29th October, 2024 Tuesday Laboratory-6

For the (N=8) Queens Puzzle, implement using the following algorithms:

a) A*

b) Hill Climbing Algorithm

A* Algorithm to solve the (N=8) queens Problem.

Step 1: Create a 8×8 chessboard in the initial state
Setup an open set to explore configurations
Setup a visited state to display different sets visited

Step 2: Calculate the number of attacking pairs that Queens should not be in the same row or same column or same diagonal

```
if state[j] == state[j] or abs(state[i])
    state[j] == j-1  // diagonal
    attacks += 1
```

Then we increment the variables attacks to determine attacking pairs.

```
open_set : []
```

Step 3: Assign initial state to open state for the first iteration. If the node is not visited, then first put it iteration. If open set and will push the node to heap q (priority q)

```
heap q = heap_push(open set, Node(new state, g, h)
```

Step 4: Total estimated cost f = g + h,
g → cost to reach the current state
h → number of attacks to reach goal state

Step 5: Main loop { remove the node with lower cost }

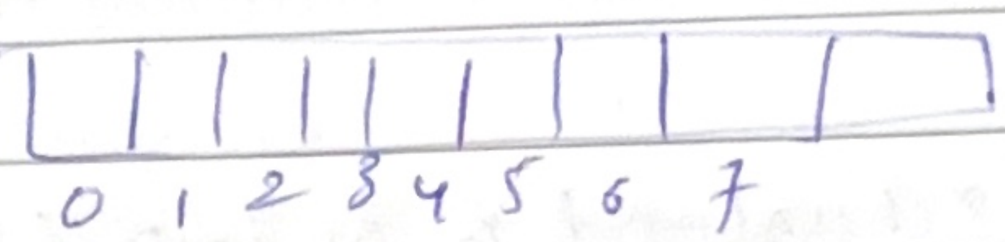
Step 6: Determination of the next row to place the queen

Step 7: Update g and calculate h

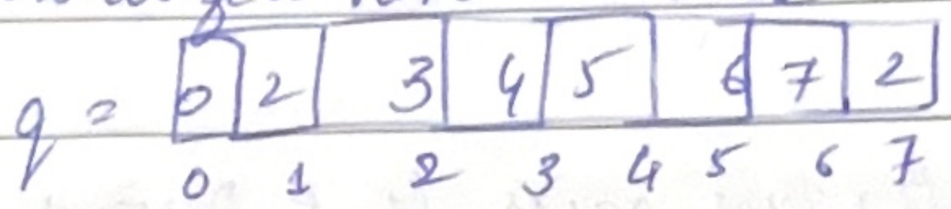
Hill climbing algorithm

Step 1: Create an array where each index represents a column and the value represents the row position of the queen in that row

```
int [] q = new int[8]
```



Step 2: Initialize a random state



Step 3: Store a heuristic value h(n) where 'h' represents the number of conflicting pairs in each state.

Step 4: Check for conflicting pairs:

```
conflicts()
{
    conflicts = 0
    for i ← 0 to 7 do
        for j ← i+1 to 7 do
            if (board[i] == board[j])
                conflicts += 1
    return conflicts
```

Step 5: current = conflicts(board)

Step 6:

```
for i ← 0 to 7 do
    for j ← 0 to 7 do
        board[i] = j
        C = conflicts(board)
```

```
if (c < current)
  { temp = current
    current = c,
  }

if (c == temp)
      return No Solution

if (c == 0)
      return current state
```

Pseudocode

Output:

Best Solution found (with 0 attacking pairs)

```
. . . . Q . . .
. . . . . . . Q .
. Q . . . . . . .
. . . . . Q . . .
. . Q . . . . . .
Q . . . . . . . .
. . . Q . . . . .
. . . . . . . . Q
```

Result

24/10/24