Local Search Techniques for a Medical Student Scheduling Problem

Eugenia Zanazzo[1], Sara Ceschia[1][0000−0003−1191−1929], Agostino Dovier[2][0000−0003−2052−8593], and Andrea Schaerf[1][0000−0001−6965−0536]

[1] DPIA, University of Udine, via delle Scienze 206, 33100 Udine, Italy,
{eugenia.zanazzo,sara.ceschia,andrea.schaerf}@uniud.it
[2] DMIF, University of Udine, via delle Scienze 206, 33100 Udine, Italy,
agostino.dovier@uniud.it

Keywords: Medical Student Scheduling · Local Search · Simulated Annealing

1 Introduction

We consider the Medical Student Scheduling (MSS) problem in the formulation proposed by Akbarzadeh and Maenhout [2], which is a simplified version of the general problem previously proposed by the same authors [1].

In the MSS problem, medical students have to be assigned in subsequent periods to a set of wards in designated hospitals, in order to complete their training by performing internships on the disciplines carried out in the specific wards.

This version of the problem takes into account, among other constraints and objectives, precedences among disciplines, student preferences, waiting periods, and hospital changes. The typical horizon considered is one year, split into either 12 periods of one month or 24 periods of two weeks.

The objective of the problem is to design a timetable that maximizes both students' desire and fairness among students, satisfying rules, regulations, and requirements for the medical school and the hosting hospitals.

We developed a local search technique for the MSS problem, based on a combination of two different neighborhood relations and guided by a Simulated Annealing procedure.

We also implemented an instance generator that was used to create challenging instances with up to 320 students. According to Akbarzadeh and Maenhout [1], such number of students represents a realistic size, though much larger than the ones in the original dataset (max 80 students). In addition, the generated instances also activate the constraint on the minimum number of students in a ward, which is included in the model but always set to 0 in the original instances. This constraint makes instances harder to be solved.

As customary, the generated instances are split into two sets, one used for the parameter tuning and the other one for the validation.

Our solution method has been able to find consistently the optimal solution value for all instances of the dataset proposed by Akbarzadeh and Maenhout [2], in much shorter runtime, though without any optimality guarantee, than their exact technique.

2 Solution method

For brevity, we do not report here the precise MSS formulation, which can be found in the original work [2].

Our local search technique uses for the search space an integer-valued matrix that assigns to each student in each period a specific ward of a specific hospital. That is, we use a single value that encodes a pair $\langle$hospital, ward$\rangle$. Since in the original data only one given discipline can be undertaken in any specific ward, this value specifies also the discipline. The conventional value -1 is used when the student is not assigned to any internship in the specific period.

We decided to include in our search space also solutions that may violate two hard constraints, namely the minimum and maximum number of students per ward per period and the precedences among disciplines. These constraints are taken care of by the cost function along with the soft constraints. As customary, the hard constraint violations are assigned a higher weight, in order to favor feasibility over optimality.

We consider the following two atomic neighborhood relations:

- Change (C). The move $\mathsf{C}\langle s, p, w, p', w'\rangle$ reassigns the student s from the period p at ward w to a new period p' and a new ward w'. The move has the precondition that s is currently idle in p', unless $p = p'$; in the latter case the move represents a reassignment of the ward in the current period p. It is also possible that $w = w'$, so that the student remains in the same ward, but at different time. It is not possible that $p = p'$ and $w = w'$, which would result in a null move.
- Swap (S). The move $\mathsf{S}\langle s, p, w, p', w'\rangle$ swaps the assigned wards w and w' of student s in the two distinct periods p and p'. The precondition here is that the student is assigned in both periods, i.e., $w \neq -1$ and $w' \neq -1$.

As guiding metaheuristic, we use Simulated Annealing, which already turned out quite effective in a number of timetabling problems (see, e.g., [6, 4, 3]). The neighborhood relation employed is the set union of Change and Swap, and the random move selection is guided by a parameter ρ_S (called *swap rate*), such that a Swap move is drawn with probability ρ_S and a Change move with probability $1 - \rho_S$.

3 Experimental results

The tuning procedure was performed on training instances of various sizes, created by our generator specifically for this purpose. We used the tool JSON2RUN [7], which performs the F-Race procedure [5] for selecting the best configuration. The winning configuration has swap rate ρ_S equal to 0.19.

Table 1. Comparative results between different solution methods and time limits.

	CPU (s)	Gap (%)	Opt (%)
MIP	8054	9	50
CP	2630	1	88
DP	166	0	100
SA-5	2.5	0.09	98.1
SA-10	4.9	0.07	99.3
SA-20	9.7	0.07	99.8
SA-50	24.1	0.06	99.9
SA-100	48.2	0	100

Table 1 shows the comparison between the methods presented by Akbarzadeh and Maenhout [2, Table 8], i.e. a Mixed Integer Programming (MIP) formulation, a Constraint Programming (CP) formulation and a Dynamic Programming (DP) method, and our method based on Simulated Annealing with increasing number of iterations ($10^6 k$ with $k \in [5, 10, 20, 50, 100]$), denoted by SA-$k$. The table reports for each method the average results obtained on all original instances in terms of computational time in seconds (CPU), final optimality gap (Gap) and percentage of instances solved to optimality (Opt) within the time limit[3]. These results show that we can obtain the optimal value already in 2.5 seconds on average, with a confidence of 98.1%. With 9.7 seconds we reach the near certainty given by the confidence of 99.8%.

The project is still ongoing, and the current work regards the experimentation on larger and more challenging instances, the development of exact methods, and the design of hybrid approaches.

Acknowledgements We thank Babak Akbarzadeh for answering all our questions about their work.

References

1. Akbarzadeh, B., Maenhout, B.: A decomposition-based heuristic procedure for the medical student scheduling problem. European Journal of Operational Research **288**(1), 63–79 (2021)
2. Akbarzadeh, B., Maenhout, B.: An exact branch-and-price approach for the medical student scheduling problem. Computers and Operations Research **129**, 105209 (2021)
3. Bellio, R., Ceschia, S., Di Gaspero, L., Schaerf, A.: Two-stage multi-neighborhood simulated annealing for uncapacitated examination timetabling. Computers and Operations Research **132**, 105300 (2021)

[3] The methods by Akbarzadeh and Maenhout [2] are implemented in the ILOG-OPL IBM environment and the time limit imposed is 5760 secs for instances with 40 students and 11520 secs for those with 80 students.

4. Bellio, R., Ceschia, S., Di Gaspero, L., Schaerf, A., Urli, T.: Feature-based tuning of simulated annealing applied to the curriculum-based course timetabling problem. Computers and Operations Research **65**, 83–92 (2016)
5. Birattari, M., Yuan, Z., Balaprakash, P., Stützle, T.: F-race and iterated F-race: An overview. In: Experimental methods for the analysis of optimization algorithms, pp. 311–336. Springer, Berlin (2010)
6. Ceschia, S., Di Gaspero, L., Schaerf, A.: Design, engineering, and experimental analysis of a simulated annealing approach to the post-enrolment course timetabling problem. Computers and Operations Research **39**, 1615–1624 (2012)
7. Urli, T.: json2run: a tool for experiment design & analysis. CoRR **abs/1305.1112** (2013)