**Hill Climbing-Depth first and Breadth first**

*“Success in creating AI would be the biggest event in human history.”*

**Abstract:**

*Hill Climbing technique is mainly used for solving computationally hard problems. It looks only at the current state and immediate future state. Hence, this technique is memory efficient as it does not maintain a search tree.*

**1. Introduction**

Hill Climbing is a heuristic search used for mathematical optimization problems in the field.

In the above definition, mathematical optimization problems imply that hill-climbing solves the problems where we need to maximise or minimise a given real function by choosing values from the given inputs. Example-Travelling salesman problem where we need to minimise the distance travelled 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 a reasonable time.

A heuristic function is a function that will rank all the possible alternatives at any branching step in the search algorithm based on the available information. It helps the algorithm to select the best route out of possible routes 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. /End

**2. How hill climbing algorithm work**

This algorithm is also known as the simple hill climbing problem.

Based on:

1. local search algorithm: this problem only bounded in certain domains they don't know about the complete domain knowledge.
2. Greedy approach: at any point in state space, the search moves in that direction only which optimises the cost of function with the hope of finding the optimal solution at the end.
3. No backtracking: no best move, no backtracking.

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

**3.Types of hills climbing**

**3.1 Simple Hill Climbing**

Simple hill climbing is the simplest way to implement a hill-climbing algorithm. It only evaluates the neighbour node state at a time and selects the first one which optimises current cost and sets it as a current state. It only checks its 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
* The less optimal solution and the solution is not guaranteed

Algorithm for Simple Hill Climbing:

* **Step 1:** Evaluate the initial state, if it is a 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:

a. If it is a goal state, then return to success and quit. b. Else if it is better than the current state then assigns a new state as a current state.

c. Else if not better than the current state, then return to step2.

* **Step 5:** Exit.

**3.2. Steepest-Ascent hill climbing:**

The steepest-Ascent algorithm is a variation of a simple hill-climbing algorithm. This algorithm examines all the neighbouring nodes of the current state and selects one neighbour node which is closest to the goal state. This algorithm consumes more time as it searches for multiple neighbours.

Algorithm for Steepest-Ascent hill climbing:

* **Step 1:** Evaluate the initial state, if it is the goal state then return success and stop, else make the current state the initial state.
* **Step 2:** Loop until a solution is found or the current state does not change.

**2.1**. Let SUCC be a state such that any successor of the current state will be better than it.

**2.2.** For each operator that applies to the current state:

a. Apply the new operator and generate a new state.

b. Evaluate the new state.

c. If it is a goal state, then return it and quit, else compare it to the SUCC.

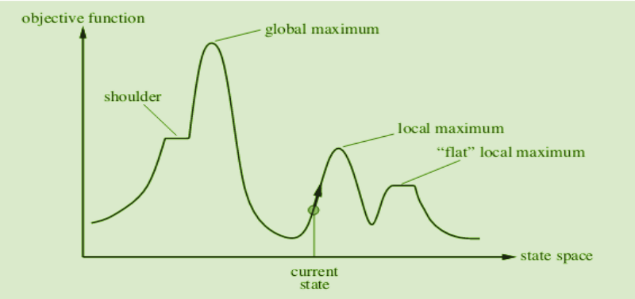
d. If it is better than SUCC, then set a new state as SUCC.

e. If the SUCC is better than the current state, then set the current state to SUCC.

* **Step3**: Exit.

**3.3. Stochastic hill climbing:**

Stochastic hill climbing does not examine all its neighbours before moving. Rather, this search algorithm selects one neighbour node at random and decides whether to choose it as a current state or examine another state.

**4. 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 maximise).

**X-axis:** denotes the state space i.e., 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 the objective function has maximum value (global maximum).

**Different regions in the state space diagram:**

**Local maximum:** It is a state which is better than its neighbouring 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 neighbours.

**Global maximum:** It is the best possible state in the state space diagram. This is because, at this stage, the objective function has the highest value.

**Plateau/flat local maximum:** It is a flat region of state space where neighbouring states have the same value.

**Ridge:** It is a region that is higher than its neighbours but itself has a slope. It is a special kind of local maximum.

**Current state:** The region of state space diagram where we are currently present during the search.

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

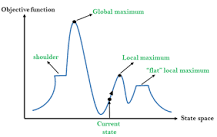
**5. Time and space complexity:**

**Time complexity:**  O (∞)

**complexity:** O(b).

Hill climbing is neither complete nor optimal and has a time complexity of O(∞) but a space complexity of O(b).

No special implementation data structure since hill-climbing discards old nodes. Because of this "amnesty", hill climbing is a suboptimal search strategy and hill-climbing is not complete.

**How can I improve my hill climbing algorithm?**

**Algorithm for Steepest-Ascent hill climbing:**

1. Apply the new operator and generate a new state.
2. Evaluate the new state.
3. If it is a goal state, then return it and quit, else compare it to the SUCC.
4. If it is better than SUCC, then set a new state as SUCC.

**6. Current research from the topic:**

**6.1 A Study on Hill Climbing Algorithms for Neural Network Training:**

This study investigates variations of hill climbing algorithms for training artificial neural networks on the 5-bit parity classification task. The experiments compare algorithms using different combinations of random number distributions, step size changes, and neural network initial weight distribution changes.

A hill-climbing algorithm that uses inline search has been proposed. Most experiments with 5-bit parity tasks have shown better performance than simulated annealing and standard hill climbing.

**6.2** [**Optimal and Suboptimal Reliable Scheduling of Precedence-Constrained Tasks in Heterogeneous Distributed Computing**](https://www.academia.edu/72474227/Optimal_and_Suboptimal_Reliable_Scheduling_of_Precedence_Constrained_Tasks_in_Heterogeneous_Distributed_Computing)**:**

This paper introduces algorithms that can produce task assignments that minimise the probability of failure for an application running on a heterogeneous distributed computing system.

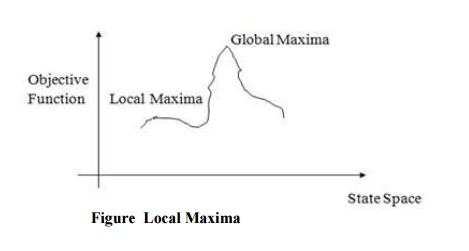
**6.3** [**Steepest Ascent Hill Climbing for Portfolio Selection**](https://www.academia.edu/71888133/Steepest_Ascent_Hill_Climbing_for_Portfolio_Selection)**:**

The construction of a portfolio in the financial field is a problem faced by many people and organisations around the world. This paper presents an approach to solving the portfolio selection problem at the steepest Ascent Hill.

**7. Drawback of this theory:**

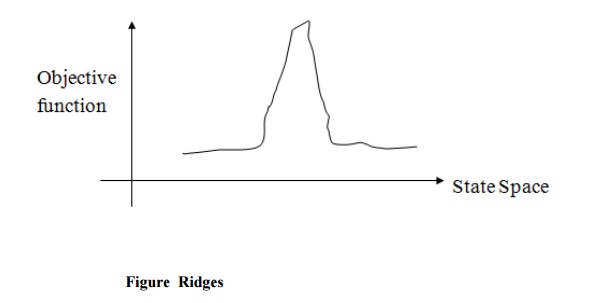
**7.1 Local Maxima:**

A local maximum is a state that is better than each of its neighbouring states, but not better than some other states further away. Generally, this state is lower than the global maximum. At this point, one cannot decide easily to move in which direction! This difficulty can be extracted by the process of backtracking i.e., backtrack to any of one earlier node position and trying to go in a different event direction. To implement this strategy, maintain a list of paths almost taken and go back to one of them. If the path was taken that leads to a dead-end, then go back to one of them.



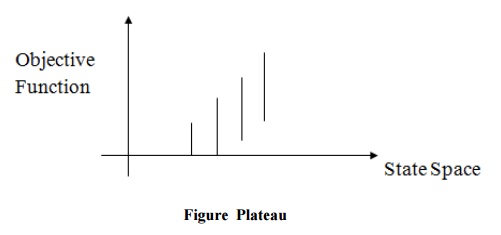
**Ridges:**

 It is a special type of local maxima. It is simply an area of search space. Ridges result in a sequence of local maxima that is very difficult to implement; the ridge itself has a slope that is difficult to traverse. In this type of situation apply two or more rules before doing the test. This will correspond to moving in several directions at once.



**Plateau:**

It is a flat area of search space in which the neighbours have the same value. So it is very difficult to calculate the best direction. So, to get out of this situation, make a big jump in any direction, which will help to move in a new direction. This is the best way to handle the problem like a plateau.

****

**8. How Hill Climbing Help in AI?**

A hill-climbing algorithm is an Artificial Intelligence (AI) algorithm that increases in value continuously until it achieves a peak solution. This algorithm is used to optimise mathematical problems and in other real-life applications like marketing and job scheduling.

**9. Applications of Hill Climbing Technique:**

Hill Climbing technique can be used to solve many problems, where the current state allows for an accurate evaluation function, such as Network-Flow, Travelling Salesman problem, 8-Queens problem, Integrated Circuit design, etc.

**10. Real life example of hill climbing algorithm:**

One of the widely discussed examples of Hill climbing algorithm is the Travelling-salesman Problem in which we need to minimise the distance travelled by the salesman. It is also called greedy local search as it only looks to its good immediate neighbour state and not beyond that.

**11. Edge cases of this algorithm:**

**What if all peak points have the same value in the hill climbing analysis.**

* Time Complexity: O [1]

**Explanation:**

If all peak points have the same value over for this case. So, we don't need to check further all other peak cases. The first peak of the algorithm is the answer and the time complexity is constant because of the same peak.

**12. Conclusion:**

In our article, we discussed hill-climbing algorithms, how it does work, and different parts of them. Along with that we also mentioned how it works and AI. Hill climbing and gradient descent algorithms belong to the folklore of computer science. Hill climbing is an example of an informed search method because it uses information about the search space to search in a reasonably efficient manner. The hill climbing search always moves towards the goal. Using heuristics, it finds which direction will take it closest to the goal. The name hill climbing is derived from simulating the situation of a person climbing the hill.

**13. References:**

* [Hill Climbing Algorithm in AI - Javapoint](https://www.javatpoint.com/hill-climbing-algorithm-in-ai)
* <https://www.geeksforgeeks.org/introduction-hill-climbing-artificial-intelligence/>
* <https://medium.com/@bhavek.mahyavanshi50/introduction-to-hill-climbing-artificial-intelligence-a3714ed2d8d8>