European Journal of Operational Research 256 (2017) 685–695

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Discrete Optimization

A column generation approach to high school timetabling modeled as a multicommodity flow problem

Árton P. Dorneles[a,*], Olinto C. B. de Araújo[b], Luciana S. Buriol[a]

[a] Instituto de Informática, Universidade Federal do Rio Grande do Sul, 91501-970 Porto Alegre, RS, Brazil
[b] Colégio Técnico Industrial de Santa Maria, Universidade Federal de Santa Maria, 97105-900 Santa Maria, RS, Brazil

ARTICLE INFO

Article history:
Received 19 August 2015
Accepted 2 July 2016
Available online 7 July 2016

Keywords:
Timetabling
MILP
Column generation
Multicommodity flow

ABSTRACT

School timetabling is a classic optimization problem that has been extensively studied due to its practical and theoretical importance. It consists in scheduling a set of class-teacher meetings in a predetermined period of time, satisfying requirements of different types. Given the combinatorial nature of this problem, solving medium and large instances of timetabling to optimality is a challenging task. When resources are tight, often it is difficult to find even a feasible solution. Several techniques have been developed in the literature to tackle the high school timetabling problem. Since the use of exact methods, as mathematical programming techniques, are considered impracticable to solve large real world instances, metaheuristics and hybrid metaheuristics are the most used solution approaches. In this paper we propose a multicommodity flow model for the high school timetabling problem. In addition, we apply Dantzig–Wolfe decomposition to the proposed model, propose a column generation algorithm, and present experimental results on well known instances of the problem. The results show that the lower bounds obtained through our approach are tight and can be generated faster than previous approaches reported in the literature.

1. Introduction

A common task to all educational institutions is to provide an assignment of classes that combines teachers, students, rooms and periods (or timeslots) to achieve a feasible timetable, satisfying requirements of different nature.

Usually, requirements are separated into *hard* and *soft* ones. By hard requirements we mean those that must be satisfied, while soft requirements may be violated, but should be satisfied whenever possible. Soft requirements can have different levels of importance and are often conflicting with each other such that it may be impossible to satisfy all of them at the same time. Typically, the quality of a solution is associated directly to the soft requirements satisfaction. The more soft requirements are satisfied, which can be accounted differently according to the level each one belongs to, the better a solution is considered.

Quality is a critical solution attribute because, once the timetable is established, it will determine the use of physical resources and the daily routine of several people, possibly thousands, for a long period that usually is about one year. Due to repetition, even minor issues can turn into major problems in the course of time, affecting directly the quality of teachers' work, the learning of students, and the expenses of the institution.

Educational timetabling problems have many variants proposed in the literature, and the set of objectives and requirements depend mostly on the context of the application, the institution and the place where it is located (Drexl & Salewski, 1997; Post et al., 2014). Although there are several variants, educational timetabling problems are commonly comprised in three classes: school timetabling, course timetabling, and examination timetabling (Schaerf, 1999). In both school and course timetabling the aim is to build a weekly schedule. However, in school timetabling a set of classes must be assigned to timeslots, whereas in course timetabling a set of university courses must be scheduled avoiding overlaps of courses with common students. Finally, in the examination problem a set of exams must be spread in a time horizon avoiding overlaps for the students.

In our present study we focus on the school timetabling problem. This problem first appeared in the scientific literature in the 60s (Gotlieb, 1963) and since then it has gained increasing attention. The most basic variant of the problem is to schedule a set of class-teacher events (or meetings) in such a way that no teacher (nor class) is required in more than one lesson at a timepoint. This

* Corresponding author.
E-mail addresses: arton.dorneles@inf.ufrgs.br, artondorneles@gmail.com (Á.P. Dorneles), olinto@ctism.ufsm.br (O.C.B. de Araújo), buriol@inf.ufrgs.br (L.S. Buriol).

http://dx.doi.org/10.1016/j.ejor.2016.07.002
0377-2217/© 2016 Elsevier B.V. All rights reserved.

basic problem can be solved in polynomial time by a min-cost network flow algorithm (de Werra, 1971). However, in real-world applications, teachers can be unavailable in some periods. If this constraint is taken into account, the resulting timetabling problem is NP-complete (Even, Itai, & Shamir, 1975).

In fact, the most real-world timetabling problems come to light as combinatorial optimization problems that fall in the NP-hard class. For this reason, researchers around the world have investigated these problems and several different techniques have been developed. Since the use of exact methods, as mathematical programming techniques, are considered impracticable for solving medium and large real-world instances, metaheuristics and hybrid metaheuristics are the most used solution approaches. For an updated overview about these approaches, we refer the reader to the survey by Pillay (2014).

Although research groups share similar interests, before 2011, the majority of publications in the literature reported results focused on solving specific problem variants from their country with application-dependent (often unavailable) test cases. This methodology, historically made it difficult to compare results among different solution approaches (Schaerf & Gaspero, 2001). As an attempt to overcome these issues, along the latest editions of International Conference on the Practice and Theory of Automated Timetabling (PATAT), a group of high school timetabling researchers has developed an XML based format, called XHSTT (Post et al., 2012; 2014), to express problems from different countries in an unified way and it has been widespread accepted by the research community. Recently, its use was promoted in the Third International Timetabling Competition (ITC-2011) that run different methods over instances originated from several countries (Post, Gaspero, Kingston, McCollum, & Schaerf, 2016). Problems that can be represented in the XHSTT format are normally refered as Generalized High School Timetabling Problem, hereafter denoted as GHSTP.

Since the ITC-2011, several heuristic approaches have been developed to solve the GHSTP. However, no known method is able to find exact solutions for non trivial instances of the problem in a reasonable amount time. In Kristiansen, Sørensen, and Stidsen (2014) an integer programming formulation was proposed for the GHSTP, which provides the best exact approach for the problem to the best of our knowledge. From the set of 38 instances, for 10 of them a solution with cost zero is already known, which defines an optimal solution by the definition of XHSTT. Gurobi was used to run the other 28 instances, and only four small instances were solved within the time limit of 86,400 seconds (1 day) per run. No exact method based in lower bounds was proposed up to now specifically tailored to solve the GHSTP. Besides estimating the quality of heuristic solutions, a good lower bound is a basic requirement for implementing efficient branch-and-bound based procedures.

In this paper, instead of tackling the whole GHSTP, we focus in the CTTPCR (Class-Teacher Timetabling Problem with Compactness Requirements), which is a subproblem of GHSTP that models a typical Brazilian school, and comprises all Brazilian instances considered in the ITC-2011. In contrast with problems originated from the majority of other countries, the CTTPCR has a more constrained search-space since all teachers are pre-assigned to each event, and room assignment is not required.

The main contribution of our work is two-fold. Differently from traditional high school timetabling models, we propose a multicommodity flow model for the CTTPCR. In addition, we also propose a column generation algorithm for solving the linear relaxation of the Dantzig–Wolfe decomposition applied in the multicommodity flow model for providing strong lower bounds for the problem quickly. These new lower bounds are important to evaluate the quality of solutions obtained with heuristic approaches to the CTTPCR, as well as for methods designed to the GHSTP. Besides being faster and simpler than previous approaches published in the literature, our column generation scales better for large instances. Moreover, new better bounds were provided for all five CTTPCR instances from the first round of the ITC-2011. By using them, we were able to prove solution optimality of two open instances of this competition.

The remainder of this paper is organized as follows. Section 2 presents some related works in the CTTPCR. Section 3 formally presents the problem tackled in this study, the notation used to represent it, and a mixed integer programming formulation. Section 4 presents a Dantzig–Wolf decomposition for the problem, a column generation algorithm for solving it, as well as two speedup strategies. Section 5 presents experimental results considering 12 real-world instances in comparison with previous approaches. Finally, Section 6 presents our major conclusions and an outline of future work.

2. Related works

The CTTPCR was first defined by Souza and Maculan (2000) that proposed the first MIP formulation for it, as well as an instance set that became a basic testbed used until nowadays. They have shown that solving the testbed instances with a general purpose MIP solver was impracticable at that time.

From the heuristic side, Souza, Ochi, and Maculan (2003) proposed a hybrid metaheuristic, called GTS-II, for solving instances of the CTTPCR. The GTS-II uses a greedy randomized constructive heuristic to build an initial solution that later is refined by a tabu search. Since their tabu search also includes infeasible solutions in the search space, it was equipped with a procedure that is invoked eventually in an attempt to restore the feasibility of the current solution. In the work of Santos, Ochi, and Souza (2005) a tabu search with diversification strategies was proposed to solve the CTTPCR. Their experiments showed that the proposed tabu search significatively outperformed GTS-II. In addition, the authors showed empirically that the proposed diversification strategy can improve the robustness of the tabu search.

Santos, Uchoa, Ochi, and Maculan (2012) proposed new MIP models, as well as a cut and column generation (CCG) algorithm by using Fenchel cuts (Boyd, 1994), providing, for the first time, strong lower bounds for CTTPCR instances. That work is considered a landmark because it established a reliable base to evaluate the quality of heuristic solutions.

Recently, we proposed a new MIP model for the CTTPCR (Dorneles, Araújo, & Buriol, 2012) and developed a hybrid approach that combines a fix-and-optimize heuristic with a variable neighborhood descent procedure (Dorneles, Araújo, & Buriol, 2014). We were able to prove solution optimality for the majority of the instances tested and to the best of our knowledge, the methodology presented provides the state-of-the-art results for the problem.

3. Problem definition and modeling

In this section we introduce a new MIP compact formulation for the Class-Teacher Timetabling Problem with Compactness Requirements (CTTPCR). The problem considers a set of classes C and a set of teachers T. A class $c \in C$ is a group of students that follow the same course and have full availability. The goal of the problem is to build a timetable for a week that is usually organized as a set of days D, and each day is split into a set of periods P. We call as *timeslot* a pair composed of a day and class period, (d, p), with $d \in D$ and $p \in P$, wherein all periods have the same duration. Teachers $t \in T$ may be unavailable in some timeslots.

The main input for the problem is a set of events that should be scheduled. Typically, an event is a meeting between a teacher t

and a class c to address a particular subject in a given room. In this paper, we denote an event by a pair (t, c). The parameter H_{tc} determines the *workload* of an event (t, c), i.e, the number of lessons that must be taught by the teacher t for the class c. In addition, each event defines how lessons are distributed in a week by requesting an amount of double lessons, restricting the number of lessons per day, and defining whether lessons taught in a same day must be consecutive.

A *feasible* timetable has a timeslot assigned to each lesson of events satisfying the hard requirements H1–H6 below:

H1 The workload of each event must be satisfied.
H2 A teacher cannot be scheduled to more than one lesson in a given period.
H3 Lessons cannot be taught to the same class in the same period.
H4 A teacher cannot be scheduled to a period in which she/he is unavailable.
H5 The maximum number of daily lessons of each event must be met.
H6 Two lessons from the same event must be consecutive when scheduled for the same day, in case it is required by the event.

Besides feasibility regarding hard constraints, as many as possible of the soft requirements S1–S3 stated below should be satisfied:

S1 Avoid teachers' *idle periods*. A period of a teacher is considered idle if she/he has lessons assigned before and after that period on the same day.
S2 Minimize the number of *working days* for teachers. In this context, working day means a day that the teacher has at least one lesson assigned to him/her.
S3 Provide the number of double lessons requested by each event.

We propose a multicommodity flow model for the CTTPCR in which arcs represent transitions of time periods in a particular network graph. This approach was inspired by the work of Steinzen, Gintner, Suhl, and Kliewer (2010) where the authors also use arcs to represent transitions of time for the integrated vehicle- and crew-scheduling problem with multiple depots. In the model, each teacher is represented by a commodity. It means that determining a teachers' schedule is the same as finding a path in an appropriate network graph. Formally, we represent this network as a directed acyclic graph $G = (V, A)$, where V is a set of nodes and A is a set of arcs. Although all commodities share the same set of nodes, including the same *source* and *sink* nodes, each commodity considers only a given subset of arcs $A_t \subseteq A$. Fig. 1 presents an illustration of the graph G where all types of arcs are shown for a given commodity t. The figure is composed of days (vertical rounded rectangles) and periods of a day (horizontal straps). Each day block has vertical shaded rectangles related to the activities of each class (two classes are considered in the example). Next we describe the types of arcs of G:

- *Lesson arcs* are used to indicate which timeslots are assigned to a given teacher and class. Lessons arcs are usually shared between commodities and have an unitary capacity associated to ensure they are used only by a single commodity (teacher) at a time. In addition, for each lesson arc $a \in A_t$ is associated a label S_{ta} that represents the duration (in periods) of the lesson represented by the arc. Lesson arcs are referred to as *single lesson arcs* when $S_{ta} = 1$, and as *double lesson arcs* when $S_{ta} = 2$. In the figure, lesson arcs are all curve arcs within a day block and within the shaded rectangle related to some class.
- W_t is the set of *idle period arcs*. These arcs are used to identify the idle periods for each teacher t. A cost ω is associated to each idle period arc. In the figure, idle period arcs are all straight arcs within a day block, outside the shaded rectangles related to classes.
- Sets Q_t^- and Q_t^+ are sets of auxiliary arcs called, respectively, as *pull-in* and *pull-out* arcs. While pull-in arcs are all arcs incoming a day block, pull-out arcs are the ones outgoing a day block.
- Y_t is the set of *working day arcs*. These arcs are used to compute the number of working days for a teacher t. A cost γ is associated to each working day arc. In the figure, for each day, the head node of the working day arc corresponds to the tail node of each pull-in arc to that day.
- B_t is the set of *day-off arcs*. These arcs are used when a teacher t does not teach any lesson in a given day. These are the arcs located in the lower base of the figure. Their tail nodes are the same as for working day arcs.

Each path in the network is composed by a binary flow denoted by the variable x_{ta}, where $t \in T$ and $a \in A_t$. Each path starts from the source node, alternates through different types of arcs, ending at the sink node, as shown in Fig. 2.

Next, we present a mixed integer linear programming formulation for the CTTPCR hereafter denoted as $\mathcal{F}_1$. For convenience, the complete notation used in the formulation is presented in Table 1.

$$\textbf{Minimize} \sum_{t\in T}\left(\sum_{c\in C}\delta g_{tc} + \sum_{a\in W_t}\omega x_{ta} + \sum_{a\in Y_t}\gamma x_{ta}\right) \tag{1}$$

Subject to

$$\sum_{a\in A_{tv}^{+}} x_{ta} - \sum_{a\in A_{tv}^{-}} x_{ta} = b_v \quad \forall t\in T, v\in V \tag{2}$$

$$\sum_{t\in T}\sum_{a\in A_{tcdp}} x_{ta} \le 1 \quad \forall c\in C, d\in D, p\in P \tag{3}$$

$$\sum_{a\in\bigcup_{d\in D,p\in P}A_{tcdp}} S_{ta}x_{ta} = H_{tc} \quad \forall t\in T, c\in C \tag{4}$$

$$\sum_{a\in\bigcup_{p\in P}A_{tcdp}} S_{ta}x_{ta} \le L_{tc} \quad \forall t\in T, c\in C, d\in D \tag{5}$$

$$\sum_{a\in\bigcup_{p\in P}A_{tcdp}} x_{ta} \le 1 \quad \forall t\in T, c\in C, d\in D, h=1 \tag{6}$$

$$g_{tc} \ge M_{tc} - \sum_{a\in G_{tc}} x_{ta} \quad \forall t\in T, c\in C \tag{7}$$

$$\sum_{a\in Y_t} x_{ta} \ge Y_t' \quad \forall t\in T \tag{8}$$

$$x_{ta}\in\{0,1\} \quad \forall t\in T, a\in A_t \tag{9}$$

$$g_{tc} \ge 0 \quad \forall t\in T, c\in C \tag{10}$$

The objective function minimizes the violation of soft constraints. The flow conservation constraint set (2) ensures the total inflow equals the total outflow of each node (except source and sink), considering a given commodity t. Constraint set (3) ensures that the unitary capacity of the lesson arcs be respected. One can note that a structure of a multicommodity flow problem is represented by the constraint sets (2) and (3) and by the first two parts of the objective function (1). This structure is only able to address the requirements H2, H3, H4, S1, and S2. In order to model the remaining requirements, we included additional constraints and the last component of the objective function. Constraint set (4) ensures that the workload of each event is attended. Constraint set (5) ensures that the maximum number of daily lessons for each event

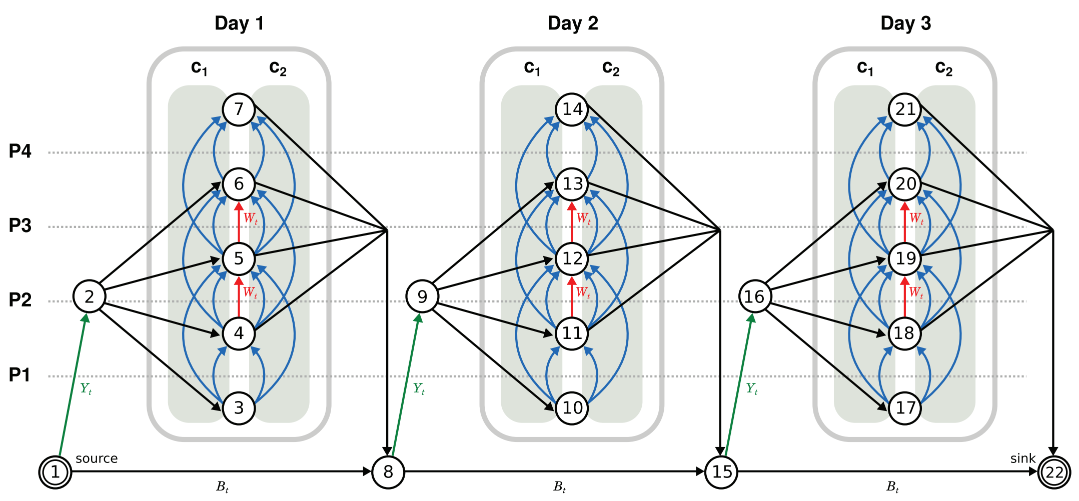

Fig. 1. Example of a network graph in a toy instance composed by three days, four periods by day (P1, P2, P3, P4), and two classes (c$_1$, c$_2$). Each day of the week is represented by a rounded rectangle where lesson arcs and idle period arcs are located. Inside each, lesson arcs appear in two groups represented by a shaded rectangle, where each group represents the lesson arcs for classes c_1 and c_2.

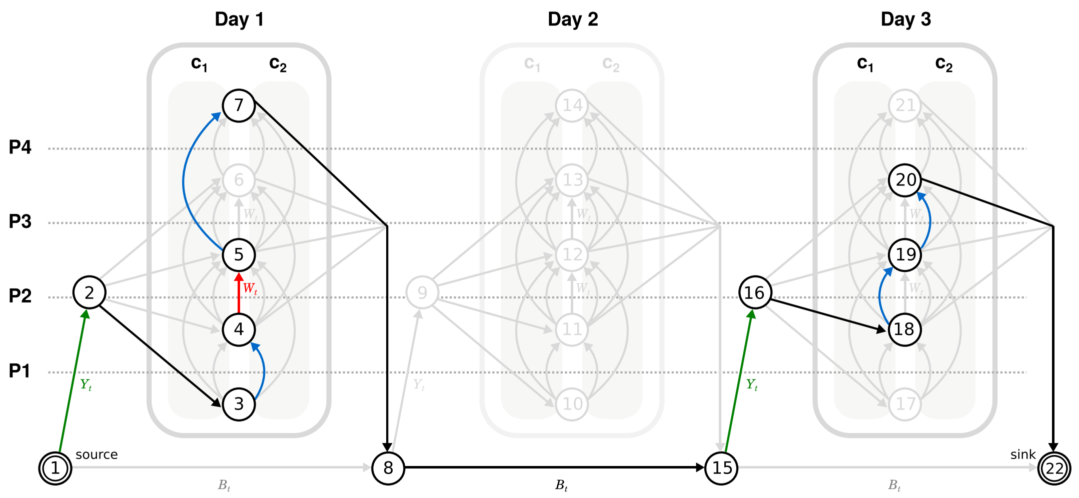

Fig. 2. Example of a feasible schedule for a teacher t represented by a path in the network. In this example, a teacher works only on days 1 and 3. On day 1, she/he teaches a single lesson for the class c_2 in the period P1, becomes idle in the period P2, and then gives a double lesson starting in the period P3 for the class c_1. On day 3, she/he teaches a single lesson for class c_1 in the period P2 and another one for class c_2 in the period P3.

is satisfied. Constraint set (6) ensures that lessons from the same event are scheduled in sequence by allowing the use of only one arc per day. This constraint is only activated when $h = 1$. Constraint set (7) computes the number of double lessons occurring in the solution. Constraint set (8) establishes a lower bound for the minimum number of working days for each teacher. This lower bound was proposed by Souza (2000) and it is computed according to Eq. (11).

$$Y'_t = \max\left\{\left\lceil\frac{\sum_{t\in T}\sum_{c\in C} H_{tc}}{|P|}\right\rceil, \max_{t\in T, c\in C}\left\{\left\lceil\frac{H_{tc}}{L_{tc}}\right\rceil\right\}\right\} \qquad \forall t \in T \tag{11}$$

By using Y'_t one can get a simple and computationally cheap lower bound (LB$_\gamma$) for the CTTPCR as defined in Eq. (12).

$$\text{LB}_\gamma = \sum_{t\in T} Y'_t \gamma \tag{12}$$

3.1. Additional cuts

Although the formulation $\mathcal{F}_1$ is suitable for representing the CTTPCR, due to the network structure, it eventually allows the construction of *unmeaningful paths* that overestimate the cost of suboptimal solutions. Fig. 3 illustrates three cases in which unmeaningful paths could occur. In Case 1, the flow path crosses through the day component by using a single node. Since the path does not contain any lesson arc, the working day arc is used unnecessarily for accessing the day component. In Case 2, the solution cost is overestimated because an idle period arc is misused. Ideally, an idle period arc should not appear in a path when succeeding a pull-in arc or preceding a pull-out arc. In Case 3, two single lessons are taught in sequence for the same class, while using a double lesson arc would be more appropriate. This situation can only occur when the requirement H6 is not being considered, i.e., $h = 0$. Otherwise, it is already avoided by the constraint set (6). In order to avoid these cases we can strengthen our formulation with the addition of some valid cut constraint sets (13)–(16). Constraint set (13) forbids Case 1, constraint sets (14) and (15) forbid Case 2, and constraint set (16) forbids Case 3.

$$x_{ti} + x_{tj} \le 1 \quad \forall t \in T, v \in V, i \in Q_t^- \cap A_{tv}^-, j \in Q_t^+ \cap A_{tv}^+ \tag{13}$$

$$x_{ti} + x_{tj} \le 1 \quad \forall t \in T, v \in V, i \in Q_t^- \cap A_{tv}^-, j \in W_t \cap A_{tv}^+ \tag{14}$$

$$x_{ti} + x_{tj} \le 1 \quad \forall t \in T, v \in V, i \in Q_t^+ \cap A_{tv}^+, j \in W_t \cap A_{tv}^- \tag{15}$$

$$x_{ti} + x_{tj} \le 1 \quad \forall t \in T, c \in C, d \in D, p \in P, i \in A_{tcdp}, j \in A_{tcdp+1}, \\ M_{tc} > 0, p < |P|, S_{ti} = S_{tj} = 1, h = 0 \tag{16}$$

Table 1
Notation used in the compact formulation $\mathcal{F}_1$.

Symbol	Definition
Sets	
$d \in D$	Days of a week.
$p \in P$	Periods of a day.
$t \in T$	Set of teachers (commodities).
$c \in C$	Set of classes.
$v \in V$	Set of all nodes.
$a \in A_t$	Set of all arcs of the commodity t ($A_t \subseteq A$).
$a \in A_{tcdp}$	Set of lesson arcs of the commodity t on class c, day d, and period p.
$a \in A^-_{tv}$	Set of all arcs incoming node v for commodity t.
$a \in A^+_{tv}$	Set of all arcs outgoing node v for commodity t.
$a \in Q^-_t$	Set of all pull-in arcs for commodity t.
$a \in Q^+_t$	Set of all pull-out arcs for commodity t.
$a \in Y_t$	Set of all working day arcs of teacher t.
$a \in W_t$	Set of all idle periods arcs of teacher t.
$a \in G_{tc}$	Set of all double lesson arcs of teacher t and class c.
Parameters	
b_v	Has value 1 when v is the source, -1 when v is the sink, otherwise 0.
$H_{tc} \in \mathbb{N}$	Number of lessons that teacher t must teach to class c.
$L_{tc} \in \{1, 2\}$	Maximum daily number of lessons that teacher t can teach to class c.
$S_{ta} \in \{1, 2\}$	Duration of arc a for the commodity t.
$M_{tc} \in \mathbb{N}$	Minimum amount of double lessons required by teacher t on class c.
$Y'_t \in \mathbb{N}$	Minimum amount of working days for teacher t.
$h \in \{0, 1\}$	Indicates whether requirement H6 is take into account.
$\delta = 1$	Cost of each required double lesson not satisfied.
$\omega = 3$	Cost for each idle period.
$\gamma = 9$	Cost for each working day.
Variables	
$x_{ta} \in \{0, 1\}$	Indicates whether commodity t uses arc a.
$g_{tc} \geq 0$	Number of unsatisfied double lessons of class c taught by teacher t.

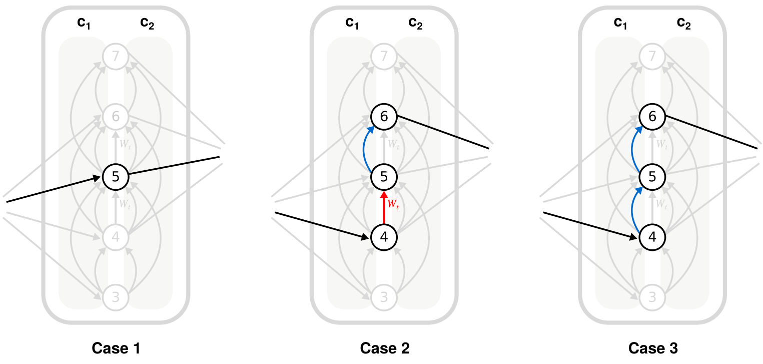

Fig. 3. Example of three different cases in which unmeaningful paths could be formed.

4. Column generation applied to the problem

By applying the Dantzig–Wolfe decomposition principles (Dantzig & Wolfe, 1960) on the compact formulation $\mathcal{F}_1$, we can obtain an alternative formulation for the CTTPCR, denoted as *master problem* (MP). In this formulation, stated by (17)–(20), let J_t be the set of all possible paths for a teacher t that satisfy all the hard requirements except H3. For each path $j \in J_t$ is associated a non-negative cost K_{tj} regarding the satisfaction of soft requirements. In addition, we define a binary variable λ_{tj} that indicates whether the path j is selected by teacher t.

$$\textbf{Minimize} \sum_{t\in T}\sum_{j\in J_t} K_{tj}\lambda_{tj} \tag{17}$$

subject to

$$\sum_{j\in J_t} \lambda_{tj} = 1 \quad \forall t \in T \tag{18}$$

$$\sum_{t\in T}\sum_{j\in J_t}\sum_{a\in A_{tcdp}} \bar{x}_{taj}\lambda_{tj} \leq 1 \quad \forall c \in C, d \in D, p \in P \tag{19}$$

$$\lambda_{tj} \in \{0, 1\} \quad \forall t \in T, j \in J_t \tag{20}$$

The objective of the MP, represented by Eq. (17), is to minimize the cost of selected paths. Constraint set (18) ensures that exactly one path is selected for each teacher. Constraint set (19) ensures that the unitary capacity of the lesson arcs is respected, where $\bar{x}_{taj}$ indicates whether the arc a is used in path j of teacher t. By solving the MP, one can obtain an integer optimal solution to the CTTPCR. However, this may be impracticable given the huge cardinality of J_t in problems faced in real applications. Instead, we propose to solve a linear relaxation of the MP through a column generation approach, with the purpose to achieve tight lower bounds for the problem.

The tightness of the bound depends mostly in how requirements are distributed between the master and pricing problems. However, as a general rule, the more requirements are handled in the pricing problem, the tighter is the bound produced by the column generation. With this rule in mind, we observed that in the extended model proposed by Santos et al. (2012), requirements H1, H3 and S3 are handled in the master problem. Hence, in this research, we propose a model in which all the requirements, except H3, are handled in the pricing problem. We were able to accomplish that by using a multicommodity representation.

In a straightforward implementation, a column generation procedure starts with a master problem fulfilled with a restricted set of columns, hereafter called *restricted master problem* (RMP). At each iteration, the RMP is solved and its dual variables are used to price out new columns by solving subproblems. During the resolution of each subproblem (pricing problem), columns with a negative reduced cost are added to the RMP. This procedure is repeated until no column with negative reduced cost is found. In our case, we consider the RMP stated by (21)–(25).

$$\textbf{Minimize} \sum_{t\in T}\left(\sum_{j\in J_t} K_{tj}\lambda_{tj} + \varepsilon_t z_t\right) \tag{21}$$

subject to

$$\sum_{j\in J_t}\lambda_{tj} + z_t = 1 \quad \forall t \in T \tag{22}$$

$$\sum_{t\in T}\sum_{j\in J_t}\sum_{a\in A_{tcdp}} \bar{x}_{taj}\lambda_{tj} \le 1 \quad \forall c\in C, d\in D, p\in P \tag{23}$$

$$\lambda_{tj} \ge 0 \quad \forall t\in T, j\in J_t \tag{24}$$

$$z_t \ge 0 \quad \forall t \in T \tag{25}$$

Note that variables are continuous and their upper bounds are implied by the constraint set (22). Besides, we chose to start the initial set of columns by introducing an artificial variable z_t for each teacher, that is penalized with a high cost in the objective function. As pointed out by Lübbecke and Desrosiers (2005), assigning arbitrarily a too high cost to artificial variables may slowdown the convergence of the column generation. Thus, in order to keep the penalization as low as possible, we defined the cost of ε_t according to Eq. (26)

$$\varepsilon_t = \delta\sum_{c\in C} M_{tc} + \omega\max(0, |P|-2)|D| + \gamma|D|. \tag{26}$$

The cost ε_t is equal to the sum of three parts that correspond, respectively, to the upper bounds of the costs of the soft constraints S1, S2 and S3. In other words, ε_t is a trivial upper bound for the cost of a teacher path (K_{tj}).

After solving the RMP, the next step consists in a multiple pricing scheme, where the subproblem $\mathcal{P}_t$ is solved for each $t\in T$. Eqs. (27)–(35) present the formulation of $\mathcal{P}_t$:

$$\mathcal{P}_t = \begin{cases} \textbf{Minimize}\;\; R_t = \sum_{c\in C}\delta g_{tc} + \sum_{a\in Y_t}\gamma x_{ta} + \sum_{a\in W_t}\omega x_{ta} \\ \qquad -\sum_{c\in C}\sum_{d\in D}\sum_{p\in P}\sum_{a\in A_{tcdp}}\sigma_{cdp}x_{ta} - \pi_t & (27)\\ \textbf{Subject to} \\ \sum_{a\in A^+_{tv}} x_{ta} - \sum_{a\in A^-_{tv}} x_{ta} = b_v \quad \forall v\in V & (28)\\ \sum_{a\in\bigcup_{d\in D,p\in P}A_{tcdp}} S_{ta}x_{ta} = H_{tc} \quad \forall c\in C & (29)\\ \sum_{a\in\bigcup_{p\in P}A_{tcdp}} S_{ta}x_{ta} \le L_{tc} \quad \forall c\in C, d\in D & (30)\\ \sum_{a\in\bigcup_{p\in P}A_{tcdp}} x_{ta} \le 1 \quad \forall c\in C, d\in D, h=1 & (31)\\ g_{tc} \ge M_{tc} - \sum_{a\in G_{tc}} x_{ta} \quad \forall c\in C & (32)\\ \sum_{a\in Y_t} x_{ta} \ge Y'_t & (33)\\ x_{ta}\in\{0,1\} \quad \forall a\in A_t & (34)\\ g_{tc}\ge 0 \quad \forall c\in C & (35)\end{cases}$$

Assuming π_t and σ_{cdp} as dual variables associated, respectively, to the constraint sets (22) and (23), the reduced cost R_t is defined by Eq. (27). Finally, observe that the remaining constraint sets, namely (28)-(35) are analogous to the ones presented in $\mathcal{F}_1$.

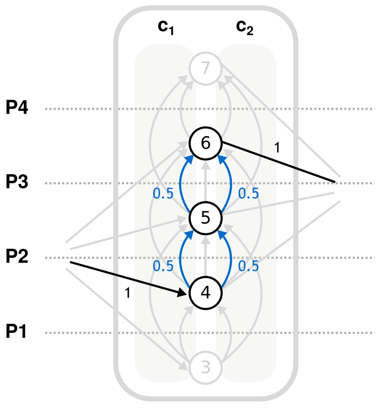

Fig. 4. Example of a relaxed subproblem solution in a working day. The number next to arcs represents the respective value flow. We can observe that the integral flows passing through the pull-in and pull-out arcs determine whether the respective teacher will teach from period P2 to P3. However, the flow is split into lesson arcs and we cannot determine for each period whether a lesson should be given either to class c_1 or c_2.

4.1. Speedup strategies

When comparing the expected computational effort required to solve the master and pricing problems, it is easy to predict that the bottleneck of the whole column generation process lies on the resolution of the pricing problem $\mathcal{P}_t$. Apart from being an integer problem, $\mathcal{P}_t$ also has to address the majority of the CTTPCR requirements. We resort to a MIP solver for solving it, however, one may note that even by using a state-of-the-art MIP solver, solving $\mathcal{P}_t$ to optimality might still be time consuming. This is particularly noteworthy with regard to the resolution of medium and large instances of the problem. Thus, in order to speedup the resolution of $\mathcal{P}_t$ with a MIP solver, we propose the two strategies described next.

The first strategy is grounded in the principle that any column with negative reduced cost contributes to improve the objective value of the restricted master problem. Hence, $\mathcal{P}_t$ does not need to be solved exactly in every iteration since this is mandatory only in the last one. With this in mind, we design our column generation algorithm to operate in two sequential phases (I and II). In phase I, instead of solving a subproblem up to optimality, we stop the solver as soon as it finds a feasible solution proved to be within a given percentage α far from optimal. When no more solutions with $R_t < 0$ can be generated using the current value of α, the algorithm switches to phase II where α is set to zero and, consequently, the subproblems are solved to optimality.

The second speedup strategy consists in solving a relaxed version of $\mathcal{P}_t$, hereafter denoted by $\mathcal{P}'_t$, where the integrality constraints are enforced only for the variables associated to pull-in and pull-out arcs. In other words, it means that $\mathcal{P}'_t$ is able to precisely determine the first and last lesson periods for each working day of a teacher. However, as a consequence of the relaxation, it may be not possible to identify exactly in which class a lesson should be given, as illustrated in Fig. 4.

In spite of losing some information, the relaxation decreases considerably the number of integer variables of the subproblem. While $\mathcal{P}_t$ has a large number of integer variables that depends on the number of events, $\mathcal{P}'_t$ uses only $|Q^-_t \cup Q^+_t| \times |D| = 2|P| \times |D|$ binary variables. In practical instances, $|D|$ is typically limited by 6 (Monday to Saturday) and $|P|$ hardly exceeds 20, even in schools holding full-day programs. In that sense, it is safe to claim that for

Table 2
Main characteristics of the tested instances.

Id	Name	$\|D\|$	$\|P\|$	$\|T\|$	$\|C\|$	$\|E\|$	$\Sigma_{(t,c)\in E}M_{tc}$	$\Sigma_{(t,c)\in E}H_{tc}$	LB_γ	BKV	BKV_{itc}
1	Inst1	5	5	8	3	21	21	75	189	202	–
2	Inst2	5	5	14	6	63	29	150	333	333	–
3	Inst3	5	5	16	8	69	4	200	414	423	–
4	Inst4	5	5	23	12	127	66	300	639	652	–
5	Inst5	5	5	31	13	119	71	325	756	762	–
6	Inst6	5	5	30	14	140	63	350	738	756	–
7	Inst7	5	5	33	20	205	84	500	999	1017	–
A	BrazilInstance1	5	5	8	3	21	21	75	189	200	11
D	BrazilInstance4	5	5	23	12	127	66	300	621	648	27
E	BrazilInstance5	5	5	31	13	119	71	325	756	775	19
F	BrazilInstance6	5	5	30	14	140	63	350	738	779	41
G	BrazilInstance7	5	5	33	20	205	84	500	999	1052	53

practical purposes, the number of integer variables of $\mathcal{P}'_t$ is constrained between $12|P|$ and 240 for any real-world instance.

Finally, it is important to point out that even with $\mathcal{P}'_t$ being less restrictive than $\mathcal{P}_t$, as we show further in the computational results, both problems often find the same objective value, i.e., $\mathcal{R}'_t \lesssim \mathcal{R}_t$. As a result, the lower bound obtained by the column generation using $\mathcal{P}'_t$ is strongly close to the one obtained by using $\mathcal{P}_t$. We refer to the column generation method that uses $\mathcal{P}_t$ in the pricing step as Integer Pricing Column Generation (IPCG), while the version that uses $\mathcal{P}'_t$ is identified as Relaxed Pricing Column Generation (RPCG).

5. Computational experiments

In this section we present an experimental evaluation for the proposed models and methods. The instances are solved by CPLEX 12.6.0 (IBM, 2013) with default settings but using a single core. The algorithms were implemented in C++ using the compiler g++ 4.6.1. The experimental results were computed in a Desktop-PC equipped with an Intel Core i5-2300 processor clocked at 2.8 gigahertz, 4 gigabytes of RAM, over a 64 bits Linux operating system. Along this section, we report results of one run for each tested algorithm, since they are deterministic. The mathematical model parameters γ, ω, and δ were set to 9, 3, and 1, respectively.

5.1. Dataset

To evaluate the algorithm we used the instances presented in Table 2. In this table, the first two columns present the instance identifier and the name. Since their names are long, we use the identifiers for shortening reference along the text. The instances are split into two sets. Instances 1–7 comprise set-1 and are available from the repository (LABIC, 2008). Requirement (H6) is not considered in this group of instances. Instances A, D, E, F, G, from set-2, are different versions of instances 1, 4, 5, 6, 7, respectively. They differ mainly in two aspects: in set-2, teachers are available in all periods, and requirement (H6) is considered. These modifications made the instances more challenging. Set-2 comprises all instances used in the first round of the Third International Timetabling Competition 2011 (ITC, 2011) that our model can handle. They are part of the XHSTT-2012 archive that is available at https://www.utwente.nl/ctit/hstt/archives/XHSTT-2012/XHSTT-2012-separate_instances.zip. Other instances available in this archive take into account requirements that CTTPCR does not handle such as resource assignment, linked events, and events with multiple teachers. We used the original versions of the instances (first round of ITC2011), since some of them were modified after the competition. Columns $|D|$ and $|P|$ show the number of days and periods, respectively, while columns $|T|$, $|C|$ and $|E|$ present the number of teachers, classes and events, respectively, where $E = \{(t, c) : t \in T, c \in C, H_{tc} > 0\}$. Columns $\Sigma_{(t,c)\in E}M_{tc}$ and $\Sigma_{(t,c)\in E}H_{tc}$ present the total double lessons required and the total workload, respectively. Column LB_γ presents a simple lower bound computed according to Eq. (12). Column *BKV* presents the best known values for these instances, all of them obtained by Dorneles et al. (2014) except instances E and G. The best known values for instances E and G were obtained in this work using the same fix-and-optimize approach proposed in Dorneles et al. (2014) albeit using a time limit of 10 hours. The solution values reported in column *BKV* were computed in the same fashion as all previous works in the CTTPCR, i.e., by using the objective function defined in formulation $\mathcal{F}_1$. However, in the ITC-2011, the solution values were reported with a different objective function by using the HSEVal validator.[1] Column BKV_{itc} presents the values provided by HSEval for the best known solutions of set-2. Cells of set-1 are filled with "-" since instances 1–7 were not tested in ITC-2011. The only difference between the solution values computed by $\mathcal{F}_1$ and HSEval is that the latter does not include the value of LB_γ as part of the solution cost, i.e., $BKV_{itc} = BKV - LB_\gamma$.

The next sections have the aim of presenting results of the algorithms and models presented in this work, as well as comparing them with the previous state-of-the-art results for the problem.

5.2. Integer solutions obtained by MIP models

The first experiment aims at comparing results from a traditional MIP model (Dorneles et al., 2014) and the flow model $\mathcal{F}_1$ proposed in this work. Each instance was run for at most 7200 seconds (2 hours) for each model. Table 3 presents for each instance and MIP model, the objective value found (*Obj*), the running times (*Time*), the number of columns (*#col*) and rows (*#row*) generated by the respective model, the gap provided by CPLEX (Gap_C) and the Gap (Gap_B) from the best known value calculated as $100*(Obj-BKV)/min(BKV,Obj)$. The lowest values for each column are shown in boldface.

From the 12 instances, the flow model was able to find better results in six instances, and similar results in two instances. Only in four instances the previous proposed model found better results than $\mathcal{F}_1$. However, on average, the traditional model found better gap results. This is due to the results for instance 7 which $\mathcal{F}_1$ found a final solution with an objective cost considerable higher than the objective value found by the traditional model. The number of columns from $\mathcal{F}_1$ was higher in only three instances, while the number of rows of the traditional model was higher in all instances.

[1] HSEVal checks the solution feasibility and also computes the solution value. It is available in http://www.it.usyd.edu.au/~jeff/cgi-bin/hseval.cgi.

Table 3
Comparison results between MIP models using a time limit of 2 hours.

Id	Dorneles et al. (2014)						$\mathcal{F}_1$					
	Obj	*Time* (hours)	#col	#row	Gap$_C$	Gap$_B$	Obj	*Time*	#col	#row	Gap$_C$	Gap$_B$
1	**202**	**2**	1306	2914	6.44	**0.00**	**202**	**2 hours**	**1163**	**1187**	**5.82**	**0.00**
2	340	2	3100	6855	2.06	2.10	**333**	**196 seconds**	**3021**	**2480**	**0.00**	**0.00**
3	**426**	**2**	2898	6532	**2.82**	**0.71**	429	**2 hours**	**2424**	**2305**	3.50	1.42
4	653	2	5221	11,642	2.14	0.15	**652**	**988 seconds**	**4174**	**4081**	**0.00**	**0.00**
5	782	**2**	**6349**	14,057	3.32	2.62	**777**	**2 hours**	6602	**6613**	**2.70**	**1.97**
6	**780**	**2**	**6850**	15,153	**5.38**	**3.17**	804	**2 hours**	7000	**6653**	8.02	6.35
7	**1043**	**2**	**9155**	20,242	**4.22**	**2.56**	1645	**2 hours**	9364	**8805**	38.81	61.75
A	**200**	**2**	2011	3910	5.50	**0.00**	**200**	**2 hours**	**1566**	**1513**	**4.00**	**0.00**
D	735	**2**	10,132	19,008	15.51	13.43	**726**	**2 hours**	**7168**	**7571**	**12.12**	**12.04**
E	868	**2**	10,244	19,513	12.90	12.00	**812**	**2 hours**	**7568**	**9594**	**5.36**	**4.77**
F	1174	**2**	11,530	21,772	37.14	50.71	**952**	**2 hours**	**8312**	**10,012**	**20.69**	**22.21**
G	**1248**	**2**	16,200	30,328	19.95	**18.63**	1285	**2 hours**	**11,380**	**14,121**	**19.88**	22.15
Avg.	**704**	2	7083	14,327	**9.78**	**8.84**	735	**6099 seconds**	**5812**	**6245**	10.08	11.05

Table 4
Comparison results between linear relaxations.

Id	$\mathcal{F}'_1$			Dorneles et al. (2014)			IPCG ($\alpha = 0$ percent)		
	LB	Time (seconds)	Gap$_L$ (percent)	LB	Time (seconds)	Gap$_L$ (percent)	LB	Time (seconds)	Gap$_L$ (percent)
1	189	**0.1**	6.88	189	**0.1**	6.88	**202**	11.3	**0.00**
2	**333**	**0.4**	**0.00**	**333**	1.1	**0.00**	**333**	8.7	**0.00**
3	414	**0.2**	2.17	414	0.9	2.17	**423**	12.0	**0.00**
4	643	**0.8**	1.40	639	2.2	2.03	**652**	10.0	**0.00**
5	756	**2.3**	0.79	756	6.8	0.79	**762**	13.5	**0.00**
6	738	**2.8**	2.44	738	9.0	2.44	**756**	40.5	**0.00**
7	999	**7.8**	1.80	999	24.1	1.80	**1017**	25.5	**0.00**
A	190	**0.1**	5.26	189	0.2	5.82	**200**	6.7	**0.00**
D	635.5	**3.8**	1.97	621	14.1	4.35	**646**	26.4	**0.31**
E	767.5	**3.9**	0.98	756	13.8	2.51	**775**	20.2	**0.00**
F	754	**5.5**	3.32	738	23.0	5.56	**773**	45.2	**0.78**
G	1023	**12.0**	2.83	999	77.2	5.31	**1039**	85.1	**1.25**
Avg.	620	**3.3**	2.49	614	14.5	3.31	**632**	25.4	**0.20**

5.3. Lower bounds for the problem

This section aims at presenting and comparing lower bound results provided by the linear relaxation of $\mathcal{F}_1$ (denoted as $\mathcal{F}'_1$), the lower bound found by the linear relaxation of the traditional model of Dorneles et al. (2014), and the method IPCG proposed in this work with $\alpha = 0$ percent. Table 4 presents for each instance and method, the lower bound found (LB), the running times (Time) in seconds, and the percentage deviation from the best known value to the lower bound Gap_L, which is computed as $100*(BKV-LB)/LB$. Best results for each column are shown in boldface.

In summary, it can be observed from the results that $\mathcal{F}'_1$ is the fastest method whereas IPCG provides the best lower bounds considering all instances tested. Both approaches, $\mathcal{F}'_1$ and IPCG, present significant improvements in comparison with Dorneles et al. (2014). $\mathcal{F}'_1$ provides better or equal lower bounds than Dorneles et al. (2014), using about four times less time, on average. Although IPCG spent about twice the time of Dorneles et al. (2014), the Gap_L achieved by the former is approximately sixteen times smaller. When comparing IPCG and $\mathcal{F}'_1$ the Gap_L found by IPCG is considerably better, but it takes longer to run.

5.4. Parameter testing for the proposed column generation algorithm

In this section we evaluate the performance of different settings and speedup strategies for the proposed column generation. Table 5 presents average results regarding to IPCG and RPCG with several values for α (presented in Section 4.1). For both methods are reported the total running times (*Time*) in seconds, the total number of columns generated (*#col*), the total number of iterations (*#iter*), and the percentage of the total time spent in the pricing step (*pr*). The shortest running time is shown in boldface. Values shown in parentheses next to the columns *#col* and *#iter* present the number of columns/iterations that were inserted/performed in the phase II of the algorithm. We recall that when α is set to zero, the phase II is not required to be performed.

We would like to highlight that regardless the use of a relaxed pricing, all lower bounds generated by RPCG matched exactly the ones obtained by IPCG in Table 4 (we further discuss this topic in the next subsection). Thus, results shown in Table 5 are focused in presenting their differences in terms of running times and number of iterations.

From the table, it is noteworthy that RPCG is faster than IPCG when the same value of α is considered. Both algorithms are affected similarly according to changes in the parameter α, taking longer when $\alpha = 0$ percent and $\alpha > 10$ percent. In these cases, the slowdown is caused due to the quality of columns generated in the pricing step. In one hand, when $\alpha = 0$ percent, extra computational time is spent to generate high quality columns by ensuring optimality for each pricing problem. In the other hand, when $\alpha > 10$ percent, although the pricing step runs faster, it adds a higher number of low quality columns into the master, what increases the number of iterations required to reach optimality, especially in phase II.

A suitable setting for α, which provides a good trade-off between quality and computational effort for generating a column, is comprised with 1 percent $\leq \alpha \leq$ 10 percent. In this range, we found the best overall results achieved by RPCG with $\alpha = 4$ percent, that combines the two acceleration strategies proposed in

Table 5
Average results for all instances comparing different settings for the proposed column generation.

	IPCG				RPCG			
α (percent)	Time (seconds)	#col	#iter	pr (percent)	Time (seconds)	#col	#iter	pr (percent)
0	25.4	33	784	94.7	15.7	33	782	91.7
1	24.8	34 (1)	778	95.1	15.0	34 (1)	781	91.3
2	21.4	35 (1)	780	94.2	13.9	35 (1)	805	90.3
3	19.7	35 (1)	781	93.8	14.0	37 (1)	830	89.6
4	19.4	36 (1)	789	93.7	**13.8**	38 (1)	847	89.2
5	18.9	36 (1)	795	93.4	14.2	40 (1)	891	88.7
6	19.0	37 (1)	799	93.3	14.6	42 (1)	927	88.2
7	18.5	37 (1)	806	92.9	15.1	45 (1)	942	88.1
8	18.2	38 (1)	798	92.8	15.9	47 (1)	987 (3)	87.3
9	18.0	39 (1)	803	92.8	15.5	47 (1)	996	86.8
10	18.6	40 (1)	822 (1)	92.8	16.9	51 (1)	1039 (1)	86.3
20	24.2	53 (3)	893 (16)	92.6	17.0	58 (1)	1074 (6)	85.8
30	41.5	101 (5)	1027 (69)	93.2	32.7	105 (2)	1441 (27)	85.7
40	44.0	112 (8)	990 (157)	94.2	39.8	129 (5)	1333 (76)	87.4
50	35.0	79 (13)	873 (266)	94.5	31.6	124 (9)	1104 (166)	89.8
60	32.1	65 (17)	830 (350)	94.8	31.4	128 (11)	1082 (225)	90.2
70	35.3	77 (18)	823 (371)	95.2	34.1	144 (11)	1078 (217)	90.6
80	35.9	79 (20)	809 (413)	95.8	32.7	134 (12)	1061 (236)	91.2
90	35.9	77 (23)	827 (464)	95.6	29.6	124 (13)	1043 (259)	91.0

Table 6
Results presenting the difference (Δ) between the reduced costs provided by $\mathcal{P}_t$ and $\mathcal{P}'_t$.

Id	#pricing	$\Delta > 0$ (percent)		$\Delta \leq 1$ (percent)		$\Delta > 1$ (percent)		Max(Δ)	Avg(Δ)
1	232	51	(21.98)	231	(99.57)	1	(0.43)	1.08	0.06
2	420	37	(8.81)	420	(100.00)	0	(0.00)	0.89	0.02
3	416	5	(1.20)	416	(100.00)	0	(0.00)	0.56	0.00
4	782	66	(8.44)	782	(100.00)	0	(0.00)	1.00	0.02
5	806	74	(9.18)	806	(100.00)	0	(0.00)	0.50	0.01
6	1020	101	(9.90)	1020	(100.00)	0	(0.00)	0.38	0.01
7	1023	114	(11.14)	1023	(100.00)	0	(0.00)	0.61	0.02
A	248	5	(2.02)	248	(100.00)	0	(0.00)	0.83	0.00
D	759	161	(21.21)	758	(99.87)	1	(0.13)	1.18	0.04
E	1023	146	(14.27)	1023	(100.00)	0	(0.00)	0.85	0.03
F	1320	270	(20.45)	1320	(100.00)	0	(0.00)	1.00	0.05
G	1650	619	(37.52)	1649	(99.94)	1	(0.06)	1.12	0.07
Avg.	808	137	(13.84)	808	(99.95)	0	(0.05)	0.83	0.03

Section 4.1. However, considering that the speedups for IPCG ($\alpha =$ 9 percent), RPCG ($\alpha = 0$ percent) and RPCG ($\alpha =$ 4percent) calculated over IPCG ($\alpha = 0$ percent) are, respectively, 1.41, 1.61 and 1.84, it can be observed that the proposed acceleration strategies are able to improve significatively the convergence of the column generation even if used exclusively. In fact, the relaxed pricing strategy can provide a higher speedup than introducing an $\alpha > 0$ percent. When both strategies are used together, the results are slightly better.

5.5. Objective values provided by $\mathcal{P}_t$ and $\mathcal{P}'_t$

In this section, we evaluate empirically the level of approximation provided by $\mathcal{P}'_t$ in comparison with $\mathcal{P}_t$. Since $\mathcal{P}'_t$ is a relaxation of $\mathcal{P}_t$, we can denote the difference between the optimal reduced cost of these problems by $\Delta = R_t - R'_t$. In order to measure the magnitude of this difference, we ran IPCG using $\alpha = 0$ percent. During the pricing step, besides solving $\mathcal{P}_t$ we also solved $\mathcal{P}'_t$ for the sake of computing the value of Δ. We report in Table 6, for each instance, the total number of pricing problems solved (*#pricing*). Moreover, for the cases of $\Delta > 0$, $\Delta \leq 1$, and $\Delta > 1$, we report the number of occurrences of each of these cases, and the percentage of these occurrences considering the total number of pricings solved. Finally, the maximum and the average values of Δ are given in the last two columns.

Analyzing the table one may note that, on average, only 13.84 percent of the pricing problems solved revealed a difference between $\mathcal{P}_t$ and $\mathcal{P}'_t$. However, almost 100 percent of the differences are, on average, less or equal to one unit cost (see column $\Delta \leq 1$). In addition, among all runnings, only three pricing problems resulted in a difference higher than one unit cost (see column $\Delta >$ 1). In these rare cases, the value of Δ still barely surpassed one unit cost.

As shown in the last column of the table, the average Δ is only 0.03, thus meaning that the difference between values computed by $\mathcal{P}'_t$ and $\mathcal{P}_t$ is tiny. In fact, 0.03 corresponds to a value which is about 33 times smaller than the least cost penalty associated to an unsatisfied double lesson ($\delta = 1$). We attribute to this tiny difference the fact that lower bounds obtained by IPCG and RPCG are the same, as mentioned in the previous section.

5.6. Comparing the proposed method with a Cut and Column Generation

In this section we compare our column generation method with the approach proposed by Santos et al. (2012), hereafter referred to as Cut and Column Generation (CCG). We used their original CCG implementation, which was kindly provided by the authors. In order to compare results, we included the requirement H6 in their implementation. Table 7 presents results for each instance comparing the performance of CGG and RPCG using $\alpha = 4$ percent. For both methods are reported the total running times (*Time*) in seconds, the total number of columns generated (*#col*), and the total number of iterations (*#iter*) performed. Both methods, CCG and

Table 7
Comparison results between the lower bounds provided by the Cut and Column Generation (CCG) proposed by Santos et al. (2012) and the proposed Relaxed Pricing Column Generation (RPCG).

				CCG			RPCG			
Id	LB	LB_{itc}	Gap (percent)	Time (seconds)	#col	#iter	Time (seconds)	#col	#iter	Speedup
1	202	–	0.00	**0.17**	351	15	3.26	248	32	0.05
2	333	–	0.00	**5.12**	960	30	6.34	476	35	0.81
3	423	–	0.00	**2.22**	916	28	4.24	414	27	0.52
4	652	–	0.00	40.25	1474	44	**6.64**	735	37	6.06
5	762	–	0.00	34.43	1888	35	**11.29**	813	28	3.05
6	756	–	0.00	72.25	2102	56	**13.37**	835	29	5.41
7	1017	–	0.00	395.63	2284	67	**16.30**	1020	32	24.27
A	200*	11*	0.00	**0.33**	443	19	3.25	232	30	0.10
D	646*	25*	0.31	50.74	1524	44	**14.35**	847	40	3.54
E	775*	19*	0.00	97.30	1918	73	**20.36**	1184	53	4.78
F	773*	35*	0.78	34.49	1865	37	**23.53**	1338	49	1.47
G	1039*	40*	1.25	451.38	2740	78	**42.22**	2026	63	10.69
Avg.	631	–	0.20	98.69	1539	44	**13.76**	847	38	5.06

RPCG, obtained exactly the same lower bound values, which are reported in column *LB*. Column LB_{itc} presents the lower bounds for instances of set-2 according to the objective function of HSEval ($LB_{itc} = LB - LB_{\gamma}$), i.e, LB_{itc} is a lower bound for BKV_{itc}. Column *Gap* reports the optimality gap for each instance. It is computed by $100*(BKV - LB)/LB$. Finally, column *Speedup* presents the speedup of RPCG over CCG. Column *Speedup* is computed by CCG/RPCG. Results with shortest running time are shown in boldface. Values marked with ($*$) are new best lower bounds.

According to the results, besides CCG and RPCG are able to provide the same lower bounds that are tighter than the ones provided by the relaxation of model $\mathcal{F}_1$, they present a distinct performance according to the instance size. While CCG achieves short running times on four small instances (1,2,3 and A), RPCG scales better, performing faster on the remaining eight medium and large instances. In addition, our method is approximately 5 times faster than CCG, on average, and particularly in the largest instances, the performance improvement becomes more prominent, being about 24 times faster on instance 7, and about 10 times faster on instance G.

Besides finding the optimal bounds for all instances of set-1, we were able to find new tighter lower bounds for all instances of set-2. These new results allow to reduce the average optimality gap for these instances from 2.09 percent, as presented in Dorneles et al. (2014), to 0.20 percent. Moreover, optimality was proved for instances A and E since the lower bounds found matched with the best known values.

6. Conclusions

In this paper we tackle the Class-Teacher Timetabling Problem with Compactness Requirements (CTTPCR), which is a well-known variant of the High School Timetabling Problem originated from Brazilian schools. Instances of the CTTPCR comprise a subset of the instances considered in the Third International Timetabling Competition held in 2011, which also evaluated instances from several other countries. In addition to a novel mathematical programming formulation based on a multicommodity flow network for the CTTPCR, we proposed a column generation approach, using two speedup strategies, for proving strong lower bounds for this problem.

In comparison with the state-of-the-art column generation for CTTPCR, the experimental results show that our approach is able to produce the same lower bounds, albeit with two significant advantages: i) the method is simpler; ii) and it is five times faster on average. Moreover, we improved best known lower bounds of 5 out of 12 instances from the literature. Among these new results, two are proved to be optimal (namely instance A and E). These results show that the proposed technique is efficient for producing lower bounds for the CTTPCR, motivating its use to variants of this problem.

As future work, there are several directions in which this research can be extended. Firstly, one may propose a tailored method for solving $\mathcal{P}_t$ or $\mathcal{P}'_t$ in a more efficient way than using a generic MIP solver. Secondly, since about 90 percent of the computational time is spent in the pricing step, the gains with parallelization might be promising. In fact, given that one pricing is solved for each teacher, they could be trivially solved in parallel. Finally, in order to harness the full potential of the column generation, one may propose branching strategies inside a branch-and-price framework for providing, ultimately, optimal integer solutions for the problem.

Acknowledgments

This work is supported by CAPES and Petrobras, Brazil. We would like to thanks Professor Haroldo Gambini Santos for sharing his column generation codes, what made possible to us compare results in the same machine.

References

Boyd, E. A. (1994). Fenchel cutting planes for integer programs. *Operations Research, 42*(1), 53–64.
Dantzig, G. B., & Wolfe, P. (1960). Decomposition principle for linear programs. *Operations research, 8*(1), 101–111.
Dorneles, Á. P., Araújo, O. C. B., & Buriol, L. S. (2012). The impact of compactness requirements on the resolution of high school timetabling problem. In *Proceedings of XLIV simposio Brasileiro de pesquisa operacional (SBPO 2012)* (pp. 3336–3347).
Dorneles, Á. P., Araújo, O. C. B., & Buriol, L. S. (2014). A fix-and-optimize heuristic for the high school timetabling problem. *Computers & Operations Research, 52*, 29–38.
Drexl, A., & Salewski, F. (1997). Distribution requirements and compactness constraints in school timetabling. *European Journal of Operational Research, 102*(1), 193–214.
Even, S., Itai, A., & Shamir, A. (1975). On the complexity of time table and multi-commodity flow problems. In *Proceedings of the 16th annual symposium on foundations of computer science, SFCS '75* (pp. 184–193). Washington, DC, USA: IEEE Computer Society.
Gotlieb, C. (1963). The construction of class-teacher timetables. In *Proceedings of IFIP, Amsterdam* (pp. 73–77).
IBM (2013). *ILOG CPLEX 12.6 user's manual. Mountain View, CA.*
ITC (2011). Third international timetabling competition. http://www.utwente.nl/ctit/hstt/itc2011/. (accessed 23.07.15).
Kristiansen, S., Sørensen, M., & Stidsen, T. R. (2014). Integer programming for the generalized high school timetabling problem. *Journal of Scheduling, 18*(4), 377–392.
LABIC (2008). Instances of school timetabling. http://labic.ic.uff.br/Instance/index.php?dir=SchoolTimetabling/. (accessed 23.07.15).
Lübbecke, M. E., & Desrosiers, J. (2005). Selected topics in column generation. *Operations Research, 53*(6), 1007–1023.
Pillay, N. (2014). A survey of school timetabling research. *Annals of Operations Research, 218*(1), 261–293.

Post, G., Ahmadi, S., Daskalaki, S., Kingston, J. H., Kyngas, J., Nurmi, C., & Ranson, D. (2012). An XML format for benchmarks in high school timetabling. *Annals of Operations Research, 194*, 385–397.

Post, G., Gaspero, L., Kingston, J. H., McCollum, B., & Schaerf, A. (2016). The third international timetabling competition. *Annals of Operations Research, 239*, 69–75.

Post, G., Kingston, J., Ahmadi, S., Daskalaki, S., Gogos, C., Kyngas, J., et al. (2014). XHSTT: an XML archive for high school timetabling problems in different countries. *Annals of Operations Research, 218*, 295–301.

Santos, H. G., Ochi, L. S., & Souza, M. J. (2005). A tabu search heuristic with efficient diversification strategies for the class/teacher timetabling problem. *Journal of Experimental Algorithmics, 10*, 1–16.

Santos, H. G., Uchoa, E., Ochi, L. S., & Maculan, N. (2012). Strong bounds with cut and column generation for class-teacher timetabling. *Annals of Operations Research, 194*, 399–412.

Schaerf, A. (1999). A survey of automated timetabling. *Artificial Intelligence Review, 13*(2), 87–127.

Schaerf, A., & Gaspero, L. D. (2001). Local search techniques for educational timetabling problems. In *Proceedings of the 6th international symposium on operations research in slovenia (sor-01)* (pp. 13–23).

Souza, M. (2000). *Programação de horários em escolas: UMA aproximação por metaheuristicas.* (Ph.D. thesis). Universidade Federal do Rio de Janeiro.

Souza, M., & Maculan, N. (2000). Melhorando quadros de horário de escolas através de caminhos mínimos. *Tendências em Matemática Aplicada e Computacional, 1*(2), 515–524.

Souza, M., Ochi, L., & Maculan, N. (2003). Metaheuristics: Computer decision–making. *A GRASP-tabu search algorithm for solving school timetabling problems* (pp. 659–672). Kluwer Academic Publishers.

Steinzen, I., Gintner, V., Suhl, L., & Kliewer, N. (2010). A time-space network approach for the integrated vehicle- and crew-scheduling problem with multiple depots. *Transportation Science, 44*(3), 367–382.

de Werra, D. (1971). *Construction of school timetables by flow methods.* (pp. 12–22). INFOR. (9)