efficient and clever because it doesn't waste time exploring unnecessary routes.
- It uses a clever trick called a heuristic which is like a smart guess to estimate how far it is from any point to the goal. This helps it find the quickest path without having to look at every single option.

AO* Algorithm

Let's briefly understand what the AO* algorithm is:

- AO* is a variant of the A* algorithm. It is designed to be more flexible and capable of adapting to changing environments.
- AO* can repair its solution whenever it encounters a change in the environment without having to start the search from scratch.
- AO* is like a supercharged version of A*. It is designed to handle situations where things might change like a dynamic environment.

- For example, imagine a robot moving around a busy room. If furniture gets rearranged AO* can quickly adjust its plan without starting from scratch.
- One of the cool things about AO* is that it uses a combination of OR and AND operations. This means it can consider multiple paths simultaneously making it really good at adapting to new information.
- It's like being able to plan a route while also keeping an eye on alternative options. This makes AO* a powerful tool for tasks that involve uncertainty and change.

2.5 Adversarial Search

Adversarial search is a search, where we examine the problem which arises when we try to plan ahead of the world and other agents are planning against us.

- In previous topics, we have studied the search strategies which are only associated with a single agent that aims to find the solution which often expressed in the form of a sequence of actions.
- But, there might be some situations where more than one agent is searching for the solution in the same search space, and this situation usually occurs in game playing.
- The environment with more than one agent is termed as **multi-agent environment**, in which each agent is an opponent of other agent and playing against each other. Each agent needs to consider the action of other agent and effect of that action on their performance.
- So, Searches in which two or more players with conflicting goals are trying to explore the same search space for the solution, are called adversarial searches, often known as Games.
- Games are modeled as a Search problem and heuristic evaluation function, and these are the two main factors which help to model and solve games in AI.

2.6 AI and Game Playing

Game playing in artificial intelligence refers to the development and application of algorithms that enable computers to engage in strategic decision-making within the context of games. These algorithms, often termed game playing algorithms in AI, empower machines to mimic human-like gameplay by evaluating potential moves, predicting opponent responses, and making informed choices that lead to favorable outcomes. This concept extends the capabilities of AI systems beyond mere computation and calculation, enabling them to participate in interactive scenarios and make choices based on strategic thinking.

Types of Game Playing in Artificial Intelligence

Game playing in artificial intelligence encompasses a diverse array of strategies, each aimed at enabling AI systems to excel in games and strategic decision-making scenarios. These

strategies, often referred to as game playing algorithms in AI, can be broadly classified into two main categories: rule-based systems and machine learning-based systems.

1. **Rule-Based Systems**

Rule-based systems, a cornerstone of game playing in AI, rely on predefined sets of rules to govern the behavior of AI agents during gameplay. These rules encapsulate strategies, tactics, and heuristics designed by human experts. These experts analyze the game, anticipate possible moves, and formulate guidelines that the AI adheres to.

In rule-based systems, decisions are based on a deterministic process where each move is evaluated against the predefined rules. These rules dictate how the AI should react to various game states, opponent moves, and potential outcomes. Rule-based approaches are particularly effective in games with well-defined rules and relatively simple decision trees, such as Tic-Tac-Toe.

2. **Machine Learning-Based Systems**

Machine learning-based systems, on the other hand, represent a paradigm shift in game playing in AI. Instead of relying on fixed rules, these systems utilize algorithms to learn from experience and adapt their strategies accordingly. These algorithms process vast amounts of data generated through gameplay, identifying patterns, correlations, and optimal moves.

Reinforcement learning is a prominent example of machine learning-based systems in game playing algorithms in AI. Here, AI agents play games repeatedly, receiving rewards for favorable moves and penalties for unfavorable ones. Over time, the AI learns to maximize rewards by exploring different strategies and refining its decision-making processes. This approach has propelled AI achievements in complex games like Go and chess, showcasing the capacity to tackle intricate decision trees.

Mini-Max Algorithm in Artificial Intelligence

The Mini-Max algorithm is a cornerstone of game playing in artificial intelligence, allowing AI agents to make strategic decisions in competitive games. It operates on the principle that in a two-player, zero-sum game, one player's gain is balanced by the other player's loss. This powerful algorithm enables AI players to assess different moves and select the one that maximizes their advantage while considering their opponent's optimal response.

Advantages of Game Playing in Artificial Intelligence

- **Enhanced Strategic Thinking:** Game-playing algorithms empower AI to strategize and make optimal choices, improving decision-making in various scenarios.
- **Adaptive Learning:** Machine learning-driven approaches enable AI to learn and refine strategies, adapting over time for better performance.
- **Real-world Relevance:** Strategies developed in games find applications in diverse fields, enhancing decision-making in practical situations.

- **Efficient Decision-making:** Algorithms like Alpha-Beta Pruning optimize computation, enabling AI to efficiently explore complex game scenarios.
- **Benchmarking AI Progress:** AI's success in games serves as a marker of advancement, showcasing the growth of AI's strategic capabilities.

Disadvantages of Game Playing in Artificial Intelligence

- **Computational Complexity:** Game-playing algorithms can be computationally intensive, limiting their real-time application in complex scenarios.
- **Limited Generalization:** Strategies developed for specific games might not readily apply to broader real-world decision-making contexts.
- **Lack of Creativity:** AI's decisions are based on algorithms and past experiences, lacking the creativity and intuition that humans possess.
- **Complexity of Game Rules:** Adapting game-playing algorithms to diverse games with intricate rules can be challenging and time-consuming.
- **Overfitting:** In machine learning-based approaches, there's a risk of overfitting to limited training data, leading to suboptimal decisions in novel situations.

1. Knowledge Representation and Reasoning

- **3.1 What to represent, Knowledge**
- Humans are best at understanding, reasoning, and interpreting knowledge. Human knows things, which is knowledge and as per their knowledge they perform various actions in the real world. **But how machines do all these things comes under knowledge representation and reasoning**. Hence we can describe Knowledge representation as following:

  - Knowledge representation and reasoning (KR, KRR) is the part of Artificial intelligence which concerned with AI agents thinking and how thinking contributes to intelligent behavior of agents.
  - It is responsible for representing information about the real world so that a computer can understand and can utilize this knowledge to solve the complex real world problems such as diagnosis a medical condition or communicating with humans in natural language.
  - It is also a way which describes how we can represent knowledge in artificial intelligence. Knowledge representation is not just storing data into some database, but it also enables an intelligent machine to learn from that knowledge and experiences so that it can behave intelligently like a human.

What to Represent:

Following are the kind of knowledge which needs to be represented in AI systems:

- **Object:** All the facts about objects in our world domain. E.g., Guitars contains strings, trumpets are brass instruments.
- **Events:** Events are the actions which occur in our world.
- **Performance:** It describebehavior which involves knowledge about how to do things.
- **Meta-knowledge:** It is knowledge about what we know.
- **Facts:** Facts are the truths about the real world and what we represent.
- **Knowledge-Base:** The central component of the knowledge-based agents is the knowledge base. It is represented as KB. The Knowledgebase is a group of the Sentences (Here, sentences are used as a technical term and not identical with the English language).

Knowledge: Knowledge is awareness or familiarity gained by experiences of facts, data, and situations. Following are the types of knowledge in artificial intelligence:

Types of knowledge

1. Declarative Knowledge:

- Declarative knowledge is to know about something.
- It includes concepts, facts, and objects.
- It is also called descriptive knowledge and expressed in declarativesentences.
- It is simpler than procedural language.

2. Procedural Knowledge

- It is also known as imperative knowledge.

- Procedural knowledge is a type of knowledge which is responsible for knowing how to do something.
- It can be directly applied to any task.
- It includes rules, strategies, procedures, agendas, etc.
- Procedural knowledge depends on the task on which it can be applied.

3. Meta-knowledge:

- Knowledge about the other types of knowledge is called Meta-knowledge.

4. Heuristic knowledge:

- Heuristic knowledge is representing knowledge of some experts in a filed or subject.
- Heuristic knowledge is rules of thumb based on previous experiences, awareness of approaches, and which are good to work but not guaranteed.

5. Structural knowledge:

- Structural knowledge is basic knowledge to problem-solving.
- It describes relationships between various concepts such as kind of, part of, and grouping of something.
- It describes the relationship that exists between concepts or objects.

- **3.2 Properties of Knowledge Representation System, Approaches**
  A good knowledge representation system must possess the following properties.

1. **1.Representational Accuracy:**
   KR system should have the ability to represent all kind of required knowledge.
2. **2.Inferential Adequacy:**
   KR system should have ability to manipulate the representational structures to produce new knowledge corresponding to existing structure.
3. **3.Inferential Efficiency:**
   The ability to direct the inferential knowledge mechanism into the most productive directions by storing appropriate guides.
4. **4. Acquisitional efficiency-** The ability to acquire the new knowledge easily using automatic methods.

Approaches to knowledge representation:

There are mainly four approaches to knowledge representation, which are givenbelow:

1. Simple relational knowledge:

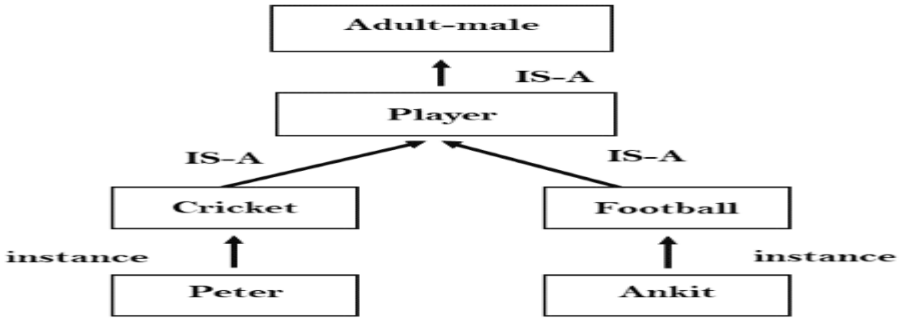
- It is the simplest way of storing facts which uses the relational method, and each fact about a set of the object is set out systematically in columns.
- This approach of knowledge representation is famous in database systems where the relationship between different entities is represented.
- This approach has little opportunity for inference.

Example: The following is the simple relational knowledge representation.

Player	Weight	Age
Player1	65	23
Player2	58	18
Player3	75	24

2. Inheritable knowledge:

- In the inheritable knowledge approach, all data must be stored into a hierarchy of classes.
- All classes should be arranged in a generalized form or a hierarchal manner.
- In this approach, we apply inheritance property.
- Elements inherit values from other members of a class.
- This approach contains inheritable knowledge which shows a relation between instance and class, and it is called instance relation.
- Every individual frame can represent the collection of attributes and its value.
- In this approach, objects and values are represented in Boxed nodes.
- We use Arrows which point from objects to their values.
- **Example:**



3. Inferential knowledge:

- Inferential knowledge approach represents knowledge in the form of formal logics.
- This approach can be used to derive more facts.
- It guaranteed correctness.
- **Example:** Let's suppose there are two statements:
    a. Marcus is a man
    b. All men are mortal
    Then it can represent as;

    man(Marcus)
    ∀x = man (x) ----------> mortal (x)s

4. Procedural knowledge:

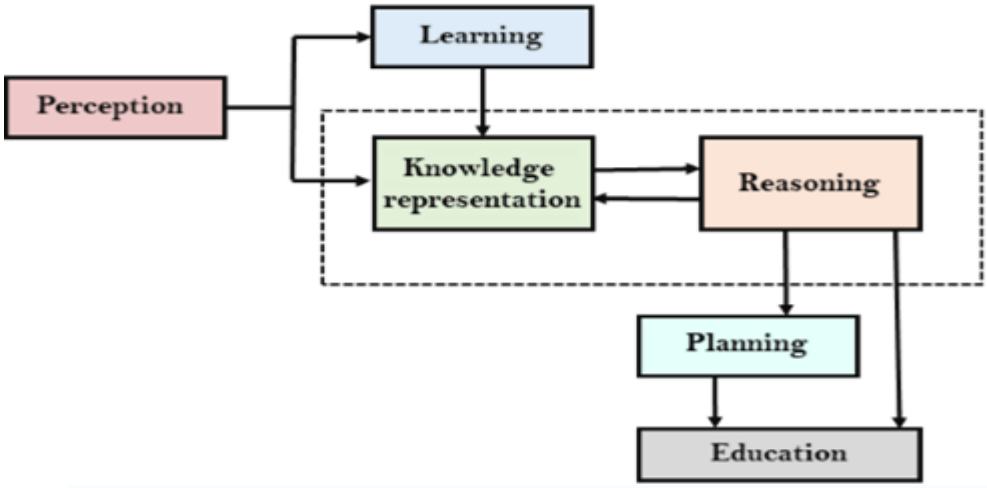
- Procedural knowledge approach uses small programs and codes which describes how to do specific things, and how to proceed.
- In this approach, one important rule is used which is **If-Then rule**.
- In this knowledge, we can use various coding languages such as **LISP language** and **Prolog language**.
- We can easily represent heuristic or domain-specific knowledge using this approach.
- But it is not necessary that we can represent all cases in this approach.

- **3.3 Knowledge Representation**

AI knowledge cycle:

An Artificial intelligence system has the following components for displaying intelligent behaviour:

- Perception
- Learning
- Knowledge Representation and Reasoning
- Planning
- Execution



Perception

Perception is the process of acquiring information from the environment through sensors. This information is then processed and interpreted to generate knowledge.

Learning

Learning is the process of acquiring new knowledge from experience. This can be achieved through supervised learning, unsupervised learning, or reinforcement learning.

Knowledge Representation & Reasoning

Knowledge representation and reasoning is the stage where acquired knowledge is transformed into a form that can be processed by machines. This involves choosing an appropriate KR technique and representing knowledge using that technique. Reasoning involves using the knowledge represented to draw inferences and make decisions.

Planning

Planning is the stage where the system uses the acquired knowledge and reasoning to generate a sequence of actions to achieve a particular goal. This involves selecting the most appropriate actions to achieve the goal while considering any constraints or limitations.

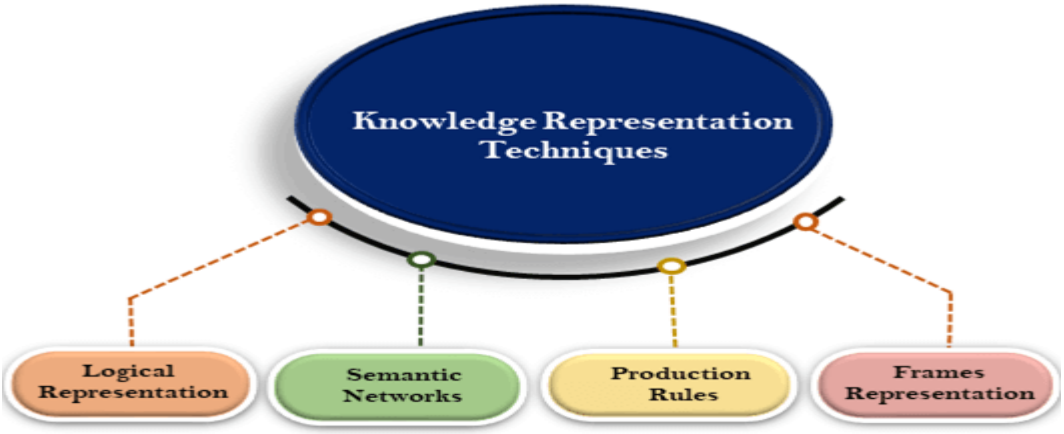
Execution

Execution is the final stage where the system performs the planned actions. The success of the execution depends on the accuracy and completeness of the knowledge representation, reasoning, and planning.

Techniques of knowledge representation

There are mainly four ways of knowledge representation which are given as follows:

1. Logical Representation
2. Semantic Network Representation
3. Frame Representation
4. Production Rules



1.Logical Representation

A language with certain concrete principles that deals with propositions and has no ambiguity in representation is referred to as logical representation. Drawing a conclusion based on numerous criteria is referred to as logical representation. Some important communication guidelines are laid out in this diagram. It's made up of well-defined syntax and semantics that facilitate sound inference. Each sentence could be translated into logics using the syntax and semantics.

Syntax:

- Syntaxes are the principles that govern how legal sentences are constructed in logic.
- It determines the symbol we can use to express knowledge.
- What is the best way to write those symbols?

Semantics:

- The rules by which we can comprehend a phrase in logic are known as semantics.
- Semantic also entails giving each statement a meaning.
- There are primarily two logics that can be used to represent logic:
    - Propositional Logics
    - Predicate logics

Advantages of logical representation:

- We can do logical reasoning with the help of logical representation.

- Programming languages are built on the foundation of logical representation.

Advantages of logical representation:

- We can do logical reasoning with the help of logical representation.
- Programming languages are built on the foundation of logical representation.

Disadvantages of logical Representation:

The various types of knowledge are as follows:

- Logical representations have some limitations and are difficult to use.
- It's possible that the logical representation technique isn't very natural, and inference isn't particularly efficient.

Note: That logical representation and logical reasoning are not the same thing; logical representation is a representation language, and reasoning is a logical thinking process.

2. Semantic Network Representation

For knowledge representation, semantic networks are an alternative to predicate logic. We can express our knowledge in Semantic Networks as graphical networks. This network is made up of nodes that represent things and arcs that describe their relationships. Semantic networks may classify objects in a variety of ways and link them together.

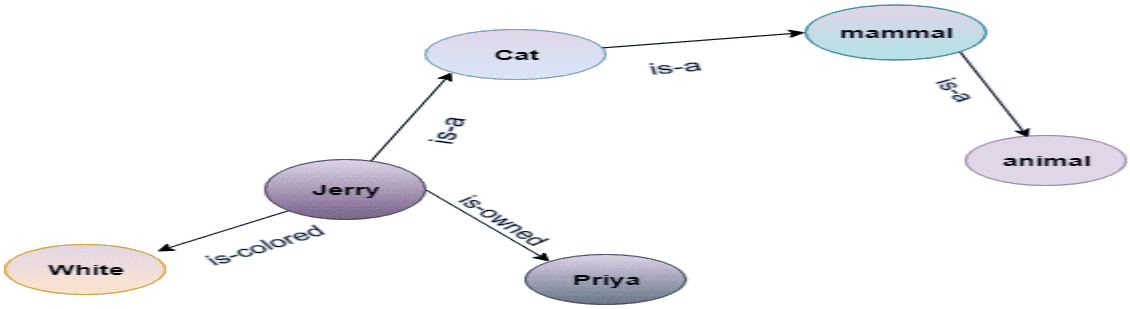
This representation consists of basically two types of relations:

1. IS-A relation (Inheritance)
2. Kind-of-relation

Example: Some statements which we have to represent in the form of nodes and arcs are as follows.

Statements:

- Jerry is a cat.
- Jerry is a mammal.
- Jerry is owned by Priya.
- Jerry is brown colored.
- All Mammals are animal.



We've used nodes and arcs to represent different types of knowledge in the diagram above. Each object has some sort of relationship with another object.

Disadvantage in Semantic representation:

- Because we need to traverse the entire network tree to answer some questions, semantic networks take longer to compute at runtime. In the worst-case situation, we may discover that the answer does not exist in this network after traversing the entire tree.
- Semantic networks attempt to replicate human-like memory (which has 1015 neurons and linkages) in order to store information, however in actuality, such a large semantic network is impossible to construct.
- These types of representations are insufficient since they lack an equivalent quantifier, such as all, some, none, and so on.
- The link names in semantic networks are not defined in any way.
- These networks aren't intelligent and rely on the system's inventor.

Advantages of Semantic network:

- Semantic networks are a natural way to represent information.
- Semantic networks are a transparent way of conveying meaning.
- These networks are straightforward and simple to comprehend.

3. Frame Representation

A frame is a record-like structure that contains a set of properties and their values to describe a physical thing. Frames are a sort of artificial intelligence data structure that splits knowledge into substructures by depicting stereotyped situations. It is made up of a set of slots and slot values. These slots can come in any shape or size. Facets are the names and values assigned to slots.

Facets: Facets are the numerous aspects of a slot machine. Facets are characteristics of frames that allow us to constrain them. Example: When data from a specific slot is required, IF-NEEDED facts are used. A frame can have any number of slots, each of which can contain any number of facets, each of which can have any number of values.

Semantic networks gave rise to frames, which later evolved into our modern-day classes and objects. A single frame is of limited utility. A frames system is made up of a group of interconnected frames. Knowledge about an object or event can be kept in the knowledge base in the frame. The frame is a form of technology that can be used in a wide range of applications.

Example: 1Let's take an example of a frame for a book

Slots	**Filter**
Title	Artificial Intelligence
Genre	Computer Science
Author	Peter Norvig
Edition	PThird Edition
Year	1996
Page	1152

Example: 2

Let's pretend we're dealing with an entity, Peter. Peter is a professional engineer, and he is 25 years old. He lives in the city of London, in the country of England. The frame representation for this is as follows:

Slots	**Filter**
Name	Peter
Profession	Doctor
Age	25
Marital Status	Single
Year	1996
Weight	78

Advantages of frame representation:

- By grouping related facts, the frame knowledge representation makes programming easier.
- Many AI applications employ the frame representation because it is rather flexible.
- Adding slots for additional attributes and relations is a breeze.
- It's simple to add default data and look for missing variables.
- The frame representation is simple to grasp and visualize.

Disadvantages of frame representation:

- The inference mechanism in a frame system is difficult to process.
- Frame representation does not allow for a smooth progression of the inference procedure.
- The approach to frame representation is rather broad

4. Production Rules

Production rules system consists of (condition, action) pairs which means, "If condition then action". It has basically three parts:

- The set of production rules
- Working Memory
- The recognize-act-cycled

The agent in production rules checks for the condition, and if it exists, the production rule fires and the appropriate action is taken. The rule's condition component determines which rule can be used to solve an issue. The action portion, on the other hand, is responsible for carrying out the corresponding problem-solving actions. A recognize-act cycle is the name given to the entire procedure.

Working memory stores a description of the present state of problem-solving, and rules can be used to write knowledge to it. Other rules may be triggered by this knowledge.

If a new scenario (state) arises, numerous production rules will be fired at the same time, which is known as a conflict set. In this case, the agent must choose a rule from among these sets, and it is called a recognize-act cycle.

Example:
IF (at bus stop AND bus arrives) THEN action (get into the bus)
IF (on the bus AND paid AND empty seat) THEN action (sit down).
IF (on bus AND unpaid) THEN action (pay charges).
IF (bus arrives at destination) THEN action (get down from the bus.

Advantages of Production rule:

1. The production rules are written in plain English.
2. We can quickly delete, add, or modify an individual rule because the production rules are relatively modular

Disadvantages of Production rule:

1. The production rule system has no learning capabilities because it does not save the solution to the problem for future use.
2. Many rules may be active during the execution of the program, making rule-based production systems inefficient.

3.4 Reasoning and Types of reasoning

Reasoning:

The reasoning is the mental process of deriving logical conclusion and making predictions from available knowledge, facts, and beliefs. Or we can say, "**Reasoning is a way to infer facts from existing data**." It is a general process of thinking rationally, to find valid conclusions.

In artificial intelligence, the reasoning is essential so that the machine can also think rationally as a human brain, and can perform like a human.

Types of Reasoning

In artificial intelligence, reasoning can be divided into the following categories:

- Deductive reasoning
- Inductive reasoning
- Abductive reasoning
- Common Sense Reasoning
- Monotonic Reasoning
- Non-monotonic Reasoning
- **Deductive Reasoning:** Deductive Reasoning is the strategic approach that uses available facts, information or knowledge to draw valid conclusions. It basically beliefs in the facts and ideas before drawing any result. Deductive reasoning uses a top-down approach. In deductive reasoning, the arguments can be valid or invalid based on the value of the premises. If the value of the premises is true, then the conclusion is also true. Deductive reasoning helps in scanning the generalized statement into a valid conclusion. Some of the examples are
  - People who are aged 20 or above are active users of the internet.
  - Out of the total number of students present in the class, the ratio of boys is more than the girls.
- **Inductive Reasoning:** Inductive reasoning is completely different from the deductive reasoning approach because Inductive reasoning is associated with the hypothesis-generating approach rather than drawing any particular conclusion to the facts at the beginning of the process. Inductive reasoning help in making generalization from specific facts and knowledge. Inductive reasoning is the bottom-up process. In inductive Reasoning even if the premises are true there is no chance that the conclusion will also be true because it depends upon the inductive argument which can be either strong or weak. Some of the examples are:
  - All the students present in the classroom are from London.
  - Always the hottest temperature is recorded in Death Valley.
- **Common Sense Reasoning:** Common sense reasoning is the most occurred type of reasoning in daily life events. It is the type of reasoning which comes from experiences. When a human face a different situation in life it gains some knowledge. So whenever in the next point of time it faces a similar type of situation then it uses its previous experiences to draw a conclusion to do situation. Some of the examples are:
  - when a bike crosses the traffic signal when it is red then it learns from its mistakes and next time the bike is aware of the signal and actions.
  - While overtaking someone on the road what all ideas should be kept in mind.
- **Monotonic Reasoning:** It is the type of reasoning which follows a different approach towards the thinking process it uses facts, information, and knowledge to

draw a conclusion about the problem but the major point is its conclusion remain fixed permanently once it is decided because even if we add new information or facts to the existing one the conclusion remains the same it doesn't change. Monotonic reasoning is used mainly in conventional reasoning systems and logic-based systems. Some Examples of monotonic are:

- The Sahara Desert of the world is one of the most spectacular deserts.
- One of the longest rivers in the world is the Nile River.

- **Abductive Reasoning:** Abductive Reasoning is a type of reasoning which acts differently from all the above reasoning strategies. It begins with an incomplete set of facts, information and knowledge and then proceeds to find the most deserving explanation and conclusion. It draws conclusions based on what facts you know at present rather than collecting some outdated facts and information. It mostly plays a great role in the daily life decision-making process. Some of the examples are:
  - Doctor drawing conclusions regarding your health based on test reports.
  - A bowl of soup is kept and vapour evaporating from it which draws the conclusion that the bowl is hot in nature.

Non-monotonic Reasoning

In Non-monotonic reasoning, some conclusions may be invalidated if we add some more information to our knowledge base.

Logic will be said as non-monotonic if some conclusions can be invalidated by adding more knowledge into our knowledge base.

Non-monotonic reasoning deals with incomplete and uncertain models.

Human perceptions for various things in daily life, "is a general example of non-monotonic reasoning.

Example: Let suppose the knowledge base contains the following knowledge:

- **Birds can fly**
- **Penguins cannot fly**
- **Pitty is a bird**

So from the above sentences, we can conclude that **Pitty can fly**.

However, if we add one another sentence into knowledge base "**Pitty is a penguin**", which concludes "**Pitty cannot fly**", so it invalidates the above conclusion.

4. Machine Learning

4.1 Machine Learning

4.2 Statistical or Unsupervised Learning

4.3 ML Properties

4.4 Reinforcement Learning

4.5 Decision Tree

What is Machine Learning?

Machine Learning is the field of study that gives computers the capability to learn without being explicitly programmed. ML is one of the most exciting technologies that one would have ever come across. As it is evident from the name, it gives the computer that makes it more similar to humans: The ability to learn. Machine learning is actively being used today, perhaps in many more places than one would expect.

Features of Machine learning

- Machine learning is data driven technology. Large amount of data generated by organizations on daily bases. So, by notable relationships in data, organizations makes better decisions.
- Machine can learn itself from past data and automatically improve.
- From the given dataset it detects various patterns on data.
- For the big organizations branding is important and it will become more easy to target relatable customer base.
- It is similar to data mining because it is also deals with the huge amount of data.

How does machine learning work?

UC Berkeley (link resides outside ibm.com) breaks out the learning system of a machine learning algorithm into three main parts.

1. A Decision Process: In general, machine learning algorithms are used to make a prediction or classification. Based on some input data, which can be labeled or unlabeled, your algorithm will produce an estimate about a pattern in the data.
2. An Error Function: An error function evaluates the prediction of the model. If there are known examples, an error function can make a comparison to assess the accuracy of the model.

3. A Model Optimization Process: If the model can fit better to the data points in the training set, then weights are adjusted to reduce the discrepancy between the known example and the model estimate. The algorithm will repeat this iterative "evaluate and optimize" process, updating weights autonomously until a threshold of accuracy has been met.