29/10/24

Lab - 05

Implement Simulated Annealing to solve N-queens problem

ALGORITHM:

function EVALUATE(state):
    n ← length of state
    attacks ← 0
    for i from 0 to n-1:
        for j from i+1 to n-1:
            if state[i] == state[j] or abs(state[i] - state[j]) == abs(i-j):
                attacks ← attacks + 1
    return attacks

function RANDOM_NEIGHBOR(state):
    n ← length of state
    new_state ← copy of state
    col ← random integer between 0 & n-1
    row_conflicts ← []

    for each row from 0 to n-1:
        if row ≠ state[col]:
            temp_state ← copy of new_state with new_state[col] = row
            row_conflicts.append((row, EVALUATE(temp_state)))

    if row_conflicts is not empty:
        best_row ← row with lowest evaluation in row_conflicts
        new_state[col] ← best_row
    return new_state

```
function SCHEDULE(t):
    return: MAX(0.01, min(1, 1-0.05*t))

function SIMULATED_ANNEALING(State, max_iteration):
    for t from 1 to max_iteration:
        T ← SCHEDULE(t)
        if T == 0:
            return state, t
        candidate ← RANDOM_NEIGHBOR(state)
        E ← EVALUATE(candidate) - EVALUATE(state)
        if E > 0:
            state ← candidate
        else:
            prob ← exp/(T*1.5)
            if random() < prob:
                state ← candidate
        if EVALUATE(state) == 0:
            print("Global max found at", t)
            return state, t
    print("Reached local maximum, found at iteration", max_iteration)
    return state, max_iteration

function SOLVE-N-QUEENS(n):
    initial_state ← array of n random integers b/w 0 & n-1
    result, iteration ← SIMULATED_ANNEALING(initial_state, max_iteration=5000)
    return result, iteration
```

function PRINT_BOARD(state):

for row from 0 to n-1:
line = " "
for col from 0 to n-1
if state[col] == row:
line ← line + "Q "
else:
line ← line + ". "
print(line)

// main code
N ← user input "Enter no of queens"
solution, iteration ← SOLVE_N_QUEENS(N)
print("sol for" N, "queen find at", iteration)
PRINT_BOARD(solution)
print("Final evaluation", EVALUATE(solution))

Output:
Enter no of queens (N) : 8
Global maximum (no attack) found at iteration 902
Solution for 8 queens in 902 iteration

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

Final evaluation (objective function value)