Class Scheduling Optimization

Janewit Nakasuwan, Piyawadee Srithip and Somrote Komolavanij
Department of Industrial Engineering, Sirindhorn International Institute of Technology, Thammasat University, Pathumthani 12121, Thailand

Abstract

Regarding the rapid growth of the number of students and the increasing number of courses offered in universities and colleges around the world, the task of scheduling classes to fit into timetables and into existing facilities is becoming much more complicated. At the present time, class scheduling not only needs to fit the courses offered but also has to be performed based on many factors, such as availability and capacity of the room, cost occurred when the rooms are engaged by any courses, losses occurred when the rooms are left out, etc. With the approach of Operations Research, linear programming, which is directly correlated with the concept of assignment problems, appears as an efficient method to cope with all the considerations. Due to the wide variety of scheduling, the scope of the study is narrowed to the class scheduling at Sirindhorn International Institute of Technology.

1. Introduction

Class scheduling, one of the most difficult problems facing all colleges and universities all over the world, has been discussed over years in order to utilize existing facilities and resources effectively and economically. Many computerized scheduling softwares are not economically applicable to some institutes. Because they might require high performance personal computers or even work stations. Anyway, many colleges, especially the overpopulated ones with limited resources, still require high potential tools to deal with the scheduling problems. The term "class scheduling" means not only to fit or assign all courses to meet different time intervals but it also includes the optimization of the available facilities in terms of operating cost. Class scheduling processes are normally effected by many related constraints that might cause never-ending complaints from instructors and students if those were not satisfied.

The development of course schedules and classroom assignment in universities is not an easy task. The problem is even more tough and complicated especially when dealing with the following restrictions.

- ❑ Class periods are not all of the same length.
- ❑ Two courses cannot be held at the same time if at least one student is registered in both courses or if both of the courses engage the same lecturer.
- ❑ A lecturer may not be available or may prefer to give the class at a certain time of the week.
- ❑ Courses taken by a large number of students have to be repeated several times during the week.

Hence, the grouping of students into course sections needs to be carried out so as to create a timetable with as few conflicts as possible. By comparison, in many universities, the room assignment is essentially unchanged from one year to the next. When the number of courses is discounted, many rooms become free. When a new course is offered, they look for the best available room at that time. Consequently, without good management, people frequently must accept rooms that do not meet their requirements, and the problem is compounded over and over.

There are many previous studies related to the timetabling problem. As already mentioned, class periods are not all the same length and two classes could not be held simultaneously. However, students should be allowed to follow their selections of courses as much as possible. The timetable should also be established according to instructors and classroom availability. One of the characteristics of the timetabling problem is its large size. This becomes a great hindrance solving timetabling problems using mathematical programming techniques. Tripathy (1984) proposed a two-phased approach in which courses and rooms are grouped in the first phase and times are selected in the second phase. Lagrangean relaxation was used to solve smaller problems obtained by grouping. The method also incorporates a branch and bound procedure which takes advantages of special ordered set of variables.

Due to the propensity of faculty and students in selecting classes in the prime period (9 A.M. – 12 and 1 P.M. – 3 P.M.) to the exclusion of alternative time slots, Mulvey (1982) developed a multiple-assignment scheduling approach based on the use of a general network model. The problem was modeled as the assignment of instructor-student classes to classroom-time slots. The network model was an attractive compromise between model complexity, informational requirements, computational costs and understandability. The approach focused on designing an interactive human interface to limit the size of the problem encoded for solution. Dinkel, Mote and Vekataramanan (1989) described a network-based decision support system approach based on Mulvey's model involving 175 faculties, over 300 sections, 20 rooms and 16 time-slots. The proposed scheduling model was a capacitated, pure network flow problem with a penalty structure in the objective function. The solution was accomplished with a commercial integer programming code.

In determining a class period schedule and specifying a feasible classroom assignment, Ferland and Roy (1985) defined problems as a problem of assigning conflicting activities to resources, for example, quadratic programming problem with 0-1 variables by introducing the complicating constraints into the objective function via penalties. Aubin and Ferland (1989) formulated timetabling subproblem and grouping subproblem as assignment type problems in which entities were assigned to resources. They first generated an initial timetable which assigned the students to the course section. Then, an iterative descent procedure was used, which adjusted the timetable and grouping successively until no more improvement of the objective function could be obtained.

Hertz (1991) developed two heuristic procedures based on Tabu search techniques to handle grouping and timetabling subproblems. The Tabu search technique was especially designed for avoiding the solution from being trapped in local minimum solution and is generally much more powerful than descent methods. More additional constraints were brought in, for example the length of the courses may not be uniform, and the course constraints, which involve courses with a large number of students have to be repeated several times during the week.

Mooney, Darden and Parameter (1996) developed and implemented the approach called CHRONOS, which was 'what-if' modeling and is able to search capabilities to support the decentralized scheduling process used at Purdue University, to solve similar problems. Such problems were included in the long-term planning, this resulted in preliminary schedules based on incomplete information and the difficulty of manual revision limited the number of possible schedule improvement iterations. The approach relied on maximization of student and instructor preferences without creating student, room or instructor schedule conflicts. This acceptable approach was used to implement the system in a client-server environment and improve the qualitative aspects of generated schedules.

Carter and Covey (1992) resolved the confusion, as to how difficult the timetabling is, by identifying cases when the problem is easy and when it is difficult. It turns out that the difficulty depends not only on the objective and size of the problem, but also on the underlying preferences of classes for rooms.

2. Problem Declaration

The problem in classical course timetabling is the reduction of the number of conflicts where courses are taking place simultaneously. For large institutions such as universities, the problem becomes more difficult since more additional constraints have to be taken in to account. For example, the length of courses may not be uniform, teachers may not be available or may prefer to teach at certain times of the week. Moreover, some courses may be optional courses and it is necessary to respect the freedom of choice of the students. There might be some pairs of courses, which have to be scheduled on the same day. Instructors may prefer to teach in consecutive time periods of three hours rather than two time periods of one and a half hours. The most difficult constraints are certainly due to courses involving a large number of students and thus have to be repeated several times during the week. This problem is strongly interrelated to the timetabling subproblem, which also considers many other constraints. In addition, because of pressures on room utilization, there may be a minimum size requirement; for example, a course section should have an enrollment of at least as close to the room capacity which it is scheduled as possible. This avoids the problems associated with scheduling a course section of twenty into a room seating one hundred in a place where space is the most important constraint. Such constraints complicate the situation since they require a large-scale binary integer-programming model in order to deal with the most general setting. In a resource-constrained environment, with great pressure on the rooms, this is an important tradeoff that must be carefully analyzed.

To illustrate the problem and approach in this study, the scheduling administration of the Sirindhorn International Institute of Technology (SIIT), Thammasat University, Thailand, has been selected as a focus of the study. There are around one thousand full-time undergraduate students in 5 departments and 17 classrooms of various sizes with a computer laboratory, an audio and visual room, and an up-coming new building. As a result of the increasing number of students, there is a tremendous pressure on the available class-room space. Prior to the adoption of the computerized scheduling approach, the administrative personnel and department secretaries made the assignment manually (See the process diagram in Figure 1).

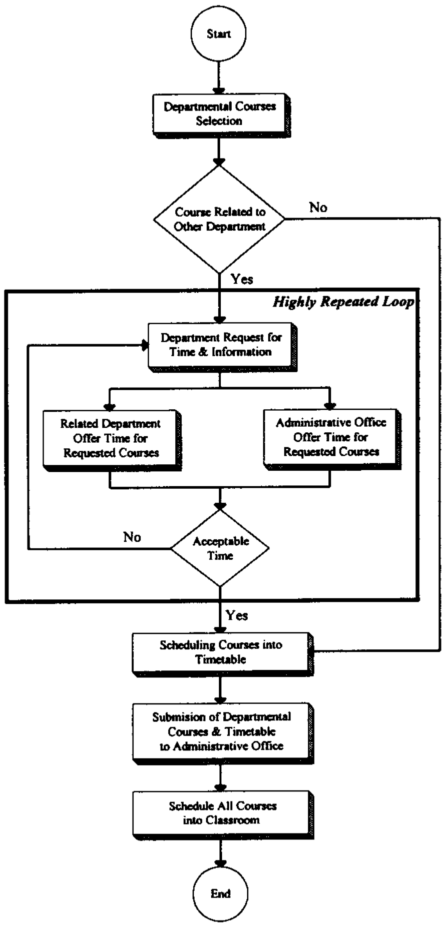
Start
Departmental Courses Selection
Course Related to Other Department
No
Yes
Highly Repeated Loop
Department Request for Time & Information
Related Department Offer Time for Requested Courses
Administrative Office Offer Time for Requested Courses
No
Acceptable Time
Yes
Scheduling Courses into Timetable
Submision of Departmental Courses & Timetable to Administrative Office
Schedule All Courses into Classroom
End

Figure 1: General Scheduling Process

The ***Highly Repeated Loop*** box contains the activities that were normally repeated during the scheduling process of SIIT. Documents were sent repeatedly back and forth between one department to another and to the administrative office. It caused delay for the institute to hand out the class schedules. Due to some specific constraints, such as instructors' constraints, class capacity constraints, time constraints, etc., there were many rooms left during weekdays, and some classes needed to be held at weekends, which involved extra operational expenses. Therefore, it would be extremely important to find effective tools or techniques that would be able to handle this situation.

3. Scope and Purpose

According to the above mentioned considerations, the development of the model, which is based on the concept of assignment problems in the Operations Research, was initiated. The objective of this study is the development of a simplified model by formulating linear programming equations, which satisfy all related constraints. The model has to be able to reduce the number of activities in the scheduling process. (See Figure 2 for the expected process diagram)

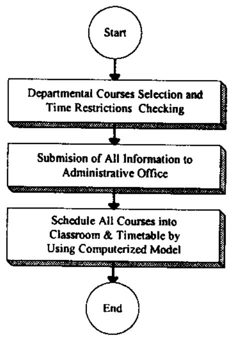
Start
Departmental Courses Selection and Time Restrictions Checking
Submision of All Information to Administrative Office
Schedule All Courses into Classroom & Timetable by Using Computerized Model
End

Figure 2: Expected Scheduling Process

The contribution of this study is the implementation of method analysis and linear programming software to solve problems of class scheduling in colleges and universities. Due to the wide variety of class scheduling problems, the study and analysis of the present class scheduling method of SIIT had been narrowed down to be the focus of the study. Problems will be simplified into a linear programming model, which is based on the basis of cost minimization. The formulated model will be subjected to restrictive functions or constraints i.e. time constraints, instructor's constraints, facilities constraints, etc. The knowledge in Operations Research together with linear programming software would enable the problems, which normally occur in the scheduling processes, to be solved.

This study is related to scheduling courses into rooms and then times, which are very tough problems for SIIT. This study is expected to solve those problems and improve the qualitative aspects of generating schedules as follows:

- ❑ Shortening the time in establishing the timetables with computer assistance and better information.
- ❑ Easing the procedure of establishing the timetables.
- ❑ Establishing the timetables according to maximize facilities utilization and instructors' preferences.
- ❑ Reducing the number of conflicts due to courses taking place simultaneously but involving the same group of common students, instructors, or even requiring the same classroom.
- ❑ Balancing the number of students in each section of a course and also balancing the number of rooms used each day for maintenance purpose.
- ❑ Developing an interactive computer link to facilitate data output and the interpretation of the results.

Besides improving scheduling of the timetable at SIIT, the expected significance of this study includes applying this methodology to other scheduling situations where there are competing objectives and multiple resources.

4. The Solution Procedures

The mathematical model for the assignment problem uses the following decision variables:

x_{ij} = 1 if assignee i performs task j,
0 otherwise,
for $i = 1,2,\ldots,n$ and $j = 1,2,\ldots,n$.

Thus each x_{ij} is a 0-1 variable (0-No and 1-Yes). In this case, the yes/no decision is; should assignee i perform task j?

By letting Z denotes the total cost, the assignment problem model is:

$$\text{Minimize } Z = \sum_{i=1}^{n}\sum_{j=1}^{m} c_{ij}x_{ij}$$

Subject to

$\sum_{j=1}^{m} x_{ij}$ for $j = 1,2,\ldots m$, (supply constraint)

$\sum_{i=1}^{n} x_{ij}$ for $i = 1,2,\ldots n$, (demand constraint)

and $x_{ij} \geq 0$, for all i and j
(x_{ij} binary, for all i and j)

The first set of functional constraints specifies that each assignee is to perform exactly one task, whereas the second set requires each task to be performed by exactly one assignee. If the parenthetical restriction that "x_{ij} be binary" has been deleted, the model clearly is a special type of linear programming problem and so can be readily solved.

5. Method of Approach for SIIT

As stated, the development of a model for course scheduling and classroom assignment is not an easy task. In the development of simplified cost based model, all constraints have to be satisfied. The basic concept for setting up the model of class scheduling is to fit courses *i* into room *j* at period *k*. The following model is a conceptual simplified cost-minimizing model, which was developed at the very beginning and had been used as a basis for further model construction.

$$\text{Minimize } \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{p} c_{ijk}x_{ijk}$$

Where: $C_{i,j,k}$ is cost incurred when course *i* is assigned to room *j* in period *k*.

And: $X_{i,j,k}$ is equal to 1 when course *i* is assigned to room *j* in period *k*, and is equal to zero otherwise.

The following constraints and assumptions were taken into account:

- Instructors' constraints
- Classrooms' capacity constraints
- Time constraints
- Classes are allowed only on weekdays
- Classes are allowed from 9:00 to 16:30
- Classes held in any other building should be considered first

Finally, in order to deal with the objective function and all constraints, the mathematical modeling language software called "LINGO" has been implemented. LINGO is a simple tool to perform a complicated linear optimization task. Data input will be interpreted to LINGO pattern so that the model can be formulated and solved to get the most optimal scheduling plan.

6. The Model

With the assumption that classes are allowed only on weekdays from 9:00 to 16:30 and classes held in any other building should be considered first, the simplified cost based model has been developed according to the requirements of SIIT. The following model is a general form developed for solving the problem of two departments and two academic levels.

$$\text{Minimize } \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{p}\left[c_{ijk}x_{ijk}(f_j - e_i) + c_{ijk}x_{ijk}\right]$$

Subject to

$$\sum_{j=1}^{m}\sum_{k=1}^{p} x_{ijk} = 2 \quad 1 \leq i \leq n \tag{1}$$

$$\sum_{i=1}^{n} x_{ijk} \leq 1 \quad 1 \leq j \leq m,\ 1 \leq k \leq p \tag{2}$$

$$e_i x_{ijk} \leq f_j \quad 1 \leq i \leq n,\ 1 \leq j \leq m,\ 1 \leq k \leq p \tag{3}$$

$$\sum_{i\in I_r}\sum_{j=1}^{m} x_{ijk} \leq 1 \quad 1 \leq k \leq p,\ 1 \leq r \leq R \tag{4}$$

$$\sum_{i\in I_r}\sum_{j=1}^{m}\sum_{k\in T_r} x_{ijk} = 0 \tag{5}$$

$$\sum_{i\in I_s}\sum_{j=1}^{m} x_{ijk} \leq 1 \quad 1 \leq k \leq p,\ 1 \leq s \leq S \tag{6}$$

$$x_{ijk} = \{0,1\} \tag{7}$$

Where: i	=	course, $1 \leq i \leq n$
j	=	room, $1 \leq j \leq m$
k	=	period, $1 \leq k \leq p$ (See Table 1)
x_{ijk}	=	one when course i is assigned to room j, at period k and zero otherwise.

f_j	=	maximum capacity of each room (facility) j.
e_i	=	number of student enrolled in course i.
c_{ijk}	=	cost incurred when course i is assigned to room j at period k.
I_r	=	set of courses i taught by instructor r.
I_s	=	set of courses i attended by student group s.
T_r	=	set of period k where instructor r is not available.

Table 1: Table of periodic variable k.

Day	Time				
	9:00 - 10:30	10:30 -12:30	12:00 - 13:30	13:30 - 15:00	15:00 -16:30
Monday	1	2	Lunch Break	3	4
Tuesday	5	6		7	8
Wednesday	9	10		11	12
Thursday	13	14		15	16
Friday	17	18		19	20

The model was formulated and separated into two important sections: the objective function and the constraints. The objective function minimizes the cost of assigning courses to rooms at any time period during the week. The cost incurred at the weekend is assumed to be much greater than that incurred during the week. Thus in this study no courses are allowed to occur over the weekend.

$$\text{Minimize} \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{p}\left[c_{ijk}x_{ijk}\left(f_j - e_i\right) + c_{ijk}x_{ijk}\right]$$

Descriptively, the term $c_{i,j,k}x_{i,j,k}\,(f_j - e_i)$ in the objective function is the cost considered to be the loss per one empty seat. This penalty will be incurred when the number of students is less than the capacity of the room in which they were engaged. This avoids the problems associated with scheduling a course section of twenty into a room seating one hundred as stated earlier. The remaining term, $c_{i,j,k}x_{i,j,k}$, is the cost which will be occurred when assigning one course to one room to one period ($x_{i,j,k}$ = 1).

To keep all restrictions in control, seven constraints have been equipped in this model. The periodic constraint (1) ensures that, during the week, each subject has to be assigned to two time periods.

$$\sum_{j=1}^{m}\sum_{k=1}^{p} x_{ijk} = 2 \quad 1 \le i \le n \qquad (1)$$

For SIIT, normal courses of three credits should have three hours of lecture during a week. In this study, courses of three credits have to be taught twice week according to the length of each period (one and a half-hour). Generally, there are two kinds of lecture class during the week, i.e. one three-consecutive-hour class and two one-and-a-half-hour classes. (See Table 2)

Table 2: Practical timetabling for SIIT

Day	Time				
	9:00 - 10:30	10:30 -12:30	12:00 - 13:30	13:30 - 15:00	15:00 -16:30
Monday	TU 112		Lunch Break		
Tuesday	IE 373	IE 346		IE 353	
Wednesday		IE 346			
Thursday	IE 373				
Friday					

Therefore, constraint (1) has to be extended so as to cope with these specific conditions. The following constraint was formulated, based on the constraint (1), to satisfy the condition of one three-consecutive-hour class. See Table 2 course TU 112 and IE 353 for example.

$$\sum_{j}^{m=j}\sum_{k\in Odd}^{p=k+1} x_{ijk} = 2 \qquad 1 \le i \le n \qquad (1.1)$$

This equation will make the class of two periods to be consecutive and, of course, these classes will be assigned to the same room. The term *Odd* in the equation means the set of odd periods (k = 1, 3, 5...), which will prevent all courses from being separated by the lunch break.

Another practical condition is that, for the one-and-a-half-hour classes, the second class of the week has to be held on any other weekday, for example, courses IE 346 and IE 373 in Table 2. This coming constraint is an example of assigning courses to the same time and the same room on Tuesday and Thursday.

$$\sum_{j}^{m=j} \sum_{\substack{k=5,6,7,8,\\13,14,15,16}}^{p=k+8} x_{ijk} = 2 \quad 1 \le i \le n \qquad (1.2)$$

Practically, courses would be divided into two more categories, I_3 and $I_{1.5}$, according to their scheduling condition. I_3 would represent courses requiring one three-consecutive-hour lecture class and $I_{1.5}$ refers to courses requiring two one-and-a-half-hour lecture classes.

The room constraint (2) is used to take care of the condition that the room must be assigned to only one course at one period.

$$\sum_{i=1}^{n} x_{ijk} \le 1 \qquad 1 \le j \le m,\ 1 \le k \le p \qquad (2)$$

The capacity constraint (3) deals with the capacity of the room; that the number of students in each course should not exceed the capacity of the assigned room.

$$e_i x_{ijk} \le f_j \qquad 1 \le i \le n,\ 1 \le j \le m,\ 1 \le k \le p \quad (3)$$

Next, instructors' constraint (4) and (5), make sure that in each period an instructor teaches only one course and precludes the assignment of courses to the time periods where instructors are not available, respectively.

$$\sum_{i \in I_r} \sum_{j=1}^{m} x_{ijk} \le 1 \quad 1 \le k \le p,\ 1 \le r \le R \qquad (4)$$

$$\sum_{i \in I_r} \sum_{j=1}^{m} \sum_{k \in T_r} x_{ijk} = 0 \qquad (5)$$

The student constraint (6) affirms that one student group can attend only one course at a time.

$$\sum_{i \in I_s} \sum_{j=1}^{m} x_{ijk} \le 1 \quad 1 \le k \le p,\ 1 \le s \le S \qquad (6)$$

The last one, binary constraint (7), is used to define the model to be a binary integer linear programming model.

$$x_{ijk} = \{0,1\} \qquad (7)$$

7. Grouping Operation

In order to preclude the conflict of a group of students attending more than one subject at the same time, grouping operation has been introduced. The student in the same group will enroll in the same stream of subjects. In SIIT, students are separated into five departments: Civil (CE), Electrical (EE), Industrial (IE), Mechanical Engineering (ME), and Information Technology (IT) (except first year students.) Therefore, groups of students are created based on the department and the academic year of students. In total, there are 17 student groups at this time: the first year student has three sections, the second and the third year have five departments, and the fourth year has only four departments (no IT students). After the groups have already been formed, the subject grouping will be originated by matching the subjects, which would be attended by the same student group. The procedure of forming the subject groups is explained through an example. According to the model, this example considers only the subjects attended by second year CE and IE students, as shown in Table 3.

Table 3: Subjects of second year student.

i	Code	Department of 2nd year student	
		CE	IE
9	EL 210	✓	
10	CE 361	✓	
11	IE 308	✓	
12	MA 218	✓	✓
13	MA 219	✓	✓
14	ME 302	✓	✓
15	XX xxx	✓	✓
19	ET 303		✓
20	ET 304		✓
21	IE 301		✓
22	ME 310		✓

The first step is selecting the subject attended by only one student group. Then, group the subjects attended by the same group of students into one subject group. The Second step is choosing the subject that two student groups attend, and then group the subjects attended by two identical groups of students. Next, the subjects which are attended by three student groups, and then the subjects which are attended by four student groups, and so on until no more subjects are left to be considered.

From this example, the subject group will be as follows:

- Subject group 1: EL 210, CE 361, and IE 308 or $i = \{9, 10, 11\}$, attended by CE students alone,
- Subject group 2: ET 303, ET 304, IE 301, and ME 310 or $i = \{19, 20, 21, 22\}$, attended by only IE students,
- Subject group 3: MA 218, MA 219, Me 302 and XX xxx (free elective) or $i = \{12, 13, 14, 15\}$, attended by both CE and IE students,

Finally, these subject groups will be named I_S, which refers to sets of courses i attended by student group s. These sets will be used in the students constraint that restricts each student to attend not more than one course at each period. The operation of forming I_S also implies the approximate number of students in each course. The other two sets, I_r and T_r, which are the set of courses i taught by instructor r and the set of period k where instructor r is not available, can be formed by using the same manner as I_S. The example of LINGO model of this example is provided in Appendix A.

8. Conclusion

This study has examined the possibility of implementing the concept of assignment problem to formulate a linear programming model for the class scheduling process of SIIT. The objective function of the model was formulated on the concept of cost minimization while correlated restrictions were formulated as mathematical constraints. The formulation of what was considered as being constraints appeared to cause many more difficulties than that of the objective function, since the accuracy and the acceptability of the result are directly dependent on the formulation of the constraints. To cope with those difficulties, the grouping operation has been introduced as a tool to increase the effectiveness of the model and the accuracy of the result.

An example of scheduling has been done in this study in order to prove the reliability and the effectiveness of the formulated mathematical model (Appendix A). For this purpose, linear programming software called LINGO has been implemented to perform the calculation. The result came out to be acceptable in the sense that all the restrictions or constraints have been satisfied. This result has reaffirmed the effectiveness and the potential of the formulated model to cope with the class scheduling problems. The advantage of this system is that it would be able to applied not only to similar scheduling problems and be extended to various types of problem sharing the same concept.

9. References

[1] Tripathy, A. (1984), School Timetabling-A Case in Large Binary Integer Linear Programming, Management Science, Vol.30, No.12, pp.1473-1489.

[2] Mulvey, J.M. (1982), A Classroom/Time Assignment Model, European Journal of Operational Research, Vol.9, No.1, pp.64-70.

[3] Dinkel, J.J., Mote, J.,& Venkataramanan, M.A. (1989), An Efficient Decision Support System for Academic Course Scheduling. Operation Research, Vol.37, No.6, pp.853-864.

[4] Ferland, J.A., & Roy, S. (1985), Timetabling Problem for University as Assignment of Activities to Resources, Computer and Operation Research, Vol.12, No.2, pp.207-218.

[5] Aubin, J., & Ferland, J.A. (1989), A Large Scale Timebling Problem, Computer & Operation Research, Vol.16, No.1, pp.67-77.

[6] Hertz, A. (1991), Tabu Search for Large Scale Timetabling Problems, European Journal of Operation Research, Vol.54, No.1, pp.39-47.

[7] Mooney, E.L., Darden, R.L., & Parameter, W.J. (1996), Large-scale Classroom Scheduling, IIE Transaction, Vol.28, No.5, pp.369-378.

[8] Carter, M.J., & Covey, C.A. (1992). When Is the Classroom Assignment Problem Hard?, Operation Research, Vol.40, Supplement 1, S28-S39.

[9] Frederick S. Hillier, Gerald J. Lieberman (1995), Introduction to Operations Research, McGraw-Hill International, 6th Edition, Industrial Engineering Series.

Appendix A: LINGO Experimental Model

```
MODEL:

SETS:

COURSE / i1, i2, i3, i4, i5, i6, i7, i8, i9, i10, i11, i12,
         i13, i14, i15, i16, i17, i18, i19, i20, i21, i22 / : E;
! E = NUMBER OF STUDENT;
    lr1 (COURSE) / i1, i8, i9 /;
    lr2 (COURSE) / i3, i5 /;
    lr3 (COURSE) / i2 /;
    lr4 (COURSE) / i4, i7 /;
    lr5 (COURSE) / i6 /;
    lr6 (COURSE) / i10 /;
    lr7 (COURSE) / i11 /;
    lr8 (COURSE) / i12, i21 /;
    lr9 (COURSE) / i13 /;
    lr10 (COURSE) / i14, i18, i19 /;
    lr11 (COURSE) / i15, i16, i20 /;
    lr12 (COURSE) / i17 /;
    lr13 (COURSE) / i22 /;

    C3 (COURSE) / i6, i11, i13, i21, i22 /;
    C1_5(COURSE)/ i2, i10, i17, i15 /;
    CRAN(COURSE)/ i1, i3, i4, i5, i7, i8, i9, i12, i14, i16, i18, i19, i20 /;

    ls1 (COURSE)/ i1, i2, i3, i4, i5, i6 /;
    ls2 (COURSE)/ i7, i8, i9, i10 /;
    ls3 (COURSE)/ i12, i13, i14, i15, i16, i17 /;
    ls4 (COURSE)/ i18, i19, i20, i21 /;
    ls5 (COURSE)/ i22 /;
    ls6 (COURSE)/ i11 /;

    ls1andls5 (COURSE)/ i1, i2, i3, i4, i5, i6, i22 /;
    ls2andls6 (COURSE)/ i7, i8, i9, i10, i11 /;
    ls3andls5 (COURSE)/ i12, i13, i14, i15, i16, i17, i22 /;
    ls4andls6 (COURSE)/ i18, i19, i20, i21, i11 /;

 ROOM / j1, j2, j3, j4, j5, j6, j7, j8, j9, j10,
        j11, j12, j13, j14, j15, j16, j17 /: F;
! F = ROOM CAPACITY;

PERIOD / k1, k2, k3, k4, k5, k6, k7, k8, k9, k10,
    k11, k12, k13, k14, k15, k16, k17, k18, k19, k20 / ;
    MON (PERIOD) / k1, k2, k3, k4 /;
    TUE (PERIOD) / k5, k6, k7, k8 /;
    WED (PERIOD) / k9, k10, k11, k12 /;
    THU (PERIOD) / k13, k14, k15, k16 /;
    FRI (PERIOD) / k17, k18, k19, k20 /;

    Tr1 (PERIOD) / k5, k6, k7, k8 /;
    Tr2 (PERIOD) / k1, k2, k3, k4 /;
    Tr3 (PERIOD) / k9, k10, k11, k12 /;
    Tr4 (PERIOD) / k13, k14, k15, k16 /;
    Tr5 (PERIOD) / k17, k18, k19, k20 /;
    Tr6 (PERIOD) / k1, k2, k3, k4 /;
    Tr7 (PERIOD) / k1, k2, k3, k4, k5, k6, k7, k8, k13, k14, k15, k16 /;
    Tr8 (PERIOD) / k5, k6, k11, k12 /;
    Tr9 (PERIOD) / k7, k8, k13, k14 /;
    Tr10 (PERIOD) / k3, k4, k17, k18 /;
    Tr11 (PERIOD) / k11, k12, k19, k20 /;
    Tr12 (PERIOD) / k1, k2, k9, k10 /;
    Tr13 (PERIOD) / k15, k16, k19, k20 /;

 ASSIGN (COURSE,ROOM, PERIOD): X, C;
 !X = UNKNOWN VARIABLE;
 !C = ASSIGNING COST;

 ENDSETS

   MIN = @SUM(ASSIGN(I,J,K): (C(I,J,K)* X(I,J,K) *(F(J) - E(I))))
       + @SUM(ASSIGN(I,J,K): (C(I,J,K)*X(I,J,K)));

 !ONE COURSE HAVE TO BE ASSIGNED TO ANY ROOM AT 2 PERIODS;
    @FOR(COURSE(I):
       @SUM(PERIOD(K):
          @SUM(ROOM(J): X(I,J,K)))=2);
```

```
!ONE COURSE HAS TO BE ASSIGNED TO ONLY ONE ROOM AT THE
SPECIFIED PERIOD;
    @FOR(C3(I):
            @FOR(PERIOD(K)| (K#EQ#1) #OR# (K#EQ#3) #OR# (K#EQ#5)
#OR# (K#EQ#7) #OR# (K#EQ#9) #OR# (K#EQ#11) #OR# (K#EQ#13) #OR#
(K#EQ#15) #OR# (K#EQ#17) #OR# (K#EQ#19):
                @FOR(PERIOD(P)| P #EQ# K+1:
                    @FOR(ROOM(J):
                        @FOR(ROOM(L)|  L  #EQ#  J:  X(I,J,K)-X
(I,L,P)=0;););););

    @FOR(C1_5(I):
        @FOR(PERIOD(K):
            @FOR(PERIOD(P)| (K#EQ#5 #AND# P#EQ#13) #OR# (K#EQ#6
#AND# P#EQ#14) #OR# (K#EQ#7 #AND# P#EQ#15) #OR# (K#EQ#8 #AND#
P#EQ#16) #OR# (K#EQ#9 #AND# P#EQ#17) #OR# (K#EQ#10 #AND#
P#EQ#18) #OR# (K#EQ#11 #AND# P#EQ#19) #OR# (K#EQ#12 #AND#
P#EQ#20):
                    @FOR(ROOM(J):
                        @FOR(ROOM(L)|  L  #EQ#  J:  X(I,J,K)-X
(I,L,P)=0;););););

!SOME COURSES CANNOT BE ASSIGNED TO TWO PERIIOD DURING
THE DAY;
    @FOR(CRAN(I):
        @SUM(ROOM(J):
            @SUM(MON(K): X(I,J,K)))<=1;);
    @FOR(CRAN(I):
        @SUM(ROOM(J):
            @SUM(TUE(K): X(I,J,K)))<=1;);
    @FOR(CRAN(I):
        @SUM(ROOM(J):
            @SUM(WED(K): X(I,J,K)))<=1;);
    @FOR(CRAN(I):
        @SUM(ROOM(J):
            @SUM(THU(K): X(I,J,K)))<=1;);
    @FOR(CRAN(I):
        @SUM(ROOM(J):
      @SUM(FRI(K): X(I,J,K)))<=1;);

!AT 1 PERIOD, THE ROOM CAN BE ASSINGED BY ONLY 1 COURSE;
    @FOR(PERIOD(K):
        @FOR(ROOM(J):
            @SUM(COURSE(I): X(I,J,K))<=1););

!ASSIGN COURSES TO APPROPRIATE ROOMS;
    @FOR(COURSE(I):
        @FOR(ROOM(J):
            @FOR(PERIOD(K): (E(I)*X(I,J,K))<=F(J);););

!IN EACH PERIOD,AN INSTRUCTOR CAN TEACH ONLY 1 COURSE;
    @FOR(PERIOD(K):
        @SUM(Ir1(I):
            @SUM(ROOM(J): X(I,J,K)))<=1);
    @FOR(PERIOD(K):
        @SUM(Ir2(I):
            @SUM(ROOM(J): X(I,J,K)))<=1);
    @FOR(PERIOD(K):
        @SUM(Ir3(I):
            @SUM(ROOM(J): X(I,J,K)))<=1);
    @FOR(PERIOD(K):
        @SUM(Ir4(I):
            @SUM(ROOM(J): X(I,J,K)))<=1);
    @FOR(PERIOD(K):
        @SUM(Ir5(I):
            @SUM(ROOM(J): X(I,J,K)))<=1);
    @FOR(PERIOD(K):
        @SUM(Ir6(I):
            @SUM(ROOM(J): X(I,J,K)))<=1);
    @FOR(PERIOD(K):
        @SUM(Ir7(I):
            @SUM(ROOM(J): X(I,J,K)))<=1);
    @FOR(PERIOD(K):
        @SUM(Ir8(I):
            @SUM(ROOM(J): X(I,J,K)))<=1);
    @FOR(PERIOD(K):
        @SUM(Ir9(I):
            @SUM(ROOM(J): X(I,J,K)))<=1);
    @FOR(PERIOD(K):
        @SUM(Ir10(I):
            @SUM(ROOM(J): X(I,J,K)))<=1);
    @FOR(PERIOD(K):
        @SUM(Ir11(I):
            @SUM(ROOM(J): X(I,J,K)))<=1);
    @FOR(PERIOD(K):
        @SUM(Ir12(I):
            @SUM(ROOM(J): X(I,J,K)))<=1);
    @FOR(PERIOD(K):
        @SUM(Ir13(I):
            @SUM(ROOM(J): X(I,J,K)))<=1);

!AN INSTRUCTOR CANNOT TEACH AT HIS UNAVAILABLE TIME;
    @SUM(Ir1(I):
        @SUM(Tr1(K):
            @SUM(ROOM(J): X(I,J,K))))=0;
```

```
@SUM(Ir2(I):
    @SUM(Tr2(K):
        @SUM(ROOM(J): X(I,J,K))))=0;
@SUM(Ir3(I):
    @SUM(Tr3(K):
        @SUM(ROOM(J): X(I,J,K))))=0;
@SUM(Ir4(I):
    @SUM(Tr4(K):
        @SUM(ROOM(J): X(I,J,K))))=0;
@SUM(Ir5(I):
    @SUM(Tr5(K):
        @SUM(ROOM(J): X(I,J,K))))=0;
@SUM(Ir6(I):
    @SUM(Tr6(K):
        @SUM(ROOM(J): X(I,J,K))))=0;
@SUM(Ir7(I):
    @SUM(Tr7(K):
        @SUM(ROOM(J): X(I,J,K))))=0;
@SUM(Ir8(I):
    @SUM(Tr8(K):
        @SUM(ROOM(J): X(I,J,K))))=0;
@SUM(Ir9(I):
    @SUM(Tr9(K):
        @SUM(ROOM(J): X(I,J,K))))=0;
@SUM(Ir10(I):
    @SUM(Tr10(K):
        @SUM(ROOM(J): X(I,J,K))))=0;
@SUM(Ir11(I):
    @SUM(Tr11(K):
        @SUM(ROOM(J): X(I,J,K))))=0;
@SUM(Ir12(I):
    @SUM(Tr12(K):
        @SUM(ROOM(J): X(I,J,K))))=0;
    @SUM(Ir13(I):
        @SUM(Tr13(K):
            @SUM(ROOM(J): X(I,J,K))))=0;

!THE STUDENT CANNOT ATTEND MORE THAN ONE COURSE IN EACH
PERIOD;
    @FOR(PERIOD(K):
        @SUM(Is1andIs5(I):
            @SUM(ROOM(J): X(I,J,K)))<=1);
    @FOR(PERIOD(K):
        @SUM(Is2andIs6(I):
            @SUM(ROOM(J): X(I,J,K)))<=1);
    @FOR(PERIOD(K):
        @SUM(Is3andIs5(I):
            @SUM(ROOM(J): X(I,J,K)))<=1);
    @FOR(PERIOD(K):
        @SUM(Is4andIs6(I):
            @SUM(ROOM(J): X(I,J,K)))<=1);

!BINARY INTEGER;
    @FOR(ASSIGN: @BIN(X););

DATA:

     F = 35,35,75,75,75,75,75,75,35,75,75,125,125,105,45,185,185;

     E = 65, 68, 60, 62, 66, 65, 63, 54, 45, 48, 130,
         70, 68, 74, 80, 85, 73, 59, 53, 67, 35, 150 ;

     C = @IMPORT(IM_EX.XLS, COST);

ENDDATA
```