Modelling and solving an acyclic multi-period timetabling problem

Mirjana Čangalović
Faculty of Organizational Sciences, University of Belgrade, Belgrade, Yugoslavia

Jan A.M. Schreuder
Faculty of Applied Mathematics, University of Twente, Postbus 217, 7500 AE Enschede, Netherlands

Received 21 December 1989
Revised 19 July 1990

Abstract

In this article a special case of the class-teacher timetabling problem is described. This case takes into consideration a partial ordering between the topics of the curriculum and special requirements in respect to their daily lectures. The problem is modelled as a discrete lexicographic optimization problem. A heuristic procedure for solving the problem is developed. The method represents a combination of two different approaches: on the general level a heuristic approach typical for resource constrained project scheduling and on the daily level a reduction to graph colouring.

Keywords. Timetabling, education, project management, lexicographic optimization, heuristic.

1. Introduction

A huge variety of timetabling problems has been described and many different approaches to their solving have been proposed in the operational research literature [10,11,6]. These problems depend on types of schools and educational systems and they can be mutually very different. Therefore, there does not exist a universal timetabling model which could be applied everywhere.

Most of the already defined timetabling problems belong to one of the following groups: the class-teacher problems [8,5], the course scheduling problems [12] and the exam scheduling problems [9].
The most frequent approaches which have been used for solving timetabling problems are: reduction to colouring of graphs, mathematical programming, and heuristics.

The largest group of timetabling problems is the group of class-teacher problems. The well-known basic class-teacher problem can be described as follows. Given are a set of classes, a set of teachers, a set of time periods and a requirement matrix \( r_{ij} \), where \( r_{ij} \) is the number of lectures given to a class \( i \) by a teacher \( j \). The length of each lecture is equal to one period. The availability of each class and teacher for each period is known. The problem is to assign lectures to the given set of periods in such a way that all requirements \( r_{ij} \) are satisfied, the availabilities of the classes and teachers are not violated and no class or teacher is involved in more than one lecture at a time.

The basic class-teacher problem is proved to be NP-complete except in the case when all the classes and all the teachers are always available [8].

In this article a real-life timetabling problem which represents a special case of the basic class-teacher problem with additional constraints is described. It is modelled as a discrete priority optimization problem. An approach to solve the problem based on its analogy with a preemptive, resource constrained multi-project scheduling is applied. Using this analogy, the associated network model of the problem is defined and a heuristic method for its solution is developed.

2. Problem description

In a real educational institution the following problem has arisen. There are several classes (groups of students) and each class should follow its own fixed curriculum. Given are a set of working days and for each day there is a set of consecutive time periods or school hours (usually 45 minutes per period) available for teaching-lectures. For each class the 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 should ideally be completed are specified. Each subject is divided in a given set of topics. A topic consists of a sequence of lectures. For each class there exists a partial ordering in the set of topics of the curriculum. The ordering specifies a precedence relation between the topics. Therefore, for each topic a set of its immediate predecessors is given. In order to achieve better educational results, two topics which have a precedence relation should not be scheduled at the same day.

Topics of different classes have no precedence relations.

The length of a lecture is the number of consecutive periods required for its completion. With respect to this length, there exist the two following types of topics.

- A static topic (s-topic) is realized if the lectures have a given sequence and have fixed lengths. For example, a topic could need the following three lectures: starting with two periods of theory, followed by an application which has a length of four, and closing with two periods of discussion.
A dynamic topic (d-topic) is a topic for which the number of lectures and their lengths are not fixed in advance. These numbers, however, are bounded. For such a topic the total number of periods required for its completion and the minimal and the maximal length of its lectures are prescribed. For example, a topic should be realized in ten periods with lectures which have a length between two and four.

Each topic can have at most one lecture per day which should be realized in a non-interrupted way. There is one teacher for each topic and a teacher can teach one or more topics to one or more classes. The teachers and the classes are always available during the given working days.

The problem is to assign the lectures of all topics from the given class curriculums to the given set of days with their periods such that the following requirements are fulfilled.

(a) Each topic is completed by its teacher according to its type satisfying the corresponding partial ordering.
(b) The release dates and due dates for all subjects are satisfied.
(c) Each class or teacher is involved in at most one topic at a time.
(d) The number of free time periods for the classes should be minimized.

A hypothetical example of three class curriculums with ten available working days is given in Table 1. The maximal daily allowed number of consecutive time periods is equal to seven for the fifth and the tenth day and six for the other days.

Table 1. Example class curriculums

<table>
<thead>
<tr>
<th>Class</th>
<th>Subject</th>
<th>Release date</th>
<th>Due date</th>
<th>Set of topics</th>
<th>Immediate predec.</th>
<th>Total number of periods</th>
<th>Teacher</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>s-topic</td>
<td>d-topic</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Total</td>
<td>Min</td>
</tr>
<tr>
<td>I</td>
<td>S1</td>
<td>1</td>
<td>7</td>
<td>T1</td>
<td>–</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>T2, T3</td>
<td></td>
<td>N3</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>T5, T6</td>
<td></td>
<td>N5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>S2</td>
<td>1</td>
<td>7</td>
<td>T4</td>
<td>–</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>T5</td>
<td></td>
<td>N4</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>T7, T8</td>
<td></td>
<td>N1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>S3</td>
<td>3</td>
<td>10</td>
<td>T6</td>
<td>T4</td>
<td>11</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>T7</td>
<td></td>
<td>N1</td>
<td></td>
</tr>
<tr>
<td>II</td>
<td>S4</td>
<td>1</td>
<td>10</td>
<td>T8</td>
<td>–</td>
<td>12</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>T9</td>
<td>T8</td>
<td>9</td>
<td>&gt;</td>
</tr>
<tr>
<td></td>
<td>S5</td>
<td>1</td>
<td>7</td>
<td>T10</td>
<td>–</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>T11</td>
<td>T8, T10</td>
<td>5</td>
<td>!</td>
</tr>
<tr>
<td></td>
<td>S6</td>
<td>4</td>
<td>8</td>
<td>T12</td>
<td>–</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>T13</td>
<td>T11, T12</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>III</td>
<td>S7</td>
<td>1</td>
<td>5</td>
<td>T14</td>
<td>–</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>T15</td>
<td>T14</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>S8</td>
<td>1</td>
<td>10</td>
<td>T16</td>
<td>–</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>T17</td>
<td>T14, T16, T18</td>
<td>13</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>S9</td>
<td>1</td>
<td>10</td>
<td>T18</td>
<td>–</td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>T19</td>
<td>T18</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

An acyclic multi-period timetabling problem
The described problem is a variant of the basic class-teacher problem mentioned in Section 1, with very complicated additional constraints. In most of the already considered class-teacher problems there exists a scheduling cycle, usually a week. As our problem takes care of the orderings between topics and some special requirements with respect to their daily lectures, each day in the timetable can have a different schedule. Therefore, this problem is called \textit{acyclic}.

It should be pointed out that, in the case of only one class, the acyclic problem can be reduced to the following simplified problem.

Assign the lectures of all topics for the class to the given working days such that requirements (a) and (b) are satisfied and the sum of the lengths of all lectures assigned to the same day is not greater than the total number of available periods for this day. The assigned lectures could be trivially scheduled to the given periods in any arbitrary order without causing free time periods. In such a way conditions (c) and (d) are also satisfied.

The reduced problem is not a typical timetabling problem when the teachers and all constraints related to them are ignored. Nevertheless, this problem is not trivial and worth investigating for its own sake. This is in contrast with the basic class-teacher problem which, in the case of one class, is trivial.

We mention that in the acyclic problem the teaching rooms are not considered. The reason is that each class has its own rooms and, therefore, assigning its lectures to rooms is easily performed after the assignment to periods.

3. \textbf{Multicriteria approach to modelling}

The acyclic problem belongs to a class of practical combinatorial problems which are very difficult both for modelling and for solving. First, the problem is not well defined. It does not include some of the practical requirements which cannot be easily caught in formulas. The requirements (a)–(d) can have various degrees of strictness strongly dependent on the individual view of the organizers in the educational institution. Therefore, during the modelling of the problem a certain amount of fuzziness and subjectivity can be expected. Secondly, the models are of a large scale and, consequently, the application of exact solution methods seems not very efficient.

A general principle for modelling difficult combinatorial problems with the previous characteristics is the following [7].

First, the set of all requirements of a problem is partitioned into three groups with respect to the various degrees of strictness.

- \textbf{Hard requirements}: must be satisfied at all costs.
- \textbf{Medium requirements}: should be satisfied, although they can be relaxed in some cases.
- \textbf{Soft requirements}: should be satisfied if the other requirements allow this.

These requirements can be more directly translated to timetabling problems [11]. The hard requirements should lead to physically feasible timetables, called work
level. The medium ones are related to educational conditions, called acceptable level. The soft requirements at last have to do with preferences, called performance level.

Further, the problem is modeled as a discrete multi-criterion optimization problem such that the hard requirements are expressed as constraints, whereas the soft requirements are modeled as objectives. Each medium requirement can appear either as a constraint or as an objective, depending on the nature of the problem investigated or the interpretation of the researchers.

We used the above principle in order to model the acyclic problem.

The requirements (a)-(d) are assigned to the following groups.

- Hard requirement:
  \( r_1 \): Each class or teacher can have no more than one lecture at a time.

- Medium requirements:
  \( r_2 \): the given set of working days and their periods,
  \( r_3 \): the types of topics and their teachers,
  \( r_4 \): the orderings between the topics,
  \( r_5 \): the release dates of the subjects and
  \( r_6 \): the due dates of the subjects should be respected.

- Soft requirement:
  \( r_7 \): as few free time periods as possible per day for the classes.

The requirements \( r_1 - r_5 \) are treated as constraints, whereas \( r_6 \) and \( r_7 \) are expressed as objectives.

The above classification of the requirements represents a fuzzy problem in itself. It is accomplished by consulting the main organizers of the teaching process in the educational institution and according to their subjective evaluation and practical experiences.

We mention that in practice the realization of \( r_6 \) is looked upon as very important. However, the due dates are very often violated in timetables constructed by hand. Therefore, \( r_6 \) is relaxed by expressing it as an objective. In this way, the solution of the acyclic problem is a little less complicated.

We show briefly how the requirements \( r_6 \) and \( r_7 \) can be formally modeled as objectives \( \phi_1 \) and \( \phi_2 \), respectively, Let \( tt \) be a feasible timetable, i.e., it satisfies \( r_1 - r_5 \). Then,

- \( \phi_1(tt) = \max_i \max(0, CD_i(tt) - DD_i) \), where the maximum is established over all the subjects \( i \) (independent of a class), \( DD_i \) is the due date of a subject \( i \) and \( CD_i(tt) \) is the day at which \( i \) is completed according to the timetable \( tt \);

- \( \phi_2(tt) = \max_d \max_k FP_k(d,tt) \), where the maxima are established over all the available days \( d \) and all the classes \( k \) whereas \( FP_k(d,tt) \) is the total number of free time periods for a class \( k \) at a day \( d \) according to the timetable \( tt \) [3].

In practice the degree of importance of \( r_6 \) and \( r_7 \) is not equal. According to the given classification, it is more important to satisfy \( r_6 \) than \( r_7 \). Therefore, the acyclic problem can be modeled as the following discrete lexicographic optimization problem.
Lex min \{\phi_1(tt), \phi_2(tt)\}, \\
s.t. \quad r_s(tt) \leq 0, \quad s=1,2, ..., 5, \\

(1)

where (1) represents the fact that \( \phi \) satisfies \( r_s \).

Each optimal solution \( t^* \) of model (1) for which \( \phi_1(t^*) = 0 \), can be considered as a solution of the acyclic problem. If \( \phi_1(t^*) > 0 \) or a feasible timetable does not exist, then the problem has no solution.

Model (1) could be formulated as a discrete lexicographic linear programming model. However, it would have large dimensions and, therefore, it could not be efficiently solved by a successive application of one of the existing exact linear integer procedures. Furthermore, it makes little sense to go through all the trouble of using such a procedure if the structure of (1) is based on fuzziness. Altogether, for solving the acyclic problem, modelled as (1), a heuristic method is developed in Section 6.

4. Analogy with multi-project scheduling

In order to solve problem (1) hierarchically, it is decomposed into the two following levels.

- **General level** - assigning lectures to working days.

  For each topic the days at which it is realized and the length of its daily lectures should be determined such that the constraints \( r_2, r_5 \) are satisfied and the objective \( \phi_1 \) is minimized. Furthermore, for each teacher or class the sum of the lengths of all its lectures assigned to the same day should not be greater than the total number of available periods for that day. This is a necessary condition for the existence of feasible daily timetables [6].

  Obviously, on the general level all requirements related to assigning lectures to periods—\( r_1 \) and \( \phi_2 \)—are ignored.

  The basic notions of the acyclic problem correspond to the following notions of the well-known resource constrained project scheduling problem [1].
  - Curriculum of a class \( \rightarrow \) project,
  - subject \( \rightarrow \) job,
  - topic \( \rightarrow \) activity,
  - teacher \( \rightarrow \) resource,
  - period in which one lecture can be given \( \rightarrow \) resource unit,
  - working day \( \rightarrow \) time period,
  - length daily lecture topic \( \rightarrow \) amount resource units assigned to activity per time period,
  - release and due dates subjects \( \rightarrow \) release and due dates subsets activities.

Starting from the above associations the problem at general level can be considered as a variant of the so-called preemptive multi-project scheduling with renewable resources, fixed and discrete resource requirements and a time optimality criterion [13]. In such problems there are several projects, where a project is a set
of activities partially ordered by a precedence relation. The activities can be arbitrarily interrupted during the realization (preemptive). Each activity requires one or several different resource types. These resource requirements per time period are either fixed in advance during the prescribed activity duration (here: s-topics) or belong to the given finite sets (here: d-topics). The total amount of each resource type is bounded at every moment.

The activities of all projects should be scheduled in time and the resources should be allocated to activities such that the precedence relations and resource constraints are satisfied and a time criterion is optimized (here: function $\phi_1$). Moreover, the problem at general level includes some additional constraints: time limits for some subsets of activities and bounds for the total amount of resources of each project.

Therefore, the problem is very specific and shows not enough resemblance with the existing project scheduling literature nor do there exist methods which could be applied to solve it.

- **Daily level - finding daily timetables.**

At each day all lectures scheduled at this day on the general level should be assigned to the given set of consecutive periods of the day such that constraint $r_1$ is satisfied and objective $\phi_2$ is minimized.

The lectures have different lengths and should not be interrupted. Therefore, on daily level one has to solve a sequence of daily timetabling problems with lectures of different lengths [3]. In each such a problem a so-called minimal free time timetable should be found. This is a feasible daily timetable for which the function

$$\max_{i} (z_i + 1 - p_i - d_i)$$

is minimal, where the maximum is established over all the classes $i$, $p_i$ and $z_i$ are, respectively, the initial and the final school hour for a class $i$ in the timetable, and $d_i$ is the sum of the lengths of the lectures of class $i$ that day.

We mention that, in the case of one class, the acyclic problem is reduced to just the general level. Furthermore, if all topics are s-topics, all release dates are one and all due dates are equal, then this problem can be considered as the well-known resource constrained single-project scheduling (RCPS) problem. RCPS with minimizing the duration of a project under fixed resource requirements is known to be NP-hard [1].

Therefore, in general the acyclic problem is NP-hard and consequently, a polynomial time exact algorithm for its solution is highly unlikely to exist. For this reason a heuristic algorithm for solving the problem is developed in Section 6.

5. **Associated network model**

In a similar way as for project scheduling problems, to each class of the acyclic problem an activity-on-arc network is associated according to the following rules. Each topic is represented as a directed arc. The initial and the terminal node of
the arc are respectively called the starting and the completion event of the topic. The ordering between topics obeys the well-known conventions for project scheduling networks.

To each arc the minimal duration of the corresponding topic is associated. It is defined as the minimal number of days within which the topic can be completed with respect to its type. Arcs which represent dummy topics (introduced only for properly expressing the order between the topics), have value 0.

Let \( RD_i \) and \( DD_i \) be respectively the release and due date of a subject \( i \) and \( TD \) be the total number of available working days.

A topic of a subject which has no predecessors (successors) in the set of all the topics corresponding to the subject according to the given ordering, is called initial (terminal) topic of the subject.

All initial topics of subjects \( i \) with \( RD_i = 1 \), which have no predecessors from the other subjects, have a unique starting event – the class starting event.

All terminal topics of the subject \( i \) with \( DD_i = TD \), which have no successors from the other subjects, have a unique completion event – the class completion event.

For each subject \( i \) with \( RD_i > 1 \) including its initial topic, a directed arc from the class starting event to the starting event of the topic is introduced. The value \( RD_i - 1 \) is assigned to this arc.

For each \( j \) with \( DD_i < TD \) in ordering 3 as terming 1c, a directed arc from the completion event of the topic to the class completion event is introduced. This arc gets value \( TD - DD_i \).

We explain the minimal duration \( DMIN_j \) (expressed in days) of a topic \( j \) in more detail.

For a s-topic \( j \), \( DMIN_j \) is equal to the total number of its lectures given by the corresponding sequence. Recall that there is at most one lecture per day per topic.

Let \( j \) be a d-topic, \( LMIN_j \) and \( LMAX_j \) the minimal and the maximal length of its daily lectures and \( TP_j \) the total number of periods required for its completion.

We assume that topic \( j \) can be completed within the given limits, i.e., there exists an integer \( m \) for which integers \( u_1, u_2, \ldots, u_m \) can be found such that

\[
LMIN_j \leq u_s \leq LMAX_j, \quad s = 1, 2, \ldots, m \quad \text{and} \quad \sum_{s=1}^{m} u_s = TP_j. \tag{2}
\]

Then \( DMIN_j \) is obviously equal to the smallest \( m \) for which (2) is satisfied. It can be shown that

\[
DMIN_j = \left\lfloor \frac{TP_j}{LMAX_j} \right\rfloor, \tag{3}
\]

where \( \lceil x \rceil \) is the smallest integer not smaller than \( x \) [2].

We mention that it may happen that for no \( m \) condition (2) is satisfied (for example, \( LMIN_j = 4 \), \( LMAX_j = 5 \) and \( TP_j = 11 \)). Therefore, the input data of the acyclic problem is defined such that all d-topics can be completed within the given limits. Necessary and sufficient conditions are presented in Section 6.1.2.
The associated network model for the example in Section 2 is presented in Fig. 1. The broken arcs represent dummy topics, while the dotted arcs indicate given release and due dates of subjects. The minimal duration of the d-topics is calculated from (3).

6. Heuristic method

Starting from the two levels of the lexicographic optimization problem and using the analogy with multi-project scheduling mentioned in Section 4, a heuristic method for its solution is developed. The method applies the so-called parallel

Fig. 1. Associated network model.
scheduling principle which has been shown to be one of the best ways for solving resource constrained project scheduling problems [1]. Namely, the method tries to find one feasible timetable for all classes simultaneously without rescheduling the already assigned lectures.

The timetable is constructed in an iterative way day by day, going successively through the sequence of all working days with scheduling procedure DAY (see below). The method stops when all topics are completely scheduled or all available working days are used. In the former case a feasible timetable is determined, while in the latter case an incomplete timetable remains, i.e., some topics are not finished.

A pseudo-code in Pascal of procedure DAY is given in Fig. 2. Its general features are the following.

The procedure INITCOM corresponds to the general level of the problem. Using the associated network model defined in Section 5, INITCOM determines in a heuristic way the initial combination for the current day. This is a set of topics (irrespective of the classes involved) with the lengths of their lectures, which can be realized at that day such that all constraints from the general level are satisfied and future violations of the due dates will be minimal. INITCOM is given in more detail in Section 6.1.

The initial combination is considered as the current combination for that day. In order to find a minimal free time timetable (see Section 4) for all lectures from the current combination, procedure MINFT [3] is applied. MINFT represents an exact algorithm for determining such a timetable. The algorithm translates the daily timetable problem to the corresponding interval colouring of the associated weighted graph. It solves the colouring by using an implicit enumeration approach. This step corresponds to the daily level of the lexicographic optimization problem.

When MINFT cannot find a feasible daily timetable—it does not exist—procedure REDUCE is applied. REDUCE reduces one or more topics of the current combination. This means that for such a topic either the length of its lecture in the combination is decreased or it is totally eliminated from the combination. These

```
procedure DAY;
begin
  dtf := false; (*dtf — daily timetable is found*)
  INITCOM; (*general level*)
  while dtf = false do
    begin
      MINFT; (*daily level*)
      if a feasible timetable doesn’t exist
        then REDUCE
        else dtf := true
    end
end;
```

Fig. 2. Pseudo-code of DAY.
topics are chosen in a heuristic way such that additional future violations of due
dates are minimal.

The reduced combination is considered as the current one and MINFT is applied
again. In this way, using MINFT and REDUCE a finite number of times, a daily
timetable is found. More details are explained in Section 6.2.

When the described method at each iteration takes care of objectives \( \phi_1 \) and \( \phi_2 \)
in lexicographic order, then it can generate a feasible timetable. This timetable, if
not optimal, is a near-optimal solution of the lexicographic optimization problem.
Namely, the method doesn't guarantee to find a solution of the acyclic problem (if
it exists), but it tries to determine a timetable in which, first, violations of the due
dates and, second, free time periods for the classes are minimized.

The method can be applied for one class after the following simplifications. In
INITCOM the teachers are ignored, see Section 6.1.4. Instead of using MINFT, a
minimal free time timetable of the initial combination is always found by assigning
its lectures to the given periods in any arbitrary order without idle periods. REDUCE
is not used.

6.1. Procedure INITCOM

INITCOM consists of several steps. First, the set of schedulable topics at the current
day is determined. Then, for each topic from this set, the set of feasible daily
lectures, the preferred daily lecture and the priority of its realization at that day are
specified. Finally, starting with these values and using a heuristic procedure, the
initial combination is found.

All these steps are described in more detail in the subsections.

6.1.1. Set of schedulable topics

The set of schedulable topics at the current day \( d \), \( S(d) \), is defined as a set of all
topics (irrespective of the classes involved) which can be scheduled at day \( d \) with
respect to the orderings and the release dates.

Obviously, \( S(1) \) contains all initial topics of subjects \( i \) with \( RD_i = 1 \). In the example
of Section 2, \( S(1) = \{ T_1, T_3, T_8, T_{10}, T_{14}, T_{16}, T_{18} \} \).

For \( d > 1 \), each topic in \( S(d) \) satisfies the following conditions. All its predecessors
(if existing) were completed at the previous \( d - 1 \) iterations, the release date of its
subject is not greater than \( d \) and the topic is not completely scheduled.

6.1.2. Set of feasible daily lectures

For each topic \( f \) from \( S(d) \), the set of feasible daily lectures of the topic at day \( d \),
\( F_f(d) \), is a set of all lengths of the lectures with which \( f \) can be realized at day \( d \)
with respect to its type.

If topic \( f \) is an s-topic, then \( F_f(d) \) contains only one number. This number is the
length (number of periods) of the first unscheduled lecture in the given sequence
of lectures for this topic. In our example, \( F_{10}(1) = \{ 3 \} \) and \( F_{16}(1) = \{ 2 \} \).
Let \( j \) be a d-topic and \( TP_j(d) \) be the total number remaining unscheduled periods of topic \( j \) after \( d-1 \) iterations. If \( j \) was not yet scheduled before day \( d \), \( TP_j(d) = TP_j \). Obviously, \( TP_j(d) \) can be completed within the given limits if condition (2) from Section 5 is fulfilled, where instead of \( TP_j \), \( TP_j(d) \) is used. Therefore, the minimal duration of \( TP_j(d) \), i.e., the smallest number of days for its completion, is equal to \( \lceil TP_j(d)/LMAX_j \rceil \), according to (3).

\( F_j(d) \) contains all integers \( u \) which satisfy the following conditions.

- \( LMIN_j \leq u \leq \min(LMAX_j, TP_j(d)) \);
- if topic \( j \) is scheduled at day \( d \) by a lecture with length \( u \), then \( TP_j(d+1) = TP_j(d) - u \) can be realized by lectures within the given limits.

For example, if \( TP_j(d) = 12 \), \( LMIN_j = 4 \) and \( LMAX_j = 6 \), then \( F_j(d) = \{4, 6\} \). Indeed, if \( j \) is realized by length 4 or 6, then the remaining 8 or 6 periods can be completed within the given limits. However, if the realized length is 5, then the remaining 7 periods cannot be completed.

\( F_j(d) \) is formally expressed as follows [2].

- For \( LMIN_j = 1 \),
  \[ F_j(d) = \{1, 2, ..., \min(LMAX_j, TP_j(d))\} \]  
  \( (4) \)

- for \( LMIN_j > 1 \),
  \[ F_j(d) = \left\{ u \mid LMIN_j \leq u \leq LMAX_j, TP_j(d) - u \in D \setminus \bigcup_{s=1}^{s*} U_s \right\}, \]  
  \( (5) \)

where \( D \) is the set of all nonnegative integers, \( U_s, s = 0, 1, ..., \) is defined as \( U_s = \{ u \mid s \cdot LMAX_j < u < (s+1)LMIN_j \} \) and \( s^* \) is the largest of all \( s \) for which \( U_s \neq \emptyset \). It can be proven that \( s^* = \lceil (LMIN_j - 1)/LMAX_j - LMIN_j \rceil - 1, U_s \neq \emptyset \) for \( s = 0, 1, ..., s^* \) and all \( U_s \) are mutually disjoint.

In our example, topic \( T_1: TP_1(1) = TP_1 = 5 \), \( LMIN_1 = 1 \), \( LMAX_1 = 3 \) and, according to (4), \( F_1(1) = \{1, 2, 3\} \), topic \( T_8: LMIN_8 = 2 \), \( LMAX_8 = 4 \) and, therefore, \( s^* = 0 \) and \( U_0 = \{1\} \). Because \( TP_8(1) = TP_8 = 12 \), then according to (5), \( F_8(1) = \{2, 3, 4\} \).

We mention that a d-topic \( j \) with \( LMIN_j > 1 \) can be completed within the given limits if and only if \( TP_j \notin U_s \), for \( s = 0, 1, ..., s^* \). Furthermore, if the lectures of topic \( j \), scheduled before day \( d \), have lengths which belong to the corresponding sets of feasible daily lectures, \( F_j(d) \) is always nonempty.

It should be noted that for \( LMIN_j = 1 \), topic \( j \) can be completed within the given limits and \( F_j(d) \neq \emptyset \) in any case.

6.1.3. Preferred daily lecture

Let us extend the notation \( TP_j(d) \) to the s-topics with the same meaning as for the d-topics. Then, the minimal duration of \( TP_j(d) \) is equal to the total number of unscheduled lectures for topic \( j \).

The preferred daily lecture of a topic \( j \) from \( S(d) \) at day \( d \), \( L^*_j(d) \), is a member of \( F_j(c) \) which satisfies the following conditions.

If \( j \) is scheduled at day \( d \) with a lecture of length \( u \), \( u \in F_j(d) \), then
- for \( u \geq L^*_j(d) \), the minimal duration of \( TP_j(d) - u \) is less than the minimal duration of \( TP_j(d) \);
- for \( u < L^*_j(d) \), the minimal duration of \( TP_j(d) - u \) is equal to the minimal duration of \( TP_j(d) \).

If \( j \) is an \( s \)-topic, then \( L^*_j(d) \) is the unique member of \( F_j(d) \). For a \( d \)-topic \( j \) it can be shown that \( L^*_j(d) \) always exists and

\[
L^*_j(d) = \begin{cases} 
L_{\text{MAX}}_j & \text{for } r_j(d) = 0, \\
\max(L_{\text{MIN}}_j, r_j(d)) & \text{for } r_j(d) > 0,
\end{cases}
\]

where \( r_j(d) \) is the remainder of \( TP_j(d)/L_{\text{MAX}}_j \) [2].

For topic \( T_i \) of the example, according to (6), \( L^*_i(1) = 2. \) Actually, the minimal duration of \( TP_i(1) \) is two. If \( u = 1 \), then the minimal duration of \( TP_i(1) - u = 4 \) is also two. However, if \( u = 2 \) or 3, then \( TP_i(1) - u \leq 3 \) and its minimal duration is one.

### 6.1.4. Priorities of schedulable topics

In most of the heuristics for resource constrained project scheduling problems, before scheduling activities to a period, priorities are determined according to a predefined priority rule. In practice one of the most used is the \textit{minimal latest start time} (LST) rule [1]. This rule is accommodated to the general level of the lexicographic optimization problem in the following way.

For a topic \( j \) from \( S(d) \), \( \text{LST of } TP_j(d) \) is the latest day at which the scheduling of \( TP_j(d) \) periods can start such that the due dates will not be violated, taking into account the minimal durations of the topic's successors and the corresponding ordering.

The LST rule is formulated as: for \( p, q \in S(d) \), topic \( p \) has higher priority at day \( d \) than topic \( q \) if \( \text{LST of } TP_p(d) \) is smaller than \( \text{LST of } TP_q(d) \).

Using the associated network model from Section 5, the LST rule can be reformulated [2]. It can be proved that \( \text{LST of } TP_j(d) \) is equal to \( \text{TD} - C_j(d) + 1 \). \( C_j(d) \) is the sum of the minimal duration of \( TP_j(d) \) and the length of the longest path from the completion event of topic \( j \) to the class completion event in the corresponding associated network.

Therefore, the LST rule is reduced to the \textit{maximal critical path} (CP) priority rule: For \( p, q \in S(d) \), topic \( p \) has higher priority at day \( d \) than topic \( q \) if \( C_p(d) > C_q(d) \). When \( C_p(d) = C_q(d) \), \( p \) has higher priority than \( q \) if the smallest member of \( F_p(d) \) is greater than the smallest member of \( F_q(d) \).

Using the network model in Fig. 1, for topics \( T_1, T_8 \) and \( T_{10} \), \( C_1(1) = 10, C_8(1) = C_{10}(1) = 9 \). Therefore, \( T_1 \) has higher priority than \( T_8 \) and \( T_{10} \). As the smallest members of \( F_8(1) \) and \( F_{10}(1) \) are respectively 2 and 3, \( T_{10} \) has higher priority than \( T_8 \).

### 6.1.5. Initial daily combination

Let for each topic \( j \in S(d) \), \( F_j(d) \), \( L^*_j(d) \) and the priority at day \( d \), according to the CP rule, be specified. All members of \( F_j(d) \) are ordered in an increasing se-
procedure FIRST;
begin
  eop := false; (*eop-end of the procedure*)
  IC(d) := Ø;
  j := the first topic from S(d);
  while eop = false do
    begin
      if j can be in IC(d) with \( L^*_{j}(d) \) then
        IC(d) := IC(d) U \{j\} and \( L_{j}(d) = L^*_{j}(d) \)
      else if \( L^*_{j}(d) > L_{MIN_{j}}(d) \) and j can be in IC(d) with \( L_{MIN_{j}}(d) \) then
        IC(d) := IC(d) U \{j\} and \( L_{j}(d) = L_{MIN_{j}}(d) \);
      if j = the last topic from S(d) then
        eop := true
      else
        j := the next topic from S(d)
    end
  end;
end;

Fig. 3. Pseudo-code of FIRST.

procedure SECOND;
begin
  eop := false;
  change := false; (*change-increasing of the lengths*)
  while eop = false do
    begin
      j := the first topic from IC(d);
      if \( L_{j}(d) \) can be increased then
        \( L_{j}(d) = L_{j}(d) + 1 \) and change := true;
      if j = the last topic from IC(d) then
        begin
          if change = false then eop := true
        end
      else
        j := the next topic from IC(d)
    end
  end;
end;

Fig. 4. Pseudo-code of SECOND.

sequence, while set \( S(d) \) is ordered according to the decreasing priorities of its topics.

Let \( IC(d) \) denote the set of topics from the initial combination for day \( d \), ordered according to decreasing priorities, and \( L_{j}(d) \), \( j \in IC(d) \), be the lengths of their lectures.

\( IC(d) \) and \( L_{j}(d) \) are determined by a heuristic procedure which consists of the following two phases.

The first phase FIRST, represented in Fig. 3, goes through all topics \( j \) in \( S(d) \), according to their ordering. It investigates whether \( j \) can be in \( IC(d) \) with length
An acyclic multi-period timetabling problem

$L^*_j(d)$, respecting the total number of available periods $H(d)$ for day $d$. If this is not possible, FIRST checks whether $j$ can be in $IC(d)$ with the smallest member $LMIN_j(d)$ of $F_j(d)$. Topic $j$ can be in $IC(d)$ with length $u$ if for all topics $p$ from $IC(d)$ with the same teacher (or class) as topic $j$, the sum of $L_p(d)$ over $p$ is not greater than $H(d) - u$.

The second phase SECOND, represented in Fig. 4, starts from $IC(d)$ and $L_j(d)$, $j \in IC(d)$, determined by FIRST. It increases lengths $L_j(d)$ whenever possible, in an equalized cyclic way.

In each cycle SECOND goes through all topics $j$ in $IC(d)$, according to their ordering, trying to increase $L_j(d)$ to the first next number in $F_j(d)$ (if exists). From (4) and (5), it can be easily seen that if $L^*_j(d)$ is not the first member after $L_j(d)$ in $F_j(d)$, then this member is equal to $L_j(d) + 1$.

$L_j(d)$ can be increased if the following conditions are satisfied.
- For all topics $p$ from $IC(d)$ with the same teacher (class) as topic $j$, the sum of $L_p(d)$ is smaller than $H(d)$;
- $L_j(d)$ is not the last member in $F_j(d)$;
- $L^*_j(d)$ is not the first member after $L_j(d)$ in $F_j(d)$.

The cycle is repeated until $H(d)$ or $F_j(d)$ are totally utilized and, therefore, further increases become impossible.

As mentioned in Section 6.1.2, $F_j(d) \neq \emptyset$ for $j \in S(d)$ and, consequently, the initial daily combination, determined by the described procedure, always exists.

In our example from Section 2, application of INITCOM to the first working day with $H(1) = 6$ is shown in Table 2: $IC(1) = S(1)$ and $L_1(1) = L_4(1) = L_5(1) = L_{11}(1) = 3$, $L_{14}(1) = L_{16}(1) = L_{18}(1) = 2$.

### 6.2. Finding a daily timetable

The cycle of finding a timetable for day $d$ starts with application of MINFT to the initial combination for this day: First, to $IC(d)$ and lengths $L_j(d)$ a weighted graph $G$ is associated such that the lecture of each topic $j$ from $IC(d)$ is represented

<table>
<thead>
<tr>
<th>$S(1)$</th>
<th>$C_j(1)$</th>
<th>Priority</th>
<th>$F_j(1)$</th>
<th>$L^*_j(1)$</th>
<th>Class</th>
<th>Teacher</th>
<th>FIRST</th>
<th>SECOND</th>
</tr>
</thead>
<tbody>
<tr>
<td>$T_1$</td>
<td>10</td>
<td>1</td>
<td>${1,2,3}$</td>
<td>2</td>
<td>I</td>
<td>$N_2$</td>
<td>2</td>
<td>+1</td>
</tr>
<tr>
<td>$T_{10}$</td>
<td>9</td>
<td>2</td>
<td>${1}$</td>
<td>3</td>
<td>II</td>
<td>$N_3$</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>$T_8$</td>
<td>9</td>
<td>3</td>
<td>${2,3,4}$</td>
<td>4</td>
<td>II</td>
<td>$N_4$</td>
<td>2</td>
<td>+1</td>
</tr>
<tr>
<td>$T_4$</td>
<td>9</td>
<td>4</td>
<td>${1,2,3}$</td>
<td>1</td>
<td>I</td>
<td>$N_1$</td>
<td>1</td>
<td>+1</td>
</tr>
<tr>
<td>$T_{14}$</td>
<td>9</td>
<td>5</td>
<td>${1,2}$</td>
<td>1</td>
<td>III</td>
<td>$N_2$</td>
<td>1</td>
<td>+1</td>
</tr>
<tr>
<td>$T_{18}$</td>
<td>8</td>
<td>6</td>
<td>${2,3,4}$</td>
<td>4</td>
<td>III</td>
<td>$N_3$</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>$T_{16}$</td>
<td>5</td>
<td>7</td>
<td>${2}$</td>
<td>2</td>
<td>III</td>
<td>$N_1$</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>
as a node with weight equal to $L_j(d)$. Two nodes are connected with an edge if the corresponding lectures have either a class or a teacher in common. Then, for each connected component $H$ of $G$, MINFT tries to find a minimal free time timetable for all lectures, translating the problem to an interval colouring of $H$ [3].

If for an $H$ such a timetable is found, it is considered as a final one for day $d$.

If there is at least one $H_i$ for which a feasible timetable does not exist, then REDUCE is applied as follows. For each such $H_i$, the set $IC_i(d)$ of all topics corresponding to its nodes is considered. A topic $j^*$ from $IC_i(d)$ with length $L_j^*(d)$ which was the last formed in INITCOM, is chosen. In particular, if lengths $L_j(d)$ of all $j \in IC_i(d)$ were formed in FIRST, then $j^*$ is a topic with the smallest priority in $IC_i(d)$. If the lengths of some topics from $IC_i(d)$ were increased in SECOND, then $j^*$ belongs to those which length was formed through the largest number of cycles and again it has the smallest priority. Topic $j^*$ is reduced in the following way. If $L_j^*(d) \geq L_{MIN_j^*}(d)$, then its length is decreased to the first member before $L_j^*(d)$ in $F_j^*(d)$. Otherwise, $j^*$ is eliminated from $IC_i(d)$.

To the reduced set of all unscheduled topics $\bigcup_i IC_i(d)$ and their lengths $L_j(d)$, MINFT and REDUCE (if necessary) are applied in the same way as for the initial combination.

The cycle is repeated until, after application of MINFT, the final timetables for all associated components $H$ are obtained.

In our example the connected weighted graph given in Fig. 5 is associated with the initial combination. The application of MINFT shows that there exists no feasible daily timetable. REDUCE considers the whole set $IC(1)$. As $T_4$ is the topic from $IC(1)$ whose length was formed last by two cycles of SECOND, see Table 2, $L_4(1)$ is decreased from 3 to 2. Further, MINFT is applied to $IC(1)$ with reduced $L_4(1)$ and a daily timetable, represented in the first column of Table 3, is determined.

Table 3 contains a complete feasible timetable, obtained by the described heuristic method, in which violations of the due dates and free periods for the classes don’t exist. Therefore, this timetable is a solution of the given acyclic problem.

![Fig. 5. Weighted graph associated to the initial combination.](image-url)
### 7. Conclusions

Modelling and solving the acyclic problem as proposed in this paper, represents one of the possible ways to deal with this difficult combinatorial problem, which is strongly dependent on the individual view of the decisionmakers from the educational system.

The heuristic method in Section 6 combines two different approaches and techniques: on the general level a heuristic approach developed like for a resource con-
strained project scheduling and on the daily level an implicit enumeration approach typical for small sized combinatorial problems.

As at each iteration the exact exponential time procedure MINFT is used a finite number of times, the method has exponential time complexity. In order to avoid time consuming situations, the method is implemented with some additional time limits as follows. MINFT generates a finite sequence of feasible timetables until a minimal free time timetable is reached. Each timetable from the sequence has less free time (given in Section 4) than the previous one [3]. Therefore the CPU time of MINFT is limited by, say, 30 seconds. If, during the application of MINFT to a connected component (Section 6.2) the given time limit is exceeded, then the last generated feasible timetable (if exists) represents the final timetable for the component. In such a case MINFT is used as a heuristic which can deal with daily subproblems of large dimensions.

In real-life applications, however, the dimensions of the daily subproblems are rather small and in most cases the timetables were found immediately without reductions. Therefore, the execution time of the method is quite reasonable.

For example, at the educational institute where the problem has arisen, the method was successfully applied to the timetable for school-year 1988/89. Dimensions of the problem were: about 220 working days with 6 or 7 periods per day, 8 classes, 34 subjects which should be realized within time intervals of lengths from 2 weeks to the whole year, 340 topics with about 60% d-topics and 40 teachers. The established timetable satisfied all requirements as mentioned in Section 2. At each application of MINFT the number of lectures were not larger than 40, their lengths were at most 4 and the given time limit was not exceeded. Furthermore, REDUCE was used in only 4% of the days, and just once a day. Total execution time was 5 CPU minutes on a VAX 11/780.

Based on this results one may expect that further applications of the method at real-life problems will be successful too.

In some instances of the acyclic problem the constraint that two topics with a precedence relation should be scheduled in different days is not required. The described method can be easily modified for solving such a case as follows. Let \( j \) be a topic from the set \( S(d) \), defined in Section 6.1.1, such that \( TP_j(d) \leq LMAX_j \). Then, all immediate successors of this topic belong to \( S(d) \). If in FIRST (Section 6.1.5) \( j \) cannot be in \( IC(d) \) with length \( L_j^*(d) \) (equal to \( TP_j(d) \)), then all its immediate successors are eliminated from \( S(d) \). When MINFT is applied to the component which contains topic \( j \) and its successors, then at all steps of the procedure an additional constraint is considered. This means that the vertices of the successors should be coloured with colours greater than the colours used for the vertex of \( j \). All other steps of the method remain the same.

The method is flexible enough to accommodate some additional requirements such as: some topics should be assigned to predefined days and predefined periods of these days, some teachers and classes are not always available etc.

If at each iteration the daily level is ignored, the method can be easily modified
An acyclic multi-period timetabling problem

to solve preemptive resource constrained multi-project scheduling problems of similar nature.

Acknowledgement

The authors wish to thank their colleagues at the University of Twente for their support, especially Wim Nawijn for his useful remarks.

References