Solving the Lecture Scheduling Problem
by the Combination of Exchange Procedure
and Tabu Search Techniques

Le problème d’établissement d’horaires de cours

DO XUAN DUONG — PHAM HUY DIEN*

* Hanoi Institute of Mathematics
Hanoi, Vietnam
{dxduong , phdien}@math.ac.vn

RÉSUMÉ. Le problème connu sous le nom de lectures scheduling problem dans le cas de grandes dimensions est connu pour être très difficile. On utilise habituellement des heuristiques dites d’échange telles que proposées par Ferland. Présentement, nous présentons une autre approche basée sur la recherche Tabou et nous montrons que la combinaison des deux méthodes, Tabou et échange peut fournir une manière efficace pour la solution des problèmes. Nous donnons quelques résultats numériques montrant l’avantage de la méthode combinée par rapport aux méthodes Tabou et échange implémentées séparément.

ABSTRACT. The lectures scheduling problem of large-scale size is considered very hard and cannot be solved by exact deterministic methods. Usually, heuristic procedures are used to deal with this problem and the one based on the Exchange Procedure proposed by Ferland is of prospect. In the present note we describe another approach based on Tabu Search technique and show that the combination of Tabu Search and Exchange Procedure methods may provide an efficient way for the solution of the problem. Some numerical results are given for showing the advantage of the combination method in comparison with the Tabu Search and Exchange Procedure techniques implemented separately.

MOTS-CLÉS : Ordonnancement des lectures, programmation en entiers, recherche Tabou
KEYWORDS: Lectures Scheduling, Timetabling Problem, Integer Programming, Tabu Search
1. Introduction

The timetabling problem is a subject that attached much attention during the last forty years. Among the well known results there are [1],[2],[3],[11]... that deal with various cases of the problem settings. As in [1,9,14], the lectures scheduling problem (Timetabling Problem) discussed in this paper deals with the case when lectures do not have the same length, and scheduling is done for each semester based on a weekly repeated timetable. Each lecture is scheduled at a time period in a week. This problem is difficult, since beside the assignment constraints ensuring that each lecture has to take place at a time period of the week, the model has to account for the side constraints that avoid conflict situations between lectures. Two lectures are said to be in conflict whenever they are held (at least partly) simultaneously and require the same lecturer, or the same classroom, or at least one student is registered in both lectures. The goal is to construct an optimal timetable in the sense that there are fewest unsolvable scheduling conflicts and the preferences of lecturers are fulfilled most.

When the lectures are not all of the same length, the conflict situations between lectures are especially complicated and the number of the constraints increases rapidly with the numbers of lectures and time periods. The problem in question cannot be solved efficiently by the known deterministic techniques, because of the nonlinearity and the huge number of constraints. In practice, some heuristic approaches proved to be more effective. Ferland et al.[1,9,14] have succeeded with the Exchange Procedure technique and gained remarkable results for middle-sized problems (number of lectures is under 500). In [4] we implemented Tabu Search technique for solving the problem and found that, with the appropriate Tabu list length and search neighborhood size, one can get a schedul cost better than that obtained by Exchange Procedure, but for a longer computing time. In the present note we construct a version of combination of Exchange Procedure and Tabu Search techniques and show that the method possesses the advantages of both these two techniques.

The mathematical models of the problem are introduced in Section 2. After giving a short description of Exchange Procedure and Tabu
Search techniques, in Section 3, we describe the method of combination of these two techniques. Section 4 provides some numerical results for demonstration and comparison of all three techniques: Exchange Procedure, Tabu Search and their combination.

2. Mathematical models

According to [7,12], the mathematical model of the problem can be formulated as follows.

Given $n$ lectures and $m$ time periods. Let $x_{ij}$ ($i = 1, \ldots, n; j = 1, \ldots, m$) be the decision variable, that is

$$x_{ij} = \begin{cases} 
1, & \text{if lecture } i \text{ starts at time period } j \\
0, & \text{otherwise}
\end{cases}$$

For any $i = 1, \ldots, n$, denote by $J_i$ a set of time periods that are suitable for lecture $i$ and by $J_{ijk}$ a subset of $J_k$ which are in conflict with the assignment of lecture $i$ to time period $j$.

Assignment constraints can be specified as follows

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

Additional side constraints that describe the conditions for avoiding conflicts can be specified as follows

$$x_{ij} + x_{kl} \leq 1, \quad l \in J_{ijk}, \quad i < k \leq n, \quad 1 \leq i \leq n, \quad 1 \leq j \leq m,$$

The goal is to determine an assignment for each lecture to a time period in order to minimize total assigning cost and fulfill additional side constraints.

Denote by $c_{1ij}$ the cost of assigning lecture $i$ to the time period $j$. This cost must be specified in terms of availability and preferences of the lecturer responsible for lecture $i$. In this model, the preference of
lecturers for each time period is ranging from 1 (highly preferred) to 4 (impossible). For that, the cost structure may be specified as follows

\[ c_{1ij} = \begin{cases} 
  k, & \text{if time period } j \text{ is ranked } k \text{ for lecture } i, \ 1 \leq k \leq 3 \\
  N, & \text{if time period } j \text{ is ranked 4 for lecture } i
\end{cases} \]

where \( N \) is a very large number and, following [9], we may choose \( N = nm + 1 \) for this problem.

In the presence of the assignment cost \( c_{1ij} \), we may assume that \( J_i = \{1, 2, \ldots, m\} \) for all \( i \), since the condition \( j \notin J_i \) could then be replaced by the condition that the time period \( j \) is ranked 4 (impossible) for the lecture \( i \) or, equivalently, \( c_{1ij} = nm + 1 \) for this case.

Then, the total cost of lectures assignment is

\[ F(x) = \sum_{i=1}^{n} \sum_{j=1}^{m} c_{1ij}x_{ij} \]

and the mathematical model can be described as follows

\[ F(x) = \sum_{i=1}^{n} \sum_{j=1}^{m} c_{1ij}x_{ij} \rightarrow \min \]

subject to

\[(P1)\]

(i) \( \sum_{j=1}^{m} x_{ij} = 1, \ i = 1, \ldots, n; \ x_{ij} \in \{0, 1\}, \ i = 1, \ldots, n, \ j = 1, \ldots, m. \)

(ii) \( x_{ij} + x_{kl} \leq 1, l \in J_{ijk}, \ n \geq k > i, \ i = 1, \ldots, n, \ j = 1, \ldots, m. \)

This is the normal type of the lectures scheduling problem and it is an \{0, 1\}-integer programming problem. The first constraints (i) ensure that each lecture is assigned at exactly one time period during a week. The remaining constraints (ii) aim at eliminating the conflict situation between every two lectures.
When the model has to account classroom availability, additional side constraints must be introduced. For this purpose, let us denote

- $B$ - the number of classrooms types,
- $K_{bj}$ - a set of lectures which take place at the time period $j$ and require a classroom of type $b$,
- $Q_{bj}$ - the number of classrooms of type $b$ that can be used at the time period $j$.

Additional side constraints ensuring that the utilization of classrooms does not exceed their availabilities can be specified as follows

$$
\sum_{i \in K_{bj}} x_{ij} \leq Q_{bj}, \ j = 1, \ldots, m, \ b = 1, \ldots, B.
$$

The mathematical model can now be described in the following form

$$
F(x) = \sum_{i=1}^{n} \sum_{j=1}^{m} c_{1ij}x_{ij} \rightarrow \min
$$

subject to

(P2) \quad (i) \quad \sum_{j=1}^{m} x_{ij} = 1, \ i = 1, \ldots, n; \ x_{ij} \in \{0, 1\}, \ i = 1, \ldots, n, \ j = 1, \ldots, m.

(ii) \quad x_{ij} + x_{kl} \leq 1, \ l \in J_{ijk}, \ n \geq k > i, \ i = 1, \ldots, n, \ j = 1, \ldots, m.

(iii) \quad \sum_{i \in K_{bj}} x_{ij} \leq Q_{bj}, \ j = 1, \ldots, m, \ b = 1, \ldots, B.

Note that, the condition $l \in J_{ijk}$ in the constraints (ii) is not a linear relation, therefore the Problem (P2) is a nonlinear one.

3. Solution Technique

The most difficult thing in the above models is how to deal with the constraints of type (ii) because with each lecture $i$ at time period $j$ there
is a set of lectures that conflict with lecture $i$ at that time. To overcome this difficulty, a penalty technique is introduced to include those constraints into the objective function. The constraints of type (iii) are also included in the similar way. Such a penalty technique also provides a flexible way to deal with the constraints, giving priority to one type of constraints or the other by changing their penalty coefficients.

The mathematical model (P2) can now be formulated as follows

$$\sum_{i=1}^{n} \sum_{j=1}^{m} \left[ c_{1ij} x_{ij} + \frac{1}{2} \sum_{k=1}^{n} \sum_{l=1}^{m} c_{2ijkl} x_{ij} x_{kl} + c_{3} \sum_{b=1}^{B} \max \left\{ 0, \sum_{i \in K_{bj}} x_{ij} - Q_{bj} \right\} \right] \rightarrow \min$$

subject to

$$\sum_{j=1}^{m} x_{ij} = 1, \quad i = 1, \ldots, n; \quad x_{ij} \in \{0,1\}, \quad i = 1, \ldots, n, \quad j = 1, \ldots, m.$$  

Here

$$c_{2ijkl} = \begin{cases} M & \text{when } l \in J_{ijk} \\ 0 & \text{when } l \notin J_{ijk} \end{cases}$$

with $M$ is a very large number and $c_{3}$ is the penalty coefficient related to the shortage of classrooms. Clearly, $c_{3}$ can be controlled by the user in relation with $c_{2ijkl}$, depending on what constraints should be given priority. Clearly (P3) is a nonlinear integer programming problem. It should be noted that the problems (P2) and (P3) are not equivalent. If the (P2) is solvable, then the both (P2) and (P3) have the same a set of optimal solutions. When (P2) is unsolvable, (P3) may be still solvable but its solution may be unacceptable for (P2).

The lectures scheduling problem (P2) and/or (P3) cannot be solved efficiently by the known deterministic techniques, because of the nonlinearity of the problem and the number of constraints for avoiding conflicts between lectures is too large. For instance, scheduling 10 lectures for only 5 successive time periods may generate the problem with several hundreds of constraints. Moreover, the number of these constraints grows very fast when the size of the problems (the number of lectures and time periods) increases. In practice, some heuristic approaches proved to be more effective for solving the problem in question. Ferland et
al.[1,9,14] have succeeded with the Exchange Procedure technique and gained remarkable results.

3.1. The Exchange Procedure Technique

According to [9], the cost for starting lecture $i$ at time period $j$ (specified in terms of the availability $c_{1ij}$ and the conflict created by this assignment) can be expressed as:

$$r_{ij} = c_{1ij} + \sum_{k=1}^{n} \sum_{l=1}^{m} c_{2ijkl}x_{kl} \quad 1 \leq i \leq n, \quad 1 \leq j \leq m.$$ 

Then, $s_{ij} = r_{ij} - r_{ij}^*$ measures the reduction in the value of the objective function if lecture $i$ starts at the time period $j$ instead of $j^*$. The single reassignment inducing the largest improvement in the objective function is determined by finding $i^*, j^*$ such that

$$s_{i^*, j^*} = \min_{1 \leq i \leq n, \quad 1 \leq j \leq m} \{s_{ij}\},$$

and reassigning $i^*$ from $j^*$ to $j^*$. This generates a new feasible solution $x$ of (P3) for which the quantities $r_{i^*j^*}$ and $s_{i^*j^*}$ can easily be computed. For example, $s_{i^*j^*}$ can be evaluated in terms of the $s_{ij}$ as follows:

$$s_{i^*j^*} = c_{1i^*j^*} - c_{1i^*j^*}^* + c_{2i^*j^*j^*} - c_{2i^*j^*j^*}^*,$$

$$1 \leq i \leq n, \quad 1 \leq j \leq m, \quad i \neq i^*.$$

and, for $i = i^*$, one has

$$s_{i^*j^*} = c_{1i^*j^*} - c_{1i^*j^*}^* + \sum_{k=1}^{n} \sum_{l=1}^{m} [c_{2i^*j^*kl} - c_{2i^*j^*kl}^*] \quad 1 \leq j \leq m.$$

If

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

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

\[
\Delta_{i^*j^*k^*l^*} = s_{i^*j^*} + s_{k^*l^*} + c_{2i^*j^*k^*l^*} - c_{2i^*j^*k^*l^*} - c_{2i^*j^*k^*l^*} + c_{2i^*j^*k^*l^*} \\
= \min_{1 \leq i, k \leq n} \{ s_{ij} + s_{kl} + c_{ijkl} - c_{ijkl} - c_{ijkl} + c_{ijkl} \}.
\]

If \(\Delta_{i^*j^*k^*l^*} < 0\) then the double reassignment is executed, where \(i^*\) is reassigned from \(j^*\) to \(j^*\) and \(k^*\) from \(j^*\) to \(l^*\). If \(s_{i^*j^*} = \Delta_{i^*j^*k^*l^*} = 0\) and if the local optimality conditions are not satisfied, then a single reassignment is executed whenever possible, inducing no improvement of the objective function.

The detailed algorithm can be found in [9]. Here we provide some numerical tests to demonstrate how the algorithm works. The numerical results listed below are applied to timetabling problem (P3) with 50 time periods (normally for one week), where the number of lectures is in the range from 300 to 700 (suitable for an university). The initial data are chosen at random. All the tests are implemented on a Celeron 433 MHz CPU.

**Test 1 :The case with 300 lectures**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>422241731</td>
<td>243982084</td>
<td>42.22%</td>
</tr>
<tr>
<td>5</td>
<td>422241731</td>
<td>76526019</td>
<td>81.88%</td>
</tr>
<tr>
<td>8</td>
<td>422241731</td>
<td>7202799</td>
<td>98.29%</td>
</tr>
<tr>
<td>258</td>
<td>422241731</td>
<td>224</td>
<td>99.99%</td>
</tr>
</tbody>
</table>

**Test 2 :The case with 400 lectures**
During the process of a lot of numerical tests implemented for the exchange method, we observed that only a little time was spent for executing the single reassignment procedure, and most of the time was spent

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>775159866</td>
<td>619407771</td>
<td>20.09%</td>
</tr>
<tr>
<td>5</td>
<td>775159866</td>
<td>443849016</td>
<td>42.74%</td>
</tr>
<tr>
<td>10</td>
<td>775159866</td>
<td>207069786</td>
<td>73.29%</td>
</tr>
<tr>
<td>20</td>
<td>775159866</td>
<td>45015510</td>
<td>94.19%</td>
</tr>
<tr>
<td>490</td>
<td>775159866</td>
<td>30610666</td>
<td>96.05%</td>
</tr>
<tr>
<td>971</td>
<td>775159866</td>
<td>29710358</td>
<td>96.17%</td>
</tr>
<tr>
<td>1450</td>
<td>775159866</td>
<td>28810053</td>
<td>96.28%</td>
</tr>
</tbody>
</table>

Test 3: *The case with 500 lectures*

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>1122676273</td>
<td>824676600</td>
<td>26.54%</td>
</tr>
<tr>
<td>10</td>
<td>1122676273</td>
<td>620308192</td>
<td>44.75%</td>
</tr>
<tr>
<td>20</td>
<td>1122676273</td>
<td>321408174</td>
<td>71.37%</td>
</tr>
<tr>
<td>35</td>
<td>1122676273</td>
<td>134145480</td>
<td>88.05%</td>
</tr>
<tr>
<td>787</td>
<td>1122676273</td>
<td>110737588</td>
<td>90.14%</td>
</tr>
<tr>
<td>1526</td>
<td>1126766273</td>
<td>108036686</td>
<td>90.37%</td>
</tr>
</tbody>
</table>

Test 4: *The case with 700 lectures*

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>2176028882</td>
<td>1744784622</td>
<td>19.82%</td>
</tr>
<tr>
<td>20</td>
<td>2176028882</td>
<td>1435081013</td>
<td>34.05%</td>
</tr>
<tr>
<td>40</td>
<td>2176028882</td>
<td>983129905</td>
<td>54.82%</td>
</tr>
<tr>
<td>80</td>
<td>2176028882</td>
<td>549184662</td>
<td>74.76%</td>
</tr>
<tr>
<td>110</td>
<td>2176028882</td>
<td>445649930</td>
<td>79.52%</td>
</tr>
<tr>
<td>1574</td>
<td>2176028882</td>
<td>434846291</td>
<td>80.02%</td>
</tr>
<tr>
<td>3111</td>
<td>2176028882</td>
<td>433945981</td>
<td>80.05%</td>
</tr>
</tbody>
</table>
for the execution of the double reassignment procedure without any progress of the objective function value. One can conclude that the single reassignment steps are carried out quickly and essentially improve the objective function value, while the double reassignment steps can hardly improve the minimizing process and are very consuming time. Another observation is that the single reassignment steps usually lead to a local minimum and therefore, in many cases, cannot reach the best possible result.

3.2. Tabu Search technique

Tabu search technique allows the search to move away from the local minimum in order to search more extensively in the feasible domain. A short term Tabu list is used as a safeguard against cycling. The length of the Tabu list (TL) is a regulable parameter depending on specific properties and the size of the problem in hand. This list (TL) includes the most recent modifications used. When the Tabu list is full, the oldest element will be eliminated if a new one is brought in. By this way, the algorithm allows a chance to come back to the points, which are formerly forbidden. With the current solution $x$, a neighborhood $N(x)$ of $x$ is generated for searching procedure. Only points of the set $N(x)\setminus TL$ are subject to search. If a searched point $a \in N(x)\setminus TL$ is better than $x$, then the current solution is moved to $a$, and $x$ is included to the Tabu list together with all the ”related” points. In the other case, the point $a$ is included in a list of searched points around $x$, which is denoted by $N^*(x)$. Since the size of the neighborhood $N(x)$ increases rapidly with the problem size, it may become unreasonable to scan the whole neighborhood to identify the best neighboring solution. In implementing Tabu search method, we often restrain the search to a subset of $N(x)$, by limiting the number of points of the set $N^*(x)$. When the number of elements of $N^*(x)$ reaches the limit value $NV$, then $x$ is included to the Tabu list and the following tasks are taking place:

- The best point of $N^*(x)$ is chosen as the current solution;
- The best current point parameter is updated by $x$ if it is not better than $x$.

The algorithm stops if one of the following situations happens:
– The number of successive iterations with no improvement reaches a certain given value, or
– The total computing time reaches a limit value, or
– The number of iterations reaches a certain maximal value.

The best current point is taken to be the final solution.

A more detailed description of the algorithm of the Tabu Search method is given in [4]. Here we provide some numerical tests to demonstrate how the algorithm works. Numerical results listed in the table below are carried out for the timetabling problem (P3) with the same data set. With the Tabu List length being 100 and the Neighborhood Size being 500, the algorithm stops if no improvement is obtained after a certain number of moves.

<table>
<thead>
<tr>
<th>Num. of Lectures</th>
<th>Com. Time (in second)</th>
<th>Init. Val. of Obj. Function</th>
<th>Fin. Val. of Obj. Function</th>
<th>Falling Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>300</td>
<td>27</td>
<td>42224173</td>
<td>3601421</td>
<td>99.14%</td>
</tr>
<tr>
<td>400</td>
<td>66</td>
<td>775159866</td>
<td>17160097</td>
<td>97.73%</td>
</tr>
<tr>
<td>500</td>
<td>117</td>
<td>1122676273</td>
<td>97233022</td>
<td>91.33%</td>
</tr>
<tr>
<td>700</td>
<td>352</td>
<td>2176028882</td>
<td>404235961</td>
<td>81.42%</td>
</tr>
</tbody>
</table>

Comparing the results above with those obtained by Exchange Procedure method in the previous part we see clearly that the Tabu Search method has certain advantages. It produces a lower cost solution in a shorter computing time. However, a detail analysis of the algorithm, particularly from the implementation aspect, shows that the Tabu Search technique is better than the Exchange Procedure when overcoming a local minimum, but it is slower than the latter in reaching a local minimum. In general, there is no guarantee that the Tabu Search procedure comes to a local minimum. This observation leads to the following technique.

### 3.3. Combination method

As we have observed before, the double reassignment steps of the Exchange Procedure can hardly bring improvement to the minimizing
process and it should be taken only if one has nothing to do further on. Our resolution is to replace this step by another one based on Tabu Search technique. The method of combination of the Exchange Procedure (EP) and Tabu Search (TS) techniques can be described as follows. From the initial point, the Exchange Procedure starts and continues until the objective function cannot be improved by a single reassignment. When this situation occurs (it is likely that a local minimum is reached), the Tabu Search procedure comes after to continue the process. When the Tabu Search procedure stops (by some criterion), the Exchange Procedure starts again and goes on to find a new local minimum. The process is continuing until the number of alternative procedures reaches some given value, or if both procedures cannot improve the objective function value after certain amount of iterations. It should be noted that, for deployment of the Tabu Search procedure, a number of parameters must be predefined (e.g., the length of Tabu list, the size of search neighborhood, the number of successive iterations with no improvement, the number for total computing time and the maximal number of iterations).

4. Numerical results and comparison

The numerical results listed below are applied to problem (P3) of 50 time periods (for a week). The number of lectures is in the range from 600 to 1000. The initial data are taken at random. The tests are performed on the Celeron 433MHz CPU. For comparison, we give the numerical results for the combination method together with the ones for the EP and TS methods implemented separately. As it concerns the EP method, we bypass the double reassignment step since it can hardly improve the minimizing process.

Results of numerical test for the Exchange Procedure method
The results of numerical test for the Tabu search method

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>600</td>
<td>1637648604</td>
<td>327710386</td>
<td>79.98%</td>
<td>54</td>
</tr>
<tr>
<td>800</td>
<td>2835949588</td>
<td>1001135822</td>
<td>64.69%</td>
<td>91</td>
</tr>
<tr>
<td>1000</td>
<td>4506908668</td>
<td>1901437231</td>
<td>57.81%</td>
<td>177</td>
</tr>
</tbody>
</table>

Table 1.

The results of numerical test for the Combination method

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>600</td>
<td>1637648064</td>
<td>100</td>
<td>236779787</td>
<td>85.54%</td>
<td>212</td>
</tr>
<tr>
<td>800</td>
<td>283594588</td>
<td>100</td>
<td>650918635</td>
<td>77.05%</td>
<td>508</td>
</tr>
<tr>
<td>1000</td>
<td>4506908668</td>
<td>100</td>
<td>1336948372</td>
<td>70.33%</td>
<td>1011</td>
</tr>
</tbody>
</table>

Table 2.

A lot of numerical tests with different data sets provide the results analogous to those listed above (Here, as it concerns the Tabu Search method, the various tests show that the reasonable length of the tabu list should be in the range from 50 to 200 and the reasonable neighborhood size should be in the range from 200 to 1000, depending on the size of problem). Comparing the results given in the Table 3 with those given in Table 1 and Table 2, we easily see that the objective function value is essentially improved in comparison with the Exchange Procedure, and the computing time is essentially reduced in comparison with the Tabu Search
method. Therefore, we may conclude that the combination technique of the Exchange Procedure and Tabu Search methods is more efficient than their separate implementations.

The authors are grateful to Prof. Ferland and Dr. Pham Canh Duong for helpful suggestions. They also express the appreciation and thanks to Pham Ngoc Hung for handling various numerical tests.

Références


