Near optimal and optimal solutions for an integrated employee timetabling and production scheduling problem

Olivier Guyon, Pierre Lemaire, Eric Pinson, David Rivreau

Near optimal and optimal solutions for an integrated employee timetabling and production scheduling problem

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

[*] *LISA-IMA, 3 place André-Leroy 49008 Angers, France*
(e-mail: {olivier.guyon, eric.pinson, david.rivreau}@uco.fr)
[**] *IRCCyN, CNRS ; École des Mines de Nantes*
4 rue Alfred Kastler 44307 Nantes, France
(e-mail: pierre.lemaire@emn.fr)

Abstract: This paper deals with integrated employee timetabling and production scheduling problem. At the first level, we have to 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 using a hierarchical approach, we try here to integrate the two stages and propose an exact method based on a specific decomposition and a cut generation process to solve the resulting problem. This exact two-stage procedure has the double advantage to converge to optimal solutions but also to find quickly near optimal solutions. The relevance of this approach is discussed here through experimental results.

Keywords: Employee Timetabling Problem, Production Scheduling Problem, Cut Generation, Feasibility Pump, Energetic Reasoning

1. INTRODUCTION

In production systems, production scheduling and employee timetabling are two problems generally considered as too complex to be treated simultaneously. On the one hand, production scheduling aims at allocating resources (human or material) to different tasks (jobs) that have to be processed. On the other hand, the objective of employee timetabling is most often to minimize labor costs (we refer to Ernst et al. (2004) and Soumis et al. (2005) for a state of the art of employee timetabling problems). Although it is clear that the best approach to solve the resulting problem would be to consider the integrated problem, it is often decomposed in practice into an assignment part and a scheduling part. This is essentially due to the fact that no effective formal algorithm has yet been considered as good enough to solve the integrated problem. Nevertheless several attempts (with encouraging results for some of them) have been made in both production and transportation domains (see in particular Artigues et al. (2006, 2009) for integrated approaches and an exhaustive state of the art, and Hooker (2005) for a hybrid method mixing Linear Programming and Constraint Programming). In this paper, we exploit the idea proposed in Lasserre (1992) and developed later in Dauzère-Pérès and Lasserre (1994) for an integrated job-shop lot-sizing and scheduling problem. In these works, the authors proposed to solve the integrated problem by alternatively solving it at two different levels, one in which lot sizes are computed for a sequence of jobs on each machine, and one in which a sequence is computed given fixed lot sizes. Such a decomposition approach has been successfully applied in Detienne et al. (2009) to an employee timetabling problem with a given work load.

We integrate here the two sub problems and propose an exact method which has the double advantage to converge to optimal solutions but also to find quickly near optimal solutions. Section 2 states formulations of the problem. An exact two-stage procedure based on a specific decomposition and a cut generation process is addressed in Section 3. We then discuss, in Section 4, about the relevance of the method by comparing its computational results with an ILP solver (Ilog Cplex 9.1) on generated instances. Some conclusions are finally drawn in Section 5.

We may mention that an approach based on a Benders decomposition has also been experimented. We will not develop it here because experimental results do not indicate any advantage of using this method for our integrated problem.

2. PROBLEM DESCRIPTION AND ILP 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, at most one unit of a given job can be processed at each time $t \in H$. 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^o_ω 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.

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:

$$[Q] : \min \sum_{o \in O} \sum_{\omega \in \Omega_o} \eta^o_\omega \cdot y^o_\omega \tag{1}$$

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

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

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

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

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

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

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

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 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 competence pattern θ if and only if they both master only and exactly the same competences $c \in \theta$.

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

$$[P] : \min \sum_{o \in O} \sum_{\omega \in \Omega_o} \eta^o_\omega \cdot y^o_\omega \tag{9}$$

(2)

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

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

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

(6)

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

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

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.

Solution methods proposed in the remainder are based on this formulation $[P]$ because it proves to be (in practice) more effective in terms of computing time.

3. SOLUTION PROCEDURE

3.1 Overall process

We present here an approach exploiting 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 *feasibility cut* is added to $[MP]$ in order to invalidate the current associated assignment.

If $[MP]$ is solved up to optimality, the process stops with an optimal solution for $[P]$ as soon as solving $[SP]$ succeeds. This is a relevant exact method, but computing times may sometimes become prohibitive (see Guyon et al. (2008)).

As we want quickly near optimal solutions, only a feasible solution for $[MP]$ is sought for. When $[SP]$ succeeds, such a solution is feasible for $[P]$. In this case we add a *boundary constraint* to $[MP]$ to ensure that future assignments have a lower cost. Note that iterating this process until $[MP]$ is unfeasible provides again an exact procedure; the optimum for $[P]$ is the last found feasible assignment. Algorithm 1 gives a formal algorithmic description of the process.

Algorithm 1 Cut generation process

UB $\leftarrow \infty$
repeat
  $\bar{y} \leftarrow$ feasible solution - cost : $c_{\bar{y}} < UB$ - for ($[MP]$)
  if $[SP_{\bar{y}}]$ is unfeasible **then**
    add a *feasibility cut* to $[MP]$
  else
    UB $\leftarrow c_{\bar{y}}$
    add a *boundary constraint* to $[MP]$
  end if
until ($[MP]$ is unfeasible) or (stop criteria)

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

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

$$\text{s.t.} \quad (2) \text{ and } (6)$$

$$Cut$$

where Cut is the set of *feasibility cuts* and *boundary constraints* iteratively added to the model.

3.2 Cut structure

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 $[SP(\bar{y})]$:

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

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

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

$$(13) - (14)$$

$f_{\bar{y}}$ 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 $\gamma_{ab} = l_{k_a} \cdot |\{o \in O/_{\theta_o = \theta_a}\}|$
  - $\forall b \in \Theta K : (b, t) \in U$
    with $\gamma_{bt} = \sum_{o \in O/_{\theta_o = b_\theta}} \sum_{\omega \in \Omega_o/_{I_{k_b} \subseteq \omega}} l_{k_b} \cdot \bar{y}_\omega^o$

Figure 1 describes $G_{\bar{y}}$.

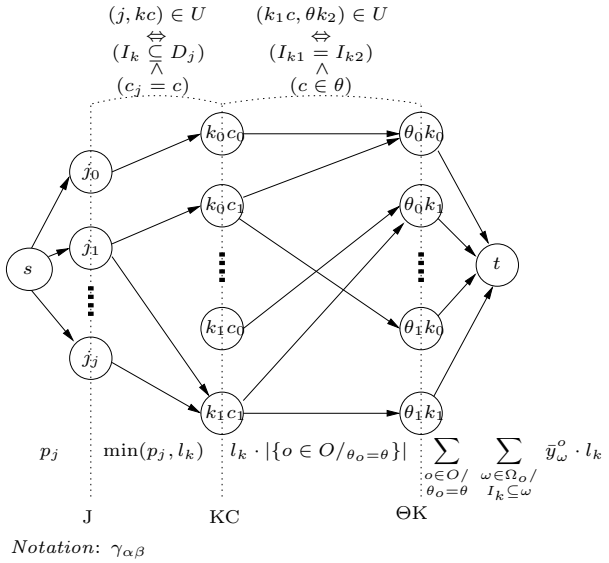

Fig. 1. Structure of $G_{\bar{y}}$

If $f_{\bar{y}} = \sum_{j \in J} p_j$, $\bar{y}$ is a feasible solution for $[P]$, we therefore add the following intuitive *boundary constraint* to $[MP]$:

$$\sum_{o \in O} \sum_{\omega \in \Omega_o} \eta_\omega^o \cdot y_\omega^o < c_{\bar{y}}$$

In return, 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 *feasibility 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 stated in Ford and Fulkerson (1962) on $G_{\bar{y}}$ (see Figure 2), we have:

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

with the following notations:

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

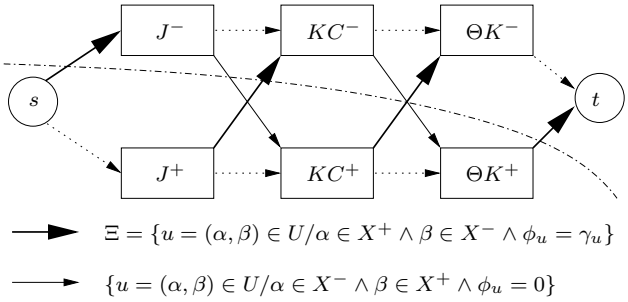

Fig. 2. Maximum flow cut Ξ of $G_{\bar{y}}$

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

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

where:

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

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 *maximum flow* cut of $G_{\bar{y}}$.

3.3 Solving the master problem

The MIP formulation of a restricted master problem $[MP^{bf}]$ with only a small subset Q_b of *boundary constraints* and a small subset Q_f of *feasibility cuts* is:

$$\begin{aligned} [MP^{bf}]: \quad \min c_y = & \sum_{o \in O} \sum_{\omega \in \Omega_o} \eta^o_\omega \cdot y^o_\omega \\ & \text{(2) and (6)} \\ & \sum_{o \in O} \sum_{\omega \in \Omega_o} \eta^o_\omega \cdot y^o_\omega < (c_{\bar{y}})^i \quad \forall i \in Q_b \\ & \sum_{b \in \Theta K^+_i} \sum_{\substack{o \in O| \\ \theta_o = \theta_b}} \sum_{\substack{\omega \in \Omega_o| \\ I_{k_b} \subseteq \omega}} l_{k_b} \cdot y^o_\omega \geq \nu_i \quad \forall i \in Q_f \end{aligned}$$

$[MP]$ is a *0-1 Multidimensional Multiple-choice Knapsack Problem* (NP-hard problem) whereas $[SP(\bar{y})]$ is a *Maximum Flow* problem (P-problem). The main difficulty of this method consists thus in quickly finding a feasible solution for $[MP]$.

Two approaches have been experimented. The first one consists in using an ILP solver tuned to stop as soon as a feasible integer solution has been found. The second one is to use the *Feasibility Pump* heuristic.

Feasiblity Pump The *Feasibility pump*, first proposed in Fischetti et al. (2005) and then successively improved in Bertacco et al. (2005) and Achterberg and Berthold (2007), is a primal heuristic for finding feasible solutions of Mixed Integer Programs. The fundamental idea of this approach is to generate two (hopefully convergent) trajectories of points that satisfy feasibility in a complementary but partial way: one satisfies the linear constraints, the other the integer requirement. These are produced by alternatively rounding a LP-feasible point (x^*) to an integer point ($\tilde{x}$) and finding a closest (with respect to a predefined distance function) LP-feasible point, which is then used as the new (x^*) for the next iteration step. Process thus iterates until this latter so-called *closest LP-feasible point* is also an integer feasible point. The procedure is illustrated in Figure 3.

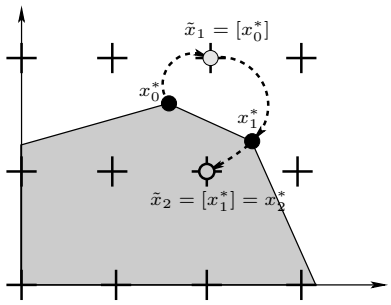

Fig. 3. Fundamental procedure of the Feasibility Pump

The relevance of these two distinct approaches experimented here is discussed in Section 4.

3.4 Speeding up by initial cuts

In the specific cut generation process, cut's 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 to initialize $[MP]$.

To define initial cuts, we use an *energetic reasoning*. This notion has been broadly successfully used, in particular in Lopez et al. (1992) and Baptiste et al. (1999), 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 $(a)^+ = \max(a, 0) \quad \forall a \in \mathbb{R}$.

We define the *required energy consumption ($u_{j\delta}$) of j over δ* as the difference between p_j and the number of units of j that can be processed apart from δ:

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

The *overall required energy consumption ($U_{c\delta}$) of competence c over δ* is thus defined as follows:

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

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* and masters a competence $c \in \mathcal{C}$, one and only one competence of o is available at each time instant $t \in \delta$. We can hence define the *capacity of available resources $\mathcal{C}$ over δ* as:

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

Energetic *constraint - Initial cut* A feasible solution for $[P]$ has hence to satisfy:

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

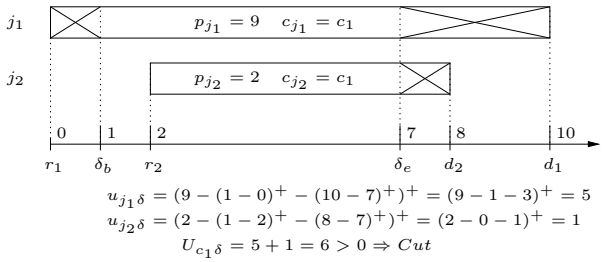

Fig. 4. Initial cut over $\delta = [1, 7]$ and with $\mathcal{C} = \{c_1\}$

The number of eligible initial cuts can be considerable, particularly when the number of time periods, δ, is large. Further, such a process can lead to a large number of cuts that 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 $\delta'_e \in [\upsilon_{j-1}, \upsilon_j]$.

In our experiments, set Θ of competence patterns is the reference set used to get initial energetic cuts. Singletons (one competence) are also used.

An intensive way of selecting initial cuts is to solve the linear relaxation of $[P]$ with cuts so far generated. Initial cuts matching constraints with small slack variables can hence be considered. In our experiments, we pick up constraints whose ratio of slack variable to second member is less or equal to 0.12. We have to notice here that unselected 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 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 *maximum flow cuts* involved all along process because these latter ones permit an overall view of the problem. Thus those two different kinds of cuts are *complementary* and their mixed use is really useful for this specific approach.

All methods described in this paper are initialized with selected initial cuts because it proves to be really more effective in practice.

4. 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 ILP solver is ILOG CPLEX 9.1.

4.1 Test bed

Our test bed is made up of generated 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 $marge_{max}$ 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 1 summarizes parameters used in order to get 270 feasible instances. Notice that 3 different instances are generated for each set of parameters.

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

Table 1. Instance parameters

4.2 Results

Table 2 reports the results of the approaches described in this paper. We use the ratio UB^*/UB as the indicator of effectiveness for each method. UB^* stands for the tightest known upper bound of the solved instance (found with a CPU time limit of one hour) and UB is the value found by the concerned method. Column UB^*/UB and σ respectively give the average and the standard deviation of the ratios UB^*/UB.

- (MIP) stands for the resolution of the direct formulation by time intervals and with competence pattern by an ILP solver.
- $(Cut : [MP] \hookrightarrow MIP)$ stands for the two-stage procedure (Cut) using an ILP solver to solve each master problem (parameter *IntSolLim* set to 1 for *Ilog Cplex 9.1*).
- $(Cut : [MP] \hookrightarrow FP)$ stands for the two-stage procedure (Cut) using the *Feasibility Pump* heuristic to solve each master problem.

In all three cases selected initial cuts (see Section 3.4) are added to the model in a pre-processing stage.

			(Cut)			
CPU limit	(MIP)		$([MP] \hookrightarrow MIP)$		$([MP] \hookrightarrow FP)$	
	UB^*/UB	σ	UB^*/UB	σ	UB^*/UB	σ
1s	70.76 %	(20.82 %)	87.24 %	(19.04 %)	95.38 %	(10.38 %)
5s	86.98 %	(19.21 %)	95.64 %	(11.67 %)	98.98 %	(3.52 %)
10s	92.00 %	(15.89 %)	98.24 %	(6.76 %)	99.52 %	(1.08 %)
30s	97.22 %	(9.29 %)	99.11 %	(4.62 %)	99.82 %	(0.52 %)
60s	99.46 %	(2.64 %)	99.53 %	(3.21 %)	99.88 %	(0.41 %)
300s	99.87 %	(0.44 %)	99.91 %	(0.48 %)	99.97 %	(0.15 %)

Table 2. Quality of upper bound over CPU time limit

We first remark (see Table 2) that both $(Cut : [MP] \hookrightarrow MIP))$ and $(Cut : [MP] \hookrightarrow FP))$ found really quickly tight upper bounds. The specific decomposition and cut generation process (Cut) is hence appropriate for the problem presented in this paper.

We can also observe that $(Cut : [MP] \hookrightarrow FP))$ is always more effective in both terms of average and standard deviation than $(Cut : [MP] \hookrightarrow MIP))$. This means that $(Cut : [MP] \hookrightarrow FP))$ is not only more accurate but also more reliable than $(Cut : [MP] \hookrightarrow MIP))$. The *Feasibility Pump* heuristic is thus really effective for solving master problems (*0-1 Multiple Choice and Multiple Dimension Knapsack*).

5. CONCLUSIONS

We have proposed an exact method for solving an integrated employee timetabling and production scheduling problem. This exact approach is based on an effective specific decomposition and cut generation process. This exact two-stage procedure has the double advantage to converge to the optimum but also to find quickly near optimal solutions.

The main points of this paper are based on the real interest of using a decomposition approach to find optimal and near optimal solutions for the integrated problem, and on the proved effectiveness of the *Feasibility Pump* heuristic for quickly finding a good feasible solution for a *0-1 Multidimensional Multiple-choice Knapsack Problem*.

REFERENCES

Achterberg, T. and Berthold, T. (2007). Improving the feasibility pump. *Discrete Optimization*, Special issue: Mixed Integer Programming, 7786.

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

Artigues, C., Gendreau, M., and Rousseau, L.M. (2009). Solving an integrated employee timetabling and job-shop scheduling problem via hybrid branch-and-bound. *Computers & Operations Research*, 36, 2330–2340.

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

Bertacco, L., Fischetti, M., and Lodi, A. (2005). A feasibility pump heuristic for general mixed-integer problems. Technical Report OR/05/5, DEIS - Università di Bologna, Italy.

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

Detienne, B., Péridy, L., Pinson, E., and Rivreau, D. (2009). Cut generation for an employee timetabling problem. *European Journal of Operational Research*, 197, 1178–1184.

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

Fischetti, M., Glover, F., and Lodi, A. (2005). The feasibility pump. *Mathematical Programming*, 104, 91–104.

Ford, L.R. and Fulkerson, D.R. (1962). *Flows in Networks*. Princeton University Press, Princeton.

Guyon, O., Lemaire, P., Pinson, E., and Rivreau, D. (2008). Couplage planification/ordonnancement: une approche par décomposition et génération de coupes. In *Actes de la 7e Conférence Internationale de Modélisation et Simulation*, 1376–1385.

Hooker, J.N. (2005). A hybrid method for planning and scheduling. *Constraints*, 10, 385–401.

Lasserre, J.B. (1992). An integrated model for job-shop planning and scheduling. *Management Science*, 38, 1201–1211.

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

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