Homework2

Zhesheng Xu
Student ID: 42353012

May 13th 2025

1 Problem Description

This document addresses the optimization problem faced by a grocery store chain operating 10 stores in a city. The objective is to maximize total revenue by selecting which stores to keep open, subject to the constraint that no two stores within 2 miles of each other can both be open. Each store's monthly revenue (in thousands of dollars) is provided, along with proximity relationships between stores.

1.1 Store Revenues

Store	Monthly Revenue ($1,000s)
1	127
2	83
3	165
4	96
5	112
6	88
7	135
8	141
9	117
10	94

1.2 Proximity Constraints

Based on the provided data, the following proximity sets of stores are within 2 miles of each other:

- Stores 1, 2, and 4
- Stores 1 and 3
- Stores 4, 5, and 6
- Stores 6, 7, and 8

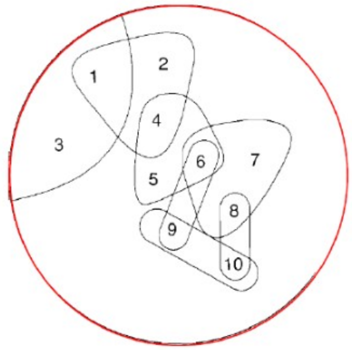

Figure 1: Store locations and 2-mile proximity relationships

- Stores 6 and 9
- Stores 8 and 10
- Stores 9 and 10

2 Model and Results

2.1 Decision Variables

Define binary variables to represent whether each store is open:

$$x_i = \begin{cases} 1 & \text{if store } i \text{ is open,} \\ 0 & \text{if store } i \text{ is closed,} \end{cases} \quad \text{for } i = 1, 2, \ldots, 10.$$

2.2 Objective Function

Maximize the total revenue:

Maximize $Z = 127x_1 + 83x_2 + 165x_3 + 96x_4 + 112x_5 + 88x_6 + 135x_7 + 141x_8 + 117x_9 + 94x_{10}$

2.3 Constraints

Ensure no two stores within 2 miles are both open, based on the updated proximity sets:

$$\begin{aligned} x_1 + x_2 + x_4 &\leq 1 \\ x_1 + x_3 &\leq 1 \\ x_4 + x_5 + x_6 &\leq 1 \\ x_6 + x_7 + x_8 &\leq 1 \\ x_6 + x_9 &\leq 1 \\ x_8 + x_{10} &\leq 1 \\ x_9 + x_{10} &\leq 1 \\ x_i &\in \{0, 1\} \quad \text{for } i = 1, 2, \ldots, 10 \end{aligned}$$

3 Source Code and Output

3.1 Source Code

The Gurobi Python code used to solve this problem is:

```python
import gurobipy as gp
from gurobipy import GRB
import numpy as np

# Define the revenue data for each store ($1000s)
revenues = {
    1: 127,
    2: 83,
    3: 165,
    4: 96,
    5: 112,
    6: 88,
    7: 135,
    8: 141,
    9: 117,
    10: 94
}

# Create a model
model = gp.Model("Grocery_Store_Optimization")

# Create binary decision variables for each store
# x[i] = 1 if store i is kept open, 0 otherwise
x = model.addVars(revenues.keys(), vtype=GRB.BINARY, name="x")

# Set objective: maximize total revenue
model.setObjective(
    gp.quicksum(revenues[i] * x[i] for i in revenues.keys()), GRB.
    MAXIMIZE
)

# Define the proximity sets (stores within 2 miles of each other)
```

```python
proximity_sets = [
    [1, 2, 4],
    [1, 3],
    [4, 5, 6],
    [6, 7, 8],
    [6, 9],
    [8, 10],
    [9, 10],
]

# Add constraints: at most one store in each proximity set can be open
for i, proximity_set in enumerate(proximity_sets):
    model.addConstr(
        gp.quicksum(x[j] for j in proximity_set) <= 1, f"proximity_set_{i+1}"
    )

# Optimize the model
model.optimize()

# Print results
if model.status == GRB.OPTIMAL:
    print(f"Optimal solution found with total revenue: ${model.objVal:.0f}k")
    print("\nStores to keep open:")
    total_revenue = 0
    open_stores = []
    for i in revenues.keys():
        if x[i].x > 0.5:  # If the store is open (allowing for small numerical errors)
            print(f"Store {i}: ${revenues[i]}k")
            total_revenue += revenues[i]
            open_stores.append(i)
    print(f"\nTotal revenue: ${total_revenue}k")
    print(f"Open stores: {open_stores}")
else:
    print("No optimal solution found.")

# Verify the solution by checking constraints
print("\nVerifying solution:")
for i, proximity_set in enumerate(proximity_sets):
    open_in_set = sum(1 for j in proximity_set if x[j].x > 0.5)
    if open_in_set > 1:
        print(
            f"ERROR: More than one store is open in proximity set {i+1}: {proximity_set}"
        )
    else:
        print(f"Constraint for proximity set {i+1} {proximity_set} is satisfied.")

# Calculate how many stores are kept open
stores_kept_open = sum(1 for i in revenues.keys() if x[i].x > 0.5)
print(f"\nTotal stores kept open: {stores_kept_open} out of 10")
```

Listing 1: Gurobi Python Code

3.2 Optimal Solution

The optimal solution suggests keeping open the following stores:

- Store 2: \$83,000
- Store 3: \$165,000
- Store 5: \$112,000
- Store 8: \$141,000
- Store 9: \$117,000

3.3 Maximum Revenue

The total revenue from the open stores is:

$$83 + 165 + 112 + 141 + 117 = 618 \text{ (thousands of dollars)} = \$618,000$$

3.4 Optimization Output

```
Optimize a model with 7 rows, 10 columns and 17 nonzeros
Model fingerprint: 0xddd7eb16
Variable types: 0 continuous, 10 integer (10 binary)
Coefficient statistics:
  Matrix range     [1e+00, 1e+00]
  Objective range  [8e+01, 2e+02]
  Bounds range     [1e+00, 1e+00]
  RHS range        [1e+00, 1e+00]
Found heuristic solution: objective 491.0000000
Presolve removed 7 rows and 10 columns
Presolve time: 0.01s
Presolve: All rows and columns removed

Explored 0 nodes (0 simplex iterations) in 0.02 seconds (0.00 work units)
Thread count was 1 (of 16 available processors)

Solution count 2: 618 491

Optimal solution found (tolerance 1.00e-04)
Best objective 6.180000000000e+02, best bound 6.180000000000e+02, gap 0.0000%
Optimal solution found with total revenue: $618k

Stores to keep open:
Store 2: $83k
Store 3: $165k
Store 5: $112k
```

```
Store 8: $141k
Store 9: $117k

Total revenue: $618k
Open stores: [2, 3, 5, 8, 9]

Verifying solution:
Constraint for proximity set 1 [1, 2, 4] is satisfied.
Constraint for proximity set 2 [1, 3] is satisfied.
Constraint for proximity set 3 [4, 5, 6] is satisfied.
Constraint for proximity set 4 [6, 7, 8] is satisfied.
Constraint for proximity set 5 [6, 9] is satisfied.
Constraint for proximity set 6 [8, 10] is satisfied.
Constraint for proximity set 7 [9, 10] is satisfied.

Total stores kept open: 5 out of 10
```

4 Conclusion

By using integer programming to optimize the grocery store chain's operations, we have determined that keeping stores 2, 3, 5, 8, and 9 open would maximize revenue while respecting the proximity constraints. This solution gives a total monthly revenue of $618,000, which is the optimal value that can be achieved under the given constraints.