Freshman Seminar Assignment Problem

Keenan Gao Binghui Ouyang
Hanwen Zhang Yiming Zong

Problem Overview

- Real Problem for Dietrich College
- Assigning freshmen to seminars based on their rankings.
- Data Size: Around 350 students, 22 seminars.
    - Seminar capacity: 16 students

Mathematical Model

- n: # students (n > 0); m: #seminars (m > 0);
- p: Max number of students allowed in one seminar;
- k: Max number of selections that a student can rank (1 ≤ k ≤ m);
- s_{ij} : The j-th selection of i-th student, where 1 ≤ i ≤ n and 1 ≤ j ≤ k s_{ij} = 0 when the Student i makes no corresponding choice for Rank j;
- q_k : The quota for k-th seminar, where 1 ≤ k ≤ m.

Mathematical Model (Cont'd)

Decision Variables:

- Y_{ij} : Indicator variables for whether Student i is assigned to Seminar j, where $1 \le i \le n$ and $1 \le j \le k$;

Data Pre-Processing

- Student Preferences: 1 top choice and 3 second choices
- Raw Data Format:

Student ID	Rankings	Final Assignment
student1	2;3,4,5	3

- Pre-processed Data Format:

$$X_{i,j} = \begin{cases} l & \text{If Student } i \text{ ranked } j \text{ as } l^{\text{th}} \text{ option, or } s_{i,l} = j \text{ for some } l \in \{1, \cdots, k\} \\ M & \text{If Seminar } k \text{ is not on Student } i\text{'s list, or } s_{i,l} \neq j \text{ for all } l \in \{1, \cdots, k\} \end{cases},$$

Note: ℓ is not unique, and $\mathcal{M}$ is an arbitrary large value.

Initial Objective Function & Constraints

Minimize $$\sum_{i=1}^{n}\sum_{j=1}^{m} X_{ij}Y_{ij}$$

Subject to $$\sum_{j=1}^{m} Y_j = 1 \qquad X_{i,j} > 0 \; for \; \forall \, i,j \in \{1..n\}$$

Problem:

- The variance of the rank distributions in the final assignment might be big.
- i.e. seminar 1 gets all students who rank it as their 1st choices while seminar 2 gets all students who have no interest.

Modified Objective Function

Goal: Minimize the variance

- All seminars are full and students are assigned in the classes that they have interest in.

Approach:

- For each seminar first assign q_k first choice students. Then fill in the rest *p* - q_k spots with second choice/no-interest students.
- Need to optimize q_k

Optimization Algorithms

- Minimize Total Rank:
  - Given student-seminar cost matrix
  - Create dummy seminars based on quota
  - Create dummy students to make square cost matrix
  - Run Hungarian Algorithm
  - Interpret Hungarian output

Optimization Algorithms (Cont’d)

- Minimize ranking variance
  - For each seminar, assign up to Q students that list it as top choice
  - Match as many second-tier as possible
  - Fill in remaining students
  - Repeat process above for best result

Algorithm Implementation

- Implemented with Python
- Efficient matrix manipulation with Numpy
- Input File Format:
    - Each line contains a student’s preference
    - Encoded as “1:2,3,4”
- Output Files:
    - One output for minimizing total rank;
    - Three best outputs for result with artificial quota
- Code available at https://github.com/ymzong/OpResearchF14

Outcome

- Minimize Total Rank with Hungarian
  - Total Students: 308
  - Assigned to Top Choice: 207 (67.2%)
  - Assigned to Second Choice: 82 (26.6%)
  - Default Random Assignment: 19 (6.2%)
  - All seminars except for one are filled entirely

Outcome (Cont'd)

- Minimize Variance with Hungarian
  - 4 minutes per run
  - Need to re-run for each value of *q* to determine optimal *q*
  - Too slow for general purpose
- Minimize Variance with Randomization
  - Work in progress
  - Flexible per number of iterations

Notes and Further Work

- Degree of “cost minimization” will vary
  - Easier to achieve cost minimizations if student preferences are initially diverse
  - Will vary from year to year
- Optimizing for speed
  - Hungarian Algorithm vs. Randomized Assignment

Summary

- Problem Statement
- Mathematical Model & Data Pre-Processing
- Optimization Algorithms
    - Minimize total rank with Hungarian
    - Minimize ranking variance with Randomization
- Algorithm Implementation
- Outcome

Acknowledgments

- **Professor Alan Frieze**
  - regular meetings and guidance
- **Brian Junker, Joseph Devine, Gloria Hill**
  - provided us with real assignment data from 2013
- **Brian Clapper**
  - Python implementation of the Hungarian algorithm

Questions?