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.

- In the above definition, mathematical optimization problems implies that hill-climbing solves the problems where we need to maximize or minimize a given real function by choosing values from the given inputs. Example - Travelling salesman problem where we need to minimize the distance traveled by the salesman.
- '**Heuristic search**' means that this search algorithm may not find the optimal solution to the problem. However, it will give a good solution in reasonable time.
- A heuristic function is a function that will rank all the possible alternatives at any branching step in search algorithm based on the available information. It helps the algorithm to select the best route out of possible routes.

Features of Hill Climbing

1.Variant of generate and test algorithm : It is a variant of generate and test algorithm. The generate and test algorithm is as follows :

1. Generate possible solutions.

2. Test to see if this is the expected solution.

3. If the solution has been found quit else go to step 1.

Hence we call Hill climbing as a variant of generate and test algorithm as it takes the feedback from the test procedure. Then this feedback is utilized by the generator in deciding the next move in search space.

2.Uses the Greedy approach : At any point in state space, the search moves in that direction only which optimizes the cost of function with the hope of finding the optimal solution at the end.

Types of Hill Climbing

1. **Simple Hill climbing** : It examines the neighboring nodes one by one and selects the first neighboring node which optimizes the current cost as next node.

Algorithm for Simple Hill climbing :

Step 1 : Evaluate the initial state. If it is a goal state then stop and return success. Otherwise, make initial state as current state.

Step 2 : Loop until the solution state is found or there are no new operators present which can be applied to the current state.

a) Select a state that has not been yet applied to the current state and apply it to produce a new state.

b) Perform these to evaluate new state

i. If the current state is a goal state, then stop and return success.

ii. If it is better than the current state, then make it current state and proceed further.

iii. If it is not better than the current state, then continue in the loop until a solution is found.

Step 3 : Exit.

2. **Steepest-Ascent Hill climbing**: It first examines all the neighboring nodes and then selects the node closest to the solution state as of next node.

Step 1 : Evaluate the initial state. If it is goal state then exit else make the current state as initial state

Step 2 : Repeat these steps until a solution is found or current state does not change

i. Let 'target' be a state such that any successor of the current state will be better than it;

ii. for each operator that applies to the current state

a. apply the new operator and create a new state

b. evaluate the new state

c. if this state is goal state then quit else compare with 'target'

d. if this state is better than 'target', set this state as 'target'

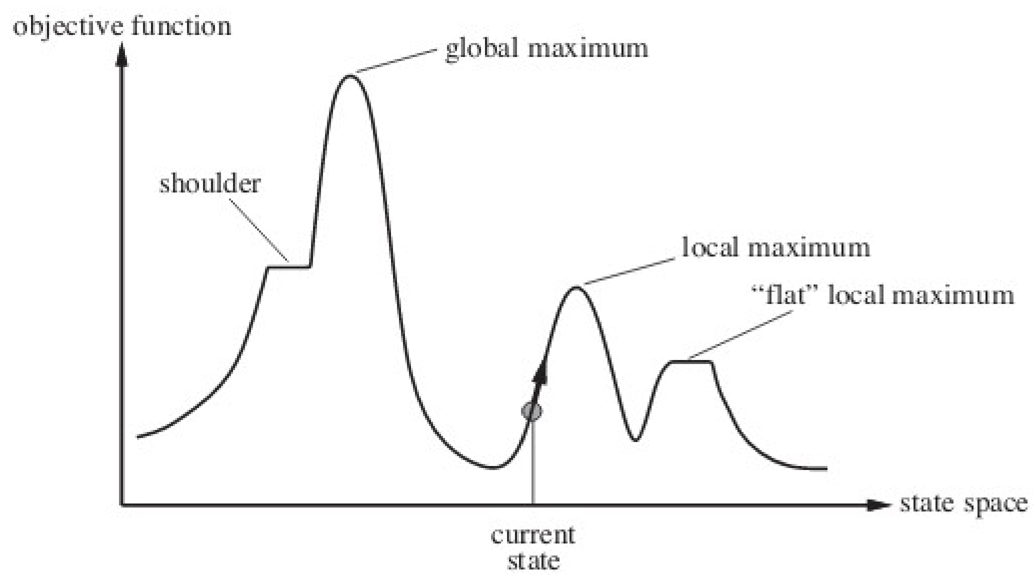
e. if target is better than current state set current state to Target

Step 3 : Exit

3. **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 (based on the amount of improvement in that neighbor) whether to move to that neighbor or to examine another.

State Space diagram for Hill Climbing

State space diagram is a graphical representation of the set of states our search algorithm can reach vs the value of our objective function(the function which we wish to maximize). X-axis : denotes the state space ie states or configuration our algorithm may reach. Y-axis : denotes the values of objective function corresponding to a particular state. The best solution will be that state space where objective function has maximum value(global maximum).



Different regions in the State Space Diagram

1. **Local maximum**: It is a state which is better than its neighboring state however there exists a state which is better than it(global maximum). This state is better because here the value of the objective function is higher than its neighbors.
2. **Global maximum** : It is the best possible state in the state space diagram. This because at this state, objective function has highest value.
3. **Plateua/flat local maximum** : It is a flat region of state space where neighboring states have the same value.
4. **Ridge** : It is region which is higher than its neighbours but itself has a slope. It is a special kind of local maximum.
5. **Current state** : The region of state space diagram where we are currently present during the search.
6. **Shoulder** : It is a plateau that has an uphill edge.

Problems in different regions in Hill climbing

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

1. **Local maximum** : At a local maximum all neighboring states have a values which is worse than the current state. Since hill-climbing uses a greedy approach, it will not move to the worse state and terminate itself. The process will end even though a better solution may exist. To overcome local maximum problem : Utilize 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.
2. **Plateau** : On plateau all neighbors have same value . Hence, it is not possible to select the best direction.

   To overcome plateaus : Make a big jump. Randomly select a state far away from the current state. Chances are that we will land at a non-plateau region
3. **Ridge** : Any point on a ridge can look like peak because movement in all possible directions is downward. Hence the algorithm stops when it reaches this state. To overcome Ridge :In this kind of obstacle, use two or more rules before testing. It implies moving in several directions at once.