Constraint programming based column generation for employee timetabling

Sophie Demassey, Gilles Pesant, Louis-Martin Rousseau

Constraint Programming based Column Generation for Employee Timetabling

Sophie Demassey, Gilles Pesant and Louis-Martin Rousseau

Centre for Research on Transportation (CRT), Université de Montréal,
C.P. 6128, succ. Centre-ville, Montreal, H3C 3J7, Canada

Abstract. The Employee Timetabling Problem (ETP) is a general class of problems widely encountered in service organizations (such as call centers for instance). Given a set of activities, a set of demand curves (specifying the demand in terms of employees for each activity for each time period) the problem consists of constructing a set of work shifts such that each activity is at all time covered by a sufficient number of employees. Work shifts can cover many activities and must meet work regulations such as breaks, meals and maximum working time constraints. Furthermore, it is often desired to optimize a global objective function such as minimizing labor costs or maximizing a quality of service measure. This paper presents variants of this problem which are modeled with the Dantzig formulation. This approach consists of first generating all feasible work shifts and then selecting the optimal set. We propose to address the shift generation problem with constraint satisfaction techniques based on expressive and efficient global constraints such as `gcc` and `regular`. The selection problem, which is modeled with an integer linear program, is solved by a standard MIP solver for smaller instances and addressed by column generation for larger ones. Since a column generation procedure needs to generate only shifts of negative reduced cost, the optimization constraint `cost-regular` is introduced and described. Preliminary experimental results are given on a typical ETP.

1 Introduction

Employee Timetabling Problems (ETP) form a wide class of optimization problems encountered in several industries and organizations. Generally, an ETP is the problem of designing valid employee schedules over a given time horizon that cover given workforce requirements. The timetabling attempts to optimize performance criteria such as the overall cost or the quality of service.

In a context where there are numerous possible activities, such as in call centers, a schedule refers to a sequence of activities performed during fixed time periods satisfying a given set of rules and regulations (e.g. a break of 15 minutes is necessary between two different work activities). Various constraints of that kind arise in real world ETP and their number and complexity quickly make these problems NP-hard. Furthermore, it is sometimes necessary to take into account the individual preferences and skills of each employee, which significantly increases the complexity of the problem.

Because it is a broad modeling methodology, constraint programming (CP) is well suited to generate the valid individual schedules. Most of the constraints in this problem are defined in terms of allowed patterns of activities and limits on the amount of work that is scheduled. These constraints can be efficiently tackled in a constraint satisfaction model using global constraints such as `gcc` [8] and `regular` [7], together able to restrict the number of occurrences and to enforce patterns of values in a sequence of variables.

This paper describes a general algorithm based on such a CP framework for solving several variants of ETP. The basic ETP can be represented as a set covering problem, where each column is a valid schedule. Our solution method is based on an integer linear formulation of the whole optimization problem, where variables represent the different permitted schedules (which are pre-computed in CP). When the number of valid schedules becomes too large to be generated and stored, the linear relaxation of the integer program is solved by column generation. In this case, only a subset of the schedules (with negative reduced costs) are generated by CP and added iteratively to the master linear program. In order to generate only negative reduced cost schedules, the `regular` constraint is replaced by its extension `cost-regular` within the CP model.

The interest of this approach is its ability to handle variations of the problem without major modifications to the algorithm itself. Indeed, comparing with pure linear programming approaches that are generally developed for ETP, CP offers a more straightforward way to model complex constraints. Furthermore, the decomposition of the problem makes the processing of these hard constraints independent from the global optimization process. This hybrid constraint-linear programming approach also differs from pure constraint programming approaches by taking more efficiently into consideration the optimization criterion.

CP-based column generation approaches have been proposed for several problems more or less related to ETP. The general framework was first introduced in [5]. It has since been applied to airline crew scheduling [3, 12], vehicle routing [10], and cutting-stock problems [4]. The subproblems solved by CP have taken the form of constrained shortest path problems and constrained knapsack problems. In our case we use `cost-regular`, which bears some similarity to a constrained shortest path, as we shall see in Section 4. Recently other hybrid CP-LP algorithms have been advantageously applied to solve ETP. In particular, Benoist et al. [1] presented a CP-based Benders decomposition for solving the timetabling problem encountered in a large call center. They use CP including the `flow` global constraint to handle the underlying flow structure of the problem.

The paper is organized as follows. The next section gives definitions and notation for employee timetabling problems. A typical set of regulation constraints and the associated CP model is presented in Section 3. Section 4 presents the optimization constraint `cost-regular`, an extension of the `regular` global constraint. Section 5 describes the linear formulations for three different ETP as well as the column generation process for each of these formulations. Section 6 contains preliminary computational results. The final section presents our conclusion and perspectives.

2 Problem Statement

Employee Timetabling Problems come in many forms. The ones adressed here are essentially divided into two major classes: the anonymous and the personalized scheduling. In the anonymous version we are only interested in building a set of valid work schedules — employees are considered interchangeable. This problem is in fact equivalent to the shift scheduling problem. This paper mostly address the anonymous case, but gives some insight on how to deal with individual schedules.

The definitions and notation given in this section apply to ETP where the *planning horizon* (ex. one day) can be partitioned as a sequence of T consecutive time intervals called *periods* or *shifts*.

Activities. Different types of activities must be fulfilled by the employees on the planning horizon. Let W denote the set of these work activities. An estimation of the *workforce requirement* of each activity is given on the whole planning duration. For each activity a and each period t, r_{at} specifies the number of workers required to achieve activity a during period t. Eventually, c_{at} will denote the cost of assigning one employee to activity a at period t.

Besides work activities, we distinguish three other activities : break (p), rest (o) and lunch (l). These activities are not subjected to costs and workforce but they may be involved in specific constraints. Let $A = W \cup \{o, b, l\}$ denote the set of generalized activities.

Valid Schedules. A *schedule* is an assignment $s : [1..T] \longrightarrow A$ where $s(t)$ stands for the activity to perform at period t. Alternatively, schedule s can be expressed by a binary matrix $B^s = (b^s_{at})_{a \in A, t \in [1..T]}$ where $b^s_{at} = 1$ if $s(t) = a$ and $b^s_{at} = 0$ otherwise. $\mathcal{S}$ denotes the set of *valid schedules* that are individual schedules satisfying all regulation constraints.

Cost and Satisfaction. Usual objectives in ETP are the minimization of the overall cost or the maximization of employee satisfaction. These criteria can be formulated by considering the cost c^s for the company of allocating schedule s to an employee. Such a cost can also represent the degree of dissatisfaction for an employee being assigned to schedule s. The objective is then to minimize the sum of the costs of the schedules assigned to each employee. In this paper, c^s is computed as the sum of the costs of performing activity $s(t)$ at period t: $c^s = \sum_{t=1}^{T} c_{s(t)t}$.

Regulation Constraints. Constraints describing permitted schedules come from legislation and contractual agreements. Three kinds of constraints are usually encountered:

– Some activities are not allowed to be performed at some times.
– Restrictions are imposed on the number of periods or on the number of consecutive periods an employee is assigned to a particular activity or to a group of activities during his schedule (e.g. at most 8 hours of work a day, at least one 15 minute break)
– Some given patterns specify the allowed sequences of activities for the workers (e.g. imposing a break before performing a new activity).

Overcoverage and Undercoverage. In some ETP formulations, the cost of the staff timetabling may include penalties due to overcoverage or undercoverage. For each activity a and period t, let $\hat{c}_{at}$ and $\check{c}_{at}$ be the additional cost when the timetabling covers the workforce demand r_{at} with, respectively, one more employee and one less employee.

Employees. When taking into consideration individual preferences and skills, the definition of a valid schedule becomes different for each employee (or team). Additional constraints restrict the assignment of a given employee to activities following his qualification or to periods following his availabilities.

For each employee $e \in \mathcal{E}$, let $\mathcal{S}_e$ be the set of valid schedules for employee e. The cost of a schedule s may then also differs between employees (for whom schedule s is permitted): let c^{es} be the cost of schedule $s \in \mathcal{S}_e$ when assigned to employee e.

3 Constraint Programming for Shift Scheduling

The subproblem (SP) of computing valid schedules can easily be modeled as a Constraint Satisfaction Problem with T decision variables $s_1, s_2, \ldots, s_T$ and finite discrete domains $D_1, D_2, \ldots, D_T$ all equal to A. There is an obvious one-to-one correspondence between complete instantiations of these variables and schedules $s : [1..T] \longrightarrow A$ by setting $s(t) = s_t$ for all periods t (i.e. $s_t = a$ means that activity a is performed at period t). The set $\mathcal{S}$ of valid schedules corresponds then to the set of all the solutions of this CSP including the regulation constraints.

The high expressiveness and modeling flexibility of constraint programming allows to formulate a wide variety of complex regulation constraints in terms of variables $s_1, \ldots, s_T$ (eventually with the help of additional variables). In particular, a number of global constraints well suited to model such constraints have been introduced in the constraint programming literature. Some of these global constraints are quickly described in Section 3.1. We give in Section 3.2 the example of a set of typical regulation constraints encountered for example in a large store where workers may be assigned to any sales activities.

3.1 Global Constraints for Shift Scheduling

A global constraint in constraint programming is both a way of modelling a specific substructure (common preferably to many decision problems) and a filtering algorithm dedicated to this substructure. In the context of ETP some global constraints of the literature are of great interest. These constraints mainly relate to the allowed values taken by a sequence of decision variables $X = (x_1, \ldots, x_n)$ (in domains $D_1 \times \cdots \times D_n$) together:

Global Cardinality Constraint [8]. This constraint does not consider the ordering of the variables but restricts the number of times each value is distributed on a set of variables. Formally,

$$\texttt{gcc}(< y_1, \ldots, y_m >, < v_1, \ldots, v_m >, X)$$

constrains variable y_j to be equal to the number of appearances of value v_j in the set of variables X. For the ETP, the `gcc` constraint is helpful to give lower and upper bounds on the total amount of work performed over a day.

Stretch Constraint [6]. A `stretch` refers to a subsequence $(x_i, x_{i+1}, \ldots, x_j)$ of variables all assigned to a same value v and that is maximal for this property in terms of inclusion (i.e. $x_{i-1} \neq v$ and $x_{j+1} \neq v$).

$$\texttt{stretch}(X, < v_1, \ldots, v_m >, < l_1^{\min}, \ldots, l_m^{\min} >, < l_1^{\max}, \ldots, l_m^{\max} >)$$

restricts the length of any stretch in X with value v_j to be at least equal to value $l_j^{\min}$ and at most to value $l_j^{\max}$.

For example in ETP, the `stretch` constraint is helpful to indicate that activities must be assigned to a certain number of consecutive periods. Note that `stretch` is more general and can also be applied to cyclic schedules.

Global Sequencing Constraint [9]. This constraint lies somewhere between the two preceding constraints since it looks like the `stretch` constraint with the difference that values do not have to appear consecutively. It can be understood as a set of global cardinality constraints defined on every subsequence of X of a given length. It may occur in the ETP if restrictions on the amount of work performed are also given over a shorter duration, say every three hours.

Regular Constraint [6]. This constraint is able to express complex patterns in a sequence. Formally, given a deterministic finite automaton Π describing a regular language, constraint

$$\texttt{regular}(X, \Pi)$$

restricts the sequence of values taken by the variables of X to belong to the regular language associated to Π. For the ETP, it is useful to enforce sequencing rules for the activities.

3.2 Example of Regulation Constraints

We based our first experimentations on a mostly generic set of regulation constraints. These constraints as well as their formulation in a CSP using global constraints are presented below.

The problem consists of generating valid schedules for employees in a shop with different sales activities $a \in W$. The planning horizon (one day) is decomposed in periods of 15 minutes ($T = 96$). A valid schedule is an assignment $s : [1..T] \longrightarrow A$, where A includes also activities p (break), o (rest) and l (lunch), and that satisfies the following constraints:

1. *Some activities $a \in F_t$ are not allowed to be performed at some periods t.*
2. *s covers between 3 hours and 8 hours of work activities.*
3. *If s is worked for at least 6 hours, then it includes exactly two breaks and one lunch break of 1 hour. Else, it includes only 1 break and no lunch break is planned.*
4. *If performed, the duration of an activity $a \in W$ is at least 1 hour.*
5. *A break (or lunch) is necessary between two different working activities.*
6. *Rest shifts have to be assigned only at the begining and at the end of the day.*
7. *Work activities must be inserted between breaks, lunch and rest stretches.*
8. *The maximum duration of a break is 15 minutes.*

The first condition simply consists of removing the forbidden activities F_t from the initial domain of each variable s_t: $D_t = A \setminus F_t$.

The next two regulation constraints need the definition of additional decision variables. They can then be modeled as explicit constraints as well as, implicitly, by restricting the initial domain of the variables. One way of modeling the second condition is to use one additional variable σ_a for each work activity $a \in W$, with domain $\{0, 1, \ldots, 32\}$ and representing the number of periods assigned to activity a, as well as a variable σ with domain $\{12, \ldots, 32\}$, standing for the total number of working periods. Variables σ_a and s_t may be linked by the `gcc`. In the same manner, we define cardinality variables σ_p, σ_o and σ_l for the non-working activities (break, rest and lunch, respectively). To model the third condition, the domains of σ_l and σ_p are initialized to $\{0, 4\}$ and $\{1, 2\}$ respectively. We can also logically deduce from the whole set of conditions that any valid schedule contains a number of rest periods between 58 and 83. As redundant constraints, we can then reduce the initial domain of σ_o to $\{58, \ldots, 83\}$.

The last five constraints can be modeled with the help of only one `regular` constraint. Indeed, the values permitted by these constraints together for the sequence of variables $(s_1, \ldots, s_T)$ can be described by a single automaton Π. Figure 1 depicts such an automaton when W contains two activities a and b.

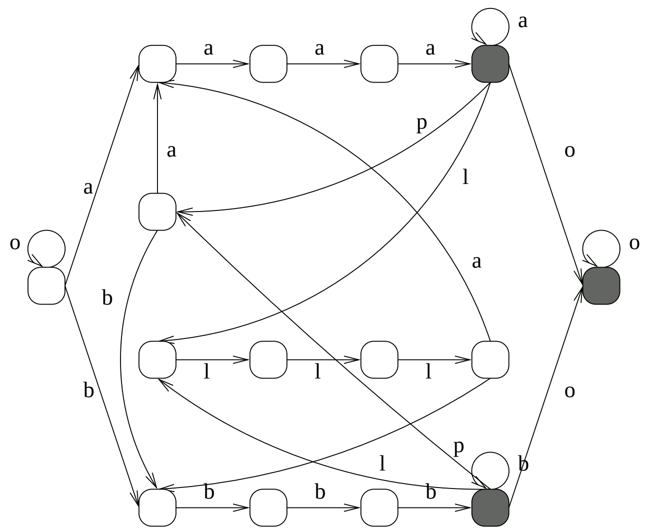

Fig. 1. An automaton for two work activities a and b. The leftmost circle represents the initial state and shaded circles correspond to accepting states.

Given automata Π, the shift scheduling problem described above can be formulated by the following Constraint Satisfaction Problem (CP):

$$\texttt{gcc}(<\sigma_a | a \in A>, <a \in A>, <s_1, \ldots, s_T>) \tag{1}$$

$$\sigma == \sum_{a \in W} \sigma_a \tag{2}$$

$$\sigma < 24 \Rightarrow (\sigma_l == 0 \wedge \sigma_p == 1) \tag{3}$$

$$\sigma \geq 24 \Rightarrow (\sigma_l == 4 \wedge \sigma_p == 2) \tag{4}$$

$$\texttt{regular}(<s_1, \ldots, s_T>, \Pi) \tag{5}$$

$$s_t \in A \setminus F_t, \quad \forall t = 1, \ldots, T \tag{6}$$

$$\sigma_a \in \{0, \ldots, 32\},\ \forall a \in W,\ \sigma \in \{12, 32\} \tag{7}$$

$$\sigma_l \in \{0, 4\},\ \sigma_p \in \{1, 2\},\ \sigma_o \in \{58, \ldots, 83\} \tag{8}$$

4 `cost-regular` Global Constraint

As indicated in Section 3.1, a `regular` constraint is specified using a deterministic finite automaton that describes the regular language to which the sequence must belong. That automaton is then unfolded into a layered directed graph where vertices of a layer correspond to states of the automaton and arcs represent variable-value pairs. This graph has the property that paths from the first layer to the last are in one-to-one correspondence with solutions of the constraint. The existence of a path through a given arc constitutes a support for the corresponding variable-value pair [7].

In the ETP, assigning a given activity at a given period has a cost. For the CP model, this translates to associating costs to variable-value pairs. For this purpose, we define `cost-regular`(X, Π, z, C) constraining X as in `regular` but also requiring that z, a bounded-domain continuous variable, represent the cost of a solution with respect to the constraint, given cost matrix C. This cost is computed as the sum of the costs of the individual arcs in the solution. [1]

Instead of simply maintaining paths, the filtering algorithm must from now on consider the cost of these paths. Supports do not come from just any path but rather from a path whose length falls within the domain of z. To check this efficiently, it is sufficient to compute and maintain shortest and longest paths from the first layer to every vertex and from every vertex to the last layer: if the shortest way to build a path through a given arc is larger than the upper limit of the interval for z, the arc cannot participate in a solution and can thus be removed; if the longest way to build a path through a given arc is smaller than the lower limit of that interval, the arc can again be removed. In this way, domain consistency is achieved for the variables of X. The domain of z can also be trimmed using the shortest and longest paths from the first to the last layer.

The time complexity for the initial computation of the shortest and longest paths is linear in the size of X and in the number of transitions appearing in the automaton, due to the special structure of the graph. Subsequently these paths are updated incrementally.

5 Integer Linear Formulations and Column Generation

This section presents linear programs modeling three different ETP. These linear programs are based on a well-known Dantzig formulation for this kind of problems, involving integer variables indexed by the set of valid schedules. Solving these programs yields an optimal staff timetable by selecting the best set of individual schedules and choosing how many employees (or which employees) will be assigned to each of these schedules.

The first two problems differ on the definition of the overall cost of the staff schedule. In the first one, the overall cost equals the sum of the costs of the schedules assigned to each employees, while in the second one it includes also an additional cost of overcoverage and undercoverage for each work activity on each period. Both models assume that employees are interchangeable. In other words, any schedule in $\mathcal{S}$ is valid for any employee. The problem is then to find how many employees will be assigned to each schedule in $\mathcal{S}$.

On the contrary, the last problem takes into account individual preferences and skills, which requires to define a different set $\mathcal{S}_e$ of valid schedules for each employee e. Here, the problem consists of choosing one schedule in $\mathcal{S}_e$ for each employee e (or none if e is not used).

[1] Note that we could refine the costs by associating one to every combination of variable, value, and state of the automaton.

In this kind of formulations, the number of variables grows exponentially with the number of activities. Beyond two work activities, the set of valid schedules which is the set of solutions of the CSP presented in Section 3, becomes too large to be computed. At this point, we use a column generation procedure to solve the linear relaxation of the integer program.

The column generation algorithm is detailed in Section 5.1 for the first formulation while the corresponding pricing problem alone is described for the two other variants.

5.1 Column Generation

The first ETP problem is to minimize the sum of the costs of the schedules while covering all the workforce requirements. The following linear formulation (ETP) uses an integer variable x_s for each valid schedule $s \in \mathcal{S}$, standing for the number of employees assigned to schedule s:

$$\min \quad \sum_{s \in \mathcal{S}} c^s x_s \tag{9}$$

$$s.t. \quad \sum_{s \in \mathcal{S}} b^s_{at} x_s \geq r_{at} \qquad \forall\, a \in W, \forall\, t \in [1..T], \tag{10}$$

$$x_s \geq 0 \qquad \forall\, s \in \mathcal{S}, \tag{11}$$

$$x_s \in \mathbb{Z} \qquad \forall\, s \in \mathcal{S}. \tag{12}$$

Let $(P) : \big((9) s.t. (10), (11)\big)$ be the linear relaxation of (ETP). The dual (D) of (P) can be written as:

$$\max \quad \sum_{a \in W} \sum_{t=1}^{T} r_{at} \lambda_{at} \tag{13}$$

$$s.t. \quad \sum_{a \in W} \sum_{t=1}^{T} b^s_{at} \lambda_{at} \leq c^s \qquad \forall\, s \in \mathcal{S}, \tag{14}$$

$$\lambda_{at} \geq 0 \qquad \forall\, a \in W, \forall\, t \in [1..T]. \tag{15}$$

Column generation applied to (P) is an iterative algorithm where, at each iteration, the so-called *master problem* (P'), that is linear program (P) restricted to a subset of columns, is solved to optimality. Duality considerations allow to formulate a *pricing problem* (SP) such that: 1) the unfeasiblity of (SP) proves that the optimal solution of the master problem can be extended to an optimal solution of (P) by setting to 0 any variables x_s that are not present in the restricted master program. 2) if (SP) is feasible then its solutions correspond to columns that can improve the solution of the master problem when added at the next iteration.

More precisely, let $\mathcal{S}' \subset \mathcal{S}$ be the subset of valid schedules corresponding to the restricted set of columns of master problem (P') at a given iteration. If x, x'

and λ' are optimal solutions for (P), (P') and (D'), the dual of P', respectively, then weak duality says that:

$$\sum_{a \in W} \sum_{t=1}^{T} r_{at} \lambda'_{at} = \sum_{s \in \mathcal{S}'} c^s x'_s \geq \sum_{s \in \mathcal{S}} c^s x_s.$$

Hence, x' (completed with 0) is an optimal solution of (P) if λ' is a feasible solution of (D), or in other words, if λ' satisfies all constraints (14), that is if:

$$\{s \in \mathcal{S} \mid c^s - \sum_{a \in W} \sum_{t=1}^{T} b^s_{at} \lambda'_{at} < 0\} = \emptyset.$$

At each iteration, the pricing problem (SP) is to find valid schedules $s \in \mathcal{S}$ with negative reduced costs rc'_s, where:

$$rc'_s = c^s - \sum_{a \in W} \sum_{t=1}^{T} b^s_{at} \lambda'_{at}.$$

Section 3.2 gives a CSP formulation (CP) for the problem of finding valid schedules. At each iteration, the pricing problem can be solved by adding to (CP) a negative reduced cost constraint. One way is to add this constraint to (CP) by using global constraint `element` :

$$\sum_{t=1}^{T} (c_{s_t t} - \lambda'_{s_t t}) < 0.$$

We propose a more efficient way to tackle this additional constraint by replacing in (CP) the `regular` constraint (5) by its variant detailed in Section 4:

$$\texttt{cost-regular}(< s_1, \ldots, s_T >, \Pi, z, < c_{at} - \lambda'_{at}, a \in A, t = 1, \ldots, T >). \quad (16)$$

This constraint ensures, as constraint (5), that the sequence of values taken by $< s_1, \ldots, s_T >$ belongs to the language defined by automaton Π of Section 3.2. But it also forces variable z to be equal to the sum of the costs of the variable assignments for $s_1, \ldots, s_T$, the cost of assigning variable s_t to activity $a \in A$ being set to $c_{at} - \lambda'_{at}$. In order to model the negative reduced cost constraint within the (CP) model we just need to define such an additional variable z with the appropriate domain:

$$z \in \,] - \infty, 0[. \quad (17)$$

Hence, the filtering algorithm of `cost-regular` processes simultanously the domains of variables $s_1, \ldots, s_T$ and z such that schedules with non-negative reduced cost are removed from the search space by this algorithm alone.

5.2 Overcoverage and Undercoverage

Overcoverage and undercoverage costs may be taken into account by slightly modifying the previous formulation of (ETP) in this way: for each activity a and period t, let variables $\hat{x}_{at} \in \mathbb{Z}$ and $\check{x}_{at} \in \mathbb{Z}$ represent the overcoverage and the undercoverage respectively. In other words, when N employees are assigned to activity a at period t in the timetable, then $\hat{x}_{at} = N - r_{at}$ and $\check{x}_{at} = 0$ if N is greater than the request r_{at} (overcoverage) and $\check{x}_{at} = r_{at} - N$ and $\hat{x}_{at} = 0$ in the other case (undercoverage).

The modified problem $(ETPl)$ becomes set partitioning problem:

$$
\begin{aligned}
\min \quad & \sum_{s\in\mathcal{S}} c^s x_s + \sum_{a\in W}\sum_{t=1}^{T} (\hat{c}_{at}\hat{x}_{at} + \check{c}_{at}\check{x}_{at}) && && (18)\\
s.t. \quad & \sum_{s\in\mathcal{S}} b^s_{at} x_s + \check{x}_{at} - \hat{x}_{at} = r_{at} && \forall\, a \in W, \forall\, t \in [1..T], && (19)\\
& x_s \in \mathbb{N} && \forall\, s \in \mathcal{S}, && (20)\\
& \hat{x}_{at} \in \mathbb{N}, \check{x}_{at} \in \mathbb{N} && \forall\, a \in W, \forall\, t \in [1..T]. && (21)
\end{aligned}
$$

Since $(ETPl)$ simply contains additional columns (and no more constraints than ETP), the pricing problem associated to a Dantzig-Wolfe decomposition of $(ETPl)$ is identical to the pricing problem for (ETP) (denoted (SP)).

5.3 Individual Shift Scheduling

Most practical timetabling problems take into consideration the preferences and skills of the employees, for instance excluding them from specific activities or work periods. The set $\mathcal{S}_e$ of possible schedules is thus different for each employee $e \in \mathcal{E}$. Such a variant of the ETP can be modeled by the following binary linear program $(ETPe)$:

$$
\begin{aligned}
\min \quad & \sum_{e\in\mathcal{E}}\sum_{s\in\mathcal{S}_e} c^{es} x_{es} && && (22)\\
s.t. \quad & \sum_{e\in\mathcal{E}}\sum_{s\in\mathcal{S}_e} b^s_{at} x_{es} \geq r_{at} && \forall\, a \in W, \forall\, t \in [1..T], && (23)\\
& \sum_{s\in\mathcal{S}_e} x_{es} \leq 1 && \forall\, e \in \mathcal{E}, && (24)\\
& x_{es} \in \{0,1\} && \forall\, e \in \mathcal{E}, \forall\, s \in \mathcal{S}_e, && (25)
\end{aligned}
$$

In this model, x_{es} is a binary variable that is equal to 1 if and only if employee $e \in \mathcal{E}$ is assigned to schedule $s \in \mathcal{S}_e$. Constraints (24) enforce that at most one schedule is assigned to each employee.

Within a column generation approach for this problem, the pricing problem (SPe) can be written[2]:

$$\{(e,s) \in \mathcal{E} \times \mathcal{S}_e \mid c^{es} - \sum_{t=1}^{T} \lambda_{s_t t} + \mu_e < 0\},$$

where (λ, μ) are the current dual values (associated to constraints (23) and (24), respectively) of the master problem at a given iteration of the column generation process. (SPe) can clearly be decomposed as several problems (one for each employee), which are treated in the same way as problem (SP). The CSP to solve for each employee e includes the personal constraints for e as well as the `cost-regular` constraint and the cost variable z with initial domain equal to $]-\infty, -\mu_e[$.

6 Computational Results

We ran some preliminary experiments on a set of generated benchmarks. The data of these instances are based on a realistic ETP for a retail store where employees can be assigned to many sale activities and the work regulation constraints are the ones described in the example of Section 3.2. As a first step, we evaluate the efficiency of our algorithm on the first formulation of ETP presented in Section 5.1. In other words, we assume that employees are interchangeable and that the objective is to minimize the sum of the employee schedule costs while covering the demand curves on the activities.

Generated benchmarks are distributed into eight groups of instances denoted $ETP_1, \ldots, ETP_8$. Each set ETP_n contains 10 instances of the timetabling problem with parameter n indicating the number of work activities. In these instances, the demand curves can require the presence of up to 12 employees at the same period. Even for instances in group ETP_1, the set of valid schedules is too large to be pre-generated quickly. Remember that for benchmarks in ETP_1, schedules are potentially any assignment from the set T of 96 periods to the set A of 4 activities.

For this reason, we directly apply the column generation algorithm to the linear relaxation (P) of program (ETP) as described in Section 5.1. In the rest of this section, we present the details of the implementation of this algorithm, a summary of the computational results obtained on the 80 generated instances and a discussion about these results as well as future research directions for improvements.

Our program was implemented in C++ using ILOG Concert libraries (ILOG Solver 6.0 to solve the CP models and ILOG Cplex 9.0 to solve the linear relaxation), compiled using g++ 3.3 and run on a biprocessor Intel Xeon 2.8GHz under Gnu/Linux 2.6.

[2] Note that constraints (25) are redundant and that in practice x is simply set to be greater or equal to 0. Otherwise, (25) requires the introduction of another dual variable.

The column generation proceeds as follows: An initial minimal set of columns is generated in order to make feasible the master linear program (P) at the first iteration. These columns correspond to mono-activity shedules worked for only four consecutive periods (one hour). Since these schedules are not valid (they violate constraint (7)), we give them "infinite" costs in the LP. At each iteration of the column generation process, the reduced costs returned by the resolution of (P) are used to update the subproblem (SP) of finding improving columns. (SP) is formulated as the CSP of Section 3.2 with the cost constraints (16) and (17) and it is solved by a backtracking algorithm returning the 50 (or less) first computed valid schedules of negative reduced cost. These schedules are then added as new columns to the master program. The algorithm stops when the subproblem of a given iteration is unfeasible. The optimal value of the master problem at the last iteration is then a lower bound LB of the optimal value of the original problem (equal to the linear relaxation bound). To estimate the quality of LB we compute an upper bound UB by solving the integer linear program with the generated columns alone. UB is the value of the best integer solution obtained by the default branch-and-bound of Cplex ran during one hour.

Table 1. Column generation algorithm results on the generated instances ETP

Group	nb	$\Delta LB/UB$		nb iterations		nb columns		CPU in sec.			
		av.	(max)	av.	(max)	av.	(max)	av.	(max)	CPav.	(CPmax)
ETP_1	10	4.9%	(16.6%)	20	(29)	914	(1361)	1.9	(5.7)	0.1	(2.3)
ETP_2	10	5.6%	(15.6%)	51	(76)	2466	(3722)	6.1	(12.0)	0.1	(3.3)
ETP_3	10	5.5%	(9.2%)	76	(106)	3749	(5226)	16.7	(45.6)	0.2	(5.8)
ETP_4	10	4.6%	(8.7%)	137	(204)	6818	(10142)	92.9	(452.4)	0.6	(13.9)
ETP_5	10	5.4%	(12.6%)	132	(207)	6558	(10300)	108.4	(354.4)	0.7	(24.6)
ETP_6	10	5.0%	(11.0%)	203	(337)	10103	(16798)	355.6	(884.6)	1.6	(268.8)
ETP_7	9	5.6%	(7.9%)	244	(337)	12186	(16814)	793.6	(2115.1)	3.1	(130.9)
ETP_8	9	5.4%	(8.5%)	296	(548)	14776	(27377)	950.3	(2531.2)	3.0	(159.9)

Table 1 provides details of the algorithm execution on each problem set ETP_n. The first column gives for each set of 10 instances the number of instances among them that have been processed by column generation in at most one hour. The following columns give, by pair, average and maximal results on these sets of solved instances. These results are, in order: the deviation of LB to the upper bound UB, the number of iterations of the column generation process, the number of generated columns, the total computation time in seconds and the time spent to solve the subproblem (SP) at one iteration.

Computation results given here come from preliminary experiments but give some insight on how to improve the general algorithm. In order to decrease the computational times, we aim to improve the subproblem resolution since the processing time is mainly spent in the CP phases. A good branching strategy has to be found for solving the CSP. For instance, it would be interesting to implement a value ordering heuristic based on the shortest path computed by the filtering algorithm of `cost-regular`. Something similar is performed in the Branch and Price library Maestro [2].

In fact, our random generated instances seem to be really diversified in each group. While, for some of them, we hardly find schedules at each iteration, some others contain a large number of valid schedules. For these last instances, the computation of the (negative reduced cost) schedules is very quick but the number of iterations of the column generation process can then be more important. In these cases, a basic backtracking algorithm has a tendency to provide a set of solutions that are almost identical. Convergence of the column generation process can then be slower when columns added at each iteration are too similar. Many stabilization techniques have been proposed to accelerate convergence ([11]). Another simple way (suggested in [10]) is to add diversity in the set of solutions generated by CP with a multi-start search process, by introducing some randomization in the variable ordering heuristic or by implementing a dedicated diversity constraint.

7 Conclusion

This paper presented a hybrid constraint programming-linear programming solution method for employee timetabing problems. The proposed decomposition applies to several formulations of this kind of problem. By using CP to generate the permitted shift schedules, it also offers a flexible way to tackle the various work regulation constraints that arise in real world timetabling problems.

The optimization criterion on the staff scheduling is handled by LP when assigning shift schedules to employees. With a column generation approach, only "lowest cost" schedules are iteratively generated by CP. The newly introduced global constraint `cost-regular` allows to efficiently take into account the cost of the schedules within their generation process by CP.

The method has now only been implemented to compute lower bounds on generated benchmarks for a first formulation of ETP with general regulation constraints. An obvious continuation of this work is to elaborate branch-and-price algorithms based on these bounds in order to solve at optimality various realistic timetabling problems.

References

1. Benoist, T., Gaudin, E., Rottembourg, B.: Constraint programming contribution to Benders decomposition: a case study. In Proc. 8th Int. Conf. on Principles and Practice of Constraint Programming – CP'02, Springer-Verlag LNCS **2470** (2000) 603–617
2. Chabrier, A.: Maestro: A column and cut generation modeling and search framework. Technical report, ILOG SA (2002)

3. Fahle, T., Junker, U., Karish, S.E., Kohl, N., Vaaben, N., Sellmann, M.: Constraint programming based column generation for crew assignment. Journal of Heuristics **8** (2002) 59–81
4. Fahle, T., Sellmann, M.: Cost based filtering for the constrained knapsack problem. Annals of Operations Research **115** (2002) 73–93
5. Junker, U., Karish, S.E., Kohl, N., Vaaben, N., Fahle, T., Sellmann, M.: A framework for constraint programming based column generation. In Proc. 5th Int. Conf. on Principles and Practice of Constraint Programming – CP'99, Springer-Verlag LNCS **1713** (1999) 261–274
6. Pesant G.: A filtering algorithm for the stretch constraint. In Proc. 7th Int. Conf. on Principles and Practice of Constraint Programming – CP'01, Springer-Verlag LNCS **2239** (2001) 183–195
7. Pesant G.: A regular language membership constraint for finite sequences of variables. In Proc. 10th Int. Conf. on Principles and Practice of Constraint Programming – CP'04, Springer-Verlag LNCS **3258** (2004) 482–495
8. Régin J.-C.: Generalized arc consistency for global cardinality constraints. In Proc. of AAAI'96, AAAI Press/The MIT Press (1996) 209–215
9. Régin J.-C., Puget J.-F.: A filtering algorithm for global sequencing constraints. In Proc. 3rd Int. Conf. on Principles and Practice of Constraint Programming – CP'97, Springer-Verlag LNCS **1330** (1997) 32–46
10. Rousseau, L.-M., Gendreau, M., Pesant, G., Focacci, F.: Solving VRPTWs with constraint programming based column generation. Annals of Operations Research **130** (2004) 199–216
11. Rousseau L.-M.: Stabilization issues for constraint programming based column generation. In Proc. 1st Int. Conf. on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems – CPAIOR'04, Springer-Verlag LNCS **3011** (2004) 402–408
12. Sellmann, M., Zervoudakis, K., Stamatopoulos, P., Fahle, T.: Crew assignment via constraint programming: integrating column generation and heuristic tree search. Annals of Operations Research **115** (2002) 207–225