Cut generation for an integrated employee timetabling and production scheduling problem

Olivier Guyon, Pierre Lemaire, Eric Pinson, David Rivreau

Cut generation for an integrated employee timetabling and production scheduling problem

O. Guyon[a,b,*], P. Lemaire[b], É. Pinson[a], D. Rivreau[a]

[a] *Institut de Mathématiques Appliquées, LISA, 3 place André-Leroy 49008 Angers, France*
[b] *IRCCyN, CNRS ; École des Mines de Nantes, 4 rue Alfred Kastler 44307 Nantes, France*

Abstract

This paper investigates the integration of the employee timetabling and production scheduling problems. At the first level, we manage a classical employee timetabling problem. At the second level, we aim at supplying a feasible production schedule for a set of interruptible tasks with qualification requirements and time-windows. Instead of hierarchically solving these two problems as in the current practice, we try here to integrate them and propose two exact methods to solve the resulting problem. The former is based on a Benders decomposition while the latter relies on a specific decomposition and a cut generation process. The relevance of these different approaches is discussed here through experimental results.

Key words: Employee Timetabling Problem, Production Scheduling Problem, Cut Generation, Benders Decomposition, Energetic Reasoning

1. Introduction

The ultimate purpose of any production system is merely to produce goods to meet some demand. To do so, one must find a production schedule, that is an allocation of human and material resources to the different tasks (or jobs) that have to be processed. However, such a schedule has to take into account the availability of the resources, in particular the human ones: this means that an appropriate employee timetable has to be built up simultaneously to the production schedule. When the production is intended to meet a given, fixed demand, production costs (raw material, energy, ...) do not vary significantly, whatever the actual schedule is, with regard to labor costs. It is hence reasonable to consider that employee timetabling should aim at minimizing labor costs while production scheduling should insure that the production can actually be done on time.

*Corresponding author.
Email addresses: olivier.guyon@uco.fr (O. Guyon)

Although an integrated approach should clearly be adopted when looking for a global optimum, the resulting problem is usually considered as too complex to be solved in practice, and it is decomposed into an assignment part and a scheduling part. As a consequence, even though there is a huge literature on both scheduling problems (e.g., see Pinedo [16]; Leung [14]) and timetabling problems (see Ernst et al. [8]; Soumis et al.[17] for states of the art), only few attempts exist for the integrated problem (see Artigues et al. [3] for an integrated approach and an exhaustive state of the art; see Hooker [11, 12] for a hybrid method mixing Linear Programming and Constraint Programming).

In this paper, we exploit the ideas of Lasserre [13] and Dauzère-Pérès and Lasserre [6] for an integrated job-shop lot-sizing problem. These authors propose to solve the integrated problem alternatively at two different stages: either to compute lot-sizes for a sequence of jobs, or to compute a sequence for given lot-sizes. A similar decomposition approach has also been successfully applied by Detienne et al. to an employee timetabling problem with a given work load [7].

In the remainder of this paper, we propose two exact methods to solve the integrated employee timetabling and production scheduling problem. The problem description is given in Section 2, together with integer linear models. An exact method based on a Benders decomposition is addressed in Section 3 while the fourth section introduces an exact method based on a specific decomposition and cut generation process. We then discuss, in Section 5, about the relevance of these methods by comparing their computational results on generated instances. Some conclusions are finally drawn in Section 6.

2. Problem description and MIP models

We want to schedule a set J of n independent jobs (tasks) with a set O of m resources (operators) over a planning horizon H. Each job $j \in J$ is characterized by a processing time p_j, a time window $D_j = [r_j, d_j]$ and requires an operator $o \in O$ who masters one needed competence $c_j \in C$. Jobs may be interrupted and can be processed by different operators. However, processing instants of any given job cannot overlap in time. An operator cannot process several jobs simultaneously. Each operator $o \in O$ has a set $C_o \subseteq C$ of competences and owns a set $\Omega_o \subseteq \Omega$ of eligible work patterns. A work pattern $\omega \in \Omega$ defines a sequence of actual working time instants and breaks over the whole planning horizon. This formulation permits to take into account several contractual, legal or other constraints (vacations, individual preferences, ...). Each relevant pair work pattern - operator (ω - o) is given a cost η_ω^o standing for the resulting labor cost of assigning work pattern ω to operator o.

Our problem consists in both scheduling the n jobs and assigning a work pattern to each operator in order to satisfy each need for workforce (number of operators and qualification requirements) at minimum cost.

Through this paper, we will consider a descriptive instance with 3 operators (o_1, o_2, o_3), 2 competences (c_1, c_2), 3 jobs (j_1, j_2, j_3) and 2 work patterns ($\omega_1 = [0, 8)$, $\omega_2 = [8, 16)$). Operators o_1 and o_2 only master competence c_1

whereas o_3 masters c_1 and c_2. The characteristics of this instance -Example 1- are given in Table 1.

	j_1	j_2	j_3
r_j	0	2	8
d_j	10	8	16
p_j	9	2	3
c_j	c_1	c_1	c_2

	o_1	o_2	o_3
$\eta^o_{\omega_1}$	10	7	2
$\eta^o_{\omega_2}$	5	3	9

Table 1: Characteristics of Example 1

A feasible solution (cost: 26) for Example 1 consists in assigning ω_1 to o_1 and o_2 and ω_2 to o_3. The following Gantt chart (Figure 1) illustrates the operations processed by operators.

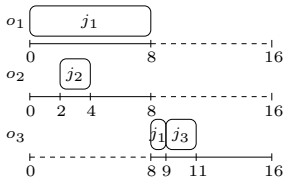

Figure 1: Gantt chart

2.1. Time-indexed formulation

y^o_ω is a binary decision variable where $y^o_\omega = 1$ if work pattern ω is assigned to operator o and $y^o_\omega = 0$ otherwise. Binary variable $x_{jt} = 1$ if and only if one unit of job j is processed at time instant t and binary variable $z_{oct} = 1$ if and only if operator o uses competence c at time t.

Any work pattern ω can be expressed by a boolean vector σ_ω over H such that $\sigma^t_\omega = 1$ if $t \in H$ is a working time instant and $\sigma^t_\omega = 0$ otherwise.

Using notations mentioned above, an intuitive MIP formulation can hence

be given:

$$[Q] : \min \sum_{o \in O} \sum_{\omega \in \Omega_o} \eta_\omega^o \cdot y_\omega^o \tag{1}$$

$$\sum_{\omega \in \Omega_o} y_\omega^o = 1 \quad \forall o \in O \tag{2}$$

$$\sum_{t \in D_j} x_{jt} = p_j \quad \forall j \in J \tag{3}$$

$$\sum_{\substack{j \in J| \\ c_j = c}} x_{jt} = \sum_{\substack{o \in O| \\ c \in C_o}} z_{oct} \quad \forall t \in H, \forall c \in C \tag{4}$$

$$\sum_{c \in C_o} z_{oct} \leq \sum_{\omega \in \Omega_o} \sigma_\omega^t \cdot y_\omega^o \quad \forall t \in H, \forall o \in O \tag{5}$$

$$y_\omega^o \in \{0,1\} \quad \forall o \in O, \forall \omega \in \Omega_o \tag{6}$$

$$x_{jt} \in \{0,1\} \quad \forall j \in J, \forall t \in H \tag{7}$$

$$z_{oct} \in \{0,1\} \quad \forall o \in O, \forall c \in C_o, \forall t \in H \tag{8}$$

Constraints (2) ensure that exactly one work pattern is assigned to each employee. Each job has to be fully processed within its time-window (3). There are as many operators using competence c as processed units of jobs requiring c at each time instant t (4). Each operator uses at most one competence at each instant he is available according to his assigned work pattern, and exactly 0 if he is not (5).

We can see that there are two decision stages in the timetabling process. It is first necessary to set effective working periods to operators by assigning a work pattern to each of them. Then we must decide which competence is really used by the related operator at each time instant.

2.2. Formulation by time intervals and with competence pattern

In order to restrict the number of variables involved in the time-indexed formulation $[Q]$, we propose a second model $[P]$ based on an aggregate of time instants in intervals and on a combination of individual employee qualifications in effective competence patterns.

We first *extract* from H the release date r_j and the due date d_j of each job j and bounds associated with presence intervals for each work pattern $\omega \in \Omega$. Such time instants are then sorted by ascending order and coupled by successive pairs in order to get a partition of H into k_{max} time intervals I_k ($k \in K = \{1, 2, \ldots, k_{max}\}$). Resource needs and requirements (operators and competences) over each time interval I_k are constant. Indeed, by definition, no job can start neither has to end and no operator can start neither has to stop to work at any *unextracted* time instant. It is thus unnecessary to distinguish time instants of each time interval I_k in order to solve the overall problem.

We then combine individual qualifications of operators in competence patterns in order to restrict the number of *operator-competence resources* variables

(z_{oct}) of $[Q]$. These variables do not directly operate in the cost function. It is therefore useless to distinguish operators with exactly the same competences when we assign competences to time intervals. So we introduce a new notation $\Theta \subseteq \mathcal{P}(C)$ to denote the set of competence patterns $\theta \in \Theta$. Each operator $o \in O$ is given a single competence pattern θ_o. Two distinct operators are given the same work pattern θ if and only if they both master only and exactly the same competences $c \in \theta$.

For Example 1, we can define four time intervals: $I_1 = [0, 2)$, $I_2 = [2, 8)$, $I_3 = [8, 10)$ and $I_4 = [10, 16)$ and two competence patterns: $\theta_1 = \{c_1\}$ for o_1 and o_2 and $\theta_2 = \{c_1, c_2\}$ for o_3.

We can therefore propose a second MIP formulation (deriving from $[Q]$):

$$[P] : \min \sum_{o \in O} \sum_{\omega \in \Omega_o} \eta_\omega^o \cdot y_\omega^o \tag{9}$$

$$(2)$$

$$\sum_{\substack{k \in K | \\ I_k \subseteq D_j}} x_{jk} = p_j \quad \forall j \in J \tag{10}$$

$$\sum_{\substack{j \in J | \\ c_j = c}} x_{jk} = \sum_{\substack{\theta \in \Theta | \\ c \in \theta}} z_{\theta ck} \quad \forall k \in K, \forall c \in C \tag{11}$$

$$\sum_{c \in \theta} z_{\theta ck} \leq \sum_{\substack{o \in O | \\ \theta_o = \theta}} \sum_{\substack{\omega \in \Omega_o | \\ I_k \subseteq \omega}} l_k \cdot y_\omega^o \quad \forall k \in K, \forall \theta \in \Theta \tag{12}$$

$$(6)$$

$$x_{jk} \in \{0, \ldots, \min(p_j, l_k)\} \quad \forall j \in J, \forall k \in K \tag{13}$$

$$z_{\theta ck} \in \{0, \ldots, l_k \cdot \text{card}\{o \in O | _{\theta_o = \theta}\}\} \quad \forall \theta \in \Theta, \forall c \in \theta, \forall k \in K \tag{14}$$

where l_k stands for the length of I_k, x_{jk} is the number of units of j processed over I_k and $z_{\theta ck}$ is the number of units of competence c used by competence pattern θ over time interval I_k. The other notations are the same as the ones used for the time-indexed formulation $[Q]$.

We ensure that a time-indexed solution can always be extracted from a solution of formulation $[P]$ by solving a *maximum flow* problem (see Appendix: Time-indexed solution).

Solution methods proposed in the remainder are based on this formulation $[P]$ because they are more effective (in practice) in terms of computing time. Furthermore experimentations reveal that using formulation $[P]$ instead of $[Q]$ reduces substantially the number of columns and rows involved. Indeed, models based on formulation $[P]$ has about 3.3 times less columns and 5.9 less rows than models based on $[Q]$.

3. Benders Decomposition

Due to its intrinsic two-decision-stage structure, it seems quite natural to investigate a Benders Decomposition for solving problem $[P]$. To ensure the existence of a bounded feasible solution, we use a slight modification of $[P]$. Positive slack variables $s_{k\theta}$ are added to constraints (12). A new MIP formulation $[P']$ is thus obtained:

$$[P']: \min \sum_{o\in O}\sum_{\omega\in\Omega_o} \eta_\omega^o \cdot y_\omega^o + M \cdot \left(\sum_{k\in K}\sum_{\theta\in\Theta} s_{k\theta}\right)$$

$$(2), (10), (11)$$

$$\sum_{c\in\theta} z_{\theta ck} - s_{k\theta} \leq \sum_{\substack{o\in O|\\ \theta_o=\theta}} \sum_{\substack{\omega\in\Omega_o|\\ I_k\subseteq\omega}} l_k \cdot y_\omega^o \quad \forall k\in K, \forall\theta\in\Theta$$

$$s_{k\theta} \geq 0 \quad \forall k\in K, \forall\theta\in\Theta \qquad (15)$$

$$(6), (13), (14)$$

Clearly, for a given $M > \sum_{o\in O}(\max_{\omega\in\Omega_o}\eta_\omega^o - \min_{\omega\in\Omega_o}\eta_\omega^o)$ [1], if $[P]$ is feasible, both $[P]$ and $[P']$ achieve the same feasible and optimal solution.

Let now us assume a fixed assignment $\bar{y}$ of work pattern to operators. We can hence introduce the related sub-problem $[SP'(\bar{y})]$ deriving from $[P'(\bar{y})]$ by setting $\bar{y}$ to $[P']$.

$$[SP'(\bar{y})]: \quad M \cdot \min \sum_{k\in K}\sum_{\theta\in\Theta} s_{k\theta}$$

$$(10), (11)$$

$$\sum_{c\in\theta} z_{\theta ck} - s_{k\theta} \leq \sum_{\substack{o\in O|\\ \theta_o=\theta}} \sum_{\substack{\omega\in\Omega_o|\\ I_k\subseteq\omega}} l_k \cdot \bar{y}_\omega^o \quad \forall k\in K, \forall\theta\in\Theta \qquad (16)$$

$$(13), (14), (15)$$

It is quite easy to check that, if $[P]$ is feasible, there is always a bounded feasible solution for $[SP'(\bar{y})]$ (thanks to slack variables $s_{k\theta}$). Also, if the optimum of $[SP'(\bar{y})]$ is zero then $\bar{y}$ is a partial feasible solution for $[P']$.

---

[1] *Proof on the appropriate choice of M.* $\sum_o \max_\omega \eta_\omega^o$ is an intuitive upper bound for $[P]$ and $\sum_o \min_\omega \eta_\omega^o + M$ is a lower bound for $[P']$ for any solution $\{\bar{x}, \bar{y}, \bar{z}, \bar{s}\}$ where it exists at least one positive slack variable $\bar{s}_{k\theta} \geq 1$, i.e. for any unfeasible solution of $[P]$. So to ensure to achieve the same feasible and optimal solution for both $[P]$ and $[P']$, one has to find a large number $M > 0$ such that $\sum_o \max_\omega \eta_\omega^o < \sum_o \min_\omega \eta_\omega^o + M$. □

Furthermore, notice that the constraint matrix of $[SP'(\bar{y})]$ is totally unimodular. Integrity constraints $((13),(14))$ can hence be relaxed since all basic feasible solutions of $[SP'(\bar{y})]$ are integer.

Dual problem $[DSP'(\bar{y})]$ of $[SP'(\bar{y})]$ is:

$$[DSP'(\bar{y})]: \quad \max g_{\bar{y}}(a,b,c,d,e) = \sum_{j\in J} a_j \cdot p_j + \sum_{k\in K}\sum_{\theta\in\Theta} c_{k\theta} \cdot \sum_{\substack{o\in O|\\ \theta_o=\theta}} \sum_{\substack{\omega\in\Omega_o|\\ I_k\subseteq\omega}} l_k \cdot \bar{y}^o_\omega + \sum_{j\in J}\sum_{k\in K} d_{jk}\cdot \min(p_j, l_k) + \sum_{\theta\in\Theta}\sum_{c\in\theta}\sum_{k\in K} e_{\theta ck}\cdot l_k \cdot \text{card}\{o\in O|_{\theta_o=\theta}\}$$

$$\alpha^k_j a_j + b_{kc_j} + d_{jk} \leq 0 \quad \forall j\in J, \forall k\in K \tag{17}$$
$$-b_{kc} + c_{k\theta} + e_{\theta ck} \leq 0 \quad \forall \theta\in\Theta, \forall c\in\theta, \forall k\in K \tag{18}$$
$$-c_{k\theta} \leq 1 \quad \forall k\in K, \forall\theta\in\Theta \tag{19}$$
$$a_j \lesseqgtr 0 \quad \forall j\in J \tag{20}$$
$$b_{kc} \lesseqgtr 0 \quad \forall k\in K, \forall c\in C \tag{21}$$
$$c_{k\theta} \leq 0 \quad \forall k\in K, \forall\theta\in\Theta \tag{22}$$
$$d_{jk} \leq 0 \quad \forall j\in J, \forall k\in K \tag{23}$$
$$e_{\theta ck} \leq 0 \quad \forall \theta\in\Theta, \forall c\in\theta, \forall k\in K \tag{24}$$

Binary parameter $\alpha^k_j = 1$ if and only if $I_k \subseteq D_j$.

Let $\zeta^*(y)$ be the optimal solution of $[SP'(y)]$. $[P']$ can thus be rewritten:

$$[P']: \quad \min \sum_{o\in O}\sum_{\omega\in\Omega_o} \eta^o_\omega \cdot y^o_\omega + M\cdot\zeta^*(y)$$
$$\text{s.t.} \quad (2)\,,(6)$$

Let D be the polyhedron defined by $D = \{(a,b,c,d,e)|_{(17),(24)}\}$. By duality, we get:

$$[P']: \quad \min \sum_{o\in O}\sum_{\omega\in\Omega_o} \eta^o_\omega \cdot y^o_\omega + M\cdot \max_{(a,b,c,d,e)\in D} g_y(a,b,c,d,e)$$
$$\text{s.t.} \quad (2)\,,(6)$$

Since a bounded feasible solution for $[SP'(y)]$ always exists, $[DSP'(y)]$ is bounded. The optimum of $[DSP'(y)]$ therefore matches an extreme point of D. Furthermore, D does not depend on y. So, if we consider the set Q of its q extreme points defined by $Q = \left\{(\tilde{a}_1,\tilde{b}_1,\tilde{c}_1,\tilde{d}_1,\tilde{e}_1),\ldots,(\tilde{a}_q,\tilde{b}_q,\tilde{c}_q,\tilde{d}_q,\tilde{e}_q)\right\}$, $[P']$ should be rewritten as follows:

$$[P']: \quad \min \sum_{o\in O}\sum_{\omega\in\Omega_o} \eta^o_\omega \cdot y^o_\omega + M\cdot \max_{l\in[1..q]} g_y(\tilde{a}_l,\tilde{b}_l,\tilde{c}_l,\tilde{d}_l,\tilde{e}_l)$$
$$\text{s.t.} \quad (2)\,,(6)$$

$[P']$ is thus equivalent to the following master problem :

$$
\begin{array}{lll}
[MP']: & \min & \psi \\
& \text{s.t.} & (2)\,,(6) \\
& & \displaystyle\sum_{o\in O}\sum_{\omega\in\Omega_o} \eta_\omega^o \cdot y_\omega^o + M \cdot g_y(\tilde{a}_l,\tilde{b}_l,\tilde{c}_l,\tilde{d}_l,\tilde{e}_l) \leq \psi \quad \forall l \in [1..q]
\end{array}
$$

If we assume that there exists at least one feasible solution to $[P]$, then a solution y_f such that the optimum of $[DSP'(y_f)]$ is zero necessarily exists. In that case, $\psi = \displaystyle\sum_{o\in O}\sum_{\omega\in\Omega_o} \eta_\omega^o \cdot (y_f)_\omega^o$ and we can state:

$$
\begin{array}{lll}
[MP']: \min & & \displaystyle\sum_{o\in O}\sum_{\omega\in\Omega_o} \eta_\omega^o \cdot y_\omega^o \\
& \text{s.t.} & (2)\,,(6) \\
& & g_y(\tilde{a}_l,\tilde{b}_l,\tilde{c}_l,\tilde{d}_l,\tilde{e}_l) \leq 0 \quad \forall l \in [1..q]
\end{array}
$$

This master problem can be solved by a cut generation process (Benders cut [5]). Let $[RMP'^r]$ be a relaxed master problem with only a small subset Q_r of constraints:

$$
\begin{array}{lll}
[RMP'^r]: & \min & \displaystyle\sum_{o\in O}\sum_{\omega\in\Omega_o} \eta_\omega^o \cdot y_\omega^o \\
& \text{s.t.} & (2)\,,(6) \\
& & g_y(\tilde{a}_l,\tilde{b}_l,\tilde{c}_l,\tilde{d}_l,\tilde{e}_l) \leq 0 \quad \forall l \in Q_r
\end{array}
$$

Using a MIP solver, we can get the optimum $\bar{y}_r$ for $[RMP'^r]$. To check whether $\bar{y}_r$ is a feasible solution for $[MP']$, we solve $[DSP'(\bar{y}_r)]$ and obtain a solution $(\bar{a}_r,\bar{b}_r,\bar{c}_r,\bar{d}_r,\bar{e}_r)$ with cost $\bar{\nu}_r^*$.

If $\bar{\nu}_r^* \leq 0$, $\forall l \in Q \quad g_{\bar{y}_r}(\tilde{a}_l,\tilde{b}_l,\tilde{c}_l,\tilde{d}_l,\tilde{e}_l) \leq 0$. It follows that $\bar{y}_r$ is an optimal solution for $[MP']$ and, by extension, for both $[P']$ and $[P]$.

If $\bar{\nu}_r^* > 0$, constraint of $[MP']$ linked to $(\bar{a}_r,\bar{b}_r,\bar{c}_r,\bar{d}_r,\bar{e}_r)$ from D is not satisfied by solution $\bar{y}_r$. We therefore add the new constraint $g_y(\bar{a}_r,\bar{b}_r,\bar{c}_r,\bar{d}_r,\bar{e}_r) \leq 0$ to $[RMP'^r]$ and solve the resulting problem. Since the number of extreme points is finite, the overall process stops in a finite number of steps.

Computational results showing the relevance of this Benders decomposition are reported in section 5.

4. Specific Cut Generation Process

4.1. Overall process

In this section, we present an alternative exact approach for solving problem $[P]$. As the Benders decomposition, it exploits the splitting of $[P]$ into two

sub problems. A master problem $[MP]$ first assigns a work pattern to each operator. Using this entry, the satellite sub-problem $[SP]$ checks feasibility in terms of processing the whole set of jobs as well as matching competences to operators. If solving $[SP]$ fails in finding a feasible solution for $[P]$, a valid cut is added to $[MP]$ in order to invalidate the current associated assignment. Since $[MP]$ provides a minimum cost assignment of work pattern to operators that both satisfies constraints relating to operators and cuts so far generated, the first assignment fixed by $[MP]$ that is proved to be feasible by $[SP]$ is optimal for $[P]$. Process can hence be stopped. If $[P]$ is unfeasible, process iterates until no feasible solution exists for $[MP]$.

Master problem $[MP]$ can be formalized as follows:

$$\begin{aligned} [MP] : \min c_y = & \sum_{o \in O} \sum_{\omega \in \Omega_o} \eta_\omega^o \cdot y_\omega^o \\ \text{s.t.} \quad & (2)\,, (6) \\ & Cut \end{aligned}$$

where Cut is the set of cuts iteratively added to the model. They invalidate solutions that are not feasible according to the whole set of constraints of $[P]$.

Let us assume a fixed assignment $\bar{y}$ of work pattern to operators as a solution for $[MP]$. We have to check whether $\bar{y}$ is feasible with regards to the other constraints of $[P]$. We thus introduce the satellite sub-problem $[SP(\bar{y})]$:

$$[SP(\bar{y})] : \quad \max f_{\bar{y}} = \sum_{j \in J} \sum_{\substack{k \in K| \\ I_k \subseteq D_j}} x_{jk}$$

$$\sum_{\substack{k \in K| \\ I_k \subseteq D_j}} x_{jk} \leq p_j \quad \forall j \in J \tag{11}$$

$$\sum_{c \in \theta} z_{\theta ck} \leq \sum_{\substack{o \in O| \\ \theta_o = \theta}} \sum_{\substack{\omega \in \Omega_o| \\ I_k \subseteq \omega}} l_k \cdot \bar{y}_\omega^o \quad \forall k \in K, \forall \theta \in \Theta \tag{13), (14}$$

$f_{\bar{y}}$ represents the number of units of jobs which can be scheduled according to $\bar{y}$. Since $[P]$ aims at fully scheduling each job, $\bar{y}$ is feasible for $[P]$ if and only if it leads to an optimal flow value $f_{\bar{y}} = \sum_{j \in J} p_j$.

$[SP(\bar{y})]$ is a *maximum flow* problem on a directed transportation network $G_{\bar{y}} = (X, U)$ with:

- Set of nodes : $X = \{s\} \cup J \cup KC \cup \Theta K \cup \{t\}$
  - s : source
  - t : sink

- Set of edges : $\{\alpha, \beta\} \in U$ - (capacity $\gamma_{\alpha\beta}$) -
  - $\forall j \in J : (s, j) \in U$ with $\gamma_{sj} = p_j$
  - $\forall j \in J, \forall a \in KC : (j, a) \in U \Leftrightarrow (c_j = c_a) \wedge (I_{k_a} \subseteq D_j)$ with $\gamma_{ja} = \min(p_j, l_{k_a})$
  - $\forall a \in KC, \forall b \in \Theta K : (a, b) \in U \Leftrightarrow (I_{k_a} = I_{k_b}) \wedge (c_a \in \theta_b)$ with $\gamma_{ab} = l_{k_a} \cdot \text{card}\{o \in O|_{\theta_o = \theta_a}\}$
  - $\forall b \in \Theta K : (b, t) \in U$ with $\gamma_{bt} = \sum_{\substack{o \in O| \\ \theta_o = b_\theta}} \sum_{\substack{\omega \in \Omega_o| \\ I_{k_b} \subseteq \omega}} l_{k_b} \cdot \bar{y}^o_\omega$

Figures 2 and 3 respectively describe $G_{\bar{y}}$ in the general case and for Example 1.

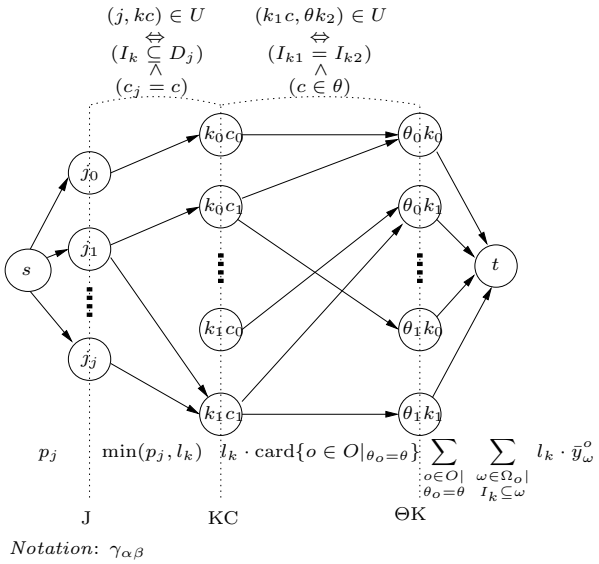

Figure 2: Structure of $G_{\bar{y}}$

If $f_{\bar{y}} < \sum_{j \in J} p_j$, jobs cannot be fully scheduled according to $\bar{y}$. We therefore have to introduce the valid cut $f_{\bar{y}} \geq \sum_{j \in J} p_j$ in order to invalidate $\bar{y}$ from the set of feasible solutions for $[MP]$.

Applying *maximum flow minimum cut* theorem [10] on $G_{\bar{y}}$ (see Figure 4), we can state:

$$f_{\bar{y}} = \sum_{j \in J^-} \gamma_{sj} + \sum_{j \in J^+} \sum_{a \in KC^-} \gamma_{ja} + \sum_{a \in KC^+} \sum_{b \in \Theta K^-} \gamma_{ab} + \sum_{b \in \Theta K^+} \gamma_{bt}$$

with the following notations:

- $\forall u \in U \quad (\phi_u, \gamma_u)$: (flow, capacity) of edge u

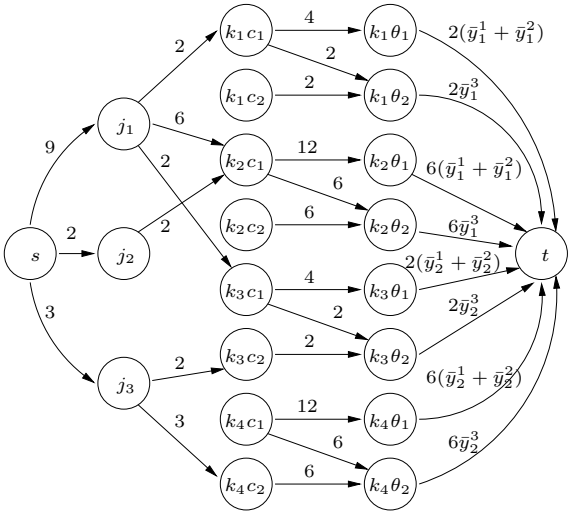

Figure 3: Structure of $G_{\bar{y}}$ for Example 1

- $X = X^+ \cup X^-$ with
  - $X^+ = \{s\} \cup J^+ \cup KC^+ \cup \Theta K^+$
  - $X^- = \overline{X^+} = J^- \cup KC^- \cup \Theta K^- \cup \{t\}$
  - $\forall u = (\alpha, \beta) \in U|_{\{(\alpha \in X^+) \wedge (\beta \in X^-)\}} : \phi_u = \gamma_u$
  - $\forall u = (\alpha, \beta) \in U|_{\{(\alpha \in X^-) \wedge (\beta \in X^+)\}} : \phi_u = 0$

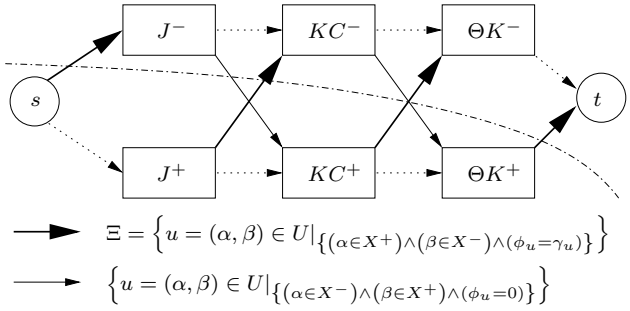

Figure 4: Minimum cut Ξ of $G_{\bar{y}}$

The valid *minimum flow* cut (that invalidates $\bar{y}$) which has to be added to set Cut of $[MP]$ is hence:

$$\sum_{b \in \Theta K^+} \sum_{\substack{o \in O| \\ \theta_o = \theta_b}} \sum_{\substack{\omega \in \Omega_o| \\ I_{k_b} \subseteq \omega}} l_{k_b} \cdot y_\omega^o \geq \nu$$

where:

$$\nu = \sum_{j\in J} p_j - f_{\bar{y}} = \sum_{j\in J^+} p_j - \sum_{j\in J^+} \sum_{a\in KC^-} \min(p_j, l_{k_a}) - \sum_{a\in KC^+} \sum_{b\in\Theta K^-} l_{k_b}\cdot \text{card}\{o\in O|_{\theta_o=\theta_b}\}$$

The only elements depending on $\bar{y}$ which restrict $f_{\bar{y}}$ are the saturated edges $\{u=(\alpha,t)\in U|_{\{\alpha\in\Theta K^+\}}\}$. Thus, to get a flow greater than $f_{\bar{y}}$, one has to find an assignment leading to higher capacities for these latter edges. The optimum assignment y^* for $[P]$ therefore has to satisfy the *minimum flow* cut of $G_{\bar{y}}$.

The MIP formulation of a restricted master problem $[MP^r]$ with only a small subset Q_r of cuts is:

$$[MP^r]: \quad \min c_y = \sum_{o\in O}\sum_{\omega\in\Omega_o} \eta_\omega^o \cdot y_\omega^o$$

$$(2)\,,(6)$$

$$\sum_{b\in\Theta K_i^+} \sum_{\substack{o\in O|\\ \theta_o=\theta_b}} \sum_{\substack{\omega\in\Omega_o|\\ I_{k_b}\subseteq\omega}} l_{k_b}\cdot y_\omega^o \geq \nu_i \quad \forall i\in Q_r$$

Using a MIP solver, we can get the optimum $\bar{y}_r$ for $[MP^r]$. To check whether $\bar{y}_r$ is a feasible solution for $[P]$, we use a *capacity scaling algorithm* [1] to solve $[SP(\bar{y}_r)]$. If $f_{\bar{y}_r} = \sum_{j\in J} p_j$, process stops. Otherwise, *minimum cut flow* of $G_{\bar{y}_r}$ is added to set Cut of $[MP]$ and process iterates. Since there is a finite number of eligible assignments of work pattern to operators, process stops in a finite number of steps. Algorithm 1 gives a formal algorithmic description of the overall process.

Algorithm 1 Specific decomposition and cut generation process

```
LB ← 0
repeat
  ȳ ← solve([MP])
  LB ← c_ȳ
  if f_ȳ < Σ_{j∈J} p_j then
    add minimum flow cut of G_ȳ to the set Cut of [MP]
  end if
until f_ȳ = Σ_{j∈J} p_j
return LB
```

$[MP]$ is a *Multiple-choice Multi-dimension Knapsack Problem* (NP-hard problem). MIP solver is hence really useful in order to solve $[MP]$ up to optimality. We can mention here that several good heuristics for *MMKP* (e.g., see [2]) are described in the literature. Such an effective heuristic would provide a near optimal solution method for $[P]$ that does not require any MIP solver. However, in this paper, we are interested in exact methods, but such a heuristic approximation could be an interesting extension of this work.

4.2. Initial cuts

In the specific cut generation process, cut quality is dominant. Indeed, the more assignments of work pattern to operators a cut invalidates the faster the process converges to the overall optimum. It easily appears that an initialization of cuts should speed up the overall process. This part of the paper is dedicated to initial cuts added to set Cut in a preprocessing stage.

To define initial cuts, we use an *energetic reasoning*. This notion has been broadly successfully used, in particular by Lopez *et al.* [15] and Baptiste *et al.* [4], for scheduling problems. Competences are considered as consumable resources, operators as suppliers and jobs as consumers. The underlying idea of this concept is to define time periods where a strictly positive *required energy consumption* can be established for a subset of jobs. On such periods, resource supplies have to be high enough to allow consumption.

Let $\delta = [\delta_b, \delta_e] \in \Delta$ be a time period with $\Delta \subseteq \mathcal{P}(H)$, $\mathcal{C} \subseteq C$ a subset of competences.

In the remainder, we will use the notation $(a)^+ = \max(a, 0) \quad \forall a \in \Re$.

4.2.1. Required energy consumption of job j over δ

The required energy consumption of j over δ is defined as the difference between p_j and the number of units of j that can be processed apart from δ:

$$u_{j\delta} = \left(p_j - (\delta_b - r_j)^+ - (d_j - \delta_e)^+\right)^+$$

4.2.2. Overall required energy consumption of competence c over δ

The overall required energy consumption of competence c over δ is defined as follows:

$$U_{c\delta} = \sum_{\substack{j \in J| \\ c_j = c}} u_{j\delta}$$

4.2.3. Capacity of available resources $\mathcal{C}$ over δ

Over δ, each operator can work as many time instants he is *available* according to his assigned work pattern. Consequently, as soon as an operator o is *available*, one and only one its competences can be used at each time instant $t \in \delta$:

$$capa_{\delta\mathcal{C}} = \sum_{t \in \delta} \sum_{\substack{o \in O| \\ \{C_o \cap \mathcal{C} \neq \emptyset\}}} \sum_{\omega \in \Omega_o} \sigma_\omega^t \cdot y_\omega^o$$

4.2.4. Energetic *constraint - Initial cut*

A feasible solution for $[P]$ has to satisfy:

$$\sum_{t \in \delta} \sum_{\substack{o \in O| \\ \{C_o \cap \mathcal{C} \neq \emptyset\}}} \sum_{\omega \in \Omega_o} \sigma_\omega^t \cdot y_\omega^o \geq \sum_{c \in \mathcal{C}} U_{c\delta}$$

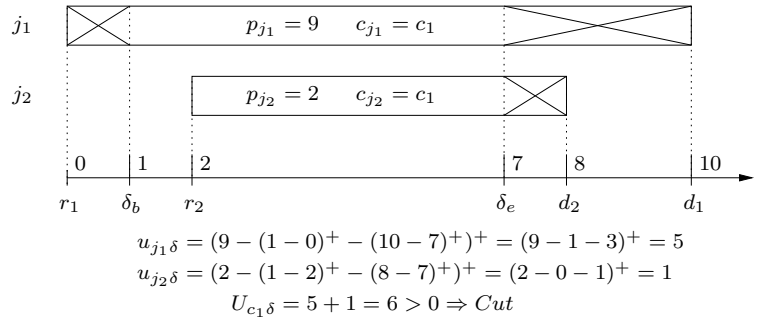

Figure 5: Initial cut over $\delta = [1, 7]$ and with $\mathcal{C} = \{c_1\}$

For Example 1, an *initial valid cut* can be generated over $\delta = [1, 7]$ and with $\mathcal{C} = \{c_1\}$ (see Figure 5).

The number of eligible initial cuts can be considerable, particularly when the number of time periods, δ, is large. Furthermore, many of these cuts might not be ultimately tight in any optimal solution. Hence, we select a set of initial valid cuts.

To introduce the selection used in our experiments, we first have to define $\tau_\delta^{\mathcal{C}} = \frac{\sum_{c \in \mathcal{C}} U_{c\delta}}{\delta_e - \delta_b}$ as the *proportional overall required energy consumption of competence pattern $\mathcal{C}$ over δ*. This meaningful indicator allows us to define likely major cuts. Indeed, if we consider two time periods $\delta \in \Delta$ and $\delta' \in \Delta$ such as $\tau_\delta^{\mathcal{C}} > \tau_{\delta'}^{\mathcal{C}}$, it seems logical to suppose that the cut generated with $\mathcal{C}$ over δ is more *restrictive* than the one generated over δ'.

The approach of selecting initial cuts used in our experiments consists in taking interest in the sorted set Υ of bounds υ associated with presence intervals for each work pattern. Indeed, these time instants are the only ones where number of available operators (and thus competences) can be modified. Let $[\upsilon_i, \upsilon_{i+1}]$ and $[\upsilon_{j-1}, \upsilon_j]$ two time periods with $(i, j) \in |\Upsilon|^2$ and $\upsilon_i < \upsilon_j$. We chose to generate, for each pair $(i, j) \in |\Upsilon|^2$, an initial energetic cut for the time period δ defined as follows:

$$\delta = \underset{\delta' = [\delta'_b, \delta'_e]}{\operatorname{argmax}} \tau_{\delta'}^{\mathcal{C}}$$

where δ'_b and δ'_e are two *extracted* time instants from H, that is to say two bounds associated with time intervals I_k ($k \in K = \{1, 2, \ldots, k_{max}\}$) (see Section 2.2). Besides $\delta'_b \in [\upsilon_i, \upsilon_{i+1}]$ and $\delta'_e \in [\upsilon_{j-1}, \upsilon_j]$.

The preprocessing stage initializing Cut used in our experiments begins by generating an initial energetic cut for each pair $(i, j) \in |\Upsilon|^2$ and each subset of competences $\mathcal{C} \in \{\Theta \cup \mathcal{S}\}$ where Θ is the set of competence patterns and $\mathcal{S}$ is the set of singletons (one competence).

Dominated cuts[2] are then eliminated and the linear relaxation of $[P]$ including the cuts so far generated is solved. Initial cuts matching constraints with small slack variables can hence be considered. In our experiments, we arbitrarily pick up constraints whose ratio of slack variable to second member is less or equal to 12%. We have to notice here that unselected (but non-dominated) initial cuts are therefore kept for the specific decomposition and cut generation process. Indeed as soon as an integer solution for a given master problem $[MP]$ is found, violated unselected initial cuts are added to the set Cut of $[MP]$.

Initial cuts are proved to be really useful because they invalidate many unfeasible assignments for a negligible calculation time. Moreover, by nature, they quickly point out strong assignment restrictions on precise time periods. This way of acting is different from *minimum flow cuts* involved all along process because these latter ones permit an overall view of the problem. A proof of this deduction is that, in practice, *minimum flow cuts* and *initial cuts* do not dominate one another. Thus those two different kinds of cuts are *complementary* and their mixed use is really useful for this specific approach.

5. Experiments

Approaches described previously have been implemented in Java and tested on a PC (Intel Pentium D 930, 3 GHz, 2 GB RAM) which operating system is MS Windows XP. Used MIP solver is ILOG CPLEX 9.1. We use all default options for ILOG CPLEX 9.1 with the exception that we set the parameter *MIP Emphasis* at *Feasibility* because experiments prove that it is an effective choice for each method.

5.1. Test bed

Our test bed is made up of generated feasible instances. Release dates, processing times and margins (time window range) are respectively distributed with an uniform and two binomial laws. A maximum processing time p_{max} and a maximum margin m_{max} are given as parameters to these probability laws. In order to better tally with reality, work patterns match 3-shift-work constraints. Quarter of an hour is the time accuracy unit for work pattern generation. Work patterns are generated over a week (without week-end) and are transposed from week to week in order to cover the whole horizon. Each operator is randomly assigned a set of eligible work patterns and a set of mastered competences. A cost is given to each pair of operator and eligible work pattern. This cost depends on working time instants (day - night), operator's salary level (length of service, qualification, ...) and on operator's requirements. Table 2 summarizes parameters used in order to get our 270 feasible instances. Notice that 3 different instances are generated for each set of parameters.

[2] $\sum_o \sum_\omega (a^o_\omega)^1 \cdot y^o_\omega \geq b_1$ dominates $\sum_o \sum_\omega (a^o_\omega)^2 \cdot y^o_\omega \geq b_2$ if $\forall o\ \forall \omega\ (a^o_\omega)^1 \cdot \frac{b_2}{b_1} \leq (a^o_\omega)^2$

Parameter	min	max	step
m	15	25	10
n	$4 \cdot m$	$6 \cdot m$	m
m_{max}	30	90	30
$\|C\|$	1	5	1
p_{max}	30		
$\|H\|$	480		

Table 2: Instance parameters

5.2. Exact Methods

The generated instances have been solved using the three methods described above: the MIP solver (MIP), the Benders decomposition ($Benders$) and the specific decomposition and cut generation process (Cut). CPU time has been limited to 5 minutes for each.

A first point is that 205 (73.7%) instances are solved by at least one of the three methods; 100 (37.0%) instances are solved by all three methods.

Table 3 reports the percentage of instances solved by each method. Columns *Success* give the percentage of success of the given method. Columns *Gap(F)* give the deviation to the best value found UB^* (found with a CPU time limit of one hour) when the concerned method fails to find optimum. Notice here that when it fails, (MIP) provides an upper bound for the instance whereas both (Cut) and ($Benders$) give a lower bound. Therefore, the deviation to the best value found is given by the ratio $\frac{UB^*}{value}$ for (MIP) and $\frac{value}{UB^*}$ for both (Cut) and ($Benders$).

For the proposed instances (Cut) is the most effective exact method, in the sense that it solves more instances than (MIP) and ($Benders$). Furthermore, (Cut) provides tight bounds when it fails.

We can also notice that ($Benders$) does not seem to be a relevant approach to our problem. It solves fewer instances than the two other methods, and when it succeeds, it requires as much computing time as (MIP) and about 5 times more time than (Cut). ($Benders$) even gives a worst lower bound than (Cut) when it fails.

Table 4 enables us to compare effectiveness (in terms of CPU time) of (MIP) and (Cut). Instances solved by both (MIP) and (Cut) are compared. Notice that (MIP) solves 8 instances where (Cut) fails. In return (Cut) solves 47 instances not solved by (MIP). Column *#inst.* indicates the number of compared instances. Column *# initial cuts* indicates the average number of non dominated initial cuts found in the preprocessing stage. Columns *# selected init. cuts* provides the average number of selected initial cuts. The average number of unselected initial cuts and of *minimum flow cuts* added along the process are respectively given by columns *# other init. cuts* and *# flow cuts*.

We remark (see Table 4) that (Cut) is several orders of magnitude faster than (MIP) for each set of parameters. Such results therefore prove the real

				(MIP)		$(Benders)$		(Cut)	
m	$\|C\|$	m_{max}	# instances	Success	Gap(F)	Success	Gap(F)	Success	Gap(F)
15	1-2	30	18	50.0%	100.0%	33.3%	97.1%	**100.0%**	X
		60	18	88.9%	100.0%	88.9%	98.5%	**100.0%**	X
		90	18	94.4%	100.0%	**100.0%**	X	**100.0%**	X
	3-5	30	27	74.1%	99.9%	3.7%	94.6%	**77.8%**	98.7%
		60	27	**85.2%**	99.4%	18.5%	97.0%	70.4%	98.3%
		90	27	77.8%	99.9%	66.7%	98.6%	**81.5%**	98.5%
$(m = 15)$			135	78.5%	99.9%	47.4%	96.4%	**85.9%**	98.5%
25	1-2	30	18	22.2%	99.8%	11.1%	94.4%	**55.6%**	99.2%
		60	18	50.0%	99.9%	55.6%	97.7%	**72.2%**	99.2%
		90	18	66.7%	99.9%	94.4%	100.0%	**100.0%**	X
	3-5	30	27	18.5%	99.0%	7.4%	95.5%	**33.3%**	97.0%
		60	27	37.0%	99.6%	22.2%	98.0%	**40.7%**	98.9%
		90	27	37.0%	99.7%	**66.7%**	97.9%	**66.7%**	99.0%
$(m = 25)$			135	37.0%	99.6%	40.7%	96.5%	**58.5%**	98.4%
$Total$			270	57.8%	99.6%	44.1%	96.4%	**72.2%**	98.4%

Table 3: Percentage of success for the 3 exact methods

				(MIP)	(Cut)				
m	$\|C\|$	m_{max}	# instances	Time	Time	# initial cuts	# selected init. cuts	# other init. cuts	# flow cuts
15	1-2	30	9/18	50.1s	**11.3s**	141.6	28.3	2.8	2.9
		60	16/18	31.4s	**1.3s**	120.4	27.0	3.0	3.9
		90	17/18	14.9s	**0.6s**	126.3	23.4	1.1	2.6
	3-5	30	18/27	53.8s	**8.3s**	450.1	42.8	13.1	21.4
		60	19/27	62.8s	**19.8s**	434.0	40.7	13.0	19.2
		90	20/27	43.7s	**30.2s**	414.2	25.4	12.3	32.3
$(m = 15)$			99/135	42.8s	**12.8s**	302.8	31.7	8.3	15.4
25	1-2	30	4/18	85.2s	**0.1s**	117.8	13.8	1.3	2.0
		60	9/18	29.0s	**1.9s**	144.1	23.9	1.7	2.6
		90	12/18	82.6s	**10.2s**	135.4	21.8	0.6	3.8
	3-5	30	5/27	30.0s	**18.4s**	622.8	31.4	11.8	16.0
		60	9/27	67.7s	**13.4s**	680.6	28.2	8.6	19.3
		90	10/27	40.8s	**4.0s**	629.3	22.8	4.0	6.6
$(m = 25)$			49/135	56.3s	**8.0s**	386.2	23.9	4.1	8.1
$Total$			148/270	47.3s	**11.2s**	330.4	29.1	6.9	13.0

Table 4: CPU time for (MIP) and (Cut)

interest of this approach for our problem.

6. Conclusions

We have proposed two exact methods for solving an integrated employee timetabling and production scheduling problem. We can especially mention the real interest of an exact method based on a specific decomposition and cut generation process which is several orders of magnitude faster than one of the best current MIP solvers (Ilog Cplex).

Future research directions should address several issues. The most promising one is to reduce CPU time spent in solving the master problems $[MP]$ for the specific decomposition and cut generation approach (Cut). Indeed solving $[MP]$ takes about 98.5% of the overall average CPU time required by this method. One possible way should be to use alternative approaches (Feasiblity Pump [9], Combining MIP and Constraint Programming as Hooker did in a production planning and scheduling problem [11, 12], ...). Designing a specific Branch-and-Bound upon the initial energetic cuts also seems to be an interesting research direction.

Appendix : Time-indexed solution

As mentioned in Section 2.2, a time-indexed solution can always been extracted from a solution of formulation $[P]$ by solving a *maximum flow* problem. Thus to get the full time-indexed solution related to solutions given by both the Benders decomposition (Section 3) and the specific cut generation process (Section 4), one has to find a *maximum flow* on a directed transportation network $G_{index} = (X, U)$ given below (Figure 6).

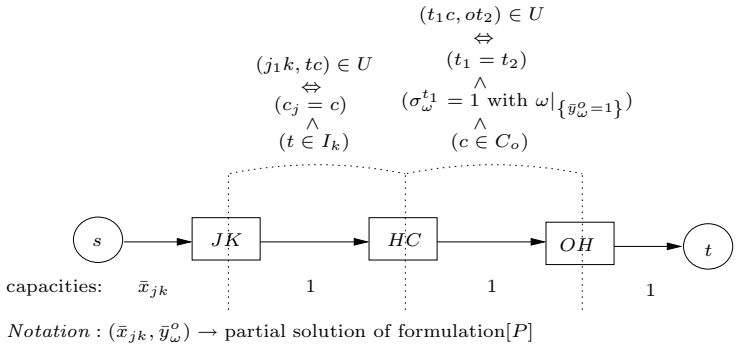

Figure 6: Structure of G_{index}

A full time-indexed solution for $[P]$ is given as follows (ϕ_u denoting the flow on the edge $u \in U$):

- $\forall j \in J, \forall t \in H \quad x_{jt}^* = 1 \Leftrightarrow \phi_{(jk)\rightarrow(tc_j)} = 1$ with $k \in K|_{\{t \in I_k\}}$
- $\forall o \in O, \forall c \in C_o, \forall t \in H \quad z_{oct}^* = 1 \Leftrightarrow \phi_{(tc)\rightarrow(ot)} = 1$

Acknowledgements

The authors would like to express their gratitude to the anonymous referees for their helpful comments and suggestions.

References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows - Theory, Algorithms, and Applications*. Prentice Hall, 1993.

[2] M. M. Akbar, E. G. Manning, G. C. Shoja, and S. Khan. Heuristic solutions for the multiple-choice multi-dimension knapsack problem. In *ICCS '01: Proceedings of the International Conference on Computational Science-Part II*, pages 659–668, London, UK, 2001. Springer-Verlag.

[3] C. Artigues, M. Gendreau, and L.-M. Rousseau. A flexible model and a hybrid exact method for integrated employee timetabling and production scheduling. *CORS / Optimization Days 2006 Joint Conference*, 2006.

[4] P. Baptiste, C. Le Pape, and W. Nuijten. Satisfiability tests and time-bound adjustments for cumulative scheduling problems. *Annals of Operations Research*, 92:305–333, 1999.

[5] J. Benders. Partitioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4:238–252, 1962.

[6] S. Dauzère-Pérès and J. B. Lasserre. Integration of lotsizing and scheduling decisions in a job-shop. *European Journal of Operations Research*, 75:413–426, 1994.

[7] B. Detienne, L. Péridy, E. Pinson, and D. Rivreau. Cut generation for an employee timetabling problem. *European Journal of Operational Research*, In Press, Corrected Proof, 2008.

[8] A. T. Ernst, H. Jiang, M. Krishnamoorthy, and D. Sier. Staff scheduling and rostering : A review of applications, methods and models. *European Journal of Operational Research*, 153:3 – 27, 2004.

[9] M. Fischetti, F. Glover, and A. Lodi. The feasibility pump. *Mathematical Programming*, 104:91–104, 2005.

[10] L. R. Ford and D. R. Fulkerson. *Flows in Networks*. Princeton University Press, Princeton, 1962.

[11] J. N. Hooker. A hybrid method for planning and scheduling. *Constraints*, 10:385–401, 2005.

[12] J. N. Hooker. Planning and scheduling by logic-based benders decomposition. *Operations Research*, 55:588–602, 2007.

[13] J. B. Lasserre. An integrated model for job-shop planning and scheduling. *Management Science*, 38:1201–1211, 1992.

[14] J. Y.-T. Leung, editor. *Handbook of Scheduling: Algorithms, Models, and Performance Analysis*. CRC Press, Boca Raton, FL, USA, 2004.

[15] P. Lopez, J. Erschler, and P. Esquirol. Ordonnancement de tâches sous contraintes : une approche énergétique. *Automatique, Productique, Informatique Industrielle*, 26:453–481, 1992.

[16] M. Pinedo. *Scheduling: theory, algorithms and systems*. Prentice Hall, second edition, 2004.

[17] F. Soumis, G. Pesant, and L.-M. Rousseau. *Gestion de Production et Ressources Humaines*, chapter 4, Gestion des horaires et affectation du personnel. Presses Internationales Polytechnique, 2005.