Artificial Intelligence Lab
AL-2002

Lab 09

Instructor: Hurmat Hidayat

Semester: Spring 2023

Artificial Intelligence Lab 09

Objective

The objective of this lab is to introduce students to Local Search algorithms and its various types. In particular, the lab focuses on Hill Climbing, , and Simulated Annealing algorithms. The lab demonstrates how to implement Hill Climbing algorithm on the 8-Queens problem.

Learning Outcomes

1. Understand the concept of Local Search algorithms and its types
2. Understand the Hill Climbing algorithm and its various types
3. Implement Hill Climbing algorithm on the 8-Queens problem

Table of Contents

Local Search Algorithms

Overview of Optimization problems

Optimization is an important tool in making decisions and in analyzing physical systems. In mathematical terms, an optimization problem is the problem of finding the best solution from among the set of all feasible solutions.

Local Search Algorithms

In many optimization problems the path to a goal state is irrelevant. The goal state itself is the solution. They start from a prospective solution and then move to a neighboring solution. Local search algorithms are useful for solving pure optimization problems, in which the aim is to find the best state/node according to an objective function.

There are mainly three types of local search Algorithms

1. Hill Climbing Search
2. Local Beam Search
3. Simulated Annealing

Applications of Local Search

Local search algorithms have many important applications in optimization problems such as integrated-circuit design, factory-floor layout, job-shop scheduling, automatic programming, telecommunications network optimization, vehicle routing, and portfolio management

Hill Climbing Search

Hill Climbing is a heuristic search used for mathematical optimization problems in the field of Artificial Intelligence. So, given a large set of inputs and a good heuristic function, the algorithm tries to find the best possible solution to the problem in the most reasonable time period. This solution may not be the absolute best(global optimal maximum) but it is sufficiently good considering the time allotted.

- Hill climbing algorithm is a local search algorithm which 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.

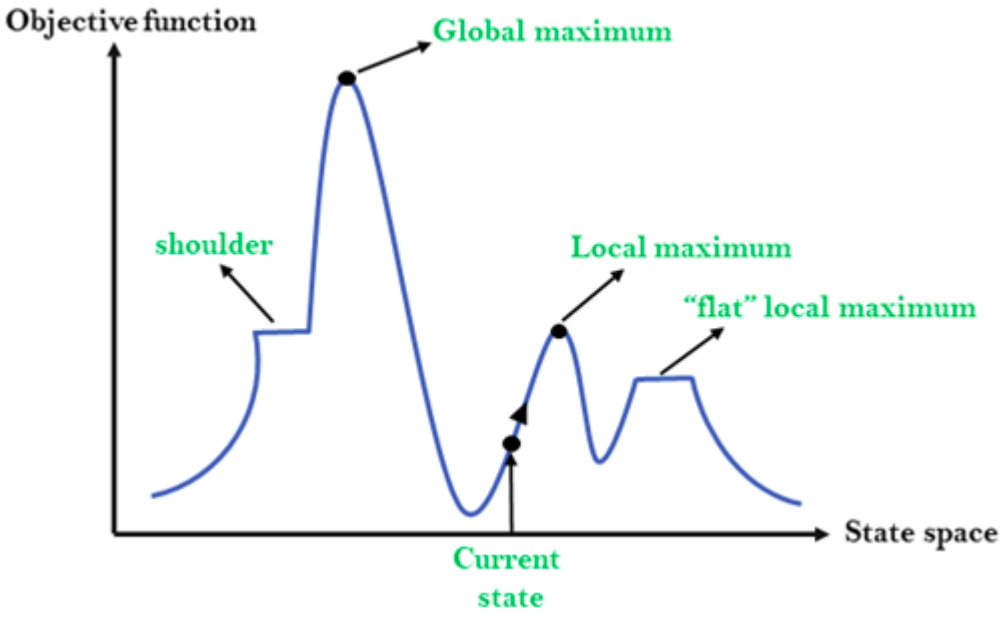
- Hill climbing algorithm is a technique which is used for optimizing the mathematical problems. One of the widely discussed examples of Hill climbing algorithm is Traveling-salesman Problem in which we need to minimize the distance traveled by the salesman.
- It is also called greedy local search as 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. Hill Climbing is mostly used when a good heuristic is available.
- In this algorithm, we don't need to maintain and handle the search tree or graph as it only keeps a single current state.

Features of Hill Climbing:

Following are some main features of Hill Climbing Algorithm:

- **Generate and Test variant:** Hill Climbing is the variant of Generate and Test method. The Generate and Test method produce feedback which helps to decide which direction to move in the search space.
- **Greedy approach:** Hill-climbing algorithm search moves in the direction which optimizes the cost.
- **No backtracking:** It does not backtrack the search space, as it does not remember the previous states.

State Diagram for hill Cimbing



Different regions in the State Space Diagram

1. **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.
2. **Global maximum :** It is the best possible state in the state space diagram. This because at this state, objective function has highest value.
3. **Plateua/flat local maximum :** It is a flat region of state space where neighboring states have the same value.
4. **Ridge :** It is region which is higher than its neighbours but itself has a slope. It is a special kind of local maximum.
5. **Current state :** The region of state space diagram where we are currently present during the search.
6. **Shoulder** : It is a plateau that has an uphill edge.

Algorithm for Simple Hill Climbing

- **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:
    1. If it is goal state, then return success and quit.
    2. else if it is better than the current state then assign new state as a current state.
    3. else if not better than the current state, then return to step 2.
- **Step 5:** Exit.

Types of Hill Climbing Algorithm:

- Simple **hill Climbing**:

  Above mentioned Algorithm.

- Steepest-Ascent **hill-climbing**:

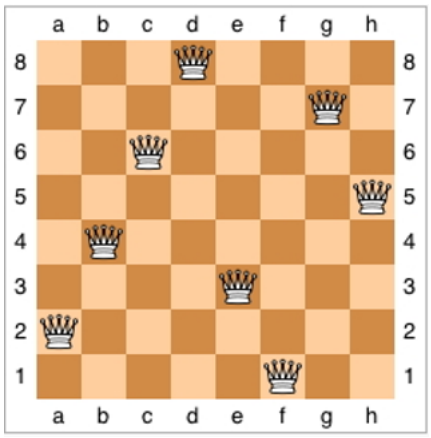
  The steepest-Ascent algorithm is a variation of simple hill climbing algorithm. This algorithm examines all the neighboring nodes of the current state and selects one neighbor node which is closest to the goal state. This algorithm consumes more time as it searches for multiple neighbors.

- Stochastic **hill Climbing**:

Stochastic hill climbing does not examine for all its neighbor before moving. Rather, this search algorithm selects one neighbor node at random and decides whether to choose it as a current state or examine another state.

Lab Task: 8-Queen Problem using Hill Climbing

The objective of the 8-queens problem is to place eight queens on a chessboard in such a way that no two queens can capture each other, which means no two queens can be on the same row, column, or diagonal.



The first observation is that, in any solution, no two queens can occupy the same column, and consequently no column can be empty. At the start we can therefore assign a column to each queen and reduce the problem to the simpler task of finding an appropriate row.

Statement of the Problem:

Consider an 8 × 8 chessboard. Place In queens on the board such that no two queens are attacking each other. The queen is the most powerful piece in chess and can attack from any distance horizontally, vertically, or diagonally. Thus, a solution must place the queens such that no two queens are in the same row, the same column, or along the same diagonal.

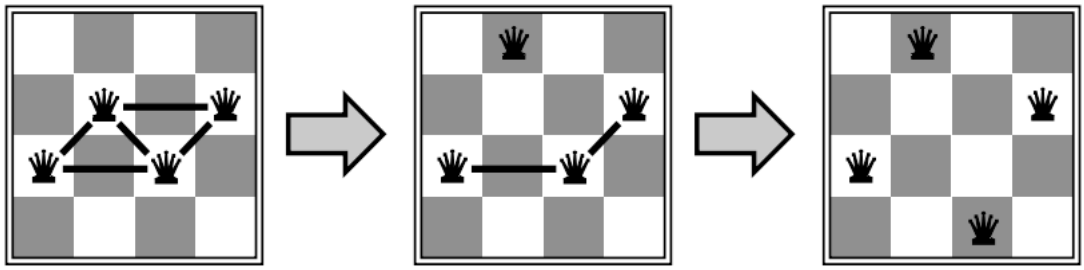
Implementation Steps

1) Create a 2D array to represent the chessboard and initialize it with zeros to represent no queen on the board.
2) Write a function that randomly places a queen in the array by setting the position index to 1.
3) Write a function to calculate the objective function, which is the number of queens attacking each other.

4) Write a function to search for the solution using the hill climbing algorithm.
   a) Initializing the starting solution by randomly placing queens on the board.
   b) Evaluating the objective function for the starting solution.
   c) Generating a set of candidate solutions by applying some neighborhood functions to the current solution.
   d) Evaluating the objective function for each candidate solution.
   e) Choosing the best candidate solution as the new current solution.
   f) Continuing this process until a stopping criterion is met.
5) Return the best solution found during the search, which represents a placement of 8 queens on the chessboard such that no queen attacks another.

Note:

The heuristic cost function "h" is the number of pairs of queens that are attacking each other, either directly or indirectly.



Hill Climbing

- Start with a random configuration.
- Calculate h(board) as the number of pairs of attacking queens.
- Consider all 56 (for N = 8) possible next boards formed by moving any one queen to any other row in its column.
- Evaluate h(board) for each of these possibilities.
- If none is better than the current h, quit (local min). Else, take the best as the new current board and repeat.

Bonus Task: 8-Queen Problem using Simulated Annealing