CAPSTONE PROJECT

APPROXIMATION ALGITHM FOR NEAREST NEIGHBOR ALGORITHM

CSA0695- DESIGN ANALYSIS AND ALGORITHMS FOR AMORTIZED ANALYSIS

SAVEETHA SCHOOL OF ENGINEERING

SIMATS ENGINEERING

Supervisor

Dr. R. Dhanalakshmi

Done by

D. Sravika Reddy (192225071)

APPROXIMATION ALGORITHM FOR NEAREST NEIGHBOR ALGOROTHM

PROBLEM STATEMENT: Optimizing Service Technician Routes for an Appliance Repair Company

CONTEXT: An appliance repair company, FixItAll, employs several service technicians who visit customer locations to perform repairs. Each day, a technician starts from the company headquarters and needs to visit multiple customer locations before returning to the headquarters. The goal is to minimize the total travel distance for each technician, ensuring all customers are served efficiently.

PROBLEM: FixItAll needs to solve the Traveling Salesman Problem (TSP) to determine the optimal route for each technician. Given the large number of customer locations, finding an exact solution is impractical, so an approximation algorithm is required.

INPUT: 1. Customer Locations: A set of locations L={l1,l2,…,ln}L = \{l_1, l_2, \ldots, l_n\}L={l1,l2,…,ln} with known coordinates.

2. Headquarters: A single headquarters location HHH.

3. Distance Matrix: A matrix MMM where MijM_{ij}Mij represents the distance between location lil_ili and location ljl_jlj.

OBJECTIVE: Design an approximation algorithm using the Nearest Neighbor Heuristic to determine the route for each technician, such that: 1. Each customer location is visited exactly once. 2. The total travel distance is minimized. 3. The route starts and ends at the headquarters

ABSTRACT:

To optimize service technician routes using the Nearest Neighbor Heuristic, start by reading the customer locations, headquarters, and the distance matrix. Initialize an array to track visited locations and create a route that begins at the headquarters. Employ the Nearest Neighbor Heuristic to iteratively select the closest unvisited location, move to it, and mark it as visited. Continue this process until all customer locations have been visited, then return to the headquarters. Calculate the total travel distance of the completed route and print both the route and the total distance. If there are multiple technicians, repeat these steps to determine optimal routes for each.

INTRODUCTION:

In the dynamic field of appliance repair, optimizing the routes for service technicians is crucial for enhancing efficiency and reducing operational costs. Each technician is tasked with visiting multiple customer locations each day, starting and ending their journey at the company's headquarters. Given the large number of customer locations, solving this problem requires an effective strategy to minimize travel distance and ensure timely service delivery.

The Traveling Salesman Problem (TSP) provides a theoretical framework for this challenge, where the objective is to find the shortest possible route that visits each location exactly once and returns to the starting point. However, finding an exact solution to the TSP for a large number of locations is computationally intensive and impractical. Therefore, approximation algorithms are often employed to provide near-optimal solutions in a reasonable timeframe.

One such approximation technique is the Nearest Neighbor Heuristic. This heuristic approach starts by selecting the nearest unvisited location from the current position, progressively constructing a route by making local optimal choices. Though this method does not guarantee the absolute shortest route, it provides a practical and efficient way to tackle the problem.

CODING:

The coding solution for optimizing technician routes involves implementing the Nearest Neighbor Heuristic in C. Begin by reading input data, including customer locations and the distance matrix. Use an array to track visited locations and construct the route by iteratively selecting the nearest unvisited location. After all locations are visited, return to the headquarters. Calculate and display the total

travel distance of the route. This approach provides a practical method for approximating efficient service routes within the constraints of computational feasibility.

C-programming

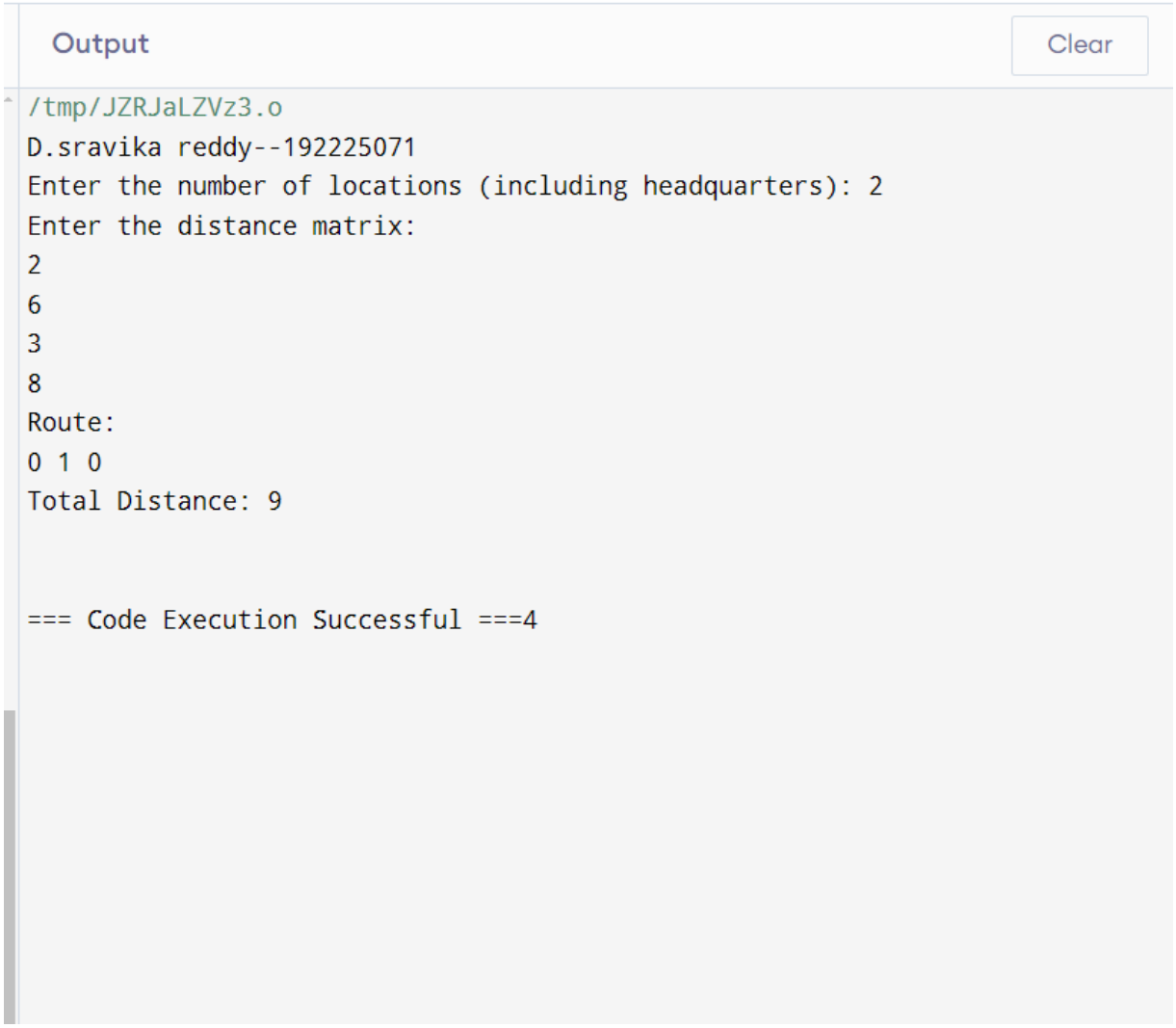
```
#include <stdio.h>

#include <limits.h>



int findNearestNeighbor(int current, int n, int distanceMatrix[n][n], int
visited[n]) {
    int minDistance = INT_MAX;
    int nearest = -1;
    for (int i = 0; i < n; i++) {
        if (!visited[i] && distanceMatrix[current][i] < minDistance) {
            minDistance = distanceMatrix[current][i];
            nearest = i;
        }
    }
    return nearest;
}



void nearestNeighborHeuristic(int n, int distanceMatrix[n][n], int route[]) {
    int visited[n];
```

```
    for (int i = 0; i < n; i++) {
        visited[i] = 0;
    }

    int current = 0;
    visited[current] = 1;
    route[0] = current;

    for (int i = 1; i < n; i++) {
        int nearest = findNearestNeighbor(current, n, distanceMatrix, visited);
        route[i] = nearest;
        visited[nearest] = 1;
        current = nearest;
    }

    route[n] = 0;
}

int calculateTotalDistance(int n, int distanceMatrix[n][n], int route[]) {
    int totalDistance = 0;
    for (int i = 0; i < n; i++) {
        totalDistance += distanceMatrix[route[i]][route[i+1]];
```

```
    }
    return totalDistance;
}

int main() {
    int n;



    printf("Enter the number of locations (including headquarters): ");
    scanf("%d", &n);

    int distanceMatrix[n][n];



    printf("Enter the distance matrix:\n");
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++) {
            scanf("%d", &distanceMatrix[i][j]);
        }
    }

    int route[n+1];
```

```
    nearestNeighborHeuristic(n, distanceMatrix, route);



    printf("Route:\n");
    for (int i = 0; i < n+1; i++) {
        printf("%d ", route[i]);
    }
    printf("\n");



    int totalDistance = calculateTotalDistance(n, distanceMatrix, route);
    printf("Total Distance: %d\n", totalDistance);

    return 0;
}
```

OUTPUT:



COMPLEXITY ANALYSIS:

Time Complexity: The time complexity of the Nearest Neighbor Heuristic for the Traveling Salesman Problem is O(n2)$O(n2)$, where nn is the number of locations. This complexity arises from the need to find the nearest unvisited location for each of the nn locations, which involves checking all remaining locations. Specifically, each step involves scanning a list of unvisited locations, leading to O(n2)$O(n2)$ operations in total. While efficient compared to exact algorithms, it still scales quadratically with the number of locations.

Space Complexity: The space complexity of the Nearest Neighbor Heuristic for the Traveling Salesman Problem is O(n)$O(n)$, where nn is the number of locations. This complexity is primarily due to the storage required for the route array and the visited status array, each of which requires space proportional to nn. The distance matrix itself also requires O(n2)$O(n2)$ space, but this is typically considered separate from the heuristic's space complexity. Therefore, the heuristic's space complexity is linear in relation to the number of locations.

BEST CASE:

In the best-case scenario for the Nearest Neighbor Heuristic, the heuristic might achieve optimal or near-optimal routing if the locations are arranged in a way that the nearest neighbor always leads to the shortest possible path. For instance, if all customer locations are in a line or circular arrangement, the heuristic could closely approximate the optimal route. However, despite this favorable arrangement, the Nearest Neighbor Heuristic does not guarantee optimality and may still produce suboptimal routes due to its greedy approach. Thus, even in the best case, the heuristic's performance can vary.

WORST CASE:

In the worst-case scenario for the Nearest Neighbor Heuristic, the algorithm may produce a route significantly longer than the optimal one. This occurs when the heuristic consistently makes locally optimal choices that lead to a suboptimal overall path. For example, if locations are arranged in a way that traps the heuristic into a lengthy, inefficient route, the discrepancy between the heuristic's solution and the true optimal route can be substantial. The heuristic's greedy nature means it might miss shorter paths by focusing on immediate proximity rather than global optimality.

AVERAGE CASE:

In the average case, the Nearest Neighbor Heuristic often provides a reasonably good approximation to the optimal route but is generally not guaranteed to be close to the best possible solution. On average, it may produce routes with a length that is a fraction of the optimal route, influenced by the distribution and arrangement of locations. The heuristic's performance varies depending on the problem instance, with its efficiency better for some configurations and worse for others. While it offers a quick and practical solution, the average-case solution is typically less optimal compared to more sophisticated algorithms..

FUTURE SCOPE:

Future advancements in route optimization for service technicians could involve integrating more sophisticated algorithms, such as Genetic Algorithms or Ant Colony Optimization, to improve accuracy and efficiency beyond the Nearest Neighbor Heuristic. Machine learning techniques could be employed to predict traffic patterns and dynamically adjust routes. Additionally, incorporating real-time data and constraints, such as time windows and service priorities, could further enhance route planning. Exploring hybrid approaches that combine multiple heuristic methods might also offer better performance. Finally, expanding the solution to multi-vehicle scenarios could address the needs of larger fleets.

CONCLUSION:

In conclusion, the Nearest Neighbor Heuristic offers a practical and efficient approach for optimizing service technician routes, balancing computational feasibility with route quality. While it may not always yield the optimal solution, it provides a useful approximation that reduces travel distance and operational costs. Future developments, including advanced algorithms and real-time data integration, hold promise for further enhancing route optimization. Embracing these innovations will enable more precise and adaptable solutions, improving overall service efficiency and customer satisfaction.