Asian Journal of Information Technology 5 (12): 1479-1484, 2006

A Decision Support System for the University Timetabling Problem with Instructor Preferences

[1]Yavuz Günalay and [2]Tuna Sahin
[1]Faculty of Business Administration, Bilkent University, 06800 Ankara,Turkey
[2]Turkish Army, 06100 Ankara,Turkey

Abstract: This study describes a decision support system for the university timetabling problem with instructor preferences, which uses Goal Programming (GP) as its modeling engine. The timetabling problem has been studied by many researchers and is an NP-complete problem. We model this difficult problem as a GP formulation and use its outputs in the decision support system. The GP model helps the user to consider the objectives of different parties involved in the problem. However, this increases the computational time of the model. Therefore, an iterative decision support system is proposed for generating the best scheduling alternative. Our model is tested using data from the Turkish Military Academy and its results are discussed.

Key words: Timetabling, instructor preferences, goal programming, DSS

INTRODUCTION

Timetabling problems are an example of the types of practical scheduling problems faced by many organizations, including universities. The problem can be described as needing to schedule a given number of lectures, involving both teachers and classrooms, over a fixed period of time (typically a week), while sometimes having to satisfy a set of additional constraints of various types. The problem is known to be a hard problem and therefore many researchers have shown an interest in it since the early 1960's. Schaerf[1] showed that the problem is NP-complete and that exact optimal solution could be obtained only for small size problems (e.g., less than 10 courses)[1]. Schaerf surveyed more than 110 articles, dated between 1963 and 1999 and classified them according to both problem specifications and solution approaches. Since this survey paper, several timetabling articles have been published, with most providing new solution methods[2,3].

Although the original problem was set in the school environment[4], it now finds a wide range of application to areas from sport event timetabling to transportation timetabling studies. As in all combinatorial scheduling models, the problem grows more complex as the number of side constraints increases. The university timetabling problem that includes teacher preferences provides an example of one such variation of the problem. Badri *et. al.*[5] approached this variant of the problem using a multi-criteria objective function and, in their model, not every timeslot in the week is eligible for course scheduling. Namely, courses can only be scheduled to the time slots that are predefined and instructors can list their preferences for the various courses and particular time-slots. They solved this problem using an integer linear programming model. As computer technology has evolved, other research studies have concentrated on using automated timetabling systems, or DSS, for the problem (Sandhu, 2001[6] for a detailed survey on Information System implementation on timetabling problem). Among these studies, Dinkel *et. al.*[7] also considers instructor preferences, but they *prioritized* the preferences by using weights and solve the problem using a multi-objective decision making model.

We study the university timetabling with instructor availability (preference) constraints and provide a Decision Support System (DSS) for the planner. While the previous weighted sum of preferences approaches can unify the disparate goals of the model, the choice of weights can be very subjective and, therefore, cannot be considered consistently reliable. Instead, we solve the problem with respect to the administration's preference listing to avoid the use of any subjective weightings. In our approach, the instructor work loads receive the highest priority, followed by the instructor preferences for the time-slots.

This approach differs from previous studies in the following two aspects: i) a multi-criteria decision making model is used as the optimization tool and ii) the solution is obtained in an iterative structure with the direct involvement of the decision maker. Although each of these methods have been individually implemented by

Corresponding Author: Yavuz Gunalay, Faculty of Business Administration, Bilkent University, 06800 Ankara, Turkey

several researchers, our model is the first one which simultaneously combines a DSS approach with the goal programming optimization tool in order to process the instructor teaching time preferences-which are the hard constraints of the problem. The model will be explained in more detail in the next section.

The paper is organized in the following fashion. Firstly, we describe the problem and its corresponding Integer Linear Programming (ILP) formulation. Secondly, the DSS and the iterative/interactive approach for solving the problem are fully explained. Thirdly, the approach is tested by using the DSS on a specific implementation of university case data from the Turkish Military Academy. The last section provides a conclusion and suggests directions for future extensions to the study.

PROBLEM DESCRIPTION

The general timetabling problem requires both the scheduling of instructors to courses and the assigning of classrooms and time slots for each course. Classroom capacity restrictions, together with teaching load constraints, make this a very hard problem to solve. In our problem we must also include additional complicating constraints that relate to instructor time slot preferences. The specific university considered is the Turkish Military Academy (TMA), which, in addition to regular full-time faculty, also employs numerous additional part-time instructors who possess very rigid time constraints. The higher ranking full-time instructors also must perform supplementary administrative workloads which can serve to significantly limit their time availability for teaching. Furthermore, the different instructors each have different teaching loads. Finally, the weekly schedule for the students is full. That is, the number of available time slots per week is exactly equal to the course load for each of the students. Although this high utilization (100%) of classrooms can be used to reduce the need for classroom assignment in the problem, it simultaneously serves to increase the complexity of the instructor scheduling problem. The timetabling problem of TMA administration can be summarized as:

- To schedule every course of the curriculum to each class with respect to its timing requirements;
- To assign an instructor for each section of every lecture;
- To make sure that there exist no conflict in the instructors' schedules;
- To pay attention to instructor teaching loads;
- To consider teaching preferences of each instructor.

TMA has four classes and each class has four sections. Although no student is allowed to take courses from two different classes simultaneously and each class has its own classroom, the problem is not decomposable into four pieces, due to the instructor resource. Students must take 8 different courses per semester. Courses have different lecture hours-two, three or four h and the faculty size for each course varies from three to nine (total faculty size is forty-nine).

The number of variables and constraints are enough to classify the problem as a hard problem. Moreover the time availability (preferences) constraints of part-time instructors further complicate the problem. Recall that all the time slots of the weekly schedule have to be filled in our problem. Therefore when the same time slot is considered undesirable by more than one instructor, a problem solution can quickly become infeasible. In its current format, the problem is solved manually and requires more than two months of work.

Due to the soft constraints like teaching load and time preferences, we approach the problem as a GP model and try to minimize the deviations from the best practices for these soft constraints. GP is indeed a special form of ILP at which certain constraints are considered as decision maker's goal/aim list, e.g., for our problem soft constraints are such goals and instead of forcing these constraints, we choose to minimize deviations from these goals as the objective. In a survey article, Tamiz *et al.*[8] explains GP techniques and recent developments.

With such a large problem, we notice that any single run of a GP algorithm would generally not produce a satisfactory solution. Consequently, we suggest a Decision Support System approach (DSS) which iteratively solves and improves the original GP model to achieve the best overall scheduling result.

THE MODEL

The university timetabling problem that we consider contains N instructors, M courses to offer and O time slots within which to schedule the courses. Our model differs from a standard university timetabling problem due to the fact that each class has a fixed size and has its own classroom. Therefore, there are no "room" constraints in our model. Also the timetable for a class is divided into fixed size time-slots. Besides these simplifying features the problem possesses the following extra constraints:

- Specific teaching load to be fulfilled by each instructor;
- Different number of time slots available each day;
- Time-slot priority listing for each instructor;

Table 1. List of system parameters and decision variables

System parameters:	
i	Instructor index, i = 1,.., N.
j	Course index, j = 1,.., M.
k	Time slot index, k = 1,..,O.
g	Available number of time slots per day.
s_j	Number of time slots course j requires.
tl(i)	Teaching load of instructor i.
V	Set of highest priority instructors.
Y	Set of second priority instructors.
Z	Set of lowest priority instructors.
Q_i	Set of undesired time slots for instructor I.
Decision variables:	
x_{ijk}	Binary variable for instructor i is teaching course j at time slot k.
I_{ij}	Indicator variable for teacher i offering course j.
d^+_i, d^-_i	Deviation from instructor i's teaching load.
e^+_i, f^+_i, h^+_i	Number of undesired assignments for instructor i, respectively for i∈V, Y and Z.

where the first and last constraints can easily make the problem infeasible. Therefore, a goal programming approach would aim to satisfy each constraint as much as possible. The instructors are divided into the three mutually exclusive and collectively exhaustive sets, *V, Y* and *Z*, according to the importance level of the instructor's teaching preference. The model notation is shown in the following Table 1.

As the teaching load constraints and instructor preferences constraints become more rigid the ILP formulation of this problem has higher chance of being infeasible. Therefore, we model the problem as a priority Goal Programming (GP) which uses each of these four constraints as targets. Satisfying the teaching load constraint is considered to be the highest priority among the identified targets, followed by the instructor preferences stated within each instructor's importance hierarchy. The following is a four-level priority GP formulation of the problem.

$$\text{Min.z} = P_1\left(\sum_{i=1}^{N}(d_i^- + d_i^+)\right) + P_2\left(\sum_{i\in V} e_i^+\right) + P_3\left(\sum_{i\in Y} f_i^+\right) + P_4\left(\sum_{i\in Z} h_i^+\right)$$

$$\sum_{i=1}^{N} I_{ij} = 1 \quad j=1,\ldots,M. \tag{1}$$

$$\sum_{k=1}^{O} x_{ijk} = s_j I_{ij} \quad \text{for all } i, j. \tag{2}$$

$$\sum_{j=1}^{M} x_{ijk} \le 1 \quad \text{for all } i,k. \tag{3}$$

$$\sum_{k=1}^{g} x_{ijk(t*g+k)} \le 2 \quad \text{for all } i, j \text{ and } t = 0,\ldots.4. \tag{4}$$

$$\sum_{k=1}^{g/2} x_{ijk(t*g+2*k-1)} \le 1 \quad \text{for all } i, j \text{ and } t = 0,\ldots.4. \tag{5}$$

$$\sum_{k=1}^{g/2} x_{ijk(t*g+2*k)} \le 1 \quad \text{for all } i, j \text{ and } t = 0,\ldots.4. \tag{6}$$

$$\sum_{j=1}^{M}\sum_{k=1}^{O} x_{ijk} + d_i^- - d_i^+ = tl(i) \; i=1,\ldots..,N. \tag{7}$$

$$\sum_{j=1}^{M}\sum_{k\in Q_i} x_{ijk} - e_i^+ = 0 \quad i\in V. \tag{8}$$

$$\sum_{j=1}^{M}\sum_{k\in Q_i} x_{ijk} - f_i^+ = 0 \quad i\in V. \tag{9}$$

$$\sum_{j=1}^{M}\sum_{k\in Q_i} x_{ijk} - h_i^+ = 0 \quad i\in Z. \tag{10}$$

In this GP model, the first and second constraints make sure that each course is offered by one instructor and is scheduled exactly in the predetermined number of time slots. Constraint (3) ensures that an instructor is teaching no more than one course in any given time slot. Constraint (4) enforces a requirement that each course can be scheduled on any day of the week, but in a day no more than two hours of the same course can be offered. Furthermore, constraints (5) and (6) require these two hours have to follow each other. Constraint (7) calculates deviations from course load assignments for each instructor. Deviations from each instructor's preferences are calculated in constraints (8), (9) and (10), with respect to their priority group. Detailed discussions of these constraints can be found in Sahin[9].

DECISION SUPPORT SYSTEM

TUTPIP, the decision support system suggested for the university timetabling problem with instructor preferences, has two main parts: a relational database and an optimization package. The first part is responsible for data entry, the storing of past course schedules and report generation. It also acts as the user interface. In our system, MS Access is used for this part and the data is divided into three databases: Old_A.mdb, Course.mdb and Instructor.mdb. The first one offers the latest timetable used for a similar period of time (e.g., semester); whereas the later two databases provide the problem requirements for the current scheduling period.

The second part of TUTPIP is the integer linear programming module to optimize the GP model of the

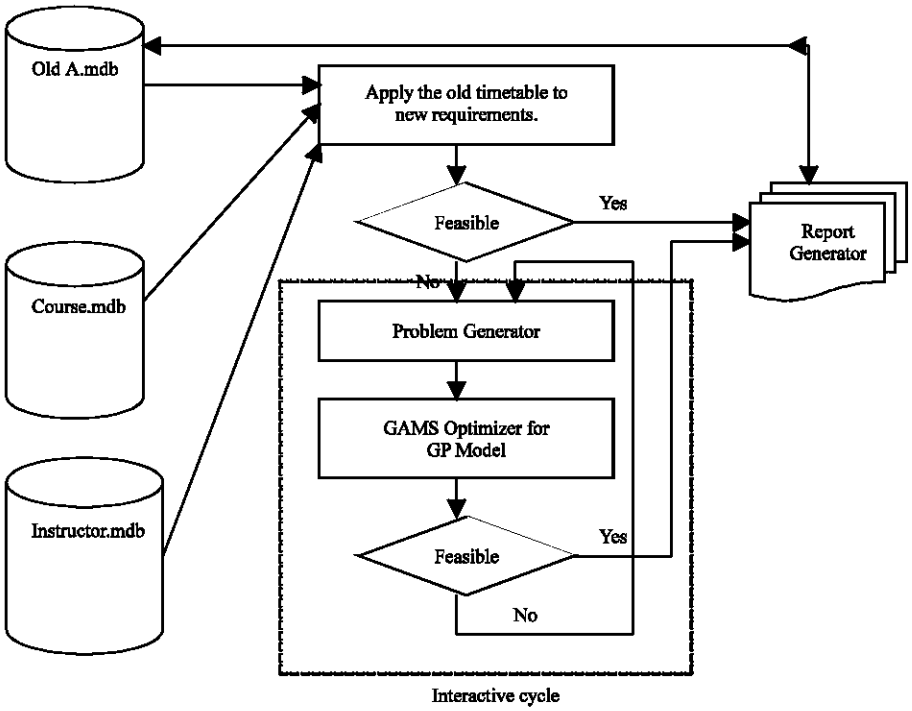

Fig. 1: Entity-Relationship Diagram of the timetabling DSS

system. We use GAMS[10] as the ILP optimizer; but even Excel Solver can be used for this portion if less computational power is required. The Entity Relationship Diagram (ERD) of the system is illustrated in Fig. 1.

The first step of the system is input stage. The system applies the old assignment exactly, if it is still feasible. Otherwise, *Problem Generator* constructs an ILP formulation of the problem based on the limitations provided during the input stage. The second stage of TUTPIP is called *interactive cycle*, because this ILP formulation and solution steps can be implemented repeatedly if the problem size is big. At this stage, the Decision Maker (DM) analyzes the model and partitions it into manageable subsystems, if the problem size is too large and constraints show a separable structure. After all such subsystems iteratively and interactively are solved using GAMS, at the final round the model of the whole problem is regenerated using the current assignments as constraints and solved to get the final timetable. This final schedule is fed to the *Report Generator* module.

At the last stage, the *Report Generator* module creates reports using the final timetable schedule, as well as the old-assignments database. First the timetable of each student section and each instructor are created. These timetables are reviewed carefully before finalizing them. Output can contain undesirable course assignments (e.g., an assignment of two hours late on Monday and two hours early on Tuesday) or impossible instructor assignments (e.g., a time-slot assignment for an instructor at which the instructor is teaching at another organization, since this is shown as a non-preferable time, not an impossible time). In such cases, necessary modifications in the constraints set of the *final model* are done and it is feed-back to stage 2. This feed-back loop can be repeated until the decision maker is fully satisfied with all the course and instructor assignments. The module is also responsible to generate reports on deviations from the instructor load and preferences. Combining these information with the past data, instructor satisfaction graphs can be plotted. DM can use these reports in the future instructor priority class assignments.

Although TUTPIP keeps only the latest timetable as the old-assignment, generating a Knowledge-Base by storing all previous timetables with the corresponding constraint sets is also possible. Later, this knowledge-base can be used to improve the scheduling process at the first stage and the efficiency of the DSS can be increased.[6]

COMPUTATIONAL RESULTS

The DSS was implemented for the course-instructor scheduling process in the Turkish Military Academy (TMA). For simplicity this study describes its application

Table 1: Comparison of goal achievements at different stages of the DSS algorithm

	Preference violations				
	1st Priority group	2nd Priority group	3rd Priority group	Total	Instructor overload
Solution of 4 subsystems individually	11	9	3	23	14
After Final Optimization	1	7	5	13	1

only to first year students who must take eight different courses during a thirty hour week (the application to all students is described in detail in Sahin[9]). First year students are divided into twenty-nine sections. There are forty-nine instructors for first year courses and some of these are part-time instructors with very limited teaching time availability. These limitations are placed into the hierarchy of 3 teaching preference sets and due to resource limitations, part-time instructors are assigned to the highest priority set. Each instructor has a different course load, but under no conditions can their weekly load exceed twenty hours. Since Old_A.mdb is empty we start the DSS at the second stage. In its original format the problem size was huge, containing 36540 decision variables and 9207 constraints. Consequently, we divided the problem into four sub-problems, with each sub-problem containing a similar number of course-section combinations and teachers.

Without loss of generality, each section of a course can, itself, be defined as a different course. Therefore each subproblem has a total of either 56 or 64 different courses to be scheduled, with twelve instructors available to teach them. Each of these sub-problems can be solved in a reasonable time using GAMS software. The result of each subsystem is analyzed by the DSS and evaluated for feasibility. Whenever infeasible schedules occur, the partitioning sets of the instructor preference must be iteratively updated by the user of the DSS. Finally, the results of all of the subsystems are concatenated and input to the GAMS solver for final optimization. The following table shows an example that compares the output of the model with respect to undesirable outcomes (Table 2).

This problem was solved in six hours and, with only 13 preference violations (mostly from Group 2) and 1 overloaded instructor, the final solution can be considered as a very good timetable by the TMA. When compared to the 2 month solution time required by the existing the manual system, this solution approach can be considered as a significant advance by providing an acceptably good solution in a very short period of time. Furthermore, TUTPIP is an easy tool to use, since its user interface build on a MS Office Suite product. It has also a weakness, which is the use of GAMS as the optimizer package. Although GAMS is very efficient optimizing software, its communication with problem generator has not been automated. To automate this link GAMS has to be switched by Excel Solver, which also a part of MS Office Suite.

CONCLUSION

The university timetabling problem with instructor preference constraints is a very difficult combinatorial problem that is generally solved using heuristics solution approaches. In this study, we have provided a DSS approach which can be used to quickly guide the decision maker toward good solutions. The DSS uses a goal programming model as its search engine and assists the decision maker in the preparation of an acceptably good, final timetable. This system has been used to reduce the preparation time of the timetable in the TMA from months to a matter of hours. The DSS can also be used to generate several reports that illustrated the efficiency of the schedules actually produced. In particular, these reports can provide information on the percentage of overloaded instructors and the number of times an instructor preference is violated.

In the study, this DSS has been tailored to the specific TMA problem. However, by performing simple modifications, the model and the DSS can be easily adapted to numerous other representations of the timetabling problem. Such applications will be studied in future work.

REFERENCES

1. Schaerf, A., 1999. A survey of automated timetabling. Artificial Intelligence Rev., 13: 87-127.
2. Burke, E.K. and S. Petrovic, 2002. Recent research directions in automated timetabling, European J. Oper. Res., 140: 266-280.
3. Smith, K.A., D. Abramson and D. Duke, 2003. Hopfield neural networks for timetabling: formulations, methods and comparative results, Computers and Industrial Engin., 44: 283-305.
4. Gotlieb, C.C., 1963. The construction of class-teacher timetables, Preceedings of IFIP Congress 62, (Edn.) Popplewell, C.M., pp: 73-77.
5. Badri, M.A., D.L. Davis, D.F. Davis and J. Hollingsworth, 1998. A multi-objective course scheduling model: combining faculty preferences for courses and times. Computers and Operations Res., 25: 303-316.

6. Kanoh, H. and Y. Sakamoto, 2004. Interactive timetabling system using knowledge-based genetic algorithms, IEEE International Conference on System, Man and Cybernetics (SMC'2004), pp: 5852-5857.
7. Dinkel, J.J., J. Mote and M.A. Venkataramanan, 1989. An efficient decision support system for academic course scheduling, Operations Res., 37: 853-864.
8. Tamiz, M., D. Jones and C. Romero, 1998. Goal programming for decision making: An overview of the current state-of-the-art. European J. Oper. Res., 111: 569-581.
9. Sahin, T., 2004. Goal programming approach to solve the timetabling problem at Turkish Military Academy, MBA thesis, Faculty of Business Administration, Bilkent University, Turkey.
10. Brooke, A., D. Kendrick, A. Meeraus and R. Raman, 1992. GAMS A user's guide, GAMS Development Corporation, Washington, D.C.