QÜESTIIÓ, vol. 21, 1 i 2, p. 201-215, 1997

A TABU SEARCH ALGORITHM TO SCHEDULE UNIVERSITY EXAMINATIONS*

RAMÓN ÁLVAREZ-VALDÉS*
ENRIC CRESPO**
JOSÉ M. TAMARIT*
Universitat de València

Scheduling examinations in a large university is an increasingly complex problem, due to its size, the growing flexibility of students' curricula and the interest in including a wide set of objectives and constraints. In this paper we present a new algorithm for this problem and its application to a university in Spain.

A combination of heuristics, based on Tabu Search, first finds a solution in which no student has two exams simultaneously and then improves it by evenly spacing the exams in the examination period. The algorithm has been imbedded into a package to be used by Faculty administrators and proposes solutions to be considered by the parties involved: administration, departments and students. Computational results, comparing our results with the timetables actually used, are presented.

Keywords: Timetabling, scheduling, heuristics, tabu search.

* Ramón Álvarez-Valdés and José M. Tamarit. Department of Statistics and Operations Research. University of València. Dr. Moliner, 50. 46100 BURJASSOT (València).

** Enric Crespo. Department of Financial and Mathematical Economy.

⋆ This work was subsidised in part by the Conselleria d'Educació i Ciència, Generalitat Valenciana, GV 2212/94

– Article rebut el juliol de 1996.

– Acceptat el febrer de 1997.

1. INTRODUCTION

The examination scheduling problem consists of assigning exams to periods and classrooms in such a way that all exams are scheduled within a given time interval and a number of different objectives are satisfied. Each university offers a large list of courses every term and allows students a great deal of flexibility in their course selection. The result is a large and complex scheduling problem. Although the main objective is that no student has two exams simultaneously, there are many other secondary objectives, such as students' convenience, classroom availabilities and the special requirements of some exams.

The problem has attracted the attention of researchers over the past 30 years. Though most papers are tailored to the characteristics of authors' universities, two trends can be pointed out: the increasing interest in considering a wide set of objectives and side constraints, and the use of new and powerful heuristics to cope with the growing size and complexity of the problems.

The survey by Carter(1986) covers the work carried out before 1986. Most of it used some modified colouring heuristics —Desroches *et al.* (1978), Mehta(1981), White *et al.* (1979)—. Two different approaches were the algorithm HORHEC of Laporte and Desroches(1984), an assignment procedure with limited backtracking, and the paper by Romero(1982), in which the computer assists in the negotiation process between administration, departments and students.

All the mentioned papers had avoiding or minimizing *first order conflicts* as their primary interest, that is, students having two exams simultaneously. However, some of them started to consider some secondary objectives, such as minimizing *back-to-back conflicts*, that is students having exams in consecutive periods, or distributing the exams evenly in the examination periods. In the papers appearing after 1986 this trend has become more pronounced, with multiphase procedures in which each objective is taken into account at each step in the process —Johnson(1990), Lofti *et al.* (1991), Thompson *et al.* (1993)— and papers in which, assuming that a solution with a minimum number of first order conflicts is known, the objective is minimizing back-to-back conflicts —Arani *et al.* (1988), Balakrishnan *et al.* (1992)—.

In recent years, examination scheduling problems have been solved by using some new heuristic procedures that have been shown to be very useful in other related problems. Thompson and Dowsland(1993) and Johnson(1990) have used Simulated Annealing. Hertz(1991) and Clark(1993) have developed Tabu Search algorithms. Clark addresses a problem quite similar to ours, though he does not consider many special characteristics we have included (forbidden periods for some exams, precedences, changes in the availability of classrooms,...). His algorithm is based on an

intensification procedure and diversification is reduced to a dynamic variation of the length of the tabu list.

Still, an emerging trend that may intensify is the evolution of solution procedures into packages that can be used directly by the final users, preferably on personal computers. A good example is EXAMINE, developed by Carter *et al.* (1994) from the algorithm of Laporte and Desroches(1984).

The purpose of this paper is to describe a new algorithm for examination scheduling and its application to a large university in Spain. In the same line of the above mentioned research on this problem, our algorithm combines several heuristics based on tabu search and aims to obtain not only a solution without simultaneous exams, but the best distribution of exams among the periods for all the students. These objectives are basically shared by all universities, but they are considered here according to the priorities and specifications established by our university in València, Spain.

The description of the problem, its constraints and the hierarchy of objectives are presented in Section 2. Section 3 describes our algorithmic approach and Section 4 the implementation and computational results. Finally, in Section 5 we draw some conclusions and outline future lines of research.

2. DATA, OBJECTIVES AND CONSTRAINTS

The University of València has 65,000 students divided into four main Areas: Social Sciences, Health Sciences, Humanities and Basic and Technical Sciences. Each Area has several Faculties or Schools that are responsible for the academic organization of classes and examinations. However, the students in a Faculty usually take some optional courses from other Faculties in their Area. Moreover, the classrooms are shared by Faculties. Therefore, examinations must be planned at Area level, making the task much more complex.

The necessary data for exam scheduling are:

- *Students*

  Each student is registered on a set of compulsory and optional courses. The number of courses is variable, but most students have to sit 4 or 5 exams each term, many of them with a theoretical and a practical part.

  From the actual registration file we can obtain the conflict matrix, that is, the number of students common to each pair of exams. This matrix has many non-zero elements and not only in the expected square submatrices corresponding to some course streams of typical students.

- *Periods*

  There is a prescribed exam interval, usually three weeks each term. Due to the length of exams, with theoretical and practical parts, it has been established that each day has only one examination period. That gives us a total of 15 periods (18 if all Saturdays are used).

- *Courses*

  Some courses do not have examinations. Semester courses usually have one exam and annual courses require one or more partial exams and a final examination. Therefore, from the list of courses we build the list of exams with their characteristics.

  Some exams have a restricted set of allowed periods or are directly preassigned to a period. Sometimes, a pair of exams are related in such a way that one must precede the other in at least a given number of periods, for instance, oral and written language exams.

  In some cases, the exam needs a special type of classroom, such as a laboratory or computer room. The size of the required classroom depends on the course enrollment and on a parameter, *the attendance factor*, which is the estimated percentage of registered students actually attending the exam. This parameter is inputted by the user, who may decide, for instance, a 100% attendance for a partial exam and a 70% for a final exam if the final is compulsory only for students below a certain mark in the partials.

- *Classrooms*

  At each period we have an available set of classrooms with their type and exam capacities, usually a half or a third of the seats, depending on classroom structure.

In the data description above the constraints have implicitly appeared: The exam interval must be strictly respected, preassignments and forbidden periods for exams must also be satisfied, as well as the precedence relations.

The objectives are basically three and there is a clear hierarchy among them:

1. No student should have two exams simultaneously. These first order conflicts have to be avoided. In fact, this condition could be considered a constraint, but we cannot be sure of finding a feasible solution if we enforce it because of the multiple combinations of students' registrations and the reduced number of periods. Therefore, we put it as the main objective and we will try to get a solution without first order conflicts.
2. For each student, his/her exams should be as evenly separated as possible along the exam interval. This idea extends the usual objective of avoiding consecutive

exams, back-to-back conflicts, and conditions of the type «Students should not write x or more examinations within any y consecutive periods» —Carter et al.(1994)—. In an examination interval of 15 periods, students with 6 exams cannot expect an average distance between exams greater than 2 periods, but that should not be applied to students with 2 or 3 exams who could obtain more distant exam dates.

This idea of evenly scattering the exams forms part of the non-written tradition of our university, especially in those faculties in which the examination timetabling has been done by hand after a negotiation process with teachers and students. Now, the complexity of the process makes it difficult to maintain, but an acceptable automatic substitute has to include this objective.

3. Classroom capacities should be respected. Clearly, this is a constraint, but not a hard one. The inclusion of some students above classroom capacity is not physically impossible, though it may not be desirable. If this condition is included as a constraint, we may loose some good solutions with reduced capacity overflow that could be considered acceptable by the parties involved.

As we mentioned above, a tradition of negotiation is being substituted with or assisted by computer systems. So far, in the Area of Basic and Technical Sciences, with four Faculties and 6,000 students, the scheduling process is done by hand in two stages. First, each Faculty builds its timetable in a negotiation between administration, departments and students. Then, the Faculties in the Area put together their proposals and look for a common classroom assignment, modifying examination dates if necessary. This procedure has serious drawbacks. In the first step, it cannot assure minimizing first order conflicts. In particular, it cannot adequately consider the courses attended by students from several Faculties. In the second step, it is very difficult to fit the timetables into the available classrooms, and changes in the dates deteriorate the quality of the solutions. Therefore, all parties involved are well aware of the need for changing this lengthy and unsatisfactory process by a more efficient procedure. However, from their experience, they will not accept solutions they do not consider at least as good as those manually obtained. Hence, timetables not only have to *be good* according to some agreed cost function, but they have to *look good* to the users, according to their knowledge of the situation and some characteristics that cannot be modelled but are clearly perceived by them just by looking at the solutions.

3. SOLUTION METHOD

The process of obtaining a good solution, according to the mentioned objectives, is divided into two phases. First, a solution without first order conflicts is built, whenever it exists; then it is improved with consideration to overall student convenience.

3.1. Building a feasible solution

An initial solution is built by randomly assigning exams to periods. The assignment for each exam is randomly chosen from the allowed periods for the exam that satisfy the precedence relations.

This process is repeated several times to obtain different initial solutions. They will evolve separately to produce several final solutions to be presented to the people involved in the decision.

These initial solutions have first order conflicts and do not satisfy classroom availability. For each of them we start a tabu search procedure, a general heuristic method for guiding the search to obtain good solutions in complex solution spaces. A general introduction to Tabu Search may be found in Glover *et al.* (1993). Though these techniques have been developed and extended in very sophisticated ways in recent years, the basic aspects of the procedure may be described as follows:

1. Construct an initial solution s in the space of solutions X

   $s^* = s$ (s^* = best solution found so far)

   $k = 1$ (k = number of iterations)

2. **While** $k <$ maximum number of iterations **do**

   $k = k + 1$

   Generate a set of solutions $V^* \subseteq N(s,k)$ (set of neighbours of s that are not tabu or for which the aspiration criterion applies)

   Choose the best s' in V^*

   $s = s'$

   If $f(s') < f(s^*)$, then $s^* = s'$

   end while

The space of solutions X in which we move is the set of timetables satisfying the allowed periods for exams and the precedence relations.

The objective function combines the number of first order conflicts with the excess of students over classroom capacities. For each solution s we define

$$f(s) = \sum_{i=1}^{N} (p \sum_{j,k \in E_i} x_{jk} + r_i)$$

where N is the number of periods, E_i the set of exams assigned to period i, r_i the room shortage at period i, x_{jk} the number of common students for exams j and k, and p the relative weight of conflicts with respect to overall room shortage.

A solution $s' \in X$ is a neighbour of solution $s \in X$ if it can be obtained from s by moving an examination to another period. To decide which neighbour s' we should move to from the current solution s, we do not explore the whole neighbourhood because that would be too costly. Instead, we select the *most conflictive period*, the period of biggest contribution to the objective function, and from among the exams in it we choose the *most conflictive examination*. We generate and evaluate all its possible moves and make the best one, as long as it is not tabu. If a move is tabu but it improves the best current solution, it is made in spite of its tabu status, applying the aspiration criterion. If all the moves are tabu and the aspiration criterion does not apply, we select the second most conflictive exam and repeat the procedure. The process is repeated as many times as necessary until a possible move is found and made.

For each move the tabu list keeps the examination that is moved and the period from which it comes. A move involving an exam and period in the tabu list is considered tabu. The aspiration criterion allows a move to be made, in spite of its tabu status, if the new solution is better than the best solution found so far.

This iterative procedure ends when we reach a conflict-free timetable. Nevertheless, sometimes the problem does not have a zero conflict solution, or even if it has, our algorithm is not guaranteed to find it. In these cases, the process stops when a limit of iterations is reached, then returning a timetable in which some students have more than one examination in a period.

3.2. Improving the solution

Once we get a feasible solution, or at least a solution with a minimum number of conflicts, we start to consider other objectives, such as a distribution of exams in the periods maximizing the distance between exams for the students. Also, the capacity of classrooms is now more important than it was in the previous phase.

The objective function of this phase reflects this new situation. For each solution s we define

$$f(s) = \sum_{i=1}^{N} \Big(\sum_{j=1}^{N} \big(p_{|i-j|} \sum_{k \in E_i} \sum_{l \in E_j} x_{kl} \big) + w r_i \Big)$$

where N is the number of periods, E_i the set of exams assigned to period i, r_i the excess of room capacity at period i, x_{kl} the number of common students of exams k and l, and $p_{|i-j|}$ the penalty associated with a pair of exams scheduled at a distance

of $|i-j|$ periods. The parameter w reflects the relative weight of room shortage with respect to conflicts.

The objective function includes conflicts of distance zero, that is, first order conflicts. Allowing these conflicts to appear makes the search more flexible. A high penalty p_0 will guide the process to solutions with a minimum number of these conflicts.

To improve the solution we have developed two procedures: *permutation of exam lists*, defined as the sets of exams assigned to the same period, and *interchange of exams*. Both procedures are complementary. The permutation of exam lists considers the list of exams in a period as a fixed set and moves the whole list from one period to another. Obviously, this procedure cannot produce first order conflicts and may improve the quality of the solution very fast. However, it is a very constrained process, because the limitations of exams (non-allowed periods, precedence relations,...) are limitations of their list. On the other hand, the interchange of exams takes the exams one at a time and considers moves or swaps. That allows us to study many more alternatives, though the improvement of the objective function is slower.

Interchange of exams

The procedure for interchanging exams is similar in spirit to that used to obtain a feasible solution but here we define a more flexible move. At each iteration we select a period t (*the most conflictive period*) and in it an exam e (*the most conflictive exam*). Then we consider its possible moves and evaluate them. If the exam e is going to a period t' and it does not produce first order conflicts, the change in the objective function depends on the distance to other exams and the room shortage. But if it produces some new first order conflicts, a high value of p_0 will produce a dramatic increase in the objective function and this move will never be made. In these cases, we study the possibility of moving some exams $e', e'', ..$ from period t' to period t to avoid some of these conflicts. Hence, instead of a single move, we make an *interchange*. This move is more complex, but allows us to study new alternatives to the current solution, reaching regions of the space of solutions that would not have been visited otherwise.

The permutation of exam lists

In this procedure, we consider changing the whole set of exams assigned in a given period to another one. If in the objective function described above we do not consider the weight of room shortage, the objective function could be written as

$$f(s) = \sum_{i=1}^{N} \sum_{j=1}^{N} p_{ij} C_{l(i)l(j)}$$

where $C_{l(i)l(j)}$ is the number of students having exams in both periods i and j and p_{ij} is the associated penalty. This is the objective function of the Quadratic Assignment Problem (QAP). Hence, the problem of finding the best permutation of lists may be viewed as a QAP. By doing that we do not get an easier problem, but we may take advantage of the recent work on it. Probably the most successful heuristic approach at this moment is the Tabu Search, as can be seen in the papers of Taillard(1991) and Skorin-Kapov(1994).

We have developed an algorithm that closely follows the work of the above mentioned authors. Nevertheless, we maintain our objective function because the availability of classrooms may vary and changes in room shortage must be reflected in the quality of moves. At each iteration we explore all the possible moves, discard those which are unfeasible and evaluate the rest (note that the reduced number of periods allows us such a complete exploration). Then we make the best feasible permutation, according to considerations about the tabu list and aspiration criterion similar to those described previously.

Iterative scheme

The two procedures are linked in an iterative scheme in which the solution obtained in one of them is the starting point for the other, until the solution cannot be improved further. This scheme may be viewed as the alternance of *intensification* and *diversification* phases of Tabu Search method. In the procedure of interchange of exams, local changes try to get a better solution in an intensification phase. When this phase cannot improve the solution, the procedure of list permutations deeply changes the solution in a diversification phase that moves the search to completely different regions of the solution space. The two phases follow the basic scheme of Section 3.1, with the same objective function and the types of moves (and corresponding neighbourhoods) defined above. The procedure switches from one phase to the other after a given number of iterations.

3.3. Assigning classrooms to examinations

The final phase of the procedure consists in assigning classrooms to examinations. This phase is by no means trivial in our problem because of the interaction of three factors: first, the number of available seats is quite reduced compared with the number of students; second, the solution obtained in the preceding phases uses the global number of seats and compares it with the total number of students having an exam at a given period, but a mechanic translation of the exam list to the classroom will produce a large number of exams having to share a classroom and that should be avoided as far as possible. Third, in our system each day is a period because the exams are very long and it is not desirable for the students to schedule two exams on the same day. However, each day has two sessions, morning and afternoon. Therefore,

the number of seats is counted twice, unless otherwise stated. The problem now is to assign exams to rooms and sessions in such a way that every exam is assigned to only one session, using the minimum number of rooms and without sharing the rooms with any other exam. Moreover, if in the timetable there are still some first order conflicts, the corresponding exams should be assigned to different sessions, to partially solve the problem of some students having two exams simultaneously. The solution, though undesirable, would be physically possible.

In this procedure the user must input two parameters: *the minimum number factor*, the minimum number of students a course must have to require a classroom (the exams of courses with very few students may take place in some other places, like some departmental room) and *the residual factor*, which is the percentage of remaining students for which a new clasroom is not needed when we assign more than one room to a large exam.

For this phase we have developed an assignment algorithm that considers the list of examinations ordered by size and first assigns each of them to a session, trying to solve, if necessary, the first order conflicts, and then to a set of classrooms. For each exam, knowing its size and the remaining available classrooms, the algorithm selects rooms minimizing their number and adjusting their overall capacity to exam size. Likewise it tries to minimize the number of invigilators and the number of classrooms shared by more than one examination.

4. IMPLEMENTATION AND COMPUTATIONAL RESULTS

The algorithms of the preceding section have been imbedded in a package to allow the user to input data and parameters, obtain several solutions, print them and modify the initial conditions and values of parameters to correct them or study new alternatives. The package has three basic and one auxiliary modules:

1. *Input module*, in which the user, through a set of menus and selection lists, edits the data about courses and rooms, sets the values of the parameters and gets information about student registration.

2. *Solution module*, including the algorithms described above, that produces several solutions from which the users may choose the most convenient.

3. *Classroom assignment module*, that produces for each solution the final list in which each examination appears, assigned to a day, a session and a set of classrooms.

   This module may also produce a classroom assignment for timetables provided by the user.

4. *Evaluation module*, an auxiliary module that evaluates a given solution according to the current objective function. This module is internally used by the algorithms of the Solution module, but may be used separately to study and compare other solutions obtained by other methods. In our case, coming from a system in which the timetable is done by hand after a negotiation process, it is especially important to compare the solutions in order to shed light on the deficiencies of the system and convince the users of the advantages of the new solutions.

The programs have been written in C language and the package has been designed to run on personal computers, the usual equipment of Faculty administrators.

To assess the performance of our algorithms, we have made two sets of tests. First, a small example, involving only the examinations of the Faculty of Mathematics, with 1,200 students, in which the difficulty comes from the complexity of students' registrations, making it very hard to find a timetable without first order conflicts.

We include four instances of the problem, corresponding to the year 1992-93. In that year all courses were annual, with partial examinations in February and June and final examinations in July. Therefore, they needed a timetable for February and another for June-July. The last one admitted two possibilites: making two separate timetables for June and July or building one common timetable.

The second test involves the whole area of Basic and Technical Sciences, with 6,000 students where together with its large size, the difficulty lies in the reduced number of available rooms and the existence of courses attended by students from different Faculties. In the new academic system we have some annual courses and a majority of semester courses. We have solved the problem for the end of the academic year 1994/95, with partial and final examinations for annual courses and final exams for spring semester courses, and also for final exams of the fall term in February 1996. In June 1995 the examination interval was of four weeks for partial examinations and five weeks for final ones. In February 1996 the exams interval was of three weeks.

The values of parameters were set after a series of trials. In the first phase of building a feasible solution we made $p = 100$, giving much more weight to first order conflicts than to room shortage. In the second phase of improving the solutions the parameters were $p_0 = 3000$, $p_1 = 100$, $p_2 = 20$, $p_3 = 5$, $p_4 = 3$, $p_5 = 1$, $p_k = 0, k > 5$ and the cost of room shortage $w = 40$. These values reflect the main objective of getting solutions without first order conflicts and the need for adjusting the examination timetables to the available classrooms.

The length of tabu lists were fixed at $\sqrt{n}$, where n is the *size* of each part of the problem. In the procedures in which the move consists of changing an exam from one period to other, the size is the number of exams (*ne*) times the number

of periods (np). In the procedure of list permutations, the size is $np(np-1)$. The number of iterations was set to 1000 in the moving-exams procedure and 500 in the list-permutation procedure. The tabu lists have a circular structure and at each iteration the oldest element on the list is replaced by the new one associated with the move.

Table 1 shows the data of the six tests (number of periods, number of exams, the total number of examinations to be done by the students and the density of conflicts matrix) and the comparison between the actual solutions obtained by the current process and those proposed by our algorithms, with respect to first order conflicts, room shortage, conflicts of distance 1 (*back-to-back conflicts*) and conflicts of distance 2.

Table 1. *Comparison of actual and proposed solutions*

Date	Per.	Ex. N.	Ex.-Stud.	Mat. dens.		F.or. conf.	Room short.	C.d.1	C.d.2
feb93	17	40	4853	0.50	actual	63	208	235	202
					alg	0	0	147	426
jun93	17	40	4853	0.50	actual	21	0	293	460
					alg	0	0	150	418
jul93	17	40	4853	0.50	actual	27	30	336	480
					alg	0	0	136	474
jj93	34	80	9706	0.50	actual	48	0*	629	946
					alg4	0	0	335	991
					joint10	1	0	439	1056
jun95	30	107	22011	0.15	actual	203	179	3076	1828
					alg	0	0	474	2753
feb96	18	132	26994	0.15	actual	267	0	3833	3860
					alg	3	0	1606	4563

The block *jj93* contains several solutions to the joint problem of partial and final examinations in 1993. The first row, *actual*, is the actual solution, obtained by putting together the solutions of *jun93* and *jul93*, with a minor modification made by the users: a final examination was moved backwards into the periods corresponding to partial exams. That improved the solution, mainly in the use of classrooms. In the second row, *alg4*, we put together our solutions for *jun93* and *jul93*. In this solution, the

minimum distance from partial and final examinations of a course is 4 periods, which is considered unacceptable for teachers and students. Therefore, the third row, *joint10* shows the solution imposing a minimum distance of 10 periods.

For all instances our algorithm obtained much better solutions than those manually built. The new academic system, which is much more flexible and contains more optional courses, plus the growing number of students, will make the problem harder and the difference between manual and automatic timetables larger, as already happens in last instances *jun95* and *feb96*.

The solutions were obtained on a personal computer with a Pentium processor at 100 Mhz. For the large problem *jun95*, it took 5 seconds to obtain the first feasible solution and 30 minutes to get the final solution. **Table 2** justifies the computational effort of the improving phase by comparing the initial feasible solutions with the final solutions obtained by asking the algorithm to obtain six alternative solutions. The last two columns show the number of times the improving phase calls on the exam interchange and list permutation procedures. Though some solutions may seem better than others, according to the criteria of the table (for instance, solution 1 looks better than solution 2), there may be some other reasons for the users to adopt one solution and discard the rest.

Table 2. *Comparison of initial and final solutions*

First order conflicts	Conflicts of distance 1	Conflicts of distance 2	Exam interchange	List permutation
0	4753	4809	-	-
0	474	2753	3	3
0	5432	4405	-	-
0	669	2828	7	7
0	5551	4578	-	-
0	726	2503	4	4
0	3684	5327	-	-
1	468	2911	3	3
0	4535	3164	-	-
0	614	2717	4	3
0	5758	3497	-	-
0	616	2355	2	2

In the trials, some other combinations of parameters, tabu list lengths and number of iterations produced better results for some of the test instances. Nevertheless, the values mentioned above produced consistently good results for the whole set of problems.

5. CONCLUSIONS AND FUTURE RESEARCH

We have developed a package to solve the examination timetabling problem for our university. Many of its characteristics are common to other universities and the procedures could be adapted and used in other places.

The use of Tabu Search algorithms has allowed us to obtain high quality solutions in very short computing times, in spite of the size of the problem and the complexity of data and objectives.

The solutions obtained in the test were presented to Faculty administrators in the Basic and Technical Sciences Area. The quality of our solutions and the serious problems faced in the process of obtaining the actual solution have convinced them of the convenience of adopting our package as a decision support system to assist the construction of future timetables.

The next step in our work will be the use of our procedure in the other Areas of the University. We are also planning to imbed these algorithms in a general academic management system we are currently developing.

6. REFERENCES

[1] **Arani T., Karwan M.** and **Lofti V.** (1988). «A Lagrangian relaxation approach to solve the second phase of the exam scheduling problem». *Eur. J. Opl Res.*, **34**, 372–383.

[2] **Balakrishnan N., Lucena A.** and **Wong R.T.** (1992). «Scheduling examinations to reduce second-order conflicts». *Computers Ops Res.*, **19**, 353–361.

[3] **Carter M.W.** (1986). «A survey of practical applications of examination timetabling problems». *Mgmt Sci.*, **34**, 193–202.

[4] **Carter M.W., Laporte G.** and **Chinneck J.W.** (1994). «A General Examination Scheduling System». *Interfaces*, **24**, 109–120.

[5] **Clark D.** (1993). «Exam scheduling by Tabu Search». *Australian Soc. Ops Res. Bulletin*, **12**, 5–9.

[6] **Desroches S., Laporte G.** and **Rousseau J.M.** (1978). «HOREX: A computer program for the construction of examination schedules». *INFOR*, **16**, 294–298.

[7] **Glover F., Taillard E.** and **de Werra D.** (1993). «A user's guide to Tabu Search». *Annals of Ops Res.*, **41**, 3–28.

[8] **Hertz A.** (1991). «Tabu search for large scale timetabling problems». *Eur. J. Opl Res.*, **54**, 39–47.

[9] **Johnson D.** (1990). «Timetabling university examinations». *J. Opl Res. Soc.*, **41**, 39–47.

[10] **Laporte G.** and **Desroches S.** (1984). «Examination timetabling by computer». *Computers Ops Res.*, **11**, 351–360.

[11] **Lofti V.** and **Cerveny R.** (1991). «A Final-exam-scheduling Package». *J. Opl Res. Soc.*, **42**, 205–216.

[12] **Mehta N.K.** (1981). «The application of graph coloring methods to an examination scheduling problem». *Interfaces*, **11**, 57–64.

[13] **Romero B.P.** (1982). «Examination scheduling in a large engineering school: A computer assisted participative approach». *Interfaces*, **12**, 17–24.

[14] **Skorin-Kapov J.** (1994). «Extensions of a Tabu Search adaptation to the Quadratic Assignment Problem». *Computers Ops Res.*, **21**, 855–865.

[15] **Taillard E.** (1991). «Robust taboo search for the quadratic assignment problem». *Parallel Computing*, **17**, 443–445.

[16] **Thompson J.** and **Dowsland K.** (1993). «Multi-objective university examination scheduling». *Working paper* EBMS/1993/12.

[17] **White G.M.** and **Chan P.W.** (1979). «Towards the construction of optimal examination schedules». *INFOR*, **17**, 219–229.