# -\*- coding: utf-8 -\*-

"""7\_Heuristic\_Search(Hill\_climbing\_Alogorithm).ipynb

Automatically generated by Colaboratory.

Original file is located at

https://colab.research.google.com/drive/1PYokr8JmteuCBlDe7zSTOHeKGGe6HROo

"""

import random

N = int(input("Enter value of N in N-Queens Problem "))

def checkPossibility(chessBoard, i, j):

for row in range(len(chessBoard)):

for element in range(len(chessBoard[row])):

if chessBoard[row][element] == 'Q':

jDist = abs(j - element)

iDist = abs(i - row)

if i == row or j == element or jDist == iDist:

return False;

return True;

def hueristicFunction(chessBoard, goal, i, j):

count = 0

for row in chessBoard:

for element in row:

if element == 'Q':

count += 1

if i != -1 and j != -1:

count += 1

return (goal - count);

chessBoard = [['-'] \* N for i in range(N)]

while hueristicFunction(chessBoard, N, -1, -1) != 0: # loop to perform random placing

chessBoard = [['-'] \* N for i in range(N)]

for row in range(len(chessBoard)):

dist = -1

i = -1

j = -1

resultList = []

for element in range(len(chessBoard[row])):

if checkPossibility(chessBoard, row, element):

hDist = hueristicFunction(chessBoard, N, row, element)

if dist <= hDist:

dist = hDist

i = row

j = element

resultList.append([row, element])

if len(resultList) > 0:

idex, jdex = resultList[random.randint(0, len(resultList) - 1)]

print("\n i index=", idex, "\n j index=", jdex)

chessBoard[idex][jdex] = 'Q'

for row1 in chessBoard:

print("\t", row1)

print("Heuristic value is: ", hueristicFunction(chessBoard, N, -1, -1))

print("Heuristic value is: ", hueristicFunction(chessBoard, N, -1, -1), "\n" )

print("\nFollowing is the Global maxima solution for", N, "Queen Problem with heuristic distance from goal state =",

hueristicFunction(chessBoard, N, -1, -1), "\n\n")

for row1 in chessBoard:

print("\t", row1)

print("\n")

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Output:

Enter value of N in N-Queens Problem 4

i index= 0

j index= 3

['-', '-', '-', 'Q']

['-', '-', '-', '-']

['-', '-', '-', '-']

['-', '-', '-', '-']

Heuristic value is: 3

i index= 1

j index= 1

['-', '-', '-', 'Q']

['-', 'Q', '-', '-']

['-', '-', '-', '-']

['-', '-', '-', '-']

Heuristic value is: 2

['-', '-', '-', 'Q']

['-', 'Q', '-', '-']

['-', '-', '-', '-']

['-', '-', '-', '-']

Heuristic value is: 2

i index= 3

j index= 2

['-', '-', '-', 'Q']

['-', 'Q', '-', '-']

['-', '-', '-', '-']

['-', '-', 'Q', '-']

Heuristic value is: 1

i index= 0

j index= 2

['-', '-', 'Q', '-']

['-', '-', '-', '-']

['-', '-', '-', '-']

['-', '-', '-', '-']

Heuristic value is: 3

i index= 1

j index= 0

['-', '-', 'Q', '-']

['Q', '-', '-', '-']

['-', '-', '-', '-']

['-', '-', '-', '-']

Heuristic value is: 2

i index= 2

j index= 3

['-', '-', 'Q', '-']

['Q', '-', '-', '-']

['-', '-', '-', 'Q']

['-', '-', '-', '-']

Heuristic value is: 1

i index= 3

j index= 1

['-', '-', 'Q', '-']

['Q', '-', '-', '-']

['-', '-', '-', 'Q']

['-', 'Q', '-', '-']

Heuristic value is: 0

Heuristic value is: 0

Following is the Global maxima solution for 4 Queen Problem with heuristic distance from goal state = 0