**UNIT-1**

# Introduction to AI

* In today's world, technology is growing very fast, and we are getting in touch with different new technologies day by day.
* Here, one of the booming technologies of computer science is Artificial Intelligence which is ready to create a new revolution in the world by making intelligent machines.The Artificial Intelligence is now all around us. It is currently working with a variety of subfields, ranging from general to specific, such as self-driving cars, playing chess, proving theorems, playing music, Painting, etc.
* AI is one of the fascinating and universal fields of Computer science which has a great scope in future. AI holds a tendency to cause a machine to work as a human.
* Artificial Intelligence is composed of two words **Artificial** and **Intelligence**, where Artificial defines *"man-made,"* and intelligence defines *"thinking power"*, hence AI means *"a man-made thinking power."*

So, we can define AI as:

 "It is a branch of computer science by which we can create intelligent machines which can behave like a human, think like humans, and able to make decisions."

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. 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
4. 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

NOTES

# Agents in Artificial Intelligence

An AI system can be defined as the study of the rational agent and its environment. The agents sense the environment through sensors and act on their environment through actuators. An AI agent can have mental properties such as knowledge, belief, intention, etc.

## What is an Agent?

An agent can be anything that perceive its environment through sensors and act upon that environment through actuators. An Agent runs in the cycle of **perceiving**, **thinking**, and **acting**. An agent can be:

* **Human-Agent:** A human agent has eyes, ears, and other organs which work for sensors and hand, legs, vocal tract work for actuators.
* **Robotic Agent:** A robotic agent can have cameras, infrared range finder, NLP for sensors and various motors for actuators.
* **Software Agent:** Software agent can have keystrokes, file contents as sensory input and act on those inputs and display output on the screen.

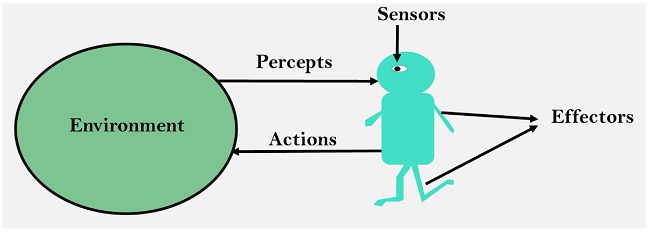
Hence the world around us is full of agents such as thermostat, cellphone, camera, and even we are also agents.

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.



## Intelligent Agents:

An intelligent agent is an autonomous entity which act upon an environment using sensors and actuators for achieving goals. An intelligent agent may learn from the environment to achieve their goals. A thermostat is an example of an intelligent agent.

Following are the main four rules for an AI agent:

* **Rule 1:** An AI agent must have the ability to perceive the environment.
* **Rule 2:** The observation must be used to make decisions.
* **Rule 3:** Decision should result in an action.
* **Rule 4:** The action taken by an AI agent must be a rational action.

Structure of an AI Agent

The task of AI is to design an agent program which implements the agent function. The structure of an intelligent agent is a combination of architecture and agent program. It can be viewed as:

1. Agent = Architecture + Agent program

Following are the main three terms involved in the structure of an AI agent:

**Architecture:** Architecture is machinery that an AI agent executes on.

**Agent Function:** Agent function is used to map a percept to an action.

1. f:P\* → A

**Agent program:** Agent program is an implementation of agent function. An agent program executes on the physical architecture to produce function f.

## PEAS Representation

PEAS is a type of model on which an AI agent works upon. When we define an AI agent or rational agent, then we can group its properties under PEAS representation model. It is made up of four words:

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

Here performance measure is the objective for the success of an agent's behavior.

### PEAS for self-driving cars:



Let's suppose a self-driving car then PEAS representation will be:

**Performance:** Safety, time, legal drive, comfort

**Environment:** Roads, other vehicles, road signs, pedestrian

**Actuators:** Steering, accelerator, brake, signal, horn

**Sensors:** Camera, GPS, speedometer, odometer, accelerometer, sonar.

## Example of Agents with their PEAS representation

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Agent** | **Performance measure** | **Environment** | **Actuators** | **Sensors** |
| **1. Medical Diagnose** | * Healthy patient * Minimized cost | * Patient * Hospital * Staff | * Tests * Treatments | Keyboard (Entry of symptoms) |
| **2. Vacuum Cleaner** | * Cleanness * Efficiency * Battery life * Security | * Room * Table * Wood floor * Carpet * Various obstacles | * Wheels * Brushes * Vacuum Extractor | * Camera * Dirt detection sensor * Cliff sensor * Bump Sensor * Infrared Wall Sensor |

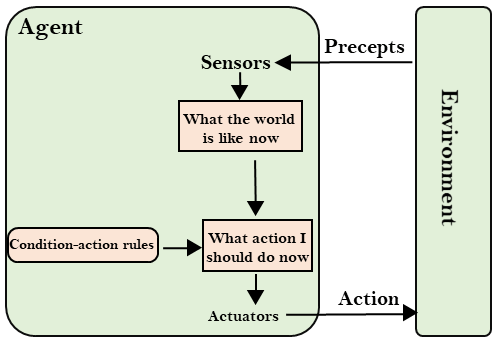
# Types of AI Agents

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:

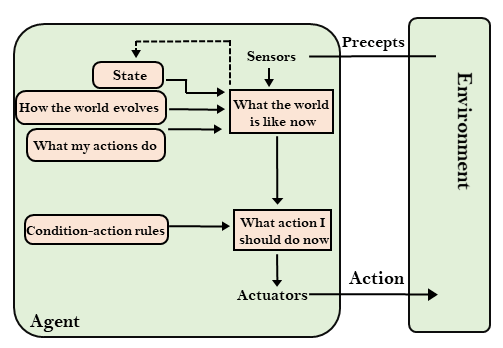
* The Simple reflex agents are the simplest agents. These agents take decisions on the basis of the current percepts and ignore the rest of the percept history.
* These agents only succeed in the fully observable environment.
* The Simple reflex agent does not consider any part of percepts history during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



## A-1.PNG

## 2. Model-based reflex agent

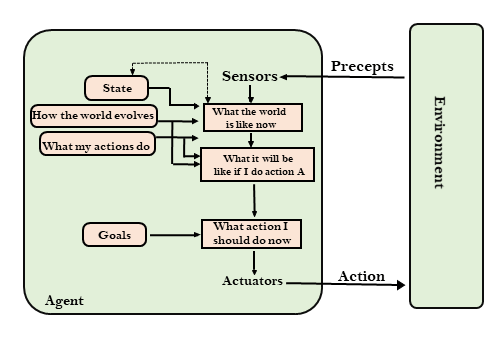
* The Model-based agent can work in a partially observable environment, and track the situation.
* A model-based agent has two important factors:
  + **Model:** It is knowledge about "how things happen in the world," so it is called a Model-based agent.
  + **Internal State:** It is a representation 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:
  + How the world evolves
  + How the agent's action affects the world.



## A-2.PNG

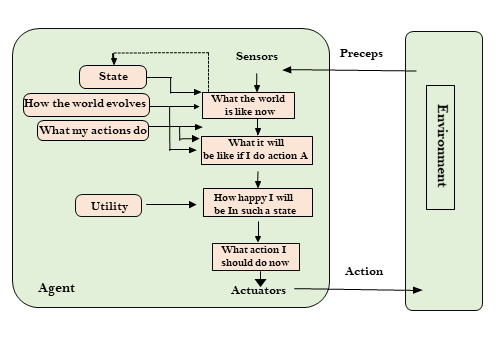
## 3. Goal-based agents

* 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.



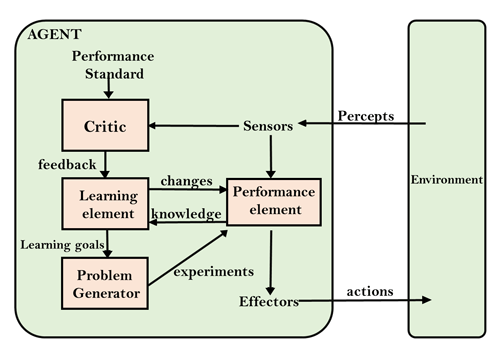
## 4. Utility-based agents

* 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:
  1. **Learning element:** It is responsible for making improvements by learning from environment
  2. **Critic:** Learning element takes feedback from critic which describes that how well the agent is doing with respect to a fixed performance standard.
  3. **Performance element:** It is responsible for selecting external action
  4. **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

As per Russell and Norvig, 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.

# Search Algorithms in Artificial Intelligence

Search algorithms are one of the most important areas of Artificial Intelligence. This topic will explain all about the search algorithms in AI.

### 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.

## Search Algorithm Terminologies:

* **Search:** Searching is a step by step procedure to solve a search-problem in a given search space. A search problem can have three main factors:
  1. **Search Space:** Search space represents a set of possible solutions, which a system may have.
  2. **Start State:** It is a state from where agent begins **the search**.
  3. **Goal test:** It is a function which observe the current state and returns whether the goal state is achieved or not.
* **Search tree:** A tree representation of search problem is called Search tree. The root of the search tree is the root node which is corresponding to the initial state.
* **Actions:** It gives the description of all the available actions to the agent.
* **Transition model:** A description of what each action do, can be represented as a transition model.
* **Path Cost:** It is a function which assigns a numeric cost to each path.
* **Solution:** It is an action sequence which leads from the start node to the goal node.
* **Optimal Solution:** If a solution has the lowest cost among all solutions.

## 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.

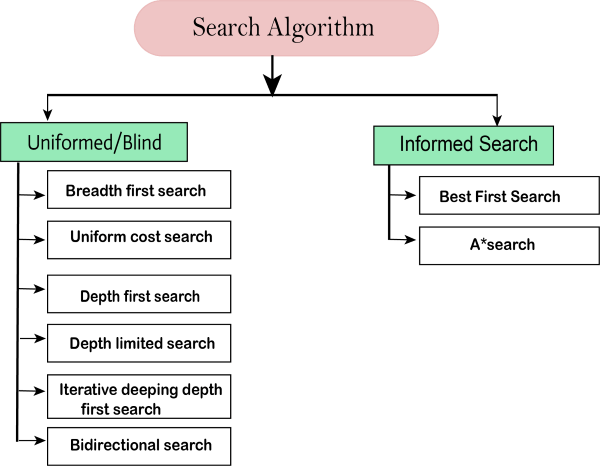
**Optimality:** If a solution found for an algorithm is guaranteed to be the best solution (lowest path cost) among all other solutions, then such a solution for is said to be an optimal solution.

**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:

The uninformed search does not contain any domain knowledge such as closeness, the location of the goal. It operates in a brute-force way as it only includes information about how to traverse the tree and how to identify leaf and goal nodes. Uninformed search applies a way in which search tree is searched without any information about the search space like initial state operators and test for the goal, so it is also called blind search.It examines each node of the tree until it achieves the goal node.

**It can be divided into five main types:**

* Breadth-first search
* Uniform cost search
* Depth-first search
* Iterative deepening depth-first search
* Bidirectional Search

### Informed Search

Informed search algorithms use domain knowledge. In an informed search, problem information is available which can guide the search. Informed search strategies can find a solution more efficiently than an uninformed search strategy. Informed search is also called a Heuristic search.

A heuristic is a way which might not always be guaranteed for best solutions but guaranteed to find a good solution in reasonable time.

Informed search can solve much complex problem which could not be solved in another way.

An example of informed search algorithms is a traveling salesman problem.

1. Greedy Search
2. A\* Search

## Searching for Solutions

## 1. Breadth-first Search:

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

**Advantages:**

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

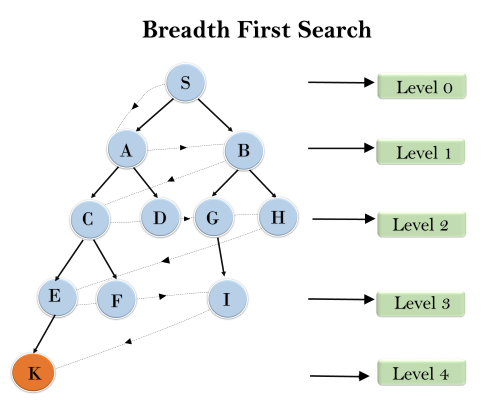
**Disadvantages:**

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

### Example:

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

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



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

**T (b) = 1+b2+b3+.......+ bd= O (bd)**

**Space Complexity:** Space complexity of BFS algorithm is given by the Memory size of frontier which is O(bd).

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

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

## 2. Depth-first Search

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

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

**Advantage:**

* DFS requires very less memory as it only needs to store a stack of the nodes on the path from root node to the current node.
* It takes less time to reach to the goal node than BFS algorithm (if it traverses in the right path).

**Disadvantage:**

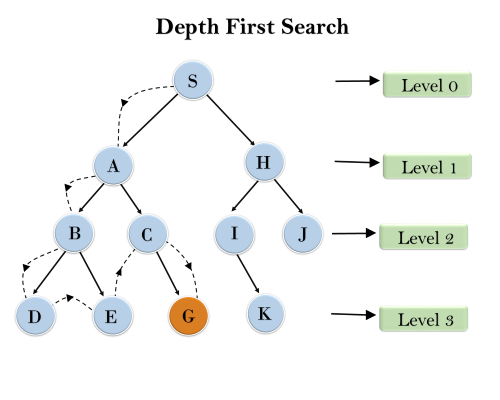
* There is the possibility that many states keep re-occurring, and there is no guarantee of finding the solution.
* DFS algorithm goes for deep down searching and sometime it may go to the infinite loop.

### Example:

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

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

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



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

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

**T(n)= 1+ n2+ n3 +.........+ nm=O(nm)**

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

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

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

# 3.Hill Climbing Algorithm

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

## Features of Hill Climbing:

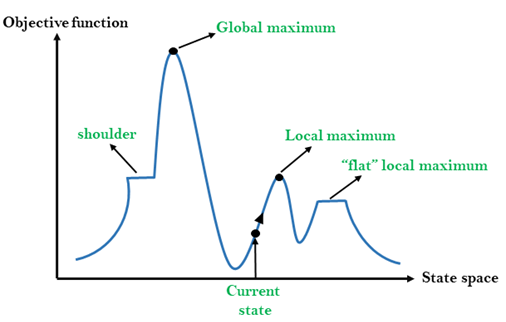
Following are some main features of Hill Climbing Algorithm:

* **Generate and Test variant:** Hill Climbing is the variant of Generate and Test method. The Generate and Test method produce feedback which helps to decide which direction to move in the search space.
* **Greedy approach:** Hill-climbing algorithm search moves in the direction which optimizes the cost.
* **No backtracking:** It does not backtrack the search space, as it does not remember the previous states.

## State-space Diagram for Hill Climbing:

The state-space landscape is a graphical representation of the hill-climbing algorithm which is showing a graph between various states of algorithm and Objective function/Cost.

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 minimum 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.



## Different regions in the state space landscape:

**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 states have the same value.

**Shoulder:** It is a plateau region which has an uphill edge.

## 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

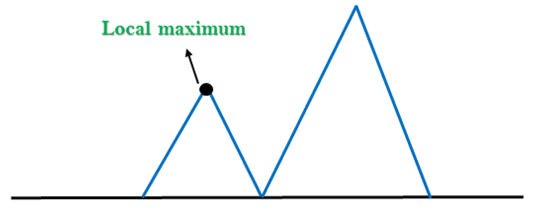
### Algorithm for Simple Hill Climbing:

* **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:
  1. If it is goal state, then return success and quit.
  2. Else if it is better than the current state then assign new state as a current state.
  3. Else if not better than the current state, then return to step2.
* **Step 5:** Exit.

## Problems in Hill Climbing Algorithm:

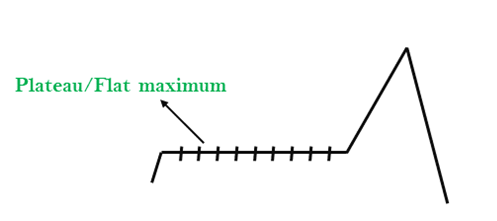
**1. Local Maximum:** A local maximum is a peak state in the landscape which is better than each of its neighboring states, but there is another state also present which is higher than the local maximum.

**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.

