Homework3

Zhesheng Xu
Student ID: 42353012

May 20th 2025

Problem Description

The Orienteering Problem (OP) is a variant of the Traveling Salesman Problem (TSP). The objective is to determine a subset of nodes to visit and the visiting sequence such that:

- Each node i has a score s_i.
- Traveling from node i to node j incurs a travel time c_{ij}.
- The total travel time must not exceed a given time budget T.
- The route starts and ends at node 0 (the depot).

The goal is to **maximize the total collected score** while staying within the travel time budget.
The input data are randomly generated in the Python script `op_random_instance.py`.

Mathematical Model

Decision Variables:

$$x_{ij} = \begin{cases} 1 & \text{if the route goes from node } i \text{ to } j \\ 0 & \text{otherwise} \end{cases}$$

$$y_i = \begin{cases} 1 & \text{if node } i \text{ is visited} \\ 0 & \text{otherwise} \end{cases}$$

$$u_i \in \{1, \dots, n-1\}, \text{position of node } i \text{ in the tour (MTZ constraints)}$$

Objective: Maximize the total collected score:

$$\max \sum_{i=0}^{n-1} s_i y_i$$

Constraints:

- Start and return to the depot: $y_0 = 1$
- Flow conservation:

$$\sum_{i \neq j} x_{ij} = y_j, \quad \sum_{i \neq j} x_{ji} = y_j \quad \forall j$$

- Time budget:

$$\sum_{i \neq j} c_{ij} x_{ij} \leq T$$

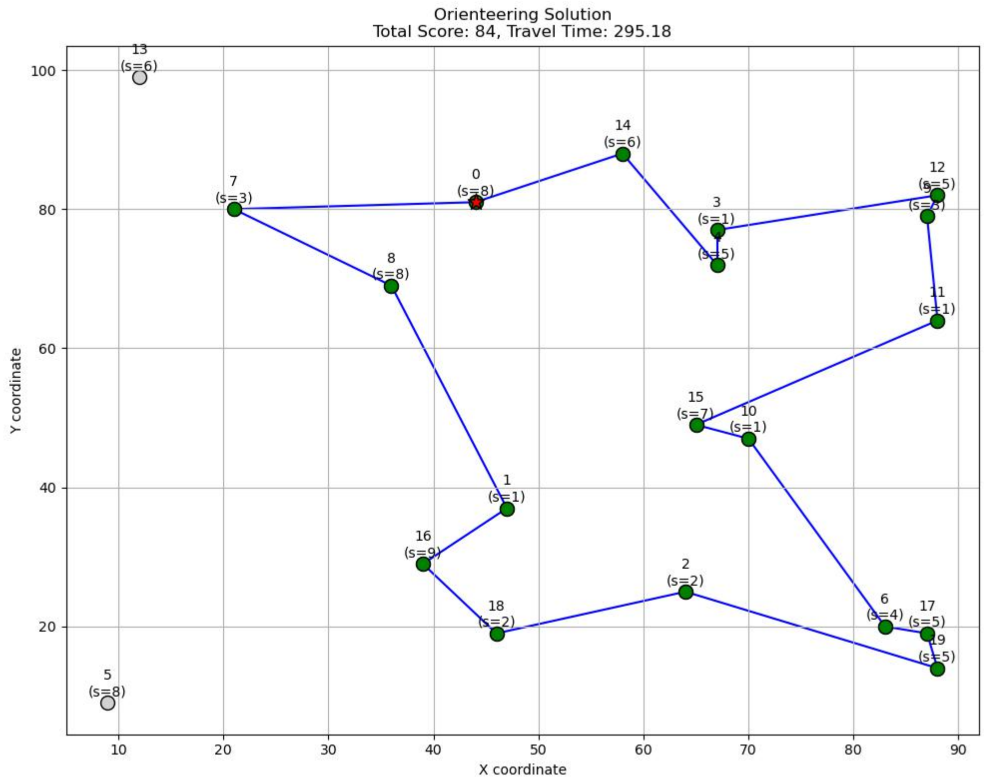
- Subtour elimination (Miller–Tucker–Zemlin constraints):

$$u_i - u_j + n x_{ij} \leq n - 1 \quad \forall i \neq j,\ i, j \in \{1, \dots, n-1\}$$

Solution Output

- **Optimal Tour:** e.g., [0, 14, 4, 3, 12, 9, 11, 15, 10, 6, 17, 19, 2, 18, 16, 1, 8, 7, 0]
- **Total Collected Score:** e.g., 84
- **Total Travel Time:** e.g., 295.18

Optimal Tour Visualization



Source Code

The Gurobi Python code used to solve this problem is:

```
import numpy as np
import matplotlib.pyplot as plt
import gurobipy as gp
from gurobipy import GRB
import networkx as nx

# ----- data -----
# number of customers
n = 20

# generate locations
np.random.seed(0)
loc_x = np.random.randint(0, 100, n)
loc_y = np.random.randint(0, 100, n)

# generate scores
s = np.random.randint(1, 10, n)

```

```python
# calculate travel time
c = {
    (i, j): ((loc_x[i] - loc_x[j]) ** 2 + (loc_y[i] - loc_y[j]) ** 2) ** 0.5
    for i in range(n)
    for j in range(n)
}

# time budget
T = 300

# Create model
m = gp.Model("Orienteering")

# Decision variables
# x[i,j] = 1 if we travel from node i to node j, 0 otherwise
x = m.addVars(c.keys(), vtype=GRB.BINARY, name="x")

# u[i] represents the position of node i in the tour (for subtour elimination)
u = m.addVars(range(1, n), lb=1, ub=n - 1, vtype=GRB.INTEGER, name="u")

# y[i] = 1 if node i is visited, 0 otherwise
y = m.addVars(range(n), vtype=GRB.BINARY, name="y")

# Objective: maximize the total collected score
m.setObjective(gp.quicksum(s[i] * y[i] for i in range(n)), GRB.MAXIMIZE)

# Constraints
# Node 0 must be the starting and ending point
m.addConstr(y[0] == 1, "visit_depot")

# Flow balance constraints
for j in range(n):
    m.addConstr(
        gp.quicksum(x[i, j] for i in range(n) if i != j) == y[j], f"in_flow_{j}"
    )
    m.addConstr(
        gp.quicksum(x[j, i] for i in range(n) if i != j) == y[j], f"out_flow_{j}"
    )

# Time budget constraint
m.addConstr(
    gp.quicksum(c[i, j] * x[i, j] for i, j in c.keys() if i != j) <= T, "time_budget"
)

# Miller-Tucker-Zemlin subtour elimination constraints
for i in range(1, n):
    for j in range(1, n):
        if i != j:
            m.addConstr(u[i] - u[j] + n * x[i, j] <= n - 1, f"mtz_{i}_{j}")

# Solve the model
m.optimize()

# Extract solution
if m.status == GRB.OPTIMAL:
    tour = []
    current = 0  # Start from depot
    tour.append(current)

    while True:
        for j in range(n):
            if j != current and x[current, j].x > 0.5:
                tour.append(j)
                current = j
                break
        if current == 0 and len(tour) > 1:
            break

```

```
    # Calculate total collected score and total travel time
    total_score = sum(s[i] for i in tour)
    total_time = sum(c[tour[i], tour[i + 1]] for i in range(len(tour) - 1))

    print(f"Optimal tour: {tour}")
    print(f"Total collected score: {total_score}")
    print(f"Total travel time: {total_time:.2f}")

    # Plot the solution
    plt.figure(figsize=(10, 8))

    # Plot all nodes
    plt.scatter(loc_x, loc_y, s=100, c="lightgray", edgecolors="black", zorder=1)

    # Plot visited nodes
    visited_nodes = [i for i in range(n) if y[i].x > 0.5]
    plt.scatter(
        loc_x[visited_nodes],
        loc_y[visited_nodes],
        s=100,
        c="green",
        edgecolors="black",
        zorder=2,
    )

    # Highlight depot
    plt.scatter(
        loc_x[0], loc_y[0], s=150, c="red", edgecolors="black", marker="*", zorder=3
    )

    # Plot edges
    for i in range(len(tour) - 1):
        plt.plot(
            [loc_x[tour[i]], loc_x[tour[i + 1]]],
            [loc_y[tour[i]], loc_y[tour[i + 1]]],
            "b-",
            zorder=1,
        )

    # Add node labels
    for i in range(n):
        plt.annotate(
            f"{i}\n(s={s[i]})",
            (loc_x[i], loc_y[i]),
            textcoords="offset points",
            xytext=(0, 5),
            ha="center",
        )

    plt.title(
        f"Orienteering Solution\nTotal Score: {total_score}, Travel Time: {total_time:.2f}"
    )
    plt.xlabel("X coordinate")
    plt.ylabel("Y coordinate")
    plt.grid(True)
    plt.tight_layout()
    plt.savefig(r"d:\Visual Studio Code\Operation Research\orienteering_solution.jpg")
    plt.show()
else:
    print("No solution found.")
```

Listing 1: Gurobi Python Code

Optimization Output

```
Optimize a model with 384 rows, 439 columns and 2207 nonzeros
```

```
Model fingerprint: 0xe2154d74
Variable types: 0 continuous, 439 integer (420 binary)
Coefficient statistics:
  Matrix range     [1e+00, 1e+02]
  Objective range  [1e+00, 9e+00]
  Bounds range     [1e+00, 2e+01]
  RHS range        [1e+00, 3e+02]
Found heuristic solution: objective 14.0000000
Presolve removed 1 rows and 21 columns
Presolve time: 0.01s
Presolved: 383 rows, 418 columns, 2202 nonzeros
Variable types: 0 continuous, 418 integer (399 binary)

Root relaxation: objective 8.997690e+01, 206 iterations, 0.01 seconds (0.00 work units)

    Nodes    |    Current Node    |     Objective Bounds      |     Work
 Expl Unexpl |  Obj  Depth IntInf | Incumbent    BestBd   Gap | It/Node Time

     0     0   89.97690    0   31   14.00000   89.97690   543%     -    0s
H    0     0                      39.0000000   89.97690   131%     -    0s
H    0     0                      44.0000000   89.97690   104%     -    0s
     0     0   87.70828    0   33   44.00000   87.70828  99.3%     -    0s
H    0     0                      46.0000000   87.58043  90.4%     -    0s
     0     0   87.58043    0   39   46.00000   87.58043  90.4%     -    0s
     0     0   87.58043    0   37   46.00000   87.58043  90.4%     -    0s
     0     0   86.67193    0   48   46.00000   86.67193  88.4%     -    0s
H    0     0                      50.0000000   86.58144  73.2%     -    0s
     0     0   86.58144    0   51   50.00000   86.58144  73.2%     -    0s
     0     0   86.52193    0   62   50.00000   86.52193  73.0%     -    0s
     0     0   86.51788    0   58   50.00000   86.51788  73.0%     -    0s
     0     0   86.51788    0   58   50.00000   86.51788  73.0%     -    0s
H    0     0                      56.0000000   86.51788  54.5%     -    0s
     0     0   84.13848    0   67   56.00000   84.13848  50.2%     -    0s
     0     0   84.13379    0   65   56.00000   84.13379  50.2%     -    0s
H    0     0                      60.0000000   84.13379  40.2%     -    0s
     0     0   83.77327    0   68   60.00000   83.77327  39.6%     -    0s
     0     0   83.15281    0   65   60.00000   83.15281  38.6%     -    0s
     0     0   83.15281    0   72   60.00000   83.15281  38.6%     -    0s
     0     0   83.15281    0   74   60.00000   83.15281  38.6%     -    0s
H    0     0                      65.0000000   83.15281  27.9%     -    0s
     0     0   83.15281    0   74   65.00000   83.15281  27.9%     -    0s
H    0     0                      70.0000000   83.15281  18.8%     -    0s
     0     0   83.15281    0   74   70.00000   83.15281  18.8%     -    0s
     0     0   83.15281    0   60   70.00000   83.15281  18.8%     -    0s
     0     0   83.15281    0   72   70.00000   83.15281  18.8%     -    0s
     0     0   83.15281    0   67   70.00000   83.15281  18.8%     -    0s
     0     0   83.15281    0   70   70.00000   83.15281  18.8%     -    0s
     0     0   83.15281    0   66   70.00000   83.15281  18.8%     -    0s
     0     0   83.15281    0   74   70.00000   83.15281  18.8%     -    0s
     0     0   83.15281    0   74   70.00000   83.15281  18.8%     -    0s
     0     0   83.15281    0   75   70.00000   83.15281  18.8%     -    0s
     0     0   83.15281    0   75   70.00000   83.15281  18.8%     -    0s
     0     0   83.15281    0   74   70.00000   83.15281  18.8%     -    0s
     0     0   83.15281    0   74   70.00000   83.15281  18.8%     -    0s
```

```
     0     0   83.15281    0   66   70.00000   83.15281  18.8%     -    0s
     0     0   83.15281    0   72   70.00000   83.15281  18.8%     -    0s
     0     0   83.15281    0   75   70.00000   83.15281  18.8%     -    0s
     0     0   83.15281    0   75   70.00000   83.15281  18.8%     -    0s
     0     0   83.15281    0   75   70.00000   83.15281  18.8%     -    0s
H    0     0                      72.0000000   83.15281  15.5%     -    0s
     0     0   83.15281    0   75   72.00000   83.15281  15.5%     -    0s
H    0     0                      73.0000000   83.11988  13.9%     -    0s
H    0     0                      75.0000000   83.11988  10.8%     -    0s
     0     2   83.11988    0   74   75.00000   83.11988  10.8%     -    0s
* 2906   653              19      76.0000000   80.58144  6.03%  14.5    1s

Cutting planes:
  Learned: 8
  Gomory: 4
  Cover: 12
  Implied bound: 32
  Clique: 6
  MIR: 55
  StrongCG: 1
  Inf proof: 17
  Zero half: 7
  Mod-K: 2
  RLT: 1
  Relax-and-lift: 30
  BQP: 2
  PSD: 9

Explored 5205 nodes (75288 simplex iterations) in 2.07 seconds (0.61 work units)
Thread count was 16 (of 16 available processors)

Solution count 10: 76 75 73 ... 46

Optimal solution found (tolerance 1.00e-04)
Best objective 7.600000000000e+01, best bound 7.600000000000e+01, gap 0.0000%
Optimal tour: [0, 14, 4, 3, 12, 9, 11, 15, 10, 6, 17, 19, 2, 18, 16, 1, 8, 7, 0]
Total collected score: 84
Total travel time: 295.18
```