Student Scheduling at Purdue University

Tomáš Müller

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

Abstract. This system demonstration presents the open-source system UniTime[1]. We will concentrate on the student scheduling process at Purdue University, which combines advising, pre-registration, and batch student scheduling, followed by open registration and wait-listing.

1 Introduction

Student scheduling, sometimes called *student sectioning*, involves assigning students to classes (course sections) based on their individual course demands. There are many hard constraints and optimization criteria, including course structure, class times and limits, reservations, student and course priorities, student preferences, course alternatives, travel times, and free-time requests. The student scheduling problem in UniTime was first presented in [2] as a proof of concept. It has gone a long way since then and was successfully turned into a software solution capable of scheduling tens of thousands of students, with a scalable web-based user interface, additional constraints, and optimization objectives, and interfacing with many other systems at the university.

Purdue University is the main contributor to the UniTime project, and it is used there for various educational timetabling and scheduling needs, including course timetabling [4], examination timetabling [1], instructor scheduling, and student scheduling. The student scheduling process starts with batch student scheduling available to all undergraduate students, followed by open registration. Graduate and professional students only use the open registration. During the batch scheduling process, a student is advised by their advisor, who fills in course recommendations, and the student submits their course requests. These contain courses the student wants to take, including alternatives and substitutes, preferences, and free-time requests. Based on these and the course timetable, the UniTime solver is run at the end of the pre-registration period, and all students are provided with an initial class schedule. Afterward, during the open registration, the students can go into the system and manually change their schedules, add and drop courses, or waitlist for courses or classes that are full.

The main campus at Purdue University has over 50,000 students. During the Spring 2024 registration, there were over 8,400 timetabled classes from 3,000 courses. Over 35,000 (out of roughly 40,000) undergraduate students filled in

[1] https://www.unitime.org

their pre-registration, which contained over 180,000 course requests, and were provided their initial class schedules.

The collected student pre-registration data can also be used during the course timetabling process, making sure that the created timetable will allow students to get the courses they need. See the ITC 2019 competition[2] for more details [3], as the competition problems were collected using UniTime, and the problem combines both the timetabling aspects (assignment of classes to times and rooms) with student scheduling (assigning students to classes). However, there are fewer student scheduling constraints during the course timetabling process as the solver only minimizes potential student conflicts as one of the optimization criteria, possibly only for a subset of the classes offered at the university at a time (at Purdue, each department builds its own course timetable on top of the centrally-timetabled large lecture room classes).

2 Student Scheduling Problem

The student scheduling problem consists of courses that have already been timetabled, students and their individual course demands, and producing a class schedule for each student. It is modeled as an assignment of student course requests with enrollments, i.e., valid combinations of classes that the student needs to take to enroll into a course. There are various hard constraints, such as students not having a time conflict (unless allowed, in which case the overlapping time is to be minimized), classes and courses being limited in size, or restrictions limiting who can attend a particular class or course. It is also an optimization problem, combining a long list of various criteria, such as maximizing the number of courses each student gets, considering student and course priorities, student preferences, penalizing alternatives and substitutes, minimizing distance conflicts or travel times, etc. In the rest of this section, the most interesting aspects of the student scheduling problem are discussed.

Course Structure Each course may be offered in multiple configurations, such as face-to-face or online, each with multiple components (subparts), such as a lecture and a lab. Each subpart may contain multiple alternative classes. A student requesting a course will get one class of each subpart of a single configuration. There can be additional parent-child relations between individual classes, which also must be followed. So, for example, a student may get Lec 1 - Lab 1, Lec 1 - Lab 2, Lec 2 - Lab 3, or Lec 2 - Lab 4 class combination if attending the course face-to-face, or Dist 1 when attending the course online. Each class can also have a limit, allowing only a certain number of students to get in.

Reservations and Restrictions Access to courses or certain components of courses, such as configurations or individual classes, may be restricted with reservations. A reservation reserves a certain number of seats in the course, or some of its components, for a particular group of students, e.g., identified by their study

[2] https://www.itc2019.org

program. A restriction does not reserve any space, but the students must follow it. For example, a student of an online program may be restricted to the online course configuration, while other students may freely choose between the two configurations, assuming space is available. Similarly, there can be 100 spaces in a course reserved for students of a Computer Science major, or the space in one of the Labs may be reserved for a particular cohort.

Alternative and Substitute Courses A student may provide alternatives to each course. The aim is to get a given number of courses, and one or more alternative courses can be provided for each course. It is also possible to provide substitute courses that can act as alternatives to any other course requests except those that have been marked as no-sub by the advisors (typically those that the student must take). The order in which the courses are requested is also important, as it helps us to break the ties, e.g., when it is not possible to get both courses because they are overlapping in time, or when there are more students requesting the course than the space available.

Student Preferences and Requirements A student can indicate which course configurations and/or classes are preferred for each course. It is also possible to provide free time requirements, which can act as unavailabilities (i.e., a student cannot get a class at the time) or preferences (i.e., an overlapping time with the free time should be minimized), depending on the position of the free time among the courses.

Students may also indicate whether they prefer online or face-to-face classes and whether back-to-backs are preferred or discouraged. For the Summer terms, which are organized into three four-week modules, it is also possible to indicate during which modules the student can have classes and whether they can attend classes on campus.

Student and Course Request Priorities As Purdue is constantly growing its student population, there can be many courses with more students requesting them than space available. A number of priorities have been added to help students graduate on time. First of all, advisors may indicate courses as **vital** to the student, which means that the student absolutely needs the course (or one of the provided alternatives) to progress towards their degree. Moreover, students are divided into priority students (such as athletes and students in university bands and orchestra), students near graduation (with 100 or more credits earned), senior students (60 or more credits), and the rest. Vital course requests take priority over student priorities.

3 Student Scheduling Solver

Student scheduling uses the same hybrid solver as other UniTime modules, combining constraint-programming primitives with local-search-based algorithms and various heuristics. While the solver can work with incomplete solutions, it does not allow the breaking of hard constraints. The search consists of various phases. During the first phase, a schedule for each student is constructed,

with vital course requests assigned first and students taken in the order of their priorities. A branch-and-bound algorithm is used for each student to find the best possible schedule using the remaining available space. After the first phase, various other heuristics and neighborhoods are employed, e.g., looking for a swap between two students. Or assigning a student to a course while some other student (or students) is bumped out, e.g., due to the class limits. A branch-and-bound with a limited depth, stochastic hill climbing, and great deluge is used at various search phases. The solver uses multiple CPU cores, benefiting from the fact that individual student schedules do not interact with each other that much, typically only when the last spot in a course, a class, or a reservation is involved.

The same constraint-based model, optimization criteria, and some algorithms are also used in the other problems of student scheduling in UniTime. For example, when a student enrolls in a new course or courses during the open registration period, the same branch-and-bound algorithm creates the best possible class schedule, given the current availability and the student's existing schedule. Or providing suggestions when moving classes around in an existing class schedule.

The student scheduling solver is also used when there is a course timetabling change. For example, UniTime will automatically move students enrolled in a canceled class to other classes of the course or put them on a waitlist when there are no other possible enrollments into the course that are both available to the student and do not conflict with the rest of their schedule.

4 System Demonstration

Student scheduling is an important hard optimization problem of a huge size. In this abstract, a number of aspects that were required to bridge the gap between theory and practice were outlined. The demonstration will present the UniTime system and its user interface, covering the whole student scheduling process at Purdue, step by step. It starts with student advising and pre-registration. It follows with the batch student scheduling, but the solver is also used to provide nightly test runs, which are used to monitor pre-registration progress and to catch any potential issues early. The demonstration will conclude with open registration, showcasing a student making a schedule change, swapping a course, and/or wait-listing for another course or class that is currently full.

References

1. Müller, T.: Real-life examination timetabling. Journal of Scheduling **19**, 257–270 (2016). https://doi.org/10.1007/s10479-014-1643-1
2. Müller, T., Murray, K.: Comprehensive approach to student sectioning. Annals of Operations Research **181**, 249–269 (2010)
3. Müller, T., Rudová, H., Müllerová, Z.: Real-world university course timetabling at the International Timetabling Competition 2019. Journal of Scheduling (To Appear)
4. Rudová, H., Müller, T., Murray, K.: Complex university course timetabling. Journal of Scheduling **14**(2), 187–207 (2011)