**Hill Climbing Algorithm in Artificial Intelligence**

December 16, 2020

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 optimize mathematical problems and in other real-life applications like marketing and job scheduling.

This article will improve our understanding of hill climbing in artificial intelligence. It discusses various aspects such as features, problems, types, and algorithm steps of the hill-climbing algorithm. The article will also highlight the applications of this algorithm.

Introduction to 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](https://en.wikipedia.org/wiki/Heuristic_(computer_science)#:~:text=A%20heuristic%20function%2C%20also%20called%20simply%20a%20heuristic%2C,For%20example%2C%20it%20may%20approximate%20the%20exact%20solution.) 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*.

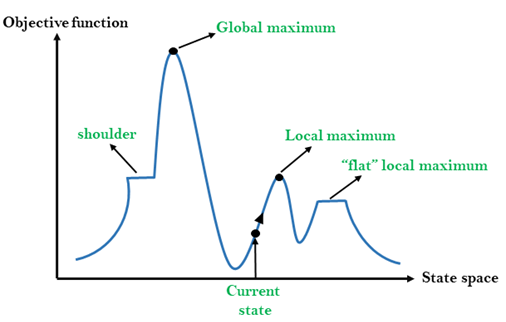
A hill-climbing algorithm’s objective is to attain an optimal state that is an upgrade of the existing state. When the current state is improved, the algorithm will perform further incremental changes to the improved state. This process will continue until a peak solution is achieved. The peak state cannot undergo further improvements.

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](https://www.section.io/engineering-education/greedy-algorithms/) 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](http://intelligence.worldofcomputing.net/ai-search/generate-and-test-search.html#:~:text=%20Algorithm%3A%20Generate-And-Test%20%201%201.Generate%20a%20possible,quit%20else%20go%20to%20step%201.%20More%20).
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](https://en.wikipedia.org/wiki/Maxima_and_minima).

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.

[Image Source: Javat Point](https://static.javatpoint.com/tutorial/ai/images/hill-climbing-algorithm-in-ai.png)

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.

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.

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.

[Image Source: Tutorial and Example](https://www.tutorialandexample.com/wp-content/uploads/2019/07/Plateau.png)

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.

[Image Source: VietMX Blog](https://www.maixuanviet.com/wp-content/uploads/2020/05/hill-climbing-algorithm-in-ai4.png)

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

* Conduct an assessment of the current state. Stop the process and indicate success if it is a goal state.
* Perform looping on the current state if the assessment in step 1 did not establish a goal state.
* Continue looping to attain a new solution.
* 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.
* 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

* Conduct an assessment of the current state. Stop the process and indicate success if it is a goal state.
* Perform looping on the current state if the assessment in step 1 did not establish a goal state.
* Continue looping to attain a new solution.
* Establish or set a state (X) such that current state successors have higher values than it.
* Run the new operator and produce a new solution.
* 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

In 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.

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](https://en.wikipedia.org/wiki/Travelling_salesman_problem) 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.

Conclusion

Hill climbing is a very resourceful technique used in solving huge computational problems. It can help establish the best solution for problems. This technique has the potential of revolutionizing optimization within artificial intelligence.

In the future, technological advancement to the hill climbing technique will solve diverse and unique optimization problems with better advanced features.