Local Search: 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.
- It is also called greedy local search as it only looks to its good immediate neighbor state.
- 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.

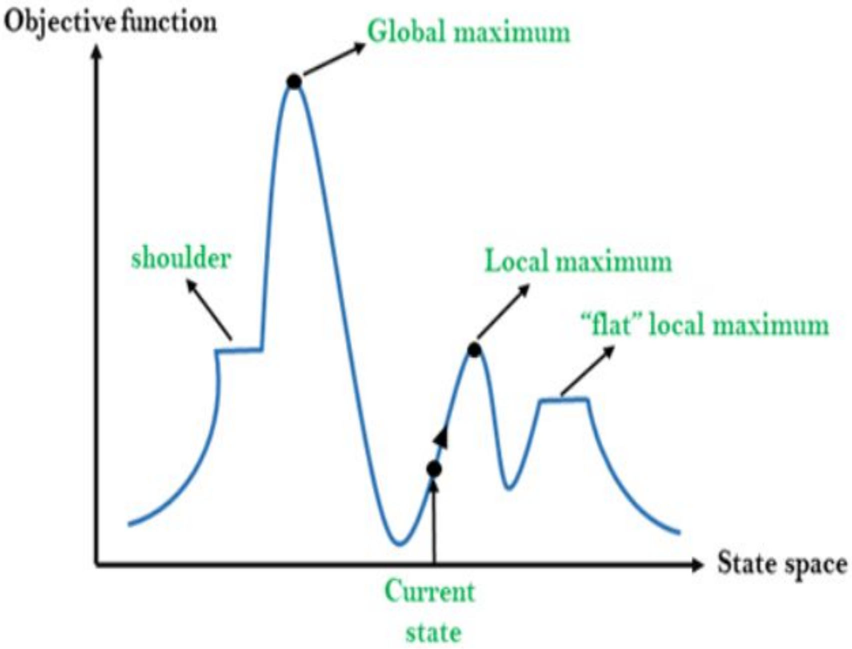
Local Search: Hill Climbing Algorithm

Features of Hill Climbing:

Following are some main features of Hill Climbing Algorithm:

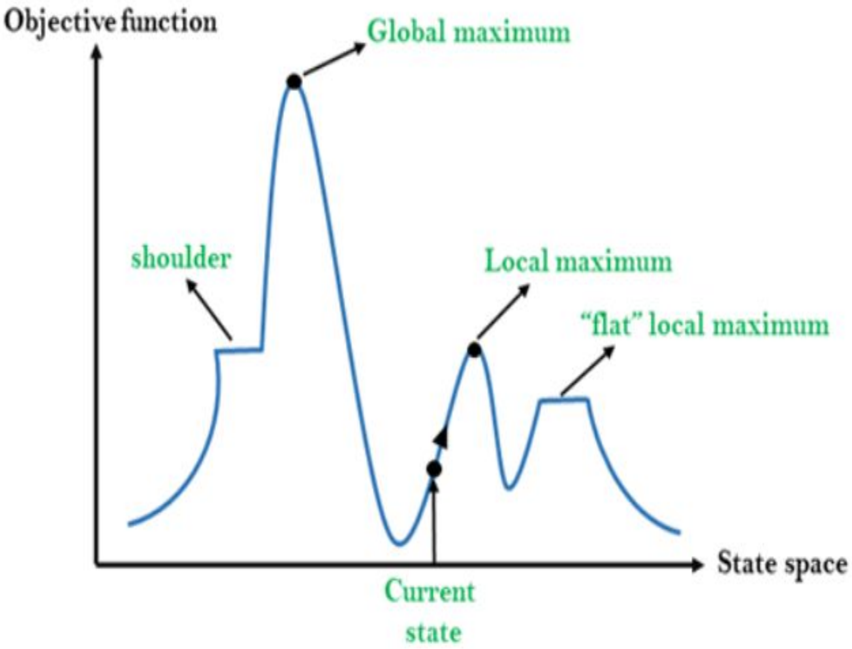
- **Generate and Test variant:** Hill Climbing is the variant of Generate and Test method. The Generate and Test method produce feedback which helps to decide which direction to move in the search space.
- **Greedy approach:** Hill-climbing algorithm search moves in the direction which optimizes the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

State Space Diagram for Hill Climbing



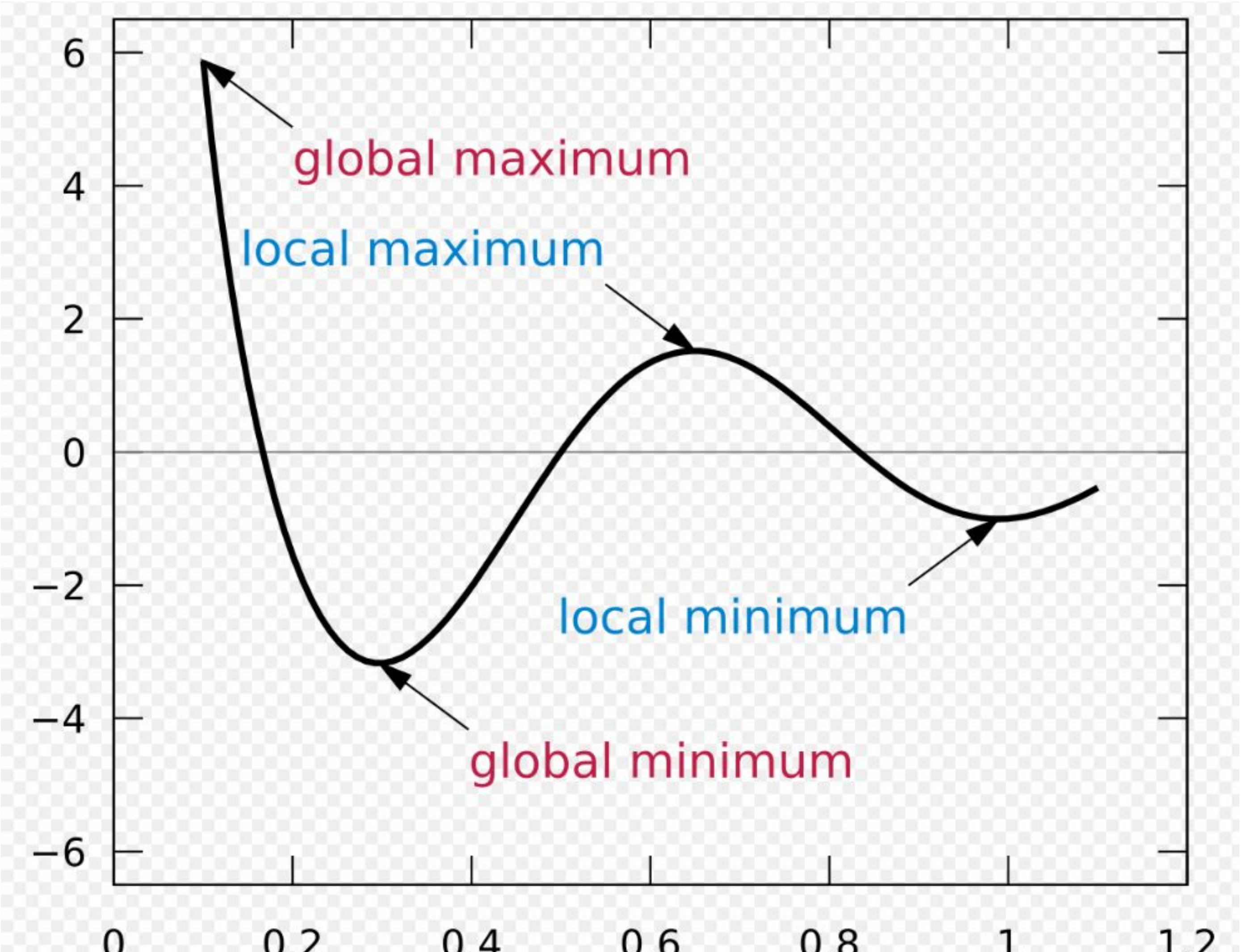
- graphical representation of the hill-climbing algorithm which is showing a graph between various states of algorithm and Objective function/Cost.
- on Y-axis we have taken the function which can be an objective function or cost function
- state-space on the x-axis
- function on Y-axis is cost then, the goal of search is to find the global and local minimum.
- function of Y-axis is Objective function, then the goal of the search is to find the global maximum and local maximum.

State Space Diagram for Hill Climbing



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.



global maximum
local maximum
local minimum
global minimum
6
4
2
0
−2
−4
−6
0
0.2
0.4
0.6
0.8
1
1.2

Types of Hill Climbing

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

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:
  a. Less time consuming
  b. Less optimal solution and the solution is not guaranteed

Simple Hill Climbing

Algorithm for Simple Hill Climbing:

- Evaluate the initial state. If it is a goal state then stop and return success. Otherwise, make the initial state as the current state.
- Loop until the solution state is found or there are no new operators present which can be applied to the current state.
    - Select a state that has not been yet applied to the current state and apply it to produce a new state.
    - Perform these to evaluate the new state.
        - If the current state is a goal state, then stop and return success.
        - If it is better than the current state, then make it the current state and proceed further.
        - If it is not better than the current state, then continue in the loop until a solution is found.
- Exit from the function.

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

Steepest Ascent Hill Climbing

Algorithm for Steepest-Ascent hill climbing:

1. Evaluate the initial state. If it is a goal state then stop and return success. Otherwise, make the initial state as the current state.
2. Repeat these steps until a solution is found or the current state does not change
   - Select a state that has not been yet applied to the current state.
   - Initialize a new 'best state' equal to the current state and apply it to produce a new state.
   - Perform these to evaluate the new state
     - If the current state is a goal state, then stop and return success.
     - If it is better than the best state, then make it the best state else continue the loop with another new state.
   - Make the best state as the current state and go to Step 2 of the second point.
3. Exit from the function.

Stochastic Hill Climbing

- It does not examine all the neighboring nodes before deciding which node to select.
- It just selects a neighboring node at random and decides

Stochastic Hill Climbing

Algorithm:

- Evaluate the initial state. If it is a goal state then stop and return success. Otherwise, make the initial state the current state.
- Repeat these steps until a solution is found or the current state does not change.
  - Select a state that has not been yet applied to the current state.
  - Apply the successor function to the current state and generate all the neighbor states.
  - Among the generated neighbor states which are better than the current state choose a state randomly .
  - If the chosen state is the goal state, then return success, else make it the current state and repeat step 2 of the second point.
- Exit from the function

Problems in Hill Climbing Algorithm

Hill climbing cannot reach the optimal/best state(global maximum) if it enters any of the following regions :

- **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.
    a. **To overcome the local maximum problem:** Utilize the backtracking technique. Maintain a list of visited states. If the search reaches an undesirable state, it can backtrack to the previous configuration and explore a new path.
- **Plateau:** On the plateau, all neighbors have the same value. Hence, it is not possible to select the best direction.
    a. **To overcome plateaus:** Make a big jump. Randomly select a state far away from the current state. Chances are that we will land in a non-plateau region.
- **Ridge:** Any point on a ridge can look like a peak because movement in all possible directions is downward. Hence the algorithm stops when it reaches this state.
    a. **To overcome Ridge:** With the use of bidirectional search, or by moving in different directions, we can improve this problem.