MODULE - II

SEARCH STRATEGIES

Hill Climbing

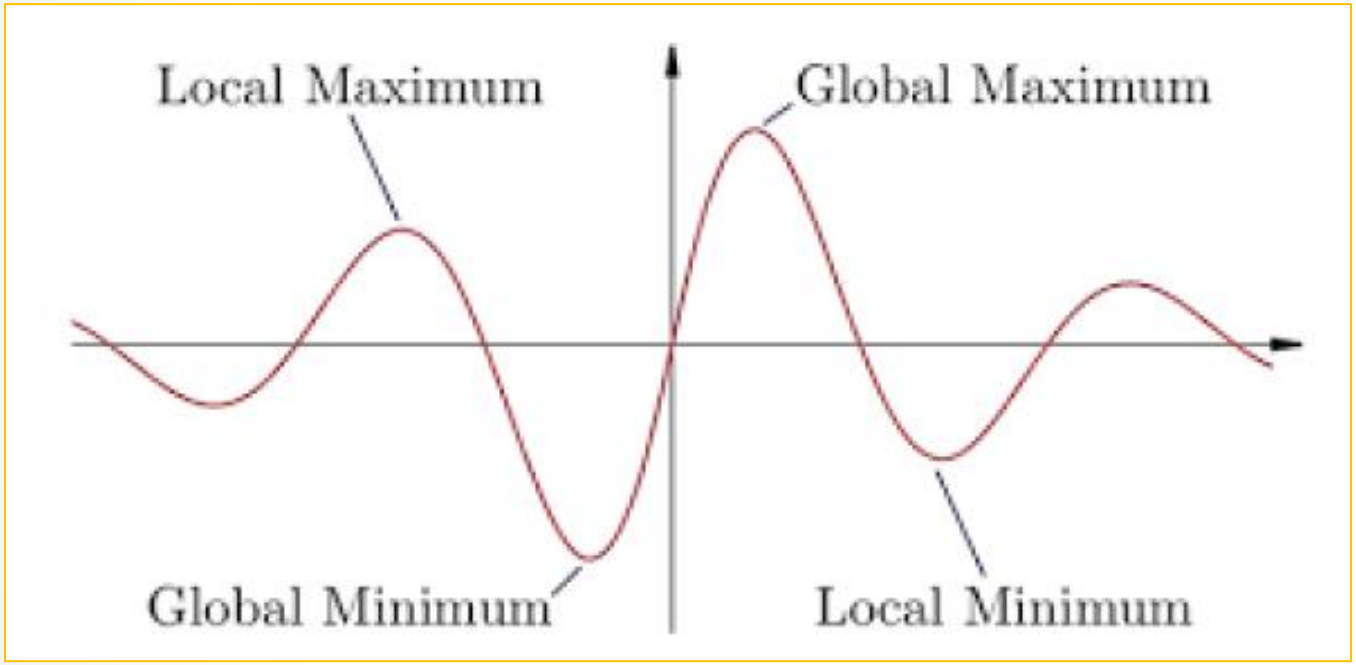
Selma Joseph A.

Hill climbing

- Hill climbing is an iterative algorithm that starts with an arbitrary solution to a problem, 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 solution obtained by this method may not be the global optimum; it will only be a local optimum. In this sense, it is a local search method.

Hill climbing

The difference between a local and a global maxima is illustrated in Figure.



In the figure, the x-axis denotes the nodes in the state space and the y-axis denotes the values of the objective function corresponding to particular states.

Hill climbing

When we perform hill-climbing, we are short-sighted. We cannot really see far. We make local best decisions with the hope of finding a global best solution.

Types of Hill Climb Algorithm

1. Simple Hill Climb Algorithm

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.

Features

Less time consuming

Less optimal solution and the solution is not guaranteed

Steps involved in simple hill climbing algorithm

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

If it is goal state, then return success and quit.

Else if it is better than the current state then assign new state as a current state.

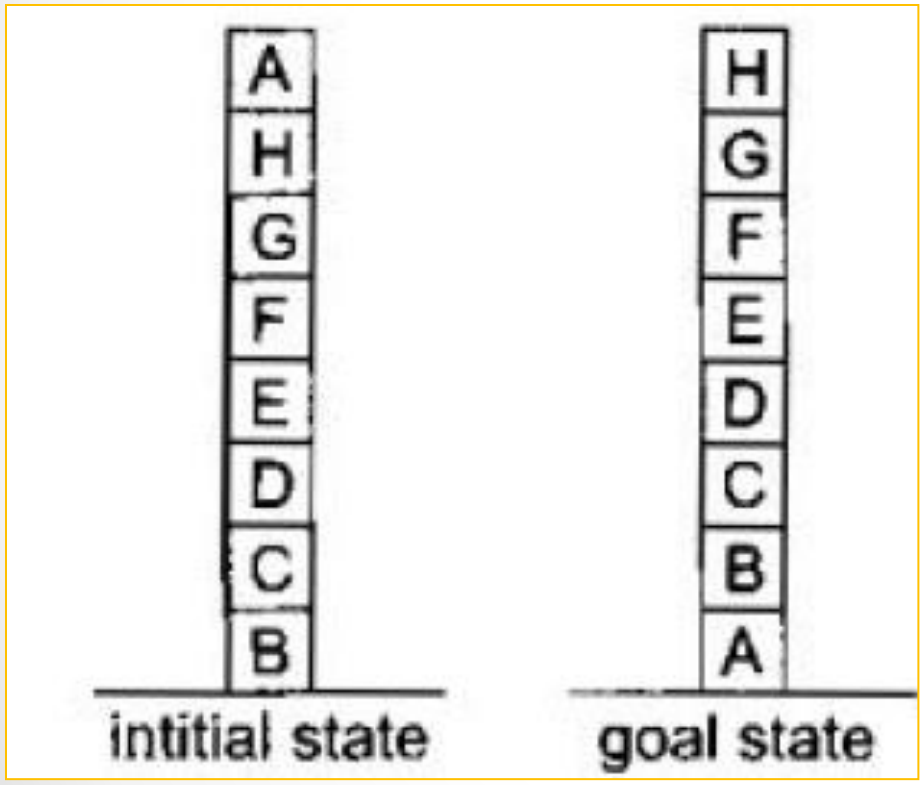
Else if not better than the current state, then return to step2.

Step 5: Exit.

Examples

Example 1

Apply the hill climbing algorithm to solve the blocks world problem shown in Figure.



Solution

To use the hill climbing algorithm we need an evaluation function or a heuristic function.

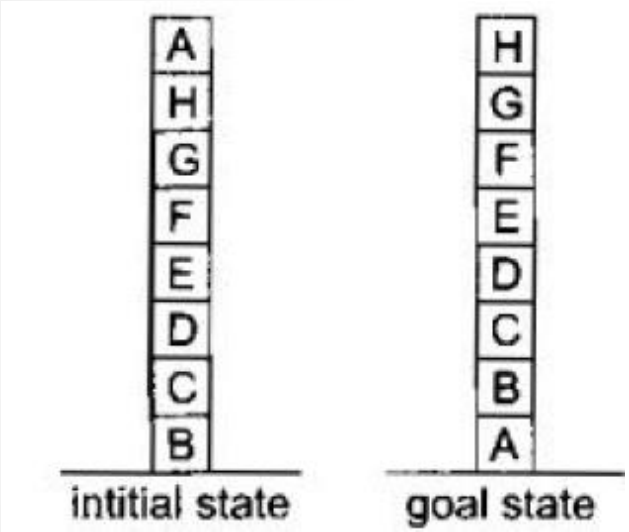
We consider the following evaluation function:

h(n) = Add one point for every block that is resting on the thing it is supposed to be resting on. Subtract one point for every block that is sitting on the wrong thing.

Examples

Example 1

We call "initial state" "State 0" and "goal state" "Goal". Then considering the blocks A, B, C, D, E, F, G, H in that order we have



$$h(\text{State 0}) = -1 - 1 + 1 + 1 + 1 + 1 + 1 + 1 = 4$$

$$h(\text{Goal}) = +1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 = 8$$

h(n) = Add one point for every block that is resting on the thing it is supposed to be resting on. Subtract one point for every block that is sitting on the wrong thing.

There is only one move from State 0, namely, to move block A to the table. This produces the State 1 with a score of 6:

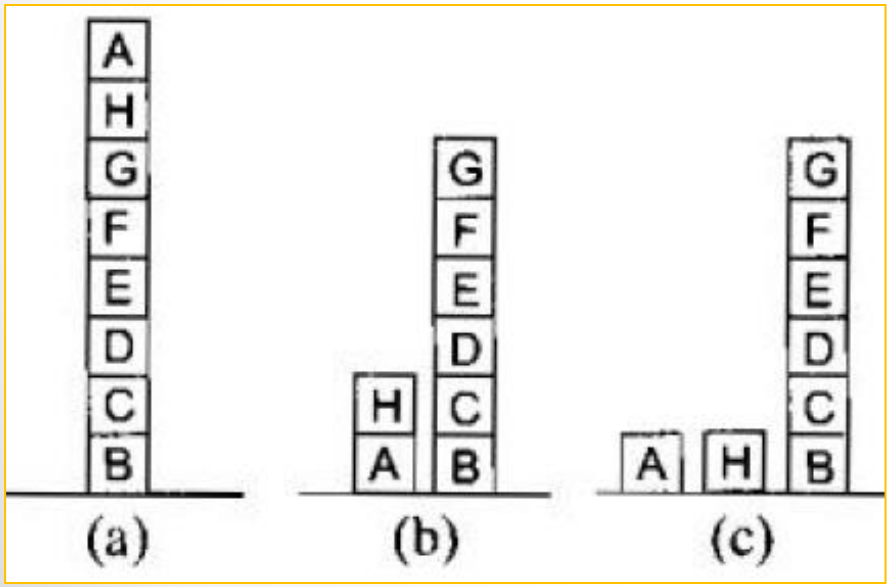
$$h(\text{State 1}) = 1 - 1 + 1 + 1 + 1 + 1 + 1 + 1 = 6.$$

Examples

Example 1

There are three possible moves form State 1 as shown in Figure. We denote these three states State 2(a), State 2(b) and State 2 (c). We also have

$$h(\text{State 2(a)}) = -1 - 1 + 1 + 1 + 1 + 1 + 1 + 1 = 4$$
$$h(\text{State 2(b)}) = +1 - 1 + 1 + 1 + 1 + 1 + 1 - 1 = 4$$
$$h(\text{State 2(c)}) = +1 - 1 + 1 + 1 + 1 + 1 + 1 - 1 = 4$$

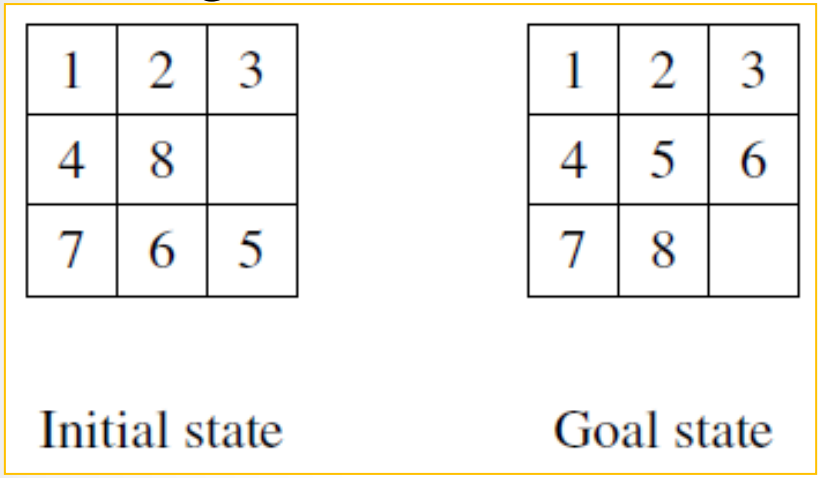


Hill climbing will halt because all these states have lower scores than the current state. The process has reached a local maximum that is not a global maximum. We have reached such a situation because of the particular choice of the heuristic function. A different choice of heuristic function may not produce such a situation.

Examples

Example 2

Given the 8-puzzle shown in Figure, use the hill-climbing algorithm with the Manhattan distance heuristic to find a path to the goal state.



Solution

By definition, the Manhattan distance heuristic is the sum of the Manhattan distances of tiles from their goal positions. In Figure, only the tiles 5, 6 and 8 are misplaced and their distances from the goal positions are respectively 2, 2 and 1. Therefore, h(Initial state) = 2 + 2 + 1 = 5:

Example 2

Examples

1	2	3
4	5	6
7	8	

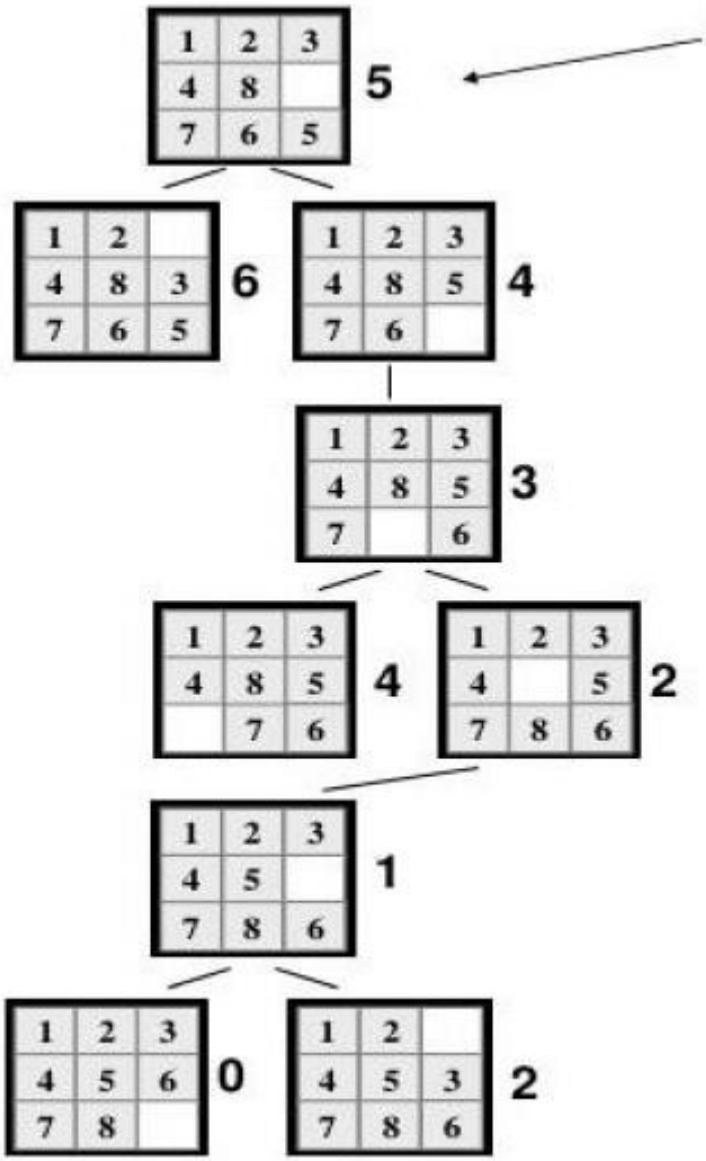


Fig shows the various states reached by applying the various operations specified in the hill-climbing algorithm. The numbers written alongside the grids are the values of the heuristic function.

Note that the problem is a minimization problem. The value of the heuristic function at the goal state is 0 which is the minimum value of the function. Note further that once we find an operation which produces a better value than the value in current state we do proceed to that state without considering whether there is any other move which produces a state having a still better value.

Types of Hill Climb Algorithm

2. Steepest-ascent hill climbing

In steepest-ascent hill climbing, we consider all the moves from the current state and selects the best as the next state. In the basic hill climbing, the first state that is better than the current state is selected.

Steps involved in Steepest-Ascent hill climbing algorithm

Step 1: Evaluate the initial state, if it is goal state then return success and stop, else make the current state as your initial state.

Step 2: Loop until a solution is found or the current state does not change. Let **S** be a state such that any successor of the current state will be better than it. For each operator that applies to the current state;

- Apply the new operator and generate a new state.
- Evaluate the new state.
- If it is goal state, then return it and quit, else compare it to the **S**.
- If it is better than **S**, then set new state as **S**.
- If the **S** is better than the current state, then set the current state to **S**.

Step 3: Exit.

Advantages and disadvantages of hill climbing

Advantages

- Hill climbing is very useful in routing-related problems like travelling salesmen problem, job scheduling, chip designing, and portfolio management.
- It is good in solving optimization problems while using only limited computation power.
- It is sometimes more efficient than other search algorithms.
- Even though it may not give the optimal solution, it gives decent solutions to computationally challenging problems.

Disadvantages

- Both the basic hill climbing and the steepest-ascent hill climbing may fail to produce a solution. Either the algorithm terminates without finding a goal state or getting into a state from which no better state can be generated. This will happen if the programme has reached either a local maximum, a ridge or a plateau.

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).

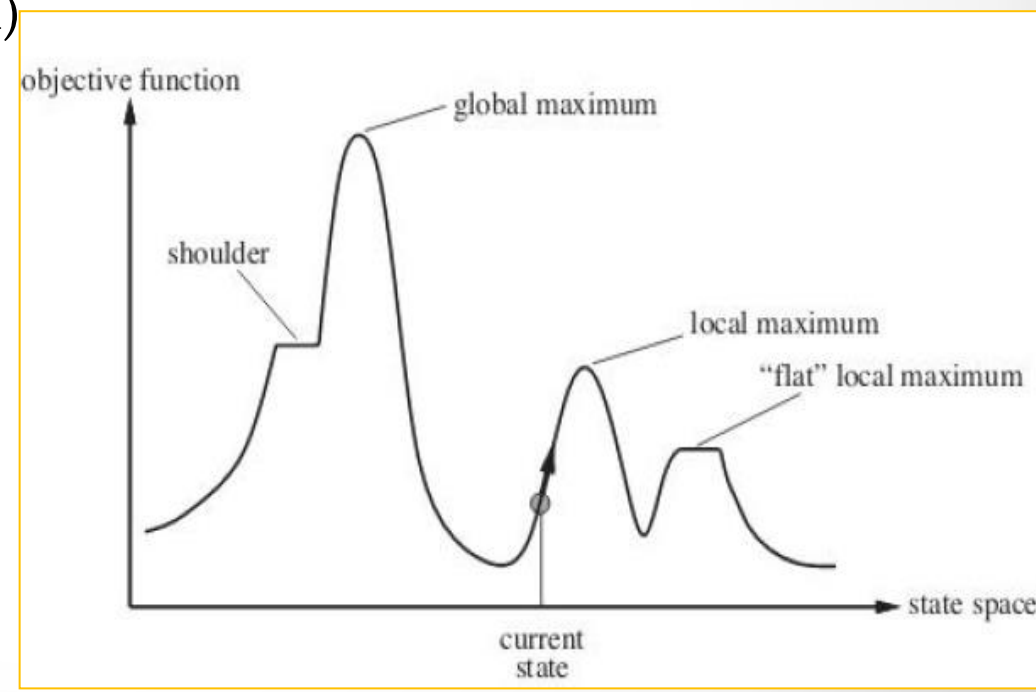
X-axis : denotes the state space ie states or configuration our algorithm may reach.

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)

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 because at this state, objective function has highest value.



Local and global maxima

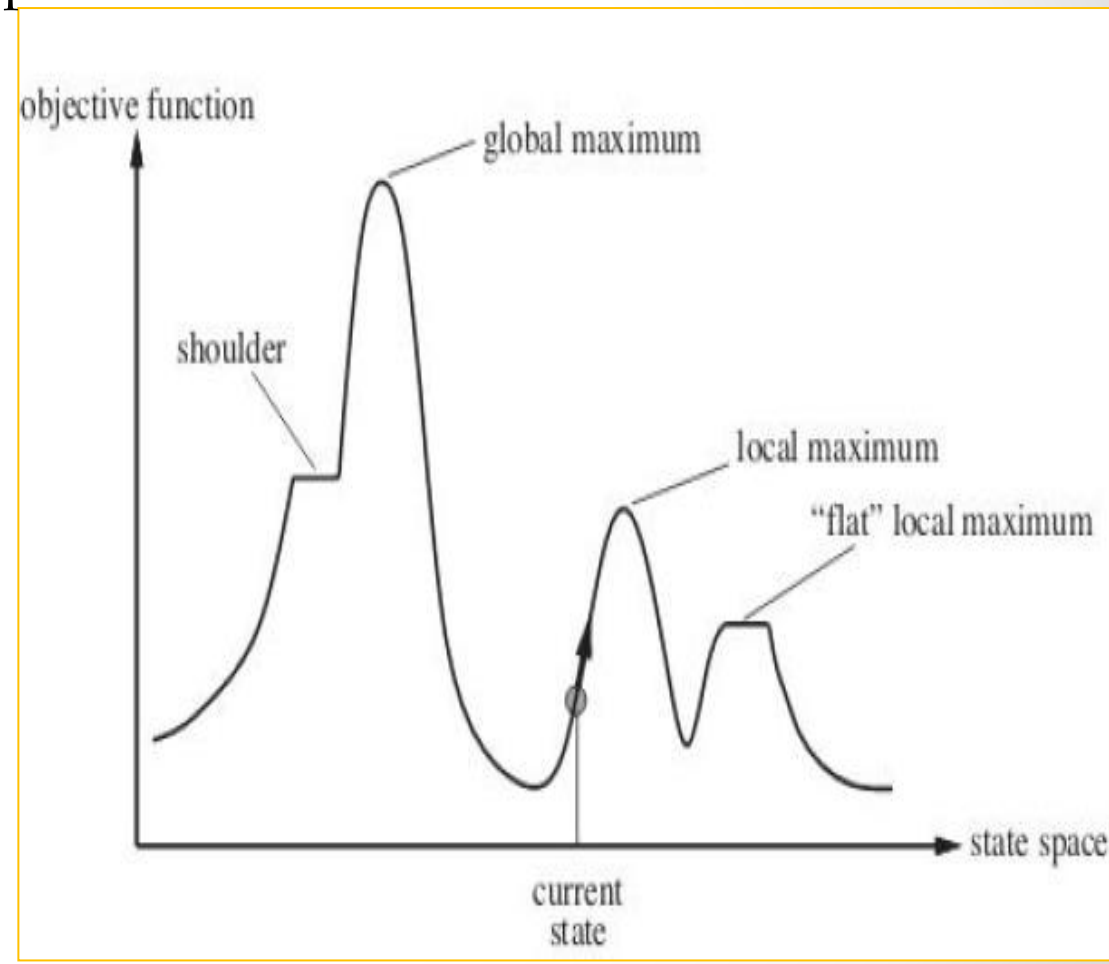
Different regions in the State Space Diagram

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

Ridge : It is region which is higher than its neighbors 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.

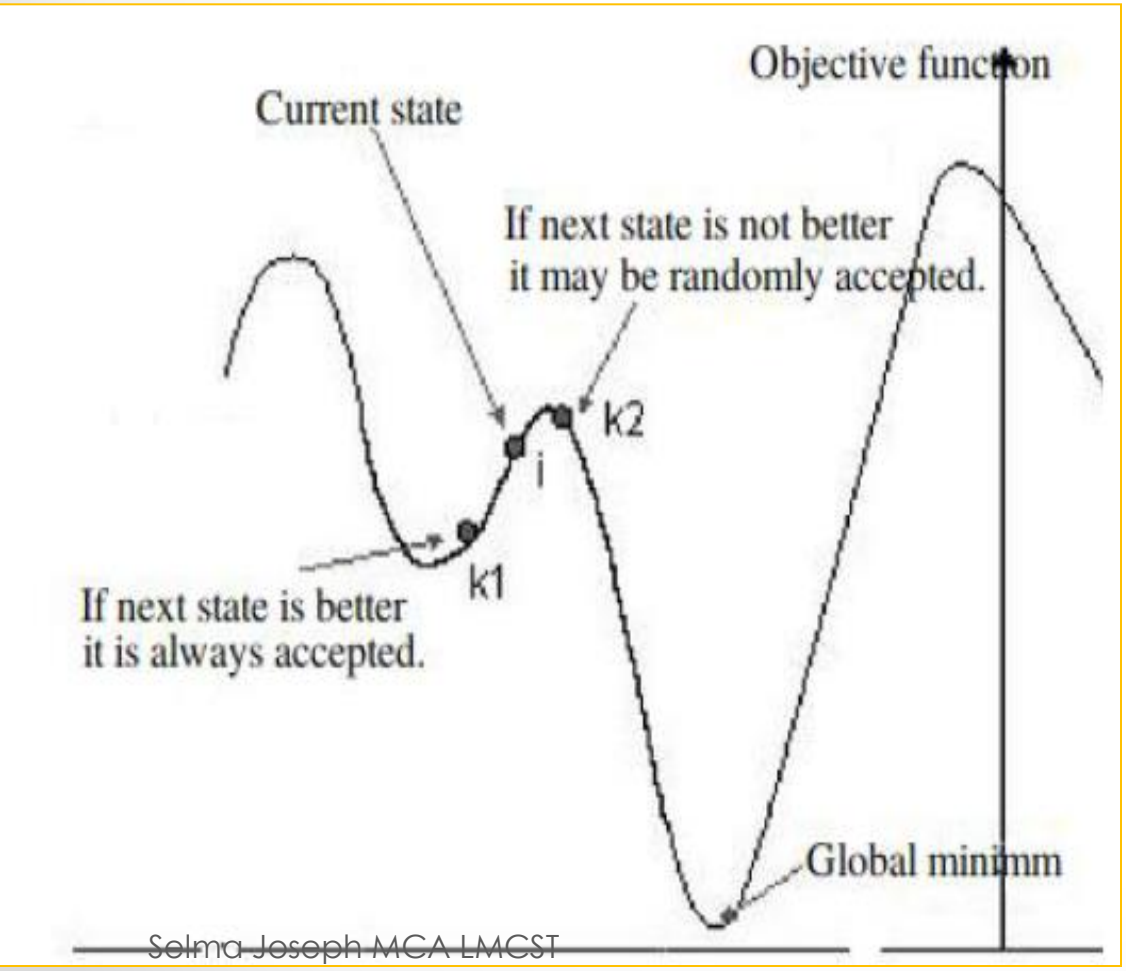
Shoulder : It is a plateau that has an uphill edge.



A ridge: The ridge is the curved line at the top joining the local maxima

Simulated annealing

Simulated annealing is a variation of hill climbing. In simulated annealing, some down-hill moves may be made at the beginning of the process. This to explore the whole space at the beginning so that we do not land in a local maximum, or plateau or ridge instead of the global maximum. The algorithm is somewhat analogous to a process known as "annealing" in metallurgy.



Annealing

In mechanical term **Annealing** is a process of hardening a metal or glass to a high temperature then cooling gradually, so this allows the metal to reach a low-energy crystalline state. The same process is used in simulated annealing in which the algorithm picks a random move, instead of picking the best move. If the random move improves the state, then it follows the same path. Otherwise, the algorithm follows the path which has a probability of less than 1 or it moves downhill and chooses another path.

A* Algorithm in Best First Search

Step1: Place the starting node in the OPEN list.

Step 2: Check if the OPEN list is empty or not, if the list is empty then return failure and stops.

Step 3: Select the node from the OPEN list which has the smallest value of evaluation function (g+h), if node N is goal node then return success and stop, otherwise

Step 4: Expand node N and generate all of its successors, and put N into the closed list. For each successor N', check whether N' is already in the OPEN or CLOSED list,

if not, then compute evaluation function for N' and place into Open list.

Step 5: Else if node N' is already in OPEN and CLOSED, then it should be attached to the back pointer which reflects the lowest g(N') value.

Step 6: Return to **Step 2**.