Fortnightly course scheduling problem: a case study

Mohammad Ranjbar
Department of Industrial Engineering, Faculty of Engineering,
Ferdowsi University of Mashhad
Mashhad, Iran
m_ranjbar@um.ac.ir

Salim Rostami
Department of Industrial Engineering, Faculty of Engineering,
Ferdowsi University of Mashhad
Mashhad, Iran
Salim.Rostami@stu-mail.um.ac.ir

Abstract

This paper studies a fortnightly university course timetabling problem. In this research, the Industrial Engineering Department of Ferdowsi University of Mashhad (Iran) is considered as the case study for investigation. An integer linear programming model is developed for the problem and is solved using ILOG CPLEX 12.1 solver. Also, a branch-and-bound algorithm is developed to find optimal solutions in which both lecturers and students' preferences are considered. Regarding to the limited size of the case study, the computational results indicate that the developed algorithm is efficient and is able to find noticeable solutions in a reasonable time.

1. INTRODUCTION

This study focuses on a particular field of scheduling problems known as university course timetabling problem which has been proved to be a NP-problem [1]. The general term of university timetabling refers to both exam and course timetabling [4]. In exam timetabling one of the goals is to spread the different exams to the best possible extent for each individual student, while in course timetabling the students want an as compact and uniform timetable as possible. Also, recurring timetables (weekly, fortnightly, etc.) are appreciated for course timetabling. The university course timetabling is the procedure of assigning lectures, which are presented by lecturers, into *room-timeslot*s subject to a given set of constraints. Constraints are usually classified into two categories: hard constraints and soft constraints. An assignment that satisfies all hard constraints is called a *feasible* timetable. The objective of the each timetabling problem is to minimize the number of soft constraints violations in a feasible timetable.

A large variety of heuristic algorithms based on Metaheuristics have been applied to course timetabling problems. Examples of these algorithms include genetic algorithm [7], simulated annealing [3], Tabu search [5], particle swarm optimization [6] and so on.

In contrast of heuristic algorithms, the literature of exact solution approaches in the context of timetabling is very limited because these approaches cannot tackle large scale timetabling problems. Integer programming approach is one of the most commonly used approaches for the course time tabling problem (see, for instance, [2]). In this paper, we study the fortnightly university course timetabling problem (FUCTP) in Ferdowsi University of Mashhad. In this problem, each lecture of a course is presented in a 120 minutes timeslot where 100 minutes of that is used for teaching and other 20 minutes for relaxing. Each course may need 1 or 1.5 lectures per week. Since it is not possible to assign half of a timeslot to a lecture, the total frequency period of the timetable is set to two weeks in which each course may need 2 or 3 lectures. Also, each semester includes 16 weeks. In contrast of weekly timetables in which course timetable is identical for all weeks, in fortnightly timetables, we have odd and even weeks. For those course that need 2 lectures per two-weeks, their corresponding timetables in odd and even weeks are identical. In contrast, for the courses with 3 lectures per two-weeks, the timetable for odd and even weeks are different such that each course has a fixed lecture per week and an alternate lecture per two weeks (either in odd or in even weeks). The faculty of Engineering of Ferdowsi University includes seven departments where each department has a given number of identical classes (identical capacity and equipment). Also, each department can share its classes with other departments where a central education office coordinates the sharing. Thus, it is assumed that for each department, a minimum number of identical classes are available in each timeslot but it may be increased in various amounts in different timeslots. In this paper, we focus on developing an optimal timetable for the department of industrial Engineering which includes only undergraduate program. For this department, at least two identical classes are available for each timeslot.

A branch-and-bound (B&B) algorithm is developed to find solutions which strictly satisfy hard constraints and minimizes the number of soft constraint violations. Branch-and-bound is a general algorithm for finding optimal solutions of various optimization problems, especially in discrete and combinatorial optimization.

2. PROBLEM SESCRIPTION AND FORMULATION

We define the FUCTP as follows: there are *n* courses gathered in set *C* which is divided into two subsets C^2 and C^3 in terms of number of required lectures per two-weeks. Set $C^2 = \{c_i : l_i = 2\}$ and $C^3 = \{c_i : l_i = 3\}$ where l_i indicates the number of required lectures per two-weeks for course c_i. These two subsets are mutually exclusive and jointly exhaustive. In another point of view, we divide all courses into *k* families where each family F_f indicates a set of courses that are usually attended by a group of students with identical entrance year. Also, the families are mutually exclusive $\left(\forall f, f' : F_f \cap F_{f'} = \emptyset\right)$ and jointly exhaustive$\left(\bigcup_f F_f = C\right)$. All families are determined based on the curriculum of the Ferdowsi University. It should be noticed that it is possible that a student take courses from different families. There are *m* professors gathered in set *P* and each professor p_j must present a subset of determined courses shown byPC_j. Each lecture of a course should be assigned to a timeslot ts_t where *t*=1,…,60. Number of timeslots has been determined based on the six timeslots in a working day ([8-10], [10-12], [12-14], [14-16], [16-18] and [18-20]) and ten working days in a two- week period. The union of timeslots $ts_{6(d-1)+1}, \dots, ts_{6d}$ constitute the day D_d.

It should be considered that each lecture of a course $c_i \in PC_j$ can be assigned to a timeslot of set PTS_j, determined by professor p_j. In order to determine the optimal timeslot for each lecture of a course, we consider preferences of students and professors, shown by sets *STSP* and *PTSP*, respectively. As hard constraints, the courses of a family should not overlap but there may be courses from different

families which are attended by a noticeable number of students. Thus, we consider a cost, shown by$oc_{ii'}$, for overlapping of courses $c_i \in F_f$ and $c_{i'} \in F_{f'}$ where $f \neq f'$. The overlapping costs are determined based on the students' enrollment data. Also, in order to have a compact program for both students and professors, we consider a cost for each busy day, a day includes at least a lecture, shown by *cbs* and *cbp* for students and professors, respectively. On the other hand, for construction a uniform timetable we consider an upper bound for maximum number of lectures in a day participated by a student (ls^{max}) or presented by a professor (lp^{max}). Since some courses are presented by professors coming from outside of the department, we have assumed that the timetable of these courses is assigned already. The parameters pa_{it} , pa_{it}^1 and pa_{it}^2 are defined to show preassignments.

3. B&B ALGORITHM

The branching strategy of our developed B&B algorithm for the FUCTP is as follows. Each node of the search tree corresponds to a partial schedule in which corresponding timeslots of some courses are determined. Each schedule **S** is represented by a vector $\mathbf{S} = (CT_1, \dots, CT_n)$ where course timetable CT_i is an ordered pair (θ_i^1, θ_i^2) if $l_i = 2$ and an ordered triple $(\theta_i^1, \theta_i^2, \theta_i^3)$ if $l_i = 3$ in which $\theta_i^1 \in \{t | t = 1, \dots, 30\}, \theta_i^2 = \theta_i^1 + 30$ and $\theta_i^3 \in \{t | t = 1, \dots, 60\}$. In each node of the search tree, a CT_i is determined. Thus, the set offspring emanated from a node includes all possible values for $\theta_i^1$and θ_i^3.

In order to determine which course must be considered in which level of the search tree, we sort courses based on the non-decreasing order of their flexibility degree (*FD*). The flexibility degree of each course $c_i \in C^2$ equals to the number of timeslots can be assigned to the only lecture of this course in a week, determined by the $PC_j$where $c_i \in PC_j$. Also, for each course $c_i \in C^3$, we define the flexibility degree as the average of two following values: number of possible timeslots for the fixed lecture in a week and number of possible timeslots for the alternative lecture in two weeks. Since pre-assigned courses have zero flexibility degree, we assume the lectures of these courses are assigned to the corresponding timeslots before starting the B&B algorithm. In each node, after the assignment of a course timetable, flexibility degrees of all courses are updated. Assume a lecture of course c_i is assigned to the timeslot ts_t. In updating procedure, we remove ts_t from possible timeslots of lecture $c_{i'}$ if one of the following conditions is satisfied: I) Courses c_i and $c_{i'}$ have an identical family; II) Courses c_i and $c_{i'}$ have an identical professor; III) The number lectures assigned to ts_t equals to cts_t. Without loss of generality, we assume $FD_1 \leq \cdots \leq FD_n$. Completion of a partial schedule based on the nondecreasing order of courses' flexibility degree prevents expansion of the search tree using reduction of creation infeasible solutions' chance.

The first found feasible solution is consider as a lower bound (*LB*) and is updated whenever a better solution is found. Also, regardless to the hard constraints, we assign each unscheduled course c_i to the timeslots results in the maximum value of $(s_{it} + p_{it})$to construct an upper bound (*UB*). Now, these values are added to the objective function value of the partial schedule, calculated based on (1) based on scheduled courses. The following dominance rule is used to obstruct nodes result in infeasible or low quality solutions.

Dominance rule 1. In each node *N*, if *UB*≤*LB*, obstruct the node.

Dominance rule 2. Assume a course timetable is assigned in node *N* and flexibility degrees are updated for all unscheduled courses. Now, if for every two different courses c_i and $c_{i'}$ we have $FD_i = FD_{i'} = 1$, there is a common timeslot among their possible timeslots and one of the following conditions is satisfied, the node *N* must be obstructed.

I) c_i and $c_{i'}$ belongs to an identical family.

II) c_i and $c_{i'}$ must be presented by a professor.

4. CONCLUSIONS

In this paper we study a case study of the FCUTP. We developed an integer linear programming model and solved it using ILOG CPLEX 12.1. Also, we developed a B&B to find optimal solution. The primary experimental results indicate the developed B&B algorithm is more efficient than CPLEX and can find near optimal solutions in a very short time.

REFERENCES

[1] V.A. Bardadym, *Computer-aided school and university timetabling: The new wave*, In E. Burke & P. Ross (Eds.), Practice and theory of automated timetabling, Lecture notes in computer science,vol.1153, pp. 22–45, Berlin: Springer, 1996.

[2] N. Boland, B.D. Hughes, L.T.G. Merlot,P.J. Stuckey,*New integer linear programming approaches for course timetabling*, Computers & Operations Research, vol. 35, pp. 2209-2233, 2008.

[3] S. Ceschia, L. Di Gaspero, A. Schaef, *Design, engineering, and experimental analysis of a simulated annealing approach to the post-enrolment course timetabling problem*, Computers & Operations Research, vol. 39, pp. 1615-1624, 2010.

[4] R. Lewis, *A survey of metaheuristic-based techniques for university timetabling problems*, OR Spectrum, vol. 30, pp. 167–190, 2008.

[5] Z. Lü, J.K. Hao, *Adaptive Tabu Search for course timetabling*, European Journal of Operational Research, vol. 200, pp. 235-244, 2010.

[6] D.F. Shiau, *A hybrid particle swarm optimization for a university course scheduling problem with flexible preferences*, Expert Systems with Applications, vol. 38(1), pp. 235-248, 2011.

[7] Y.Z. Wang, *Using genetic algorithm methods to solve course scheduling problems*, Expert Systems with Applications, vol. 25, pp. 39–50, 2003.