ITC 2019: Results Using the UniTime Solver

Tomáš Müller

Purdue University, West Lafayette, Indiana, USA
muller@unitime.org

Abstract. This abstract presents results on using the Uni-Time solver on the International Timetabling Competition 2019 late data sets. The results are compared with the best solutions that are published on the competition website.

Keywords: University course timetabling · ITC 2019 · UniTime

1 Introduction

Building on the success of the earlier timetabling competitions, the International Timetabling Competition 2019 (http://www.itc2019.org) is aimed to motivate further research on complex university course timetabling problems coming from practice. The competition data sets are based on real-world problems that have been collected using the UniTime application [11]. The individual timetabling problems are quite large with the largest problem having close to 9,000 classes and over 38,000 students. The data for the competition have been collected from 10 institutions around the world and there are a lot of differences between them. For example, some instances have no students (classes are spread in time using hundreds of non-overlapping constraints), some instances are based on student pre-registrations (aka post-enrollment course timetabling) and some instances are based on curricular data. There have been three sets of 10 instances published during the competition: early, middle and late.

The competition problem combines student sectioning together with standard time and room assignment of individual course events [8]. Classes are organized in a course structure defining the valid combinations of classes a student can take. For example, each student taking a Mathematics course needs to attend a lecture and a lab that is associated with the lecture. The problem also deals with travel times between individual rooms, classes that have different lengths and multiple meetings on a week, classes that are meeting only during certain weeks, and various additional distribution constraints, such as minimizing gaps between classes of an instructor or defining how many class hours an instructor can teach on a day.

UniTime [12] is a comprehensive educational scheduling system that supports developing course and exam timetables, managing changes to these timetables, sharing rooms with other events, and scheduling students to individual classes. It is a distributed system that allows multiple university and departmental schedule managers to coordinate efforts to build and modify a schedule that meets their diverse organizational needs while allowing for minimization of student course conflicts. The software is distributed free under an open-source license and the UniTime project is a part of the Apereo Foundation, a non-profit organization whose mission is to develop and sustain open-source software for higher education.

As the UniTime course timetabling problem is quite complex, with many additional aspects, some simplifications have been made in the competition problem as well as on the problems collected from UniTime. The aim was to reduce the modeling complexity without losing any of the hardness (or computational complexity) of the problems. For example, in UniTime, it is possible for a class to need two or more rooms, or in certain cases, for multiple classes to share a room. Also, some distribution constraints have been removed or reformulated in the competition problem. For example, instead of having a back-to-back constraint, the competition problem requires such classes to be placed in the same room, on the same day, and with limited time between the first and the last class. This makes for the same outcome when the constraint is satisfied, but the penalization of a partially violated soft constraint is a bit different. More details are discussed in [8].

The paper is organized as follows: in the next chapter, the competition solver is described. There is a short description of the UniTime solver and the code written to make the solver work on the competition problem. Results are presented in the following chapter and conclusions are presented at the end of the paper.

2 The Solver

In this work, the UniTime course timetabling solver is used as it is, even using the default configuration that ships with the UniTime application. New code has been only needed to load the competition problem into the UniTime solver and to save the solution in the competition format. Other than that, some of the penalizations of violated soft distribution constraints have been changed to follow the competition problem. The code is open-source (under the Apache license) and available in GitHub [6].

The UniTime solver is based on an iterative forward search (IFS) algorithm [11]. This algorithm is similar to local search methods; however, in contrast to classical local search techniques, it operates over feasible, though not necessarily complete, solutions. In these solutions, some classes may be left unassigned. All hard constraints on assigned classes must be satisfied. Such solutions are easier to visualize and more meaningful to human users than complete but infeasible solutions. Because of the iterative character of the algorithm, the solver can also easily start, stop, or continue from any feasible timetable, either complete or incomplete.

The algorithm makes use of Conflict-based Statistics (CBS) [9] to prevent itself from cycling. The IFS algorithm is used until a complete timetable is found. In the next phase, a local optimum is found using a Hill Climbing (HC) algorithm. Once a solution can no longer be improved using this method, the Great Deluge (GD) technique [1] is used. The GD algorithm is altered so that it allows some oscillations of the bound that is imposed on the overall solution value [7].

The solver splits the problem into two sub-problems: student sectioning and class assignment. In the beginning, students are assigned to individual classes following their course demands and course structure. Students with similar courses are kept together as much as possible, using a simple construction heuristics while sectioning one course at a time. This allows for the computation of potential student conflicts between individual classes, that is, the numbers of students assigned to pairs of classes that are overlapping in time or are one after the other in rooms that are too far apart. During the solver run, classes are assigned in times and rooms while the number of student conflicts is minimized, together with the other penalizations on assigned times, rooms, and violated soft distribution constraints. When the class assignment solver is finished, a local-search technique is used to move students between alternative classes or to swap two students between such classes. During the class assignment, student conflicts between two classes that have some alternatives are weighted less (0.2 of the weight defined in the problem) than the conflicts between classes with no alternatives (i.e., conflicts that cannot be removed by re-sectioning).

More details about the UniTime solver, including various improvements that have been done over the years, are presented in [7].

3 Results

The best and the average penalty from 10 independent runs are presented in the following table. The results were computed using a 2021 model of MacBook Pro with an Apple M1 Max processor, 64 GB memory, OS X 12.3 and Java 8. The solver uses only one CPU core, and the time limit was restricted to two hours. To make use of multiple processor cores, 8 independent runs were done in parallel. UniTime solver cpsolver-1.3.189 was used in the experiment. All the runs were done with the same parameters (using the UniTime's default solver configuration), without any parameter tuning or consideration of a particular instance. The results are compared with the best solutions available at the time of the experiment.

Table 1 shows the results from the experiment compared with the best solutions uploaded at the competition website as of June 27, 2022. The first two columns (named *UniTime*) show the results of this experiment. For each of the late instances, the penalty from the best solution of the 10 independent runs and the average penalty from all the 10 runs is listed respectively. These results are compared with the best solutions from the five competition finalists (columns *Holm* [4], *Rappos* [10], *Gashi* [3], *Er-rhaimini* [2], and *Lemos* [5]), which together with the solver of the author of this paper (column *Müller*) are the six best solvers available at the time of the writing.

Table 1. UniTime solver results compared with the best results on the late instances.

Late	UniTime		Best Result at ITC2019.org					
Instance	2h Best	Average	Holm	Rappos	Gashi	Er-rhaimini	Lemos	Müller
agh-fal17	130 635	133 754.9	140 194		184 030	153 236	142 687	**117 627**
bet-spr18	352 249	353 373.5	**348 524**	360 057	360 437	373 039	353 920	348 536
iku-spr18	40 765	43 082.0	**25 863**	36 711	85 969	70 932	45 537	35 783
lums-fal17	398	411.1	**349**	386	486	558	813	368
mary-fal18	4 924	5 101.4	**4 331**	5 637	7 199	6 944	44 097	4 805
muni-fi-fal17	3 506	3 789.5	**2 837**	3 794	4 712	4 820	4 161	3 180
muni-fspsx-fal17	12 455	15 639.9	12 390	33 001	41 933	104 625	101 317	**10 058**
muni-pdfx-fal17	117 382	125 200.6	**82 258**	151 464	159 203	191 887	151 461	97 449
pu-d9-fal19	46 067	47 441.5	**39 081**	134 009	82 757	70 450	47 543	44 603
tg-spr18	16 140	20 418.2	**12 704**	12 856	15 992	19 738	31 900	14 548

The best know solution of each instance is marked in **bold**. Solutions of the finalists that were improved after the competition has ended are underlined. This means that in these cases a better solution was uploaded on the ITC 2019 website after the competition.

Within the short period of time, the solver was consistently able to produce a solution that is better than the second best solver from the finalists in seven cases. This is indicated by the table colors. The second best results from the five finalists is indicated by blue color. All UniTime results from this experiment that are better than this result are marked with violet color. The remaining three instances (iku-spr18, lums-fal17, and tg-spr18) are the only three late instances that do not have any students.

Better results can be achieved with longer run times and some parameter tuning, which has only been done to some extent. These include some additional improvements, e.g., allowing students to be re-sectioned continuously during the search or removing some of the complexity of the solver (that is not needed for the competition). The best results achieved are listed in the last column (named *Müller*). With these changes, the UniTime solver has produced the best know solution for two late instances (agh-fal17 and muni-fspsx-fal17), and it was able to produce second best results for all but one instance (tg-spr18).

4 Conclusion

The presented solver did not compete in the competition as the author of this abstract is the technical lead and principal developer of the UniTime system and a co-organizer of the competition. Nonetheless, the presented results can provide a good reference of how the UniTime solver would do on the competition problems.

While the UniTime system benefits of almost two decades of research and development, it is good to see that the competitors are able to produce results that are in par or better than what UniTime would produce out of the box using a reasonable runtime.

References

1. Dueck, G.: New optimization heuristics: The great deluge algorithm and the record-to record travel. Journal of Computational Physics **104**, 86–92 (1993)
2. Er-rhaimini, K.: Forest growth optimization for solving timetabling problems. In: ITC 2019: International Timetabling Competition (2020)
3. Gashi, E., Sylejmani, K., Ymeri, A.: Simulated annealing with penalization for university course timetabling. In: Proceedings of the 13th International Conference on the Practice and Theory of Automated Timetabling - PATAT 2021. vol. 2, pp. 361–366 (2021)
4. Holm, D.S., Mikkelsen, R.Ø., Sørensen, M., Stidsen, T.J.R.: A graph-based MIP formulation of the international timetabling competition 2019. Journal of Scheduling (2022), published: 11 March 2022
5. Lemos, A., Monteiro, P.T., Lynce, I.: ITC 2019: University course timetabling with MaxSAT. In: Proceedings of the 13th International Conference on the Practice and Theory of Automated Timetabling - PATAT 2021. vol. 1, pp. 105–128 (2020)
6. Müller, T.: UniTime ITC 2019 solver source codes, `https://github.com/tomas-muller/cpsolver-itc2019`
7. Müller, T.: University course timetabling: Solver evolution. In: Practice and Theory of Automated Timetabling 2016 Proceedings. pp. 263—-282 (2016)
8. Müller, T., Rudová, H., Müllerová, Z.: University course timetabling and international timetabling competition 2019. In: Practice and Theory of Automated Timetabling 2018 Proceedings. pp. 5–31 (2018)
9. Müller, T., Barták, R., Rudová, H.: Conflict-based statistics. In: EU/ME Workshop on Design and Evaluation of Advanced Hybrid Meta-Heuristics (2004)
10. Rappos, E., Thiémard, E., Robert, S., Hêche, J.F.: A mixed-integer programming approach for solving university course timetabling problems. Journal of Scheduling (2022), published: 15 February 2022
11. Rudová, H., Müller, T., Murray, K.: Complex university course timetabling. Journal of Scheduling **14**(2), 187–207 (2011)
12. UniTime: University timetabling – Comprehensive academic scheduling solutions, `https://www.unitime.org`