Seminar 1: Optimizing Conference Paper Assignment Using Genetic Algorithms

Intelligent Systems

October 28, 2024

Introduction

In this seminar assignment, the primary objective is to utilize genetic algorithms to develop a smart and efficient system for assigning conference papers to reviewers. The system must optimize assignments to maximize overall satisfaction by aligning reviewer preferences while satisfying all constraints. This optimization task is essential for conference organizers to automate the paper-review process effectively, ensuring fairness and efficiency. The task must consider multiple constraints, such as the number of papers each reviewer can handle, the number of reviews each paper needs, friendship relationships between reviewers to prevent conflicts of interest (e.g., friends reviewing the same paper), and **authorship constraints** where reviewers cannot review papers they have authored.

Dataset Examples

Example 1

- **Number of Papers (N):** 5
- **Number of Reviewers (K):** 5
- **Reviewer Capacity (R):** 3
- **Review Requirements per Paper:** Each paper must receive between 3 and 5 reviews.
- **Authorship Constraints:** Certain reviewers have authored specific papers and cannot review them.

Preference Matrix (P_{ij}):

Reviewer \Paper	Paper 1	Paper 2	Paper 3	Paper 4	Paper 5
Reviewer 1	5	4	5	3	4
Reviewer 2	4	5	4	5	3
Reviewer 3	3	4	5	4	5
Reviewer 4	5	3	4	5	4
Reviewer 5	4	5	3	4	5

Friendship Matrix ($F_{ii'}$):

Reviewer \Reviewer	Rev 1	Rev 2	Rev 3	Rev 4	Rev 5
Reviewer 1	0	1	0	0	0
Reviewer 2	1	0	1	0	0
Reviewer 3	0	1	0	1	0
Reviewer 4	0	0	1	0	1
Reviewer 5	0	0	0	1	0

Authorship Matrix (A_{ij}):

Reviewer \Paper	Paper 1	Paper 2	Paper 3	Paper 4	Paper 5
Reviewer 1	1	0	0	0	0
Reviewer 2	0	1	0	0	0
Reviewer 3	0	0	1	0	0
Reviewer 4	0	0	0	1	0
Reviewer 5	0	0	0	0	1

Example 2

- **Number of Papers (N):** 6
- **Number of Reviewers (K):** 8
- **Reviewer Capacity (R):** 2
- **Review Requirements per Paper:** Each paper must receive between 3 and 5 reviews.
- **Authorship Constraints:** Certain reviewers have authored specific papers and cannot review them.

Preference Matrix (P_{ij}):

Reviewer \Paper	Paper 1	Paper 2	Paper 3	Paper 4	Paper 5	Paper 6
Reviewer 1	5	2	3	4	1	5
Reviewer 2	4	5	2	3	5	1
Reviewer 3	3	1	5	2	4	3
Reviewer 4	2	3	4	5	2	4
Reviewer 5	1	4	1	2	5	5
Reviewer 6	5	5	3	1	3	2
Reviewer 7	2	3	5	4	1	3
Reviewer 8	3	2	1	5	4	2

Friendship Matrix ($F_{ii'}$):

Reviewer \Reviewer	Rev 1	Rev 2	Rev 3	Rev 4	Rev 5	Rev 6	Rev 7	Rev 8
Reviewer 1	0	1	0	0	0	0	0	0
Reviewer 2	1	0	1	0	0	0	0	0
Reviewer 3	0	1	0	1	0	0	0	0
Reviewer 4	0	0	1	0	1	0	0	0
Reviewer 5	0	0	0	1	0	1	0	0
Reviewer 6	0	0	0	0	1	0	1	0
Reviewer 7	0	0	0	0	0	1	0	1
Reviewer 8	0	0	0	0	0	0	1	0

Authorship Matrix (A_{ij}):

Reviewer \Paper	Paper 1	Paper 2	Paper 3	Paper 4	Paper 5	Paper 6
Reviewer 1	1	0	0	0	0	0
Reviewer 2	0	1	0	0	0	0
Reviewer 3	0	0	1	0	0	0
Reviewer 4	0	0	0	1	0	0
Reviewer 5	0	0	0	0	1	0
Reviewer 6	0	0	0	0	0	1
Reviewer 7	0	0	0	0	0	0
Reviewer 8	0	0	0	0	0	0

Task 1 - Representation (25%)

Develop a function to process the reviewer preferences and yield the optimal assignment of papers to reviewers, as derived by the genetic algorithm. The primary steps involve:

- **Converting the preference data into an appropriate format.**
  - *Hint:* First, write a data structure to represent the preference matrix, friendship matrix, and authorship constraints.

- **Deciding on an effective solution representation.**
  - Represent assignments as a matrix where each cell indicates whether a reviewer is assigned to a paper.
- **Designing a fitness function.**
  - The fitness function should evaluate how well an assignment maximizes the preference score while minimizing constraint violations, including reviewer capacity, paper review requirements, friendship constraints, and authorship constraints.
- **Implementing and executing the genetic algorithm.**
  - *Hint:* An initial solution can be generated using generative AI, but it should be further refined to meet all constraints effectively.

Task 2 - Crossover and Mutation (30%)

Standard crossover and mutation functions might not be ideal for evolving assignment solutions, as they may produce invalid or infeasible assignments. Thus, it's pivotal to:

- **Modify the crossover and mutation functions to ensure the generation of valid assignments.**
  - Ensure that the offspring solutions satisfy all constraints, including reviewer capacities, friendship relationships, and authorship constraints.
- **Consider the structure and constraints of the assignment problem during mutation and crossover.**
  - Maintain the integrity of reviewer capacities, friendship constraints, and authorship constraints when performing genetic operations.
- **Utilize existing genetic algorithm libraries, such as PyGAD, and adapt them to suit this problem.**
  - *Hint:* While libraries provide standard functions, customizing them to handle specific constraints will yield better performance.

Note: An initial solution can be generated using generative AI, but students are encouraged to design custom and more advanced crossover and mutation functions for optimized performance and higher scores. Specify the prompts used for generating the initial solution and discuss subsequent improvements made manually.

Task 3 - Complexity and Diversity (20%)

Introduce diversity in the genetic algorithm to explore a broader space of potential solutions. This can include varying genetic operators and incorporating additional constraints, such as ensuring that friends and authors do not review the same paper. The goal is to allow the algorithm to navigate complex assignment scenarios. Additionally, when constructing the assignments, aim for efficiency by penalizing violations like over-assignment or under-assignment.

Task 4 - Evaluation and Report (25%)

Compile a comprehensive report detailing your approach, showcasing code highlights, and presenting the results. Prepare a single notebook for the results. It's crucial to:

- **Compare the performance under different genetic algorithm configurations** (e.g., different mutation rates, crossover methods, selection criteria).
- **Assess the approach across varied datasets to understand its robustness and versatility.**
- **Visualize how the complexity of the dataset affects the algorithm's runtime and efficiency using appropriate graphs.**

1 Submission

- **Deadline:** December 1, 2024
- **Format:** Jupyter Notebook
- **Group Work:** Maximum of two people per group (if you need a partner, please contact an assistant)
- **Use of Generative AI:** Using ChatGPT to initiate the solution is permitted. Please attach the conversation as a single `.txt` file alongside your Jupyter Notebook.