Using Information on Unconstrained Student Demand To Improve University Course Schedules

Gary M. Thompson
Cornell University

Final version published in *Journal of Operations Management* (2005), 23, 197-208.

Author Note:

Gary M. Thompson is professor of Services Operations Management in the School of Hotel Administration at Cornell University.

Correspondence concerning this article should be addressed to Gary Thompson, School of Hotel Administration, Cornell University, Ithaca, NY 14853.
E-mail address: gmtl@cornell.edu.

Abstract

We examine how using information on unconstrained demand can improve operational decisions. Specifically, we examine the widespread problem of developing course schedules in not-for-profit university settings. We investigate the potential benefit of incorporating, into the scheduling process, information on the unconstrained demand of students for courses. Prior to this study, the status quo in our college, like that in a large proportion of university settings, was building the course schedule to avoid time conflicts between required courses and to minimize time conflicts between designated groups of courses, such as electives in a particular area. Compared to the status quo approach, we find that, based on three semester's worth of actual data, an approach that explicitly considers students' course preferences improves a student-based metric of schedule quality on the order of over 4% (which is the equivalent, in our setting, of improving service for over 20% of students).

Keywords: Course scheduling; Unconstrained demand; Heuristics

1. Introduction

In many not-for-profit service settings, decisions about capacity are often based on constrained, or filtered, demand information rather than on the better information that can be provided by unconstrained, or true, demand. Consider, for example, a state driver’s license bureau office and a library. In a state driver’s license bureau office (where lines of customers are often long), staffing decisions typically are made based on the number of customers that were served in the past (constrained demand), rather than the number of customers who desired service (unconstrained demand). The amount by which unconstrained demand exceeds constrained demand is determined by the number of customers who were unable to enter the system because the service was operating at capacity, plus the number of customers who decided not to enter the service system because they believed that the service was incapable of serving them in a timely manner. Moreover, a computer system that records when customers are served (such as point-of- sale systems in retail settings), and uses that for staffing purposes, will perpetuate poor service unless it has a means of tracking the number of people *waiting* for service.

In libraries, demand is constrained by the operating hours. One can track the number of patrons that show up during the hours that the facility is open, but one cannot easily track the number of patrons who would show up under different, longer operating hours.

Although it is obvious, we will state it anyway: making capacity decisions, such as staffing levels, based on constrained demand can perpetuate poor service. The challenge facing many not-for-profit service businesses, however, is actually obtaining information on unconstrained demand. The key question becomes, Is it worth paying attention to unconstrained demand? In this paper, we provide the answer to that question, at least in one not-for-profit service setting.

In the not-for-profit service settings of universities and other post-secondary educational institutions, the issue of constrained versus unconstrained demand arises in relation to course scheduling. (Certainly there are for-profit post-secondary educational institutions. However, the issues facing those institutions are not the focus of the current investigation.) Course scheduling is the determination of which courses are taught at what days and times, in which rooms, and taught by whom. The typical process followed in postsecondary educational institutions in the U.S.A. is that a schedule is developed, without regard to true demand for courses, and then students sign up for courses in the schedule. The schedule may be developed considering such things as avoiding conflict between required courses and spreading elective courses around as much as possible, while also considering the preferences of faculty members and the constraints imposed by the organization's facilities. Faculty members can also be consulted regarding which courses they believe should not conflict with the ones they teach, because of students' desired course co-enrollments. While avoiding conflict between required courses, spreading elective courses around as much as possible, and avoiding conflict between designated courses are all motivated by a desire for a student-friendly schedule, these actions do not get at the true demand of students for courses. This incongruence provided the motivation for our investigation.

We investigate the benefits that can accrue by constructing university course schedules using information on unconstrained student demand for courses. Specifically, we compare schedules developed using several approaches to student demand, including: constructing schedules while avoiding conflict only for required courses, constructing schedules while avoiding conflict for required courses and considering faculty specifications as to which courses should not conflict, and constructing schedules while avoiding conflict for required courses and considering students' course preferences. We measure schedule quality using the criterion of the extent to which students are able to take the courses for which they express preferences. We test the approaches using data collected for our college's Fall 2003, Spring 2004 and Fall 2004 course schedules.

Ensuring that students can get the courses that they desire is important for a variety of reasons. First, it can help ensure that they graduate on time. This is particularly important for transfer students, who typically take a much less regular set of courses compared to non-transfer students. Second, it can lead to higher student satisfaction, which is a factor in some rankings of programs. Third, it can lead to a better educational experience, which might translate into higher student commitment to the institution and generate such benefits as stronger word-of-mouth referrals and increased monetary giving to the institution. Fourth, if the reasons for asking for students' course preferences are well articulated and communicated to the students, then the educational institution can better manage students' impressions of the institution, which can yield benefits similar to those identified in the previous point. Fifth, there a significant amount of attention has been paid to student satisfaction (see, for example, Athiyaman, 1997; Elliott and Shin, 2003; Lau, 2003; Thomas and Galambos, 2004; Umbach and Porter, 2002). The ability for students to get more of the courses they want adds to the array of devices for increasing satisfaction.

The structure of the remainder of this paper is as follows. We begin by reviewing relevant literature. We then describe the experiment we conducted, which uses information from a real university setting. Next, we present the results of our experiment. We close with conclusions and suggestions for future research.

2. Relevant literature

A rather extensive literature exists on the problem of course scheduling, or timetabling as it is more commonly known outside the U.S.A. The basic form of the problem is finding an assignment that fills the timetable (i.e., rooms by times) with courses so that certain constraints are satisfied (Abramson, 1991; Kovacic, 1993).

A variety of methodologies have been employed to solve the course scheduling problem, including constraint programming (Valouxis and Efthymios, 2003), decision support systems (DSS) (Foulds and Johnson, 2000; Johnson, 1993), heuristics (Abramson, 1991; Alvarez-Valdes et al., 2002; Asratian and de Werra, 2002; Colorni et al., 1998; Cooper and Kingston, 1993; Costa, 1994; Dimopoulou and Miliotis, 2001; Dowsland, 1990; Ferland and Fleurent, 1994; Hinkin and Thompson, 2002; Kovacic, 1993; Sampson and Weiss, 1995; Sampson et al., 1995; Stallaert, 1997), logic programming (Kang and White, 1992), mathematical programming (Badri, 1996; Badri et al., 1998) and neural networks (Smith et al., 2003). Because of the complexity of the environment we consider, we employ a heuristic, similar to that described by Hinkin and Thompson (2002).

To date, the most variable- and constraint-rich representation of the course scheduling problem has been provided by Hinkin and Thompson (2002). Their main variables defined who teaches each course and each course's assigned meeting pattern (which incorporates the assignment of rooms, days and times). They used a number of other variable types to measure different aspects of the schedule, such as which days faculty members were teaching, and deviation variables measuring the extent to which soft constraints were violated. In Appendix A, we present our model for course scheduling, which is based on Hinkin and Thompson's (2002) model.

Although Hinkin and Thompson’s (2002) model provides a basis for our investigation, it lacks the ability to explicitly include students’ preferences for courses. In Appendix B, we describe the modifications we made to their model to do this.

The works by Sampson et al. (1995) and Sampson and Weiss (1995) were the first to address the issue of scheduling courses in a university setting based on unconstrained student demand for schedules. Sampson et al. (1995) reported on a heuristic procedure they developed to construct schedules for their school. While the scheduling process previously used at their school resulted in close to one half of students not being able to receive two or more of their requested courses, their heuristic lowered the percentage of students not receiving one or more of their requested courses to less than 5% of the total number of students. Sampson and Weiss (1995) tested a similar heuristic to that proposed by Sampson et al. (1995) using a set of randomly created test problems.

The research on which we report in this paper can best be viewed as an extension of the work of Sampson et al. (1995), Sampson and Weiss (1995), and Hinkin and Thompson (2002). We draw upon the richness of Hinkin and Thompson’s (2002) problem representation and the idea of considering unconstrained student demand first presented by Sampson et al. (1995) and Sampson and Weiss (1995). Using a combined framework, we examine the effects, on a student-based metric of schedule quality, of using information on unconstrained demand information when building the course schedule. To our knowledge, this is the first multi-semester investigation of the benefits of using unconstrained demand in any (profit, or not-for-profit) educational institution. Moreover, we are the first to evaluate, in a university scheduling context, how employees’ (i.e., faculty members’) perceptions of customer (i.e., student) demand aligns with customers’ (students’) actual demand.

3. Experimental design

In this section, we describe: the process we followed to collect unconstrained student demand information, our metric of schedule quality, and the three scheduling approaches we examined. We begin by describing the course-scheduling paradigm we employed.

3.1. Course-scheduling paradigm

For our general course-scheduling paradigm, we use the model presented in Appendix A (which, as noted earlier, is based on Hinkin and Thompson's (2002) model). Hinkin and Thompson's (2002) model (and ours presented in Appendix A) allowed one to define sets of classes that, respectively, must not and should not conflict. These constraints (constraint sets (7) and (8) in our model) are very important to our investigation. We describe our use of these constraint sets below, in reference to the three solution approaches we examined.

3.2. Collecting unconstrained student demand

To collect unconstrained student demand for courses, we polled students, via the internet, before the schedule was constructed. Each student was given a unique login and instructions to select, from the list of courses to be offered in the term, which particular courses he or she wished to take. The list of courses had a single entry per course: if a course was to be offered multiple times in the term (each offering designed as a separate lecture or section), we listed only one, without a lecture or section designator.

For the Fall 2003 semester, a total of 237 students specified course preferences within the required time period, out of a total of 646 students from whom we had requested preferences. We thus had unconstrained demand information on 36.7% of total students. Our total student count exceeds 800, but we did not poll the students who would be graduating before the fall (i.e., second-semester seniors). For the Spring 2004 schedule, we received preference information from 201 students (34.5% of those we polled); while for the Fall 2004 schedule, we received preference information from 196 students (32.7% of those we polled).

3.3. Schedule quality metrics

We used one measure of schedule quality, *StdCourse*. *StdCourse* is a count, across all students specifying preferences, of the number of students' course preferences that are satisfied. The upper bound on *StdCourse*, which is simply a count of the number of course preferences we received across all students, was: 1420 for Fall 2003; 1097 for Spring 2004; and 1223 for Fall 2004. *StdCourse* is somewhat similar to the student-based metrics used by Sampson et al. (1995) and Sampson and Weiss (1995) except that these earlier works used weighted preferences for courses.

3.4. Scheduling approaches

We use a simulated-annealing based heuristic solution procedure, which is similar to that described by Hinkin and Thompson (2002) except, as noted below, for how one of our methods accounts for student course preferences. We used three approaches for building the course schedule. They are:

- *S-Control*: This was the control case, where the schedule was constructed with avoiding (or minimizing) conflict between the required courses being the only student-related constraints. For example, our masters-level students have a core set of courses that they must take within the same term, so the schedule must be constructed to avoid any time conflicts between those required courses. These constraints are imposed by sets (7) and (8) in our model presented in Appendix A.

- *S-Faculty*: This approach used the same student- based constraints as in *S-Control*, but added two types of restrictions based on the suggestions of faculty members. The first type of restriction was to minimize conflict among courses in particular areas. For example, *S-Faculty* includes a constraint that says to minimize time conflicts among all undergraduate-marketing-elective courses and another to minimize time conflicts among all undergraduate-finance-elective courses. The rationale behind these constraints is that students who wish to take a marketing or finance elective will have a greater set of choices if the courses within each group have fewer time conflicts. The second type of additional restriction was of the form ‘‘course A should not conflict with course B.’’ For example, a faculty member who teaches a course in Services Marketing might say that his or her course should not conflict with a Service Operations Management course, since students like to take both courses in the same term.
- *S-Assign*: The *S-Assign* scheduling approach, in addition to the requirement that required courses not conflict, explicitly considers student course preferences. *S-Assign* does not use constraint sets (7) or (8) from our model in Appendix A, but instead actually develops an assignment of students to courses each time the scheduling heuristic develops a new schedule. Explicitly assigning students to courses requires that we supplement our model presented in Appendix A (based on Hinkin and Thompson’s (2002) model) with new variables and constraints. This supplemental information is provided in Appendix B.

Insert Table 1 Here

3.5. Schedule replication

We had three problem settings, since we were building our college's Fall 2003, Spring 2004 and Fall 2004 course schedules. To get a sense for the true performance of each approach, we chose to generate 10 different schedules for semester for each solution approach. For each schedule replication, we ran the scheduling heuristic for 15 min on a personal computer with dual Pentium IV Xeon 2.8 MHz processors. In the next section, we compare the performance of the solution approaches based on the quality of the schedules they generated.

Since two solution approaches do not explicitly consider student-to-course assignments (*S-Control* and *S-Faculty*), to judge their performance on the *StdCourse* performance metric, we needed some means of assigning students to courses. To do this, we used the heuristic assignment described in Appendix B. For consistency, we also used this heuristic to measure the *StdCourse* performance of the *S-Assign* solution approach.

4. Experimental results

Table 1 displays the raw results from the experiment. For Fall 2003 and Spring 2004, the best performing method was S-Assign, followed by S-Faculty. For Fall 2004, the best performing method was *S-Assign*, while *S-Faculty* was inferior to *S-Control*.

As noted earlier, if all students' course preferences were satisfied, *StdCourse* would have been 1420 for Fall 2003, 1097 for Spring 2004, and 1223 for Fall 2004. *S-Assign*'s schedules were thus, on average, able to deliver 96.2%, 98.2%, and 91.7% of the theoretical maximum preference values for Fall 2003; Spring 2004; and Fall 2004, respectively.

Table 2 summarizes the average results. Overall, *S- Assign* increased student's ability to take the courses they preferred by over 4% compared to the control schedule. *S-Faculty*, building the schedule based on what faculty believe that students want, was identical overall to the control schedules. The only statistically significant difference, of the main and interaction effects, is between *S-Assign* and *S-Control* (and *S- Faculty*). This difference is significant at the 0.0001 level.

Insert Table 2 Here

5. Discussion

5.1. Implications for course scheduling in not-for-profit education environments

There are several interesting observations to be made from the results of our experiment. First, and perhaps most troubling, is that course schedules constructed based on faculty members' ideas about which courses should not conflict (i.e., based on *S-Faculty*), were *no better* than constructing schedules with the only student- based constraint set being to avoid conflict among required courses (i.e., based on *S-Control*).

We believe there are two reasons for the identical performance. The first reason relates to the idea that conflict should be minimized among similar type courses (for example, minimizing the time conflicts between all marketing electives, or between all finance electives). In the case of our college, it appears that this approach may not work because students choose courses of different types *in the same term*, which is not controlled by only limiting time conflicts among similar courses. The second reason for *S-Faculty* performing comparably to *S-Control* relates, we believe, to its course-pair constraints. Recall that course-pair constraints are those where faculty members specify the courses that should not conflict with the courses they teach. Faculty members can become aware of these constraints when a student approaches him or her and says something like ‘‘I would like to take your course, but I cannot because course B, which is a higher priority for me, is offered at the same time.’’ It may seem counterintuitive how using such information can fail to improve the schedule. However, we believe that the schedule is no better because of the missing information on all the students who *were* able to take the courses they wanted. It is this missing information, the information on true, unconstrained student demand for courses, which enables *S-Assign* to yield superior schedules (based on the student measure, *StdCourse*) compared to *S-Faculty*. Students’ course preferences also change over time, and so there is likely to be a lag effect between when students’ preferences change and when faculty members become aware of the change.

A second interesting observation was the magnitude of improvement of course schedules when one considers unconstrained demand. The difference in the *StdCourse* performance metric for *S-Assign* versus *S- Control*, 47.1 (averaged across all three semesters), can be interpreted as slightly more than 47 students being able to enroll in one more of their preferred courses. Since an average of 211.3 students specified their course preferences across the three semesters, the schedule improvement means that 22.2%, one in every 4.5 students, would be able to enroll in one more of his or her preferred courses. Thus, in our setting, we judge it worthwhile to continue to collecting student course preference information (and using that information in the scheduling process). Improving service for close to a quarter of one’s customers is clearly significant in a practical sense.

Regarding limitations of our research, one consideration comes to mind: the fact that only 32.7-36.7% of students supplied course preference requests. While the preferences of the students who did not report course preference information may well improve the apparent effectiveness of the *S-Faculty* scheduling approach, it is hard to imagine how incorporating the missing preference information would undermine the general benefit of using the unconstrained demand information.

There are several issues arising from our investigation that may benefit from future research. First, it would be interesting to examine the relationship between students' stated course preferences and the courses for which they actually registered. Second, it would be interesting to examine how schedule quality is affected by the percentage of students who specify course preferences. Third, over the last few years, we have seen learning outcomes related to course scheduling issues, and incorporating this into the scheduling process could be fruitful. For example, weekend day courses can yield a higher learning retention rate than evening courses. Fourth, for this experiment, we judged 15 min to be a reasonable run time for the scheduling heuristics; should additional performance criteria be considered, one may wish to examine the effect of increasing the run time.

5.2. Implications for other not-for-profit services

There are two important implications for other not-for-profit services. First, these services should not depend on their employees to provide a sense of unconstrained customer demand. If our setting is typical, the information provided by employees (faculty members) was of no value (schedules were no better when based on that information). Second, the not-for-profit service organizations should collect, and use, information on unconstrained customer demand. In our environment, we were able to improve the service delivered to over 20% of customers. As we achieved that improvement for very little marginal effort, the endeavor was well worthwhile.

In this paper, we have seen that by incorporating information about unconstrained student demand for courses, we were able to obtain course schedules that better fit the needs of students. In this case, better information is definitely an enabler. We expect this to be the case in many not-for-profit service settings.

Appendix A. The mathematical model for building the base course schedule

In this appendix, we present the mathematical model that we used to build the base course schedule. This model, which is based on that presented by Hinkin and Thompson (2002), defines course meeting patterns (the set P), using specific meeting-day, start-time, meeting length and room assignment options. In our school, there are over 2000 unique patterns for courses. Our model is:

Indices:

c, c'	courses
d	days
f, f'	faculty members
p, p'	meeting patterns (incorporating start times, days offered, meeting lengths and room options)
r	rooms

Constants:

CrsEnrl_{c}	estimated enrollment of course *c*
FacMaxDays_{f}	maximum number of teaching days for faculty member *f*
FacMinDays_{f}	minimum number of teaching days for faculty member *f*
$\text{PatRmTmDif}_{pp'r}$	number of minutes separating the meeting times of meeting patterns *p* and *p'* in room *r*
RmMinBtw_{r}	required minimum number of minutes to have between courses in room *r*
RmCap_{r}	smallest seating capacity of the room assignments in meeting pattern *p*
w_i	weight on objective criterion *i*

Sets:

C	set of all courses
CrsFac_{c}	faculty members sufficiently skilled to teach course *c*
CrsFacDyPt_{cfd}	meeting patterns that are consistent with course *c* and faculty member *f* and that meet on day *d*
CrsFacPat_{cf}	meeting patterns that are consistent with course *c* and faculty member *f*
CrsPat_{c}	meeting patterns that are consistent with and allowable for course *c*. The set is selected such that a course can never be assigned to a room with fewer seats than the minimum number the course requires and so that the course is never scheduled in its unavailable time
CrsAvoidHrd_{c}	courses that definitely must not conflict with course *c*
CrsAvoidSft_{c}	courses that should not conflict with course *c*

CrsSameDy_c	courses that should meet on the same day as course *c*
CrsSimul_c	courses that should meet simultaneously with course *c*
D	days on which courses may be scheduled
F	faculty
FacCrs_f	courses that faculty member *f* is sufficiently skilled to teach
FacPat_f	meeting patterns that faculty member *f* is available to teach
P	meeting patterns for courses (incorporates start times, days offered and room assignments)
PatCnflt_p	meeting patterns that have time conflicts with meeting pattern *p*
PatTrnst_p	meeting patterns that cannot be taught by a faculty member teaching meeting pattern *p* because of excess transit time
PatAsynC_p	meeting patterns that do not meet simultaneously with meeting pattern *p*
PatDifDy_p	meeting patterns that meet on different day(s) than meeting pattern *p*
RmCrs_r	courses that can meet in room *r*

Variables:

$$x_{cfp} = \begin{cases} 1, \text{if course } c \text{ is taught by faculty member } f \text{ and assigned to meeting pattern } p \\ 0, \text{otherwise.} \end{cases}$$

$$y_{fd} = \begin{cases} 1, \text{if faculty member } f \text{ is teaching on day } d \\ 0, \text{otherwise.} \end{cases}$$

$$s_{1f}^{-} = \text{shortfall in teaching days (below minimum desired) for faculty member } f.$$

$$s_{2,c}^{-} = \text{shortage of faculty assigned to course } c\,(s_{2,c}^{-} = 1, \text{if course } c \text{ is not scheduled}).$$

$$s_{3,c}^{+} = \text{number of time conflicts between course } c \text{ and those courses with which it should not conflict.}$$

$$s_{4,c}^{+} = \text{number of courses that should be, but are not, offered on the same day(s) as course } c.$$

$$s_{5,c}^{+} = \text{number of courses that should be, but are not, offered simultaneously with course } c.$$

A.1. Faculty member constraints

There are four faculty member-based constraints in our model. The first ensures that faculty members are not double-booked and that they have sufficient time between their courses to move to different buildings within the same, or across different campuses:

$$\sum_{c \in \text{FacCrs}_f} \sum_{p \in \text{CrsFacPt}_{cf}} \sum_{c' \in \{\text{FacCrs}_f | c' \neq c\}} \sum_{p' \in \{\text{CrsFacPat}_{c'f} \cap \text{PatTrnst}_p\}} x_{cfp} \cdot x_{c'fp'} = 0 \quad \text{for all } f \in F.$$

Our model controls the number of days each faculty member spends teaching, by first measuring this number for each faculty member:

$$y_{fd} \leq \sum_{c \in \mathrm{FacCrs}_f} \sum_{p \in \mathrm{CrsFacDayPat}_{cfd}} x_{cfp} \leq M \cdot y_{fd} \quad \text{for all } f \in F, d \in D.$$

We next measure the number of days each faculty member is scheduled below his or her minimum specified number of teaching days:

$$\sum_{d \in D} y_{fd} + s_{1f}^{-} \geq \mathrm{FacMinDays}_f \quad \text{for all } f \in F,$$

and impose a hard constraint on the maximum number of teaching days for each faculty member:

$$\sum_{d \in D} y_{fd} \leq \mathrm{FacMaxDays}_f \quad \text{for all } f \in F.$$

A.2. Room constraints

Our model contains only a single room-based constraint set that guarantees that no room be double-booked and that each room has its minimum required downtime between courses:

$$\sum_{c \in \mathrm{RmCrs}_r} \sum_{f \in \mathrm{CrsFac}_c} \sum_{p \in \mathrm{CrsFacPat}_{cf}} \sum_{c' \in \{\mathrm{RmCrs}_r | c' \neq c\}} \sum_{f' \in \mathrm{CrsFac}_{c'}} \sum_{p' \in \{\mathrm{CrsFacPat}_{c'f'} | \mathrm{PatRmTimDif}_{pp'r} < \mathrm{RmMinBtw}_r\}} x_{cfp} \cdot x_{c'f'p'} = 0 \quad \text{for all } r \in R, d \in D.$$

A.3. Course constraints

The first course-based constraint in our model measures the number of unscheduled courses:

$$\sum_{f \in \mathrm{CrsFac}_c} \sum_{p \in \mathrm{CrsFacPat}_{cf}} x_{cfp} + s_{2,c}^{-} = 1 \quad \text{for all } c \in C.$$

The following constraint set guarantees that courses that cannot conflict with one another actually do not conflict:

$$\sum_{f \in \mathrm{CrsFac}_c} \sum_{p \in \mathrm{CrsFacPat}_{cf}} \sum_{c' \in \mathrm{CrsAvoidHrd}_c} \sum_{f' \in \mathrm{CrsFac}_{c'}} \sum_{p' \in \{\mathrm{CrsFacPat}_{c'f'} \cap \mathrm{PatCnflt}_p\}} x_{cfp} \cdot x_{c'f'p'} = 0 \quad \text{for all } c \in C.$$

while the following constraint set measures the extent of conflict between those courses where conflict should be avoided, if possible:

$$\sum_{f \in \mathrm{CrsFac}_c} \sum_{p \in \mathrm{CrsFacPat}_{cf}} \sum_{c' \in \mathrm{CrsAvoidSft}_c} \sum_{f' \in \mathrm{CrsFac}_{c'}} \sum_{p' \in \{\mathrm{CrsFacPat}_{c'f'} \cap \mathrm{PatCnflt}_p\}} x_{cfp} \cdot x_{c'f'p'} - s^{+}_{3,c} = 0 \quad \text{for all } c \in C.$$

For pedagogical reasons, we have some courses that we wish to have meet on the same days as other courses. The following constraint set measures violations of this constraint:

$$\sum_{f \in \mathrm{CrsFac}_c} \sum_{p \in \mathrm{CrsFacPat}_{cf}} \sum_{c' \in \mathrm{CrsSameDyt}_c} \sum_{f' \in \mathrm{CrsFac}_{c'}} \sum_{p' \in \{\mathrm{CrsFacPat}_{c'f'} \cap \mathrm{PatDifDy}_p\}} x_{cfp} \cdot x_{c'f'p'} - s^{+}_{4,c} = 0 \quad \text{for all } c \in C.$$

In a similar fashion, we also measure violations of requirements that certain courses meet at the same times as other courses:

$$\sum_{f \in \mathrm{CrsFac}_c} \sum_{p \in \mathrm{CrsFacPat}_{cf}} \sum_{c' \in \mathrm{CrsSimul}_c} \sum_{f' \in \mathrm{CrsFac}_{c'}} \sum_{p' \in \{\mathrm{CrsFacPat}_{c'f'} \cap \mathrm{PatAsync}_p\}} x_{cfp} \cdot x_{c'f'p'} - s^{+}_{5,c} = 0 \quad \text{for all } c \in C.$$

A.4. Objective

The objective function, which has five components, is:

$$\mathrm{Min}\, Z = \sum_{f \in F} (w_1 \cdot s^{-}_{1f}) + \sum_{c \in C} (w_2 \cdot s^{-}_{2,c} + w_3 \cdot s^{-}_{3,c} + w_4 \cdot s^{-}_{4,c} + w_5 \cdot s^{-}_{5,c})$$

In order of appearance in (11), the components are: minimizing the number of teaching days that faculty members are scheduled below the minimum acceptable number; minimizing the number of unscheduled courses; minimizing time conflicts between courses that should not experience time conflicts; minimizing violations of courses that should be scheduled on the same day; and minimizing cases in which courses that should be scheduled simultaneously are not.

Appendix B. The model for assigning students to courses

In this appendix, we present the model we used for assigning students to courses. We also describe the heuristic we used for solving the model. Our model is:

Indices:

c, c'	course sections
u, u'	unique courses
s	students

Constants:

CrsCapacity_c	capacity of course section *c*

Sets:

Students	students
UniqueCrs	unique courses
Sections_u	alternate offerings (sections) of unique course *u*
CrsDesired_s	unique courses desired by student *s*
StdWanting_u	students wishing to take unique course *u*
$\text{TimeConflicts}_{uc}$	unique courses (and sections) that have time conflicts with section *c* of unique course *u*

Variables:

$$y_{suc} = \begin{cases} 1, \text{ if student } s \text{ is assigned to section } c \text{ of unique course } u \\ 0, \text{ otherwise.} \end{cases}$$

Constraints:

Our model for matching students to course sections has three constraint sets. The first is that each student can be assigned to no more than one section of each course he or she desires:

$$\sum_{c \in \text{Sections}_u} y_{suc} \leq 1 \quad \text{for all } s \in \text{Students}, \quad u \in \text{CrsDesired}_s.$$

We also need to ensure that no student is double-booked at any time

$$\sum_{u'c' \in \text{TimeConflicts}_{uc}} y_{suc} + y_{su'c'} \leq 1 \quad \text{for all } s \in \text{Students}, \quad u \in \text{CrsDesired}_s, \quad c \in \text{Sections}_u.$$

Finally, we need to ensure that no more students are assigned to a section than its capacity allows:

$$\sum_{s \in \text{StdWanting}_u} y_{suc} \leq \text{Capacity}_{uc} \quad \text{for all } u \in \text{UniqueCrs}, \quad c \in \text{Sections}_u.$$

The objective is to maximize the number of courses assigned to students:

$$\text{Maximize } Z = \sum_{s \in \text{Students}} \sum_{u \in \text{CrsDesired}_s} \sum_{c \in \text{Sections}_u} y_{suc}.$$

B.1. Solving the model

To solve the model for assigning students to courses, we used a heuristic based on simulated- annealing. In its schedule building phase, the heuristic would randomly select a student who had one or more unsatisfied course preferences; and then, randomly select a course for which he or she had an unsatisfied preference. The heuristic would then attempt to assign the student to one of the sections of that course. It would repeat the examination process for all of the unsatisfied courses for that student; and then, it would randomly select another student with unsatisfied preferences. Once all students had been considered, the building phase was complete and the schedule was evaluated, marking the completion of one iteration of the heuristic. Following the schedule evaluation, the schedule would be perturbed by dropping 1.5% of the student- course assignments and then the rebuild process was started anew. The annealing parameters we used were as follows: initial temperature, 1.0; cooling rate, 0.95; iterations at same temperature, 5; temperature decrements, 25.

When solving the model as part of the overall course scheduling problem, we used a modified building process. Rather than start building the student-course assignments from scratch every time we came up with a new course schedule, we went back to the previous schedule. We then eliminated the student-course assignments only for those courses that had changed since the previous iteration, and we then ran the rebuilding process on the partial schedule. This reduced the time to find comparably good student-course schedules.

Table 1. Results across the 10 replications of each semester's schedule for the three solution heuristics

Replicate	Fall 2003	Spring 2004	Fall 2004						
	S-Control	S-Faculty	S-Assign	S-Control	S-Faculty	S-Assign	S-Control	S-Faculty	S-Assign
1	1321	1315	1358	1020	1042	1079	1081	1061	1117
2	1324	1319	1383	1021	1011	1081	1115	1055	1115
3	1327	1336	1392	1020	1030	1080	1092	1076	1134
4	1356	1333	1381	1027	1032	1068	1040	1062	1119
5	1325	1314	1368	1036	1027	1078	1083	1061	1138
6	1330	1342	1330	999	1031	1066	1100	1089	1082
7	1263	1347	1376	1040	1040	1084	1082	1060	1120
8	1304	1324	1368	1038	1043	1075	1083	1068	1117
9	1299	1322	1325	1014	1025	1083	1094	1051	1145
10	1320	1338	1380	1030	1032	1080	1058	1073	1133
Average	1316.9	1329	1366.1	1024.5	1031.3	1077.4	1082.8	1065.6	1122
Standard Deviation	24.35	11.71	22.45	12.49	9.43	6.04	21.07	11.2	17.52

Table 2. Performance of the three solution procedures across the three semesters, measured relative to *S-Control*.

Semester	Heuristic solution method		
	S-Control (%)	S-Faculty (%)	S-Assign (%)
Fall 2003	100	100.92	103.74
Spring 2004	100	100.66	105.16
Fall 2004	100	98.41	103.62
Overall	100	100	104.17

References

Abramson, D., 1991. Constructing college timetables using simulated annealing: sequential and parallel algorithms. *Management Science 37* (1), 98-113.

Alvarez-Valdes, R., Crespo, E., Tamarit, X.M., 2002. Design and implementation of a course scheduling system using Tabu search. *European Journal of Operational Research 137* (3), 512-523.

Asratian, A.S., de Werra, D., 2002. A generalized class-teacher model for timetabling problems. *European Journal of Operational Research 143* (3), 531-542.

Athiyaman, A., 1997. Linking student satisfaction and service quality perceptions: the case of university education. *European Journal of Marketing 31* (7), 528-540.

Badri, M.A., 1996. A two-stage multi-objective scheduling model for faculty-course-time assignments. *European Journal of Operational Research 94* (1), 16-28.

Badri, M.A., Davis, D.L., Davis, D.F., Holllingsworth, J., 1998. A multi-objective course scheduling model: combining faculty preferences for courses and times. *Computers and Operations Research 25* (4), 303-316.

Colorni, A., Dorigo, M., Maniezzo, V., 1998. Metaheuristics for high college timetabling. *Computational Optimization and Applications 9* (3), 275-298.

Cooper, T.B., Kingston, J.H., 1993. The solution of real instances of the timetabling problem. *The Computer Journal 36* (7), 645-653.

Costa, D., 1994. A tabu search algorithm for computing an operational timetable. European *Journal of Operational Research 76* (1), 99-110.

Dimopoulou, M., Miliotis, P., 2001. Implementation of a university course and examination timetabling system. *European Journal of Operational Research 130* (1), 202-213.

Dowsland, K.A., 1990. A timetabling problem in which classes are inevitable. *Journal of the Operational Research Society 41* (10), 907-918.

Elliott, K.M., Shin, D., 2003. Student satisfaction: an alternative approach to assessing this important concept. *Journal of Higher Education Policy and Management 24* (2), 197-209.

Ferland, J.A., Fleurent, C., 1994. SAPHIR: a decision support system for course scheduling. *Interfaces 24* (2), 105-115.

Foulds, L.R., Johnson, D.G., 2000. SlotManager: a microcomputer-based decision support system for university timetabling. *Decision Support Systems 27* (4), 367-381.

Hinkin, T.R., Thompson, G.M., 2002. SchedulExpert: course scheduling in the Cornell University School of Hotel Administration. *Interfaces 32* (6), 45-57.

Johnson, D., 1993. A database approach to course timetabling. *Journal of the Operational Research Society 44* (5), 425433.

Kang, L., White, G.M., 1992. A logic approach to the resolution of constraints in timetabling. *European Journal of Operational Research 61* (3), 306-317.

Kovačič, M., 1993. Timetable construction with Markovian neural network. *European Journal of Operational Research 69* (1), 9296.

Lau, L.K., 2003. Institutional factors affecting student retention. Education 124 (1), 126-136.

Sampson, S.E., Freeland, J.R., Weiss, E.N., 1995. Class scheduling to maximize participant satisfaction. *Interfaces 25* (3), 30-41.

Sampson, S.E., Weiss, E.N., 1995. Increasing service levels in conference and educational scheduling: a heuristic approach. *Management Science 41* (11), 1816-1825.

Smith, K.A., Abramson, D., Duke, D., 2003. Hopfield neural networks for timetabling: formulations, methods, and comparative results. *Computers and Industrial Engineering 44* (2), 283305.

Stallaert, J., 1997. Automated timetabling improves course scheduling at UCLA. *Interfaces 27* (4), 67-81.

Thomas, E.H., Galambos, N., 2004. What satisfies students? Mining student-opinion data with regression and decision tree analysis. *Research in Higher Education 45* (3), 251-269.

Umbach, P.D., Porter, S.R., 2002. How do academic departments impact student satisfaction? *Research in Higher Education 43* (2), 209-243.

Valouxis, C., Efthymios, H., 2003. Constraint programming approach for college timetabling. *Computers and Operations Research 30* (2), 1555-1572.