

Search Algorithms

ACCA6011

Dr. Seena Joseph

Search Algorithms

Terminology

- **Agent:** it's the human, model, or algorithm that interacts with its environment.
- **State:** a particular environment or set of elements that represent the problem at a given moment.
- **State space:** the set of all possible reachable states.
- **Goal state:** the final state, where certain conditions are met.
- **Action:** a function or decision the agent must make in a given state to move to the next one.
- **Example for search problem: Computer maps providing path finding.**
 - Find the best route from a current location to a destination
 - Best route ?
 - The shortest route (least distance route)
 - The quickest route
 - The route that uses the least energy
 - The-lowest cost route (cost in terms of money, time etc.)

Search Algorithms

- A search algorithm seek to answer question by exploring a defined domain.
 - Let's say that we want to look up someone's phone number.
- The domain or search space is the set of all possible answers to the question.
 - All the entries of the phone book.
- Example of searches
 - Timetable optimisation
 - Pathfinding
 - Puzzle searches
 - Game searches
 - Chess/Tic-Tac-Toe

Search Algorithms

- Searching – Finding a path to a goal in a graph
- Agent is provided with a problem with a description of recognizing a solution – Not an algorithm to solve it
- Human – find satisfying or good enough solutions but not optimal solutions.

Search Algorithms

- **State Space** – formulation of intelligent action is in terms of a state space
- State – contains all information necessary to predict the effects of an action and to determine whether a state satisfies the goal.
- State space searching assumes :
 - The agent has a perfect knowledge of the state space
 - At any time, it knows what state it is in; i.e the world is fully observable
 - The agent has a set of actions with known deterministic effects
 - The agent has a goal to achieve and can determine whether a state satisfies the goal.

Search Algorithms

- **State Space**

- A solution to a search problem is a sequence of actions that will get the agent from its current state to a state that satisfies its goal.
- A set of all possible states that an agent can be in.
- **Example** : Consider a video game played on a grid, where an agent can move up, down, left, or right one step as long as the move is not blocked by a wall. The agent has to collect four coins, $C_1, \dots, C_4$, where each coin starts at a known position. The agent uses one unit of fuel at each step, and cannot move if it has no fuel. It can get filled up (to 20 units of fuel) at the fuel station. In this case the state needs to take into account the position of the agent, the amount of fuel, and for each coin whether it has collected that coin. The state could be represented as the tuple :
 - $((x, y), \text{fuel}, c_1, c_2, c_3, c_4)$ -> Position in coordinates, fuel level, Status of coin collection;
 - $((5,8), 6, f, t, f, f)$? Position (?, ?) ; Fuel Level : ?; Coin collected ?

Search Algorithms

- State Space

Blocked space

- Initial state :

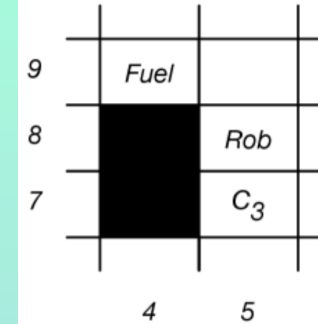
- Starts at position (0,0)
- Fuel is full of 20 units
- No coins have been collected

- Intermediate state

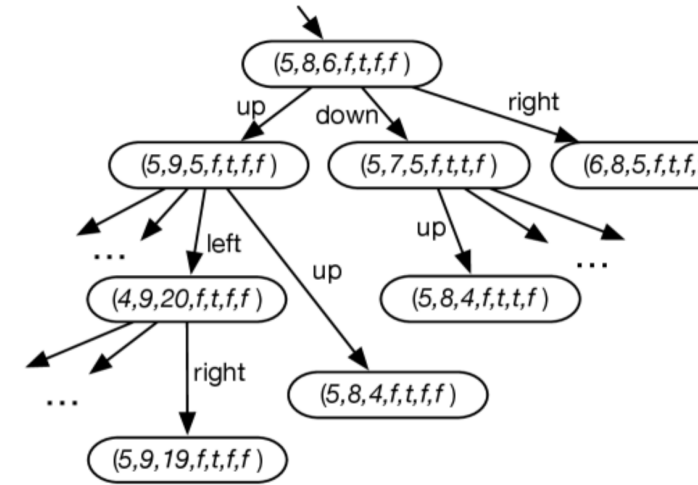
- ((5, 8), 6, f, t, f, f) position ?; fuel level ? Coin collection status

- Goal state

- ((6,7), 3, t,t,t,t)



(a)



(b)

Search Algorithms

- State Space
 - Size of the state space
 - Size of the grid * Fuel level * coin collection combination
 - M x N Grid; 0 to 20 (21); 2 state per coin (True or False) = 16 states)
 - Action
 - Move right
 - Move left
 - Move up
 - Move down
 - Refuel
 - State transition
 - Agent can move to an adjacent position if it has fuel
 - If agent is at the fuel station, it can refill
 - When the agent reaches the position of a coin, the coin is marked as True/collected

Search Algorithms

- **State Space**

- A **state space search** problem consists of :
 - A set of space
 - A distinguished state called the **start state**
 - A set of actions available to the agent for each state
 - An **action function** that given a state and an action, returns a new state
 - A **goal** specified as a Boolean function: true when state s satisfies the goal, goal state s ; goal(s)
 - A **criteria** that specifies the quality of an acceptable solution
- **Optimum solution**: a solution that is best according to some criteria

Search Algorithm

- **State space S** :all possible situation or configuration of the problem domain
 - All states reachable from initial by any sequence of actions
 - Each state is an abstract representation of an environment
 - The state space contain an initial state and a goal state
- **Operators (Action) A** : available actions
 - Something that the agent can choose
- **Search Tree**
 - A graph with no undirected loops; root node is the start state; set of children for each node consists of the states reachable by taking any action.
- **Search node**
 - a node in the search tree
- **Goal**
 - A state that the agent is trying to reach

Search Algorithm

- **Path:**
 - A sequence of states and operators
- **Path cost:**
 - The number associated with any path
 - Measures the quality of the path
 - Sum of costs of individual actions along the path
 - Smaller the better
- **Solution of a search problem**
 - A path from the initial stage to the goal stage
- **Optimal solution**
 - Any path with minimum cost
- **Goal Test**
 - Test to determine if the agent is at goal state

Search Algorithm

Search Algorithms – to explore search space

- Graph searching
 - Searching a graphs provides an appropriate abstract model of problem solving independent of a particular domain.
 - Directed Graph – consist of a set of nodes and a set of directed arcs between nodes. To find a path along these arcs from the start node to a goal node.
 - Node – represents states
 - Arc – represents action

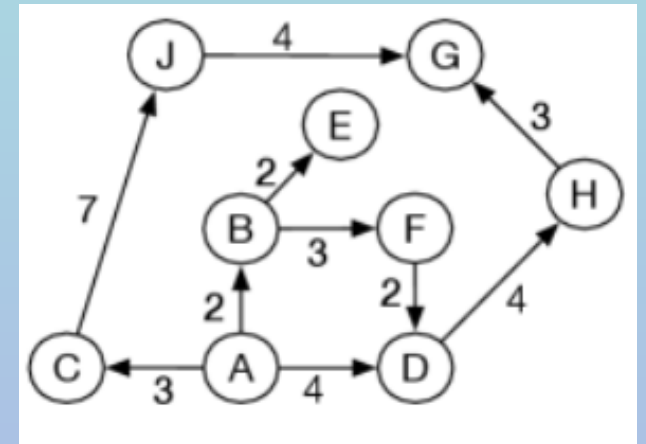


Figure 1

Search Algorithm

- Graph searching
 - Node – { A, B, C, D, E, F, G, H}
 - Arc – {(A, B), (A,D),(A,C), (C, J), (J,G), (B,E), (B,F), ...}
 - Node E has no neighbors
 - Node B has two neighbors
 - There are three paths from A to G
 - (A,D,H,G)
 - ...?
 -?
 - Which one of these is the optimal solution ?

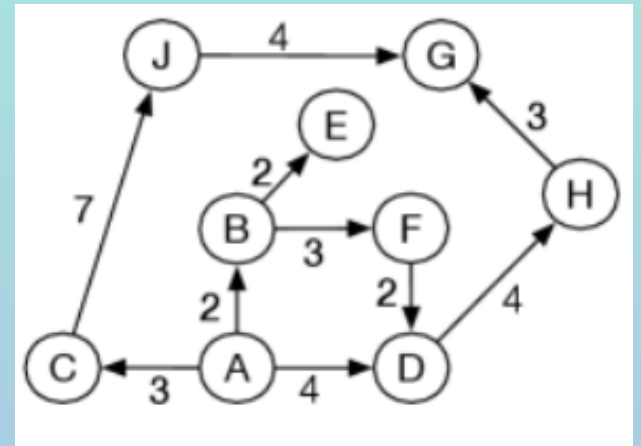


Figure 1

Search Algorithm

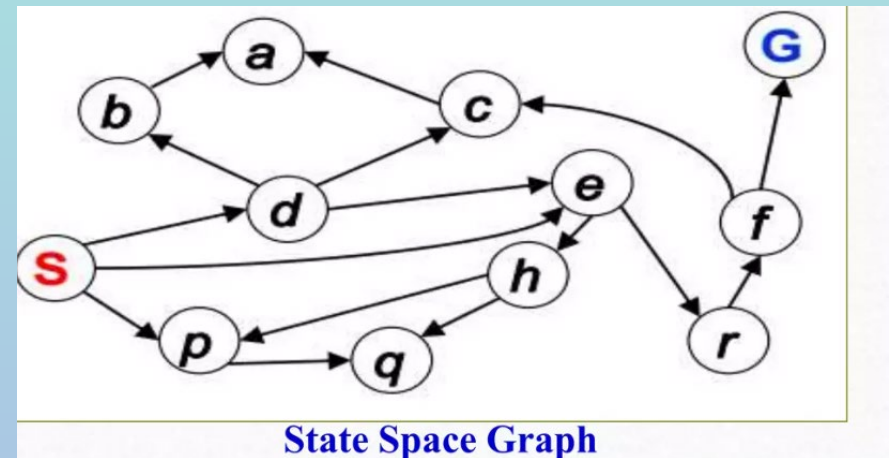
- **Graph searching**
 - **Cycle** – a non-empty path where the end node is the same as the start node.
 - i.e. $(n_0, n_0, \dots, n_k)$ such that $n_0 = n_k$.
 - **Directed Acyclic Graph (DAG)** – a directed graph without a cycle
 - Graph in Figure 1 is a DAG
 - **Tree** – A DAG where one node with no incoming arcs; all other nodes has exactly one incoming arc;
 - **Root** – node with no incoming arc;
 - **Leaf** – node with no outgoing arc
 - In a tree an arc goes from parent to child
 - **Forward branching factor** – number of outgoing arcs from the node
 - **Backward branching factor** – number of incoming arcs to the node
 - **What are the forward and backward branching factor for the nodes, A, B and E ?**

Search Algorithm

State Space Graph versus Search Tree

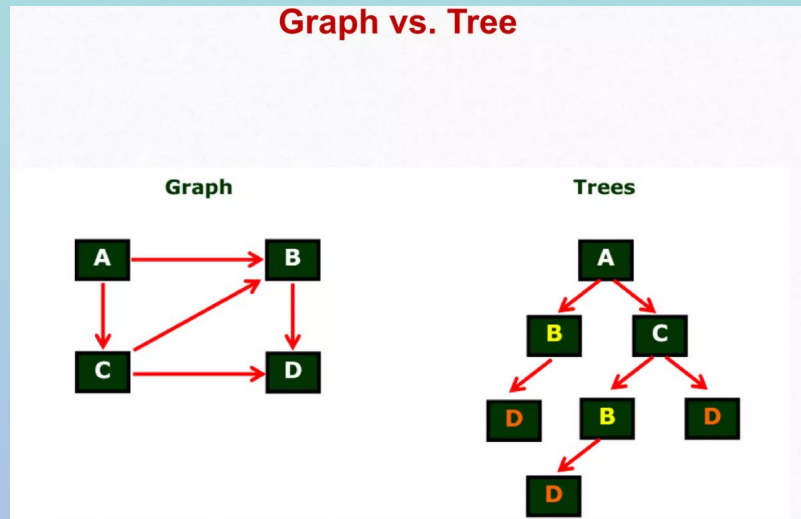
- **State Space Graph**

- Graph of states with arrows pointing to the successors
- A state is a node in which a system can be at any given time
- May contain a loop



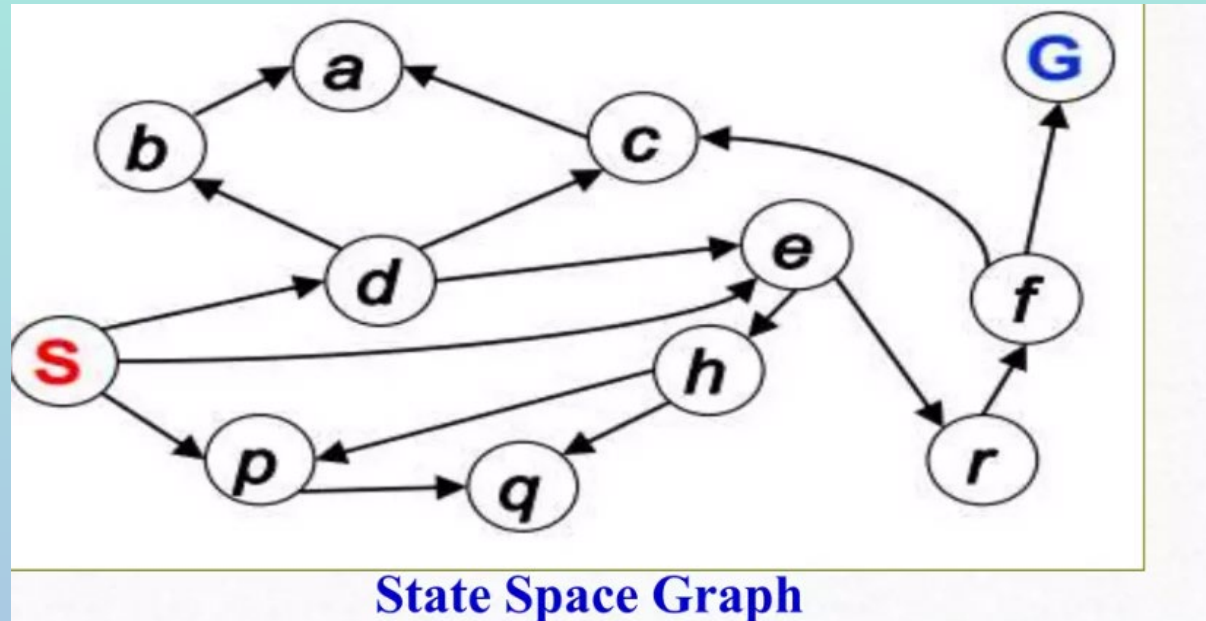
Search Algorithm

- State Space Tree
 - Tree is a special case of a graph
 - It is hierarchical in nature
 - It is a connected acyclic graph with a designated root node
 - The topmost node is the root node; all operations of the tree begin at the root node



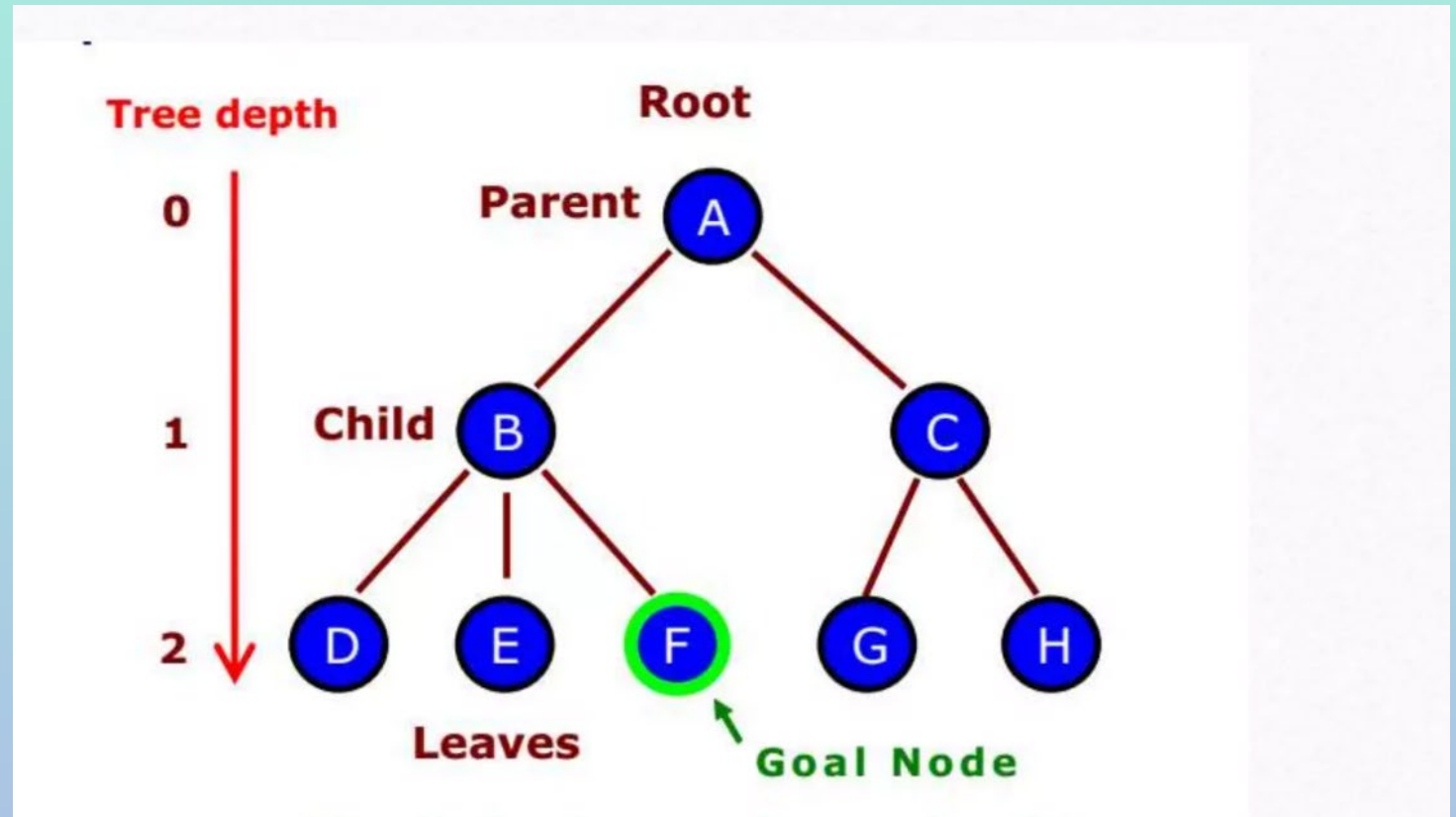
Graph Vs Tree

- What is the corresponding tree representation of the Graph below ?



Search Algorithm

- Search problem



Search Algorithms

A problem determine the graph, start node and the goal , not the which path to select from the frontier.

Search strategy - it defines the order in which paths are selected from the frontier.

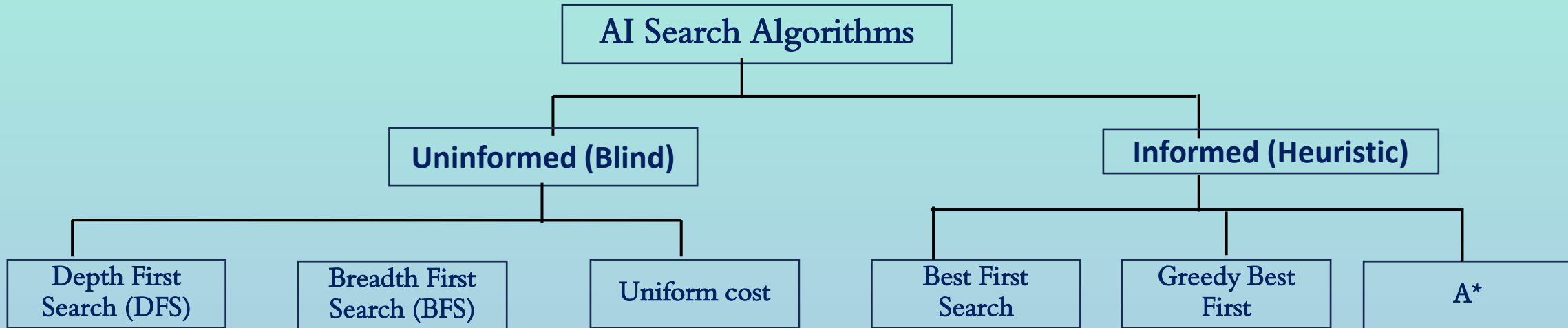
Information availability in search Algorithm

It refers to the knowledge a search algorithm has about the problem domain and the path to the goal.

Search Algorithms

Uninformed and Informed

This classification is based on the knowledge or heuristic



Search Algorithms

Uninformed Search Algorithm

- Explore problem space without using the heuristic information;
- Without any prior knowledge
- Based on the search path search and the information received during the search process
- Used for small problem space where the domain specific knowledge is unavailable or not useful.

Search Algorithms

Informed Search Algorithm

- Based on heuristics or domain specific knowledge
- Based on additional information such as estimated cost to reach the goal
- Likelihood of a path leading to the solution
- Generally outperform the uninformed search algorithms in terms of efficiency and effectiveness especially in complex problems.

Search Algorithms - Uninformed

Breadth-First Search (BFS)

- Explores all neighbor nodes at the present depth level before moving on to nodes at the next depth level.
- the highest layer of a decision tree is searched completely before proceeding to the next layer
- A queue (FIFO-First in First Out) data structure is used by BFS.
- The time complexity of Breadth-First Search (BFS) is commonly represented as $O(b^d)$ [Big O Notation]
- The space complexity of BFS is represented by $O(b^{d+1})$
 - Where b is the branching factor (average number of children for each node) and d represents the depth or distance from the root node.

Search Algorithms - Uninformed

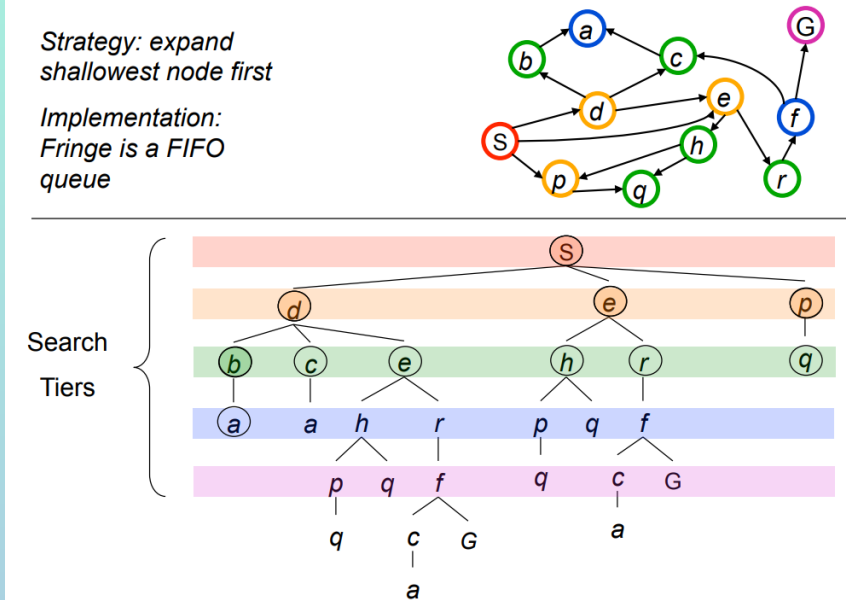
Breadth First Search (BFS)

- Imagine you're looking for a friend in a building:
- You check everyone on the ground floor first.
- Then go to the first floor, then the second, and so on.

Breadth First Search

Strategy: expand shallowest node first

Implementation: Fringe is a FIFO queue



- [BFS: Breadth First Search Implementation in Python - Pierian Training](#)

Search Algorithms - Uninformed

Depth First Search

- Expands the deepest node in the current frontier of the search tree
- Uses LIFO
- DFS is often preferred for its simplicity and lower space complexity compared to Breadth-First Search (BFS), especially in cases where memory usage is a concern.
- In each level minimum number of nodes are stored
- Space complexity is $O(bd)$
- Time complexity $O(b^d)$

Search Algorithms - Uninformed

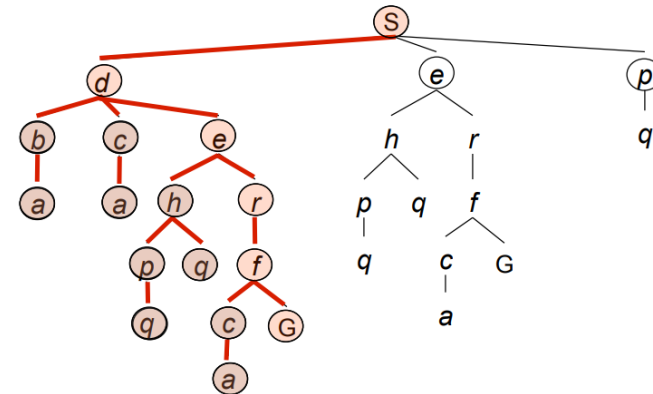
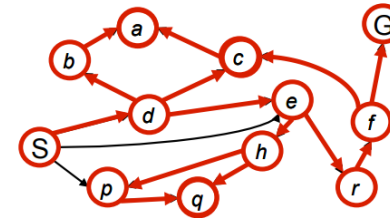
Depth First Search

- Imagine you're exploring a maze:
- You pick one path and follow it all the way.
- If it's a dead end, you **go back** and try another.

Depth First Search

Strategy: expand deepest node first

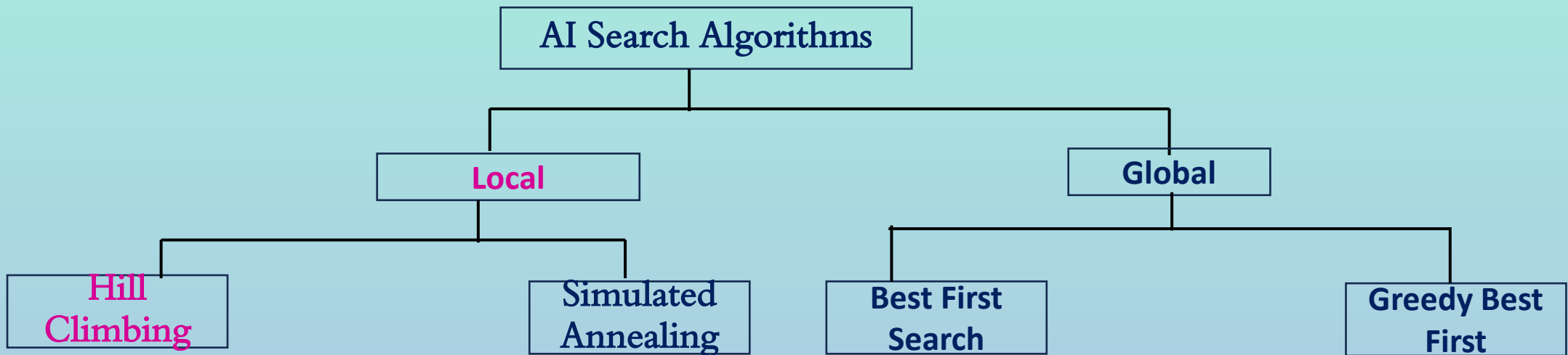
Implementation: Fringe is a LIFO stack



Search Algorithms

Local and Global Search Algorithm

The classification is based on the search space exploration



Search Algorithms

Global search explores the entire search space to find the best or guaranteed solution.

Local search explores only nearby options to quickly find a good (but not always best) solution.

Local Search Algorithms

- Local search is an algorithm that is used to find the best solution to a problem by starting with a random solution and then making small changes to it until it finds a better solution.
- Local search algorithm doesn't explore the entire search space

Search Algorithms

Local Search Algorithms

- General working principle
 1. Initialization – start with an initial solution (either randomly or through heuristic method)
 2. Evaluation – Evaluate the quality of the initial solution using an objective function
 3. Neighbor generation – Generate a set of neighboring solution by making small changes to the current state
 4. Selection – Choose one of the neighboring solutions based on a criterion such as improvement over objective function
 5. Termination – continue the process iteratively, moving to the selected neighboring solution and repeating steps 2 to 4 until a termination condition is met. This condition could be a maximum number of iteration, or reaching a predefined threshold or finding a satisfactory solution.

Search Algorithms

Global Search Algorithms

Global search algorithm aim to explore the entire solution space systematically to find the optimum solution.

Search Algorithms

Objective function - Mathematical function that evaluates and measures the quality or "fitness" of a solution in optimization problems.

The objective function is either maximized (for problems where higher values are better, e.g., profits) or minimized (for problems where lower values are better, e.g., costs, distances).

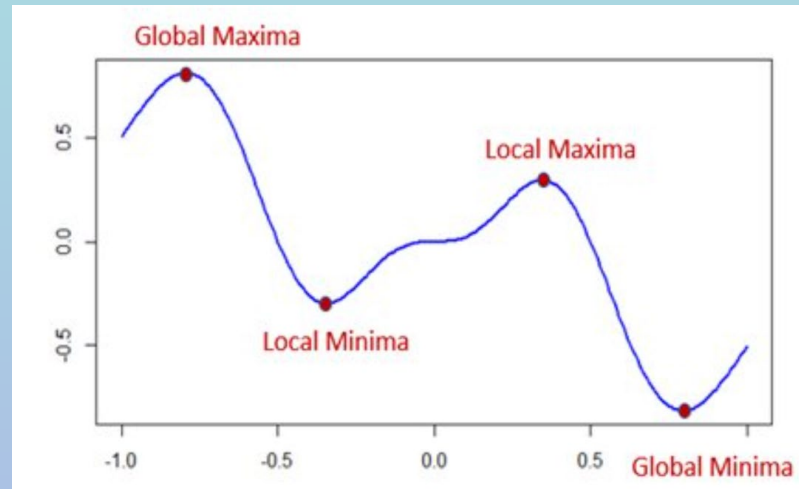
Search Algorithms

Local minimum - is a point where the function value is smaller than at nearby points, but possibly greater than at a distant point.

Global minimum - a point where the function value is smaller than at all other feasible points.

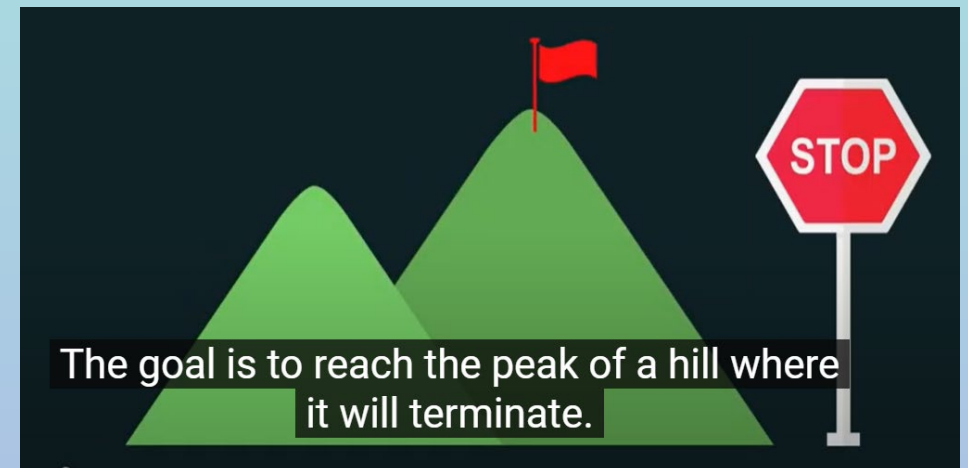
Local maximum - is a point where the function value is larger than at nearby points, but possibly smaller than at a distant point.

Global maximum - a point where the function value is greater than at all other feasible points.



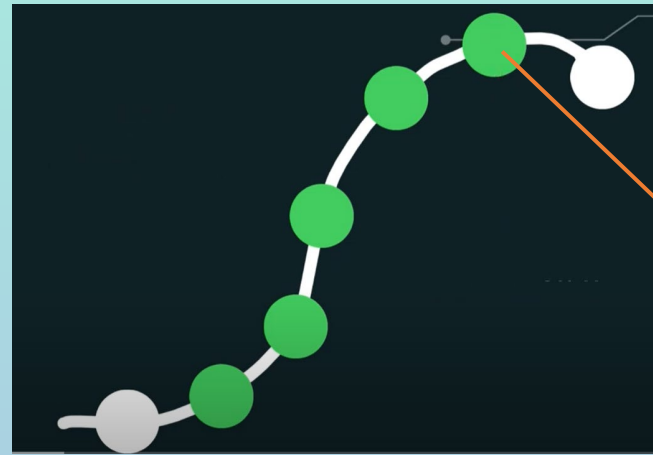
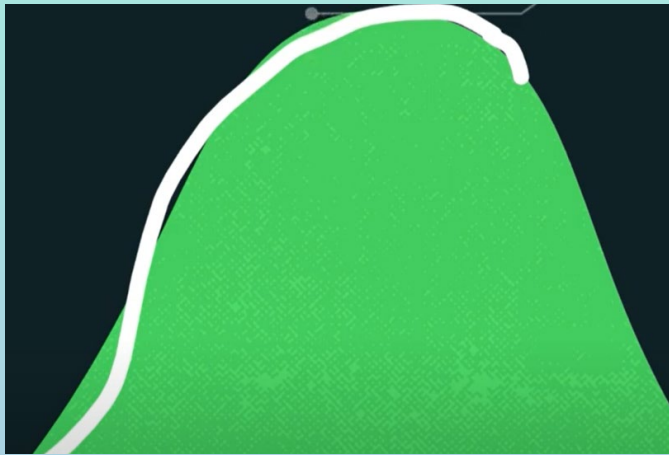
Search Algorithms - Local

- **Hill climbing** –an iterative algorithm that starts with an arbitrary solution to a problem and attempts to find a better solution by making an incremental changes to the current solution.
- Always choosing the neighboring solution with the highest improvement.
- It is a goal driven search technique that focus on the goal not the path to goal.
- It will terminate when it reaches the goal



Search Algorithms - Local

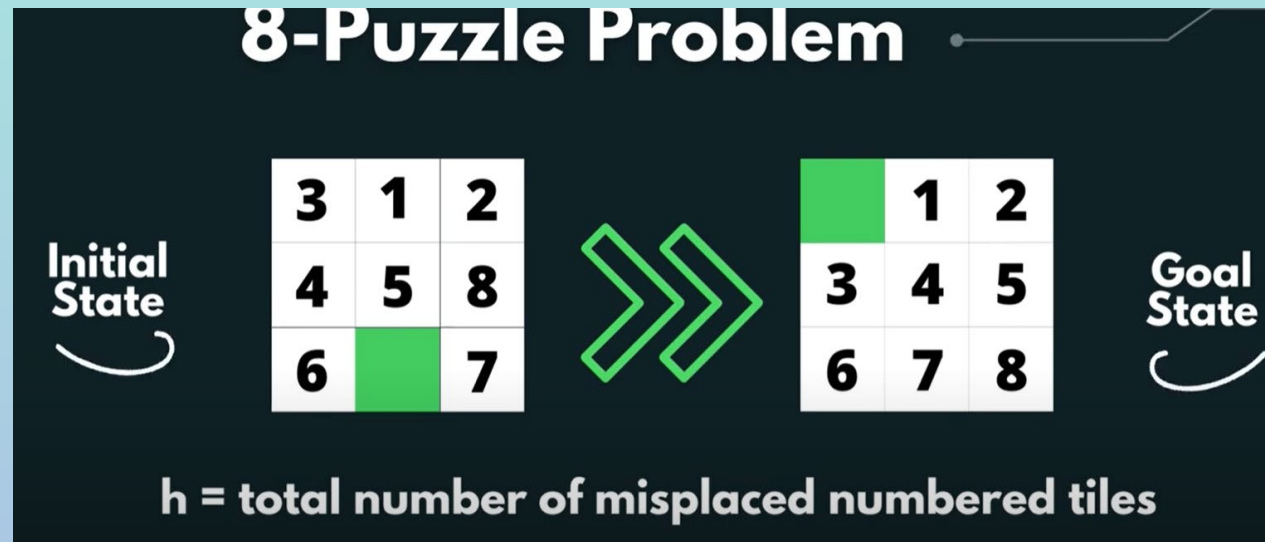
- Hill climbing



Solution State

Search Algorithms - Local

- Hill climbing - Example 8 Puzzle problem

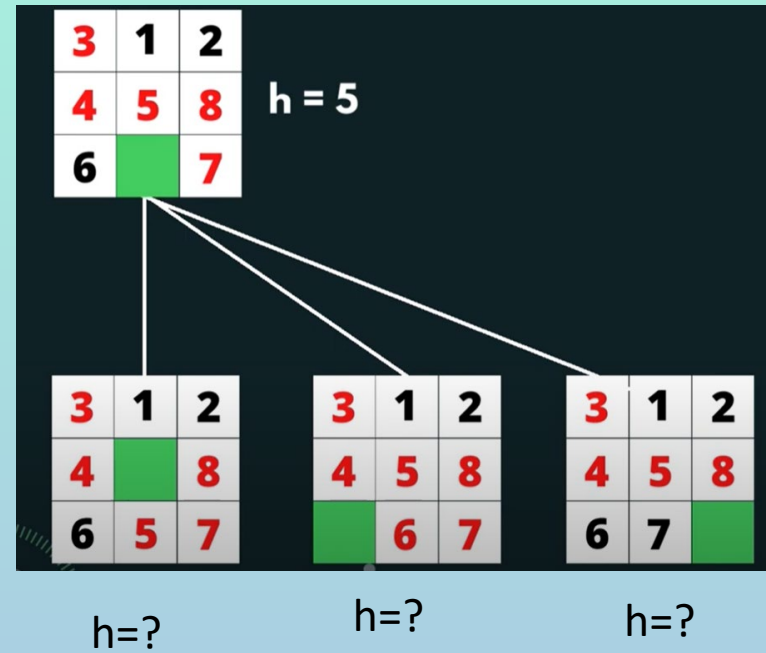


Search Algorithms - Local

- Hill climbing – Example 8 Puzzle problem

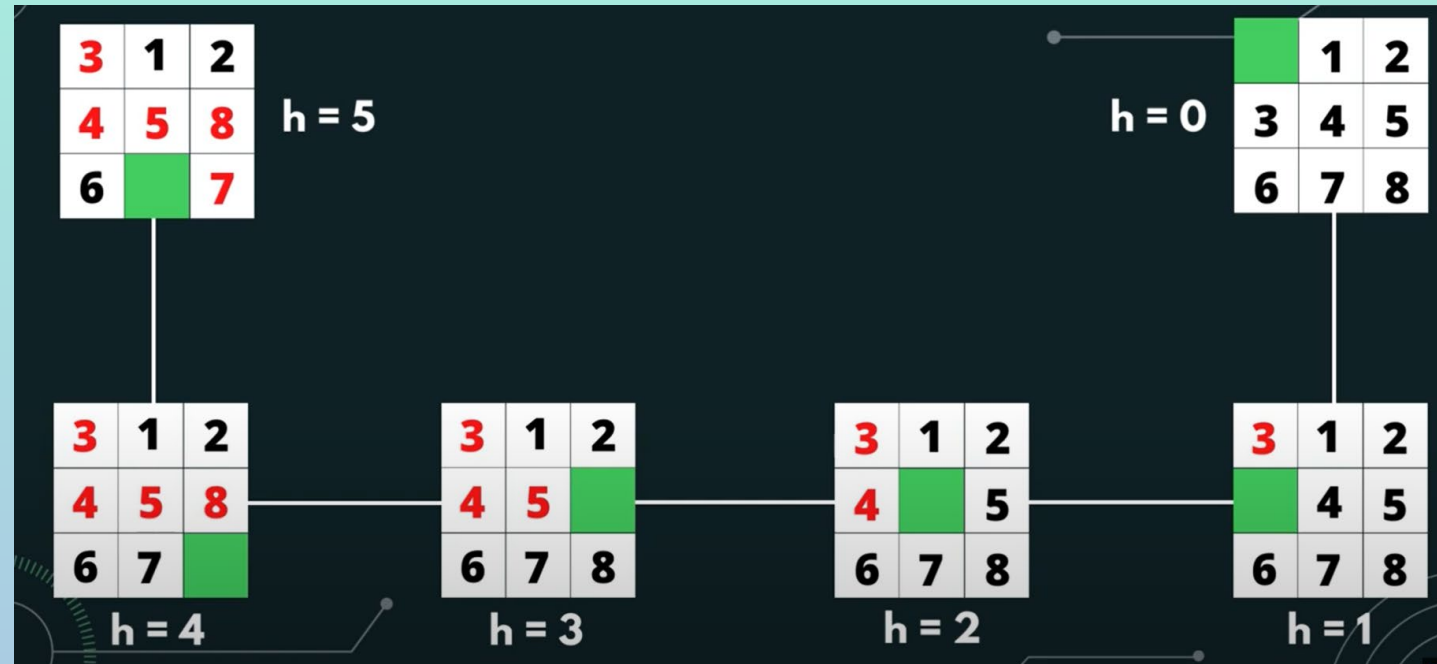
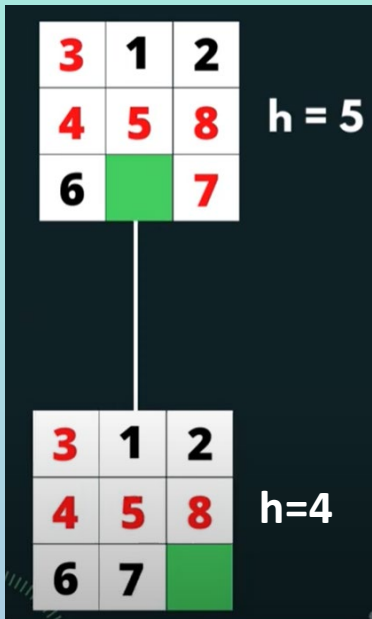
3	1	2
4	5	8
6		7

$h = 5$



Search Algorithms - Local

- Hill climbing – Example 8 Puzzle problem



Search Algorithms - Local

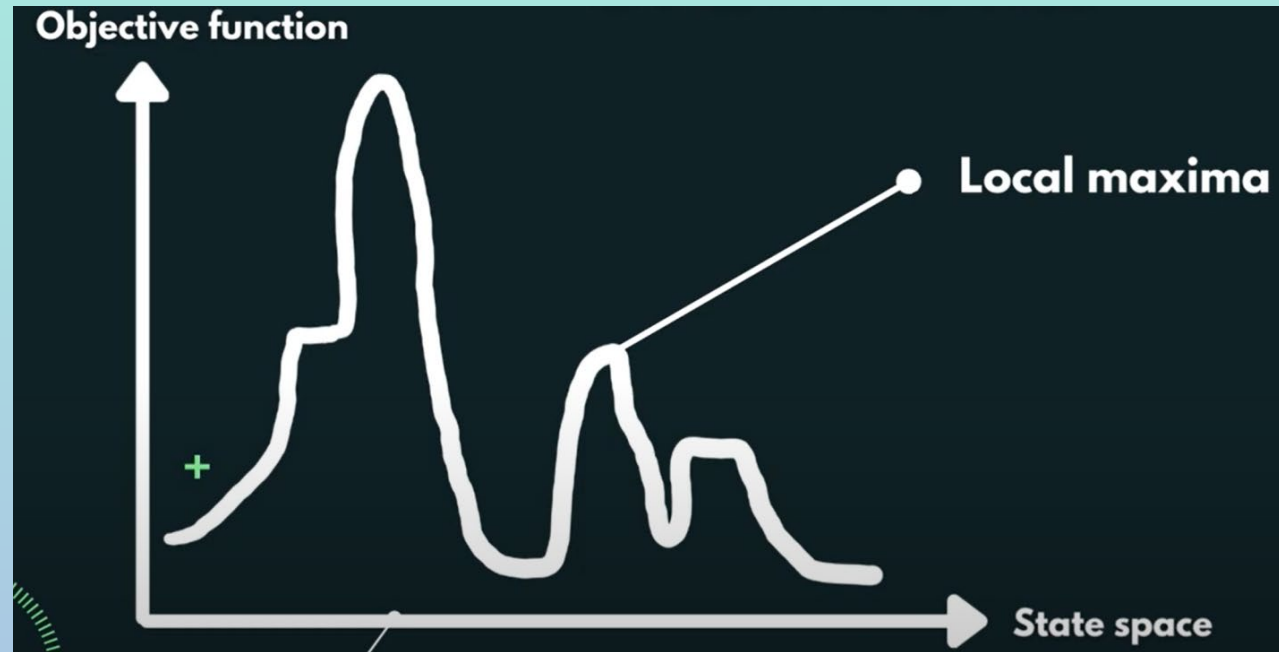
- Hill climbing –

1. Evaluate the initial state. If it is goal state then return it, otherwise continue with the initial state as the current state.
2. Loop until the solution is found or until there are no new operators to apply to the current state
 - a. Select an operator that not has been applied to current state and apply it to get the new state
 - b. Evaluate the new state
 - a. If it is a goal state then return it and terminate
 - b. If it is not a goal state, but better than then current state then make it as a the current state
 - c. If it is not better than the current state then continue in the loop

Search Algorithms - Local

- Hill climbing – Drawbacks

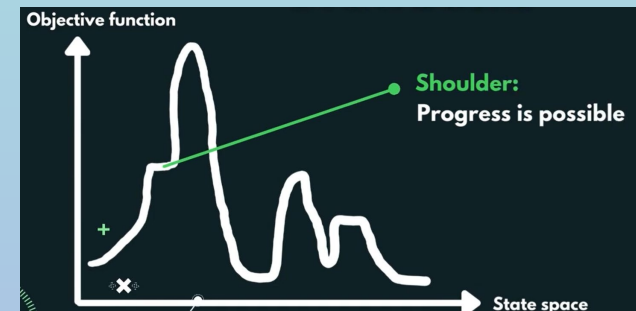
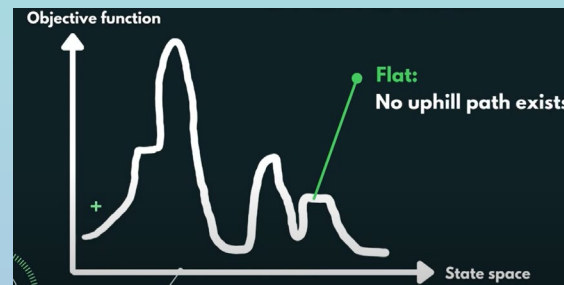
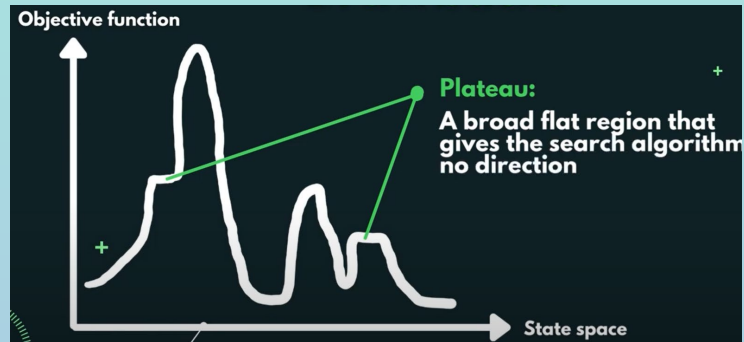
1. Since it does not backtrack and only looks at its neighbors, it might not find the highest peak in the search space, but only the highest peak nearby.



Search Algorithms - Local

- Hill climbing – Drawbacks

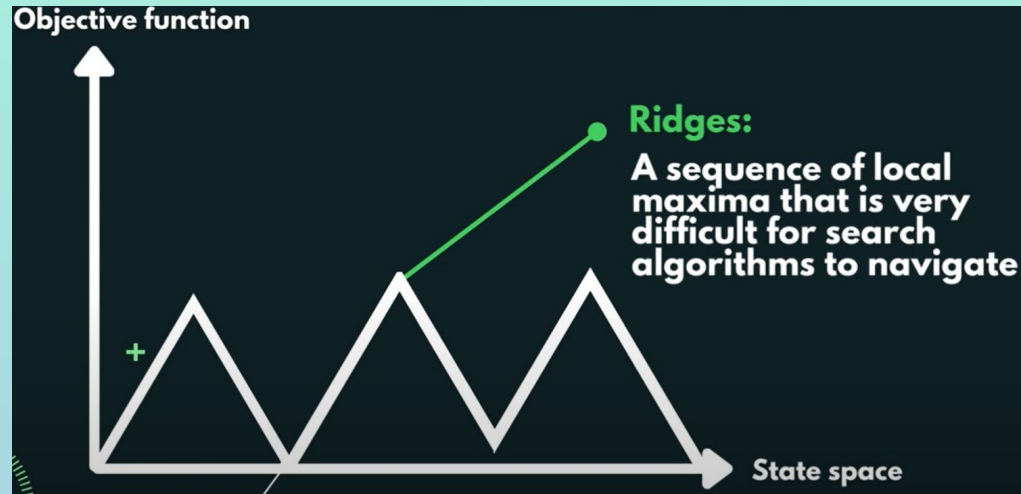
- Plateau – A flat region that gives the search algorithm no direction
 - It can be a flat local maximum, from which no uphill path exists or
 - Shoulder from which progress is possible



Search Algorithms - Local

- Hill climbing – Drawbacks

- Ridges – Peak of the ridge is followed by downward movement instead of upward movement.



- Dependency on Initial solution: The quality of the final solution can be highly dependent on the initial solution chosen.

Activity

- Application of hill climbing
- Simulated annealing
 - Self-study

Resources

- [AI | Search Algorithms | Hill Climbing | Codecademy](#)
- [Hill Climbing in Artificial Intelligence – GeeksforGeeks](#)
- [Local Search Algorithm in Artificial Intelligence – GeeksforGeeks](#)
- [AI Search Algorithms With Examples | by Pawara Siriwardhane, UG | Nerd For Tech | Medium](#)