UNIT - I Introduction: AI problems, Agents and Environments, Structure of Agents, Problem Solving Agents Basic Search Strategies: Problem Spaces, Uninformed Search (Breadth-First, Depth-First Search, Depth-first with Iterative Deepening), Heuristic Search (Hill Climbing, Generic Best-First, A*), Constraint Satisfaction (Backtracking, Local Search)

Introduction

What is artificial intelligence?

Artificial Intelligence is the branch of computer science concerned with making computers behave like humans.

John McCarthy, who coined the term in 1956, defines it as "the science and engineering of making intelligent machines, especially intelligent computer programs."

Why Artificial Intelligence?

Before Learning about Artificial Intelligence, we should know that what is the importance of AI and why should we learn it. Following are some main reasons to learn about AI:

- With the help of AI, you can create such software or devices which can solve real-world problems very easily and with accuracy such as health issues, marketing, traffic issues, etc.
- With the help of AI, you can create your personal virtual Assistant, such as Cortana, Google Assistant, Siri, etc.
- With the help of AI, you can build such Robots which can work in an environment where survival of humans can be at risk.
- AI opens a path for other new technologies, new devices, and new Opportunities.

Goals of Artificial Intelligence

Following are the main goals of Artificial Intelligence:

1. Replicate human intelligence
2. Solve Knowledge-intensive tasks
3. An intelligent connection of perception and action
4. Building a machine which can perform tasks that requires human intelligence such as:
    - Proving a theorem
    - Playing chess
    - Plan some surgical operation

- Driving a car in traffic

5. Creating some system which can exhibit intelligent behavior, learn new things by itself, demonstrate, explain, and can advise to its user.

Advantages of Artificial Intelligence

Following are some main advantages of Artificial Intelligence:

- **High Accuracy with less errors:** AI machines or systems are prone to less errors and high accuracy as it takes decisions as per pre-experience or information.
- **High-Speed:** AI systems can be of very high-speed and fast-decision making, because of that AI systems can beat a chess champion in the Chess game.
- **High reliability:** AI machines are highly reliable and can perform the same action multiple times with high accuracy.
- **Useful for risky areas:** AI machines can be helpful in situations such as defusing a bomb, exploring the ocean floor, where to employ a human can be risky.
- **Digital Assistant:** AI can be very useful to provide digital assistant to the users such as AI technology is currently used by various E-commerce websites to show the products as per customer requirement.
- **Useful as a public utility:** AI can be very useful for public utilities such as a self-driving car which can make our journey safer and hassle-free, facial recognition for security purpose, Natural language processing to communicate with the human in human-language, etc.

Disadvantages of Artificial Intelligence

Every technology has some disadvantages, and the same goes for Artificial intelligence. Being so advantageous technology still, it has some disadvantages which we need to keep in our mind while creating an AI system. Following are the disadvantages of AI:

- **High Cost:** The hardware and software requirement of AI is very costly as it requires lots of maintenance to meet current world requirements.
- **Can't think out of the box:** Even we are making smarter machines with AI, but still they cannot work out of the box, as the robot will only do that work for which they are trained, or programmed.
- **No feelings and emotions:** AI machines can be an outstanding performer, but still it does not have the feeling so it cannot make any kind of emotional attachment with human, and may sometime be harmful for users if the proper care is not taken.

- **Increase dependency on machines:** With the increment of technology, people are getting more dependent on devices and hence they are losing their mental capabilities.
- **No Original Creativity:** As humans are so creative and can imagine some new ideas but still AI machines cannot beat this power of human intelligence and cannot be creative and imaginative.

Application of AI

Artificial Intelligence has various applications in today's society. It is becoming essential for today's time because it can solve complex problems with an efficient way in multiple industries. such as Healthcare, entertainment, finance, education, etc. AI is making our daily life more comfortable and fast.

Following are some sectors which have the application of Artificial Intelligence:

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.

AI in Healthcare

- In the last, five to ten years, AI becoming more advantageous for the healthcare industry and going to have a significant impact on this industry.
- 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.

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.

AI in Finance

- AI and finance industries are the best matches for each other. The finance industry is implementing automation, chatbot, adaptive intelligence, algorithm trading, and machine learning into financial processes.

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.

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.

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.

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.

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.

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.

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.

AI in E-commerce

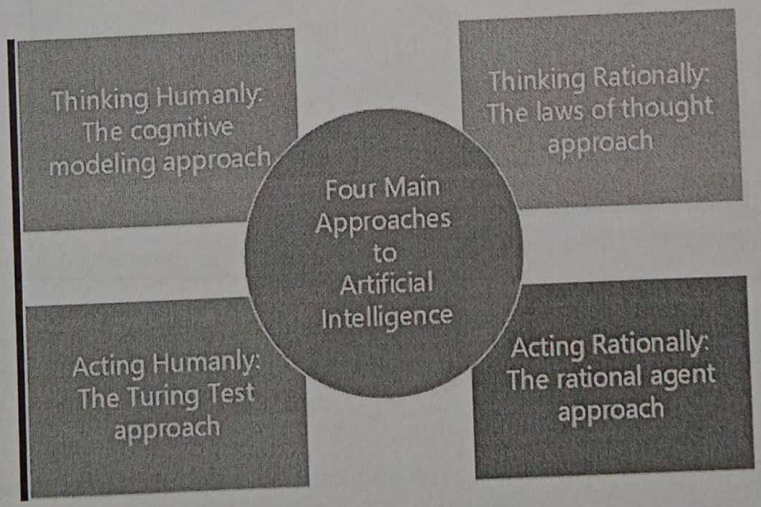
- AI is providing a competitive edge to the e-commerce industry, and it is becoming more demanding in the e-commerce business. AI is helping shoppers to discover associated products with recommended size, color, or even brand.

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.

AI has 4 Components which are

- Thinking Humanly
- Thinking Rationally
- Acting Humanly
- Acting Rationally



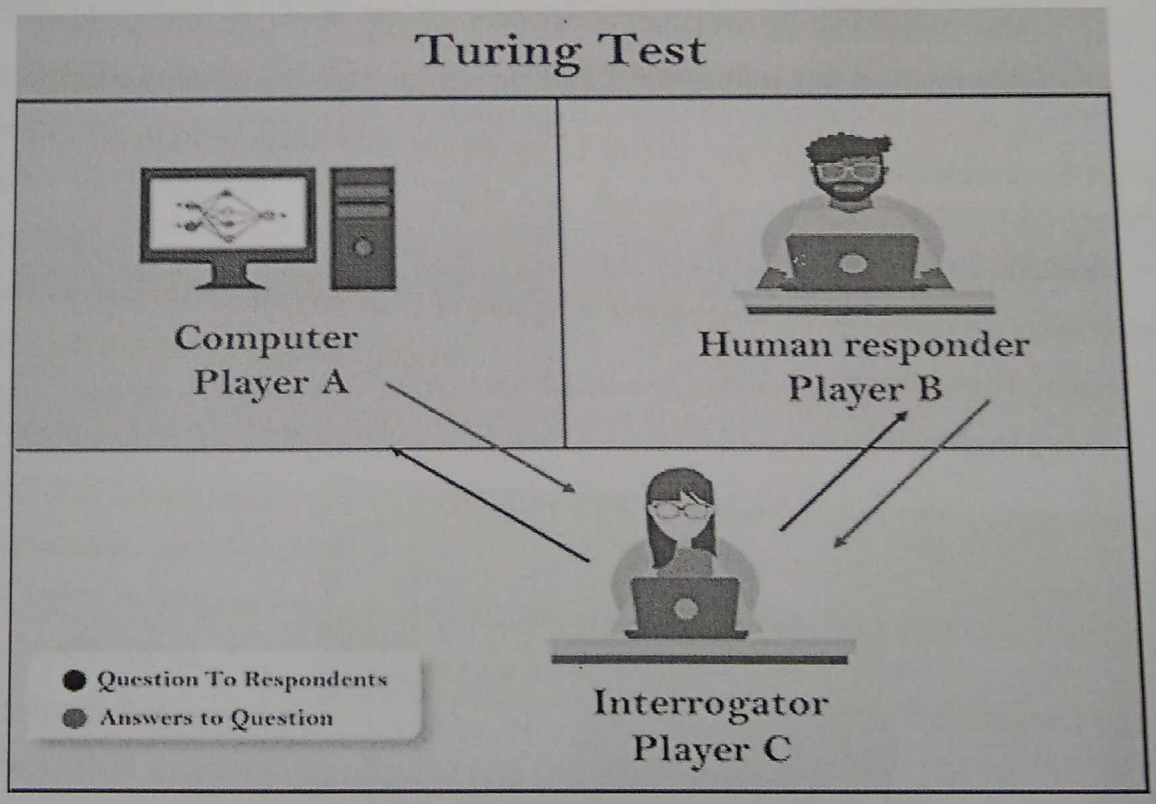
Turing Test in AI

In 1950, Alan Turing introduced a test 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.

Turing Test was introduced by Turing in his 1950 paper, "Computing Machinery and Intelligence," which considered the question, "Can Machine think?"

The Turing test is based on a party game "Imitation game," with some modifications. This game involves three players in which one player is Computer, another player is human responder, and the third player is a human Interrogator, who is isolated from other two players and his job is to find that which player is machine among two of them.

Consider, Player A is a computer, Player B is human, and Player C is an interrogator. Interrogator is aware that one of them is machine, but he needs to identify this on the basis of questions and their responses.



	Humanly	Rationally
Thinking	Systems should solve problems as humans do	Use of Logic Patterns for arguments structures that yields correct conclusion
Acting	Acting like humans- Natural language processing Automated reasoning Knowledge representation	Study of rational agents Expected to act so as to achieve best expected outcome

What are Agent and Environment?

An **agent** is anything that can perceive its environment through **sensors** and acts upon that environment through **effectors.**

- A **human agent** has sensory organs such as eyes, ears, nose, tongue and skin parallel to the sensors, and other organs such as hands, legs, mouth, for effectors.
- A **robotic agent** replaces cameras and infrared range finders for the sensors, and various motors and actuators for effectors.
- A **software agent** has encoded bit strings as its programs and actions.
- Hence the world around us is full of agents such as thermostat, cellphone, camera, and even we are also agents.

Before moving forward, we should first know about sensors, effectors, and actuators.

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.

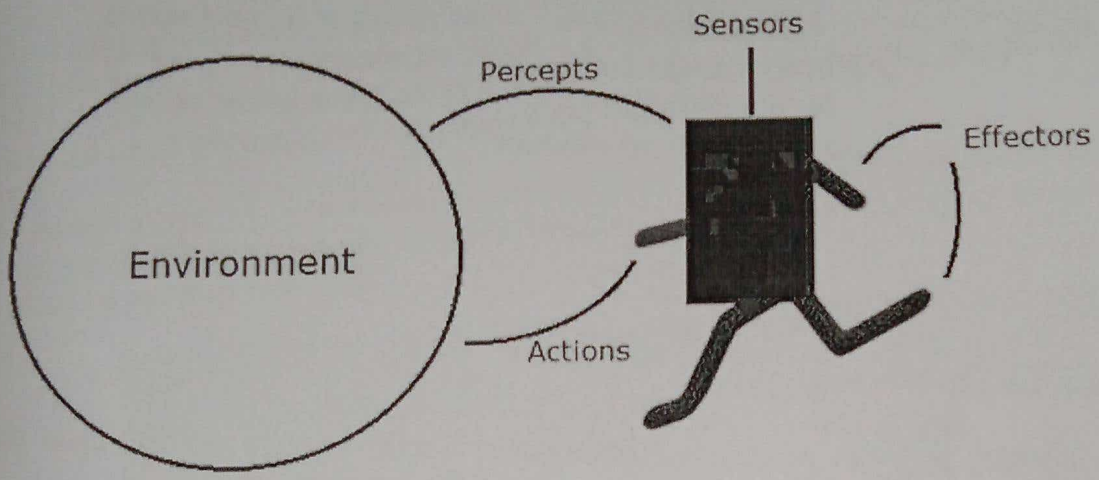
Actuators: Actuators are the component of machines that converts energy into motion. The actuators are only responsible for moving and controlling a system. An actuator can be an electric motor, gears, rails, etc.

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

The Structure of Intelligent Agents

Agent's structure can be viewed as –

- Agent = Architecture + Agent Program
- Architecture = the machinery that an agent executes on.
- Agent Program = an implementation of an agent function



Types of AI Agents

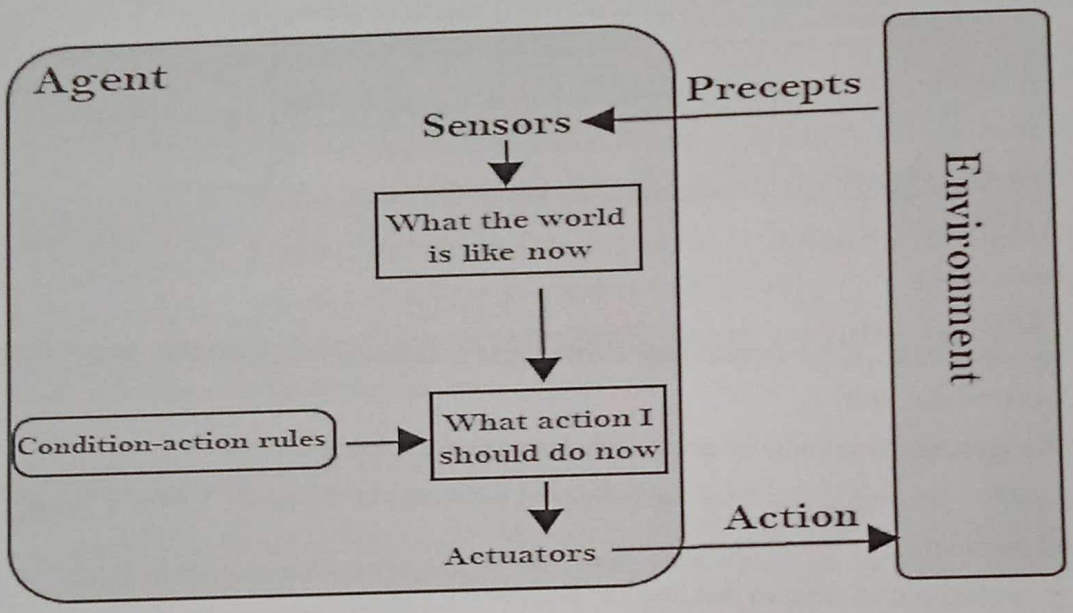
Agents can be grouped into five classes based on their degree of perceived intelligence and capability. All these agents can improve their performance and generate better action over the time. These are given below:

- Simple Reflex Agent
- Model-based reflex agent
- Goal-based agents
- Utility-based agent
- Learning agent

1. Simple Reflex agent:

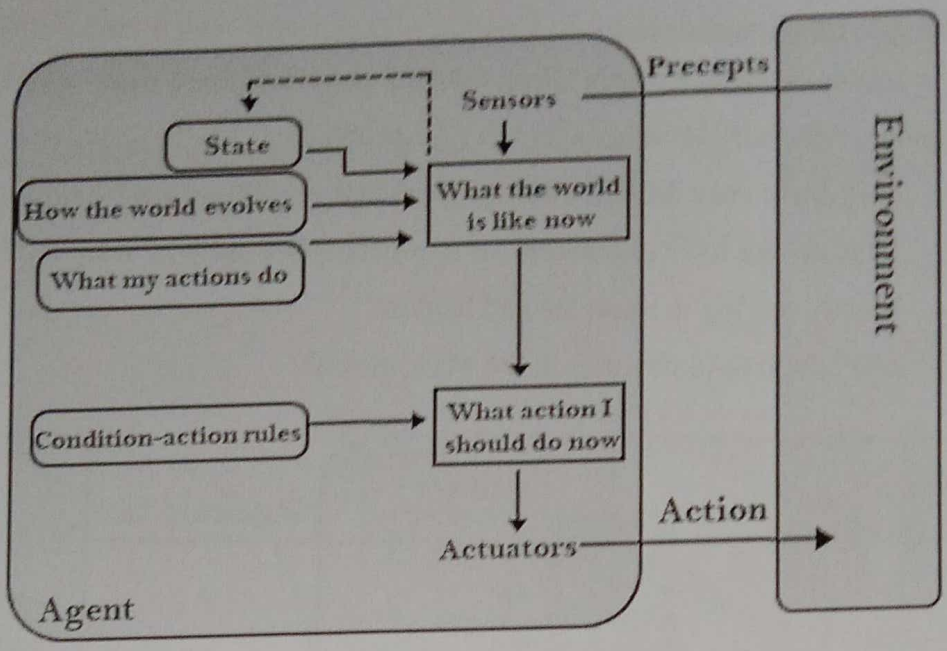
- The Simple reflex agents are the simplest agents. These agents take decisions on the basis of the current percepts and ignore the rest of the percept history.
- These agents only succeed in the fully observable environment.
- The Simple reflex agent does not consider any part of percepts history during their decision and action process.

- The Simple reflex agent works on Condition-action rule, which means it maps the current state to action. Such as a Room Cleaner agent, it works only if there is dirt in the room.
- Problems for the simple reflex agent design approach:
  - They have very limited intelligence
  - They do not have knowledge of non-perceptual parts of the current state
  - Mostly too big to generate and to store.
  - Not adaptive to changes in the environment.



2. Model-based reflex agent

- The Model-based agent can work in a partially observable environment, and track the situation.
- A model-based agent has two important factors:
  - **Model:** It is knowledge about "how things happen in the world," so it is called a Model-based agent.
  - **Internal State:** It is a representation of the current state based on percept history.
- These agents have the model, "which is knowledge of the world" and based on the model they perform actions.
- Updating the agent state requires information about:
  a. How the world evolves
  b. How the agent's action affects the world.



3. Goal-based agents

- The knowledge of the current state environment is not always sufficient to decide for an agent to what to do.
- The agent needs to know its goal which describes desirable situations.
- Goal-based agents expand the capabilities of the model-based agent by having the "goal" information.
- They choose an action, so that they can achieve the goal.
- These agents may have to consider a long sequence of possible actions before deciding whether the goal is achieved or not. Such considerations of different scenario are called searching and planning, which makes an agent proactive.

4. Utility-based agents

- These agents are similar to the goal-based agent but provide an extra component of utility measurement which makes them different by providing a measure of success at a given state.
- Utility-based agent act based not only goals but also the best way to achieve the goal.
- The Utility-based agent is useful when there are multiple possible alternatives, and an agent has to choose in order to perform the best action.
- The utility function maps each state to a real number to check how efficiently each action achieves the goals.

5. Learning Agents

- A learning agent in AI is the type of agent which can learn from its past experiences, or it has learning capabilities.
- It starts to act with basic knowledge and then able to act and adapt automatically through learning.
- A learning agent has mainly four conceptual components, which are:
    a. **Learning element:** It is responsible for making improvements by learning from environment
    b. **Critic:** Learning element takes feedback from critic which describes that how well the agent is doing with respect to a fixed performance standard.
    c. **Performance element:** It is responsible for selecting external action
    d. **Problem generator:** This component is responsible for suggesting actions that will lead to new and informative experiences.
- Hence, learning agents are able to learn, analyze performance, and look for new ways to improve the performance.

Agent Environment in AI

An environment is everything in the world which surrounds the agent, but it is not a part of an agent itself. An environment can be described as a situation in which an agent is present.

The environment is where agent lives, operate and provide the agent with something to sense and act upon it. An environment is mostly said to be non-feministic.

Features of Environment

An environment can have various features from the point of view of an agent:

1. Fully observable vs Partially Observable
2. Static vs Dynamic
3. Discrete vs Continuous
4. Deterministic vs Stochastic
5. Single-agent vs Multi-agent
6. Episodic vs sequential
7. Known vs Unknown
8. Accessible vs Inaccessible

1. Fully observable vs Partially Observable:

- If an agent sensor can sense or access the complete state of an environment at each point of time then it is **a fully observable** environment, else it is **partially observable.**
- A fully observable environment is easy as there is no need to maintain the internal state to keep track history of the world.
- An agent with no sensors in all environments then such an environment is called as **unobservable.**

2. Deterministic vs Stochastic:

- If an agent's current state and selected action can completely determine the next state of the environment, then such environment is called a deterministic environment.
- A stochastic environment is random in nature and cannot be determined completely by an agent.
- In a deterministic, fully observable environment, agent does not need to worry about uncertainty.

3. Episodic vs Sequential:

- In an episodic environment, there is a series of one-shot actions, and only the current percept is required for the action.
- However, in Sequential environment, an agent requires memory of past actions to determine the next best actions.

4. Single-agent vs Multi-agent

- If only one agent is involved in an environment, and operating by itself then such an environment is called single agent environment.
- However, if multiple agents are operating in an environment, then such an environment is called a multi-agent environment.
- The agent design problems in the multi-agent environment are different from single agent environment.

5. Static vs Dynamic:

- If the environment can change itself while an agent is deliberating then such environment is called a dynamic environment else it is called a static environment.

- Static environments are easy to deal because an agent does not need to continue looking at the world while deciding for an action.
- However for dynamic environment, agents need to keep looking at the world at each action.
- Taxi driving is an example of a dynamic environment whereas Crossword puzzles are an example of a static environment.

6. Discrete vs Continuous:

- If in an environment there are a finite number of percepts and actions that can be performed within it, then such an environment is called a discrete environment else it is called continuous environment.
- A chess game comes under discrete environment as there is a finite number of moves that can be performed.
- A self-driving car is an example of a continuous environment.

7. Known vs Unknown

- Known and unknown are not actually a feature of an environment, but it is an agent's state of knowledge to perform an action.
- In a known environment, the results for all actions are known to the agent. While in unknown environment, agent needs to learn how it works in order to perform an action.
- It is quite possible that a known environment to be partially observable and an Unknown environment to be fully observable.

8. Accessible vs Inaccessible

- If an agent can obtain complete and accurate information about the state's environment, then such an environment is called an Accessible environment else it is called inaccessible.
- An empty room whose state can be defined by its temperature is an example of an accessible environment.
- Information about an event on earth is an example of Inaccessible environment.

Problem-solving agents:

In Artificial Intelligence, Search techniques are universal problem-solving methods. **Rational agents** or **Problem-solving agents** in AI mostly used these search strategies or algorithms to

solve a specific problem and provide the best result. Problem-solving agents are the goal-based agents and use atomic representation. In this topic, we will learn various problem-solving search algorithms.

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.

Types of search algorithms

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) search	informed search (Heuristic search)
1.Search without information	1.Search with information
2.No knowledge	2.Use Knowledge to find steps to solution
3.Time Consuming	3.Quick Solution
4.More Complexity(Time,Space)	4.Less Complexity
5.less Efficient	5.More Efficient
6.BFS,DFS.Depthfirst with interative Deeping	6.A*,AO*,Hill Climbing, Best-First

Uninformed (Blind search) search

BFS (Breadth-first search)

→ It 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

→ BFS implemented using FIFO Queue.

→ level search technique

→ complete

→ optimal

→ Time complexity $O(V+E)$

V = no of node

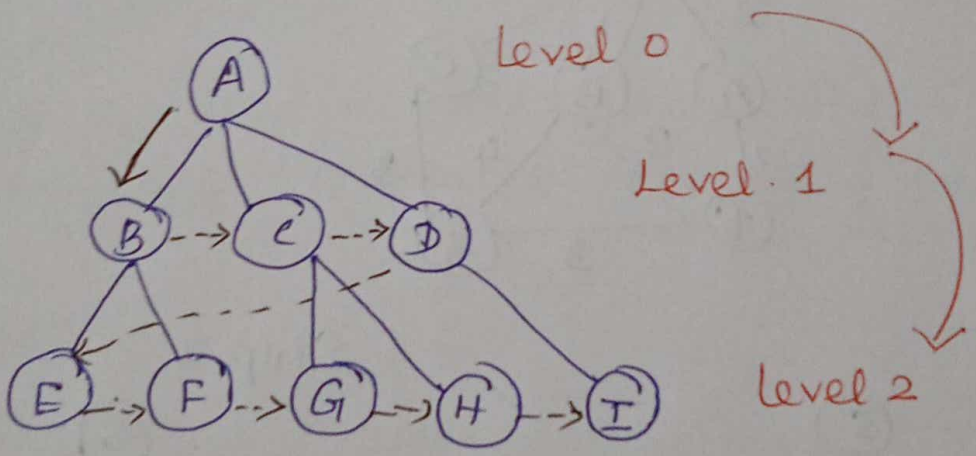
E = no of edge

space complexity $O(b^d)$

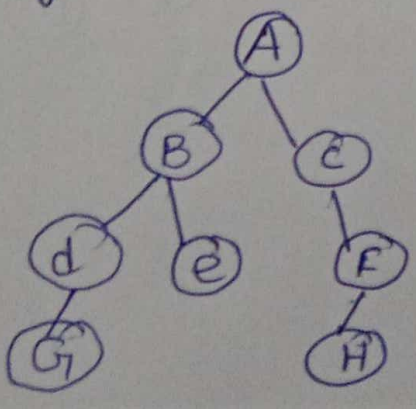
b = branch factor

d = depth

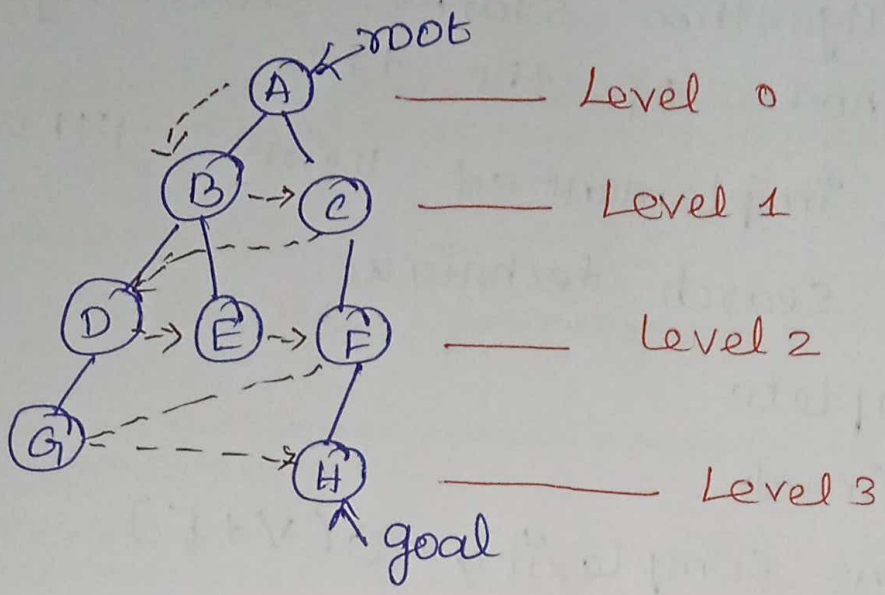
eg



Find the route from A to H using BFS for an given Tree

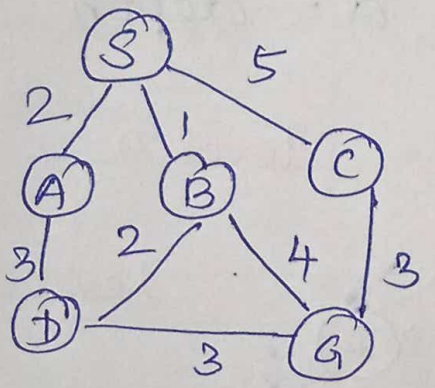


Step - 1



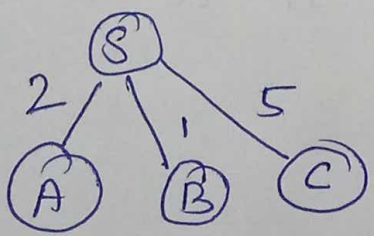
A → B → C → D → E → F → G → H

2. Find the route from S to G using BFS for a given graph.

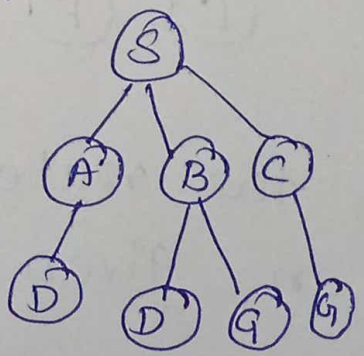


Step 1

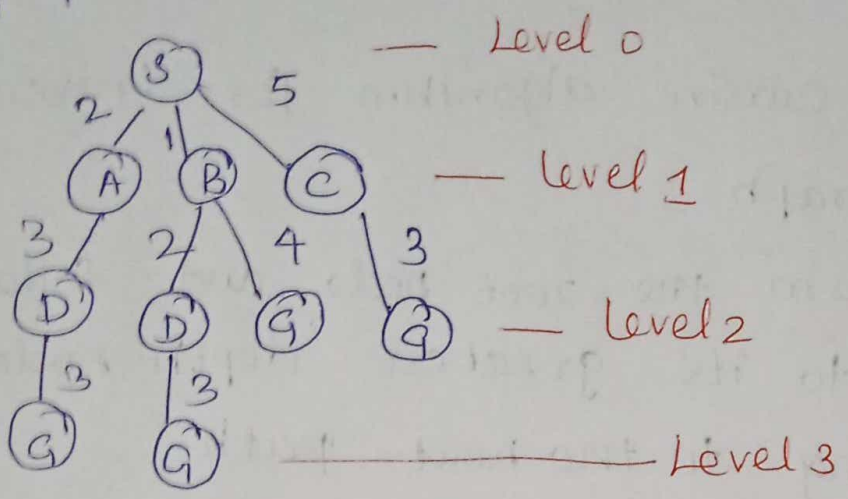
Step 2



Step 3



Step 4



Route from S → G

S → A → B → C → D → G

Applications of BFS :

- BFS can be used to find the neighbor ing locations from a given source location
- BFS is used to determine the shortest Path and minimum spanning tree

drawback

- Requires more memory than DFS
- Can be quite complex to run
- In efficient for deep solution ie if a solutio is far away then it consumes time.

Depth-first search

→ DFS is a recursive algorithm for traversing a tree or graph

→ 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 (LIFO) data structure for its implementation

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

advantage

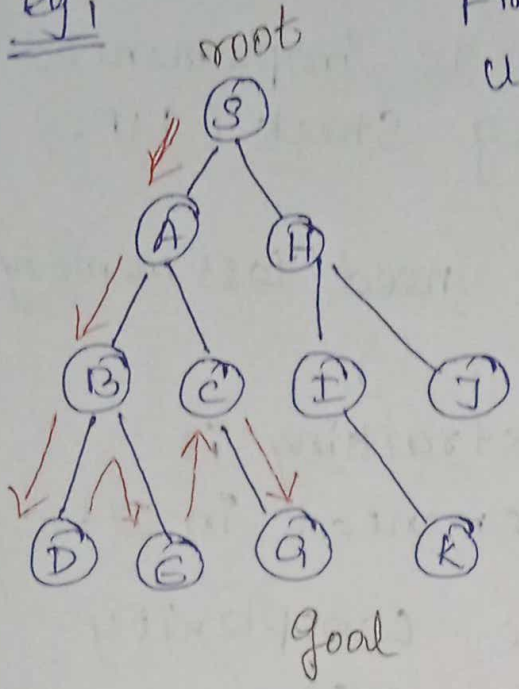
- DFS need very less memory
- It takes less time to reach to the goal node than BFS.

disadvantage

- DFS algorithm goes for deep down searching and sometime it may go to the infinite loop.

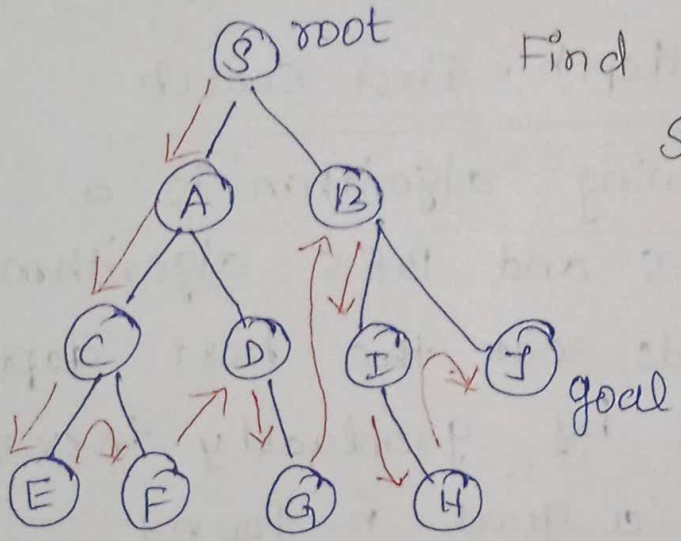
eg 1

Find the route Between S to G using DFS.



$S \rightarrow A \rightarrow B \rightarrow D \rightarrow E \rightarrow C \rightarrow G$

eg 2



Find the route between S → J using DFS

$S \rightarrow A \rightarrow C \rightarrow E \rightarrow F \rightarrow D \rightarrow G \rightarrow B \rightarrow I \rightarrow H \rightarrow J$

BFS VS DFS

BFS	DFS
i) Breadth First Search	Depth First Search
ii) BFS traverses the tree level wise	ii) DFS traverses the tree Depth wise

BFS	DFS
iii) BFS is implemented using queue (FIFO)	iii) DFS is implemented using stack (LIFO)
iv) Bfs requires more memory	iv) DFS need less memory
v) NO backtracking	v) Backtracking is implemented in DFS
vi) Time complexity O(V+E)	vi) Time complexity O(V+E)

Iterative Deepening depth-first search

→ The iterative deepening algorithm is a combination of DFS and BFS algorithms.

→ This algorithm finds out the best depth limit and does it by gradually increasing the limit until a goal is found

→ This algorithm perform depth-first search up to a certaen depth limit and it keeps increasing the depth limit after each iteration until the goal node is found.

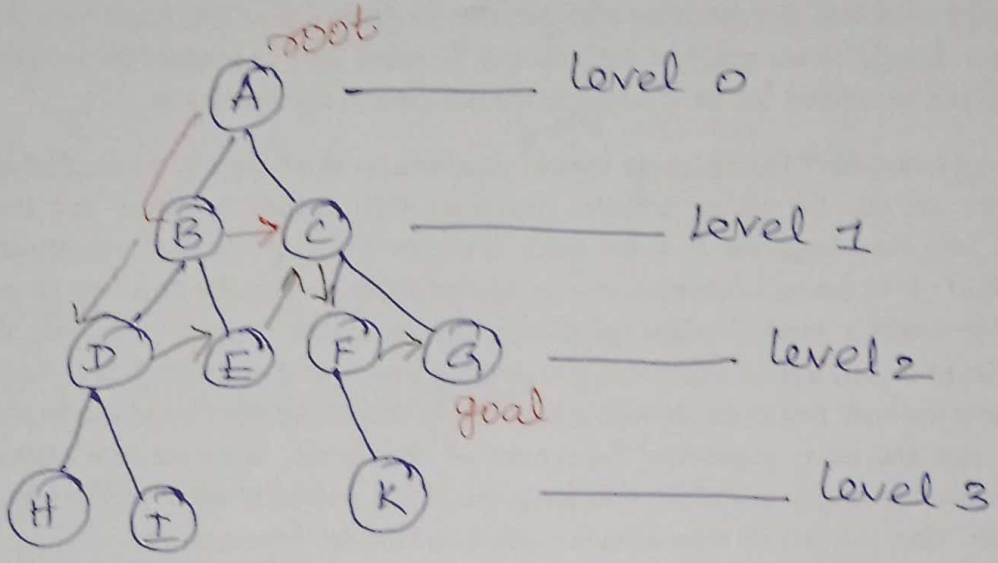
advantage

→ It combines the benefits of BFS and DFS search algorithm in terms of fast search and memory efficiency.

disadvantage

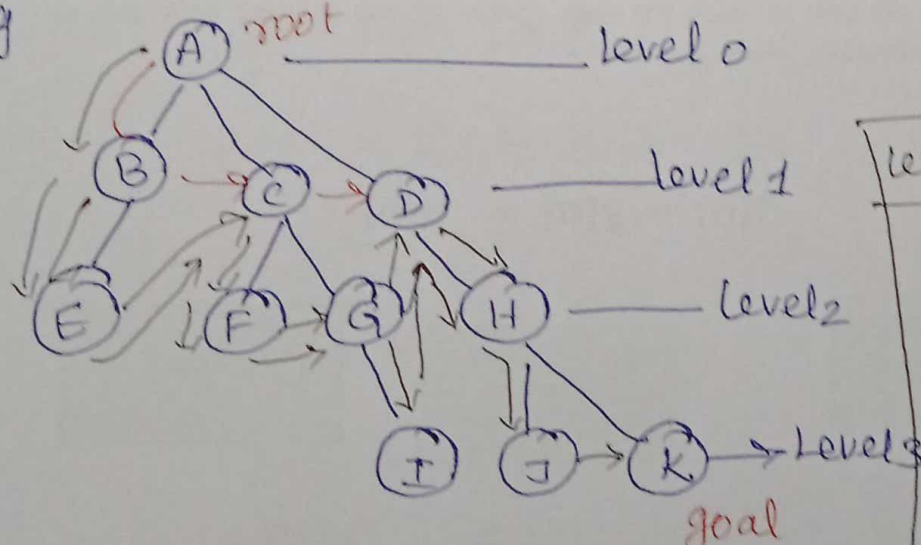
→ It repeats all the work of the previous phase

eg



Level	IDDFS
0	A
1	A B C
2	A B D E C F G

eg



Level	IDDFS
0	A
1	A B C D
2	A B E C F G D H
3	A B E C F G I D H J K

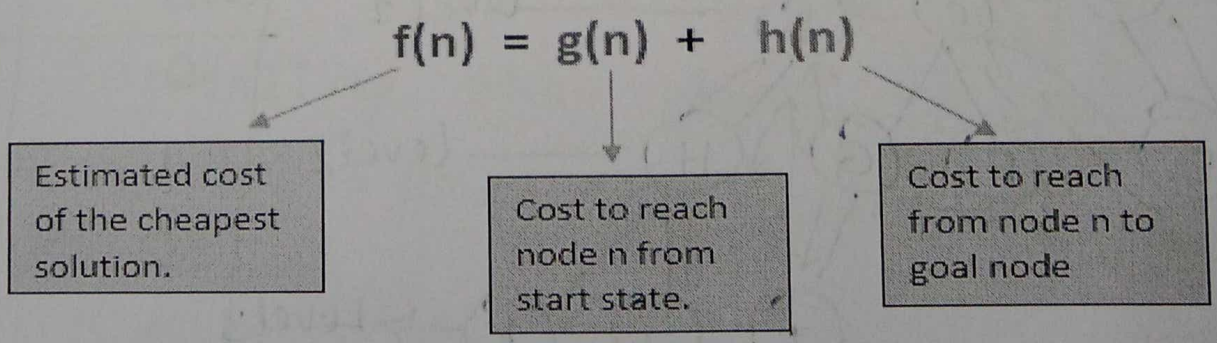
What is Heuristics? A heuristic is a technique that is used to solve a problem faster than the classic methods. These techniques are used to find the approximate solution of a problem when classical methods do not. Heuristics are said to be the problem-solving techniques that result in practical and quick solutions. Heuristics are strategies that are derived from past experience with similar problems. Heuristics use practical methods and shortcuts used to produce the solutions that may or may not be optimal, but those solutions are sufficient in a given limited.

Why do we need heuristics? Heuristics are used in situations in which there is the requirement of a short-term solution. On facing complex situations with limited resources and time, Heuristics can help the companies to make quick decisions by shortcuts and approximated calculations. Most of the heuristic methods involve mental shortcuts to make decisions on past experiences. The heuristic method might not always provide us the finest solution, but it is assured that it helps us find a good solution in a reasonable time. Based on context, there can be different heuristic methods that correlate with the problem's scope. The most common heuristic methods are - trial and error, guesswork, the process of elimination, historical data analysis. These methods involve simply available information that is not particular to the problem but is most appropriate. They can include representative, affect, and availability heuristics.

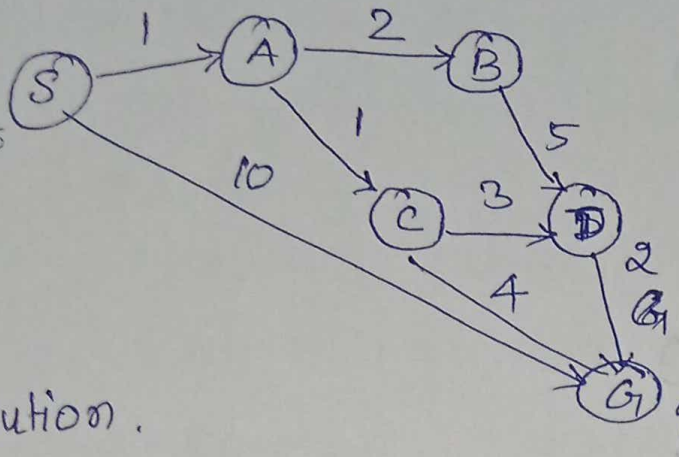
Admissibility of the heuristic function is given as:

$h(n) <= h^*(n)$ **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.**

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**.



eg



State	h(n)
S	5
A	3
B	4
C	2
D	6
G	0

Solution.

① S → A

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

$= 1 + 3 = 4$

S → G :

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

$= 10 + 0 = 10$

$4 < 10$

take 4

S → A

② S → A → B

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

$g(n) = 1 + 2 = 3$

$f(n) = 3 + 4 = 7$

S → A → C

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

$g(n) = 1 + 1 = 2$

$f(n) = 2 + 2 = 4$

$4 < 7$

take 4

S → A → C

③ S → A → C → D

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

$g(n) = 1+1+3 = 5$

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

$= 5 + 6 = 11$

$S \rightarrow A \rightarrow C \rightarrow G$

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

$g(n) = 1+1+4 = 6$

$f(n) = 6 + 0$

$= 6$

$6 < 11$

$\boxed{S \rightarrow A \rightarrow C \rightarrow G}$

Final path

it provides the optimal Path with cost 6.

advantage

→ A* algorithm is the best algorithm than other Search algorithms.

→ A* search algorithm is optimal and complete

→ This algorithm can solve very complex problems.

disadvanage

→ It does not always produce the shortest Path as it mostly based on heuristics

Best first search

→ It is a technique that decides which node is to be visited next, by checking which nodes is the most promising one.

→ Informed search

→ It always selects the path which appears best at that moment

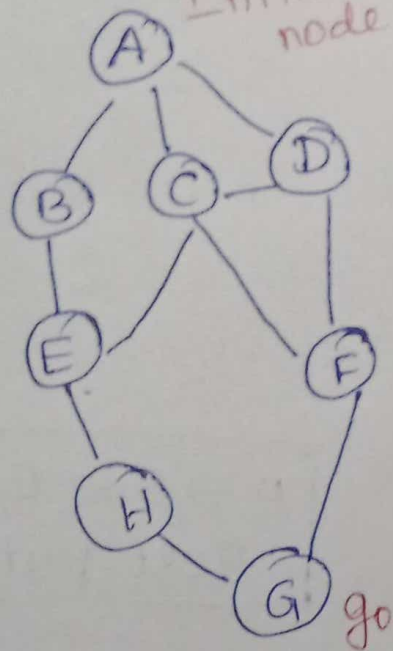
→ combination of DFS and BFS.

Two list

open list → Expanded, not yet evaluated

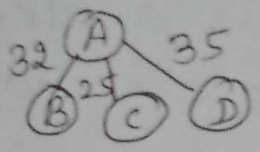
closed list → Already evaluated

eg

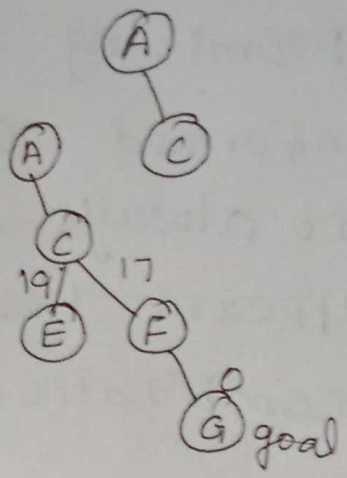


node	h(n)
A	40
B	32
C	25
D	35
E	19
F	17
H	10
G	0

open	closed
[A]	[]
[C,B,D]	[A]
[B,D]	[A C]
[F,E,B,D]	[A C]
[G, E, B, D]	[A C F]



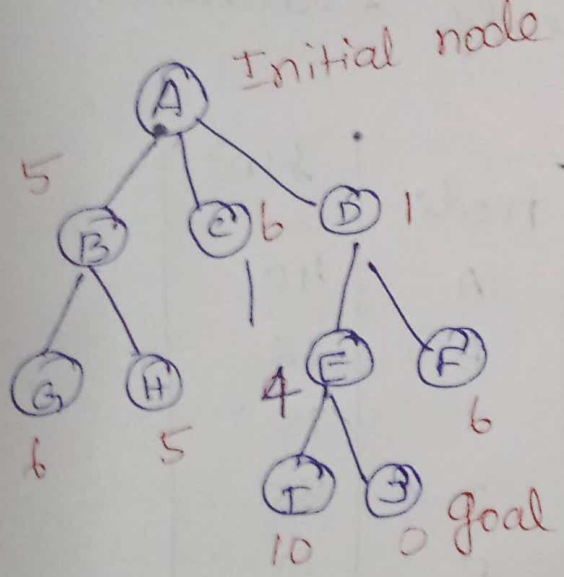
CBD



ACFG: Path

A → C → F → G

eg 2



open	closed
[A]	[]
[D B C]	[A]
[E F B C]	[A D]
[J I F B C]	[A D]
	A D E
	[A D E J]

A → D → E → J
Best-Path

Advantage

BFS can switch between BFS and DFS by gaining the advantage of both the algorithms.

→ This algorithm is more efficient than BFS and DFS algorithm.

disadvantage

1. Chances of getting stuck in a loop are higher.

—x—

Constraint Satisfaction problem (CSP)

→ CSP consist of 3 components

V, D, C

V = set of variables {V1, V2 ... Vn}

D = set of Domain {D1, D2 ... Dn}

C = set of constraints.

C_i = (scope, relation)

→ Scope is a set of variables that participate in constraints

→ Relation that define the values that variable can take

V_1 V_2

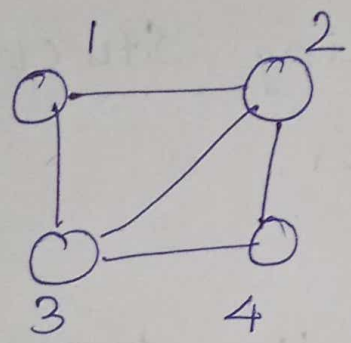
A B

$C_1 = ((V_1, V_2)\ (V_1 \neq V_2))$

$C_1 = ((V_1, V_2)\ (A, B))$

CSP problem solved by backtracking algorithm / Methods (DFS)

Constraint graph



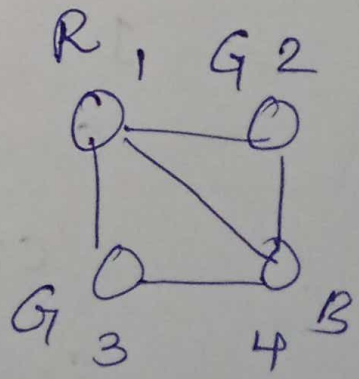
$V = \{1, 2, 3, 4\}$

$D = \{$ Red, green, Blue $\}$

$C = \{1 \neq 2,\ 1 \neq 3,\ 1 \neq 4,\ 2 \neq 4,\ 3 \neq 4\}$

adjacent nodes should not have same color

	1	2	3	4
Initial Domain	RGB	RGB	RGB	RGB
1 = R	R	GB	GB	GB
2 = G	R	G	GB	B
3 = B	R	G	B	B
3 = G	R	G	G	B



Crypt - Arithmatic problem

constraints

① Every character must have a unique value

② digits should be from 0 - 9 only

③ starting character of number cannot be '0'

④ Cryptarithmetic problem will have only one solution

⑤ Addition of no with itself is always even.

⑥ In case of addition of 2 no's if there is carry to next step then the carry can only be 1.

eg

```
  TO
+ GO
-----
 OUT
```

T	2
O	1
G	8
U	0

```
  2  1
+ 8  1
-------
1  0  2
```

2 + 8 = 10 (2 digit) with carry 1

eg 2

$C_4\ C_3\ C_2\ C_1$

S E N D

\+ M O R E

M O N E Y

character	code
S	9
E	5
N	6
D	7
M	1
O	0
R	8
Y	2

$C_4 = 1$, $M = 1$

if $M = 1$ then $S = 9$

if $M = 1, S = 9$ then $O = 0$

C_4 C_3 ① C_2 ① C_1

[9 S] [5 E] [6 N] [7 D]

[1 M] [0 O] [8 R] [5 E]

[1 M] [0 O] [6 N] [5 E] [2 Y]

Step 1

$C_2 + E + O = N$

if $C_2 = 0$

$0 + E + 0 = N$

$E = N$ ✗

if $C_2 = 1$

$1 + E + 0 = N$

$1 + E = N$ — ①

Step 2

$C_1 + N + R = E + 10$

if $C_1 = 1$

$1 + (1 + E) + R = 10 + E$

$R = 10 - 2$

$R = 8$

Step 3

$D + E = Y$

Possible no

2, 3, 4, 5, 6 7

let $E = 5$ then $N = 6$

$D + 5 = 10 + Y$

$D = 5 + Y$

$D = 7$ & $Y = 2$

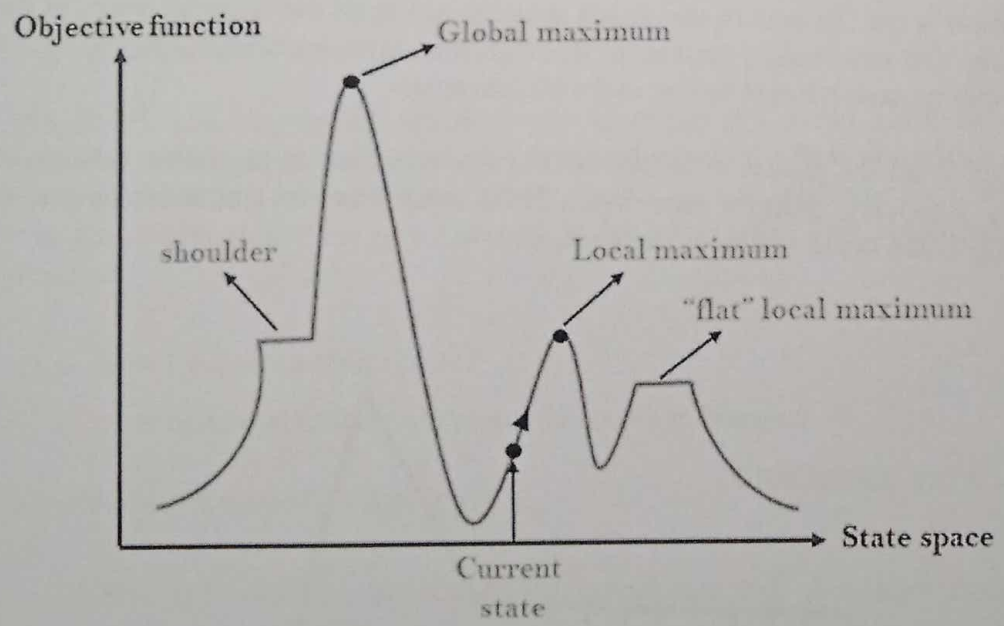
Hill Climbing Algorithm in Artificial Intelligence

- 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

- 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.

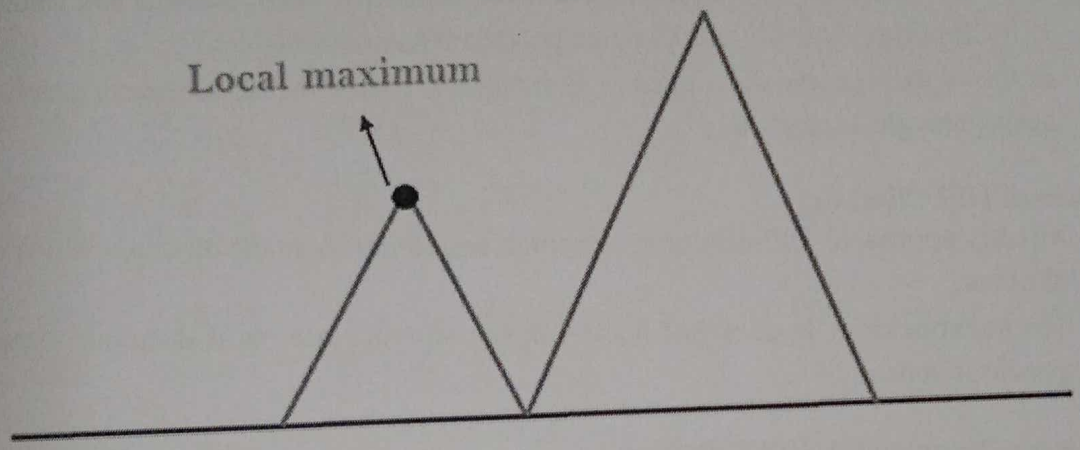
State-space Diagram for Hill Climbing:



Problems in Hill Climbing Algorithm

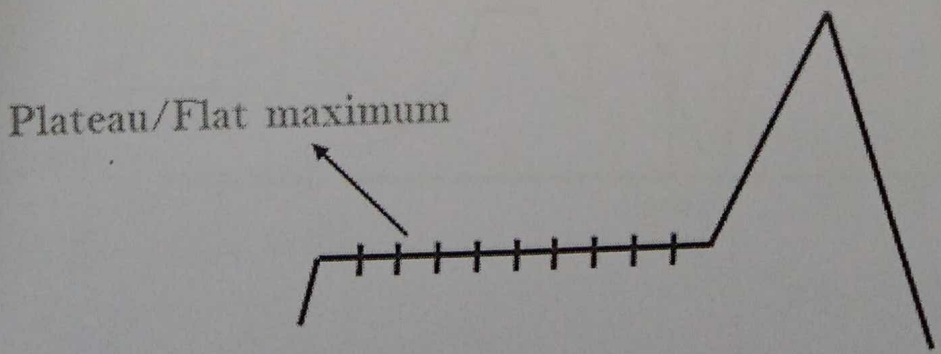
1.Local Maximum: A local maximum is a peak state in the landscape which is better than each of its neighboring states, but there is another state also present which is higher than the local maximum.

Solution: Backtracking technique can be a solution of the local maximum in state space landscape. Create a list of the promising path so that the algorithm can backtrack the search space and explore other paths as well.



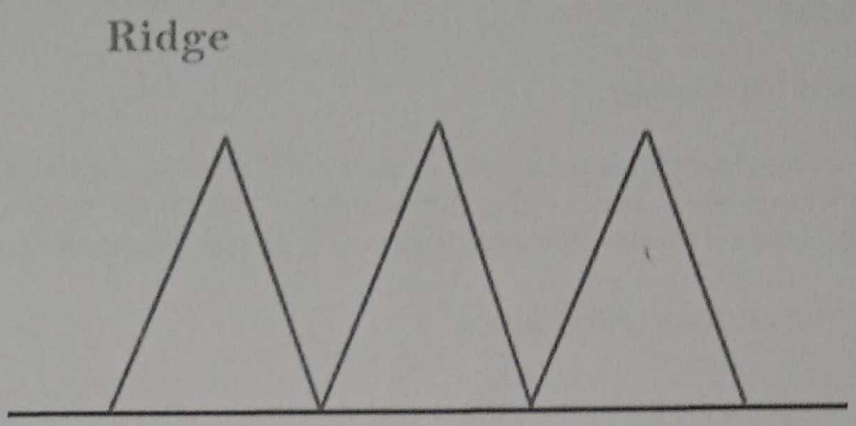
2. Plateau: A plateau is the flat area of the search space in which all the neighbor states of the current state contains the same value, because of this algorithm does not find any best direction to move. A hill-climbing search might be lost in the plateau area.

Solution: The solution for the plateau is to take big steps or very little steps while searching, to solve the problem. Randomly select a state which is far away from the current state so it is possible that the algorithm could find non-plateau region.



3. Ridges: A ridge is a special form of the local maximum. It has an area which is higher than its surrounding areas, but itself has a slope, and cannot be reached in a single move.

Solution: With the use of bidirectional search, or by moving in different directions, we can improve this problem.



Types of Hill Climbing Algorithm

- Simple hill Climbing:
- Steepest-Ascent hill-climbing:
- Stochastic hill Climbing:

1. Simple Hill Climbing:

Simple hill climbing is the simplest way to implement a hill climbing algorithm. **It only evaluates the neighbor node state at a time and selects the first one which optimizes current cost and set it as a current state**. It only checks it's one successor state, and if it finds better than the current state, then move else be in the same state. This algorithm has the following features:

- Less time consuming
- Less optimal solution and the solution is not guaranteed

Algorithm for Simple Hill Climbing:

- **Step 1:** Evaluate the initial state, if it is goal state then return success and Stop.
- **Step 2:** Loop Until a solution is found or there is no new operator left to apply.
- **Step 3:** Select and apply an operator to the current state.

- **Step 4:** Check new state:
    a. If it is goal state, then return success and quit.
    b. Else if it is better than the current state then assign new state as a current state.
    c. Else if not better than the current state, then return to step2.
- **Step 5:** Exit.

2. Steepest-Ascent hill climbing:

The steepest-Ascent algorithm is a variation of simple hill climbing algorithm. This algorithm examines all the neighboring nodes of the current state and selects one neighbor node which is closest to the goal state. This algorithm consumes more time as it searches for multiple neighbors

Algorithm for Steepest-Ascent hill climbing:

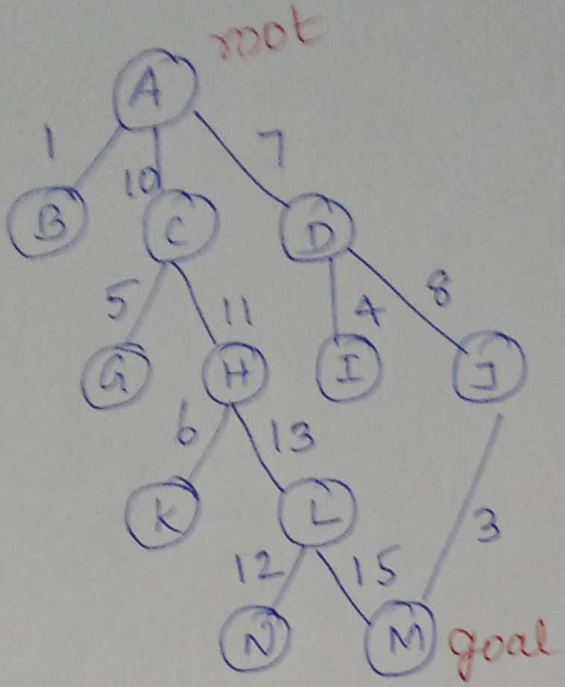
- **Step 1:** Evaluate the initial state, if it is goal state then return success and stop, else make current state as initial state.
- **Step 2:** Loop until a solution is found or the current state does not change.
    a. Let SUCC be a state such that any successor of the current state will be better than it.
    b. For each operator that applies to the current state:
        a. Apply the new operator and generate a new state.
        b. Evaluate the new state.
        c. If it is goal state, then return it and quit, else compare it to the SUCC.
        d. If it is better than SUCC, then set new state as SUCC.
        e. If the SUCC is better than the current state, then set current state to SUCC.
- **Step 5:** Exit.

3. Stochastic hill climbing:

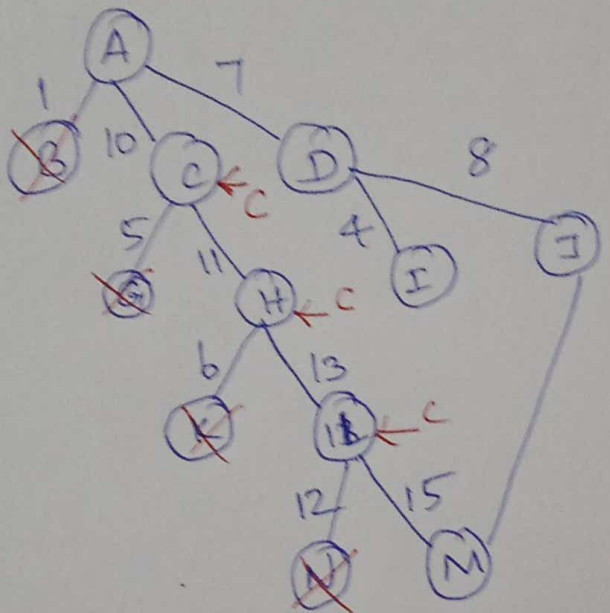
Stochastic hill climbing does not examine for all its neighbor before moving. Rather, this search algorithm selects one neighbor node at random and decides whether to choose it as a current state or examine another state.

Simple hill climbing

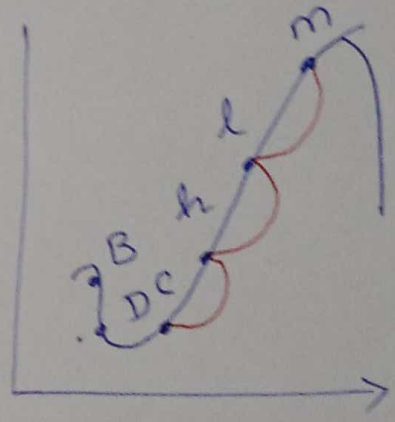
eg



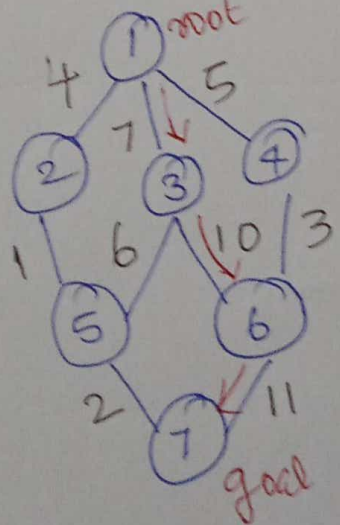
Step 1



A → C → h → l → m



Steepest Ascent hill climbing



consume less time

1 → 3 → 6 → 7