AIML
ASSIGNMENT-I

1) what is AI? Explain the AI techniques?

what is AI?

- According to the father of Artificial Intelligence, John McCarthy, it is "The Science and engineering of making intelligent machines, especially intelligent computer programs".
- Artificial Intelligence is a way of making a computer, a computer-controlled robot, or a software think intelligently, in the similar manner the intelligent humans think.

AI is accomplished by studying how human brain thinks, how humans learn, decide, and work while trying to solve the problem, and then using the outcomes of this study as a basis of developing intelligent software and systems.

- Machine Learning (ML): ML is an application of AI that provides systems the ability to automatically learn and improve from experience without being explicitly programmed.
- ML focuses on the development of computer programs that can access data and use it learn for themselves.
- Deep Learning (DL): DL is a part of broader family of ML methods based on learning data representations as opposed to task specific algorithms.
- Learning can be supervised, semi-supervised and unsupervised
- Natural Language Processing (NLP): NLP is the ability of a computer program to understand human language as it is spoken. NLP is a component of AI.

: Expert System :- is a computer system that emulates the decision making ability of human expert.

What is AI Technique :-

AI techniques encompass a wide range of methods and approaches used to develop intelligent systems. Here are some of the key techniques :

1. Machine Learning (ML)
   - Supervised Learning: Training models on labeled data to make predictions to classify new data. Examples include linear regression, decision trees, and neural networks.
   - Unsupervised Learning: finding patterns or structures in data without explicit labels. Techniques include clustering (like k-means) and dimensionality reduction (like PCA).
   - Reinforcement Learning: Training agents to make sequences of decisions by rewarding desirable actions. Used in areas like robotics and game playing.

2. Deep Learning (DL):
   - Neural Networks: A type of ML model inspired by the human brain, consisting of layers of interconnected nodes (neurons).
   - Convolutional Neural Networks (CNNs) specialized for processing grid-like data such as images.
   - Recurrent Neural Networks (RNNs): Designed for sequential data, such as time series or text.
   - Transformers: A deep learning model designed for handling sequential data but more powerful than RNNs used in tasks like language translation and text generation.

3. Natural Language Processing (NLP):

- Text classification: categorizing text into predefined categories.
- Sentiment Analysis: Determining the sentiment expressed in a text.
- Machine Translation: Automatically translating text from one language to another.
- Text Generation: Creating coherent and contextually relevant text.

4. Robotics & Automation:

Automation aims to enable machines to perform boring, repetative jobs, increasing productivity and delivering more effective, efficient, and affordable results. To automate processes, many business employ machine learning, artificial neural, and graphs.

By leveraging the CAPTCHA technique, this automation can avoid fraud problems during online payments.

Robotic process automation is designed to carry out high-volume, repetitive jobs while being capable of adapting to changing conditions.

5. Computer vision:

- Object Detection: Identifying and locating objects in images or videos.
- Image Segmentation: Dividing an image into meaningful parts or segments.
- Face Recognition: Identifying or verifying a person from an image or video.

2) Explain breifly about the goals of AI, its advantages and disadvantages.

Goals of AI

- To create Expert Systems :- The systems which exhibit intelligent behavior, learn, demonstrate, explain, and advice its users.
- To implement Human Intelligence in Machines:- Creating systems that understand, think, learn, and behave like humans :- It makes you feel awesome when you think about a machine that feels and that expresses the emotions.
- Enhancing efficiency and Automation:- One of the primary goals of AI is to streamline processes and enhance efficiency across industries.
- Reasoning :- It is also one of important goals. There are different kinds of reasoning which include codification of relation in ideas, the transition between the sets of facts that can be interpreted by the computer system, and others.
- Decision-Making and predictive Analytics :-
  AI strives to revolutionize decision-making by leveraging data to provide insights that human might not readily discern.
- personalization and user experience :-
  As AI systems become more sophisticated, they aim to provide personalized experiences for users.

- Creativity and Innovation

Contrary to the belief that AI could stifle human creativity, AI aims to augment human creativity and innovation.

- Healthcare Advancements:

The healthcare industry stands to gain immensely from AI advancements. AI algorithms can analyze medical images, diagnose diseases, and predict patient outcomes.

Advantages:

1. Efficiency and productivity:- one of the most significant advantages of AI is its ability to perform tasks with remarkable efficiency and accuracy.
2. Enhance Decision Making:- AI can analyze large datasets to identify patterns and trends that may not be apparent to human analysts.
3. personalization and customer experience:- AI enables businesses to provide personalized experiences to customers.
4. Innovation and Development:- AI drives innovation by enabling the development of new products, services, and solutions.
5. Risk Reduction:- In hazardous environments, AI can perform dangerous tasks, reducing the risk to human workers.

Disadvantages

1. Job Displacement: The automation of tasks through AI can lead to job displacement in certain sectors. While AI creates new job opportunities, it also renders some roles obsolete, leading to workforce displacement and economic disruption.

2. High Implementation costs: Developing and implementing AI systems require significant investment.

3. Ethical and privacy concerns: AI systems can pose ethical and privacy challenges. Issues such as data privacy, surveillance, and the potential for biased decision-making are critical concerns that need to be addressed.

4. Dependency and Reliability: Over-reliance on AI can lead to problems if the systems fail or produce incorrect results.

5. Lack of emotional intelligence: AI lacks the ability to understand and respond to human emotions efficiently

---

3) Explain briefly about State Space Search with an example.

State Space Search

- In AI a state space consists of the following elements,

1. A (possibly infinite) set of states

a) Out of the possible states, one state represent state that is the initial state of the problem

b) Each state represents some significant configuration so that it is reachable from the start state

c) out of the possible states, some states may be goal states (solutions).

2. Set of Rules,

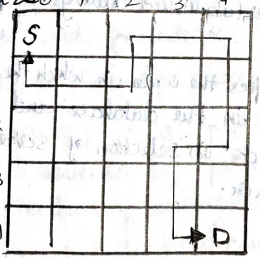
a) Applying a rule to the current state, transforms it to another or a new state in the state space.

b) All operators may not be applicable to all states in the state space.

- State spaces are used extensively in AI to represent and solve problems.

Example

Maze



A maze problem can be represented as a state-space

- Each state represents "where you are" that is the current position in the maze
- The start state or initial state represents your starting position.
- The goal state represents the exit from the maze
- Rules (for a rectangular maze) are: move north, move south, move east, and move west
- Each rules takes you to a new state (maze location)
- Rules may not always apply, because of walls in the maze.

4) Explain in detail about production systems and its applications.

Production systems:

- Production systems provide appropriate structures for performing and describing search processes.
- A production system has four basic components as enumerated below.

1. A set of rules each consisting of a left side that determines the applicability of the rule and a right side that describes the operation to be performed if the rule is applied.
2. A database of current facts established during the process of inference.
3. A control strategy that specifies the order in which the rules will be compared with facts in the database and also specifies how to resolve conflicts in selection of several rules or selection of more facts.
4. A rule firing module.

The production rules operate on the knowledge database.

- Each rule has a precondition - that is either satisfied or not by the knowledge database
- If the precondition is satisfied, the rule can be applied. Application of the rule changes the knowledge database.
- The control system choses which applicable rule should be applied and ceases computation when a termination condition on the knowledge database is satisfied.

Applications

production systems are widely used in various domains due to their flexibility and ability to encode complex knowledge in a structured manner. Some notable applications include:

1. Expert Systems:
   - Medical Diagnosis: Production systems are used to create expert systems that assist doctors in diagnosing diseases by matching symptoms (conditions) to possible diseases (actions).
   - Legal Reasoning: Expert systems can analyze legal cases and provide advice based on established legal rules and precedents.

2. Natural Language processing (NLP):
   - Grammar parsing: production systems can be used to parse sentences by applying grammatical rules to identify the structure of sentences.
   - Chatbots: Rule-based chatbots used production systems to generate responses based on user input by matching keywords or patterns.

3. Robotics:
   - Behavior control: in robotics, production systems can control robot behavior by selecting actions based on sensor input and environmental conditions.

4. Game AI:
   - Decision making: In games, production systems can be used to model decision-making processes of non-player characters (NPCs) allowing them to react to player actions and

environmental changes.

5. Automated planning:
   - Task scheduling: production systems can assist in scheduling tasks by applying rules that prioritize certain activities based on constraints and goals.
   - Resource Allocation: In complex systems like manufacturing or logistics, production systems can optimize resource allocation by applying rules that balances efficiency and cost.

---

5) Explain the issues in the design of search programs.

- Each search process can be considered to be a tree traversal.
- Designing effective search programs in AI involves addressing several key issues to ensure that they are efficient, scalable, and applicable to variety of problems. Here are some of the primary challenges:

1. Problem Definition and State Space Representation

→ Defining the problem:

- A precise definition of the problem, including the initial state, goal state, and set of possible actions. its crucial for the search program. Ambiguities in problem definition can lead to inefficient searches or incorrect solutions.

-> State Space Representation:

-> The way the state space is represented impacts the efficiency of the search. A well defined state space helps in systematically exploring possible solutions. complex problems require sophisticated representations to manage the vast number of potential states.

2. Algorithm Selection and performance:

-> choice of search Algorithm:

selecting the appropriate search algorithm (eg., Breath-first search, Depth-first search, A* Dijkstra's algorithm) is critical. Each algorithm has its own trade-offs in terms of time complexity, space complexity, completeness, and optimality.

-> Heuristics:

Informed search algorithms like A* relay on heuristics to guide the search process. Designing effective heuristics that are both admissible and consistent is a significant challenges as they impact that efficiency and accuracy of the search.

---

6) Explain in details about un-informed search, BFS, and DFS with an example.

un-informed search:

Uninformed search, also known as blind search, is a search algorithm that explores a problem space without any specific knowledge or informed about the problem other than the initial state and the possible action to take.

These algorithms are typically less efficient than informed search techniques algorithms but can be useful in certain scenarios or as a basis for more advanced search techniques.

Type of uniformed search Algorithms:

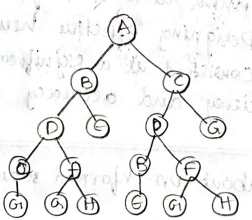
→ Depth first search
→ Breath first search.

1) Breath first search:

→ consider the state space of a problem that takes the form of a tree. Now, if we search the goal along each breath of the tree, starting from the root and continuing up to the largest depth, we call it breath first search.

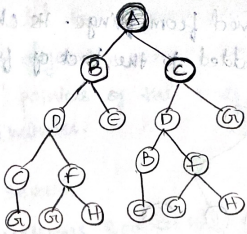
Example of BFS illustrated:

Step1: initially fringe contains only one node corresponding to the source state A



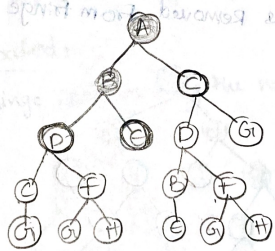
FRINGE: A

Step2: A is removed from fringe. the node is expanded and its children B and C are generated. They are placed at the back of fringe.



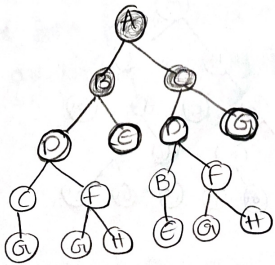
fringe: B C

Step3: Node B is removed from fringe and is expanded. Its children D, E are generated and put at the back of fringe.



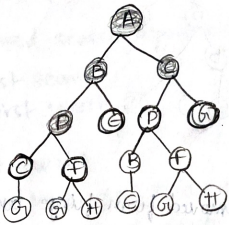
fringe: C D E

Step4: Node C is removed from fringe and is expanded. Its children D and G are added to the back of fringe



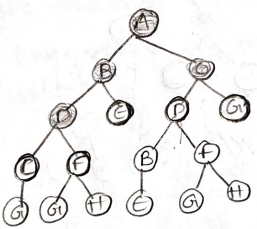
fringe: D E D G

Step5: Node D is removed from fringe. Its children Cand F are geneated and added to the back of fringe



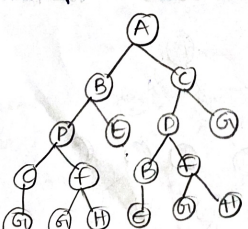
fringe: E D G C F

Step 6: Node E is Removed from fringe. It has no children



fringe: D G C F

Step 7: D is expanded; B and F are put in open.



fringe: G C F B F

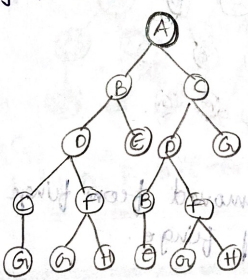
Step8: G_1 is selected for expansion. It is found to be a goal node. So the algorithm returns the path A C G_1 by following the parent pointers of the node corresponding to G_1. The algorithm terminates.

Depth First - Search

We may sometimes search the goal along the largest depth of the tree and move up only when further traversal along the depth is not possible, we then attempt to find alternative offspring of the parent of the node (state) last visited. This is called depth first - search.

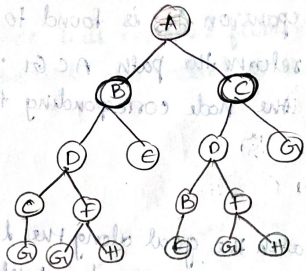
Example :- Illustrated :-

Step1: Initially fringe contains only the node for A.



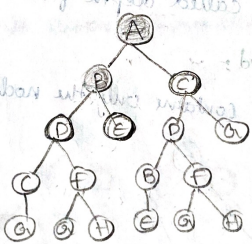
fringe: A

Step2: A is removed from fring. A is expanded and its children B and C are put in front of fringe.



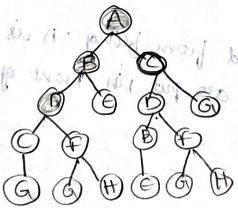
fringe: B C

Step3: Node B is removed from fringe, and its children D and E are pushed in front of fringe.



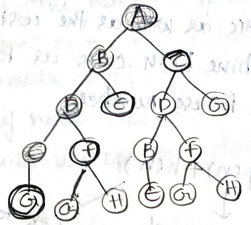
fringe: D E C

Step4: Node D is removed from fringe. C and F are pushed in front of fringe.



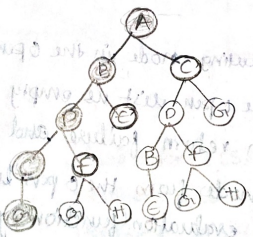
fringe: C F E C

Step 5:- Node C is removed from fringe. Its child G is pushed in front of fringe



fringe : G F E C

Step 6:- Node G is expanded and found to be a goal state



fringe : G F E C

the solution path A - B - D - C - G is returned and the algorithm terminates.

---

7) Explain in detail about A* Algorithm with an example

→ It is a searching algorithm is used to find the shortest path between an initial and a final point.

→ It is a handy algorithm that is often used for map traversal to find the shortest path to be taken.

→ This algorithm can take the path with the least cost, and find the best route in terms of distance and time.

A* Algorithm

→ Steps:

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.

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

- $f(n)$: Estimated cost of the cheapest solutions
- $g(n)$: Cost to reach node n from start state
- $h(n)$: cost to reach from node n to goal state.

Algorithm

→ Step 1:- Place the starting node in the open list

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

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

→ Step 4: Expand node n and generate all of its successors, and put n into the closed list. for each successor n', check whether n' already in the open or closed list, if not then compute function for n' and place in to open list.

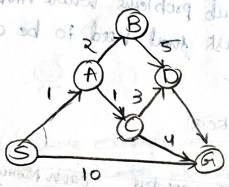
→ Step 5:- Else if node n' is already in open and closed, then it should be attached to the back pointer which reflects the lowest g(n') value.

→ Step 6:- Return to Step 2.

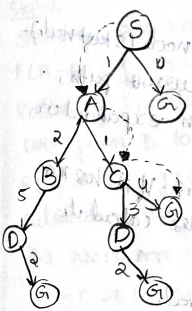
Example:-

→ In this example, we will traverse the given graph using the A* algorithm. The heuristic value of all states is given in the below table so we will calculate the f(n) of each state using the formula f(n) = g(n) + h(n) where g(n) is the cost to reach any node start state.

→ Here we will use open and closed list.



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



Initialization: {(S, 5)}

Iteration 1: {(S→A, 4), (S→G, 10)}

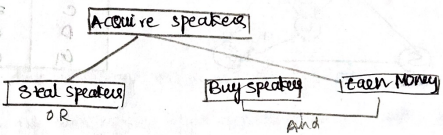
Iteration 2: {(S→A→C, 4), (S→A→B, 7), (S→G, 10)}

Iteration 3: {(S→A→C→G, 6), (S→A→C→D, 11), (S→A→B, 7), (S→G, 10)}

Iteration 4:- Will give the final result, as S→A→C→G it provides the optimal path with cost 6.

8) Explain in detail about AO* Algorithm with an example

* AO* algorithm is a best first search algorithm.
* AO* algorithm uses the concept of AND-OR graphs to decompose any complex problem given into smaller set of problems which are further solved
* AND-OR graphs are specialized graphs that are used in problems that can be broken down into sub problems where AND side of the graph represent a set of task that need to be done to achieve the main goal.

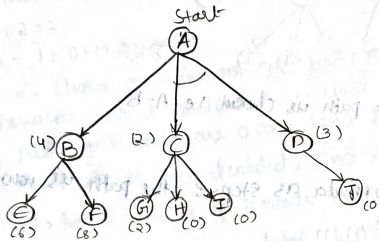


Algorithm:

→ Step 1:- Create an initial graph with a signal node (start node).
Step 2:- Traverse the graph following the current path, accumulating node that has not yet been expanded or solved.
Step 3:- Select any of these nodes and explore it. If it has no successors then call this value - FUTILITY else calculate f(n) for each of the successors.
Step 4:- If f'(n) = 0, then Mark the node as solved.
Step 5:- Change the value of f(n) for the newly created node to reflect its successors by back propagation.
Step 6:- Wherever possible use the most promising routes, If a node is marked as solved then mark the parent node as solved.

Step 7:- If the starting node is solved or value is greater than futility then stop else repeat from step 2.

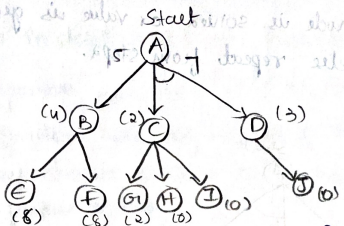
Example



Here, in the above example all numbers in brackets are the heuristic value i.e h(n). Each edge is considered to have a value of 1 by default.

Step 1

Starting from node A, we first calculate the best route path.
$f(A-B) = g(B) + h(B) = 1+4 = 5$ where 1 is the default cost value of travelling from A and B and 4 is the estimated cost from B to Goal state.
$f(A-C-D) = g(C) + h(C) + g(D) + h(D) = 1+2+1+3 = 7$, here we are calculating the path cost as both C and D because they have the AND-Arc, The default cost value of travelling from A-C is 1, and from A-D is 1, but the heuristic value given for C and D are 2 and 3 respectively hence making the cost as 7.



The minimum cost path is chosen i.e. A-B.

Step 2:

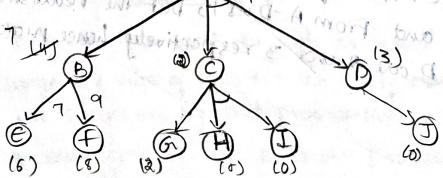
using the same formula as step-1, the path is now calculated from the B node,

$f(B-E) = 1+6 = 7$

$f(B-F) = 1+8 = 9$

Hence, the B-E path lesser cost. Now the heuristics have to be updated since there is a difference between actual and heuristic value of B. The minimum cost path is chosen and is updated as the heuristic, in our case the value is 7. And because of change in heuristic of B there is also change in heuristic of A which is to be calculated again.

$f(A-B) = g(B) + \text{updated } (h(B)) = 1+7 = 8$



Step-3

Comparing path of f(A-B) and f(A-C-D) it is seen that f(A-C-D) is smaller. Hence f(A-C-D) needs to be explored. Now the current node becomes C node and the cost of the path is calculated,

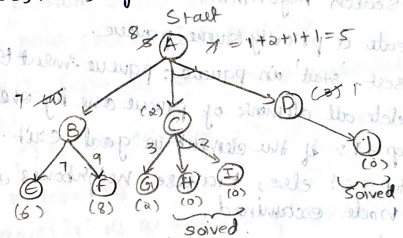
f(C-G) = 1+2 = 3

f(C-H-I) = 1+0+1+0 = 2

Change in heuristic since it matches the actual cost. Heuristic of path of H and I are 0 and hence they are solved, but path A-D also needs to be calculated, since it has an AND-arc.

f(D-J) = 1+0 = 1, hence heuristic of D needs to be updated.

f(A-C-D) = g(C)+h(C)+g(D)+updated (Ch(D)) = 1+2+1+1 = 5



As we can see that the solved path is f(A-C-D).

---

9) Explain in detail about Best-first Search with an example.

- If we consider searching as a form of a traversal in graph, an uniformed search algorithm would blindly traverse to the next node in a given manner without considering the cost associated with that step.

- An informed search, like BFS, on the other hand, would use an evaluation function to decide which among the various available nodes is the most promising (or 'best') before traversing to that node.

- BFS uses the concept of priority queue and heuristic search. To search the graph space, the BFS method uses two lists for tracking the traversal.
- An 'open' list that keeps track of the current 'immediate' nodes available for traversal and a 'closed' list that keeps track of the nodes already traversed.

Best first search Algorithm:

Step 1: create a priority queue pqueue.

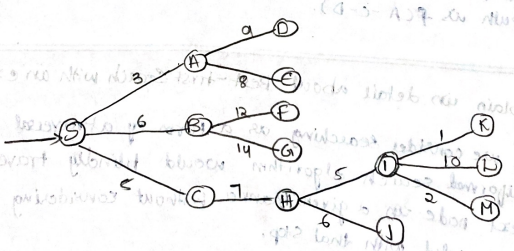
Step 2: insert 'start' in pqueue: pqueue.insert(start)

Step 3: delete all elements of pqueue one by one

- Step 3.1: if the element is goal, exit.
- Step 3.2: else, traverse, neighbours and mark the node examined.

Step 4: end.

· Example



- we start from source 'S' and search for goal 'I' using given costs and Best First search.
- pq initially contains S
  - we removes from and process unvisited neighbors of S to pq.
    - pq now contains {A, C, B} (C is put before B because C has lesser cost)
- we remove A from pq and process unvisited neighbors of A to pq.
  - pq now contains {C, B, E, D}
- we remove C from pq and process neighbors of C to pq.
  - pq now contains {B, H, E, D}
- we remove B from pq and process unvisited neighbors of B to pq.
  - pq now contains {H, E, D, F, G}
- we remove H from pq.
- since our goal "I" is a neighbor of H, we return.

---

10) Explain in detail about Hill climbing with an example.

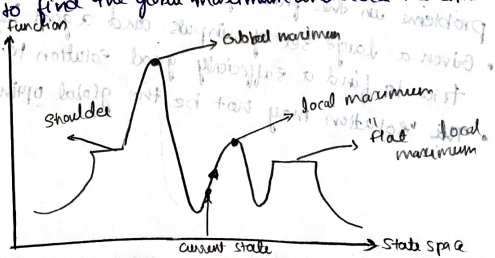
- Hill climbing is a heuristic search used for mathematical optimization problems in the field of Artificial intelligence.
- Given a large set of inputs and a good heuristic function, it tries to find a sufficiently good solution to the problem.
- This solution may not be the global optimal maximum.

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.
- Hill climbing algorithm is a technique which is used for optimizing the mathematical problems.
- 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.

State-space Diagram for Hill Climbing:

- The state-space landscape is a graphical representation of the hill-climbing algorithm.
- On Y-axis we have taken the function which can be an objective function or cost function, and state-space on the X-axis.
- If the function on Y-axis is cost then, the goal of search is to find the global maximum and local minimum.
- If the function of Y-axis is objective function, then the goal of the search is to find the global maximum and local maximum.



- Local Maximum: Local maximum is a state which is better than its neighbor states, but there is also another state which is higher than it.
- Global Maximum: Global maximum is the best possible state of state space landscape. It has the highest value of objective function.
- Current state: It is a state in a landscape diagram where an agent is currently present.
- Flat local maximum: It is a flat space in the landscape where all the neighbor states of current state have the same value.
- Shoulder: It is a plateau region which has an uphill edge.

Example:-

Problem: Imagine you are trying to find the highest peak in a landscape using a map with elevation data. The map represents the height of different points, and your goal is to find the highest point.

Steps:

1. Initial State: Start at an arbitrary point on the map. Suppose this point has an elevation of 100 meters.
2. Generate Successors: From your current point, you can move to neighboring points (eg, north, south, east, west). Suppose the neighboring points have elevations of 120 meters (north), 95 meters (south), 110 meters (east), and 105 meters (west).
3. Evaluate: Compare the elevation of these neighboring points to the current point. The point to the north with 120 meters has the highest elevation.
4. Move: Move to the point with the highest elevation (120 meters).

5. Repeat: From the new point, generate successors and evaluate them. Continue moving to the neighboring point with the highest elevation until you reach a point where all neighboring points have lower elevations or the same elevation (a local maximum).

6. Termination: If you reach a point where no neighboring points have a higher elevation, the search terminates. This point is either a local maximum or if you are fortunate, the global maximum.