2.13 EXHAUSTIVE SEARCH TO SOLVE THE ASSIGNMENT PROBLEM

Question:

You are given a cost matrix where each element cost[i][j] represents the cost of assigning worker i to task j. Develop a program that utilizes exhaustive search to solve the assignment problem. The program should Define a function total_cost(assignment, cost_matrix) that takes an assignment (list representing worker-task pairings) and the cost matrix as input. It iterates through the assignment and calculates the total cost by summing the corresponding costs from the cost matrix Implement a function assignment_problem(cost_matrix) that takes the cost matrix as input and performs the following Generate all possible permutations of worker indices (excluding repetitions).

AIM

To determine the optimal assignment of workers to tasks that results in the minimum total cost, using the exhaustive search method.

ALGORITHM

1. Start
2. Define total_cost(assignment, cost_matrix) to Iterate over each worker index and its assigned task.
3. Sum up the cost from cost_matrix[worker][task].
4. Define assignment_problem(cost_matrix) to Generate all permutations of task indices using itertools.permutations.
5. For each permutation: Calculate total cost using total_cost.
6. Keep track of the lowest cost and corresponding assignment.
7. Return the optimal assignment and minimum total cost.
8. End

PROGRAM

```
import itertools

def total_cost(assignment, cost_matrix):
    return sum(cost_matrix[i][assignment[i]] for i in range(len(assignment)))

def assignment_problem(cost_matrix):
    n = len(cost_matrix)
    min_cost = float('inf')
    best_assignment = []
    for perm in itertools.permutations(range(n)):
        cost = total_cost(perm, cost_matrix)
        if cost < min_cost:
            min_cost = cost
            best_assignment = perm
    return min_cost, best_assignment

def run_assignment_problem():
    print("Enter cost matrix row by row (space-separated):")
    matrix = []
    for _ in range(int(input("Enter number of workers/tasks: "))):
        row = list(map(int, input().split()))
        matrix.append(row)
    cost, assignment = assignment_problem(matrix)
    print("Optimal Assignment:", [(f"worker {i+1}", f"task {assignment[i]+1}") for i in range(len(assignment))])
    print("Total Cost:", cost)
run_assignment_problem()
```

Input:

[[3, 10, 7],

[8, 5, 12],

[4, 6, 9]]

Output:

```
Enter cost matrix row by row (space-separated):
Enter number of workers/tasks: 3
3 10 7
8 5 12
4 6 9
Optimal Assignment: [('worker 1', 'task 3'), ('worker 2', 'task 2'), ('worker 3', 'task 1')]
Total Cost: 16
>>>
```

RESULT:

Thus, Solving the Assignment Problem using Exhaustive Search is successfully executed and the output is verified.

PERFORMANCE ANALYSIS:

· Time Complexity: O(n!)

· Space Complexity: O(n)