# Submission to ICT: Track 2

Hadrien Cambazard, Emmanuel Hebrard, Barry O'Sullivan, Alexandre Papadopoulos

Cork Constraint Computation Centre

**Abstract.** Our approach is a local search based on the principles used in [1–3]. We intend to hybridize it with a Constraint Programming (CP) approach in a Large Neighborhood Search scheme to address the hardness of finding feasible solutions. The present submission only includes our local search algorithm.

*Local search.* The search for feasible solutions is performed in a local search manner by considering a unit cost per hard constraint violation: infeasible timeslot for an event, infeasible room for an event, two events sharing a student in the same timeslot, two events violating a precedence between them. The neighbourhood structure used is the following :

1. $T$ : translate an event to a free position in the timetable.
2. $S$ : swap two events.
3. $ST$ : swap two timeslots.
4. $M$ (Matching) : Reassign the events within this same timeslot to minimize the number of room conflicts. Knowing whether a set of events can fit in a timeslot regarding the room capacities is a matching problem. By allowing violation of those contraints, we end up solving a maximum matching problem.
5. $T + M$ : translate an event to a timeslot and evaluate the best way to insert it into the corresponding timeslot by solving the previous matching problem.
6. $H$ (Hungarian) : pick a set of event assigned in different timeslots (at most 45 events) that share no precedences and reassign them optimally solving an assignment problem with the Hungarian algorithm. The violation of the hard constraints for placing each event in each timeslot is known as it does not depend on the other removed events (because they don't share precedences and only a single event is removed per timeslot). A matching problem is solved to evaluate the cost regarding room capacities of placing an event in a given timeslot. As the number of such moves is exponential, the size of the neighborhood is restricted and $k$ sets ($k = 20$ in practice) are built to include conflicting events and completed randomly.

The moves are ranked regarding their time complexities and a move is included in the neighborhood at a given iteration depending on a probability decreasing with its complexity. Time consuming moves are therefore less performed than fast ones. Side-walk moves are always accepted and no emphasis is put on conflicting events except by move $H$. Finally a simple tabu mechanism prevents cycling and a pure random configuration is used to start.

Once a feasible solution has been found, a simulated annealing (SA) is used for optimizing the soft cost and its neighborhood is defined by the single move $T+M$ maintaining the soft cost. Only the moves preserving feasibility are chosen. The initial temperature is chosen dynamically as the average of the variation of the objective function when running the SA at a temperature of 1. A standard geometric cooling takes place for the remaining time.

The problem defined by hard constraints only can be seen as a constrained list coloring where the graph is made of many large cliques (a node being an event). Such cliques come from the choices of student but can often be larger by including the incompatibilities between events due to rooms. This intensification step tries to take advantage of the presence of such large cliques by iteratively applying move $H$ on each clique containing at least one conflicting event. All events of the clique have indeed to be in different timeslots and define an assignment problem in the current timetable.

*Hybridation with Constraint Programming.* We developed various Constraint Programming (CP) models for the problem that are not included in the current submission because they were not competitive with our local search baseline. Our best CP approach remained unable to find a feasible solution to instances 1,2,9,10 However, we intend to use CP as a move of the local search procedure in a Large Neighborhood Search (LNS) approach. We believe that this approach should give much more flexibility to the optimization phase by allowing to relax the feasibility on a subset of the problem. The SA is indeed severely limited by trying to keep feasibility while moving.

## References


1. Marco Chiarandini, Mauro Birattari, Krzysztof Socha, and Olivia Rossi-Doria. An effective hybrid algorithm for university course timetabling. *J. Scheduling*, 9(5):403–432, 2006.
2. Philipp Kostuch. The university course timetabling problem with a three-phase approach. In *PATAT*, pages 109–125, 2004.
3. Olivia Rossi-Doria, Michael Sampels, Mauro Birattari, Marco Chiarandini, Marco Dorigo, Luca Maria Gambardella, Joshua D. Knowles, Max Manfrin, Monaldo Mastrolilli, Ben Paechter, Luis Paquete, and Thomas Stützle. A comparison of the performance of different metaheuristics on the timetabling problem. In *PATAT*, pages 329–354, 2002.