3.2.2 Hill Climbing Search

- This algorithm generally moves up in the direction of increasing value that is-uphill. It breaks its "moving up loop" when it reaches a "peak" where no neighbour has a higher value.
- It does not maintain a search tree. It stores current node data structure. This node records the state and its objective function value. Algorithm only look out for immediate neighbours of current state.
- It is similar to greedy local search in a sense that it considers a current good neighbour state without thinking ahead.
- Greedy algorithm works very well as it is very easy to improve bad state in hill climbing.

3.2.2.1 Algorithm for Hill Climbing

- The algorithm for hill climbing is as follows : -
  1) Evaluate the initial state. If it is goal state quit, otherwise make current state as initial state.
  2) Select a new operator that could be applied to this state and generate a new state.
  3) Evaluate the new state. If this new state is closer to the goal state than current state make the new state as the current state. If it is not better, ignore this state and proceed with the current state.
  4) If the current state is goal state or no new operators are available, quit. Otherwise repeat from 2.

3.2.2.2 Problems with Hill Climbing

1) Local maxima - can't see higher peak.
2) Shoulder - can't see the way out.

Local maxima - It is a state where we have climbed to the top of the hill, and missed on better solution.

It is the mountain - A state that is better than all of its neighbours, but not better than some other states further away. **[Shown in Fig. 3.2.3]**

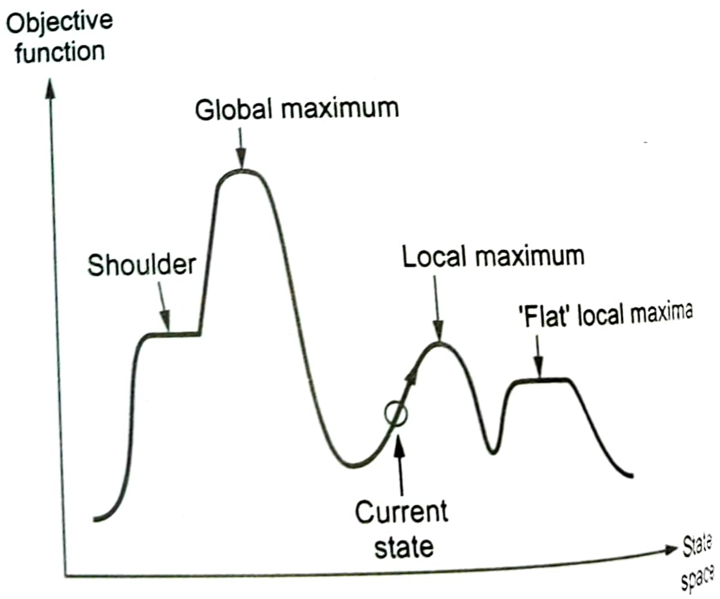

Fig. 3.2.2 Various stages in hill climbing search

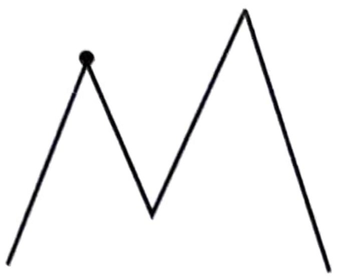

Fig. 3.2.3 Local maxima

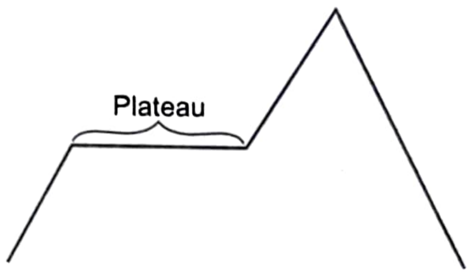

Fig. 3.2.4 Plateau

Plateau : It is a state where everything around is about as good as where we are currently. In other words a flat area of the search space in which all neighbouring states have the same value. **[Shown in Fig. 3.2.4]**

Ridges : In this state we are on a ridge leading up, but we can't directly apply an operator to improve the situation, so we have to apply more than one operator to get there. **[Shown in Fig. 3.2.5]**

Illustration of ridges : The grid of states (dark circles) is superimposed on a ridge rising from left to right, creating a sequences of local maxima that are not directly connected to each other. From each local maximum all the available actions point downhill.

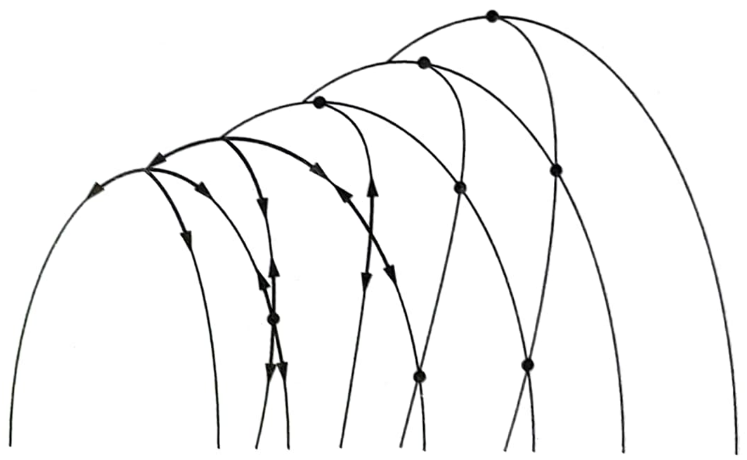

Fig. 3.2.5 Ridges

3.2.2.3 Solving Problems Associated with Hill Climbing

- All the above discussed problems could be solved using methods like backtracking, making big jumps (to handle plateaus or poor local maxima), applying multiple rules before testing (helps with ridges) etc. Hill climbing is best suited to problem where the heuristic gradually improves, the closer it gets to the solution; it works poorly where there are sharp drop-offs. It assumes that local improvement will lead to global improvement.

3.2.2.4 Example for Local Search

- Consider the 8-queens problem :
- A complete-state formulation is used for local search algorithms. In 8-queens problem, each state has 8-queens on the board one per column. There are two functions related with 8-queens.

1) The successor function : It is function which returns all possible states which are generated by a single queen move to another cell in the same column. The total successor of the each state $8 \times 7 = 56$.

2) The heuristic cost function : It is a function 'h' which hold the number of attacking pair of queens to each other either directly or indirectly. The value is zero for the global minimum of the function which occurs only at perfect solutions.

3.2.2.5 Advantages of Hill Climbing

- Hill climbing is an optimization technique for solving computationally hard problems.
- It is best used in problems with the property that the" state description itself contains all the information needed for a solution".
- The algorithm is memory efficient since it does not maintain a search tree. It looks only at the current state and immediate future states.
- In contrast with other iterative improvement algorithms, hill-climbing always attempts to make changes that improve the current state. In other words, hill-climbing can only advance if there is a higher point in the adjacent landscape.
- It is often useful when combined with other methods, getting it started right in the immediate general neighbourhood.

3.2.2.6 Variations of Hill Climbing

- Many variants of hill-climbing have been invented as discussed below.

1) Stochastic hill climbing : Chooses at random from among the uphill moves ; the probability of selections can vary with the steepness of the uphill move.

2) First choice hill climbing : Implements stochastic hill climbing by generating successors randomly until one is generated that is better than the current state. This is a good strategy when a state has many (e.g. thousands) of successors.

3) Random restart hill climbing : Adopts the well known saying "If at first you don't succeed, try, try again". It conducts a series of hill climbing searches from randomly generated initial states, stopping when a goal is found.