Bafna Gold
Date: 22/10/24 Page:

Lab - 04

8 Queens Using Hill Climbing Search

Algorithm

Step 1: Take input from User
- Initial positions of queens provided by the user as a list (8 elements, each from 0 to 7)
- Maximum number of iterations.

Step 2: Procedure Hill_Climbing_8queens
1. Initialize current_state ← user-provided initial state.
2. Set current_value ← Calculate Conflicts (current)
3. for each iteration from 1 to max_iteration
   → Generate all neighbors of current_state:
   for each queen in each column:
   move the queen to another row in the same column to create a new neighbor state.
   → Evaluate each neighbor:
   Find the neighbor with the minimum number of conflicts (best_neighbor)
   Set next_value ← Calculate Conflicts (best_neighbor)
   → If next_value < current_value:
   move to the best_neighbor (current_state ← best_neighbor)
   Update current_value ← next_value
   Else: Break (local optimum reached)
4. Output the final current_state and current_value (number of conflicts)

5. If current value == 0 :
print "Solution found with no conflicts"
Else:
print "Best solution found with conflicts"

Step 3: Procedure calculate Conflicts (state):

1. Set conflicts ← 0
2. For each pair of queens in state (queen1, queen2):

If they are in the same row or on the same diagonal:
increment conflicts
conflicts ++;
3. Return conflicts.

End procedure.

Output:

Enter the initial row positions for each queen:
Enter row position for queen in column 0 : 3
1 : 1
2 : 2
3 : 0
Enter maximum iterations : 10

Initial Board:

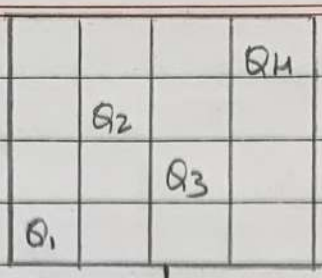
. . . Q
. Q . .
. . Q .
Q . . .

Final Board

. . . Q
. Q . .
. . Q .
Q . . .

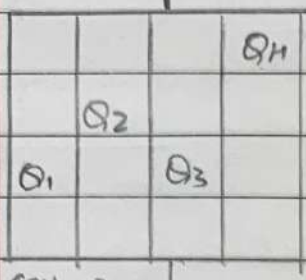
Best solution : [3, 1, 2, 0], Conflicts : 2
Solution found in 1 iterations
No perfect solution found.

Initial state

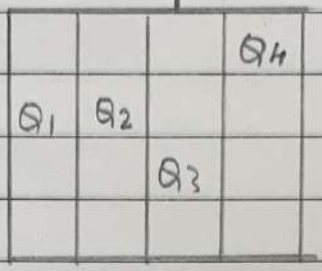


Cost = 1

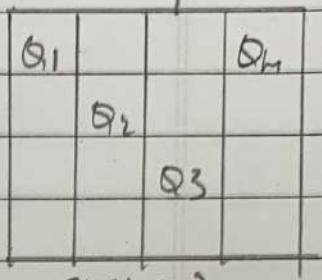
movement of 1st queen



Cost = 2

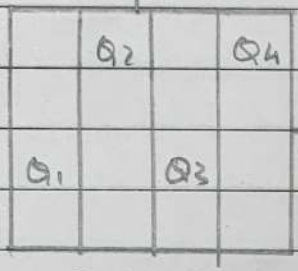


Cost = 2



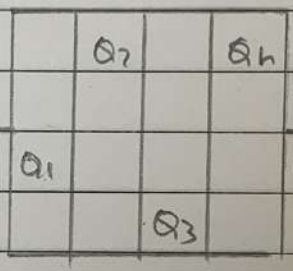
Cost = 2

movement of 2nd queen



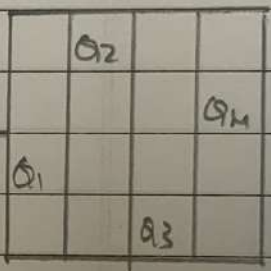
Cost = 2

movement of 3rd queen



Cost = 1

movement of 4th queen



Cost = 0

Goal state