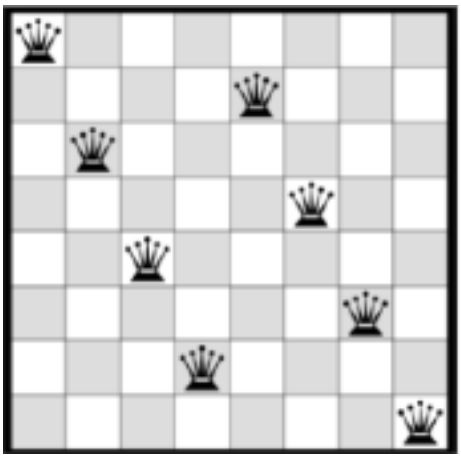
Hill Climbing Search for N Queens

Rahul Ratra

Mohit Varma

Grishma Verma

Problem Formulation: The N Queen is the problem of placing N chess queens on an N×N chessboard so that no two queens attack each other. Here we will place each given queen in a different column such that it doesn't threaten any other queen in any manner, not in a diagonal or side way or up and down.



Hill Climbing Search and its variants:

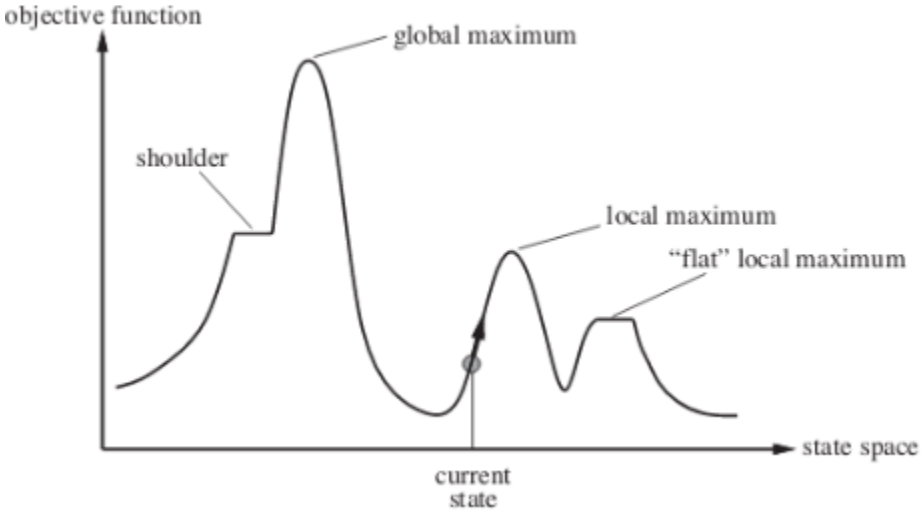
In Hill-Climbing technique, starting at the base of a hill, we walk upwards until we reach the top of the hill. In other words, we start with initial state and we keep improving the solution until it's optimal.

The workflow of hill climbing looks as follows.

1.First we define the current state as an initial state
2. Then we loop until the goal state is achieved or no more operators can be applied on the current state:

- First, we apply an operation to current state and get a new state
- Now we compare the new state with the goal
- We quit if the goal state is achieved
- After this we evaluate new state with heuristic function and compare it with the current state
- If the newer state is closer to the goal compared to current state, update the current state

It reaches the goal state with iterative improvements. In Hill-Climbing algorithm, finding goal means reaching the top of the hill.



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

Global Maximum: It is the best possible state in the state space diagram. This is because objective function has highest value at this state.

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 value of objective function is higher than its neighbors

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.

The problem due to local maxima can be solved by following methods:

Steepest-Ascent Hill-Climbing:
This differs from the basic Hill climbing algorithm by choosing the best successor rather than the first successor that is better. This indicates that it has elements of the breadth first algorithm.

Hill Climbing with sideways moves:
When stuck on a ridge or plateau (i.e., all successors have the same value), allow it to move anyway hoping it is a shoulder and after some time, there will be a way up.

Random-restart hill-climbing:
Random-restart hill-climbing: Random-restart hill-climbing conducts a series of hill-climbing searches from randomly generated initial states, running each until it halts or makes no

discernible progress. This enables comparison of many optimization trials and finding a most optimal solution thus becomes a question of using sufficient iterations on the data.

Program structure:

1.def get_h(self)

It returns heuristic value of a particular object

2.def get_state(self)

It turns the matrix of that particular object

3.def child_nodes(i,j,state)

Input parameters:

i,j is the position of 1(queen)

state : 8*8 board

Returns the possible solution state for every queen

4.def heuristic(state)

state: object representation of 8*8 matrix which stores the h value It returns heuristic value of particular board

5.def find_queen(parent_state)

parent state:object representation of the parent state

it returns the child nodes of a chess board

6.def sideway(state)

state: object representation of state

it performs the side way operation

it returns the state if true and false if not

7.def hill_climbing(state)

state: object representation of state it performs the entire

8.def get_Input(length,sample_size) length:chess board size

sample size: number of samples

9.def get_Input(length,sample_size) length:size of chess board sample_size:number of samples

10.def Random_restart(length) length: size of chess board

Implementation of Steepest Ascent Hill Climbing:

a)SuccessRate:14.49%
b)FailureRate:85.51%
c)Average steps when it succeeds:4.03
d)Average steps when it fails:3.005

Implementation of Hill Climb Sideway Movement:

a)SuccessRate:95.8%
b)Failure Rate:4.2%
c)Average steps when it succeeds:18.61
d)Average steps when it fails:58.285

Random Restart hill climbing:

a)Average number of restarts required without sideway moves:5.7
b) Average number of steps required without sideway moves:25
c)Average number of restarts used with sideway moves:1.7
d)Average number of steps required with sideway moves:14.29

Snapshots of Steps for Hill Climbing .

```
rahuls-MacBook-Pro:Hill_climbing3 rahulratra$ python3 HillClimbing.py
enter the no:of samples to find the solution 4
enter the size of board8
state:
[[0 0 0 0 0 0 0 0]
 [0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 1 0 0 1 0 1]
 [1 1 0 0 0 0 0 0]
 [0 0 0 1 0 0 1 0]
 [0 0 0 0 0 0 0 0]]
heuristic 6
[[0 0 1 0 0 0 0 0]
 [0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 1 0 1]
 [1 1 0 0 0 0 0 0]
 [0 0 0 1 0 0 1 0]
 [0 0 0 0 0 0 0 0]]
heuristic 4
[[0 0 1 0 0 0 0 0]
 [0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 0 0 0 1 0 0]
 [1 1 0 0 0 0 0 0]
 [0 0 0 1 0 0 1 0]
 [0 0 0 0 0 0 0 0]]
heuristic 2
[[0 0 1 0 0 0 0 0]
 [0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 0 0 0 1 0 0]
 [0 1 0 0 0 0 0 0]
 [0 0 0 1 0 0 1 0]
 [1 0 0 0 0 0 0 0]]
heuristic 1
[[0 0 1 0 0 0 0 0]
 [0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 0 0 0 1 0 0]
 [0 1 0 1 0 0 0 0]
 [0 0 0 0 0 0 1 0]
 [1 0 0 0 0 0 0 0]]
heuristic 0
[[0 0 1 0 0 0 0 0]
 [0 0 0 0 1 0 0 0]
 [0 1 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 0 0 0 1 0 0]
 [0 0 0 1 0 0 0 0]
 [0 0 0 0 0 0 1 0]
 [1 0 0 0 0 0 0 0]]
```

```
state:
[[0 1 0 0 0 0 0 0]
 [0 0 0 1 0 1 0 0]
 [0 0 0 0 0 0 0 0]
 [1 0 0 0 0 0 0 0]
 [0 0 1 0 0 0 0 0]
 [0 0 0 0 0 0 1 1]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 1 0 0 0]]
heuristic 5
[[0 1 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0]
 [0 0 0 0 0 1 0 0]
 [1 0 0 0 0 0 0 0]
 [0 0 1 0 0 0 0 0]
 [0 0 0 0 0 0 1 1]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 1 0 0 0]]
heuristic 3
[[0 1 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0]
 [0 0 0 0 0 1 1 0]
 [1 0 0 0 0 0 0 0]
 [0 0 1 0 0 0 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 1 0 0 0]]
heuristic 3
state:
[[0 0 0 0 0 0 0 0]
 [0 1 1 0 0 1 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 1 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1]
 [1 0 0 0 1 0 0 0]]
heuristic 5
[[0 0 0 0 0 1 0 0]
 [0 1 1 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 1 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1]
 [1 0 0 0 1 0 0 0]]
heuristic 3
[[0 0 0 0 0 1 0 0]
 [0 1 1 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 1 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0]
 [0 0 0 0 0 0 0 1]
 [1 0 0 0 1 0 0 0]]
heuristic 2
[[0 0 0 0 0 1 0 0]
 [0 1 0 0 0 0 0 0]
 [0 0 1 0 0 0 0 0]
 [0 0 0 0 0 0 1 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0]
```

```
heuristic 2
[[0 0 0 0 0 1 0 0]
 [0 1 0 0 0 0 0 0]
 [0 0 1 0 0 0 0 0]
 [0 0 0 0 0 0 1 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0]
 [0 0 0 0 0 0 0 1]
 [1 0 0 0 1 0 0 0]]
heuristic 2
state:
[[0 0 0 0 0 0 0 0]
 [0 0 0 0 0 1 0 0]
 [0 1 0 0 1 0 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 0 0 0 0 0 0]
 [0 0 1 0 0 0 0 0]
 [1 0 0 1 0 0 0 0]
 [0 0 0 0 0 0 1 0]]
heuristic 5
[[0 0 0 0 0 1 0 0]
 [0 0 0 0 0 0 0 0]
 [0 1 0 0 1 0 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 0 0 0 0 0 0]
 [0 0 1 0 0 0 0 0]
 [1 0 0 1 0 0 0 0]
 [0 0 0 0 0 0 1 0]]
heuristic 3
[[0 0 0 0 0 1 0 0]
 [0 1 0 0 0 0 0 0]
 [0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 0 0 0 0 0 0]
 [0 0 1 0 0 0 0 0]
 [1 0 0 1 0 0 0 0]
 [0 0 0 0 0 0 1 0]]
heuristic 1
[[0 0 0 0 0 1 0 0]
 [0 1 0 0 0 0 0 0]
 [0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 0 1]
 [1 0 0 0 0 0 0 0]
 [0 0 1 0 0 0 0 0]
 [0 0 0 1 0 0 0 0]
 [0 0 0 0 0 0 1 0]]
heuristic 1
1
4
The Success percent is = 25.0
Average success steps required: 5.0
Average failure steps required:2.6666666666666665
rahuls-MacBook-Pro:Hill_climbing3 rahulratra$
```

Steepest Ascent Hill Climbing With Sideways Walk.

```
rahuls-MacBook-Pro:Hill_climbing3 rahulratra$ python3 steepest_ascent_sideways.py
enter the number of samples to find the solution 4
enter the size of board 8
state:
[[0 0 0 1 0 0 0 0]
 [1 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 1 0 0 0 0 1 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 1 0 1]
 [0 0 0 0 0 0 0 0]
 [0 0 1 0 0 0 0 0]]
7
[[0 0 0 1 0 0 0 0]
 [1 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 1 0]
 [0 1 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 1 0 1]
 [0 0 0 0 0 0 0 0]
 [0 0 1 0 0 0 0 0]]
heuristic 3
[[0 0 0 1 0 0 0 0]
 [1 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 1 0]
 [0 1 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 1 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 1 0 0 0 0 0]]
heuristic 2
[[0 0 0 1 0 0 0 0]
 [1 0 0 0 0 0 0 0]
 [0 0 0 0 1 0 1 0]
 [0 1 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 1 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 1 0 0 0 0 0]]
heuristic 1
[[0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 1 0]
 [1 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0]
 [0 1 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 0 0 0 1 0 0]
 [0 0 1 0 0 0 0 0]]
heuristic 0
[[0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 1 0]
 [1 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0]
 [0 1 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 0 0 0 1 0 0]
 [0 0 1 0 0 0 0 0]]
state:
```

```
state:
[[0 0 0 1 0 0 0 0]
 [0 1 0 0 0 0 0 1]
 [0 0 1 0 0 0 0 0]
 [1 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 1 0 1 0]
 [0 0 0 0 0 1 0 0]]
8
[[0 0 0 1 0 0 0 0]
 [0 1 0 0 0 0 0 1]
 [0 0 1 0 0 0 0 0]
 [1 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 1 0]
 [0 0 0 0 1 0 0 0]
 [0 0 0 0 0 1 0 0]]
heuristic 4
[[0 0 0 1 0 0 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 1 0 0 0 0 0]
 [1 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 1 0 0 0 0 1 0]
 [0 0 0 0 1 0 0 0]
 [0 0 0 0 0 1 0 0]]
heuristic 3
[[0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 1 0 0 0 0 0]
 [1 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0]
 [0 1 0 0 0 0 1 0]
 [0 0 0 0 1 0 0 0]
 [0 0 0 0 0 1 0 0]]
heuristic 2
[[0 0 0 0 0 1 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 1 0 0 0 0 0]
 [1 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0]
 [0 1 0 0 0 0 1 0]
 [0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 0 0]]
heuristic 1
[[0 0 1 0 0 0 0 0]
 [0 0 0 0 0 1 0 0]
 [0 0 0 0 0 0 0 1]
 [1 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0]
 [0 0 0 0 0 0 1 0]
 [0 0 0 0 1 0 0 0]
 [0 1 0 0 0 0 0 0]]
```

```
heuristic 0
[[0 0 1 0 0 0 0 0]
 [0 0 0 0 0 1 0 0]
 [0 0 0 0 0 0 0 1]
 [1 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0]
 [0 0 0 0 0 0 1 0]
 [0 0 0 0 1 0 0 0]
 [0 1 0 0 0 0 0 0]]
state:
[[0 1 1 0 0 0 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 0 1 1 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 1 0 0]
 [1 0 0 0 0 0 1 0]]
6
[[0 1 1 0 0 0 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 0 1 1 0 0 0]
 [0 0 0 0 0 0 1 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 1 0 0]
 [1 0 0 0 0 0 0 0]]
heuristic 4
[[0 1 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 0 1 1 0 0 0]
 [0 0 0 0 0 0 1 0]
 [0 0 1 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 1 0 0]
 [1 0 0 0 0 0 0 0]]
heuristic 3
[[0 1 0 0 1 0 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 0 1 0 0 0 0]
 [0 0 0 0 0 0 1 0]
 [0 0 1 0 0 0 0 0]
 [0 0 0 0 0 0 0 0]
 [0 0 0 0 0 1 0 0]
 [1 0 0 0 0 0 0 0]]
heuristic 2
[[0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 0 1 0 0 0 0]
 [0 0 0 0 0 0 1 0]
 [0 0 1 0 0 0 0 0]
 [0 1 0 0 0 0 0 0]
 [0 0 0 0 0 1 0 0]
 [1 0 0 0 0 0 0 0]]
```

```
heuristic 1
[[0 0 0 0 1 0 0 0]
 [0 1 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0]
 [0 0 0 0 0 0 1 0]
 [0 0 1 0 0 0 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 0 0 0 1 0 0]
 [1 0 0 0 0 0 0 0]]
heuristic 0
[[0 0 0 0 1 0 0 0]
 [0 1 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0]
 [0 0 0 0 0 0 1 0]
 [0 0 1 0 0 0 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 0 0 0 1 0 0]
 [1 0 0 0 0 0 0 0]]
state:
[[0 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0]
 [1 0 0 0 0 1 0 0]
 [0 0 0 0 0 0 0 0]
 [0 1 1 0 0 0 0 1]
 [0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 1 0]
 [0 0 0 0 0 0 0 0]]
6
[[0 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0]
 [1 0 0 0 0 1 0 0]
 [0 0 0 0 0 0 0 1]
 [0 1 1 0 0 0 0 0]
 [0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 1 0]
 [0 0 0 0 0 0 0 0]]
heuristic 3
[[1 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0]
 [0 0 0 0 0 1 0 0]
 [0 0 0 0 0 0 0 1]
 [0 1 1 0 0 0 0 0]
 [0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 1 0]
 [0 0 0 0 0 0 0 0]]
heuristic 2
[[1 0 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0]
 [0 0 0 0 0 1 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 1 0 0 0 0 0]
 [0 0 0 0 1 0 0 0]
 [0 0 0 0 0 0 1 0]
 [0 1 0 0 0 0 0 0]]
heuristic 1
```

```
heuristic 1
[[0 1 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0]
 [0 0 0 0 0 1 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 1 0 0 0 0 0]
 [1 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 1 0]
 [0 0 0 0 1 0 0 0]]
heuristic 0
[[0 1 0 0 0 0 0 0]
 [0 0 0 1 0 0 0 0]
 [0 0 0 0 0 1 0 0]
 [0 0 0 0 0 0 0 1]
 [0 0 1 0 0 0 0 0]
 [1 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 1 0]
 [0 0 0 0 1 0 0 0]]
Success_percent is = 100.0
Average_Number_of_success_steps18.5
/Library/Frameworks/Python.framework/Versions/3.7/lib/python3
  avg = a.mean(axis)
/Library/Frameworks/Python.framework/Versions/3.7/lib/python3
  ret = ret.dtype.type(ret / rcount)
Average_number_of_failure_stepsnan
rahuls-MacBook-Pro:Hill_climbing3 rahulratra$
```