**37. Job assignment problem**

**Code:**

#include <stdio.h>

#include <limits.h>

#define N 4 // Number of workers and tasks, assuming square matrix for simplicity

// Function to find the minimum cost assignment using Branch and Bound

void branchAndBound(int costMatrix[N][N], int assigned[N], int worker, int cost, int \*minCost) {

// Base case: if all workers are assigned tasks, update minimum cost

if (worker == N) {

if (cost < \*minCost)

\*minCost = cost;

return;

}

// Prune the search if the current cost exceeds the minimum cost

if (cost >= \*minCost)

return;

// Try assigning the current worker to each task and recur for remaining workers

for (int task = 0; task < N; task++) {

if (!assigned[task]) {

assigned[task] = 1;

branchAndBound(costMatrix, assigned, worker + 1, cost + costMatrix[worker][task], minCost);

assigned[task] = 0; // Backtrack

}

}

}

int main() {

int costMatrix[N][N];

int assigned[N] = {0}; // Array to keep track of task assignments

int minCost = INT\_MAX;

// Taking input for the cost matrix

printf("Enter the cost matrix (%d x %d):\n", N, N);

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

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

scanf("%d", &costMatrix[i][j]);

}

}

// Start from the first worker and compute the minimum cost using branch and bound

branchAndBound(costMatrix, assigned, 0, 0, &minCost);

printf("Minimum cost of assignment: %d\n", minCost);

return 0;

}

**Output:**

Enter the cost matrix (4 x 4):

0 1 2 3

1 2 3 4

2 3 4 5

3 4 5 6

Minimum cost of assignment: 12

--------------------------------

Process exited after 16.56 seconds with return value 0

Press any key to continue . . .

