Discrete Applied Mathematics 35 (1992) 255-270
North-Holland

Finding a feasible course schedule using Tabu search

Alain Hertz

Département d'Informatique et de Recherche Opérationnelle, Université de Montréal, Case Postale 6128, succursale A, Montréal, Que., Canada H3C 3J7

Received 6 December 1989
Revised 19 July 1990

Abstract

Hertz, A., Finding a feasible course schedule using Tabu search, Discrete Applied Mathematics 35 (1992) 255-270.

We consider a course scheduling problem in which the total number of time periods assigned to each topic has to be split into daily amounts of consecutive time periods called daily quantums. These daily quantums may be arbitrary but are bounded by given minimal and maximal values.

In addition to the classical constraints of course scheduling problems, precedence and time windows requirements are taken into account. We describe a new method based on Tabu search techniques for finding a feasible course schedule.

Keywords. Course scheduling, Tabu search, assignment problems.

1. Introduction

Finding a feasible course schedule is not an easy task. Conflicts due to courses taking place simultaneously but involving common students or teachers have to be avoided. In addition to these classical constraints, teachers availabilities, precedence, compactness, geographical and time windows requirements have to be taken into account. Moreover, a maximal allowed number of time periods is assigned to each working day.

Furthermore, a total number of time periods is assigned to each topic; this total amount has to be split into daily amounts of consecutive time periods called daily quantums and bounded by given minimal and maximal values.

The set of consecutive time periods of a daily quantum is defined to be a course. It follows that the number of courses is not known in advance and their length (number of consecutive time periods) is not fixed.

Timetabling problems have already been studied by many authors. Different ap-

0166-218X/92/$05.00 © 1992 — Elsevier Science Publishers B.V. All rights reserved

proaches have been proposed [1–5, 7, 9, 12, 15] but most of them take into account only parts of the constraints and cannot be easily adapted to the case where the number of courses and their length are not known in advance. The purpose of this paper is to present a new global approach based on Tabu search techniques for tackling course scheduling problems in which the length of the courses are not necessarily known in advance. We shall first develop a model for the class-teacher timetabling problem. It will then be shown that this model can easily be extended for handling course scheduling problems in general.

In the next section the course scheduling problem is formulated. The basic ideas of Tabu search techniques are sketched in Section 3. The adaptation of Tabu search to the class-teacher timetabling problem is described in Section 4, and experiments are reported in Section 5. Finally, extensions are discussed in Section 6 and concluding remarks are given in Section 7.

2. Problem formulation

This paper has been motivated by a real-life class-teacher timetabling problem which is described in this section.

There are several classes of students and each class follows its own fixed curriculum. A set of available working days within which all class curriculums must be completed is given. Each curriculum consists of a given set of *subjects*. For each subject, the earliest working day (release date) at which it can start and the latest working day (due date) at which it must be completed are specified.

Each subject consists of a given set of *topics*. The topics of the curriculum of a class are partially ordered: this ordering specifies a predecessor-successor relation between the topics. Topics of different subjects can be in relation while topics from different classes are not in a predecessor-successor relation.

A fixed teacher is assigned to each topic and a teacher can teach one or more topics to one or more classes. The teachers and the classes are not necessarily available at each time period (a time period is usually defined as a 45 minute lecture-hour) of each working day. For each working day a total number of time periods is given which can vary from day to day. Moreover lunch breaks are fixed in advance and a course should not be interrupted by these breaks.

A total number of time periods is assigned to each topic. This total amount must be split during the scheduling process into daily amounts of consecutive time periods called *daily quantums*. The set of consecutive time periods of a daily quantum is defined to be a *course*. The *length* of a course is its number of consecutive time periods.

We distinguish two kinds of topics:

- for *static topics*, the daily quantums are fixed and ordered in advance,
- for *dynamic topics*, the daily quantums are arbitrary but bounded by given minimal and maximal values.

For example, a dynamic topic with twelve time periods and for which each course consists of two or three consecutive time periods can be divided into the following unordered sequences of daily quantums: (2,2,2,2,2,2), (2,2,2,3,3) or (3,3,3,3).

Dynamic topics appear quite naturally in practical problems. When a school has to build a course schedule, each teacher initially indicates how the total amount of time periods of his topics should approximately be distributed. This kind of information is in general not precise and the scheduler decides which is the length of each course. These choices may dramatically reduce the number of feasible course schedules. A model with dynamic topics avoids these undesired restrictions.

Let us show by an example how the requests of a teacher can be taken into account without fixing the length of the courses in advance. Let us consider a teacher who has to distribute eight time periods in a week. He would like to teach one course of length two on Friday and the six remaining time periods should be scheduled before Thursday. The length of each course should be two or three. In such a situation, the scheduler can define two different topics:

- One static topic (with one daily quantum of value two) representing the course to be scheduled on Friday. The release and the due date of this topic are Friday.
- One dynamic topic with six time periods. The release data of this topic is Monday, its due date is Wednesday, the maximal value of a daily quantum is three and the minimal value is two.

The six time periods of the dynamic topic represent either three courses of length two or two courses of length three. A model without dynamic topics would forbid one of these two possibilities.

The problem to solve is to find a feasible course schedule for the given class curriculums and the given set of working days such that:

(a) each teacher is involved in at most one topic at a time,

(b) each class is involved in at most one topic at a time,

(c) the predecessor-successor relation between the topics is satisfied,

(d) the daily quantums of a static topic are correctly ordered,

(e) the release and the due dates of all subjects are respected,

(f) each topic is scheduled in such a way that the corresponding teacher is available during each time period of each course of the topic,

(g) each course is scheduled in an uninterrupted way during available time periods of a working day,

(h) two courses of a same topic are scheduled on two different working days,

(i) the minimal and maximal numbers of consecutive time periods of each daily quantum of a dynamic topic are not violated.

This problem is known to be difficult. If we relax constraints (c), (d), (e), (h) and (i), and if we assume that we have no dynamic topic, the relaxed problem of finding a course schedule satisfying constraints (a), (b), (f) and (g) is proved to be NP-complete unless all teachers and all classes are always available [8].

In classical timetabling problems, a given set of courses has to be scheduled, taking into account several constraints. We solve here a quite different problem. We

do not know the number of courses and their length in advance. This makes the problem more complicated. The numerous scheduling methods described in the literature can generally not be applied in this case. For solving this timetabling problem we shall adapt the Tabu search technique which will be described in the next section.

3. Tabu search techniques

Tabu search is a metaheuristic designed for getting a global optimum to a combinatorial optimization problem. It has been first suggested by Glover [10] and independently by Hansen et al. [11] for a specific application, and later developed in a more general framework. A description of the method can be found in [6,10]. Tabu search has already been efficiently adapted to a large collection of applications [6,11–14,16]. We shall sketch here the basic ideas of the technique.

An objective function f has to be minimized on a set X of feasible solutions. A neighborhood $N(s)$ is defined for each solution s in X. The set X and the definition of the neighborhood induce a state-space graph G (which may by the way be infinite). Tabu search is basically an iterative procedure which starts from an initial feasible solution and tries to reach an optimal solution by moving step by step in the state-space graph G. Each step consists in first generating a collection V^* of solutions in the neighborhood $N(s)$ of the current solution s, and then moving to the best solution s' in V^*, even if $f(s') > f(s)$. Let us write $s' = s \oplus m$ with the meaning that s' is obtained by applying a modification m to s. The solutions consecutively visited in the iterative process induce an oriented path in G. Finding the best solution in V^* may sometimes be a nontrivial matter. It may be necessary to solve the optimization problem $\min\{f(s_i) \mid s_i \in V^*\}$ by a heuristic procedure.

A risk of cycling exists as soon as a solution s' worse than s is accepted. In order to prevent cycling to some extent, modifications which would bring us back to a previously visited solution should be forbidden. But it may sometimes be useful to come back to an already visited solution and continue the search in another direction from there. This is realized in Tabu search by keeping a list T containing the last k modifications (k may be fixed or variable). Whenever a modification m is made for moving from s to s', m is introduced in T and its reverse is considered as tabu.

Deciding that some moves are tabu moves may be too absolute: it is shown in [6] that moves to solutions which have not been visited may be tabu. For this reason it should be possible to cancel the tabu status of a move if it seems desirable to do so. This is realized as follows. Let s be the current solution and m a modification which we want to apply to s. A penalization $a(s, m)$ and a threshold value $A(s, m)$ are computed: if $a(s, m) < A(s, m)$, then the tabu status of m (at s) is cancelled. We can for example define $a(s, m) = f(s \oplus m)$ and $A(s, m) = f(s^*)$ where s^* is the best solution encountered so far: the tabu status of m is then cancelled if the solution

$s' = s \oplus m$ is better than the previous best solution s^*. The function A is called the *aspiration function*.

Stopping rules have to be defined. If a lower bound f^* of the minimum value of f is known then the process may be interrupted when the value of the current solution is close enough to f^*. Moreover, the procedure is terminated if no improvement of the best solution s^* found so far has been made during a given number nmax of iterations.

A general description of Tabu search is given in Fig. 1. In the next section we shall describe the adaptation of Tabu search to the course scheduling problem.

4. Adaptation of Tabu search

Many authors have formulated the course scheduling problem as an assignment problem [3,4,9,12]. Guidelines for adapting Tabu search to assignment problems have been described in [12]; the same paper contains a description of the algorithm TATI which is an example of such an adaptation to a course scheduling problem where starting times have to be assigned to courses. For our timetabling problem we cannot use the algorithm TATI since the number of courses and their length are not known in advance.

Let us however formulate our course scheduling problem as an assignment problem where each element from a set S of conflicting objects is assigned to exactly one element of a set P.

The daily quantums of a static topic are fixed in advance. They induce courses of given length. These courses are defined to be *s-objects*.

Initialization
  s := initial solution in X;
  nbiter := 0; (*current iteration *)
  bestiter := 0; (*iteration when the best solution has been found*)
  s^* := s; (*best solution *)
  $T := \emptyset$;
  initialize the aspiration function A;

while ($f(s) > f^*$) **and** (nbiter − bestiter < nmax) **do**
  nbiter := nbiter + 1;
  generate a set V^* of solutions $s_i = s \oplus m_i$ in $N(s)$ such that
    either $m_i \notin T$ or else $a(s, m_i) < A(s, m_i)$;
  choose a best solution s' in V^*;
  update the tabu list T and the aspiration function A;
  if $f(s') < f(s^*)$ **then**
    $s^* := s'$, bestiter := nbiter;
  $s := s'$,

Fig. 1. The Tabu search technique.

We shall consider each time period of a dynamic topic as a second kind of objects which are called *d-objects*.

The set S consists of all s-objects and d-objects, and the set P consists of all available time periods. Now, an available time period of a working day has to be assigned to each object of S; for an s-object the time period represents the starting time of the course. Any assignment satisfying all the constraints (a)–(i) induces a feasible course schedule.

4.1. The set X of feasible solutions

Eiselt and Laporte [7] have divided the requirements into *hard*, *medium* and *soft* ones. The hard requirements have to be respected at all costs while soft ones are only desirable; medium requirements should be satisfied but are nevertheless relaxable.

In most approaches, a timetable is defined to be feasible if it satisfies all hard requirements: soft requirements are modelled as objectives while hard ones appear as constraints. The solution methods designed for dealing with timetabling problems usually consist of two phases: in phase I a heuristic procedure tries to find a feasible timetable, and in phase II the current solution is improved by satisfying more medium or soft requirements and without violating any hard one.

Let us define the following state-space graph G:

- each feasible timetable is a node in G,
- an arc links node x to node y if y is considered as an improvement of x in phase II.

Phase I can be viewed as a generation of a node in G, and phase II is a descent method which moves step by step in the state-space graph G until a local optimum is reached. For large scale timetabling problems, finding a feasible course schedule is not an easy task. Moreover, if phase I succeeds in finding a feasible solution, this solution and the best one are not necessarily in the same connected component of G.

We shall define a connected state-space graph G, and phase II will be replaced by an adaptation of Tabu search techniques. We have now to define the set X of feasible solutions, that is the nodes of the state-space graph G.

There are two ways for handling with a constraint: it can either be satisfied by each solution of the set X of feasible solutions, or else it can be penalized in the objective function. It is to be decided which from among all the requirements will always be satisfied and which one will be penalized.

An object should never be assigned to a time period if such an assignment can only induce course schedules which are not feasible. For example, each object is a portion of a subject x and thus should never be scheduled after the due date or before the release date of x. For this reason, constraint (e) should always be satisfied.

On the other hand, a conflict between two objects should not be a sufficient condition for hindering an assignment. Constraint (a) for example should only be penalized and not always satisfied.

Guided by these ideas, it was decided to penalize constraints (a)–(d) and always satisfy constraints (e)–(h).

Constraint (i) has been split into two distinct requirements: (i1) (respectively (i2)) requires that the minimal (respectively maximal) number of consecutive time periods of each daily quantum of a dynamic topic is not violated.

It was decided to penalize (i1). This choice is justified by the fact that the assignment of exactly one object is changed at each iteration of the Tabu search procedure: for dynamic topics this means that one time period is moved at a time, and removing from a working day a whole daily quantum with minimum value >1 can only be achieved by violating constraint (i1). Constraint (i2) is included in the set of always satisfied requirements since any feasible schedule can always be obtained without violating this requirement.

For a dynamic topic, the set of d-objects which are scheduled on the same day is defined to be a course. Let us consider a course of a dynamic topic satisfying constraint (g). The following modifications can be obtained with several steps of the Tabu search procedure while always satisfying constraint (g):

- the amount of consecutive time periods is not changed but the course is moved to another portion of the working day,
- the daily quantum is increased or decreased.

These justify our decision to never violate requirement (g).

Constraint (h) has to be considered more carefully. If a static topic can only be scheduled in a number n of different working days and if its number of daily quantums is equal to n than constraint (h) should only be penalized since the daily quantums are not necessarily correctly ordered, and satisfying constraint (d) can then only be achieved by violating requirement (h). But this case is not frequent in real-life timetabling problems and it was supposed that such a situation does not occur. Moreover, the definition of a course of a dynamic topic implies that constraint (h) is satisfied for any dynamic topic. For these reasons, it was decided to forbid any solution violating requirement (h).

In summary, the set X of feasible solutions contains any assignment of time periods to the objects of S such that constraints (e), (f), (g), (h) and (i2) are satisfied.

4.2. The objective function

For each topic t, we denote:

- $n(t)$: its number of daily quantums,
- $q_i(t)$: $(i=1,\ldots,n(t))$ one of its courses,
- $r(t)$: the minimum number of time periods in a daily quantum,
- $b_i(t)$: the first time period of course $q_i(t)$,
- $e_i(t)$: the last time period of course $q_i(t)$,
- $B(t)$: the first time period of t $(B(t)=\min\{b_i(t) \mid i=1,\ldots,n(t)\})$,
- $E(t)$: the last time period of t $(E(t)=\max\{e_i(t) \mid i=1,\ldots,n(t)\})$,
- $P(t)$: the set of immediate predecessors of t (if t_1 and t_2 precede t and t_1 precede t_2, then only t_2 belongs to $P(t)$).

If t is a static topic, then $n(t)$ is fixed and $q_i(t)$ must precede $q_j(t)$ for any $j>i$. For a dynamic topic t, $n(t)$ and the values $e_i(t)-b_i(t)$ are variable.

For two objects i and j of S, the value $d(i,j)$ denotes the number of time periods during which i and j are held simultaneously. Since the length of an s-object may be greater than one, it follows that two objects assigned to different time periods can be in conflict.

The class involved in an object i of S is denoted $c(i)$, and $t(i)$, is the teacher which gives the course. Let ntopic denote the number of topics.

For any solution s of X, the following objective function f is defined:

$$
\begin{aligned}
f(s) = p_1 & \sum_{i=1}^{|S|-1} \sum_{\substack{j>i \\ t(i)=t(j)}} d(i,j) \\
& + p_2 \sum_{i=1}^{|S|-1} \sum_{\substack{j>i \\ c(i)=c(j)}} d(i,j) \\
& + p_3 \sum_{t=1}^{\text{ntopic}} \sum_{p \in P(t)} \max\{0, E(p)-B(t)+1\} \\
& + p_4 \sum_{\text{static topic } t} \sum_{i=1}^{n(t)-1} \sum_{j=i+1}^{n(t)} \max\{0, e_i(t)-b_j(t)+1\} \\
& + p_5 \sum_{\text{dynamic topic } t} \sum_{i=1}^{n(t)} \max\{0, r(t)+b_i(t)-e_i(t)-1\}.
\end{aligned}
$$

The parameters $p_1, \ldots, p_5$ give relative weights to constraints (a), (b), (c), (d) and (i1) respectiveiy. A solution s induces a feasible course schedule if and only if $f(s)=0=f^*$.

4.3. *The neighborhood N(s)*

A solution $s' \in X$ is considered as a neighbor solution of s in X if it is obtained by changing the assignment of exactly one object o of a topic t. Let d be the current working day in which o is assigned. In order to satisfy constraints (g) and (h), the set of possible modifications must be restricted to ten types which are represented in Fig. 2:

- If o is an s-object, then it can be assigned to a new working day $d' \neq d$ which does not contain any course of t (type 1), or it can be moved to another portion of d (type 2).
- If t is a dynamic topic and o is the first or the last time period of a course, then it can be assigned to any working day d'. If $d' \neq d$ and d' already contains a course q of t, than o can be placed exactly before the first time period of q (types 3, 4) or exactly after its last time period (types 5, 6). If $d' \neq d$ and d' does not contain any course of t, then o can be assigned to any time period of d' (types 7, 8). If $d'=d$,

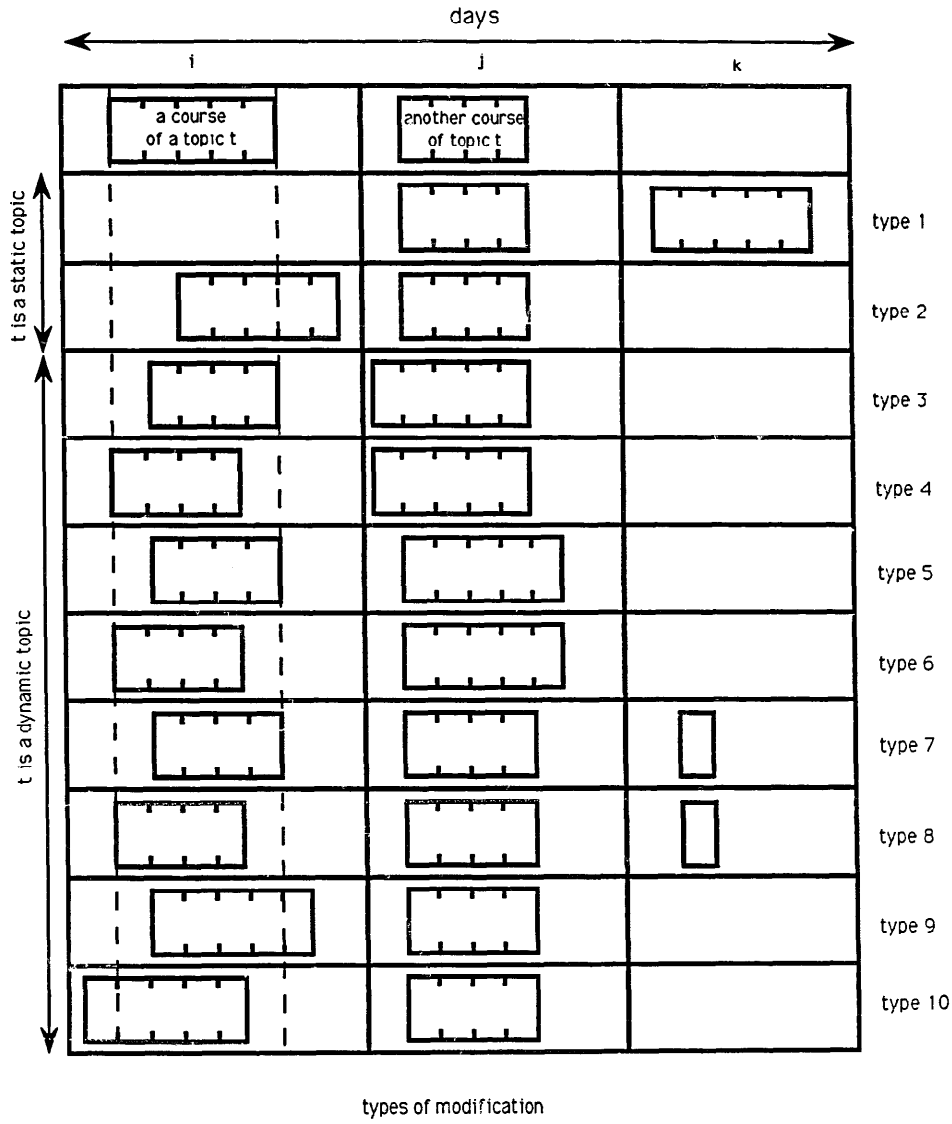

Fig. 2.

then the first time period of a course can be moved exactly after the last one (type 9) and the last time period can be moved before the first one (type 10).
Of course, all these modifications m are acceptable at s only if $s \oplus m = s'$ belongs to X.

4.4. The other ingredients

When an object o currently scheduled on the working day d is moved to a new time period, the pair (o, d) is introduced in the tabu list T. This means that for $|T|$

steps of the Tabu search procedure, it is not allowed to assign any time period of d to o.

The set V^* is generated randomly but contains only new assignments obtained by changing the starting time of an object which gives a strictly positive contribution to $f(s)$. In other words, objects which respect all the constraints are not moved.

In order to allow the tabu status of a move to be cancelled, we consider the penalization function $a(s, m) = f(s \oplus m)$ and the aspiration function $A(s, m) = f(s^*)$.

5. Numerical results

The main purpose of this paper is to develop a new model capable of tackling scheduling problems in which the number of courses and their length are not known in advance. This research has been motivated by a real-life problem in an educational institution in Yugoslavia.

Table 1. A class-teacher timetabling problem

Subjects	Release date	Due date	Set of topics	Immediate predecessors	Total number of time units				Teacher	Class
					s-topic	d-topic				
						Total	Min	Max		
			$T1$	–		5	1	3	$t2$	
$S1$	1	7	$T2$	$T1$	2+1+1				$t5$	
			$T3$	$T2, T6$		3	1	2	$t3$	
			$T4$	–		7	1	3	$t1$	I
$S2$	1	7	$T5$	$T4$	3+3				$t4$	
			$T6$	$T4$	5				$t6$	
$S3$	3	10	$T7$	$T6$		11	3	4	$t1$	
			$T8$	–		12	2	4	$t4$	
$S4$	1	10	$T9$	$T8$		9	3	5	$t1$	
			$T10$	–	3+3+2				$t3$	II
$S5$	1	7	$T11$	$T8, T10$		5	1	2	$t5$	
			$T12$			4	1	2	$t6$	
$S6$	4	8	$T13$	$T11, T12$	3				$t4$	
			$T14$	–		3	1	2	$t2$	
$S7$	1	5	$T15$	$T14$	1+2				$t3$	
			$T16$	–	2+4				$t1$	III
$S8$	1	10	$T17$	$T14, T16, T18$		13	2	5	$t5$	
			$T18$			16	2	4	$t3$	
$S9$	1	10	$T19$	$T18$	4+2+2+1				$t2$	

Hypothetical data from this school were available to test the efficiency of the adaptation of Tabu search. These data are summarized in Table 1. For static topics, the amounts of time periods of each daily quantum are given in an ordered way. Ten working days (two weeks) are available which all contain six consecutive time periods except for the fifth and the tenth working days which contain seven time periods. No working day has a lunch break: such a break can easily be taken into account by dividing the working day into two parts and adding to X a new constraint (h') analogous to constraint (h) and requiring that two courses of a same topic cannot be scheduled in both parts of a working day.

The nineteen topics induce 106 objects (18 s-objects and 88 d-objects). We have set $p_i = 1$ $(i = 1, \ldots, 5)$, $|T| = 10$, $|V^*| = 50$ and $\text{nmax} = |S| = 106$. The value of the initial solution was $f(s) = 6103$; this solution has been randomly generated. A feasible course schedule was obtained after 322 steps: this means that each object has been moved only three times in average.

The iterations when the best solution s^* has been improved are given in Table 2. The last column of this table gives the value of the worst solution encountered between two improvements of s^*.

It is to be observed that the Tabu search procedure has easily reached a solution s with $f(s) = 5$. Only nine iterations without improvement of s^* had to be performed. Note that since V^* is not equal to $N(s)$, when s^* is not improved in one iteration this does not mean that s^* is a local minimum.

The objective function has been considerably increased before reaching an optimal solution (that is a feasible course schedule): a solution s' with $f(s') = 27$ has been visited.

Table 2. Improvements of s^*

	Iteration when s^* has been improved	Number of iterations without improvement of s^*	$f(s^*)$	Value of the worst solution encountered since last improvement of s^*
Initial solution	0	–	6103	–
First "local minimum"	177	0	10	–
	182	4	9	11
	185	2	8	9
	187	1	7	8
	189	1	6	7
	191	1	5	6
	208	16	4	12
	227	18	3	24
	319	91	2	27
	321	1	1	2
	322	0	0	0

The Tabu search procedure needed 341 iterations for finding a feasible course schedule using only nine days. This solution is represented in Fig. 3. Each box is a course and the numbers inside the boxes represent the corresponding topic.

All the parameters p_i, $|T|$, V^* and nmax have to be chosen by the user of the method. It seems that the best length $|T|$ of the tabu list is not proportional to the size of the problem. We have set $|T| = 10$ and recommend this value for other experiments.

The values $|V^*|$ and nmax should be proportional to the number $|S|$ of objects. We have chosen $|V^*| = |S|/2$ and nmax $= |S|$. A user of the method could choose

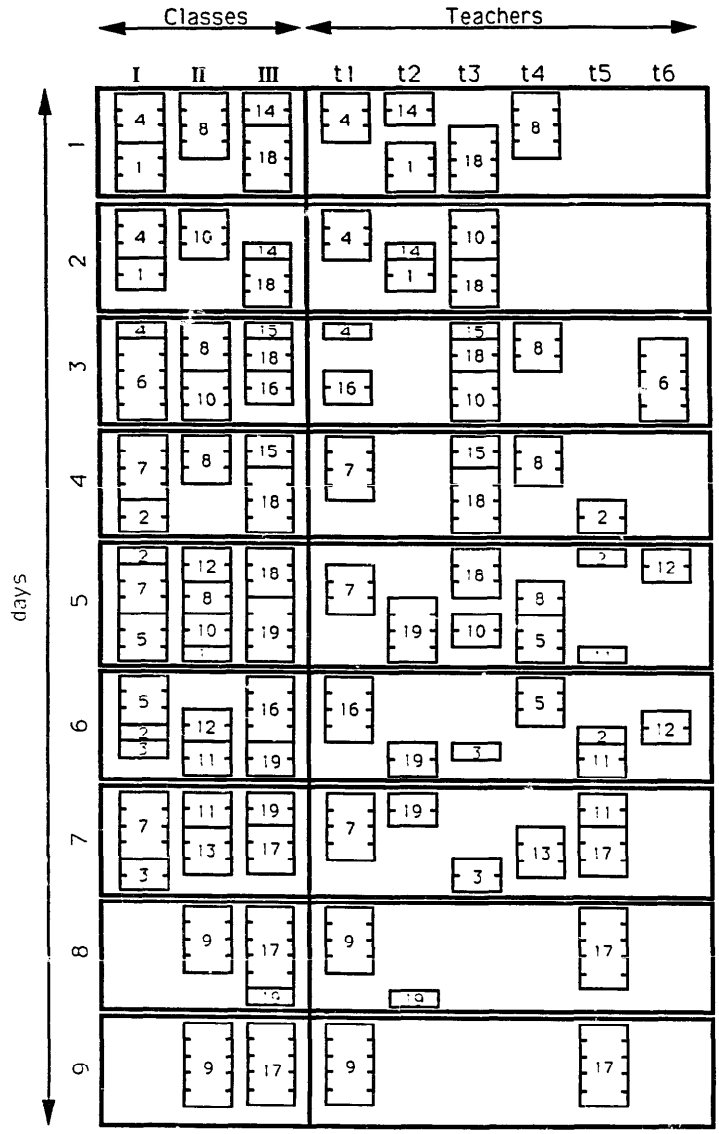

Fig. 3.

the same values but it would be more secure to set nmax $= 2 * |S|$ or even more.

The parameters p_i $(i = 1, \ldots, 5)$ have to be adapted to each particular problem. The constraints which are the most difficult to satisfy should have the greatest weight. In our case we easily found a feasible course schedule by setting $p_i = 1$ for each index i. If the best solution s^*, obtained by setting $p_i = 1$, is not a feasible timetable, then the weights p_i should be chosen proportional to the number of constraints not satisfied in s^*.

Since constraints (i1) and (i2) are not very different, we have tried to penalize (i2) rather than always satisfy this requirement. The method was also able to find a feasible course schedule but it needed 1111 iterations and nmax had to be chosen greater than 402. The whole process is summarized in Table 3 and can be compared with Table 2.

It is tempting to conclude that the guidelines defined in Section 4.1 should always be followed: X should forbid an assignment of a time period to an object only if such an assignment can never induce a feasible course schedule.

6. Extensions

Up to this point, we have described a new model capable of tacking class-teacher

Table 3. Improvements of s^* (constraint (i2) is penalized)

	Iteration when s^* has been improved	Number of iterations without improvement of s^*	$f(s^*)$	Value of the worst solution encountered since last improvement of s^*
Initial solution	0	–	6669	–
First "local minimum"	172	0	17	–
	177	4	16	17
	178	0	14	14
	180	1	13	14
	195	14	12	14
	197	1	11	12
	216	18	10	13
	217	0	9	9
	218	0	8	8
	219	0	7	7
	220	0	6	6
	269	48	5	9
	540	270	4	18
	658	117	3	10
	707	48	2	10
	708	0	1	1
	1111	402	0	21

timetabling problems in which the length of the courses are not known in advance. We shall now briefly show how the model can easily be extended for handling course scheduling problems in general.

In the course scheduling problem, a school offers a collection of courses, but there is no fixed curriculum: each student may choose a certain number of courses. Let us define a class to be a group of students with the same curriculum, and let $c(i)$ denote the set of classes involved in an object i of S (in the class-teacher problem, $c(i)$ contains only one class).

When two courses involving common students are held simultaneously, the cost of the conflict should be proportional to the number of students registered in both courses. For this reason, the second component of the objective f should be replaced by the following cost:

$$p_2 \sum_{i=1}^{|S|-1} \sum_{\substack{j>i \\ c(i)\cap c(j)\neq\emptyset}} d(i,j)\, g(i,j),$$

where $g(i, j)$ denotes the number of students registered in both objects i and j of S.

With these changes, we can now use the adaptation of Tabu search described in Section 4 for handling course scheduling problems in general. It is to be observed that the number of objects does not depend on the number of classes. Thus, even in the case where no two students have the same curriculum, the modifications described in this section do not increase the size of the problem to be solved.

Real-life course scheduling problems may have other constraints then those described in Section 2. For example, courses which involve a large number of students have to be repeated several times during the week and a grouping of the students into the course sections has to be found. There may be options courses, compactness (teachers may prefer to be involved in consecutive time periods), geographical (sufficient time should be provided for moving from one building to another) and preassignment requirements. All these additional constraints can be included in the model by using exactly the same approach as the one described in [12].

In other words, the new adaptation of Tabu search developed in this paper generalizes the procedures TATI and TAG designed in [12] for solving large scale course scheduling problems.

7. Concluding remarks

We have developed a new approach for finding a feasible course schedule. The model can handle timetabling problems with options courses, time windows and preassignment requirements, compactness, geographical and precedence constraints.

Moreover, in their initial requirements, the teachers do not necessarily specify the length of each course. In most approaches described in the literature for tackling course scheduling problems, it is assumed that these lengths are initially fixed by the

scheduler. We do not impose such restrictions which may dramatically reduce the size of the set of feasible course schedules.

It is to be observed that in most timetabling models, the hardest requirements appear as constraints. Our approach is different and is justified by the two following desired properties:

- it should be easy to generate an initial feasible solution, that is a schedule satisfying the requirements which are modelled as constraints,
- starting with any initial feasible solution, it should be possible to reach an optimal solution by using an iterative procedure which, at each step, moves from a feasible solution to a new one by reassigning exactly one course to a different period.

These properties are usually not satisfied when all hard requirements are modelled as constraints.

Finally, it is to be observed that the method developed in this paper can easily be implemented by using object-oriented programming. This certainly contributes to the development of a user friendly decision support for the timetabling operation.

Acknowledgement

I would like to thank Mirjana Cangalovic for suggesting the problem to me and for communicating the example.

References

[1] J. Aubin and J.A. Ferland, A large scale timetabling problem, Comput. Oper. Res. 16 (1989) 67–77.
[2] M. Cangalovic, Exact colouring algorithm for weighted graphs applied to timetabling problems with lectures of different length, European J. Oper. Res. 51 (1991) 248–258.
[3] R.C. Carlson and G.L. Nemhauser, Scheduling to minimize interaction cost, Oper. Res. 14 (1966) 52–58.
[4] F.E. Davis, M.D. Devine and R.P. Lutz, Scheduling activities among conflicting facilities to minimize conflict cost, Math. Programming 6 (1974) 224–228.
[5] D. de Werra, An introduction to timetabling, Europ. J. Oper. Res. 19 (1985) 151–162.
[6] D. de Werra and A. Hertz, Tabu search techniques: a tutorial and an application to neural networks, OR Spektrum 11 (1989) 131–141.
[7] H.A. Eiselt and G. Laporte, Combinatorial optimization problems with soft and hard requirements, J. Oper. Res. Soc. 38 (1987) 785–795.
[8] S. Even, A. Itai and A. Shamir, On the complexity of timetable and multicommodity flow problems, SIAM J. Comput. 5 (1976) 691–703.
[9] J.A. Ferland and S. Roy, Timetabling problem for university as assignment of activities to resources, Comput. Oper. Res. 12 (1985) 207–218.
[10] F. Glover, Future paths for integer programming and links to artificial intelligence, Comput. Oper. Res. 13 (1986) 533–549.
[11] P. Hansen and B. Jaumard, Algorithms for the maximum satisfiability problem, RUTCOR Res. Rept. 43–87, Rutgers University, New Brunswick, NJ (1987).

[12] A. Hertz, Tabu search for large scale timetabling problems, European J. Oper. Res. 54 (1991) 39–47.

[13] A. Hertz and D. de Werra, Using Tabu search techniques for graph coloring, Computing 39 (1987) 345–351.

[14] A. Hertz and D. de Werra, The Tabu search metaheuristic: how we used it, Ann. Math. Artificial Intelligence 1 (1990) 111–121.

[15] A. Tripathy, A Lagrangian relaxation approach to course scheduling, J. Oper. Res. Soc. 31 (1980) 599–603.

[16] M. Widmer and A. Hertz, A new heuristic method for the flow stop sequencing problem, European J. Oper. Res. 41 (1989) 186–193.