26 Travelling salesman

#include <stdio.h>

#include <limits.h>

#define MAX\_N 100

int min(int a, int b) {

return a < b ? a : b;

}

// Function to construct the cost matrix and parent table for the optimal TSP solution

void constructOptimalTSP(int graph[][MAX\_N], int n, int cost[][MAX\_N], int parent[][MAX\_N]) {

// Initialize base case (single node)

for (int i = 0; i < n; i++) {

cost[i][1 << i] = 0;

}

// Fill the cost and parent tables using dynamic programming

for (int subset = 2; subset <= n; subset++) {

for (int i = 0; i < n; i++) {

int min\_cost = INT\_MAX;

int best\_parent = -1;

// Consider all possible subsets of size (subset - 1) containing 'i'

for (int j = 0; j < n; j++) {

if ((subset & (1 << j)) > 0 && i != j) {

int remaining = subset ^ (1 << i); // Remaining nodes in the subset

int potential\_cost = cost[j][remaining] + graph[i][j];

if (potential\_cost < min\_cost) {

min\_cost = potential\_cost;

best\_parent = j;

}

}

}

cost[i][subset] = min\_cost;

parent[i][subset] = best\_parent;

}

}

}

// Function to reconstruct the optimal TSP tour from the parent table

void printOptimalTour(int parent[][MAX\_N], int n, int start) {

int path[n];

int current = start;

printf("Optimal Tour: ");

// Traverse the parent table to reconstruct the path

for (int i = 0; i < n; i++) {

path[i] = current;

current = parent[current][(1 << n) - 1];

}

// Print the tour starting from the source node

for (int i = 0; i < n; i++) {

printf("%d -> ", path[i]);

}

printf("%d\n", path[0]);

}

// Function to calculate the cost of the optimal tour

int calculateTourCost(int cost[][MAX\_N], int n, int start) {

return cost[start][(1 << n) - 1];

}

int main() {

int graph[MAX\_N][MAX\_N] = {

// Example adjacency matrix (modify as needed)

{0, 10, 15, 20},

{10, 0, 35, 25},

{15, 35, 0, 30},

{20, 25, 30, 0}

};

int n = sizeof(graph[0]) / sizeof(graph[0][0]); // Number of vertices

int cost[MAX\_N][MAX\_N], parent[MAX\_N][MAX\_N];

constructOptimalTSP(graph, n, cost, parent);

int start = 0; // Starting node (can be modified)

printf("Minimum Cost: %d\n", calculateTourCost(cost, n, start));

printOptimalTour(parent, n, start);

return 0;

}

