Comput. & Ops. Res. Vol. 12, No. 2, pp. 207–218, 1985
Printed in the U.S.A.

0305–0548/85 $3.00 + .00
Pergamon Press Ltd.

TIMETABLING PROBLEM FOR UNIVERSITY AS ASSIGNMENT OF ACTIVITIES TO RESOURCES†

JACQUES A. FERLAND‡ and SERGE ROY§
Département d'informatique et de recherche opérationnelle, Université de Montréal,
Montréal, Canada

(*Received February* 1983; *in revised form October* 1983)

Abstract—A mathematical programming approach is presented tô solve the timetabling problem in a University. It includes two subproblems that are solved sequentially. Both subproblems have the same structure consisting of a 0–1 assignment problem of conflicting activities to resources. A solution method is derived for a relaxed version of an equivalent 0–1 quadratic assignment programming problem. The paper concludes with implementation remarks, numerical results and extensions.

Scope and Purpose—The problem analyzed in this paper is to specify a timetable for class periods allowing the students to follow their selection of courses as much as possible. Furthermore, the timetable is established according to teachers and classrooms availabilities. The approach to solve the problem is to treat two subproblems sequentially. First a timetable is specified where the objective is to reduce the number of conflicts (two lectures involving the same students or requiring the same classroom scheduled simultaneously are said to be in conflict) and to respect the teacher availability as far as possible. Then, given a timetable, the classrooms are selected for the class periods according to their availability and to the specific requirements for the courses (laboratory, large room, small room, etc.). If there is a shortage of classrooms during some time periods, then the timetabling problem is solved again with additional conflicting situation for courses scheduled during the peak periods. The procedure is repeated until a classroom assignment can be specified for a timetable.

1. INTRODUCTION

The development of course schedules and classroom assignments in a University is not an easy task. The problem is especially complicated because class periods are not all of the same length. Two classes cannot be held simultaneously if at least one student is registered in both, or if they are given by the same lecturer. Sufficient time must be provided for students to move from one building to another, if necessary. Furthermore, a lecturer may not be available or may prefer to give the class at certain times of the week.

In this paper, the timetabling problem is decomposed into a class period scheduling problem, and a classroom assignment problem. The mathematical programming models for these subproblems have the same structure of assignment problem of conflicting activities to resources. The overall approach is to determine a class period schedule, and then to try to specify a feasible classroom assignment. If a class assignment cannot be found for this schedule, the data for the class period scheduling problem are modified accordingly and a new schedule is generated. The procedure is repeated until a feasible classroom assignment has been generated for some class period schedule.

Other approaches have been proposed to solve this problem or special versions of it. Some of these use mathematical programming methods and others use heuristic methods. Many of these approaches are summarized in [7]. Bloomfield and McSharry[1] and Knauer[8] propose heuristic methods for this problem. Mulvey[9] suggests a network model.

In our approach, each subproblem is formulated as a problem of assigning conflicting

†This research was supported by NSERC (Grant A83312 and a Summer scholarship) and by FCAC (Grant 0899).

‡Jacques A. Ferland is professor in the department of Informatique et Recherche Opérationnelle at the Université de Montréal. He received his Ph.D. in Operations Research and M.Sc. in Statistics from Stanford. He also obtained an M.Sc. in Informatique and a B.Sc. in Mathematics from the Université de Montréal. His main research interests include routing and scheduling problems, transportation problems, large scale mathematical programming, and generalized convexity.

§Serge Roy is research analyst at the Centre de Recherche sur les Transports of the Université de Montréal. He received an M.Sc. in Informatique and a B.Sc. in Informatique et Recherche Opérationnelle from the Université de Montréal. His research interests include routing and scheduling problems and transportation problems.

activities to resources. This 0–1 mathematical programming problem is transformed into an equivalent quadratic programming problem with 0–1 variables by introducing the complicating constraints (defining conflicts) into the objective function via penalties. It can be shown that an integer optimal solution always exists for the relaxed version of the quadratic problem. The solution method identifies a local minimum which is a "good" solution to the original problem.

In Section 2, the two subproblems are formulated. Equivalent quadratic assignment problem formulations are presented in Section 3. In Section 4, the relationships between these quadratic assignment problems and those proposed by Carlson and Nemhauser[2] and by Davis *et al.*[3] are analysed. In Section 5 an algorithm to solve a quadratic assignment problem is outlined and the details are confined in the appendix. Finally, some remarks on implementation and numerical results are given in Section 6, and some concluding extensions are suggested in Section 7.

2. PROBLEM FORMULATION

A mathematical formulation for each problem is now introduced. It will become apparent that both problems have the structure of an assignment problem of conflicting activities to resources.

2.1 *Period assignment*

Each course involves a number of class periods each week. The length of these periods is not uniform; some may last one hour, other may last two or three. These class periods defined as *activities* have to be assigned to time periods in the week (*resources*). If τ denotes the length of the longest class period, then a resource is associated with each period of τ consecutive hours starting at a specific time of a specific day of the week (for instance

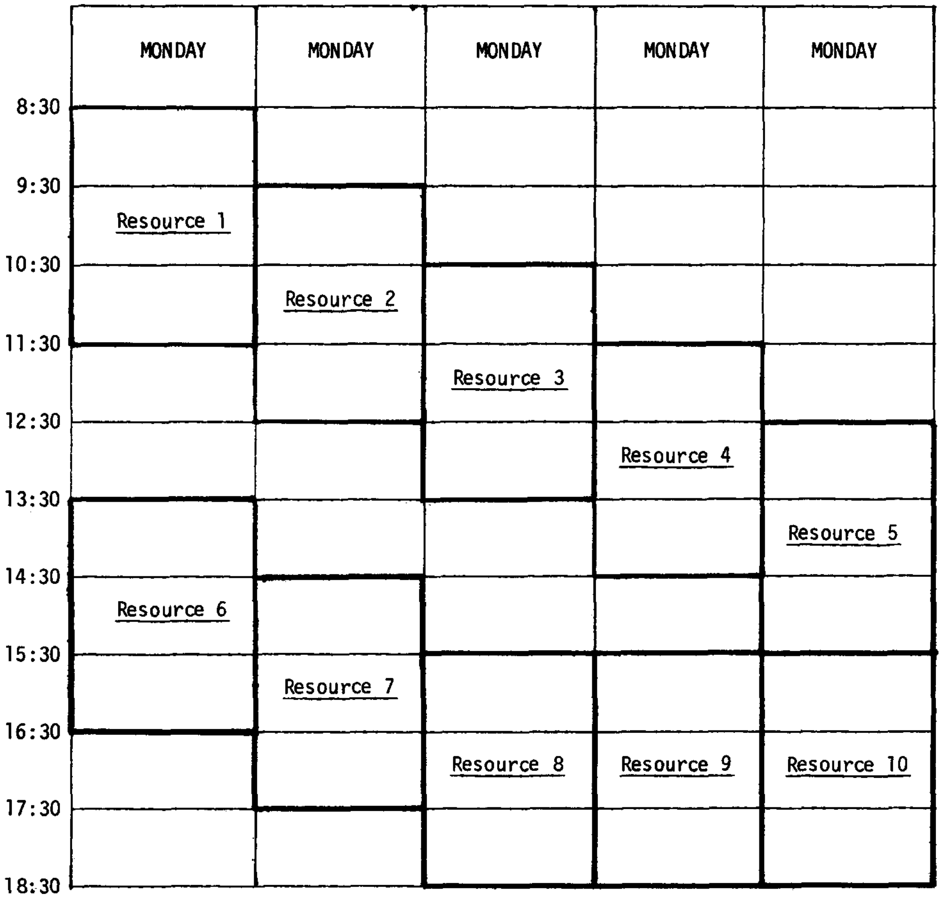

Fig. 1. Resources for Monday if $\tau = 3$.

if $\tau = 3$, then a resource is associated with the period 09:30–12:30 on Monday). Shorter class periods are assumed to be held as early as possible during a time period (for instance a class period of 2 hr is held from 09:30 to 11:30 on Monday if it is assigned to the resource described above). Note that τ copies of the last resource of each day are necessary to allow a one hour class period to be held during any of the last τ one hour periods of the day. This is illustrated in Fig. 1.

For each activity i and resource j, $1 \leq i \leq n$, $1 \leq j \leq m$, the decision variable x_{ij} is defined as follows:

$$x_{ij} = \begin{cases} 1 & \text{if activity } i \text{ is assigned to resource } j \\ 0 & \text{otherwise.} \end{cases}$$

c_{ij} is the cost of assigning activity i to resource j. In this problem, c_{ij} measures the availability and preferences of the lecturer responsible for the class period i during period j.

The concept of conflict between class periods must be introduced. Two class periods of different courses cannot be held simultaneously if a student is registered in both courses, or if they are both given by the same lecturer, or if they require the same specific classroom (laboratory for instance). Furthermore, sufficient time has to be provided for students to move from one building to another, if necessary. One way of simulating this is to increase the length of class periods for those lectures given in a building other than the one where most other lectures take place. It follows from our definition of the resource that two class periods assigned to different periods might be in conflict. For instance if a two hour class period i is assigned to resource period j from 09:30 to 12:30 on Monday and a one hour class period k to resource period l from 10:30 to 13:30 on Monday, then both classes are in progress from 10:30 to 11:30. If these courses include common students, or a common lecturer, or require the same room, a conflict is generated. On the other hand, if a two hour class period i' is assigned to resource period j' from 10:30 to 13:30 on Monday and a one hour class period k' to resource period l' from 09:30 to 12:30, then the class periods do not overlap in time. This is illustrated in Fig. 2.

The mathematical model can now be specified as follows

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

(AP) Subject to

$$\sum_{j=1}^{m} x_{ij} = 1, \quad 1 \leq i \leq n \tag{2.1}$$

$$x_{ij} + x_{kl} \leq 1, \quad l \in J_{ijk}, k > i, \tag{2.2}$$

$$1 \leq i \leq n, \; 1 \leq j \leq m$$

$$x_{ij} = 0 \text{ or } 1, \quad 1 \leq i \leq n, \; 1 \leq j \leq m$$

where J_{ijk} is the set of resource periods for activity k which generate a conflict with the assignment of activity i to resource j. In the examples illustrated in Fig. 2, $l \in J_{ijk}$ and $l' \notin J_{i'j'k'}$. With constraints $x_{ij} = 0$ or $1 (1 \leq i \leq n, \; 1 \leq j \leq m)$, constraints (2.1) require that each class period be assigned to a resource, and constraints (2.2) indicate that at most one of the two variables x_{ij} or x_{kl} can equal 1 if $l \in J_{ijk}$.

The model allows the lecturer to specify preferences for each time period ranging from 1 (highly preferred) to 4 (lecturer unavailable). The cost structure is specified accordingly:

$$c_{ij} = \begin{cases} k \text{ is resource } j \text{ is ranked } k \text{ for class period } i, \; 1 \leq k \leq 3 \\ n \cdot m + 1 \text{ if resource } j \text{ is ranked 4 for class period } i. \end{cases} \tag{2.3}$$

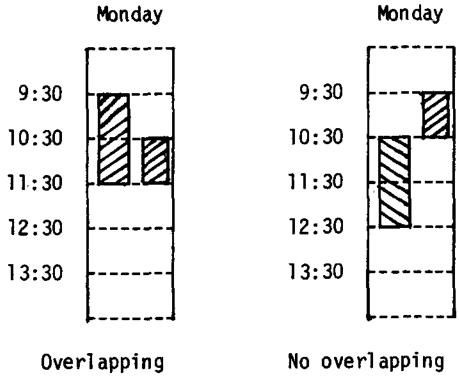

Fig. 2.

Hence an activity is never assigned to a resource ranked 4 to permit another activity to move from a resource ranked $k (k \leq 3)$ to another ranked $k - 1$.

2.2 *Classroom assignment*

It is assumed that the period assignment problem has been solved. So the problem is now to specify the classrooms for the class periods given in the schedule. As before, the class periods correspond to *activities* (within the specified schedule) and the *resources* are the classrooms.

For each activity i and resource r, $1 \leq i \leq n$, $1 \leq r \leq w$, the decision variable y_{ir} is defined as follows:

$$y_{ir} = \begin{cases} 1 & \text{if activity } i \text{ is assigned to resource } r \\ 0 & \text{otherwise.} \end{cases}$$

d_{ir} is the cost of assigning activity i to resource r and is specified as follows:

$$d_{ir} = \begin{cases} \text{the number of empty seats if class period } i \text{ is assigned to classroom } r \\ n \cdot \Gamma + 1 \text{ if the capacity of classroom } r \text{ is too small for the number of} \\ \qquad \text{students taking class period } i. \end{cases}$$

where Γ is greater than the size of the largest classroom. So, classrooms capacities are respected as much as possible, and the most appropriate classroom is assigned to each class.

Conflicts between activities for resources are dependent on the class period schedule obtained in Section 2.1. Let $x_{ij} = x_{kl} = 1$, i.e. class period i is assigned to time period j and class period k to time period l. If class periods i and k are held simultaneously for at least one hour, then they are in conflict for the use of any classroom; i.e. they cannot be held in the same classroom.

The mathematical model is now introduced.

$$\text{Min} \sum_{i=1}^{n} \sum_{r=1}^{w} d_{ir} y_{ir}$$

(AC) Subject to

$$\sum_{r=1}^{w} y_{ir} = 1, \quad 1 \leq i \leq n \tag{2.4}$$

$$y_{ir} + y_{kr} \leq 1, \quad k \in K_i,\ k > i \tag{2.5}$$

$$1 \leq i \leq n,\ 1 \leq r \leq w$$

$$y_{ir} = 0 \text{ or } 1, \quad 1 \leq i \leq n,\ 1 \leq r \leq w$$

where K_i is the set of class periods in conflict with i for the assignment of classrooms. With constraints $y_{ir} = 0$ or 1, constraints (2.4) require that each class period be assigned to a classroom, and constraints (2.5) indicate that at most one of two class periods in conflict is assigned to a given classroom.

Note that the conflict structure is simpler in this case, because two class periods are either in conflict or not, and if they are, the conflicts exists for each classroom. Nevertheless, problems (AP) and (AC) have the same structure, and the same procedure is used to solve them.

3. THE QUADRATIC ASSIGNMENT PROBLEM

The number of constraints (2.2) in (AP) and (2.5) in (AC) may be very large. So it may be worthwhile to include these constraints in the objective functions via an exact penalty approach. This leads to the following model that can be shown to be equivalent in a sense to (AP)

$$\text{Min } z(x) = \sum_{i=1}^{n} \sum_{j=1}^{m} \left[c_{ij} x_{ij} + \frac{1}{2} \sum_{k=1}^{n} \sum_{l=1}^{m} p_{ijkl} x_{ij} x_{kl} \right]$$

(QP) Subject to

$$\sum_{j=1}^{m} x_{ij} = 1, \quad 1 \le i \le n,$$

$$x_{ij} = 0 \text{ or } 1, \quad 1 \le i \le n,\ 1 \le j \le m,$$

where

$$p_{ijkl} = \begin{cases} (n \cdot m + 1)n + 1 & \text{if assigning } k \text{ to } l \text{ generates a conflict when } i \\ & \text{is assigned to } j \text{ (i.e. } l \in J_{ijk}) \\ 0 & \text{otherwise.} \end{cases} \tag{3.1}$$

Note that p_{ijkl} is chosen to ensure some equivalence between (QP) and (AP).

Similarly, the following model is equivalent to (AC)

$$\text{Min} \sum_{i=1}^{n} \sum_{r=1}^{w} [d_{ir} y_{ir}] + \frac{1}{2} \sum_{i=1}^{n} \sum_{k=1}^{n} \sum_{r=1}^{w} q_k y_{ir} y_{kr}$$

(QC) Subject to

$$\sum_{r=1}^{w} y_{ir} = 1, \quad 1 \le i \le n,$$

$$y_{ir} = 0 \text{ or } 1, \quad 1 \le i \le n,\ 1 \le r \le w,$$

where

$$q_{ik} = \begin{cases} (n\Gamma + 1) \cdot n + 1 & \text{if class periods } i \text{ and } k \text{ are in conflict (i.e. } k \in K_i) \\ 0 & \text{otherwise.} \end{cases} \tag{3.2}$$

It is easy to verify that if (AP) ((AC)) is feasible, then the $p_{ijkl}(q_{ik})$ are specified to insure that both problems (AP) and (QP) ((AC) and (QC)) have the same set of optimal solution[4, 10].

In Section 5, a solution procedure is given to solve (QP) and (QC). However, note that the feasible domain of these problems is never empty. Hence, even if (AP) ((AC)) is not feasible, an optimal solution exists for (QP) ((QC)), but it includes conflicts.

4. PREVIOUS MODELS

Special versions of problem (QP) have been analyzed before by Carlson and Nemhauser[2] and Davis *et al.*[3]. Unfortunately, neither of these models were appropriate for the assignment problems introduced in Section 2.

In [2] Carlson and Nemhauser introduce the following model

$$\text{Min}\,\frac{1}{2}\sum_{i=1}^{n}\sum_{k=1}^{n}\sum_{j=1}^{m} f_{ik}x_{ij}x_{kj}$$

(QCN) Subject to

$$\sum_{j=1}^{m} x_{ij}=1, \quad 1\le i\le n$$

$$x_{ij}=0 \text{ or } 1, \quad 1\le i\le n, 1\le j\le m.$$

Selecting the values of f_{ik} appropriately this problem could be shown equivalent to determining a feasible solution for the following assignment problem

$$\sum_{j=1}^{m} x_{ij}=1, \quad 1\le i\le n$$

(ACN) $$x_{ij}+x_{kj}\le 1, \quad k\in K_i,\ 1\le i\le n,\ 1\le j\le m$$

$$x_{ij}=0 \text{ or } 1, \quad 1\le i\le n,\ 1\le j\le m$$

where K_i is the set of activities in conflict with i. Hence (AC) is more general than (ACN) since it takes into account the size of the resources. Furthermore, contrasting (AP) with (ACN), note that (ACN) could be used to specify a feasible class period schedule only in the case where all class periods and all resource periods have exactly the same length (one hour for instance).

In [3] the authors generalize the model in [2].

$$\text{Min}\,\frac{1}{2}\sum_{i-1}^{n}\sum_{k-1}^{n}\sum_{p\in J_i}\sum_{q\in J_k} f_{ik}t_{pq}x_{ip}x_{kq}$$

(QDDL) Subject to

$$\sum_{p\in J_i} x_{ip}=1, \quad 1\le i\le n$$

$$x_{ip}=0 \text{ or } 1, \quad p\in J_i,\ 1\le i\le n$$

where the parameters are specified as follows

$$t_{pq}=\begin{cases}1 & \text{if resources } p \text{ and } q \text{ are in conflict}\\ 0 & \text{otherwise.}\end{cases}$$

$$J_i\subset\{1,2,\ldots,m\}.$$

Selecting the values of f_{ik} appropriately, this problem could be shown equivalent to determine a feasible solution for the following assignment problem

$$\text{(ADDL)} \qquad \begin{aligned} &\sum_{p\in J_i} x_{ip} = 1, \quad 1 \le i \le n \\ &x_{ip} + x_{kq} \le 1, \quad k \in K_i,\ q \in T_p,\ 1 \le i \le n,\ 1 \le p \le m \\ &x_{ip} = 0 \text{ or } 1, \quad 1 \le i \le n,\ 1 \le p \le m \end{aligned}$$

where K_i is the set of activities in conflict with i and T_p is the set of resources in conflict with p. As in (ACN) there is no assignment cost in (ADDL). Furthermore, the conflicting state of each pair of resources is specified *a priori* and independently of the activity assigned to them. (ADDL) could be used to specify a feasible schedule for assigning class periods of identical length to resource periods of identical length smaller than or equal to the length of class periods. For instance if the class periods are three hours long and resource periods are one hour long, each pair of consecutive resource periods of the same day are declared in conflict. This is more restrictive than in the model (AP).

5. SOLUTION PROCEDURE

The solution procedure for (QP) and (QC) is a straightforward extension of the one proposed by Carlson and Nemhauser in [2] for (QCN). Some preliminary results extending those in [10] are introduced, but their proofs are omitted. They will be introduced for the more general model (QP), but similar results exist for (QC).

The first result indicates that the integrality constraint on the variables x_{ij} of (QP) can be relaxed. For future reference, this relaxed version of (QP) is denoted (RQP).

THEOREM 5.1

Suppose the integrality constraints $x_{ij} = 0$ or 1 in (QP) are replaced by $x_{ij} \ge 0$ to generate the relaxed problem (RQP). These exists an optimal solution x^* of (RQP) such that the integrality constraints $x^*_{ij} = 0$ or 1 are satisfied for all $1 \le i \le n$, $1 \le j \le m$.

To analyse the optimality conditions for (RQP) we use the Kuhn–Tucker conditions:

$$\begin{aligned} c_{ij} + \sum_{k-1}^{n}\sum_{l-1}^{m} p_{ijkl}x_{kl} - \pi_i \ge 0, &\qquad 1 \le i \le n,\ 1 \le j \le m \\ x_{ij}\left[c_{ij} + \sum_{k-1}^{n}\sum_{l-1}^{m} p_{ijkl}x_{kl} - \pi_i\right] = 0, &\qquad 1 \le i \le n,\ 1 \le j \le m \\ \sum_{j=1}^{m} x_{ij} = 1, &\qquad 1 \le i \le n \\ x_{ij} \ge 0, &\qquad 1 \le i \le n,\ 1 \le j \le m. \end{aligned}$$

To ease the notation, let

$$r_{ij} = c_{ij} + \sum_{k-1}^{n}\sum_{l-1}^{m} p_{ijkl}x_{kl}, \qquad 1 \le i \le n,\ 1 \le j \le m.$$

These conditions are necessary, but they are not sufficient because the objective function of (QP) is not convex. Indeed, it is easy to verify that the quadratic term $\sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{n}\sum_{l=1}^{m} p_{ijkl}x_{ij}x_{kl}$ is not convex. Fortunately, it is possible to derive additional conditions for a vector x satisfying the Kuhn–Tucker conditions to be a local minimum of (RQP).

THEOREM 5.2.

If vectors $x \in E^{n \times m}$ and $\pi \in E^n$ satisfy the Kuhn–Tucker conditions for problem (RQP), and if at most n of the inequalities $r_{ij} - \pi_i \geq 0$ hold with equality, then x is a local minimum for (RQP).

It follows from this result that if x is a feasible solution of (RQP) such that for all $1 \leq i \leq n$, there exist an index j_i such that

$$\pi_i = r_{ij_i} < r_{ij}, \qquad 1 \leq j \leq m, \qquad j \neq j_i$$

$$x_{ij_i} = 1,$$

then x is a local minimum of (RQP).

The algorithm to be introduced at the end of this section relies on the following result to measure the effect of reassigning an activity to a different resource.

THEOREM 5.3.

Let x be a feasible solution of (QP) such that activity i is assigned to resource j_i (i.e. $x_{ij_i} = 1$). If activity i is assigned to resource j instead, then the modification induced on the objective function is equal to $(r_{ij} - r_{ij_i})$.

It follows from this result that if activity i is assigned to resource j_i but $r_{ij_i} > \min_{1 \leq j \leq m} \{r_{ij}\}$, then the objective function could be reduced by reassigning i. Such an operation is referred to as a *single reassignment* in the algorithm.

Once the objective function cannot be improved with single reassignments, the algorithm proceed with *double reassignments* where two different activities are reassigned. The increase in computer time induced by this type of reassignment prohibits the use of higher level of reassignment involving more than two activities. We rather prefer to use some single reassignments inducing no improvement of the objective function. These might entail new single or double reassignments improving the objective function. In [2] the authors do not use double reassignments but they rather proceed directly to single reassignments inducing an improvement of the objective function.

As illustrated for single reassignment, the improvement of the objective function is measured with the quantities

$$s_{ij} = r_{ij} - r_{ij_i} \qquad 1 \leq i \leq n, 1 \leq j \leq m$$

related to the Kuhn–Tucker conditions. An expression involving the s_{ij} and the p_{ijkl} can be derived to evaluate the modification of the objective function induced by a double reassignment. Furthermore, the s_{ij} can be updated easily after reassignments. Details on this are given in the appendix.

A specific algorithm is given in the appendix, but the major steps can be outlined as follows:

Step 0. Determine a feasible solution for (QP).

Step 1. Improve the objective function with single reassignments.

Step 2. Proceed to improve the objective function with double reassignments.

Step 3. If the solution is a local minimum, then terminate.

Step 4. Execute single reassignments inducing no improvement of the objective function. If the objective function can be improved with single or double reassignments, repeat Step 1 or Step 2, respectively. Otherwise, terminate.

6. IMPLEMENTATION AND RESULTS

The critical factor in the computer implementation of the algorithm of Section 5 is the number of parameters p_{ijkl}. This number is too large to keep each parameter explicitly in memory. On the other hand, computing these parameters each time they are needed, requires too much computer time. The following compromise approach was used in implementation. Focussing more closely the notion of conflict, two class periods i and k cannot be held simultaneously, or are said to be in conflict if either:

(c1) at least one student is registered in both classes;
(c2) the same lecturer teaches both classes;
(c3) they require the same classroom (laboratory for instance).

Furthermore, $p_{ijkl} \neq 0$ if one of the following conditions is satisfied:

(h1) Class periods i and k belong to different courses that are in conflict and there is a time overlap between the position of i in period j and the position of k in period l.

(h2) Class periods i and k belong to different courses that are in conflict and there is insufficient time for a student to move from the building where i is held to the one where k is held if i in period j and k in period l.

(h3) Both class periods belong to the same course and periods k and l belong to the same day.

Hence if $p_{ijkl} \neq 0$, then k and l belong to the same day. Furthermore, the possible conflicting situations between two class periods in a day vary with the length of these periods. It is sufficient to specify these typical situations for each pair of class periods of different lengths to be able to analyze any specific pair of class periods. For instance, in Fig. 3 where resource periods are 3 hr long, a possible conflict situation between a class period i of 2 hr assigned in period j and a class period k of 3 hr assigned in period l is indicated by a 1 in cell (j, l) of the matrix. Note the special status of the last 3 rows and 3 columns due to the fact that the last period is duplicated 3 times (see Fig. 1). Similar matrices are used to specify conflicting situations when time for movement of students is required (see h2). The matrix illustrated in Fig. 4 corresponds to the conflict situation in the previous example but with an additional hour added to allow for the movement of students.

These matrices are stored explicitly in memory. The p_{ijkl} are evaluated rapidly using these matrices and a boolean matrix indicating the course in conflict. It should be noted that the dimensions of the preceeding matrices do not depend on the size of the problem.

		3 hour class period									
		08:30	09:30	10:30	11:30	12:30	13:30	14:30	15:30	16:30	17:30
2 hour class period	08:30	1	1	0	0	0	0	0	0	0	0
	09:30	1	1	1	0	0	0	0	0	0	0
	10:30	1	1	1	1	0	0	0	0	0	0
	11:30	0	1	1	1	1	0	0	0	0	0
	12:30	0	0	1	1	1	1	0	0	0	0
	13:30	0	0	0	1	1	1	1	0	0	0
	14:30	0	0	0	0	1	1	1	1	1	1
	15:30	0	0	0	0	0	1	1	1	1	1
	16:30	0	0	0	0	0	0	1	1	1	1
	17:30	0	0	0	0	0	0	1	1	1	1

Fig. 3. Resource period denoted by its starting time.

		3 hour class periods									
		08:30	09:30	10:30	11:30	12:30	13:30	14:30	15:30	16:30	17:30
2 hour class periods	08:30	1	1	1	0	0	0	0	0	0	0
	09:30	1	1	1	1	0	0	0	0	0	0
	10:30	1	1	1	1	1	0	0	0	0	0
	11:30	1	1	1	1	1	1	0	0	0	0
	12:30	0	1	1	1	1	1	1	1	1	1
	13:30	0	0	1	1	1	1	1	1	1	1
	14:30	0	0	0	1	1	1	1	1	1	1
	15:30	0	0	0	0	1	1	1	1	1	1
	16:30	0	0	0	0	0	1	1	1	1	1
	17:30	0	0	0	0	0	1	1	1	1	1

Fig. 4.

The overall procedure to solve the entire problem of period assignment and classroom assignment goes as follows:

Step 1. Solve (QP) to generate a class period schedule.

Step 2. Using this schedule, specify the parameters for the classroom assignment.

Step 3. Solve (QC) to generate a classroom assignment to class periods. If there is no conflict; i.e. if

$$\sum_{i=1}^{n} \sum_{k=1}^{m} \sum_{r=1}^{w} q_{ik} y_{ir} y_{kr} = 0,$$

then stop with this feasible classroom assignment for this class period schedule.

Step 4. If there are some conflicts, for each pair of class periods i and k such that

$$\sum_{r-1}^{w} q_{ik} y_{ir} y_{kr} > 0,$$

assume that i and k are in conflict of type (c3) (i.e. assume they require the same classroom). Modify the parameters of (QP) accordingly, and repeat step 1.

In practice, the solution of (QP) in step 1 may not necessarily generate a feasible schedule because there may be some conflicts. In this case, some conflicts have to be relaxed to allow a feasible schedule. Note also that this procedure assumes that it is always possible to obtain a feasible classroom assignment for some feasible class period schedule.

The algorithm outlined in Section 5 and induced in the appendix was implemented in PASCAL on the CYBER 173 at the computing center of the Université de Montréal. It was tested with success on several course scheduling problems.

The behavior of the algorithm for three different course scheduling problems is analysed in Table 1. Note that an important part of the execution time is used to initialize the s_{ij}. Furthermore in these examples no single reassignment inducing no improvement of the objective function were executed. In another example it was possible to eliminate one more conflicting hour after some of these reassignments were executed.

7. EXTENSIONS

Referring to (3.1) where p_{ijkl} is defined, note that the number of students involved in a conflict is not considered. The definition of p_{ijkl} can be modified to include a measure of the seriousness of conflict proportional to the number of students involved.

Table 1.

Problem					Initial solution		Initializing matrix S	Single reassignment		Double reassignment		Overall execution		Final solution	
umber f courses	Number of periods				Number of conflicting hours	Number of students involved	time* in sec.	Number	time* in sec.	Number	time* in sec.	time* in sec.	space required	Number of conflicting hours	Number of students involved
		1hr	2hr	3hr											
43	69	31	31	7	23	61	14.56	19	7.73	4	41.29	63.58	15K	0	0
83	154	98	53	3	15	149	58.49	35	26.10	5	108.0	192.59	21K	1	1
174	297	139	131	27	35	76	222.23	65	88.10	9	280.43	590.76	37K	1	1

* CYBER -173
CDC

An interactive computer system has been developed to facilitate data input and the use of the program by users without training in computer science. More details are available in [5]. Quadratic assignment approach can be extended to analyse other problems like Examination Timetabling and Game Scheduling[6].

REFERENCES

1. S. D. Bloomfield and M. M. McScharry, Preferential course scheduling. *Interfaces*, **9**, 24–31 (1979).
2. R. C. Carlson and G. L. Nemhauser, Scheduling to minimize interaction cost. *Ops Res*. **14**, 52–58 (1966).
3. F. E. Davis, M. D. Devine and R. P. Lutz, Scheduling activities among conflicting facilities to minimize conflict cost. *Math. Prog*. **6**, 224–228 (1974).
4. J. A. Ferland and S. Roy, *Course scheduling and classroom assignment in a university*, Publication No. 459. Département d'informatique et de recherche opérationnelle, Université de Montréal (1982).
5. J. A. Ferland and S. Roy, *Système informatique en mode conversationnel de confection d'horaires de cours*, Publication No. 460. Département d'informatique et de recherche opérationnelle, Université de Montréal (1982).
6. J. A. Ferland, S. Roy and G. Loc, *Quadratic assignment models for examination timetabling and game scheduling*, Publication No. 485. Département d'informatique et de recherche opérationnelle, Université de Montréal (1983).
7. O. B. de Gans, A computer timetabling system for secondary schools in the Netherlands. *Europ. J. Op. Res*. 7, 175–182 (1981).
8. B. A. Knauer, Solution of a timetable problem. *Comput. Ops Res*. **1**, 363–375 (1974).
9. J. M. Mulvey, A classroom/time assignment model. *Eur. J. Op. Res*. **9**, 64–70 (1982).
10. S. Roy, *Problème d'affectation généralisée avec conflits*: *application au problème de confection d'horaire de cours*. Mémoire de maîtrise. Département d'informatique et de recherche opérationnelle, Université de Montréal (1982).

APPENDIX

This appendix includes the algorithm outlined in Section 5. Let us denote

$$s_{ij} = r_{ij} - r_{ij_i} \qquad 1 \leq i \leq n,\ 1 \leq j \leq m,$$

and $S = [s_{ij}]$ an $n \times m$ matrix.

To specify the *single reassignment* inducing the largest improvement in the objective function, determine i^* and j^* such that

$$s_{i^*j^*} = \underset{\substack{1 \leq i \leq n \\ 1 \leq j \leq m}}{\text{Min}} \{s_{ij}\}$$

and reassign i^* from j_{i^*} to j^*. This generates a new feasible solution x' for (QP) for which the quantities r'_{ij} and s'_{ij} have to be computed to repeat the procedure. It is easy to verify that the s'_{ij} can be evaluated in terms of the s_{ij} as follows:

$$s'_{ij} = s_{ij} + p_{ij_ii^*j_{i^*}} - p_{iji^*j_{i^*}} + p_{iji^*j^*} - p_{ij_ii^*j^*}, \quad 1 \leq j \leq m,\ 1 \leq i \leq n,\ i \neq i^*. \tag{A1}$$

For $i = i^*$, the values s_{i^*j} are calculated from scratch; i.e.

$$s'_{i^*j} = c_{i^*j} - c_{i^*j^*} + \sum_{k=1}^{n} \sum_{l=1}^{m} [p_{i^*jkl} - p_{i^*j^*kl}], \quad 1 \leq j \leq m. \tag{A2}$$

If

$$s_{i^*j^*} = \min_{\substack{1 \le i \le n \\ 1 \le j \le m}} \{s_{ij}\} = 0,$$

then the objective function cannot be decreased by a single reassignment. But a *double reassignment* involving two activities might induce an improvement of the objective function. In this case, we determine the pairs i^*, j^* and k^*, l^* such that

$$\Delta_{i^*j^*k^*l^*} = s_{i^*j^*} + s_{k^*l^*} + p_{i^*j^*k^*l^*} - p_{i^*j_{i^*}k^*l^*} - p_{i^*j^*k^*j_{k^*}} + p_{i^*j_{i^*}k^*j_{k^*}} = \min_{\substack{1 \le i,k \le n \\ 1 \le j,l \le m}} \{s_{ij} + s_{kl} + p_{ijkl} - p_{ij_ikl} - p_{ijkj_k} + p_{ij_ikj_k}\}.$$

If $\Delta_{i^*j^*k^*l^*} < 0$, then the double reassignment where i^* is reassigned from j_{i^*} to j^* and k^* from j_{k^*} to l^* is executed.

If $s_{i^*j^*} = \Delta_{i^*j^*k^*l^*} = 0$ and if the local optimality conditions are not satisfied, then a single reassignment inducing no improvement of the objective function is executed whenever possible. Indeed, select a pair $\tilde{i}, \tilde{j}$ such that $s_{\tilde{i}\tilde{j}} = 0$ and $\tilde{j} \ne j_{\tilde{i}}$. The single reassignment of $\tilde{i}$ from $j_{\tilde{i}}$ to $\tilde{j}$ might eventually induce an improvement of the objective function. In [2] the authors do not use double reassignment but they rather proceed to single reassignment with zero improvement when $s_{i^*j^*} = 0$.

The algorithm is now summarized as follows

Step 0. Determine a feasible solution for (QP). Let $i = \bar{j} = 0$. Evaluate s_{ij}, $1 \le i \le n$, $1 \le j \le m$.

Step 1. Evaluate $s_{i^*j^*} = \min_{\substack{1 \le i \le n \\ 1 \le j \le m}} \{s_{ij}\}$. If $s_{i^*j^*} = 0$, then go to step 2. Otherwise $(s_{i^*j^*} \le 0)$ execute the single reassignment: $x_{i^*j_{i^*}} = 0$ and $x_{i^*j^*} = 1$, and go to step 6.

Step 2. Evaluate $\Delta_{i^*j^*k^*l^*}$. If $\Delta_{i^*j^*k^*l^*} = 0$, then go to step 3. Otherwise $(\Delta_{i^*j^*k^*l^*} < 0)$ execute the double reassignment: $x_{i^*j_{i^*}} = 0$ and $x_{i^*j^*} = 1$, $x_{k^*j_{k^*}} = 0$ and $x_{k^*l^*} = 1$, and go to step 6.

Step 3. (*Local minimum test*). If $\min_{1 \le j \le m} \{s_{ij}\}$ is unique for all $1 \le i \le n$, then x is a local minimum for (RQP) and the algorithm stops. Otherwise, determine the lexicographically smallest pair $(\tilde{i}, \tilde{j})$ greater than $(\bar{i}, \bar{j})$ such that $\tilde{j} \ne j_{\tilde{i}}$ and $s_{\tilde{i}\tilde{j}} = 0$. If such a pair exists, go to step 5.

Step 4. x does not satisfy the conditions to be a local minimum of (RQP), but it is not possible to improve the objective function with single and double reassignment. Stop.

Step 5. Execute the single reassignment (with a zero improvement of the objective function): $x_{\tilde{i}j_{\tilde{i}}} = 0$ and $x_{\tilde{i}\tilde{j}} = 1$, and modify the matrix S according to relations (A1) and (A2). Set $\bar{i} = \tilde{i}$ and $\bar{j} = \tilde{j}$ and repeat step 1.

Step 6. Modify the matrix S according to relations (A1) and (A2). Set $\bar{i} = \bar{j} = 0$, and repeat step 1.

Note also that in step 6 relations (A1) and (A2) are applied successively for the pairs i^*, j^* and k^*, l^* to modify the matrix S after a double reassignment.

Finally, it can be shown[4, 10] that a single reassignment in this algorithm corresponds to an iteration of the convex simplex method.