Salsabeel Dalaq

COMP SCI 461

Apr 20, 2025

1. Challenges in Writing the Program:

After planning the basic structure, the development of the genetic algorithm-based scheduler went relatively smoothly. One of the main challenges was handling multiple constraints simultaneously, such as avoiding double-booked rooms or facilitators and matching facilitator preferences with activities. Implementing the fitness function required careful thought to ensure it rewarded good behavior (like preferred facilitators or minimal room underutilization) while penalizing conflicts effectively. Also, managing population initialization, crossover, and mutation in a balanced way required some tuning, especially for mutation rates and elitism strategies.

1. Evaluation of the Generated Schedule:

Overall, the schedule generated by the algorithm was logically sound and respected the major constraints. The best schedules typically avoided major conflicts and used room space efficiently. However, in some runs, a few oddities remained, such as less-preferred facilitators being assigned when a preferred one was available, or SLA100A and SLA100B not always aligning at the same time. These inconsistencies are likely due to a combination of random factors and the relative weightings in the fitness function.

1. Potential Improvements:

If I were to continue developing the program, I would consider the following improvements:

* Enhancing the fitness function: Incorporating penalties for minor inefficiencies (like excessive overcapacity) more gradually or using a more nuanced scoring system.
* Time-slot balancing: Ensuring that the same facilitator isn't used too frequently across unrelated sessions.
* Crossover and mutation enhancements: Applying smarter crossover (e.g., uniform crossover) or adaptive mutation rates that respond dynamically to fitness stagnation could help the algorithm converge faster.
* Constraint satisfaction: Adding a repair function after mutation to fix hard constraint violations, ensuring feasible solutions are always maintained.

1. Final Thoughts:

This project was a great exercise in both algorithmic thinking and practical problem-solving. Seeing the genetic algorithm iteratively improve the population over generations is satisfying. Eclipse and Java were effective tools for this project, although managing file output required a quick refresh trick in the IDE. I’d be interested in extending this project into a web-based scheduler or visualizing the generation evolution graphically. Also, not gonna lie, it was kind of cool seeing a "best schedule" evolve out of chaos!