Modeling Curriculum-Based Course Timetabling of University of Potsdam (Extended Abstract)

Sebastian Schellhorn

University of Potsdam, Germany
sebastian.schellhorn@uni-potsdam.de

Education is a fundamental part of universities, usually addressed by offering a variety of courses may containing lectures, exercises, seminars, projects, etc. within a defined study program. In order to obtain optimal timetables, all offered courses must be assigned to a time slot and room, respecting some constraints. A time slot is a combination of a day and a time. The corresponding problem is called Curriculum-Based Course Timetabling (CB-CTT; [3]) and considers possible constraints when looking for an assignment of courses to time slots and rooms. A solution to a CB-CTT problem is an assignment of courses to time slots and rooms, such that each corresponding hard constraint is satisfied. An optimal solution is a solution with minimal penalty regarding soft constraints. Thanks to its simple modeling language and high performance solving capabilities, Answer Set Programming (ASP; [2]) is a well suited approach to model CB-CTT problems, as [1] showed.

In our work, we focus on the real world application of the University of Potsdam and its corresponding needs. We developed a prototype web interface for lecturers and planers to collect courses and their preferences regarding time slots, rooms, equipment, etc. The collected data from the web interface is represented by ASP facts. An underlying ASP encoding and solver is used to obtain an optimal timetable, which is visualized in the web interface.

In the case of the University of Potsdam, a course may consists of several course components of possibly different types, e.g. a lecture and an exercise. We identify a course component by a tuple of a name and its type. For simplicity, we consider course components only and refer to them as courses in the following. A cohort of students is given by a corresponding program and semester wrt the curriculum. We modeled the following hard (H1-8) and soft (S1-11) constraints.

H1. Courses: One preference for each course must be assigned to a time slot and room.

H2. Cohort: Obligatory lectures of the same cohort must be all scheduled in different time slots.

H3. RoomOccupancy: Two courses cannot take place in the same room and time slot, except explicitly stated.

H4. Lecturer: Courses sharing a lecturer cannot be scheduled at the same time slot, except explicitly stated.

H5. NotParallel: Avoiding two courses to take place at the same time slot, whenever explicitly stated.

H6. Parallel: Two courses have to take place at the same time slot, whenever explicitly stated.

H7. Simultaneous: Two courses have to take place at the same time slot and room, whenever explicitly stated.

H8. Consecutive: Two courses have to be scheduled in consecutive time slots, whenever explicitly stated.

S1. RoomCapacity: For each course, penalty points for the number of students that are expect to attend the course minus the number of seats in the corresponding room are imposed on each violation.

S2. Cohort: Courses addressed to the same cohort should be scheduled in different time slots, except explicitly stated to be in parallel.

S3. Gaps: For a cohort as well as lecturer, corresponding courses should be scheduled in time slots as close as possible. The penalty regarding two courses sharing a cohort or lecturer, and a day is given by subtracting the earlier time from the later time.

S4. RoomStability: Two courses stated to be consecutive should be given in the same room. The penalty points, reflecting each violation.

S5. MaxLoad: For a cohort as well as lecturer the number of corresponding courses per day should be below or equal to a given maximum. The penalty points reflecting the number of courses above the maximum.

S6. TravelTime: For a cohort as well as a lecturer, traveling time between rooms with two adjacent courses should be as small as possible. The penalty is reflected by the traveling time itself.

S7-11. RoomSuitability: Some courses prefer particular equipment like a large board, projector, computer, microphone, or camera. Each violation counts as a penalty point.

Note that some constraints are similar to those presented in [1]. The system of [1], reads instances of a standard input format [4], translates them into ASP facts and assigns potentially any course to any time slot. In contrast, we used a direct modeling dedicated to constraints and preferences of the University of Potsdam, which partially cannot be covered by the standard input format. As a design decision, we reduced the search space to collected preferences only, instead of checking for all possible time slots and rooms for each course.

For winter 2023/24, a plan was found for about 160 courses aimed at big lecture halls of the Faculty of Science, all 90 courses of the Institute of Mathematics and 100 courses of the Institute of Computer Science, involving H1, H3, H5, S1 and S7. *clingo* was integrated to the web interface and needed about 170 seconds to solve the problem regarding the faculty. Finding plans for the institutes took less than a second, respectively.

For a timetable for summer 2024, we plan to involve all hard and soft constraints mentioned above. Future work will be to define and implement interfaces to automatically access courses offered in the past as well as rooms, lecturers etc. Furthermore, we plan to automatically integrate resulting solutions in the existing system of the University of Potsdam.

References

1. Banbara, M., Inoue, K., Kaufmann, B., Okimoto, T., Schaub, T., Soh, T., Tamura, N., Wanko, P.: teaspoon: Solving the curriculum-based course timetabling problems with answer set programming. Annals of Operations Research 275(1), 3–37 (2019)
2. Baral, C.: Knowledge Representation, Reasoning and Declarative Problem Solving. Cambridge University Press (2003)
3. Bettinelli, A., Cacchiani, V., Roberti, R., Toth, P.: An overview of curriculum-based course timetabling. TOP 23(2), 313–349 (2015)
4. Bonutti, A., De Cesco, F., Di Gaspero, L., Schaerf, A.: Benchmarking curriculum-based course timetabling: Formulations, data formats, instances, validation, visualization, and results. Annals of Operations Research 194(1), 59–70 (2012)