International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 07 Issue: 11 | November - 2023 SJIF Rating: 8.176 ISSN: 2582-3930

Applications Hill Climbing Algorithm

Dr. Suneel Pappala, Associate Professor, Computer Science & Engineering,
Lords Institute of Engineering & Technology, Osmania University

Abstract:
Hill climbing is mathematical optimization techniques that belong to the local search family. It is an iterative algorithm that starts with an arbitrary solution to a problem and then attempts to find a better solution by making an incremental change to the solution. If the change produces a better solution, another incremental change is made to the new solution, and so on until no further improvements can be found. The hill-climbing algorithm is a local search algorithm that moves continuously upward (increasing) until the best solution is attained. This algorithm comes to an end when the peak is reached. The state space diagram provides a graphical representation of states and the optimization function. If the objective function is the y-axis, we aim to establish the local maximum and global maximum. Hill climbing is useful in the effective operation of robotics. It enhances the coordination of different systems and components in robots.

Keywords: Hill climbing, Local Maximum, Global Maximum, Ridges.

Hill climbing algorithm

A hill-climbing algorithm is a local search algorithm that moves continuously upward (increasing) until the best solution is attained. This algorithm comes to an end when the peak is reached.This algorithm has a node that comprises two parts: state and value. It begins with a non-optimal state (the hill's base) and upgrades this state until a certain precondition is met. The heuristic function is used as the basis for this precondition. The process of continuous improvement of the current state of iteration can be termed as climbing. This explains why the algorithm is termed as a *hill-climbing algorithm*. A hill-climbing algorithm's objective is to attain an optimal state that is an upgrade of the existing state. When the current state is improved, the algorithm will perform further incremental changes to the improved state. This process will continue until a peak solution is achieved. The peak state cannot undergo further improvements. Hill climbing is a simple optimization algorithm used in Artificial Intelligence (AI) to find the best possible solution for a given problem. It belongs to the family of local search algorithms and is often used in optimization problems where the goal is to find the best solution from a set of

possible solutions. 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** imply 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 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.
- In Hill Climbing, the algorithm starts with an initial solution and then iteratively makes small changes to it in order to improve the solution. These changes are based on a heuristic function that evaluates the quality of the solution. The algorithm continues to make these small changes until it reaches a local maximum, meaning that no further improvement can be made with the current set of moves.
- There are several variations of Hill Climbing, including steepest ascent Hill Climbing, first-choice Hill Climbing, and simulated annealing. In steepest ascent Hill Climbing, the algorithm evaluates all the possible moves from the current solution and selects the one that leads to the best improvement. In first-choice Hill Climbing, the algorithm randomly selects a move and accepts it if it leads to an improvement, regardless of whether it is the best move. Simulated annealing is a probabilistic variation of Hill Climbing that allows the algorithm to occasionally accept worse moves in order to avoid getting stuck in local maxima.

Hill Climbing can be useful in a variety of optimization problems, such as scheduling, route planning, and resource allocation. However, it has some limitations, such as the tendency to get stuck in local maxima and the lack of diversity in the search space. Therefore, it is often combined with other optimization techniques, such as genetic algorithms or simulated annealing, to overcome these limitations and improve the search results.

Advantages of Hill Climbing algorithm:

1. Hill Climbing is a simple and intuitive algorithm that is easy to understand and implement.
2. It can be used in a wide variety of optimization problems, including those with a large search space and complex constraints.
3. Hill Climbing is often very efficient in finding local optima, making it a good choice for problems where a good solution is needed quickly.
4. The algorithm can be easily modified and extended to include additional heuristics or constraints.

Disadvantages of Hill Climbing algorithm:

1. Hill Climbing can get stuck in local optima, meaning that it may not find the global optimum of the problem.
2. The algorithm is sensitive to the choice of initial solution, and a poor initial solution may result in a poor final solution.
3. Hill Climbing does not explore the search space very thoroughly, which can limit its ability to find better solutions.
4. It may be less effective than other optimization algorithms, such as genetic algorithms or simulated annealing, for certain types of problems.

Features of a hill climbing algorithm

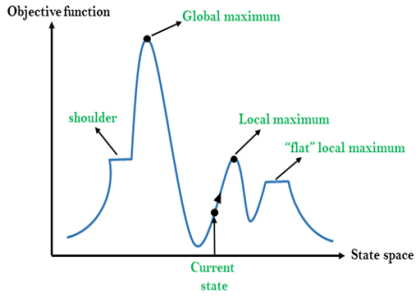
A hill-climbing algorithm has four main features:

1. It employs a greedy approach: This means that it moves in a direction in which the cost function is optimized. The greedy approach enables the algorithm to establish local maxima or minima.
2. No Backtracking: A hill-climbing algorithm only works on the current state and succeeding states (future). It does not look at the previous states.
3. Feedback mechanism: The algorithm has a feedback mechanism that helps it decide on the direction of movement (whether up or down the hill). The feedback mechanism is enhanced through the generate-and-test technique.
4. Incremental change: The algorithm improves the current solution by incremental changes.

State-space diagram analysis

A state-space diagram provides a graphical representation of states and the optimization function. If the objective function is the y-axis, we aim to establish the local maximum and global maximum. If the cost function represents this axis, we aim to establish the local minimum and global minimum. More information about local minimum, local maximum, global minimum, and global maximum can be

found here. The following diagram shows a simple state-space diagram. The objective function has been shown on the y-axis, while the state-space represents the x-axis.



A state-space diagram consists of various regions that can be explained as follows;

- **Local maximum:** A local maximum is a solution that surpasses other neighboring solutions or states but is not the best possible solution.
- **Global maximum:** This is the best possible solution achieved by the algorithm.
- **Current state:** This is the existing or present state.
- **Flat local maximum:** This is a flat region where the neighboring solutions attain the same value.
- **Shoulder:** This is a plateau whose edge is stretching upwards.

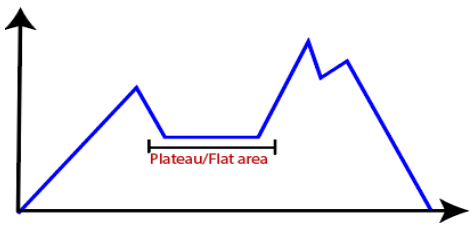
Problems with hill climbing

There are three regions in which a hill-climbing algorithm cannot attain a global maximum or the optimal solution: local maximum, ridge, and plateau.

1. Local maximum At this point, the neighboring states have lower values than the current state. The greedy approach feature will not move the algorithm to a worse off state. This will lead to the hill-climbing process's termination, even though this is not the best possible solution.This problem can be solved using momentum. This technique adds a certain proportion (m) of the initial weight to the current one. m is a value between 0 and 1. Momentum enables the hill-climbing algorithm to take huge steps that will make it move past the local maximum.

2. Plateau

In this region, the values attained by the neighboring states are the same. This makes it difficult for the algorithm to choose the best direction.This challenge can be overcome by taking a huge jump that will lead you to a non-plateau space.



Ridge: The hill-climbing algorithm may terminate itself when it reaches a ridge. This is because the peak of the ridge is followed by downward movement rather than upward movement.This impediment can be solved by going in different directions at once.

Types of hill climbing algorithms

The following are the types of a hill-climbing algorithm:

Simple hill climbing: This is a simple form of hill climbing that evaluates the neighboring solutions. If the next neighbor state has a higher value than the current state, the algorithm will move. The neighboring state will then be set as the current one.This algorithm consumes less time and requires little computational power. However, the solutions produced by the algorithm are sub-optimal. In some cases, an optimal solution may not be guaranteed.

Algorithm

1. Conduct an assessment of the current state. Stop the process and indicate success if it is a goal state.
2. Perform looping on the current state if the assessment in step 1 did not establish a goal state.3.
3. Continue looping to attain a new solution.
4. 4. Assess the new solution. If the new state has a higher value than the current state in steps 1 and 2, then mark it as a current state.
5. Continue steps 1 to 4 until a goal state is attained. If this is the case, then exit the process.

Steepest – Ascent hill climbing

This algorithm is more advanced than the simple hill-climbing algorithm. It chooses the next node by assessing the neighboring nodes. The algorithm moves to the node that is closest to the optimal or goal state.

Algorithm

1. Conduct an assessment of the current state. Stop the process and indicate success if it is a goal state.

2. Perform looping on the current state if the assessment in step 1 did not establish a goal state.

3. Continue looping to attain a new solution.

4. Establish or set a state (X) such that current state successors have higher values than it.

5. Run the new operator and produce a new solution.

6. Assess this solution to establish whether it is a goal state. If this is the case, exit the program. Otherwise, compare it with the state (X).

If the new state has a higher value than the state (X), set it as X. The current state should be set to Target if the state (X) has a higher value than the current state.

Stochastic hill climbing

this algorithm, the neighboring nodes are selected randomly. The selected node is assessed to establish the level of improvement. The algorithm will move to this neighboring node if it has a higher value than the current state.

Steepest-Ascent Hill climbing

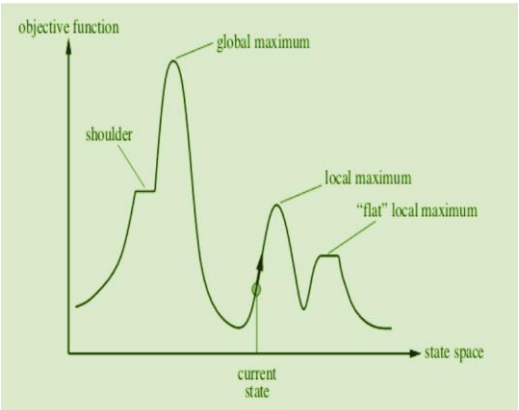
Algorithm for Steepest Ascent Hill climbing :

- Evaluate the initial state. If it is a goal state then stop and return success. Otherwise, make the initial state as the current state.
- 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.
- Exit from the function.

State Space diagram for Hill Climbing

- **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 a state space where the objective function has a maximum value(global maximum).



Different regions in the State Space Diagram:

- **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.
- **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 neighboring states have the same value.
- **Ridge:** It is a region that is higher than its neighbors but itself has a slope. It is a special kind of local maximum.
- **Current state:** The region of the state space diagram where we are currently present during the search.
- **Shoulder:** It is a plateau that has an uphill edge.

Applications of hill climbing algorithm

The hill-climbing algorithm can be applied in the following areas:

Marketing

A hill-climbing algorithm can help a marketing manager to develop the best marketing plans. This algorithm is widely used in solving Traveling-Salesman problems. It can help by optimizing the distance covered and improving the travel time of sales team members. The algorithm helps establish the local minima efficiently.

Robotics

Hill climbing is useful in the effective operation of robotics. It enhances the coordination of different systems and components in robots.

Job Scheduling

The hill climbing algorithm has also been applied in job scheduling. This is a process in which system resources are allocated to different tasks within a computer system. Job scheduling is achieved through the migration of jobs from one node to a neighboring node. A hill-climbing technique helps establish the right migration route.

Let's look at the Simple Hill climbing algorithm:

Generate-And-Test Algorithm

It's a very simple technique that allows us to algorithmize finding solutions:

1. Define current state as an initial state

2. Apply any possible operation on the current state and generate a possible solution
3. Compare newly generated solution with the goal state
4. If the goal is achieved or no new states can be created, quit. Otherwise, return to the step 2

It works very well with simple problems. As it is an exhaustive search, it is not feasible to consider it while dealing with large problem spaces. It is also known as British Museum algorithm (trying to find an artifact in the British Museum by exploring it randomly).It is also the main idea behind the Hill-Climbing Attack in the world of biometrics. This approach can be used for generating synthetic biometric data.

Simple Hill-Climbing Algorithm

Hill-Climbing technique, starting at the base of a hill, we walk upwards until we reach the top of the hill. In other words, we start with initial state and we keep improving the solution until its optimal.It's a variation of a generate-and-test algorithm which discards all states which do not look promising or seem unlikely to lead us to the goal state. To take such decisions, it uses heuristics (an evaluation function) which indicates how close the current state is to the goal state.

simple words, **Hill-Climbing = generate-and-test + heuristics**

Let’s look at the Simple Hill climbing algorithm:

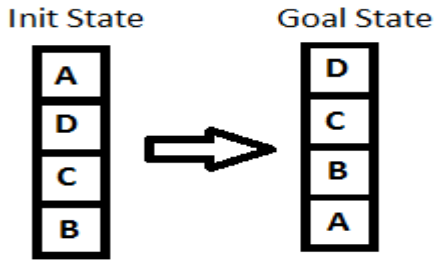
1. Define the current state as an initial state
2. Loop until the goal state is achieved or no more operators can be applied on the current state:
    1. Apply an operation to current state and **get a new state**
    2. **Compare** the new state with the goal
    3. **Quit** if the goal state is achieved
    4. Evaluate new state with heuristic function and **compare it with the current state**
    5. **If the newer state is closer** to the goal compared to current state, **update the current state**

As we can see, it reaches the goal state with iterative improvements. In Hill-Climbing algorithm, finding goal is equivalent to reaching the top of the hill.

Example

Hill Climbing Algorithm can be categorized as an informed search. So we can implement any node-based search or problems like the n-queens problem using it. To understand the concept easily, we will take up a very simple example.

Let's look at the image below:

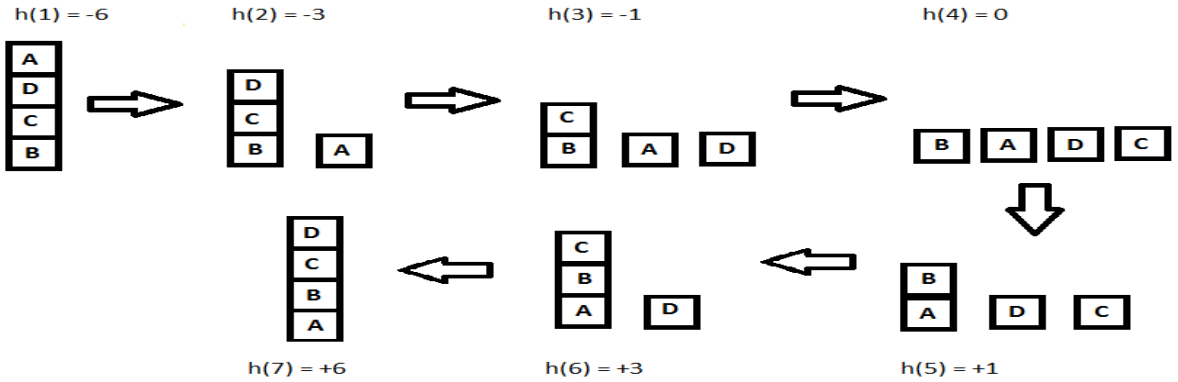


Key point while solving any hill-climbing problem is to choose an appropriate heuristic function.

Let's define such function h:

h(x) = +1 for all the blocks in the support structure if the block is correctly positioned otherwise -1 for all the blocks in the support structure.

Here, we will call any block correctly positioned if it has the same support structure as the goal state. As per the hill climbing procedure discussed earlier let's look at all the iterations and their heuristics to reach the target state:



Bibliography:

I am Dr Suneel Pappala, Associate Professor, Information Technology, Lords Institute of Engineering & Technology, Hyderabad, Telangana, INDIA. I have 15 years of Experience in Computer Science & Engineering and Information Technology as a Associate Professor.

References:

- Aarts, E., and Korst, J. (1989), Simulated Annealing and Boltzmann Machines - a Stochastic Approach to Combinatorial Optimization and Neural Computers, Wiley, New York. 11
- Ahuja, R. K., Ergun, O., Orlin, J. B., and Punnen, A. P. (2002), "A Survey of Very Large-Scale Neighborhood Search Techniques," Discrete Applied Mathematics, 123, 75- 102.
- B¨ack, T., Fogel, D., and Michalewicz, Z. (1997), Handbook of Evolutionary Computation, IOP Publishing and Oxford University Press, New York, NY.
- Baumert, S., Ghate, A., Kiatsupaibul, S., Shen, Y., Smith, R. L., and Zabinsky, Z. B., Discrete Hit-and-Run for Generating Multivariate Distributions over Arbitrary Finite Subsets of a Lattice, forthcoming in Operations Research, 2009.
- B´elisle, C. J. P. (1992), "Convergence Theorems for a Class of Simulated Annealing Algorithms on R d ," J. Applied Probability, 29, 885-895.
- B´elisle, C. J. P., Romeijn, H. E., and Smith, R. L. (1993), "Hit-and-Run Algorithms for Generating Multivariate Distributions," Mathematics of Operations Research, 18, 255-266.
- Blum, C., and Roli, A., (2003) "Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison," ACM Computing Surveys, 35(3), 268-308.
- Boender, C.G.E, and Romeijn, H.E. (1995), "Stochastic Methods," in Handbook of Global Optimization, edited by R. Horst and P. M. Pardalos, Kluwer Academic Publishers, Netherlands, 829-869. [9] Brooks, S. H. (1958), "A Discussion of Random Methods for Seeking Maxima," Operations Research, 6, 244-251.
- Bulger, D., Baritompa, W. P., and Wood, G. R. (2003) "Implementing Pure Adaptive Search with Grover's Quantum Algorithm," Journal of Optimization Theory and Applications, 116, 517-529.
- Bulger, D. W., and Wood, G. R. (1998), "Hesitant Adaptive Search for Global Optimization," Mathematical Programming, 81, 89-102.
- Cohn, H., and Fielding, M. (1999), "Simulated annealing: searching for an optimal temperature schedule," SIAM Journal on Optimization, 9(3), 779-802.
- Davis, L. (1991), Handbook of Genetic Algorithms, Van Nostrand Reinhord, New York.
- De Jong, K. A., Spears, W. M., and Gordon, D.F. (1995), "Using Markov Chains to Analyze GAFOs," in Foundations of Genetic Algorithms 3, edited by D. Whitley, and M. Vose, Morgan Kaufmann, San Francisco, 115-137.
- Del Moral, P. (2004), Feynman-Kac Formulae: Genealogical and Interacting Particle Systems with Applications, Springer, New York.

- Del Moral, P., and Miclo, L. (1999), "On the convergence and applications of generalized simulated annealing," SIAM Journal of Control and Optimization, 37(4), 1222-1250.
- Devore, J. L. (1995), Probability and Statistics for Engineering and the Sciences, Fourth Edition, Wadsworth, Inc. Belmont, CA.
- Dixon, L. C. W., and Szeg¨o, G. P. (1975), Towards Global Optimization, North-Holland, Amsterdam.
- Dixon, L. C. W., and Szeg¨o, G. P. (1978), Towards Global Optimization 2, NorthHolland, Amsterdam.
- Dorigo, M., and St¨utzle, T. (2004), Ant colony optimization, MIT Press, Cambridge, MA.
- Dyer, M. E., and Frieze, A. M. (1991), "Computing the Volume of Convex Bodies: A Case Where Randomness Provably Helps," Proceedings of Symposia in
- Applied Mathematics, 44, 123-169.
- Dyer, M., Frieze, A., and Kannan, R. (1991), "A random polynomial time algorithm for approximating the volume of convex bodies," Journal of the ACM, 38, 1-17.
- Fielding, M. (2000), "Simulated annealing with an optimal fixed temperature," SIAM Journal on Optimization, 11(2), 289-307.
- Glover, F., and Kochenberger, G. A. (2003), Handbook of Metaheuristics, International series in operations research & management science, 57, Kluwer Academic Publishers, Boston.
- Glover, F., and Laguna, M. (1993), "Tabu Search," in Modern Heuristic Techniques for Combinatorial Problems, edited by C. R. Reeves, Halsted Press, New York, 70-150.
- Hajek, B. (1988), "Cooling schedules for optimal annealing," Math. Oper. Res., 13, 311-329.
- Horst, R., and Hoang, T. (1996), Global Optimization: Deterministic Approaches, Third Edition, Springer-Verlag, Berlin.