Progress Report: Automated Timetable Generation for FSTM using Metaheuristic Optimization

Group Members: Youssef Ait Bahssine , Mustapha Zmirli , Mohamed Bajadi
Module: Metaheuristics - Master in Artificial Intelligence

December 29, 2025

1 Introduction and Problematic

The University Timetabling Problem (UTP) is a highly complex combinatorial optimization problem, proven to be NP-complete. At the Faculty of Sciences and Techniques of Marrakech (FSTM), the challenge involves assigning a set of courses, teachers, and student groups to specific time slots and rooms while respecting a diverse set of institutional requirements.

The core problematic of this project lies in the idiosyncratic nature of these constraints; a "feasible" timetable must satisfy all mandatory requirements, while an "optimal" one must further minimize the violation of soft preferences to ensure the satisfaction of both students and faculty.

2 Problem Definition

The project objective is to design and implement an automated system that generates a weekly timetable by assigning five primary entities:

- Courses/Modules: The academic units to be scheduled.
- Time Slots: Available periods within the academic week.
- Classrooms / Laboratories: The physical locations for events.
- Teachers: The academic staff delivering the modules.
- Student Groups: The cohorts attending the sessions.

3 Constraints and Objective Function

Following the taxonomy provided by Lewis (2007), we classify our constraints into two categories:

3.1 Hard Constraints (Feasibility)

These are mandatory conditions. A solution is only valid if zero hard constraints are violated:

- Teacher Conflict: A teacher cannot teach two courses simultaneously.
- Student Group Conflict: A student group cannot attend more than one course at a time.
- Room Conflict: A room cannot host multiple courses at the same time.
- Room Capacity: The assigned room must accommodate the group size.
- Room Type: Courses requiring labs or amphitheaters must be assigned to appropriate room types.

3.2 Soft Constraints (Quality)

These represent preferences that improve the timetable's utility:

- Schedule Gaps: Minimizing idle periods (windows) for both teachers and students.
- Load Balancing: Distributing teaching hours evenly across the week.
- Session Timing: Minimizing very early or very late sessions.

3.3 Objective Function

We will use a Weighted Cost Function $f(S)$ to evaluate a solution S:

$$f(S) = \sum_i w_H \cdot H_i(S) + \sum_j w_S \cdot S_j(S)$$

where H_i are hard constraint violations, S_j are soft constraint violations, and $w_H \gg w_S$ to ensure feasibility is prioritized.

4 Methodology

Our approach will follow these development stages:

1. Formal Modeling: Representing the UTP as a graph-based or resource-assignment model.
2. Solution Encoding: Using a direct representation (e.g., an array where indices represent time/room slots).
3. Constraint Handling: Adopting a "Two-Stage" or "One-Stage" strategy as described in the survey.
4. Experimental Evaluation: Testing the algorithm against FSTM-specific data instances.

5 Potential Metaheuristic Algorithms

Based on the survey of state-of-the-art techniques, we will evaluate the following algorithms:

- Simulated Annealing: Useful for escaping local optima by allowing occasional "worse" moves early in the process.
- Evolutionary Algorithms (Genetic Algorithms): Population-based search to explore large solution spaces.

6 Mathematical Formulation of Constraints

To ensure clarity and facilitate implementation, we formalize the constraints using mathematical notation. This is based on standard UTP models.

6.1 Notation (Parameters)

- **Sets**:
  - C: Set of courses (e.g., —C— = 50 from FSTM data).
  - T: Set of teachers (—T— = 25).
  - G: Set of student groups (—G— = 20).
  - R: Set of rooms (—R— = 34).

- S: Set of time slots (—S— = 120).

- **Variables**:
  - $x_{c,t,g,r,s}$: Binary variable (1 if course c taught by teacher t to group g is assigned to room r at slot s; 0 otherwise).

- **Parameters**:
  - cap_r: Capacity of room r (e.g., 400 for amphitheaters, 48 for labs).
  - $size_g$: Size of student group g (e.g., 30-50).
  - $type_c$: Required room type for course c (e.g., 1 for lab).
  - $type_r$: Type of room r (matching $type_c$).
  - w_H, w_S: Weights for hard/soft constraints (e.g., $w_H = 1000$, $w_S = 1$).

6.2 Hard Constraints (Feasibility)

A solution is feasible if all are satisfied (violations = 0). Each is a penalty term in the objective.

1. **Teacher Conflict**: No teacher teaches multiple courses at the same slot. Formula:

$$H1 = \sum_{t \in T} \sum_{s \in S} \max\left(0, \sum_{c1 \neq c2, g1, g2, r1, r2} x_{c1,t,g1,r1,s} + x_{c2,t,g2,r2,s} - 1\right)$$

Explanation: For each teacher t and slot s, count overlaps beyond 1. Parameter justification: Ensures scheduling realism; t and s are from FSTM's teacher list and weekly slots.

2. **Student Group Conflict**: No group attends multiple courses at the same slot. Formula:

$$H2 = \sum_{g \in G} \sum_{s \in S} \max\left(0, \sum_{c1 \neq c2, t1, t2, r1, r2} x_{c1,t1,g,r1,s} + x_{c2,t2,g,r2,s} - 1\right)$$

Explanation: Similar to H1, for groups. Parameters: g and $size_g$ are critical for FSTM's diverse group sizes.

3. **Room Conflict**: No room hosts multiple events at the same slot. Formula:

$$H3 = \sum_{r \in R} \sum_{s \in S} \max\left(0, \sum_{c1 \neq c2, t1, t2, g1, g2} x_{c1,t1,g1,r,s} + x_{c2,t2,g2,r,s} - 1\right)$$

Explanation: Ensures exclusivity. r types from FSTM inventory.

4. **Room Capacity**: Assigned room must fit the group. Formula:

$$H4 = \sum_{c,t,g,r,s} x_{c,t,g,r,s} \cdot \max(0, size_g - cap_r)$$

Explanation: Penalizes if group exceeds capacity. Parameters: cap_r and $size_g$ are determined from FSTM data.

5. **Room Type**: Course must match room type. Formula:

$$H5 = \sum_{c,t,g,r,s} x_{c,t,g,r,s} \cdot |type_c - type_r|$$

Explanation: Binary mismatch. Parameters: $type_c$ (e.g., lab-required courses) and $type_r$ from data.

6.3 Soft Constraints (Quality)

Minimize violations for better schedules.

1. **Schedule Gaps**: Minimize idle slots for teachers/groups. Formula:

$$S1 = \sum_{t \in T} \sum_{day} (max\_gaps - actual\_consecutive\_slots)$$

Explanation: Encourages compact schedules. Parameters: max_gaps (e.g., 1; user-defined for FSTM preferences).

2. **Load Balancing**: Even distribution of teaching hours. Formula:

$$S2 = variance \left(\sum_{c,t,g,r,s} x_{\dots} \text{ for each } t \right)$$

Explanation: Reduces burnout. Parameters: Target hours (e.g., 20/week from FSTM norms).

3. **Session Timing**: Avoid early/late slots. Formula:

$$S3 = \sum_{c,t,g,r,s} x_{\dots} \cdot penalty_s$$

Explanation: Improves satisfaction. Parameters: $penalty_s$ (e.g., 10 for undesirable slots; based on surveys).

6.4 Objective Function Update

$$f(S) = \sum_i w_H \cdot H_i(S) + \sum_j w_S \cdot S_j(S)$$

Parameters: $w_H \gg w_S$ (e.g., 1000 vs. 1) to enforce hard constraints first.

7 Incorporation of FSTM Data

We analyzed the provided Excel data, a provisional timetable for FSTM's Autumn 2025-2026 session. Key findings:

- **Time Slots**: 120 slots/week (20/day $\times 6$ days, 08:30-18:30 in 30-min increments), but only 40% occupied—ideal for gap minimization.
- **Rooms**: 35 rooms (capacities: 48-400; types: amphitheaters A1-A4=400, labs/classrooms S5-G2=48-90).
- **Courses**: 50 codes (e.g., MIASI, ISA, GMP), often combined.
- **Assignments**: Entries show teacher-group-room linkages. Observed conflicts in data.
- **Insights**: High lab demand, uneven usage, amphitheater blocks.

8 Planning for Next Steps

To ensure timely completion from today (December 29, 2025) to January 7, 2026, we adopt a compressed agile timeline, focusing on critical deliverables within approximately 10 days.

- **Days 1-2 (Dec 29-30: Data Processing)**: Clean Excel data, validate parameters (e.g., capacities, group sizes). Deliverable: Processed dataset in CSV format. Risk: Data errors—mitigate with group reviews and quick surveys.

- **Days 3-5 (Dec 31-Jan 2: Algorithm Implementation)**: Code Simulated Annealing prototype with constraint handling. Deliverable: Working prototype that generates feasible timetables. Risk: Bugs in code—mitigate with unit testing and pair programming.
- **Days 6-8 (Jan 3-5: Evaluation)**: Run simulations on FSTM data, evaluate performance (feasibility, soft violations), and compare with baseline (e.g., manual assignments). Deliverable: Preliminary results report with metrics. Risk: Poor convergence—fallback to Genetic Algorithms if needed.
- **Days 9-10 (Jan 6-7: Finalization)**: Optimize parameters, refine the system, compile final report and code documentation. Deliverable: Complete optimized system and full project report. Risk: Time pressure—prioritize core features.

Overall Timeline: 10 days, with daily stand-ups. Tools: GitHub for code versioning, Trello for task tracking. Justification: Agile sprints ensure rapid progress, aligning with metaheuristic development cycles for iterative testing and refinement.

9 Conclusion

This report formalizes the UTP with FSTM data, advancing from conceptual to implementable. Challenges include data estimation; next: Prototype algorithms.