Course Scheduling and Timetabling at USMA

Dr. Hanif D. Sherali

Department of Industrial & Systems Engineering
Virginia Polytechnic Institute and State University
hanifs@vt.edu

LTC Patrick J. Driscoll

Department of Mathematical Sciences
U.S. Military Academy
pat_driscoll@usma.edu

OR METHOD:
Integer Linear Programming
APPLICATION AREA:
Manpower, personnel

ABSTRACT

We present a study of the course scheduling and cadet timetabling problem faced at the United States Military Academy (USMA) at West Point, New York. We discuss the salient features of this mixed integer assignment problem and the unique characteristics of the academic program at USMA. Accordingly, we develop two models to address their needs. The first model assumes that an assignment of *course-sections* to available daily time slots is given, and it seeks to determine a desirable, balanced, feasible set of cadet timetables using an objective function similar to that employed in goal programming. This model employs a weighted, linear objective function that penalizes soft constraint parameters in order to minimize deviations from constraint satisfaction and to capture the relative degree to which achieving this goal is emphasized. This model is currently being implemented by the Academy Registrar to investigate or evaluate several compositions of cadet course schedules, designed according to various stated cadet and organizational preferences. The second model considers the joint problem of scheduling course-sections within appropriate time slots as well as the allocation of cadets to classes. A logic-based, variable fixing solution procedure is presented that enables conflict-free timetabling.

INTRODUCTION

The academic program at the United States Military Academy (USMA) is designed around the requirement that all cadets must complete the program in four years, a total of eight academic semesters or terms, an objective that USMA shares with the other four military academies as well. Adding to the unique character of USMA is the fact that each cadet's daily activities are a carefully regimented balance of academic, military, and physical requirements. Scheduling options that typically provide flexibility to individual academic departments and Registrar operations in other colleges, universities, and training institutions, such as evening classes or offering a course once every two years, are usually not permissible at USMA. These, coupled with additional structural considerations, mandate that a customized approach be developed to automate the scheduling and timetabling operations at USMA.

The system being used to accommodate USMA's 19 majors and 23 fields of study has evolved into a complex manpower-intensive event that has outgrown the ability of the Academy to efficiently manage without updated computational support. To complicate matters further, the unique operating environment at USMA dictates that a customized approach be developed to facilitate this computational support due to the inability of off-the-shelf scheduling software to meet USMA's needs. Some recent effort has gone into developing a scheduling approach based on simulated annealing that is sufficiently flexible to deal with the many different conditions existing across a broad spectrum of institutions (see Thompson and Dowsland, 1998), along with genetic algorithms (see Burke, et al., 1995), but both their applications have so far been limited to the context of examination scheduling. The need to uniquely attack this type of assignment problem for large-scale systems was recognized early on by Tripathy (1984) and persists today.

Considerable attention has been paid in the operations research literature to academic course scheduling. Carter (1986), Junginger (1986), and White (1987) provide brief surveys of the literature during the early years. Sampson et al. (1995) provide a more recent literature review that subdivides scheduling attempts of this kind based on problem assumptions.

The mathematical programming approach that we present in this paper is largely in the spirit of Bloomfield and McSharry (1979), Dinkel et al. (1989), Junginger (1986), Kang and White (1992), Shih and Sullivan (1977), and Sampson et al. (1995), in that we assume that cadets have selected an eight term academic plan (8TAP) in conjunction with their academic advisors, and the problem remains to develop individual timetables for multiple-sectioned courses having finite capacity. Our problem is unique in that not only are the cadets' schedules fixed, as in Colijn (1973), Graves et al. (1993), Hosios and Rousseau (1980), Laporte and Desroches (1986), and Winters (1971), but institutional and departmental requirements, coupled with a fixed 4-year program-of-study, necessitate a problem formulation with extensive use of soft constraints and objective function penalty terms. Our development of the penalty values in the objective function parallels the methodology used to proportionally weight constraint deviations in a single-priority level goal programming objective function. In this sense, our approach is akin to goal programming without explicit recognition of conflicting institutional policies and faculty preferences, as in Badri et al. (1998), who developed a similar model for a scheduling problem required to explicitly recognize conflicting institutional policies, faculty teaching load requirements, classroom limitations, and faculty preferences. USMA currently resolves these multi-objective conflicts in advance of the timetabling problem. The models we develop allow us to modify them to accommodate automation of these tasks at some point in the future, should the Registrar's office so desire. Unlike the tacit assumption in goal programming that there exists no feasible solution without some compromise in constraint satisfaction, previous years' manual scheduling has sufficiently demonstrated the existence of such solutions. On the other hand, the environment has simply grown beyond the current manual methodology's ability to identify such feasible solutions in an efficient and effective manner.

This integer programming approach is being implemented as a core optimization module that is part of a larger database development project for USMA. Our approach is also conditioned on the following expressed desires of the Registrar.

- The cadet course schedules must be "good" in the sense of being conflict-free and satisfying all current term scheduling requirements of the cadets.
- The scheduling and timetabling application must run on a desktop PC and not require special computational resources to be successful.
- An inexperienced user should be able to make modifications to the program as requirements change.

In response, we introduce a dynamic, logic-based solution procedure that uses a primal LP relaxation to initiate a looping process involving extensive variable fixing. A very small integer program is used within the loop to help resolve the integrality requirements for individual cadet schedules. Our motivation in taking this approach is to resolve as much of the problem as possible prior to directly solving the pure integer program because of the computational complexity introduced by the size of the assignment problem. The Registrar's final requirement of the three itemized above is in response to their past experience in adopting customized programming solutions for their use.

In Section 1, we describe the unique environment at USMA that motivates this study. Section 2 then presents in detail the general modeling considerations specified by the Registrar that address the day-to-day class decomposition, courses, classrooms, and departmental priorities. Additionally, we introduce and define the notion of a *course-section* as the fundamental assignment unit. In Section 3, we develop the integer programming formulation for the timetabling problem alone, under the assumption that an assignment of course-sections to time slots is given *a priori.* Section 4 then relaxes this assumption, presenting a mixed integer program that addresses the joint problem of assigning course-sections to time slots and assigning cadets to course-sections.

Section 5 introduces the cadet-based and course-based logical tests used within the general logic-based solution procedure designed to produce conflict-free timetables for the cadets. We incorporate a new penalty term based on congestion related delays in traffic flows that provides a smoother incremental penalty for the objective function. Additionally, both the variable-fixing rules and the integer programming subproblem **CADET-IP** that is used to resolve the integrality of individual cadet timetables are developed. Finally, we conclude the paper with some comments on implementation and future research on this problem.

THE PROBLEM ENVIRONMENT

At USMA, the course scheduling and timetabling process is an annual event that encompasses approximately 18 weeks. As illustrated in Figure 1, a subtask known as the "535" period begins the scheduling process. During this short period, each of the 13 academic departments reviews their official course offerings

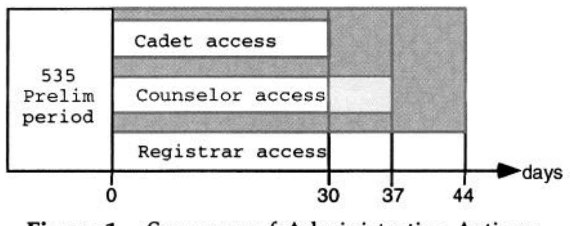

Figure 1. Sequence of Administrative Actions.

listed for the upcoming academic year. Additions, deletions and modifications to course listings are accomplished during this time. As of this writing, these tasks are accomplished manually, although the new corporate database initiative, of which course scheduling and cadet timetabling is a part, will migrate this task to a web-based, on-line activity in a manner similar to that used by Mathaisel and Comm (1991). The Cadet On-line Registration Process (CORPS) period follows 535, and is the time during which each new cadet has the opportunity to create an initial eight-term academic plan (8TAP), or, in the case of sophomores, juniors, and seniors, to make changes to an existing 8TAP to accommodate their individual academic programs. The CORPS period is a local area network implementation that subdivides into three overlapping periods.

Cadets make alterations to their individual 8TAPs in a manner that is consistent with the official course offerings, and that meets the guidelines established for their particular program of study. Academic counselors are then required to check each cadet's 8TAP for correctness, signing and submitting the approved 8TAP to the Office of the Dean. This step provides the first error-checking event in the scheduling process. At the end of the seven day extension allotted to the counselors for this purpose, the cadets and their counselors are locked out of the system. The remaining seven days allotted to the Registrar at the end of CORPS are used to make manual changes driven by unforeseen events such as loss of instructors, cancellation of course offerings, and so forth.

At the end of the CORPS period, projected enrollments for each course offering in the upcoming academic year are known. Although satisfying valid course offering criteria, it is commonplace for these course enrollments to violate departmental instructional desires. Thus, further changes are made in order to strike a term-to-term balance.

For example, consider the enrollments shown in Table 1. Physics 304 (PH304) requires balanced enrollments between academic terms 961 and 962 because the Physics department does not have sufficient instructors to teach the 37 sections (based on 17 cadets per section) requested in term 962. Similarly, it is inefficient for the English department to commit an instructor to teach three cadets in English 401 (EN401) during the term 962. The English department finds it more advantageous to free the enrollment during this semester by shifting the small enrollment to term 961.

Table 1. Term to term balancing

	Start		Finish	
	Term 961	Term 962	Term 961	Term 962
PH304	478	624	522	590
EN401	37	3	40	0

Changes such as these have a cascading effect, inducing other changes in previously feasible cadet schedules that again are manually altered. As a result, the Registrar's office has evolved a complex set of heuristics that they apply to cope with both the scheduling interactions and underlying combinatorial difficulties. As an aside, when we started working on this project, these complex heuristics had become so ingrained in the timetabling operation that the solution had essentially become the problem for the Registrar's office, and it was nearly impossible for them to recognize the fundamental nature of this assignment problem. It is the well-known combinatorial nature of this assignment problem that prohibits any enumerative implementation, and this fact motivates us to exploit both linear programming relaxations and an extensive use of variable fixing strategies when attempting to generate academic schedules that meet the requirements described herein.

MODEL DEVELOPMENT CONSIDERATIONS

There are about 4200 cadets at USMA, roughly 1000 per class. Each cadet enrolls in a set of five to eight courses during each term of their four years at USMA. Although the scheduling requirements of each cadet appear to be sufficiently unique to warrant a separate identity, there are obvious benefits to be gained by grouping sets of cadets having similar academic profiles and then block scheduling these groups where possible. From a mathematical programming viewpoint, this would do much to reduce the dimensionality of the problem and relieve some of the complicating combinatorics. In fact, this "templating" approach was routinely implemented at USMA during the years when only a single engineering major was awarded to graduates.

If such a grouping is performed, then the following development can be readily adapted by switching from single cadets to groups of cadets, with the added recognition that each group would now have a cardinality exceeding one, as opposed to cadet groups of size one considered herein.

At USMA, 1-hour time periods are defined for Day-1 meeting slots and Day-2 meeting slots, where Day-1 and Day-2 alternate over the working days for each term, there being 40 of each Day-type. These Day-1 and Day-2 time period configurations are illustrated in Figure 2 along with their time of occurrence, including what is called an R/S/T/U 2-hour laboratory time slot scheduled during the Day-2 periods. These laboratory periods satisfy the needs of a limited number of courses offered at USMA to augment their regular course meetings with eight periodic two-hour laboratory periods. Hence, each of these R, S, T, U slots appear once every four Day-2 occurrences, and are scheduled cyclically in this order. They occupy both the Dean's hour and K-hour. The shaded area between time periods indicates the 10-minute travel time allotted between classes.

Students are generally assigned to only 1 of the 4 possible R/S/T/U laboratory sessions during any term. This laboratory assignment precludes any additional classes from being scheduled during the free hours when the laboratory is not meeting, *i.e.*, any K-hour assignments are subsequently blocked. For convenience, we will ascribe a numerical index for these lettered time slots in which A = 1, B = 2, and so on until L = 12.

Courses

There are five types of courses that are offered at USMA. They are:

1. 40 lesson courses that require 1-hour class meetings on a Day-1 or a Day-2 schedule. An assignment of a course to slot K is undesirable since it blocks out any accompanying R/S/T/U laboratory assignment.
2. 47/48 lesson courses that are scheduled for 40 1-hour class meetings on Day-1 or Day-2, plus an additional 7/8 1-hour meetings during an R/S/T/U 2-hour laboratory period. Again, a slot K assignment is deemed undesirable.
3. Courses requiring greater than 48 hours of meeting time, and up to 80 hours, are scheduled in 2-hour blocks on a Day-1 or Day-2 schedule. The preferred blocks are AB, CD, EF, GH, and IJ. The block KL is admissible, but undesirable, because this prohibits any accompanying R/S/T/U assignment. Similarly, blocks BC and HI are admissible, but undesirable, because this eliminates a scheduling of 2-hour blocks of type AB and CD, or GH and IJ, respectively, for such cadets.
4. A set of three particular core mathematics courses: MA103, MA104, and MA205, along with their associated accelerated counterparts (X-courses), and possibly MA100, are scheduled across days, being offered on both Day-1 and Day-2 schedules over periods AG, BH, CI, or DJ. (Each cadet would take at most one such course per term.) Note that an AG assignment for any cadet would block out the 2-hour slots AB and GH, but would permit the scheduling of classes during the single hour slots B and H for this cadet.

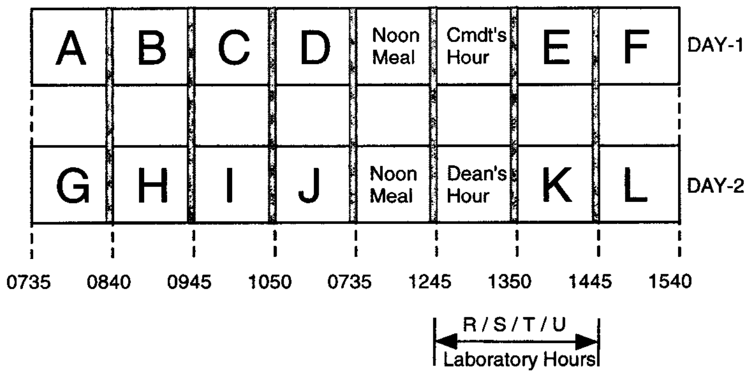

Figure 2. Day-1 and Day-2 class periods.

5. There are also certain peculiar 16 meeting-period courses that are usually assigned in 2-hour blocks over the R/S/T/U laboratory periods, and certain special, low enrollment courses that are assigned to special hours designated by the Registrar as X, Y, and Z hours. The allocation of such courses is considered to be pre-fixed, and extraneous to the models that are developed below.

As an example, during the Spring term 952, there were over 1500 classroom sections taught across all 4 cadet classes, representing 316 courses. Of these, 190 were single-hour courses, 44 were double-hour courses, 44 were single-hour along with an R/S/T/U laboratory assignment, and there were 38 courses of type (5). Of the latter, 2 were double-hour 16-period courses assigned to R/S/T/U time slots, and 36 were designated as Z-hour courses.

Several comments regarding the pre-fixing of courses to time slots are in order. There exists a small number of courses requiring laboratories to be scheduled during the R/S/T/U time slots. Because of the aforementioned conflicts between a K-hour course and such R/S/T/U assignments, a course of this type should not be scheduled in a K-hour slot during the first pass of a solution procedure whenever the schedule of the course is yet to be determined. However, specific courses can be *pre-fixed* to a K-hour assignment and the associated conflicts can be accordingly analyzed, realizing that cadets enrolled in this course would not be permitted to enroll in any R/S/T/U laboratory sessions. This can be done in either the first pass itself, or during a second pass of the solution procedure after a first pass has indicated a need to do so.

In a similar vein, we assume that laboratory assignments to R/S/T/U time slots are done *after* the other scheduling and cadet timetabling is complete, so that assignment possibilities, desires, and conflicts are evident. This can be done manually, or an additional model can be constructed to automate this allocation process in a manner similar to that done herein. By this token, courses of type (1) and (2) can be treated uniformly as requiring 40 1-hour meetings over the regular Day-1 or Day-2 slots. Note that such a multiple pass solution process can also treat the assignment of courses to X, Y, and Z hour slots, in an attempt to resolve infeasibilities and conflicts, besides the regular pre-assignment of type (5) special courses to such time slots.

Classroom Availability

Classrooms at USMA are categorized according to capacity, corresponding to the following types:

Type	Number of Classrooms
18	240
40	1
50	5
80	5
Auditorium	4

This listing dictates the maximum number of sessions that can be held in parallel during any particular time slot. Note that it might be possible that some courses cannot occur in parallel because of a physical location or subject constraint. We assume that these conflicts have been resolved prior to the use of the models presented in the sequel, with the model parameters and variables being accordingly defined.

Course-sections

The projected number of sections needed for each course during a term is predetermined by the individual academic departments prior to scheduling and timetabling. For each course, this value is based on the total enrollment request and the class size, as per the information provided by the academic departments during the 535 scheduling period. For example, for the most common class size of 18, the following formula is used (with 17 being used as a conservative factor):

Number of sections

$$= \left\lceil \frac{\text{total enrollment request}}{17} \right\rceil$$

where $\lceil \cdot \rceil$ denotes the rounding-up operation. As done above, we refer to the collection of sections over all courses jointly as *course-sections*. For example, if the total enrollment requests for MA102 and EN201 were 948 and 872, yielding 56 and 52 sections respectively, then the total number of course-sections over these two courses is 108. A department will limit the number of sections offered for a particular course only when they cannot provide an instructor to teach it.

Departmental Priorities

At USMA, cadet needs are treated with the highest priority, and take precedence over individual faculty and departmental desires or requests. However, given that cadet needs are met, departmental and faculty requests are accommodated to the extent possible. Such desires, for example, stated faculty preferences for teaching during desired hours, can be accommodated within our models via appropriate objective function coefficient manipulations. In Model 1, developed in the following section, we assume that the assignment of course-sections to time slots has occurred and an accompanying feasible set of cadet timetables is sought. Several combinations of course schedules can be composed using expressed faculty desires, and these can be individually investigated for feasibility using Model 1. Or, a direct methodology of incorporating faculty teaching preferences into the model itself could be accommodated, as in Badri et al. (1998).

The cadet 8TAPs provide a base enrollment projection for the upcoming semester, which in turn identifies the projected number of course sections required for each course. Faculty members are then queried in advance of cadet timetabling as to what day/hour they wish to teach for the coming semester. Resulting conflicts are manually removed based on normal operating constraints (e.g., two courses from different departments cannot be taught during the same hour because both are required for a particular major) and based on projected course workload (papers, projects, etc.). Some attempt is also made to balance the course-section offerings over Day-1 or Day-2 schedules, while attempting to iteratively design an overall schedule. The cadet/faculty ratio at USMA is ideally targeted to be 15/1, with 17/1 being considered normal and 20/1 an upper bound.

Modeling Considerations

There is a high degree of constancy in the academic cycle at USMA from year to year, a fact that can be conveniently exploited within our model. Thus, we opted to use historical information to first investigate if a previously successful course schedule would admit a feasible timetabling of cadets. In doing so, we also recognize that minor modifications due to variations in course-sections offered or expressed faculty desires may have to be incorporated.

As stated before, Model 1 assumes that the course-sections are already scheduled, and one is to determine a desirable assignment of cadets to classes. The second model we develop then considers the *joint problem* of scheduling course-sections during appropriate time slots and the allocation of cadets to classes (timetabling). The outputs from solving either model are written to text files that populate record fields in the database via a separate interface. This allows the Registrar to produce various reports that are cadet-based, classroom-based, course-based, as well as teacher-based, as desired.

An important fact to bear in mind is that the computerized models and procedures to be implemented are treated as an *aid to decision making*, and are not intended to replace the Registrar as the decision maker. Both models' main purpose is to help the scheduler by transferring the burden of the combinatorial nature of the problem from the individual to the computer, and to provide an interactive tool that greatly reduces the 18 weeks it presently takes an *experienced* scheduler to construct a timetable for a given term.

MODEL 1: TIMETABLING

This model assumes that an assignment of course-sections to time slots is given, and it seeks to determine a desirable timetable for each cadet. There are various constraints that must be satisfied as *hard constraints*, and there exist *soft constraints* whose satisfaction is driven by a desirability index expressed through weighted coefficients in the objective function. These constraints, along with other preferences represented in the objective function, are all described below within the formulation of the model. It is important to point out that this model can be solved separately for cadets in each class year, thereby reducing the size of the problem to be solved. If used, this option must be accompanied by an explicit prioritization for scheduling by year group because many of the elective course offerings have mixed year group cadet enrollments. Fortunately, such a natural prioritization exists because of the large proportion of mandatory courses in the first two years. Senior cadets receive the highest priority for course scheduling, followed by juniors, sophomores, and then freshman.

Notation

The following notation is used in common for both Model 1 and Model 2.

Notation	*Representing*
$i = 1, \ldots, C$	Individual cadets
$r = 1, \ldots, R$	Individual courses
$j = 1, \ldots, J$	Total course-sections
$m = 1, \ldots, M$ for $M = 12$	Individual 1-hour time slots (A, B, ..., L)
$S_r = \{j$: course-section j belongs to course $r\}$	Subgrouping of course-sections
$T_i = \{r$: cadet i requires course r in the current term$\}$	Student's needed courses
$P_m = \{j$: course section j occupies or overlaps period m$\}$	Period m conflict flag set
E_r (parameter)	Average enrollment per section of course r
Q_j^* (parameter)	Desired enrollment for course-section j, with an excess beyond this being subject to penalty
$Q_{j\,min}$, $Q_{j\,max}$ (parameters)	Absolute minimum and maximum enrollment for course-section j
J_1, J_2	Subdivision of course-sections for Day-1 and Day-2
$q_{r\,max}$ (parameter)	Maximum enrollment deviation permitted over sections of course r
δ_{max} (parameter)	Maximum difference permitted between Day-1 and Day-2 cadet course loads
C_A	Set of all cadet athletes
C_{SW}	Set of all Corps Squad (varsity) Wrestlers
C_{SA}	Set of all Corps Squad (varsity) Athletes
J_{AJ}	Set of course-sections j offered during periods A or J
J_{FL}	Set of course-sections j offered during periods F or L
$\lfloor\cdot\rfloor$	Denotes the rounding down operation

Decision variables. These decision variables are binary integer variables defined as:

x_{ij} = {1 if cadet i is assigned to course-section j, 0 otherwise}.

Deviation variables. These variables are used to absorb violations of soft constraints, if any such violation occurs during the solution process. Because they have positive coefficients in the objective function, the minimization will attempt to force these violations to zero as possible, and the objective function value is penalized when they assume nonzero values.

q_r = enrollment deviations among sections of course r, with a maximum cap of $q_{r\,max}$, $\forall r$

ξ_j = excess enrollment for course-section j over desired enrollment Q_j^*, $\forall j$

δ_i = deviation over Day-1 versus Day-2 course loads for cadet i, $\forall i$

ζ_j = excess (over 60%) of the proportion of athletes in course-section j, $\forall j$

The value of ξ_j is bounded above in the formulation that follows by the value $Q_{j\,max} - Q_j^*$. The value of δ_i is likewise bounded above by δ_{max}.

Model Constraints

The following constraints define the problem as described by the Registrar. This is the portion of the mathematical program that will be modified if the problem requirements change.

Student assignment and non-conflict constraints. These assign a cadet precisely once to each particular course-section they require:

$$\sum_{j \in S_r} x_{ij} = 1 \quad \forall r \in T_i, \forall i \tag{1}$$

while insuring that no two scheduled course-sections meet during the same time period:

$$\sum_{j \in P_m} x_{ij} \le 1 \quad \forall m, \forall i. \tag{2}$$

Hard constraints on course-section enrollments. These constraints allow the deviation variable ξ_j to assume values between its bounds, and to maintain the enrollment for each course-section j above its minimum specified level, while providing the facility to penalize enrollments above the desired level Q_j^* via the variable ξ_j being used in the objective function.

$$\xi_j \geq \sum_i x_{ij} - Q_j^* \text{ where } 0 \leq \xi_j \leq Q_{j\,\max} - Q_j^* \quad \forall j \quad (3)$$

$$\sum_i x_{ij} \geq Q_{j\,\min} \quad \forall j \quad (4)$$

Soft constraints balancing enrollments over sections of course r. These constraints allow the scheduler to suggest the maximum deviation, $q_{r\,\max}$, permitted in course-section enrollments for each course r, as compared to an average course-section enrollment value E_r. Deviations away from this suggested value are penalized using the objective function.

$$\lceil E_r \rceil - q_r \leq \sum_i x_{ij} \leq \lfloor E_r \rfloor + q_r \quad \forall j \in S_r, \quad \forall r \quad (5)$$

$$0 \leq q_r \leq q_{r\,\max} \quad \forall r. \quad (6)$$

Soft constraints balancing each cadet's Day-1 and Day-2 course loads. The minimum course load per semester for all cadets is five courses. In these constraints, the maximum deviation in load for any cadet, $\delta_{\max}$, is specified in advance of solving the timetabling problem. A positive deviation for any cadet i is penalized in the objective function.

$$-\delta_i \leq \sum_{j \in J_1} x_{ij} - \sum_{j \in J_2} x_{ij} \leq \delta_i \quad \forall i \quad (7)$$

$$0 \leq \delta_i \leq \delta_{max}. \quad (8)$$

Note that these constraints can be enforced as hard load-balancing constraints requiring perfect balance between Day-1 and Day-2 by setting $\delta_i = 0$ if $|T_i|$ is even, and $\delta_i = 1$ if $|T_i|$ is odd.

Soft constraints to achieve fewer than 60% athletes in any course-section.

$$\zeta_j \geq \sum_{i \in C_A} x_{ij} - 0.6 \sum_i x_{ij} \quad \forall j \quad (9)$$

where $\zeta_j \geq 0$ is penalized in the objective function for taking on positive values. This constraint is desired to foster student interaction. When the population of athletes exceeds this threshold, a noticeable partitioning of the students occurs that can create dysfunctional cliques. The target value of 60% is set as a parameter that can be varied as desired to perform 'what if?' analyses. This constraint can also be enforced as a hard constraint by giving ζ_j an excessive penalty coefficient in the objective function or setting it equal to zero.

Corps squad (varsity) wrestler/athletes free period constraints. These constraints prevent Corps squad wrestlers from being assigned to a course-section occupying either an A or J hour time slot. This prevents their academic schedule from conflicting with mandatory practice sessions. Likewise, they prevent all Corps squad (varsity) athletes from being scheduled into F or L hour time slots for the same reason.

$$x_{ij} \equiv 0 \quad \forall i \in C_{SW} \text{ and } j \in J_{AJ} \quad (10)$$

$$x_{ij} \equiv 0 \quad \forall i \in C_{SA} \text{ and } j \in J_{FL}. \quad (11)$$

Departmental constraints and considerations. It was obvious at the start of the project that the academic departments had added a considerable number of requirements that were complicating the scheduling heuristics being used by the Registrar. Our math programming formulation enables the Registrar to determine which of these additional constraints are confounding the scheduling process by rendering it infeasible, and which are binding, having the most influence on the resulting schedules. A representative sample of methods that can be used to incorporate these types of requirements is illustrated below.

1. Regimental sectioning of Leadership (PL300) can be enforced as a hard constraint by defining T_i accordingly, or as a soft constraint by rewarding appropriate x_{ij} assignments in the objective function.
2. To accommodate a last free hour for the cadet field music group, this free hour can be considered as a course-section that is assigned to period F or L, with the corresponding cadets being required to be assigned to this course-section via constraints (1) and (2). Note that this assignment would be excluded from other balancing constraints.
3. Separate course-sections that are designated as "first year cadets only" can be accommodated by giving a special designation to such

course-sections, and by defining the sets T_i for first year and other cadets accordingly.

The Objective Function

In a similar spirit as a goal programming objective function, which seeks to minimize the total deviation from conflicting objectives, the minimization of this objective function drives the optimization. It forces the soft constraint parameters (deviation variables) as close to zero as possible while insuring that the feasibility of the constraints is maintained.

$$\text{Minimize} \sum_j \alpha_j \xi_j + \sum_r \beta_r q_r + \sum_i \gamma_i \delta_i + \sum_j \theta_j \zeta_j + [\text{Dept constraint weighting on } x_{ij} \text{ as appropriate}] \quad (12)$$

where α_j, β_r, γ_i, and θ_j and other objective function coefficient weights are suitably scaled to be commensurate with each other, and to recognize desired priorities established by the Registrar's office. For example, as a simple starting scaling strategy, we noted that ξ_j, q_r, and ζ_j are in terms of enrollments, while δ_i is in terms of courses, and that most courses have about 17 cadets per class. Consequently, we let $\alpha_j = 1 \; \forall \; j$, $\beta_r = 1 \; \forall \; r$, $\theta_j = 1 \; \forall \; j$, and $\gamma_i = 17 \; \forall \; i$. The Registrar's office suggested that an equal prioritization between these soft constraints was appropriate since violations to any of the specified constraint conditions would require a similar administrative action. Hence, no further modification of penalty coefficients was mandated beyond that of insuring commensurate weighting as noted above. Had they expressed a different degree of prioritization for constraint satisfaction, we would have then taken the above simple weighting scheme and modified it with additional weights in proportion to the reverse order of the Registrar's priority ranking, as in Badri et al. (1998).

MODEL 2: JOINT COURSE SCHEDULING AND TIMETABLING

This second model addresses the joint problem of assigning course-sections to daily time slots and assigning cadets to course-sections. The common resource requirement that ties together the different class years is that of classrooms. However, by sequentially scheduling class year groups in the priority order suggested earlier, or by restricting the number of classrooms of each type that can be used by each class year group in such a sequential process, the problem can again be solved separately for each class year, thereby reducing the problem size.

It should be noted that this second model only provides suggestions for assigning course-sections to time slots, subject to the availability of the required types of classrooms during each time slot. It does not assign course-sections to *specific* classrooms, which is currently done prior to scheduling and timetabling. This latter assignment can be done subsequently during the report phase either manually or by solving a suitable matching problem formulated for this purpose.

Notation

In addition to the notation used for Model 1, the following notation and definitions are used in Model 2.

Decision variables

$$y_{jm} = \begin{cases} 1 & \text{if course-section } j \text{ begins at time slot } m \\ 0 & \text{otherwise,} \quad \forall j, m. \end{cases}$$

The situation in which a 2-hour course-section needs to be assigned to two sequential 1-hour time slots is accommodated during the solution process. For example, if $y_{j1} = 1$ for a course-section j requiring two time slots, this means that the course is actually offered over the AB (index 1 and 2) block duration. We maintain this reference information in a master options matrix that contains all the valid course offering time slot combinations. Such a two-hour course requirement then requires only a single time slot index reference to identify and appropriately schedule it. In addition, we define

$$z_{ijm} = \begin{cases} 1 & \text{if cadet } i \text{ is assigned to course-section } j \text{ beginning at time slot } m \\ 0 & \text{otherwise, } \forall i, j \in J(i), m \in M_j \text{ where } J(i) \text{ and } M_j \text{ are defined below.} \end{cases}$$

It is important to note that an interrelationship exists among the three decision variables we have defined via $z_{ijm} = x_{ij}y_{jm}\ \forall i, j, m$.

Notation	*Representing*
M_j	Set of periods m during which course-section j can be assigned to begin.
$J(i) \equiv \cup_{r \in T_i} S_r$	Set of course-sections that are admissible for cadet i.
Q_p	Set of course-sections for which no more than $\tilde{q}_p$ can be scheduled simultaneously during any time slot, $p = 1, \ldots, P$, where P is the number of such sets that have been identified.
$\tilde{q}_p$ (parameter)	Maximum number of course-sections in Q_p that can be scheduled simultaneously during any time slot $p = 1, \ldots, P$.
n_v (parameter)	Number of classrooms of size-type v available for scheduling, $v = 1, \ldots, V$.
F_{tv}	Sets of index pairs (j, m) in which $y_{jm} = 1$, indicating that a classroom of size-type v is occupied during period t, $v = 1, \ldots, V$, and $t = 1, \ldots, 12$.
$F_t \equiv \cup_V F_{tv}$	Sets of index pairs (j, m) in which $y_{jm} = 1$ indicates that period t is occupied for cadets enrolled in course-section j that is assigned to begin at time period $m = 1, \ldots, 12$.
M_1, M_2	Sets indicating Day-1 and Day-2 periods: $M_1 = \{1, \ldots, 6\}$, $M_2 = \{7, \ldots, 12\}$.

Several comments are in order. The creation of sets Q_p might restrict the number of sections of any particular course that can be offered simultaneously, or might prohibit the parallel scheduling of certain pairs/sets of courses that are taught in single sections and that cannot be offered simultaneously due to location or subject content reasons. If any of the latter type of courses are offered in multiple sections, then a similar type of constraint can be used by introducing suitable new deviation variables.

The new subscript index t is required because, although t and m index the same 12 time slots, the two subscripts will only coincide when 1-hour course-sections are being assigned. When a 2-hour course-section j is assigned to, say, block AB, *i.e.*, $y_{j1} = 1$, $m = 1$ provides direct information that period A is being occupied, but period B will be implicitly accounted for, noting that both of the classroom assignment tracking sets F_{1v} and F_{2v} will contain the index pair $(j, 1)$. This added subscripting is a modeling convenience in addition to the master options matrix mentioned earlier. Notice also that a core mathematics course that is scheduled across days, say AG, has $y_{j1} = 1$ so that both F_{1v} and F_{7v} contain index pair $(j, 1)$.

To alleviate undue complications, rather than treating all course-section assignments to time periods as unknown, some particular course-sections can be pre-assigned to certain time slots by simply fixing $y_{jm} = 1$ at the onset. For example, we anticipate that this may be an appropriate strategy for core mathematics courses that need to be scheduled across days. Simplifying the constraints and decision variables in this manner is a significant aid in reducing the computational burden involved with solving the mathematical program, and is a common pre-processing step in practice.

Model Constraints

The following constraints define Model 2. Whenever a particular constraint is the same as that defined in Model 1, we simply note this fact.

Decision variable relationships

$$x_{ij} = \sum_{m \in M_j} z_{ijm} \quad \forall i \text{ and } \quad \forall j \in J(i) \qquad (13)$$

$$x_{ij} + y_{jm} - 1 \leq z_{ijm} \leq y_{jm} \quad \forall i \text{ and } \quad \forall j \in J(i), m \in M_j. \qquad (14)$$

Note that due to (13) the variables x_{ij} can be directly eliminated from the problem if desired, except if they serve to facilitate the operation of a designed solution algorithm.

Student assignment and non-conflict constraints:

$$\sum_{j \in S_r} x_{ij} = 1 \quad \forall r \in T_i, \text{ and } \quad \forall i \tag{15}$$

$$\sum_{(j,m) \in F_t : j \in J(i)} z_{ijm} \leq 1 \quad \forall i, t \tag{16}$$

Course-section enrollment constraints. Same as constraints (3) and (4) of Model 1.

Course enrollment balancing constraints. Same as constraints (5) and (6) of Model 1.

Course-section assignment and non-conflict constraints. These constraints enforce that each course-section must be assigned to some time slot while insuring that no classroom conflicts occur for assigned courses.

$$\sum_{m \in M_j} y_{jm} = 1 \quad \forall j = 1, \ldots, J \tag{17}$$

$$\sum_{(j,m) \in F_{tv}} y_{jm} \leq n_v \quad \forall t = 1, \ldots, M \quad \text{and } v = 1, \ldots, V \tag{18}$$

Soft constraints balancing each cadet's Day-1 and Day-2 course loads. These constraints are analogous to (7) and (8) defined in Model 1.

$$-\delta_i \leq \sum_{j \in J(i)} \left(\sum_{m \in M_1} z_{ijm} - \sum_{m \in M_2} z_{ijm} \right) \leq \delta_i \quad \forall i \tag{19}$$

$$0 \leq \delta_i \leq \delta_{max}. \tag{20}$$

Soft constraints to achieve fewer than 60% athletes in any course-section. These constraints are the same as (9) but now can be written in terms of z_{ijm} in a disaggregated fashion for each m. This is in contrast with directly employing the substitution (13) in (9).

Corps squad wrestler/athlete constraints. Same as constraints (10) and (11) in Model 1.

Simultaneous offering restrictions. These constraints prevent the maximum number of course-sections available for simultaneous assignment from being exceeded.

$$\sum_{j \in Q_p : m \in M_j} y_{jm} \leq \tilde{q}_p \quad \forall p = 1, \ldots, P \quad \text{and } m = 1, \ldots, M. \tag{21}$$

Departmental constraints and considerations. As in the case of Model 1, a number of model-specific constraints are added to accommodate departmental desires. A representative sample of these constraints are described below.

1. Some departments desire to offer elective courses three times on the same day, or offer electives twice over consecutive blocks on the same day. Although such constraints can be modeled as disjunctive constraints using binary variables, this would unduly complicate the model by increasing its combinatorial complexity. If such a desire needs to be satisfied, it is better to attempt to fix the corresponding y_{jm} variables *a priori*, and then attempt to find a feasible completion to the schedule. However, a preferred approach would be to let the objective function drive toward attaining these restrictions by rewarding some selected sets of y_{jm} variable assignments that conform with such requests. Similarly, the teaching time preferences expressed by faculty can be suitably rewarded in the objective function in order to seek solutions that tend to comply with such requests, if feasibility so permits.
2. To discourage scheduling course-sections into non-traditional blocks such as BC or HI, the corresponding y_{jm} variables should be penalized in the objective function, rather than being explicitly fixed at zero. This consideration is therefore treated as a matter of optimality that effects the *quality* of the resulting solution, as opposed to feasibility, which concerns the *existence* of a viable solution.
3. Some scheduling desires associate courses in different academic departments, such as Computer Science 105, with certain core mathematics courses. For example, if MA103 is offered with a normal class period during time slot B and a mathematics laboratory in time slot H, then Computer Science 105 is offered during periods G and I on the non-teaching day of MA103. We can accommodate this desire by including the constraint (for clarity, we employ course names and time slot letters rather than indices below):

$$y_{CS105,G} \geq y_{MA103,B}$$

with the appropriate index pair (j, m) included in $F_{1,v}$ if $y_{CS105,G} = 1$. Similar constraints can be written for $y_{MA103,H} = 1$, implying that the laboratory is during period B, and for which CS105 would then be assigned to periods A and C.

Objective Function

The objective function to be minimized for Model 2 is of the same type as for Model 1, with additional terms incorporating rewards or penalties on y_{jm} variables due to departmental constraints and considerations as noted above. Note that in the spirit of Model 1, if a prior schedule defined by some fixed values $\bar{y}$ for the y-variables is available, then the objective function can be encouraged toward this prior timetable via the linear objective function expression:

$$\text{Maximize} \sum_{j} \sum_{m} \bar{y}_{jm} y_{jm}.$$

This expression can be combined with the aforementioned objective function (12) using suitable relative weights, again chosen to insure that the different terms are commensurate with each other.

LOGIC-BASED SOLUTION PROCEDURE

One of the goals of this research was to create a desktop PC scheduling tool that will identify feasible, and possibly near-optimal assignments. With this criteria in mind, we concluded that a naive use of off-the-shelf optimization software to solve either Model 1 or Model 2 would be ill-advised due to either model's size, taken in consideration with the underlying combinatorial challenge. This is particularly true with Model 2. We recognize that with some appropriate aggregation of cadets into groups with similar requirements, such as grouping Operations Research majors in the same year group, Model 1 could possibly be solved using standard optimization software and are currently investigating this possibility.

The proposed logic-based, variable-fixing procedure is based on solving a sequence of primal LP relaxations. Within the main loop, the integrality of each cadet's timetable is resolved via a small IP subproblem. The advantages to this heuristic approach are that we obtain near-optimal integer solutions and the subproblems are simple. The disadvantages are that the manipulations and looping procedures are computationally intensive, although the deliberate steps prescribed become necessary to insure feasibility under tight scheduling situations.

Although Model 1 is being used for the initial implementation, in the event the Registrar desires to incorporate the features of Model 2 at a later date, we intend to fix the y-variables above in a first stage, thereby reducing Model 2 to Model 1, and then apply the solution procedure for Model 1 that is described below. In both models, we recognize that some iterative perturbations in the fixing of variables might be necessary/advisable in deriving a final solution. We have incorporated this feature into the implementation of Model 1, and are currently researching the degree to which this is necessary.

Yet another possibility for solving Model 2 is to first determine an assignment of cadets to course-sections (as dictated by the x-variables), and then assign the *manned-course-sections* to time slots in an attempt to determine a feasible schedule. Tripathy (1984) demonstrates how a related resulting problem in y-variables could be modeled and solved using techniques similar to those introduced herein.

Recall that the previous scheduling system that is being replaced by these models is a simple bin-packing algorithm in COBOL that schedules cadets one-at-a-time until the requested assignment is blocked. When no feasible assignment is available, the schedule is rejected from the program. The rejection rate is approximately 20–30%. Out of a class of 1000 cadets, these 200–300 rejected schedules are then manipulated by-hand to resolve the remaining cadets' schedules. As undesirable as this sounds, a 70–80% success rate with an unsophisticated algorithm led us to suspect that a logic-based approach that employs rather sophisticated variable-fixing schemes might be sufficient to solve the problem.

Figure 3 presents the overall procedure. The initialization step establishes the data structures of the problem and sets up key elements that are required to construct the model representation and execute the algorithmic steps. Since some variables are pre-fixed as recommended earlier, certain logical tests are performed to fix additional x-variables at binary values governed by the hard constraints, or to penalize certain assignments as identified by

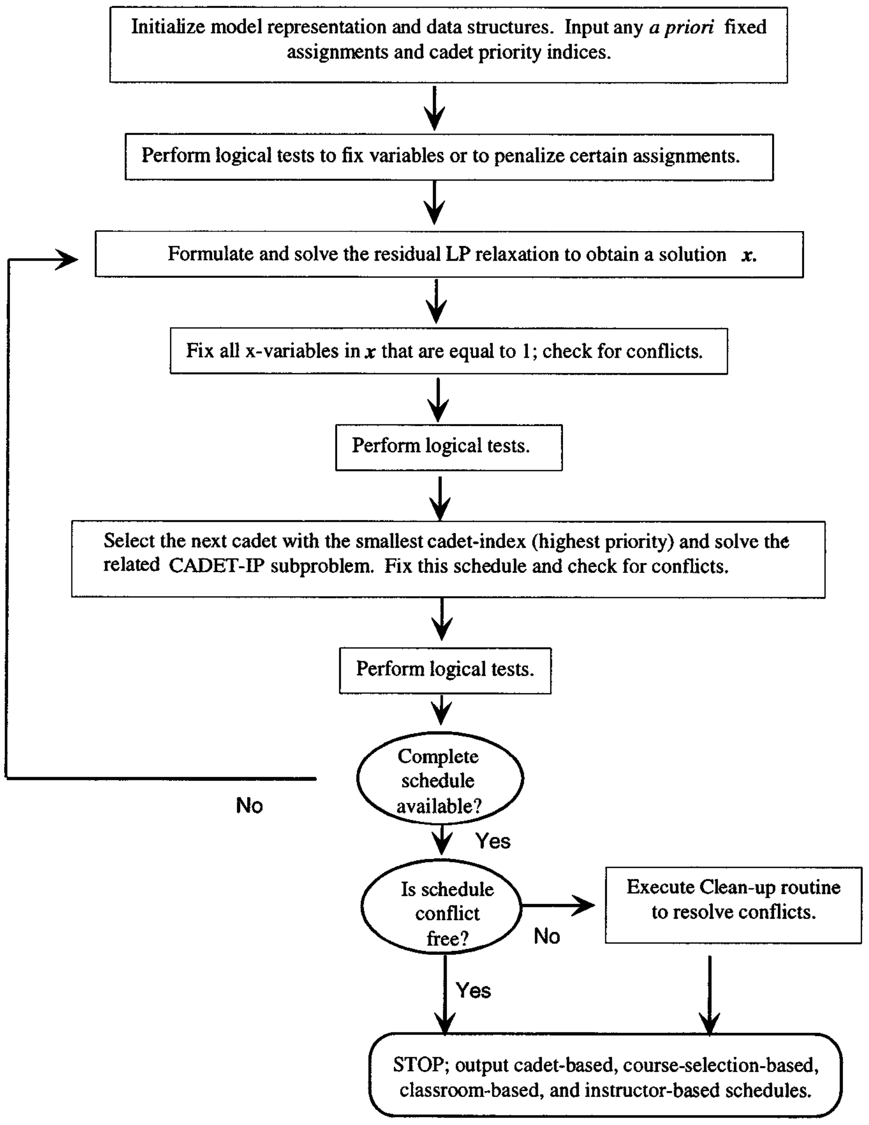
Initialize model representation and data structures. Input any *a priori* fixed assignments and cadet priority indices.
Perform logical tests to fix variables or to penalize certain assignments.
Formulate and solve the residual LP relaxation to obtain a solution *x*.
Fix all x-variables in *x* that are equal to 1; check for conflicts.
Perform logical tests.
Select the next cadet with the smallest cadet-index (highest priority) and solve the related CADET-IP subproblem. Fix this schedule and check for conflicts.
Perform logical tests.
Complete schedule available?
No
Yes
Is schedule conflict free?
No
Execute Clean-up routine to resolve conflicts.
Yes
STOP; output cadet-based, course-selection-based, classroom-based, and instructor-based schedules.

Figure 3. Logic-based solution procedure.

the soft constraints. Following this, the primal LP relaxation is solved. The algorithm now enters a main loop in which a partial fixing of variables and a subsequent updating of the LP solution is conducted. The variable fixing is done based on the variables that take on a value of 1 in the solution to the LP relaxation in concert with an IP subproblem called **CADET-IP** that completes a single cadet's schedule. The choice of which cadet to consider next is based on a special prioritizing cadet-index introduced in the next section that can be influenced by user input. Finally, in order to resolve any conflicts that might exist in the constructed solution, a clean-up routine is run. Any persistent conflicts can be either resolved manually, or by running the algorithm again using certain modified pre-fixed assignments or prioritized cadet-indices. In what follows, at any stage of the procedure, we will have some variables that are *fixed* at 0 or 1 values, while the remaining variables will be referred to as *free* variables.

Initialization and Data Structures

The principal data structure employed maintains for each cadet i the set of courses remaining to be scheduled (T_i), and for each course $r \in T_i$, the course-sections j that are available for this cadet on a Day-1 and Day-2 schedule. Notationally, for each cadet $i = 1, \ldots, C$, let

$$C_{ir1} = \{j \in S_r : j \in J_1,\ x_{ij} \text{ if a free variable with } x_{ij} = 1 \text{ permissible}\} \quad \forall r \in T_i \tag{22}$$

$$C_{ir2} = \{j \in S_r : j \in J_2,\ x_{ij} \text{ is a free variable with } x_{ij} = 1 \text{ permissible}\} \quad \forall r \in T_i \tag{23}$$

$$C_{ir} = C_{ir1} \bigcup C_{ir2} \quad \forall r \in T_i \tag{24}$$

$$C_i = \bigcup_{r \in T_i} C_{ir} \tag{25}$$

Note that the sets T_i, C_{ir1} and C_{ir2}, with the accompanying sets C_{ir} and C_i are *residual* sets that are continually updated when any variable fixing occurs. We also need to maintain a cadet-index $\theta_i \equiv |C_i|$ for all $i = 1, \ldots, C$ that we use to establish a prioritization for scheduling cadets. In a sense, θ_i measures the assignment flexibility of a cadet. If $|C_j| < |C_i|$, then cadet j has fewer options remaining for course-section assignment than cadet i. The smaller the value of θ_i, the higher will be the priority for constructing the schedule for cadet i first. This approach is analogous to the computationally effective saturation degree rule proposed by Brelaz (1979) and implemented by Carter, et al. (1996) to identify examinations having the least flexibility in choice of scheduling periods. These indices may be modified at the onset by subtracting suitable constants from them to induce a desired prioritization, or by setting group-based priorities for classes of cadets (e.g., in-season varsity swimmers).

Notice also that by these set definitions the set of free x-variables in the problem are given by x_{ij}, $j \in C_i$, $i = 1, \ldots, C$, with any prohibited assignments being absent. For cadets having partially fixed schedules, some of the x_{ij} variables will be fixed at one, having induced other conflicting assignment variables to zero values during the pre-processing stage. Thus, the model constructed is composed of *free* variables alone.

The Model 1 objective function is used here, but with additional penalties for certain free variables as described below. These penalties are identified by the logical tests described later.

Lastly, we need to keep a running track of fixed enrollments of course-sections using the notation

$$FE_{rj} = \text{currently fixed enrollment of course-section } j \text{ for course } r, \quad \forall j \in S_r, \quad \forall r \tag{26}$$

$$FE_r = \sum_{j \in S_r} FE_{rj} \quad \forall r. \tag{27}$$

One further refinement in the objective function is possible. For the purposes of balancing instructor workload, the Registrar would like to have a uniform distribution of enrollments over the various sections j of each course r. While the objective penalty induced by positive x_{ij} once the enrollment of any course j exceeds its desired level Q_j^* achieves this to an extent, we can provide a smoother, incremental penalty as the fixed enrollment assignment of a course-section begins to increase above its minimum level $Q_{j\,\min}$. Adapting an analogous notion from congestion related delays in traffic flows (see the traveltime delay functions prescribed by the Bureau of Public Roads, 1964) we introduce the following term into the objective function:

$$\sum_i \sum_{j \in C_i} \varphi_1 \left[1 + \max\left\{0, \frac{FE_{rj} - Q_{j\,\min}}{Q_{j\,\max} - Q_{j\,\min}}\right\}\right]^{\varphi_2} x_{ij} \tag{28}$$

In traffic applications, the constant φ_1 represents the base travel time on a road segment with zero volume assigned and constant φ_2 equals 4, providing a smooth, s-shaped degradation in travel time as the ratio of assigned volume to practical capacity (analogous to the ratio shown in (28)) increases. Because this is a unique application of this term, the constants φ_1 and φ_2 will be empirically determined during beta implementation testing. We will initially use parameter values in proportion to other objective function weights, in a similar manner as those determined in traffic congestion applications, and then iteratively adjust the values, until the desired smoothness is achieved.

Logical Tests

The principal purpose of these logical tests is to determine which free variables should necessarily be fixed at values of 0 or 1 based on the current fixed assignments. As indicated by Figure 3, these logical tests are performed subsequent to any variable fixing, where the fixing occurs (1) during the initial step (*a priori* variable fixing by the Registrar), after solving an LP relaxation (variable fixing based on naturally occurring $x_{ij} = 1$), and (2) after solving an IP subproblem (cadet schedule fixing). Additionally, since we also have certain soft constraints in the problem, these tests determine the set of variables that ought to be penalized in the objective function, in lieu of forcing them to be zero. The main operation we use is the following:

- If a free variable $x_{iq} = 1$ is fixed, where $q \in C_{ir}$ for $r \in T_i$,

  then $x_{ij} = 0$ for all $j \in S_r, j \neq q$,
  and $x_{ij} = 0$, for all $j \in C_i$ such that $\{j, q\} \subseteq P_m$ for some period m.

That is, if a cadet is assigned to a course-section for a desired course, we disallow an assignment to any other course-section for that course, and disallow an assignment to any other course-section that occupies the same time slot. Note that course-section q can occupy more than one time slot, as in the case of a two-hour class meeting.

The related logical tests that follow are cadet-based, course-section-based, and course-based. These are discussed below in turn, and are performed in a loop until no additional fixing or penalizing occurs in a complete pass.

Cadet-based Logical Tests

This particular test is designed to fix cadet assignments that must be made to achieve feasible scheduling.

For each cadet i:

- If the number of course-sections of course r that are available to cadet i is limited to 1 ($|C_{ir}| = 1$ for any $r \in T_i$), then fix $x_{ij} = 1$ for the corresponding $j \in C_{ir}$.
- If the number of course-sections of course r that are available to cadet i is 0 while this course has not been scheduled (that is, $|C_{ir}| = 0$ for any $r \in T_i$ while $x_{ij} \neq 1$ for any $j \in S_r$), then cadet i has a *fatal conflict* in regard to course r, and course r is temporarily removed from cadet i's requirement. Cadet i is tagged for post processing resolution.

Next, examining the Day-1 and Day-2 course load balance in (8) for each cadet i, compute two new parameters, λ_{i1} and λ_{i2}. The parameter λ_{i1} represents the number of courses $r \in T_i$ that must be scheduled on Day-1 because there are no course-sections on Day-2 available for this cadet, *i.e.*, $|C_{ir2}| = 0$, plus the fixed courses assigned on Day-1 for cadet i, and vice-versa for λ_{i2}.

Let λ_i represent the remaining free courses to be taken by cadet i that could be scheduled on Day-1 or Day-2. If $|\lambda_{i1} - \lambda_{i2}| - \lambda_i \geq \delta_{\max}$, then all the remaining λ_i courses are scheduled either on a Day-1 or Day-2 basis (as possible) depending upon which of λ_{i1} or λ_{i2} is smaller. Accordingly, we can adjust the current objective function by ascribing relatively high penalties to the x_{ij} variables for $j \in \cup_{r \in T_i} C_{ir2}$ or $\cup_{r \in T_i} C_{ir1}$, respectively. This type of penalizing is more desirable than setting these variables to zero to avoid blatant infeasibilities, given (8) is a soft constraint.

Course-based Logical Tests

The principal use of these logical tests is to make runtime global adjustments to course availability in the event that hard constraint limitations are violated.

In relation to constraint (3), if the current fixed enrollment for course-section j equals or exceeds the maximum allowed, then any further assignments of cadets to that course-section are disallowed:

- If $FE_{rj} \geq Q_{j\,\max}$, then set $x_{ij} = 0$ for all $i \ni j \in C_i$.

Furthermore, for each course r, if the total current enrollments to the course-sections $j \in S_r$ equal or exceed the expected value of the enrollment, then only allow further assignment to those course-sections which have not yet achieved the desired minimum:

- If $\Sigma_{j \in S_r} \max\{FE_{rj}, Q_{j\,\min}\} \geq |S_r| E_r$,

  then set $x_{iq} = 0$ for all $i \ni x_{iq}$ is a free variable for each course-section q for which $FE_{rq} \geq Q_{q\,\min}$.

We also introduce two runtime penalties to inhibit course-section from exceeding desired variation limits set by $q_{r\max}$. In relation to (5) and (6),

- If $FE_{rj} \geq \lfloor E_r \rfloor + q_{r\max}$ for any $j \in S_r$,

  then penalize the free variables x_{ij} in the objective function for all i.

Finally,

- If $\Sigma_{j \in S_r} \max\{FE_{rj}, \lceil E_r \rceil - q_{r\max}\} \geq |S_r| E_r$,

  then we penalize the objective function variables x_{iq} for all i such that x_{iq} is a free variable, for each course-section q for which $FE_{rq} \geq \lceil E_r \rceil - q_{r\max}$.

Because of the inherent potential for misdirecting the optimization, we insure that the penalty values incorporated into the objective function by the above routines are selected relative to the scale of the other penalty coefficients. One might also observe that if either of the last two logical tests are triggered as a strict inequality, then the current partial solution has no feasible completion with respect to (5) and (6). In this case, we relax these soft constraints by adjusting $q_{r\max}$ to a larger value such that the logical condition with the largest slack now holds as an equality. Moreover, we simultaneously increase the penalty value for the corresponding free x-variables that should then be preferably zero.

Rules and Subproblems for Fixing Variables

At any stage of the solution procedure where we have solved the residual LP relaxation and have obtained a solution $\bar{x}$, we first examine all the variables $\bar{x}_{ij}$ that are at the value 1 in the order of increasing values of cadet-indices θ_i and subsequently fix these at unit values, provided this does not create a conflict with respect to the hard constraints (1) and (2) along with the absolute maximum course-section enrollment constraint. Following this, we perform the foregoing logical tests to fix or penalize additional variables.

Next, we select a single cadet p for which $\theta_p = \min\{\theta_i$ for all i for whom a complete schedule is not yet determined$\}$. Our goal is to determine a complete schedule for this cadet p by determining corresponding values of x_{pj}, $j \in C_p$, that are as close to the respective values of $\bar{x}_{pj}$ as possible, while recognizing penalized free variables as well. Accordingly, we define:

$$\phi_{pj} = \varphi_3 \bar{x}_{pj} - \phi_{pj}^{\text{PEN}} \quad \forall j \in C_p, \tag{29}$$

where ϕ_{pj}^{PEN} is the penalized objective coefficient of x_{pj} (including the term (28) and where φ_3 is a suitable positive constant that compromises between the two terms in (29) as desired. Using this, we formulate the following integer subproblem.

(CADET IP):

$$\text{Maximize} \sum_{j \in C_p} \phi_{pj} x_{pj}$$

subject to:

$$\sum_{j \in S_r \cap C_p} x_{pj} = 1 \quad \forall r \in T_p \tag{30}$$

$$\sum_{j \in P_k \cap C_p} x_{pj} \leq 1 \quad \forall k \tag{31}$$

$$x_{pj} \text{ binary} \quad \forall j \in C_p. \tag{32}$$

Note that the complete schedule for cadet i is determined by composing the fixed variable assignments with the solution to Problem **CADET IP** for this cadet. This IP subproblem is a manageably sized, specially structured 0-1 program that can readily be solved using off-the-shelf software such as CPLEX-MIP or OSL-MIP. A maximum size of this problem (assuming an average of 4 sections per course and 8 courses) is 32 variables and 20 constraints.

If **CADET IP** is infeasible for some cadet i, then this cadet is tagged as having a fatal conflict as noted earlier, and the corresponding θ_i value is increased by a fixed large value to indicate that the conflicting schedule for this cadet will be resolved at a later time. In this case, another (untagged) cadet is selected and the corresponding problem **CADET IP** is solved. The process continues until no more untagged cadets remain.

Once **CADET IP** is solved for some cadet, thus having fixed this cadet's timetable as prescribed by the resulting solution plus the fixed variables, the logical tests described earlier are conducted as indicated in the flowchart of Figure 3, and the procedure loops back to updating the LP relaxation solution. Once a complete schedule is constructed in this fashion, where a cadet who has been tagged as having a fatal conflict is assumed tentatively to be part of a completed schedule, a check is made to see if there are any tagged cadets of the latter type. If so, then the procedure enters the clean-up routine described in the next section. Otherwise, the overall procedure is terminated and the outputs are passed to the database.

Clean-up Routine

Suppose that the solution procedure results in conflict-free schedules for a set of cadets, while some other cadets have been tagged as having "fatal conflicts." We now resolve these conflicts as follows.

First, fixing the conflict-free schedules alone and freeing all other variables and resetting related sets, we perform the proposed logical tests, except that this time, we also treat the minimum and maximum enrollment constraints as soft constraints. Here, if the first two fixed enrollment logic constraints hold true, in lieu of fixing the stated x-variables to zero, we penalize these variables in the objective function using a suitable penalty, retaining the congestion-related objective penalty term.

Next, for each cadet i having a fatal conflict, with the associated variables x_{ij} for $j \in \cup_{r \in T_i} S_r$ being free, the residual LP relaxation is solved. Now, the procedure is made to enter the loop of Figure 3, except that no fixing of variables as suggested by the LP relaxation solution x is done, and the logical tests are modified as indicated above. Hence, the problem **CADET-IP** is the only device used to fix variables. Since we are only considering the hard constraints (1) and (2) in **CADET-IP**, if it turns out that **CADET-IP** is infeasible for some cadet, then this cadet can have *no possible feasible timetable.* Such a cadet is isolated from the problem, and the output of the procedure indicates that some manual adjustment will be necessary for this cadet.

Recognizing that the final schedule thus produced could possibly violate the minimum and/or maximum course-section enrollment constraints, an additional amount of slack could be introduced into the problem by attempting to free-up the schedules of a set of cadets who have a greater degree of flexibility in their course requirements. Once this enrollment feasibility is restored, the clean-up routine can be re-run, considering the remaining schedules as fixed, in order to determine the schedules for this new set of "free" cadets that would hopefully yield an overall feasible solution. We largely suspect that if this is the case, some iteration with minor manual manipulations (following current practice) might be necessary to obtain a final acceptable schedule. This underscores the importance of implementing a solution procedure that is suitably designed for 'what if?' exploration on a reasonably equipped desktop PC. An early beta version of Model 1's implementation is currently running on a P-II 450 MHz Dell Dimension XPS with 64 Mb of RAM.

CONCLUSION

This study has presented two new large-scale integer programming formulations for the solution of the scheduling and timetabling problem at USMA. We have developed a logic-based solution strategy that attempts to reduce the computational burden of the underlying combinatorics while obtaining near-optimal timetables.

We are currently developing an alternative approach based on iteratively solving a Lagrangian dual relaxation of the integer program. This approach uses a Lagrangian dual and primal l_1-penalty function to solve various continuous relaxations of these problems, and then employs a sequential restriction heuristic which, within an iterative loop, fixes variables in batches. The resulting problem's continuous relaxation is then solved in order to dictate a fixing of additional variables until a complete assignment solution is obtained. Sherali and Brown (1994) successfully employed a variation of this approach for assigning aircraft to gates at an airport. Recognizably, although this methodology will necessarily produce much larger subproblems than the current logic-based strategy, we conjecture that the reduction in computational burden via avoiding a direct pursuit of an exact solution along with the increased flexibility afforded by a Lagrangian technique will offset this cost. Furthermore, in this type of approach, it will be possible to exploit the network sub-structure evident in the formulation of Model 2 within a Lagrangian relaxation solution process.

As an additional area for further research, we intend to examine the viability of using special structure-based Reformulation-Linearization Technique (RLT) relaxations developed by Sherali, Adams, and Driscoll (1998) for the residual programs as well as for the subproblems. At present, such a technique does not appear necessary given the size and structure of these problems. However, when modified algorithmic schemes are devised that consider subproblems enveloping several cadets, then more sophisticated strategies such as RLT might be worth imbedding into the solution methodology.

Acknowledgements

The authors wish to acknowledge the generous contribution of time and effort given by LTC Charles Goertz, Office of the Registrar, in support of the development of these models. We also thank the two anonymous referees whose comments and suggestions added to the quality of this paper. This work is dedicated to the memory of the late Dr. William J. Penny, former Assistant Dean, Operations and Registrar Division, U.S. Military Academy, West Point, New York.

REFERENCES

Badri, M.A., Davis, D.L., Davis, D.F., and J. Hollingsworth. 1998. A multi-objective course scheduling model: combining faculty preferences for courses and times, *Computers & Operations Research*, Vol 25, No 4, 303–316.

Bloomfield, S.D., and M.M. McSharry. 1979. Preferential course scheduling, *Interfaces*, Vol 9, No 4, 24–31.

Brelaz, D. 1979. New methods to color the vertices of a graph. *Communications of the Association for Computing Machinery*, Vol 22, 251–256.

Bureau of Public Roads (June, 1964) Traffic Assignment Manual, U.S. Department of Commerce, Urban Planning Division, Washington, D.C.

Burke, E.K., D.G. Elliman, and R.F. Weare. 1995. The automation of the timetabling process in higher education, *Journal of Educational Technology*, Vol. 23, No. 4, 353–362.

Carter, Michael W. 1986. A survey of practical applications of examination timetabling algorithms, *Operations Research*, Vol 34, No 2, 193–202.

Carter, M.W., G. Laporte, and S.Y. Lee. 1996. Examination timetabling: algorithmic strategies and applications, *Journal of the Operational Research Society*, Vol 47, 373–383.

Colijn, A. 1973. A sectioning algorithm, *Information*, Vol 11, No 3, 210–225.

de Warre, D. 1985. An introduction to timetabling, *European Journal of Operational Research*, Vol 19, 151–162.

Dinkel, J.J., J. Mote, and M.A. Venkataramanan. 1989. An efficient decision support system for academic course scheduling, *Operations Research*, Vol 37, No 6, 853–864.

Graves, R.L., L. Schrage, and J. Sankaran. 1993. An auction method for course registration, *Interfaces*, Vol 23, No 5, 81–92.

Hosios, A.J., and J.M. Rousseau. 1980. A heuristic scheduling algorithm, *Journal of the Operational Research Society*, Vol 31, No 8, 749–753.

Johnson, D. 1993. A database approach to course timetabling, *Journal of the Operational Research Society*, Vol 44, No 5, 425–433.

Junginger, W. 1986. Timetabling in Germany—a survey, *Interfaces*, Vol 16, No 4, 66–74.

Laporte, G., and S. Desrochers. 1986. The problem of assigning students to course sections in a large engineering school, *Computers & Operations Research*, Vol 13, No 4, 387–394.

Mathaisel, D., and C. Comm. 1991. Course and classroom scheduling: An interactive computer graphics approach, *Journal of Systems and Software*, Vol 15, No 2, 149–157.

Sampson, S.E., J.R. Freeland, and E.N. Weiss. 1995. Class scheduling to maximize participant satisfaction, *Interfaces*, Vol 25, No 3, 30–41.

Sherali, H.D., W.P. Adams, and P.J. Driscoll. 1998. Exploiting special structures in constructing a hierarchy of relaxations for 0-1 mixed integer problems, *Operations Research*, Vol 46, No 3, 396–405.

Sherali, H.D., and E.L. Brown. 1994. A quadratic partial assignment and packing model and algorithm for the airline gate assignment problem, DIMACS Series in Discrete Mathematics and Theoretical Computer Science, *American Mathematical Society*, eds. P.M. Pardalos and H. Wolcowicz, Vol 16, 343–364.

Shih, W., and J.A. Sullivan. 1977. Dynamic course scheduling for college faculty via zero-one programming, *Decision Sciences*, Vol 8, No 4, 771–791.

Tripathy, A. 1984. School timetabling—a case in large binary integer linear programming, *Management Science*, Vol 30, No 12, 1473–1489.

White, G.P. 1987. A survey of recent management science applications in higher education administration, *Interfaces*, Vol 17, No 2, 97–108.

Williams, H.P. 1998. *Model Building in Mathematical Programming*, 3rd edition. John Wiley, 20–44.

Winters, W.K. 1971. A scheduling algorithm for a computer assisted registration system, *Communications of the ACM*, Vol 14, No 3, 166–171.