IQRA University

CSC 471 Artificial Intelligence

Lab# 06 Local Search & Hill Climbing Algorithm

Objective:

This lab introduces the students to the concept of optimization problems. The lab further demonstrates the use of Local Search schemes such as Hill Climbing and and its variants for solving basic search problems.

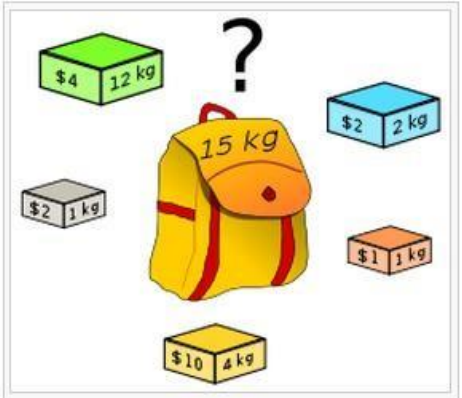
Name of Student: Faraz Alam

Roll No: 13948 **Sec.** Saturday (12:00pm-14:00pm)

Date of Experiment: 11/27/23

Lab 06: Optimization Problems

Optimization problems are a fundamental concept in artificial intelligence, aiming to find the best solution from a set of possible solutions, often within a complex and dynamic environment. These problems arise in various AI applications, from route planning and resource allocation to game playing and machine learning.



At their core, optimization problems involve exploring the solution space to identify the most favorable outcome based on a set of criteria or objectives. The goal is to efficiently navigate through the possibilities, ultimately converging on the solution that optimizes a given metric. This metric, often referred to as the "objective function," quantifies the quality of a solution.

Solving optimization search problems often involves striking a balance between exploration and exploitation, as the algorithm must navigate the trade-off between discovering new, potentially better solutions and refining existing ones. As AI continues to advance, optimization search techniques evolve, contributing to the development of more sophisticated and efficient algorithms that can tackle increasingly complex problems across diverse domains.

Local Search

Local search algorithms are optimization algorithms that explore the solution space by iteratively moving from one solution to a neighboring one, with the hope of reaching an optimal or satisfactory solution. These algorithms are particularly useful for large and complex problem spaces where it's impractical to explore all possible solutions. Instead of a global view of the entire solution space, local search algorithms focus on the immediate neighborhood of the current solution.

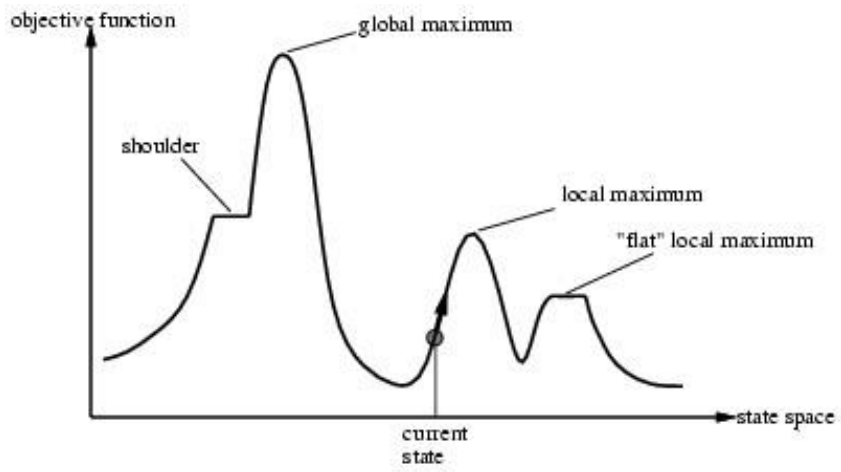
Hill-climbing search

Hill Climbing is 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 implies 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 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 reasonable time.
- The success of hill climbing depends very much on the shape of the state-space landscape: if there are few local maxima and plateau, random-restart hill climbing will find a good solution very quickly. On the other hand, many real problems have a landscape that looks more like a family of porcupines on a flat floor, with miniature porcupines living on the tip of each porcupine needle, ad infinitum.
- NP-hard problems typically have an exponential number of local maxima to get stuck on. Despite this, a reasonably good local maximum can often be found after a small number of restarts.

Algorithm

```
function HILL-CLIMBING(problem) returns a state that is a local maximum
    inputs: problem, a problem
    local variables: current, a node
                     neighbor, a node
    current ← MAKE-NODE(INITIAL-STATE[problem])
    loop do
        neighbor ← a highest-valued successor of current
        if VALUE[neighbor] ≤ VALUE[current] then return STATE[current]
        current ← neighbor
```



Variants of Hill-Climbing

- **Stochastic hill climbing**

  Chooses at random from among the uphill moves;

  The probability of selection can vary with the steepness of the uphill move.

  This usually converges more slowly than steepest ascent, but in some state landscapes it finds better solutions.

- **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.

  The hill-climbing algorithms described so far fail to find a goal when one exists because they can get stuck on local maxima.

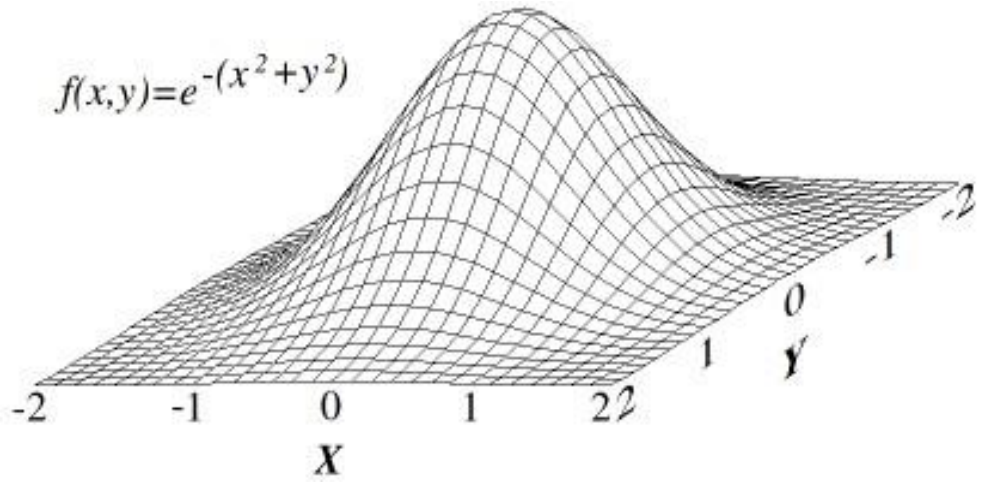
- **Random-restart hill climbing**

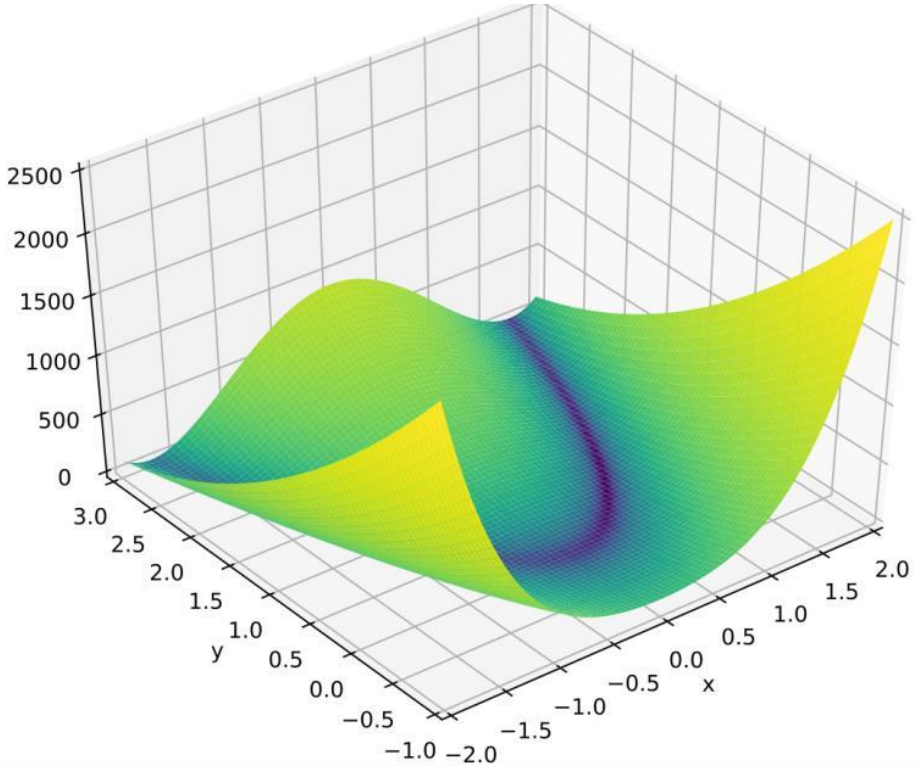
Adopts the well-known adage, "If at first you don't succeed, try, try again."

It conducts a series of hill-climbing searches from randomly generated initial state, stopping when a goal is found.

Student Exercise

Task 1





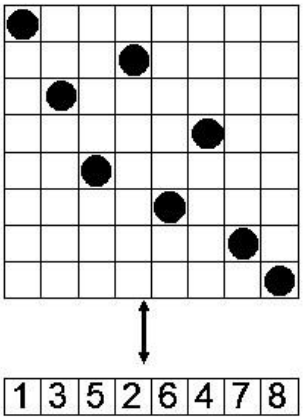
$$f(x, y) = (1 - x)^2 + 100(y - x^2)^2$$

Implement Hill climbing solutions for the given two function. The Hill Climbing solution should provide values for the x and y parameters where the value of the function maximizes. The figures

also provide the range of values the solution exist within. Restrict your search within the domain of the variables for a quicker response.

Task 2:

Implement an Hill climbing search for the 8 queen problem represented below. Using the solution representation shown in the figure below might reduce your solution space size.



- **Penalty of one queen: the number of queens she can check.**
- **Penalty of a configuration: the sum of the penalties of all queens.**
- **Note: penalty is to be minimized**
- **Fitness of a configuration: inverse penalty to be maximized**

Your current execution might get stuck at a local optimum. Implement a random restart technique hill climbing variant to improve your results.