78. Exhaustive Search

Code:

```
import itertools

def tsp(cities, distances):
    n = len(cities)
    min_distance = float('inf')
    best_path = None

    for path in itertools.permutations(cities):
        total_distance = 0
        for i in range(n - 1):
            total_distance += distances[path[i]][path[i + 1]]
        total_distance += distances[path[-1]][path[0]]

        if total_distance < min_distance:
            min_distance = total_distance
            best_path = path

    return best_path, min_distance
cities = [0, 1, 2, 3]
distances = [
    [0, 10, 15, 20],
    [10, 0, 35, 25],
    [15, 35, 0, 30],
    [20, 25, 30, 0]
]
path, min_distance = tsp(cities, distances)
print("Shortest TSP path:", path, "with distance:", min_distance)
```

Output:

```
Shortest TSP path: (0, 1, 3, 2) with distance: 80
```

Time Complexity:

- T(n)= O(2^n)