Development & Implementation of a Matching Algorithm for Student-Project Allocation

Project defence presentation

June 17, 2025

Walid LAHDADI - Majda KEMMOU - Oussama SOUIDI

Introduction

- Importance of allocating students to projects effectively
- Manual allocation challenges: time-consuming, error-prone
- Motivation for automation and optimization

Agenda

- Background: Gale-Shapley algorithm & challenges
- Student Project Allocation (SPA) algorithm
- Mathematical model & optimization
- Implementation & web interface
- Results & performance evaluation
- Discussion & comparison
- Conclusion & future work

Background: Gale-Shapley algorithm

- Stable marriage problem foundation
- One-to-one matching explained
- Proposal and acceptance concept
- Limitations in many-to-one context (projects with multiple students)

Challenges with Gale-Shapley

- Difficulty handling many-to-one allocations
- Instability risks when quotas > 1
- Computational complexity with grouping students
- Need for a more flexible approach

SPA algorithm overview

- Purpose-built many-to-one stable matching algorithm
- Supports student project preferences & project quotas
- Produces stable matchings respecting constraints

Proposed Solution : MILP

- **Research paper :** “Handling preferences in student-project allocation” The university of Southern Denmark - 2017

- We model the allocation problem as a **Mixed Integer Linear Program**.

How the algorithm works

Step 1: Input Data Preparation

- Collect inputs:
    - **Students:** Each student provides a ranked list of preferred projects.
    - **Projects:** Each project has a minimum and maximum capacity.

How the algorithm works

Step 2: Define Decision Variables

- For each student s and each project p in their preference list:
    - **x_s,p = 1** if student s is assigned to project p, otherwise 0.
- For each project p:
    - **y_p = 1** if project p is active (has at least one student), otherwise 0.
- **z:** represents **the worst rank** (maximum dissatisfaction) across all student assignments.

How the algorithm works

Step3 : Minimize the Worst Rank (z)

- The algorithm adds a variable z to represent the worst rank assigned to any student.
- Then it adds a constraint for every student:
- → If student i is assigned project j, and j is their rank r_ij, then r_ij ⩽ z.
- This forces **z** to be at least as bad as the worst rank used, and the solver will try to **make z as small as possible.**
- This way, **the optimization** is focused on **fairness**: making sure no student gets a really bad project.

How the algorithm works

Step 4: Solve & Extract Solution

The MILP solver searches for values of x[i][j] (assignments) that:

- Respect all constraints (1 project per student, capacity limits)
- Ensure that if a student gets a project, its rank $\leqslant$ z
- And minimize z (so all assigned ranks are as good as possible)

Results overview

MILP Solution (Automated):

Rank Distribution

- **Rank 1:** 61 students
- **Rank 2:** 7 students
- **Rank 3:** 4 students

Manual Allocation:

Rank Distribution

- **Rank 1:** 59 students
- **Rank 2:** 8 students
- **Rank 3:** 4 students
- **1 student left unmatched** (assigned to a non-preferred project)

Results overview

- 72 students allocated successfully
- 84.7% received first choice, 9.7% second, 5.6% third
- Zero unmatched students
- ~92% overall satisfaction

Another Proposed Solution : (Based on Gale Shapley algorithm)

- **Research paper :** “Two Algorithms for the Student-Project Allocation Problem” by Abraham et al. (2004).
- Constraints:
  - Each project has a capacity.
  - Lecturers (implicitly via project preferences) have ranked preferences.
  - Students can express preferences over a limited set of projects.

Implementation

- Inputs:
  - students.csv: student preferences
  - projects.csv: project quotas and preferences
- Results and Output
  - Output Matching printed and saved as CSV
  - Summary Statistics:
    - Number of students who got 1st/2nd/3rd choice
    - Number of unmatched students

How the Algorithm Works

- Step 1: Initialization
  - All students are unmatched
  - All projects are empty
  - Each student has a list of preferred projects
  - Each project has a quota
- Step 2: Student Proposes
  - Pick any unmatched student with remaining preferences
  - The student proposes to their top remaining project

How the Algorithm Works

- Step 3: Tentative Assignment
  - The project tentatively accepts the student
  - If it exceeds its quota, it rejects the least preferred among all assigned students (according to the project’s ranking)
- Step 4: Update Lists
  - The rejected student becomes unmatched again
  - Any student worse than the worst accepted is removed from the project’s preference list

How the Algorithm Works

- Step 5: Repeat
  - Keep iterating until there are no unmatched students with available preferences

- The result is a stable matching:
  - No unmatched student and project would rather be with each other than with their current match

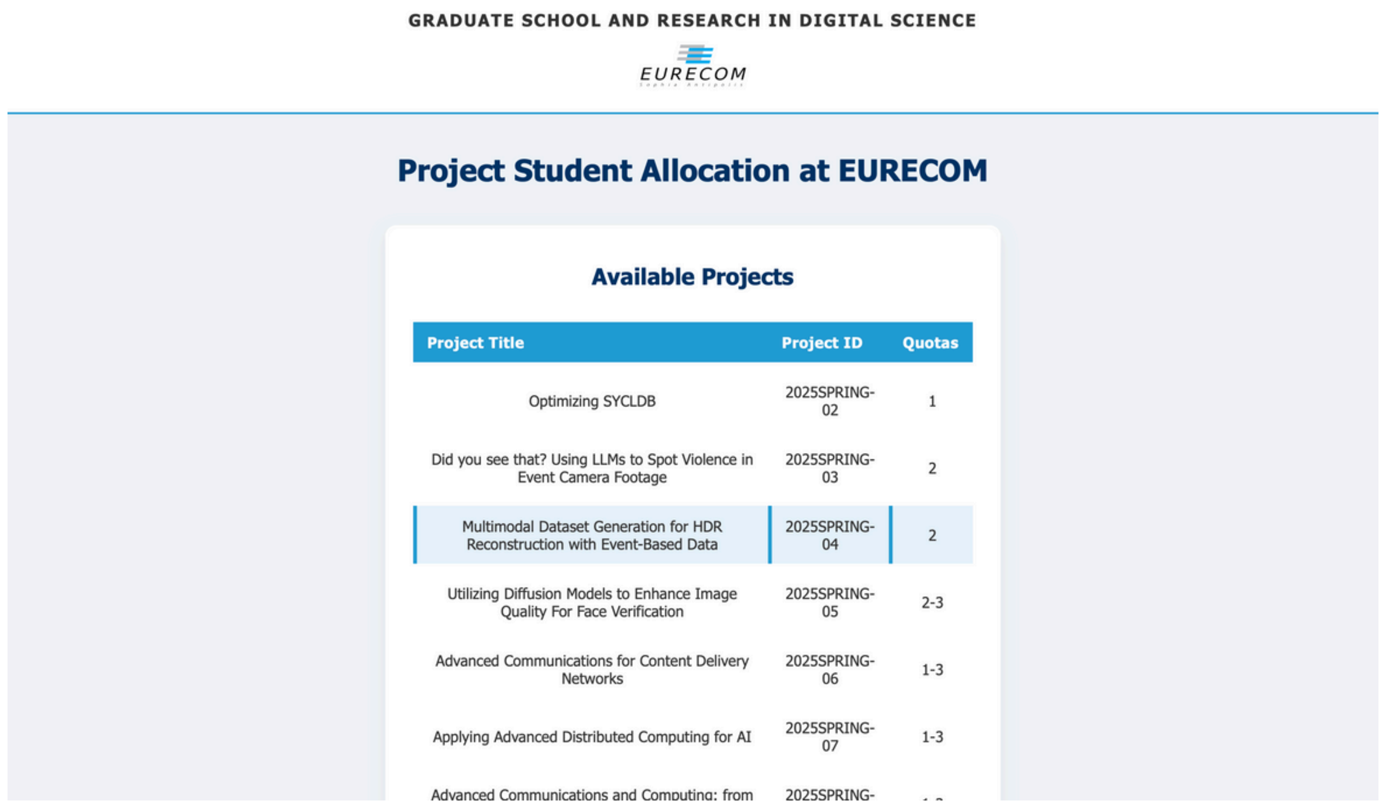
Results overview

- Total students: 72
- Matched students: 68
- Unmatched students: 4
- Students who got their 1st choice: 56
- Students who got their 2nd choice: 7
- Students who got their 3rd choice: 5

Implementation overview

- Dual web interface: Students & Administrators
- Students rank projects; admins run algorithm and export results
- Backend algorithm deployment and Excel output

Student web interface



GRADUATE SCHOOL AND RESEARCH IN DIGITAL SCIENCE

EURECOM

Project Student Allocation at EURECOM

Available Projects

Project Title	Project ID	Quotas
Optimizing SYCLDB	2025SPRING-02	1
Did you see that? Using LLMs to Spot Violence in Event Camera Footage	2025SPRING-03	2
Multimodal Dataset Generation for HDR Reconstruction with Event-Based Data	2025SPRING-04	2
Utilizing Diffusion Models to Enhance Image Quality For Face Verification	2025SPRING-05	2-3
Advanced Communications for Content Delivery Networks	2025SPRING-06	1-3
Applying Advanced Distributed Computing for AI	2025SPRING-07	1-3
Advanced Communications and Computing: from	2025SPRING-	

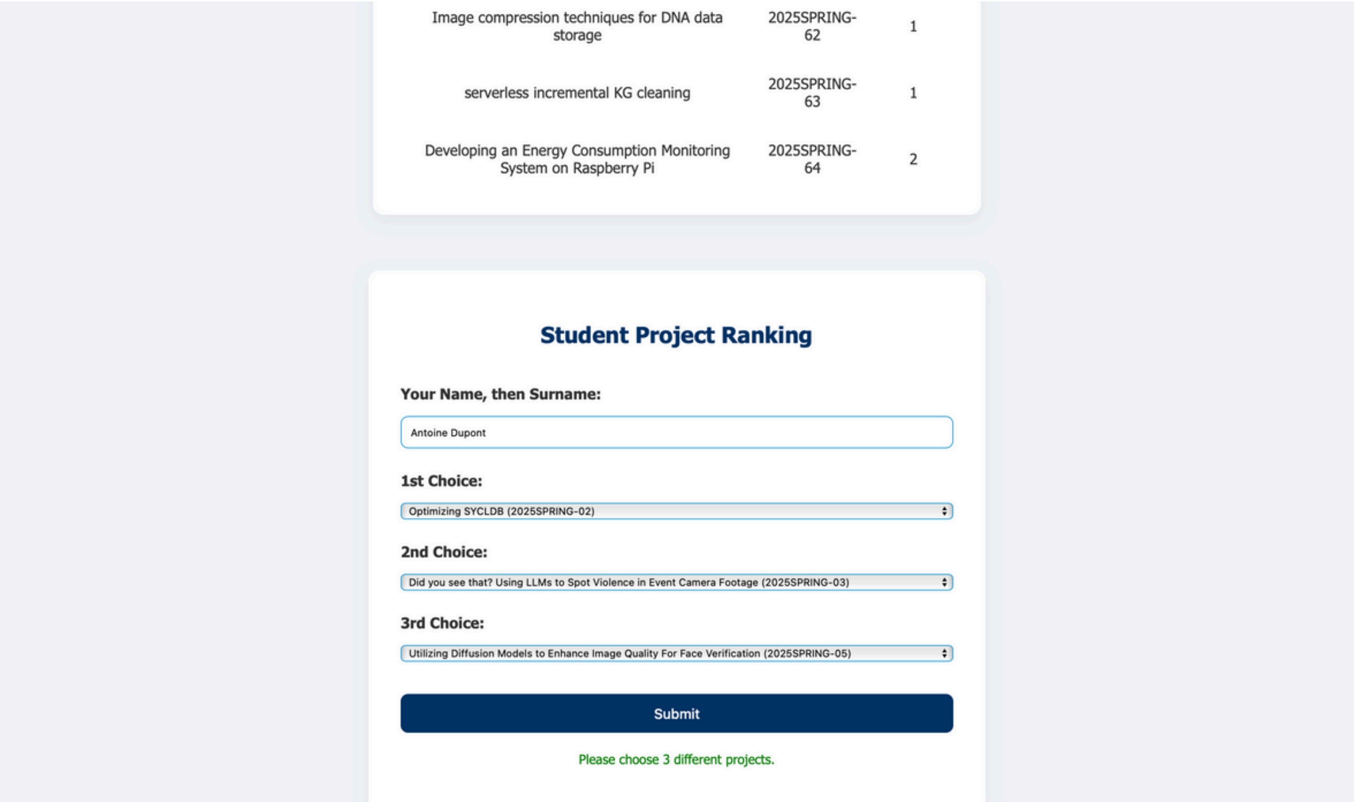


Image compression techniques for DNA data storage	2025SPRING-62	1
serverless incremental KG cleaning	2025SPRING-63	1
Developing an Energy Consumption Monitoring System on Raspberry Pi	2025SPRING-64	2

Student Project Ranking

Your Name, then Surname:

Antoine Dupont

1st Choice:

Optimizing SYCLDB (2025SPRING-02)

2nd Choice:

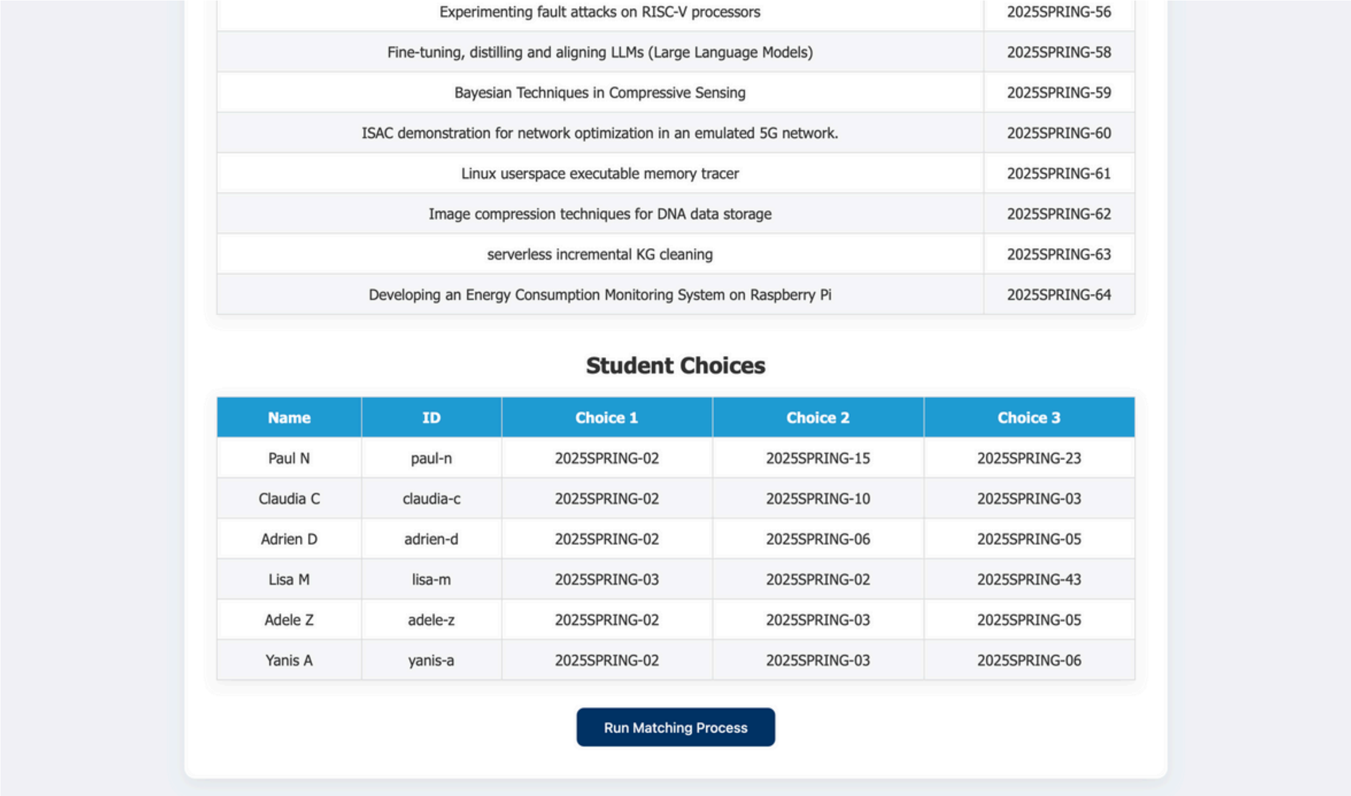
Did you see that? Using LLMs to Spot Violence in Event Camera Footage (2025SPRING-03)

3rd Choice:

Utilizing Diffusion Models to Enhance Image Quality For Face Verification (2025SPRING-05)

Please choose 3 different projects.

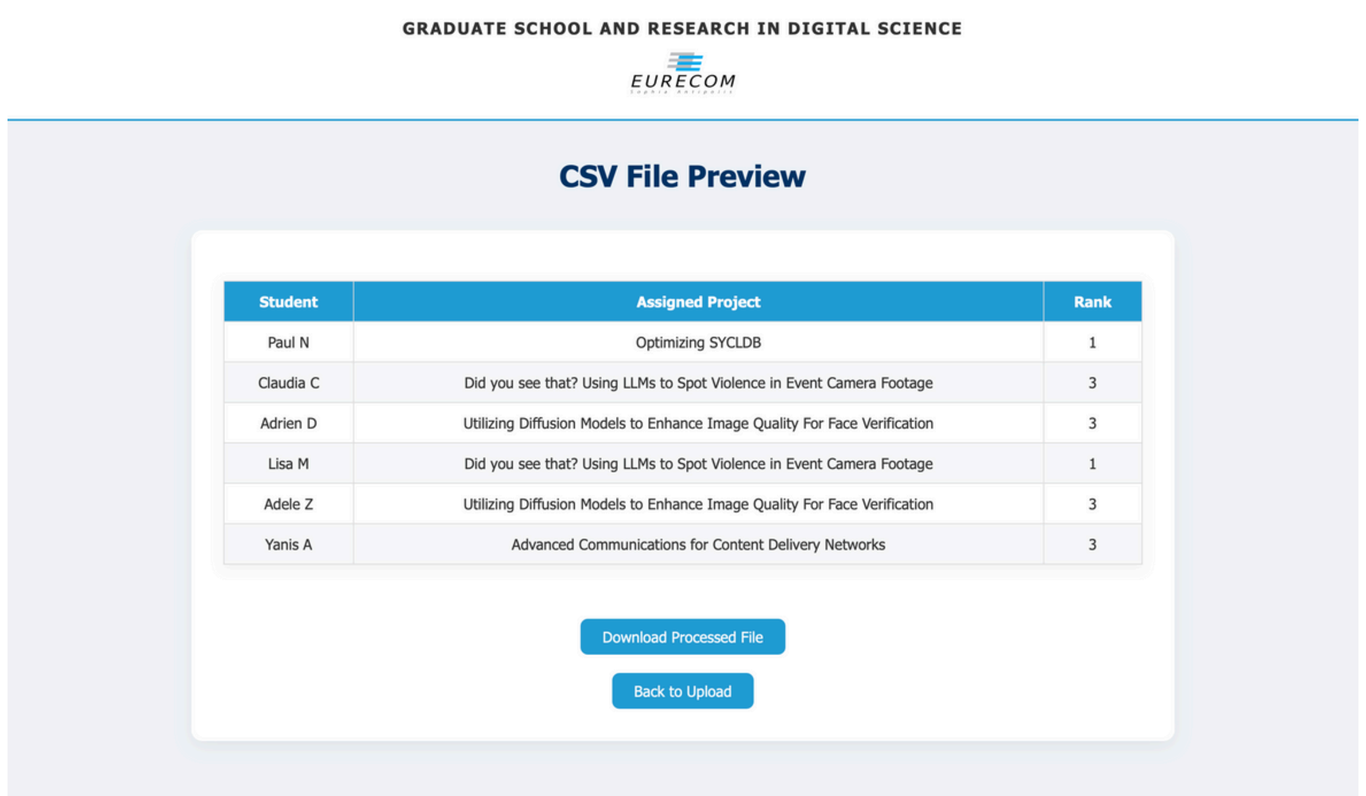
Admin web interface



Experimenting fault attacks on RISC-V processors	2025SPRING-56
Fine-tuning, distilling and aligning LLMs (Large Language Models)	2025SPRING-58
Bayesian Techniques in Compressive Sensing	2025SPRING-59
ISAC demonstration for network optimization in an emulated 5G network.	2025SPRING-60
Linux userspace executable memory tracer	2025SPRING-61
Image compression techniques for DNA data storage	2025SPRING-62
serverless incremental KG cleaning	2025SPRING-63
Developing an Energy Consumption Monitoring System on Raspberry Pi	2025SPRING-64

Student Choices

Name	ID	Choice 1	Choice 2	Choice 3
Paul N	paul-n	2025SPRING-02	2025SPRING-15	2025SPRING-23
Claudia C	claudia-c	2025SPRING-02	2025SPRING-10	2025SPRING-03
Adrien D	adrien-d	2025SPRING-02	2025SPRING-06	2025SPRING-05
Lisa M	lisa-m	2025SPRING-03	2025SPRING-02	2025SPRING-43
Adele Z	adele-z	2025SPRING-02	2025SPRING-03	2025SPRING-05
Yanis A	yanis-a	2025SPRING-02	2025SPRING-03	2025SPRING-06



GRADUATE SCHOOL AND RESEARCH IN DIGITAL SCIENCE

EURECOM

CSV File Preview

Student	Assigned Project	Rank
Paul N	Optimizing SYCLDB	1
Claudia C	Did you see that? Using LLMs to Spot Violence in Event Camera Footage	3
Adrien D	Utilizing Diffusion Models to Enhance Image Quality For Face Verification	3
Lisa M	Did you see that? Using LLMs to Spot Violence in Event Camera Footage	1
Adele Z	Utilizing Diffusion Models to Enhance Image Quality For Face Verification	3
Yanis A	Advanced Communications for Content Delivery Networks	3

Class Timetabling

Problem Statement:

- Allocate school courses to rooms and timeslots.
- Avoid conflicts (room or teacher overlap).
- Respect teacher preferences and room capacities.
- satisfaction, and minimal constraint violations.

Class Timetabling

Input Data:

- **Courses:** Name, teacher, number of students, duration (short/long).
- **Rooms:** Name, type (classroom or amphitheater), capacity.
- **Timeslots:** Combinations of day and time (e.g., Monday AM), excluding unavailable periods.
- **Teacher Constraints:** Availability and preferences.

Goal:

- Generate valid and optimized timetables with no conflicts, high satisfaction, and minimal constraint violations.

How the algorithm works

Step 1: Initialization

- Generate a random set of possible timetables (solutions).

Step 2: Evaluation

- Score each timetable based on:
  - Hard constraints: No conflicts (room/teacher), room capacity respected.
  - Soft constraints: Teacher preferences, availability.

Step 3: Selection

- Keep the best-scoring solutions.

How the algorithm works

Step 4: Crossover & Mutation

- Crossover: Combine two good timetables.
- Mutation: Small changes to improve diversity.

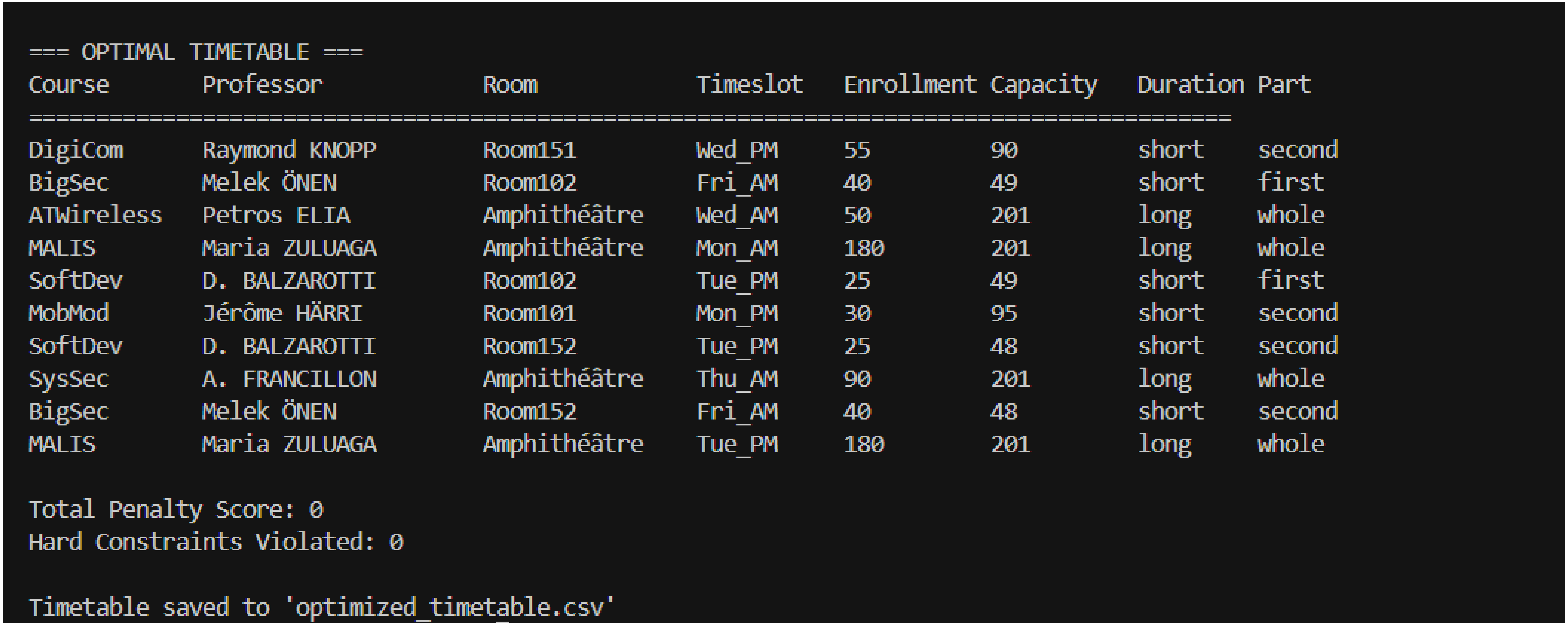
Step 5: Local Optimization

- Slightly adjust good solutions to fix small issues (e.g., better time or room).

Final Output:

- A valid timetable: conflict-free, fair, and preference-aware.

Results Overview



```
=== OPTIMAL TIMETABLE ===
Course       Professor            Room            Timeslot   Enrollment Capacity   Duration Part
==========================================================================================
DigiCom      Raymond KNOPP        Room151         Wed_PM     55         90         short    second
BigSec       Melek ÖNEN           Room102         Fri_AM     40         49         short    first
ATWireless   Petros ELIA          Amphithéâtre    Wed_AM     50         201        long     whole
MALIS        Maria ZULUAGA        Amphithéâtre    Mon_AM     180        201        long     whole
SoftDev      D. BALZAROTTI        Room102         Tue_PM     25         49         short    first
MobMod       Jérôme HÄRRI         Room101         Mon_PM     30         95         short    second
SoftDev      D. BALZAROTTI        Room152         Tue_PM     25         48         short    second
SysSec       A. FRANCILLON        Amphithéâtre    Thu_AM     90         201        long     whole
BigSec       Melek ÖNEN           Room152         Fri_AM     40         48         short    second
MALIS        Maria ZULUAGA        Amphithéâtre    Tue_PM     180        201        long     whole

Total Penalty Score: 0
Hard Constraints Violated: 0

Timetable saved to 'optimized_timetable.csv'
```

Conclusion & future work

- Summary of achievements
- Potential enhancements