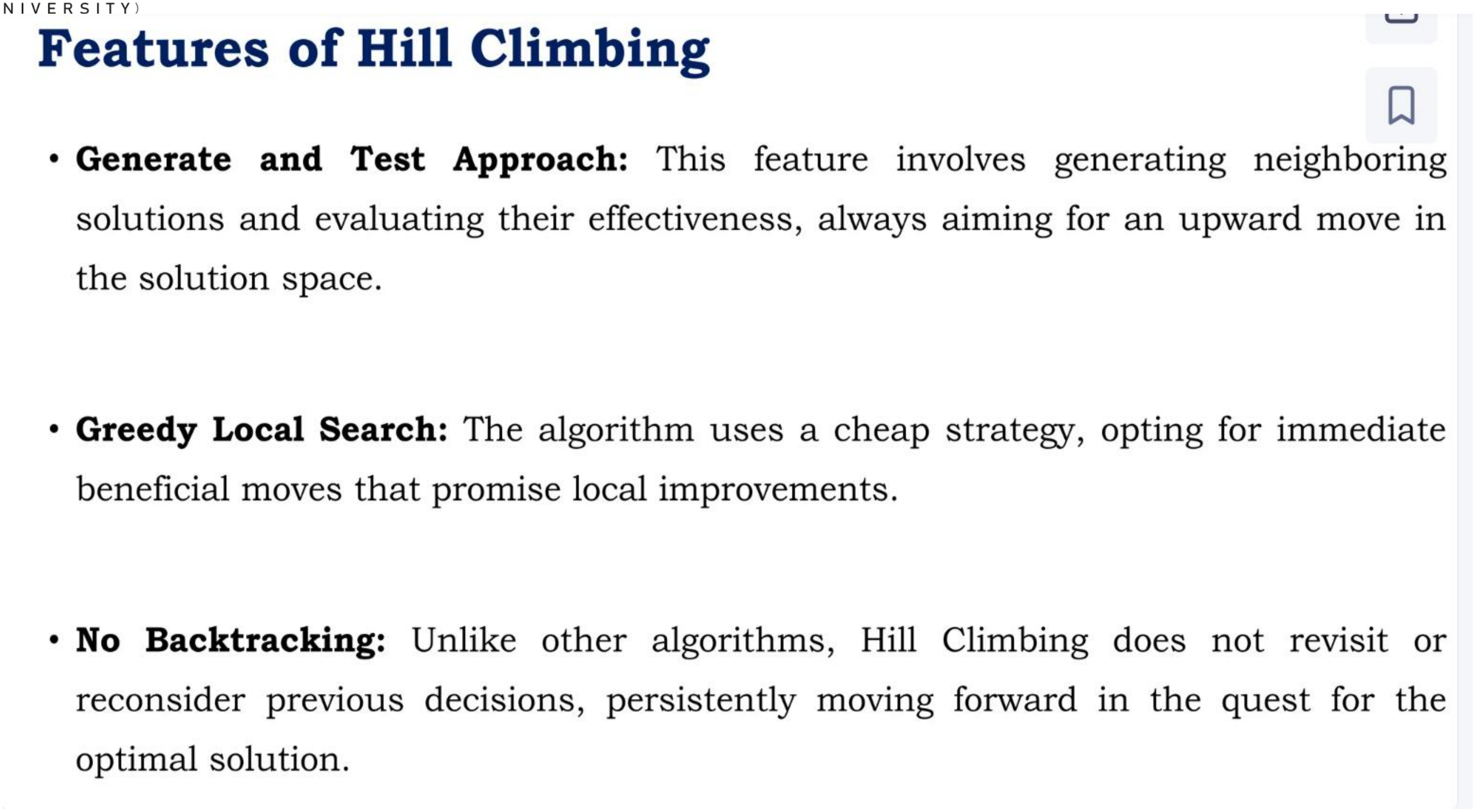
ARTIFICIAL INTELLIGENCE & MACHINE LEARNING

SESSION NO : 8

HEURISTIC SEARCH
HILL CLIMBING,VARIANTS.

Hill climbing

- Hill climbing algorithm is a local search algorithm
- It continuously moves in the direction of increasing elevation/value to find the peak of the mountain or best solution to the problem.
- It terminates when it reaches a peak value where no neighbor has a higher value.
- It is a heuristic search technique
- It is also called greedy local search as it only looks to its good immediate neighbor state and not beyond that
- A node of hill climbing algorithm has two components which are **state and value.**

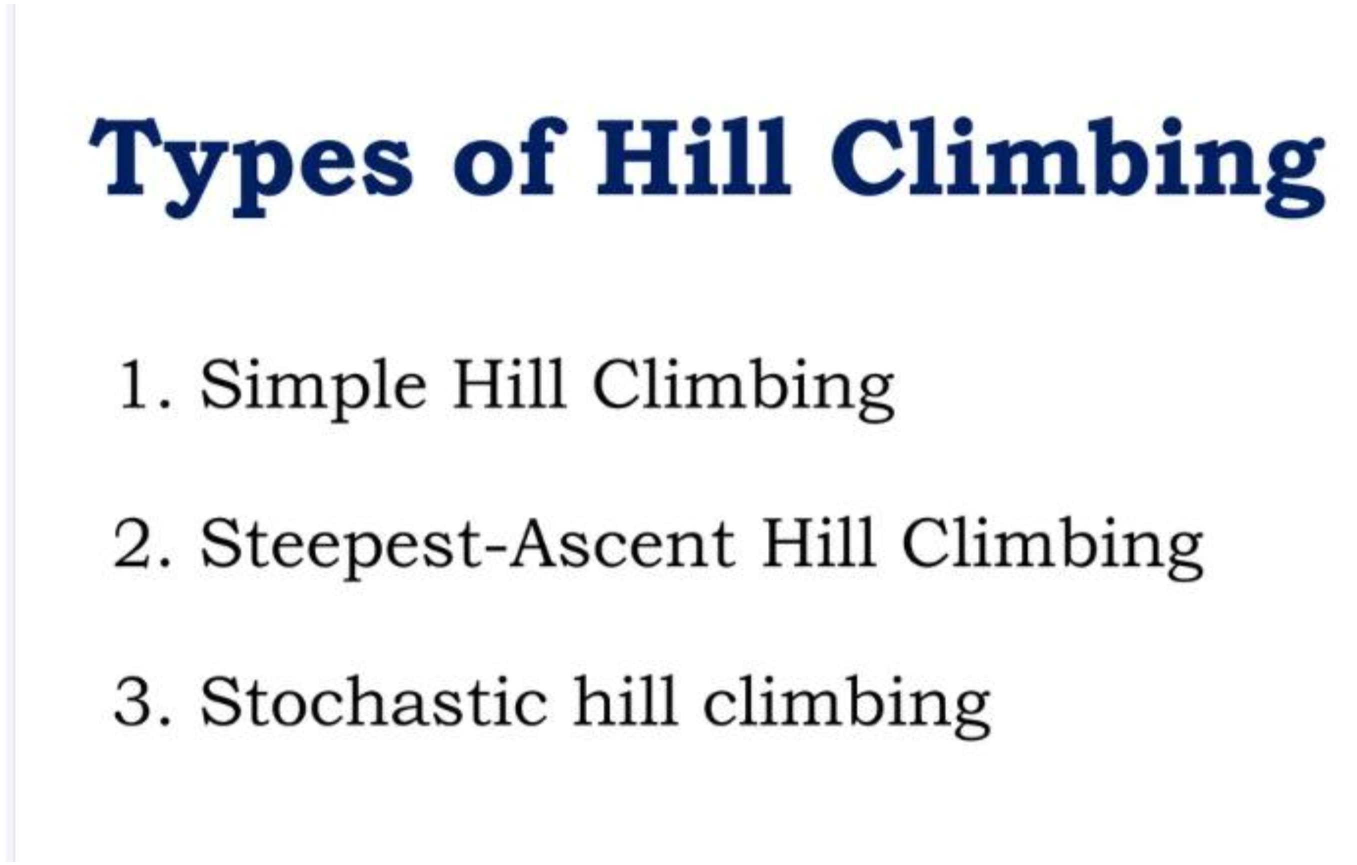


Features of Hill Climbing

- **Generate and Test Approach:** This feature involves generating neighboring solutions and evaluating their effectiveness, always aiming for an upward move in the solution space.

- **Greedy Local Search:** The algorithm uses a cheap strategy, opting for immediate beneficial moves that promise local improvements.

- **No Backtracking:** Unlike other algorithms, Hill Climbing does not revisit or reconsider previous decisions, persistently moving forward in the quest for the optimal solution.



Types of Hill Climbing

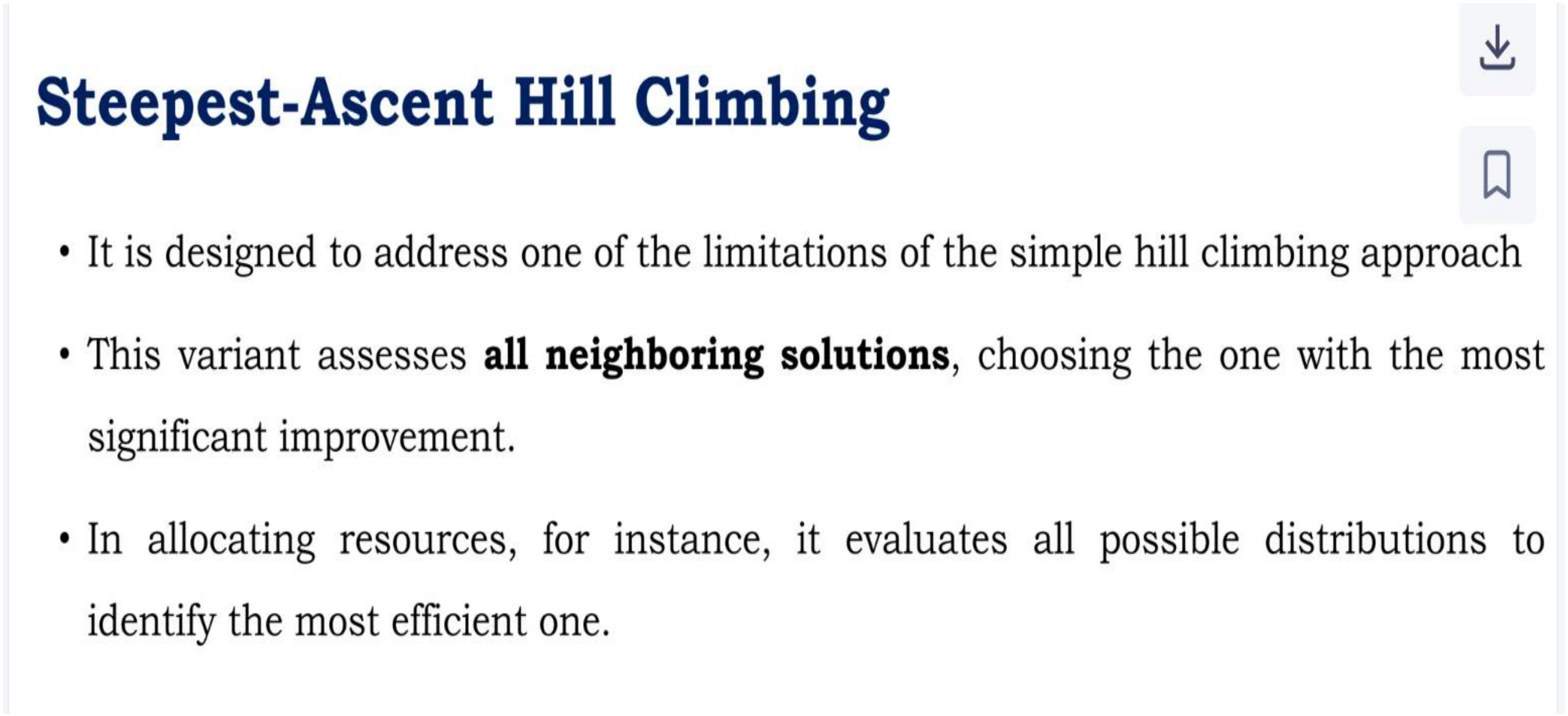
1. Simple Hill Climbing
2. Steepest-Ascent Hill Climbing
3. Stochastic hill climbing

Simple Hill Climbing

- This version evaluates neighboring solutions and selects the **first one that improves** the current state.
- For example, optimizing delivery routes might pick the first alternate route that shortens delivery time, even if it's not optimal.

Simple Hill Climbing Algorithm:

- Step 1: Start with an initial state.
- Step 2: Check if the initial state is the goal. If so, return success and exit.
- Step 3: Enter a loop to search for a better state continuously.
    - Select a neighboring state within the loop by applying an operator to the current state.
    - Evaluate this new state:
        - If it's the goal state, return success and exit.
        - If it's better than the current state, update the current state to this new state.
        - If it's not better, discard it and continue the loop.
- Step 4: End the process if no better state is found and the goal isn't achieved.



Steepest-Ascent Hill Climbing

- It is designed to address one of the limitations of the simple hill climbing approach
- This variant assesses **all neighboring solutions**, choosing the one with the most significant improvement.
- In allocating resources, for instance, it evaluates all possible distributions to identify the most efficient one.

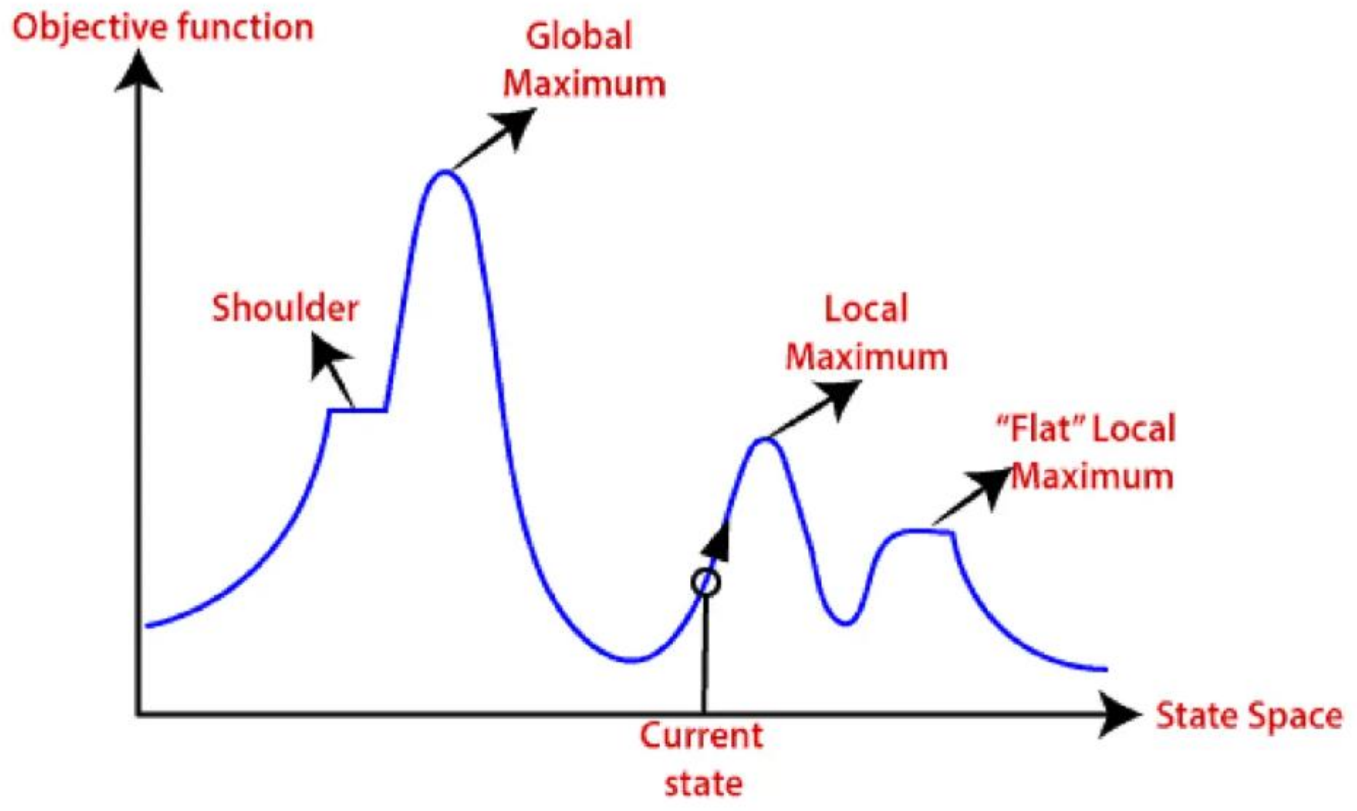
Steepest-Ascent Hill Climbing Algorithm:

- **Step 1:** Evaluate the initial state. If it's the goal, you can return success; otherwise, set it as the current state.
- **Step 2:** Repeat until a solution is found or no further improvement is possible.
- Initialize "BEST_SUCCESSOR" as the best potential improvement over the current state.
- For each operator, apply to the current state, then evaluate the new state.
  - If it's the goal, return success.
  - If better than "BEST_SUCCESSOR," update "BEST_SUCCESSOR" to this new state.
- If "BEST_SUCCESSOR" is an improvement, update the current state.
- **Step 3:** Stop the algorithm if no solution is found or further improvement possible.

Stochastic Hill Climbing

- Stochastic Hill Climbing **doesn’t look at all its neighboring** nodes to check if it is better than the current node instead, it **randomly selects** one neighboring node, and based on the pre-defined criteria it decides whether to go to the neighboring node or select an alternate node.

State-space Diagram for Hill Climbing and Analysis



A one-dimensional state-space landscape in which elevation

Different Regions in the State Space Diagram

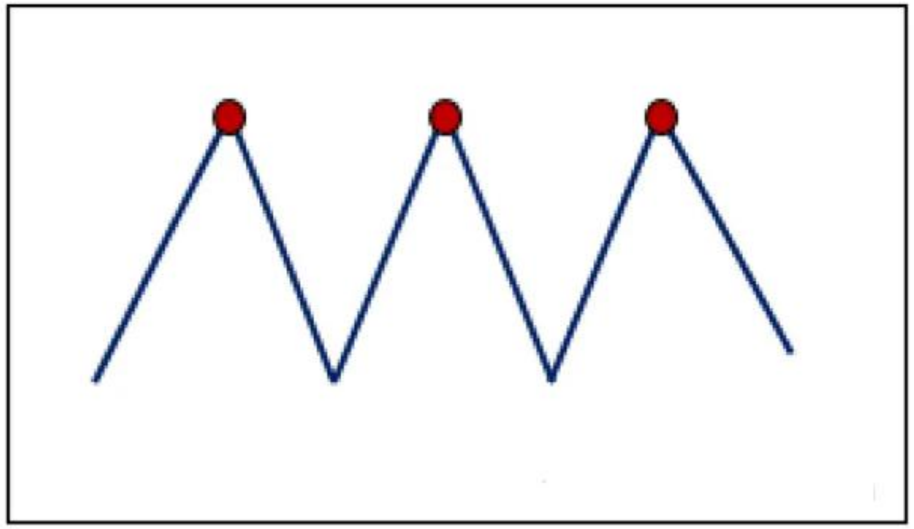
- **Local Maximum:** As the diagram makes clear, this is the state that is marginally superior to its neighboring states but never higher than the highest state.
- **Global maximum:** Its cost function value is at its highest, and it is the highest state in the state space.
- **Current State:** Current State is the state where the agent is present currently
- **Flat local maximums** are what happens when all the neighboring states have the same value and can be visualized as flat spaces (as shown in the diagram).
- **Shoulder region:** A region with an upward edge, it is also one of the issues with algorithms for climbing hills.

Problems in Different Regions in Hill climbing

- **1. Local Maximum**
- The algorithm terminates when the current node is local maximum as it is better than its neighbours. However, there exists a global maximum where objective function value is higher
- **Solution:** Back Propagation can mitigate the problem of Local maximum as it starts exploring alternate paths when it encounters Local Maximum

Problems in Different Regions in Hill climbing

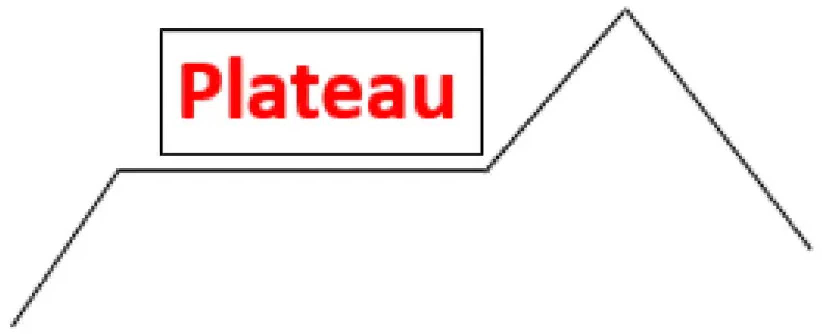
- **2. Ridge**
- Ridge occurs when there are multiple peaks and all have the same value or in other words, there are multiple local maxima which are same as global maxima



- **Solution**: Ridge obstacle can be solved by moving in several directions at the same time

Problems in Different Regions in Hill climbing

- **3. Plateau**
- Plateau is the region where all the neighbouring nodes have the same value of objective function so the algorithm finds it hard to select an appropriate direction.



- **Solution:** Plateau obstacle can be solved by taking making a big jump from the current state which will land you in non-plateau region

SIMULATED ANNEALING

- Simulated Annealing is a modified version of stochastic hill climbing.
- It is a heuristic search algorithm applied to achieve optimization in artificial intelligence issues.
- It explores the search space and avoids local optimum by employing a probabilistic method to accept a worse solution with a given probability.
- In simulated annealing, "temperature" is a parameter that controls the probability of accepting worse solutions as the algorithm progresses. It is analogous to the temperature in the physical annealing process where materials are heated and then slowly cooled to remove defects and find a low-energy state.

Example:
https://www.youtube.com/watch?v=uK6nbn7hArQ

SIMULATED ANNEALING

Key Points about Temperature in Simulated Annealing:

- **High Temperature**: At the start, the temperature is set high, allowing the algorithm to accept worse solutions with higher probability. This helps in exploring the solution space more thoroughly and avoids getting trapped in local optima.
- **Cooling Schedule**: The temperature is gradually reduced according to a cooling schedule. This schedule dictates how quickly the temperature decreases over time.
- **Low Temperature**: As the temperature decreases, the probability of accepting worse solutions decreases, allowing the algorithm to converge to an optimal or near-optimal solution.

steps of the simulated annealing algorithm

- Start with an initial solution.
- Set the initial temperature to a high value.
- Repeat the following steps until the stopping criterion is met:
  - Generate a new solution by making a small modification to the current solution.
  - Evaluate the objective function of the new solution.
  - If the new solution improves the objective function, accept it as the new current solution.
  - If the new solution does not improve the objective function, accept it with a probability that depends on the difference between the objective function values of the current and new solutions and the current temperature.
  - Decrease the temperature according to a cooling schedule.

steps of the simulated annealing algorithm

- Return the current solution as the final solution.
- The main principle of the simulated annealing algorithm is to control the level of randomness in the search process by altering the temperature parameter.
- High temperatures enable the algorithm to explore new regions of the search space by increasing its propensity to accept non-improving moves. The algorithm becomes more selective and concentrates on improving the solution as the temperature drops.
- Simulated annealing has been successfully applied to a wide range of optimization problems, such as the traveling salesman problem, the vehicle routing problem, and the job shop scheduling problem. However, it requires careful tuning of the

Travelling Salesman Problem

- The Traveling Salesman Problem (TSP) is a well-known example of a combinatorial optimization problem in which the goal is to determine the quickest path between the starting city and a given set of cities. No known algorithm can complete it in polynomial time because it is an NP-hard problem.
- **Local search algorithms for the TSP work** by starting with an initial solution and incrementally improving it by making small changes to it one at a time until no more advancements are possible.

The probability $P = \exp(-\Delta E/T)$

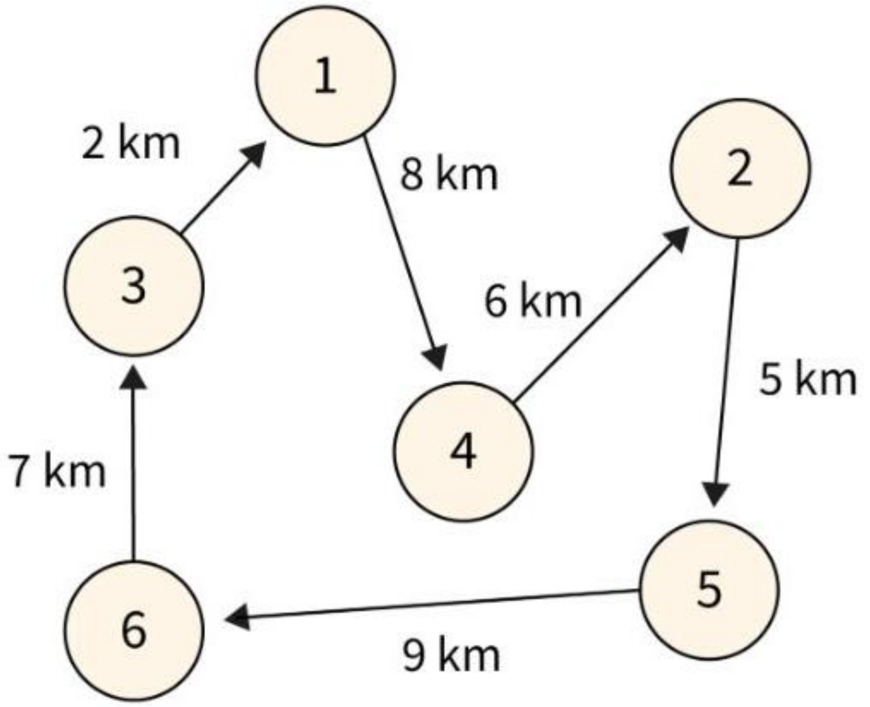
Scenario:

- Current tour cost = 120 km
- New tour cost = 130 km
- So, $\Delta E = 130 - 120 = 10$
- $T = 100$

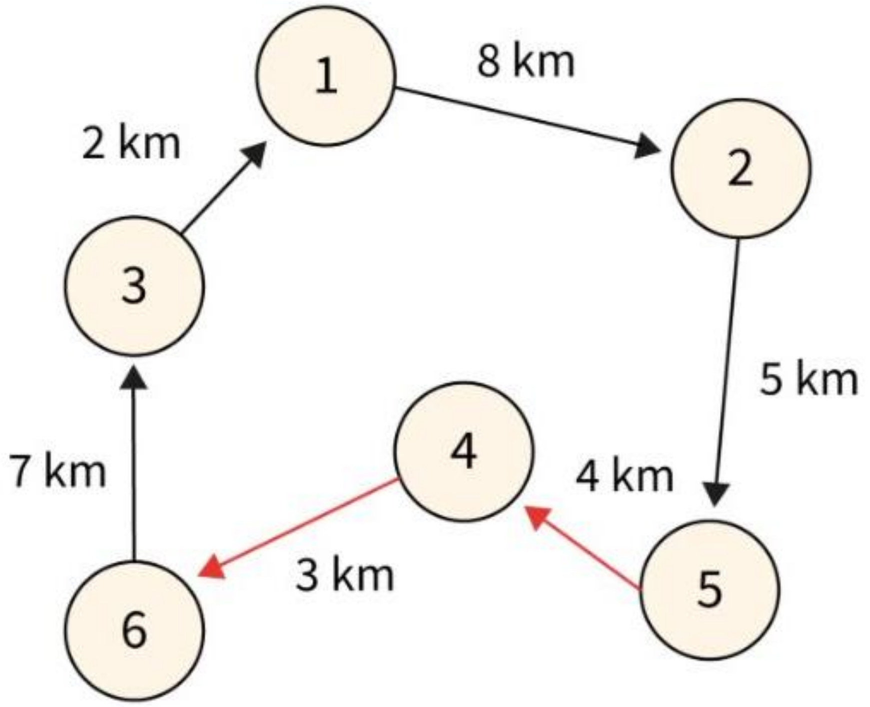
Probability Calculation:

$$P = e^{-10/100} = e^{-0.1} \approx 0.905$$

Travelling Salesman Problem



Total distance = 37km



Total distance = 31km

summary

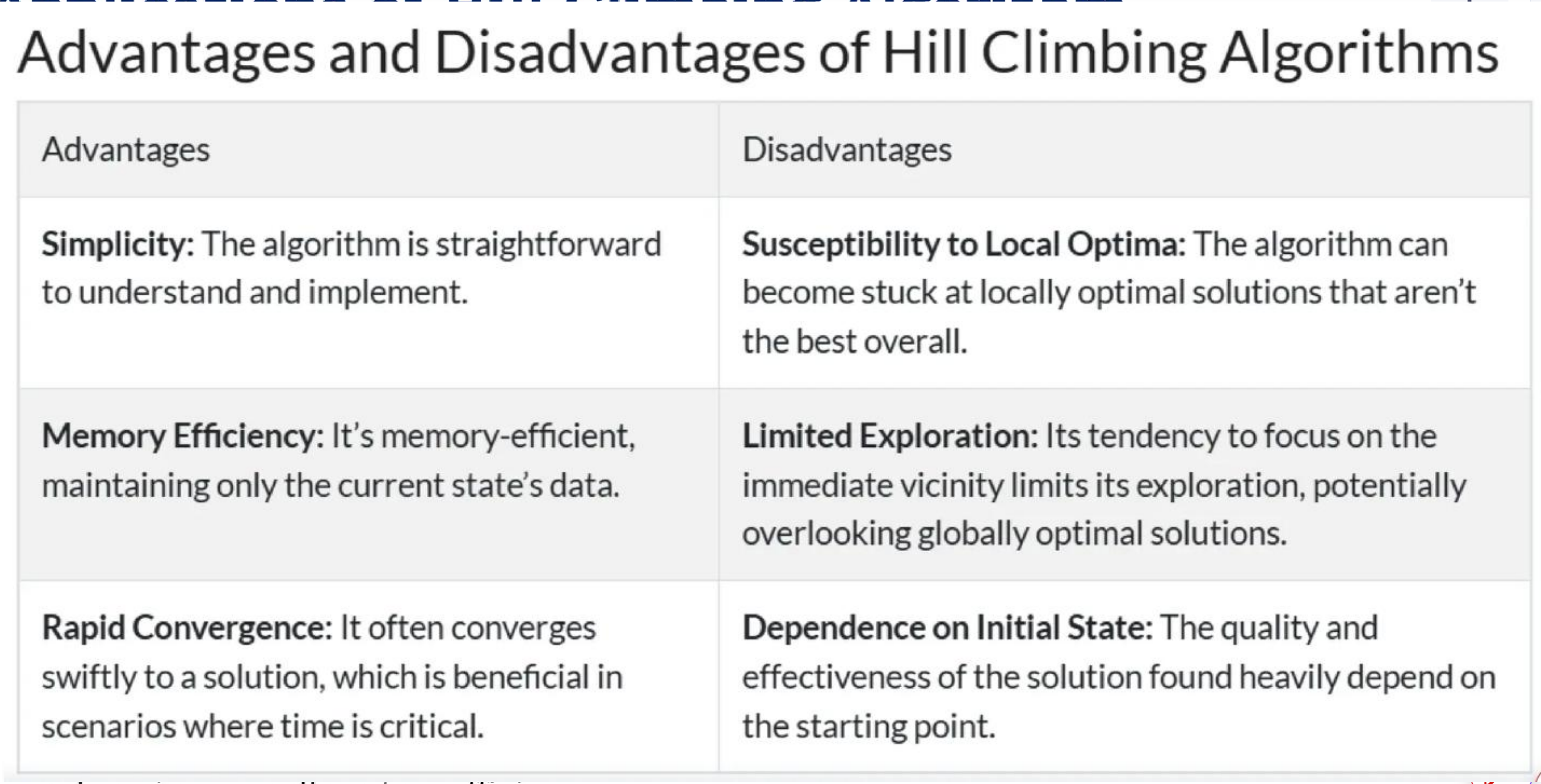
Local search algorithms iteratively improve a single solution by making small modifications.

- Local search algorithms are often simple and efficient and can handle large solution spaces.
- However, they are prone to get stuck in local optima, where the solution is not the global optimum.
- To overcome this limitation, various modifications to local search algorithms have been proposed, such as tabu search, simulated annealing, and genetic algorithms.

Advantage of Hill Climbing Algorithm

- 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 **the optimization problem** while using only limited computation power
- It is more **efficient than other search** algorithms

Applications of Hill Climbing Algorithm



Advantages and Disadvantages of Hill Climbing Algorithms

Advantages	Disadvantages
Simplicity: The algorithm is straightforward to understand and implement.	**Susceptibility to Local Optima:** The algorithm can become stuck at locally optimal solutions that aren't the best overall.
Memory Efficiency: It's memory-efficient, maintaining only the current state's data.	**Limited Exploration:** Its tendency to focus on the immediate vicinity limits its exploration, potentially overlooking globally optimal solutions.
Rapid Convergence: It often converges swiftly to a solution, which is beneficial in scenarios where time is critical.	**Dependence on Initial State:** The quality and effectiveness of the solution found heavily depend on the starting point.

enhancing overall system efficiency.

Applications of Hill Climbing Algorithm

- **Marketing:** The Hill Climbing algorithm is a game-changer for marketing managers cra top-notch strategies. It's instrumental in solving the classic Traveling-Salesman problems, optimizing sales routes, and reducing travel time. This leads to more efficient sales operations and better resource utilization.
- **Robotics:** The algorithm plays a critical role in robotics, enhancing the performance and coordination of various robotic components. This leads to more sophisticated and efficient robotic systems performing complex tasks.
- **Job Scheduling:** Within computing systems, Hill Climbing is key in job scheduling, optimizing the allocation of system resources for various tasks. Efficiently managing the distribution of jobs across different nodes ensures optimal use of computational resources, enhancing overall system efficiency.