CAHIER DU LAMSADE

Laboratoire d'Analyse et Modélisation de Systèmes pour l'Aide à la Décision
(Université Paris-Dauphine)
Unité de Recherche Associée au CNRS n° 825

COMPARAISON DE DEUX APPROCHES SUR UN PROBLÈME DE PLANIFICATION D'HORAIRES : LA CONSTRUCTION DE CYCLES DE TRAVAIL

CAHIER N° 142
octobre 1996

Ariane PARTOUCHE [1]

reçu : mai 1996.

[1] *LAMSADE, Université Paris-Dauphine, Place du Maréchal De Lattre de Tassigny, 75775 Paris Cedex 16 (e-mail: partouche@lamsade.dauphine.fr).*

Comparaison de deux approches sur un problème de planification d'horaires : la construction de cycles de travail

Ariane Partouche
Université Paris Dauphine-Lamsade
Place du Maréchal de Lattre de Tassigny
75 775 Cedex 16 Paris
e-mail: partouche@lamsade.dauphine.fr

Table des matières

Comparison of two approaches on a crew scheduling problem: the crew rostering

Abstract

This paper deals with the crew rostering problem, a sub-problem of the crew scheduling problem. A roster is a cyclic working program on few consecutive weeks concerning workers on multiple-shifts.

Our purpose is to build a cyclic roster, succession of shifts and days-off, so as to cover a requirement curve while respecting working regulations and maximizing the social satisfaction, measured by the level of respect of a set of soft constraints.

We suggest and compare two approaches of the same model: the first is based on 0/1 linear programming and the second uses constraint logic programming.

KEY WORDS: cyclic roster, schedule, shifts, linear programming, constraint logic programming.

Comparaison de deux approches sur un problème de planification d'horaires : la construction de cycles de travail

Résumé

Cet article traite de la construction de cycles étiquetés, cas particulier des problèmes de planification des horaires. Un cycle correspond à un programme de travail sur plusieurs semaines pour des agents en horaires décalés qui effectuent un roulement.

Construire un cycle consiste à agencer des vacations et des repos de manière à couvrir une charge tout en respectant la réglementation du travail et en maximisant la satisfaction sociale mesurée par le niveau de respect d'un ensemble de contraintes « flexibles ».

Nous proposons et comparons deux approches du même modèle : la première est basée sur la programmation linéaire en variables 0/1, la seconde utilise la programmation par contraintes.

MOTS CLÉS : grilles, cycles de travail, vacations, horaires, programmation linéaire, programmation par contraintes.

1 Introduction

Les problèmes de planification d'horaires se posent dans les organisations où la semaine de travail des agents ne couvre pas intégralement la semaine d'activité. Il en va ainsi des entreprises opérant au-delà des horaires administratifs de huit heures par jour, cinq jours par semaine. Citons en particulier les grandes structures du secteur tertiaire comme les entreprises de transport (compagnies aériennes, rail ou transport urbain), les hôpitaux, les standards téléphoniques, quelques banques, etc. Afin d'assurer un service sur toute la plage d'activité, les agents de ces organisations travaillent en horaires décalés. Il convient alors de planifier pour chacun les horaires de travail et les jours de repos.

Les recherches sur la planification d'horaires portent sur trois types de problèmes comme les présentent Morris et Showalter (1983) ou Jarrah *et al.* (1994) : la construction de vacations (*shift scheduling problem*), le positionnement des jours de repos (*days-off scheduling problem*) et la construction de grilles de travail (*tour scheduling problem*), éventuellement cycliques (*rosters*).

Le problème de construction de vacations consiste à déterminer, pour une journée de travail, les horaires à effectuer par les agents pour couvrir la charge. On distingue les charges décrites par une liste de tâches insécables, comme celles des chauffeurs de bus ou des personnels navigants (cf. Rubin 1973; Rousseau et Lessard 1983; Martello et Toth 1986; Lavoie *et al.* 1988; Desrochers et Soumis 1989; Blais *et al.* 1990) et les charges constituées de tâches courtes ou sécables pouvant être traduites en courbe de charge, expression du nombre d'agents nécessaires par période de temps (cf. Segal 1974; Henderson et Berry 1976; Buffa *et al.* 1976; Keith 1979; Jacquet-Lagrèze et Meziani 1988). Dans le cas où les jours de repos sont déterminés de façon fixe, par exemple lorsque l'organisation ne fonctionne que cinq jours par semaine, la construction de vacations constitue l'unique problème de planification.

A l'inverse, si les horaires de travail des agents sont déterminés pour chaque jour mais que l'activité s'étend au-delà de la période hebdomadaire de travail, il convient de gérer le positionnement des repos (cf. Baker 1974; Brownel et Lowerre 1976; Baker et Magazine 1977; Bartholdi et Ratliff 1978). Enfin, lorsqu'il s'agit simultanément de construire un programme de travail quotidien et de déterminer le positionnement des repos, on se trouve confronté au problème de construction de grilles de travail (cf. Bechtold *et al.* 1991; Easton et Rossin 1991; Brusco et Jacobs 1993). C'est alors la combinaison d'un ensemble de grilles hebdomadaires individuelles qui couvre la charge périodique de période une semaine.

Différentes procédures heuristiques ont été proposées dans la littérature pour résoudre le problème de construction de grilles. Suivant Easton et Rossin (1991), on distingue trois catégories d'approches. Les premières s'appuient sur la génération et l'évaluation de toutes les grilles hebdomadaires admissibles pour se ramener à un problème de couverture d'ensembles. Lorsque la

solution du problème relâché n'est pas entière, des procedures heuristiques d'arrondi sont proposées (cf. Miller *et al.* 1976; Smith 1976; Keith 1979; Morris et Showalter 1983). Les secondes approches formulent le problème avec un sous-ensemble de grilles hebdomadaires admissibles et les choisissent une à une jusqu'à ce qu'un test d'arrêt soit vérifié (cf. Henderson et Berry 1976; Mabert et Watts 1982). Enfin, les procédures en deux phases, suggérées par Baker (1976), évitent la génération préalable de toutes les grilles hebdomadaires admissibles. Le problème de construction de grilles est ici décomposé: dans une première étape, on résout soit le problème de positionnement des repos (cf. Bailey (1985)), soit le problème de construction de vacations (cf. Buffa *et al.* (1976); Bechtold et Showalter (1987)), puis on utilise les résultats comme entrées du problème résiduel. Dans le cas où des listes de vacations sont issues de la première phase, la seconde phase consiste à enchaîner les vacations et à insérer des repos pour construire directement une grille unique et complète sur plusieurs semaines.

Nous nous inscrivons ici dans ce dernier type d'approche et nous traitons la phase d'enchaînement des vacations et des repos pour construire une grille de travail cyclique, appelée cycle. Un cycle, programme de travail sur plusieurs semaines, est donc construit directement en affectant une vacation ou un repos à chaque jour de manière à respecter une série de règles et à placer toutes les vacations. Pour simplifier ce problème, les vacations sont groupées et caractérisées par des étiquettes. Cette démarche permet de limiter la combinatoire de cycles admissibles tout en conservant un niveau de précision suffisant pour exprimer toutes les contraintes.

Nous décrivons d'abord le problème et les hypothèses permettant de nous ramener à un problème d'affectation ainsi que l'ensemble des règles de construction à respecter (section 2). Nous proposons ensuite un modèle d'affectation dont nous présentons deux variantes selon que le problème est résolu par la programmation linéaire en variables 0/1 (PL) ou par la programmation par contraintes (PPC) (section 3). Nous comparons enfin les résultats des deux approches en termes de cycles construits et de temps de calcul (section 4).

2 Description du problème

2.1 Quelques concepts et hypothèses

Une vacation désigne un horaire de travail; elle est caractérisée par une heure de début et une heure de fin et représente la présence d'un agent sur une journée donnée. En entrée du problème de construction de cycle, on dispose de listes de vacations à effectuer pour couvrir la charge. La démarche simplificatrice largement proposée dans la littérature et que nous reprenons, consiste à grouper les vacations de chaque jour selon une typologie formant une partition. Ici, la typologie utilisée fait référence à l'heure de début des

vacations. On définit ainsi une **table de besoins** exprimant le nombre de vacations de chaque type nécessaires chaque jour (cf. exemple table 1).

Vacations	Lun	Mar	Mer	Jeu	Ven	Sam	Dim
Matin	5	5	4	5	6	5	7
Soir	1	1	2	2	2	3	1
Nuit	1	1	1	1	1	1	1

TAB. 1 - *Exemple d'une table de besoins.*

Un cycle correspond au planning de travail d'un ensemble d'agents qui effectuent un roulement sur leur programme hebdomadaire. On le représente par un tableau où les lignes correspondent aux semaines et les colonnes aux jours. Considérons n agents en roulement sur un même cycle de longueur n semaines. Un agent affecté à la ligne l pendant la semaine courante sera affecté, la semaine suivante, à la ligne $l+1$ si $l < n$ et à la ligne 1 si $l = n$. **Un cycle étiqueté** est un cycle où les vacations sont remplacées par des étiquettes. La table 2 correspond au programme de travail de douze agents pendant douze semaines et couvre la table de besoins donnée ci-dessus (on lira M : Matin, S : Soir, N : Nuit, R : Repos).

Semaine	Lun	Mar	Mer	Jeu	Ven	Sam	Dim
1	R	R	M	M	M	S	N
2	R	R	R	M	M	M	M
3	M	R	R	M	M	M	M
4	N	R	R	R	M	M	M
5	M	M	M	R	R	R	M
6	M	N	N	N	R	R	M
7	M	S	S	S	S	R	R
8	R	M	M	M	N	N	R
9	R	M	S	S	S	S	R
10	R	M	M	R	R	M	M
11	M	M	R	R	M	S	S
12	S	R	R	M	M	M	M

TAB. 2 - *Exemple d'un cycle associé à la table de besoins 1.*

Une séquence est un enchaînement de vacations et de repos. Elle commence par une vacation et se termine à la fin de la première série de repos. Elle peut être de longueur variable. **Une séquence simplifiée** est un nombre de jours de travail suivi d'un nombre de jours de repos, par exemple (6T/3R).

On notera que la complexité du problème de construction de cycles étiquetés augmente avec la finesse de la typologie. Si toutes les vacations sont groupées sous la seule étiquette « travail » se distinguant seulement des repos, on retrouve le problème du positionnement des repos. A l'inverse, s'il

existe autant d'étiquettes que de vacations, on retrouve le problème initial de construction de grilles. Nous considérons ici un degré de précision intermédiaire où les vacations sont groupées selon trois types : Matin, Soir, Nuit.

Burns et Koop (1987) résolvent un problème analogue de construction de cycles étiquetés. Ils supposent un *niveau de charge constant* du lundi au vendredi et, éventuellement distinct mais toujours constant, les samedi et dimanche. Ils s'imposent de couvrir *au moins* cette charge. Leur démarche repose sur la combinaison de « modules », enchaînements pré-définis de vacations étiquetées et de repos sur plusieurs semaines. Du fait de ces modules, la variété des cycles pouvant être construits est limitée. De plus, l'hypothèse faite sur la charge est très restrictive.

Nos hypothèses sont différentes. Si nous considérons également une charge périodique définie sur une semaine type, toutes les variations de la charge au sein de la semaine type sont néanmoins autorisées. Par ailleurs, la couverture de charge telle que la formulent Burns et Koop (« *au moins* ») autorise le placement dans le cycle de vacations hors table de besoins. Nous imposons à l'inverse que seules les vacations de la table de besoins soient placées dans le cycle. Nous nous contraignons donc à une égalité entre le nombre de vacations nécessaires, donné dans la table de besoins, et le nombre de vacations effectuées dans le cycle. Ceci implique que les jours du cycle non affectés de vacations soient des repos. Ainsi, on calcule le nombre de repos à placer chaque jour, c'est-à-dire dans chaque colonne, par la différence entre la longueur du cycle, soit le nombre de lignes, et la somme des vacations du jour[1]. On sait donc exactement ce qu'il faut placer dans chaque colonne et il ne reste qu'à déterminer l'ordre de placement. Le problème se ramène donc à un problème d'affectation de l'ensemble des étiquettes contenues dans la table de besoins tout en respectant des contraintes d'enchaînement.

2.2 Les contraintes de construction de cycles

Les contraintes de construction de cycles sont de deux natures. D'une part, la réglementation du travail impose des contraintes à respecter strictement, auxquelles s'ajoutent des contraintes de cohérence éventuellement liées à la problématique d'analyse multicritère (cf. section 2.3) : ce sont les *contraintes rigides*, toujours présentes dans le modèle. D'autre part, un certain nombre de règles issues de conventions d'entreprise sont à respecter de manière plus ou moins souple et donnent lieu à des contraintes dites *flexibles*. Un cycle sera admissible même s'il ne respecte pas ces contraintes mais il sera meilleur, selon un critère social, s'il les respecte. Les contraintes flexibles peuvent être activées ou non. Activées, elles sont donc conçues, non pas de façon à être respectées « absolument » mais de façon à être respectées

1. Le calcul du nombre d'agents nécessaires pour couvrir la table de besoins n'est pas traité ici. On se réfèrera par exemple aux travaux de Baker et Magazine (1977), Burns et Carter (1985) ou Burns et Koop (1987). L'effectif supposé connu, on déduit la longueur du cycle à construire (le nombre d'agents devant être un multiple de la longueur).

« au mieux ». Rappelons qu'un mode classique de résolution de problèmes à contraintes flexibles a été proposé par Charnes et Cooper (1977) à travers la technique de type « Goal Programming ». Nous utiliserons cette approche dans la résolution par programmation linéaire présentée en section 3.2.

Les contraintes rigides

- contraintes de cohérence :
  1. Une vacation ou un repos par jour.
  2. Couverture exacte de la table des besoins (liée à l'hypothèse de qualité de service parfaite présentée en section 2.3).
- contraintes légales :
  1. Le nombre de semaines dans un cycle est limité à 12.
  2. Le nombre de vacations successives est limité à 6.
  3. La durée de repos entre deux vacations doit être supérieure à 11 heures. Dans le cas étiqueté, cela se traduit par l'interdiction d'enchaîner certains types de vacation. En particulier :
     - il est interdit d'enchaîner une vacation du matin après une vacation du soir ;
     - il est interdit d'enchaîner une vacation du matin ou du soir après une vacation de nuit.

Les contraintes flexibles

Les contraintes d'us et coutumes internes à l'entreprise sont paramétrables selon la classe de personnel à laquelle s'applique le cycle et à respecter plus ou moins strictement. Notons qu'un cycle correspond à une seule classe de personnel. Pour chaque classe, on définit les paramètres suivants :

1. les nombres minimum et maximum de jours de repos successifs (ceci permet par exemple d'éviter les repos isolés ou d'interdire les semaines complètes de repos) ;
2. la séquence simplifiée cible, c'est-à-dire un rythme de travail (nombre de jours travaillés-nombre de repos).
3. le nombre maximum de semaines sans repos le jour j ou le week-end : contrainte de répartition des repos.

2.3 Les critères

Comme le rappellent Easton et Rossin (1991), les problèmes de planification de personnel font intervenir trois catégories de critères conflictuels.

1. un critère de qualité de service correspondant à la couverture de la table des besoins ;

2. un critère de coût correspondant aux frais de personnel engendrés par la grille construite ;

3. un critère social correspondant à la satisfaction des agents vis à vis de leur programme de travail : celle-ci s'exprime soit par la prise en compte de préférences individuelles, soit par une série de règles de « confort » correspondant aux contraintes flexibles de la section précédente. Plus une grille respecte ces règles, mieux elle se place sur le critère social.

Easton et Rossin suggèrent de classifier les méthodes analytiques de planification de personnel selon la priorité accordée à chacun de ces critères. Certains modèles, comme le modèle général de planification de personnel proposé par Glover et McMillan (1986), visent à construire des grilles qui assurent une qualité de service convenable tout en respectant les desiderata individuels. D'autres, comme les modèles classiques de construction de grilles (cf. Bailey (1985); Bechtold et Showalter (1987)), cherchent une planification de coût minimum qui garantisse une bonne qualité de service mais ne respecte que la règlementation du travail et non les préférences individuelles.

Conformes à toutes les situations rencontrées dans la littérature, nous nous plaçons ici dans une hypothèse de qualité de service maximale qui se traduit par la couverture de la table de besoins. En effet, imposer de placer dans le cycle toutes les vacations de la table signifie rester au même niveau de qualité de service que celui donné en entrée. D'autre part, nous avons retenu l'hypothèse de couverture *exacte* de la table. Or ne pas ajouter de vacations par rapport au nombre nécessaire revient à ne pas intervenir non plus sur le critère de coût. Ainsi, dans le cadre de nos hypothèses, le seul critère intervenant dans la construction de cycles est la satisfaction sociale. Ce critère regroupe des aspects légaux pris en compte par des contraintes rigides et des aspects d'us et coutumes pris en compte par des contraintes flexibles. Le niveau de satisfaction sociale, nécessairement au-delà d'un seuil fixé par les contraintes rigides, se mesure donc par le niveau de respect des contraintes flexibles. Considérer plusieurs contraintes flexibles en parallèle revient à agréger plusieurs objectifs.

3 Un modèle d'affectation pour la construction de cycles

3.1 Les données du problème

3.1.1 Les indices

$\mathcal{I}$ = Ensemble des indices représentant les semaines du cycle ; $i \in \mathcal{I} = \{1, .., i_{\max}\}$.

$\mathcal{J}$ = Ensemble des indices représentant le jour de la semaine ; $j \in \mathcal{J} = \{1, .., j_{\max} = 7\}$.

$\mathcal{T}$ = Ensemble des indices représentant les jours du cycle ; $t \in \mathcal{T} = \{1, .., i_{\max} \times j_{\max}\}$. Notons que $t = 7i + j$.

$\mathcal{K}$ = Ensemble des indices correspondant aux étiquettes (vacations et repos) ; $k \in \mathcal{K} = \{1, .., k_{\max}\}$. On aura par exemple :
$k = 1$ correspond à une vacation de type Matin ;
$k = 2$ correspond à une vacation de type Soir ;
$k = 3$ correspond à une vacation de type Nuit ;
$k = 4$ correspond à un Repos ;

$\mathcal{H}$ = Ensemble des types de contraintes flexibles h.

3.1.2 Données et paramètres

b_{jk} représente la charge du jour j en vacations de type k donnée dans la table des besoins ;

(a, b) représente le couple (nombre de jours de travail, nombre de jours de repos) dans la séquence cible ;

n_j représente le nombre maximum de semaines sans repos le jour j.

3.2 L'approche PL

3.2.1 Les variables de décision

$$x_{ijk} = \begin{cases} 1 & \text{si le jour } j \text{ semaine } i \text{ est couvert par une vacation de type } k \\ 0 & \text{sinon.} \end{cases}$$

Afin de simplifier l'écriture du modèle, on utilisera les variables auxiliaires suivantes :

$$\widetilde{x}_t = \begin{cases} 1 & \text{si le jour } t \text{ (jour } j \text{ de la semaine } i) \text{ est un repos} \\ 0 & \text{sinon.} \end{cases}$$

Par exemple, $\widetilde{x}_{12} = 1$ correspond à $x_{2,5,4} = 1$, c'est-à-dire que le douzième jour de la grille (cinquième jour de la deuxième semaine) est un repos.

3.2.2 Les contraintes rigides

1. ***Affectation d'une seule vacation à chaque jour :*** il est impossible d'effectuer deux vacations dans la même journée. Cela revient à dire qu'on place une vacation et une seule chaque jour de chaque semaine :

$$\sum_{k \in \mathcal{K}} x_{ijk} = 1 \qquad \forall i \in \mathcal{I},\ \forall j \in \mathcal{J} \tag{1}$$

2. ***Couverture des besoins :*** nous l'avons vu, l'effectif présent doit être strictement égal à la charge, c'est-à-dire que le nombre de vacations de chaque type sur un jour doit être strictement égal à la valeur donnée dans la table des besoins, soit :

$$\sum_{i \in \mathcal{I}} x_{ijk} = b_{jk} \qquad \forall j \in \mathcal{J},\ \forall k \in \mathcal{K} \tag{2}$$

3. ***Au maximum a jours successifs de travail :*** signifie que, sur $(a+1)$ jours consécutifs, on place au moins un repos. Cette contrainte est glissante pour tous les enchaînements de $(a+1)$ jours, c'est-à-dire qu'elle s'écrit pour chaque jour du cycle. D'après la législation du travail, on a $a = 6$ d'où :

$$\sum_{m=0}^{6} \widetilde{x}_{t+m} \geq 1 \qquad \forall t \in \mathcal{T} \tag{3}$$

4. ***Certains enchaînements sont interdits :*** par exemple, on ne peut faire suivre un soir d'un matin,... Si k_1 suivi de k_2 est interdit, cela signifie que, sur deux jours successifs, on ne peut pas avoir le premier jour une vacation k_1 et le second une vacation k_2, ce qui s'écrit :

$$x_{i,j,k_1} + x_{i,j+1,k_2} \leq 1 \qquad \forall i \in \mathcal{I},\ \forall j \in \mathcal{J} \tag{4}$$

   Remarque : Lors de l'écriture détaillée du modèle, on s'assure du bouclage, c'est-à-dire que toutes les contraintes concernant un enchaînement de vacations soient également imposées sur l'enchaînement « fin de cycle - début de cycle ».

3.2.3 Les contraintes flexibles

Afin de formaliser les contraintes flexibles, on introduit des *variables d'écart.* Les contraintes flexibles étant de nature binaire (respectée ou violée), les variables d'écart seront également binaires et elles prendront la valeur 1 chaque fois qu'une contrainte sera violée dans le cycle. Elles permettent ainsi de compter le nombre de fois où un certain type de contrainte est violé. L'objectif du modèle sera alors de minimiser les valeurs de ces variables d'écart. Soit :

$$d_t^h = \begin{cases} 1 & \text{si la contrainte de type } h \text{ n'est pas respectée le jour } t \\ 0 & \text{sinon.} \end{cases}$$

$\sum_{t\in\mathcal{T}} d_t^h$ correspond donc au nombre de fois dans la grille où la contrainte de type h n'est pas respectée.

1. ***Repos isolés :***

   Ecrivons d'abord la contrainte de **repos isolé interdit** : si on a un repos le jour t, on impose d'avoir un repos le jour $t-1$ et/ou le jour $t+1$, ce qui s' écrit :

   $$\widetilde{x}_{t-1} - \widetilde{x}_t + \widetilde{x}_{t+1} \geq 0 \qquad \forall t \in \mathcal{T} \tag{5}$$

   On constate en effet que :

   - $\widetilde{x}_t = 1$ impose à l'une au moins des deux variables $\widetilde{x}_{t-1}$ et $\widetilde{x}_{t+1}$ de prendre la valeur 1 ;
   - $\widetilde{x}_t = 0$ n'impose aucune restriction sur $\widetilde{x}_{t-1}$ et $\widetilde{x}_{t+1}$.

   Sous cette forme, la contrainte est rigide. Pour la rendre contrainte flexible, on introduit les variables d_t^1. La contrainte devient :

   $$\widetilde{x}_{t-1} - \widetilde{x}_t + \widetilde{x}_{t+1} + d_t^1 \geq 0 \qquad \forall t \in \mathcal{T} \tag{6}$$

   Cela signifie que si le jour t est un repos isolé, d_t^1 prendra la valeur 1, et, minimiser la somme des d_t^1, force d_t^1 à valoir 0 sinon. Compter les repos isolés revient donc à compter le nombre de fois où d_t^1 vaut 1. Minimiser les repos isolés revient donc à minimiser $\sum_{t\in\mathcal{T}} d_t^1$.

2. ***Séquence simplifiée cible :***

   On peut vouloir imposer un rythme de travail sur tout le cycle. Ce rythme se définit par une certaine séquence simplifiée, exprimée en nombre de jours travaillés / nombre de jours de repos. Si l'on impose par exemple des séquences de type 6 Travail / 3 Repos, on construit un cycle de longueur 9 semaines extrêmement régulier et offrant un effectif présent chaque jour constant (cf. table 3).

   Afin de laisser une certaine marge de manœuvre dans la construction du cycle, nous n'imposons pas à toutes les séquences d'être de la forme a Travail/b Repos, ce qui reviendrait à imposer à la fois :

   - a jours de travail successifs et
   - après a jours de travail, b repos.

Semaine	Lun	Mar	Mer	Jeu	Ven	Sam	Dim
1	T	T	T	T	T	T	R
2	R	R	T	T	T	T	T
3	T	R	R	R	T	T	T
4	T	T	T	R	R	R	T
5	T	T	T	T	T	R	R
6	R	T	T	T	T	T	T
7	R	R	R	T	T	T	T
8	T	T	R	R	R	T	T
9	T	T	T	T	R	R	R
Total	6T	6T	6T	6T	6T	6T	6T

TAB. 3 - *Le cycle 6T/3R*

Nous n'intégrons en fait que ce second aspect qui se formule de manière linéaire :

$$\sum_{n=0}^{b-1} \widetilde{x}_{t+n} + b \times \sum_{n=1}^{a} \widetilde{x}_{t-n} \geq b \qquad \forall t \in \mathcal{T} \tag{7}$$

En effet :

- si $\sum_{n=1}^{a} \widetilde{x}_{t-n} = 0$,
  c'est-à-dire que les jours $(t-1, t-2, ..., t-a)$ sont travaillés,
  cela impose $\sum_{n=0}^{b-1} \widetilde{x}_{t+n} = b$,
  c'est à dire que $\widetilde{x}_t = \widetilde{x}_{t+1} = ... = \widetilde{x}_{t+b} = 1$,
  ce qui signifie que les jours $(t, t+1, ..., t+b)$ sont des repos.
- si $\sum_{n=1}^{a} \widetilde{x}_{t-n} \geq 0$, aucune restriction sur $\widetilde{x}_t$, $\widetilde{x}_{t+1}$, ..., $\widetilde{x}_{t+b}$.

Formulée ainsi, cette contrainte est rigide. Si on l'active, toutes les séries de a jours travaillées seront suivies de b repos. On rend cette contrainte flexible en ajoutant une variable d'écart d_t^2 :

$$\sum_{n=0}^{b-1} \widetilde{x}_{t+n} + b \sum_{n=1}^{a} \widetilde{x}_{t-n} + b d_t^2 \geq b \qquad \forall t \in \mathcal{T} \tag{8}$$

d_t^2 vaudra 1 si a jours de travail sont suivis de moins de b repos, 0 sinon. Minimiser le nombre de séquences de a travail suivies de moins de b repos revient à minimiser la somme des d_t^2.

3. ***Répartition des repos :***

Afin d'équilibrer la répartition des repos sur l'ensemble de la grille, on impose, pour chaque jour de la semaine, une périodicité maximale des repos. Si, pour le jour j, on veut avoir au moins un repos toutes les n_j

semaines cela revient à dire que sur $(n_j + 1)$ semaines, successives, on doit avoir au moins un repos le jour j :

$$\sum_{m=0}^{n_j} \widetilde{x}_{t+7m} \geq 1 \qquad \forall t \in \mathcal{T} \tag{9}$$

Cette contrainte est rigide. On jouera sur les valeurs des n_j pour la rendre plus ou moins restrictive. On peut également la transformer en contrainte flexible en ajoutant d_t^3.

$$\sum_{m=0}^{n_j} \widetilde{x}_{t+7m} + d_t^3 \geq 1 \qquad \forall t \in \mathcal{T} \tag{10}$$

d_t^3 prendra la valeur 1 chaque fois que l'on retrouve dans le cycle (n_j+1) semaines successives sans repos le jour j. La somme des d_t^3 correspond donc au nombre de fois où la contrainte n'est pas vérifiée.

Les contraintes flexibles décrites ci-dessus sont optionnelles c'est-à-dire que l'on choisit de les activer ou non. Chaque configuration donne alors lieu à un sous-modèle différent.

3.2.4 Les objectifs

Soit $\mathcal{H}$ l'ensemble des types de contraintes flexibles. L'objectif s'écrit :

$$\text{Min} \quad \sum_{h \in \mathcal{H}} \sum_{t \in \mathcal{T}} w_h d_t^h \tag{11}$$

Selon les valeurs des coefficients w_h, trois cas se présentent :

- Si $w_h = 1$ et $w_l = 0 \quad \forall l \neq h$, on ne considère qu'un seul type de contrainte flexible à la fois. La valeur de l'objectif correspond alors au nombre de fois dans le cycle où la contrainte de type h est violée.

- Si $w_h = 1 \quad \forall h \in \mathcal{H}$, on considère tous les types de contraintes flexibles à importance égale. La valeur de l'objectif représente ici le nombre de fois où une contrainte flexible de type quelconque est violée dans le cycle.

- Si les coefficients w_h prennent différentes valeurs selon h, cela différencie l'importance relative de non respect de chaque type de contraintes. La valeur de l'objectif correspond cette fois à une somme pondérée du nombre de violations de chaque type de contraintes. Un cas particulier se présente lorsque les coefficients w_h ont des valeurs trés différenciées traduisant ainsi une hiérarchie lexicographique des critères.

3.3 L'approche PPC

Nous proposons maintenant une approche du modèle de construction de cycles par la programmation par contraintes. La résolution par propagation de contraintes se prête tout à fait à ce type de problème où l'unique objectif est de satisfaire le maximum de contraintes. Cette approche a été développée sur le produit Ilog Solver, une bibliothèque de programmation par contraintes orientée objet. L'outil Ilog Solver a d'ailleurs prouvé son efficacité pour la résolution d'un problème similaire de construction de grilles de travail non cycliques pour la Banque Bruxelles Lambert (Jacques 1995).

3.3.1 Les variables de décision

Pour chaque jour j de chaque semaine i, on définit une variable x_{ij}. A chaque variable est associé un domaine de valeurs possibles comportant les différentes étiquettes autorisées. Si l'ensemble des étiquettes est (Matin/Soir/Nuit/Repos), chaque variable pourra prendre une et une seule de ces valeurs.

Chaque fois qu'une variable est instanciée en choisissant une valeur dans son ensemble de valeurs possibles, les contraintes se propagent, entraînant l'élimination de certaines valeurs dans les ensembles de certaines autres variables non encore instanciées. Si cette propagation n'a pas généré d'ensemble vide, on l'accepte ; sinon, on efface la dernière instanciation (*backtracking*) et on affecte une nouvelle valeur à la variable précédente. L'ordre dans lequel on choisit les variables à instancier ainsi que celui dans lequel on propose les valeurs dans le domaine définissent la stratégie de résolution et sont déterminants pour la vitesse de calcul.

3.3.2 Les contraintes rigides

On retrouve dans l'approche par PPC globalement les mêmes contraintes que dans l'approche PL. On note cependant quelques nuances liées directement aux méthodes. En particulier, certaines contraintes présentes en PL ne sont plus nécessaires en PPC ; inversement, des contraintes supplémentaires sont ajoutées en PPC pour faciliter la résolution.

1. ***Une seule vacation par jour***

   Cette contrainte est vérifiée par définition des variables dans la mesure où chaque variable ne peut prendre qu'une seule valeur de son domaine.

2. ***Couverture de la table des besoins***

   On utilise une contrainte pré-définie par Ilog Solver permettant de compter des variables (**CtCount**). Pour tout jour j et pour toute étiquette k, la somme sur i des vacations de type k doit être égale à la charge b_{jk}. Il y a donc $j_{\max} \times k_{\max}$ contraintes de ce type.

3. ***Au maximum a vacations travaillées successives***

   On utilise une contrainte **CtCount** pour chaque jour de la grille, y compris les derniers qui bouclent avec le début de la grille. La somme des vacations travaillées sur $a+1$ jours successifs doit être inférieure ou égale à a. Il y a $i_{\max} \times j_{\max}$ contraintes de ce type.

4. ***Enchaînements interdits***

   On définit deux contraintes sur mesure (par opposition aux contraintes pré-définies par Ilog Solver). La première correspond à celle de l'approche PL, la seconde est une contrainte supplémentaire d'anticipation servant à faciliter la résolution.

   - **Enchaînement Interdit** : A chaque instanciation d'une variable, si la variable précédente et/ou suivante n'est pas encore instanciée, on supprime de son domaine les valeurs interdites.
   - **Anticipe Enchaînement** : Cette contrainte permet de propager les interdictions. A chaque modification du domaine d'une variable -et plus seulement à chaque instanciation-, cette contrainte supprime, des variables précédentes et suivantes, les valeurs nouvellement interdites.

     Illustrons le comportement de cette contrainte par un exemple. Nous considérons la séquence interdite Soir-Matin. Un Soir le jour j ne pourra donc être suivi que d'un autre Soir, d'une Nuit ou d'un Repos le jour $j+1$. Supposons qu'il reste un Soir à placer dans la colonne du jour j. Si tous les Soir, Nuit, Repos du jour $j+1$ ont déjà été placés et qu'ils sont tous précédés de variables instanciées, le Soir du jour j, où qu'il se place dans la colonne, ne pourra être suivi d'aucune étiquette. Si le problème était initialement faisable, cela signifie que de mauvais choix ont été effectués et il faut revenir en arrière.

     L'objectif de cette contrainte est donc d'anticiper les mauvais choix en supprimant dès que possible des domaines les valeurs qui ne sont pas directement interdites mais qui conduiraient à des situations sans solution réalisable. Elle permet ainsi d'éviter de nombreux backtracking et donc d'améliorer sensiblement les temps de calcul.

     Chacune de ces contraintes Anticipe Enchaînement ne gère qu'un seul des enchaînements interdits à la fois. Elle serait sans doute plus efficace si elle prenait en compte toutes les séquences interdites en même temps mais cette extention n'est pas proposée ici.

3.3.3 Les contraintes flexibles

Comme nous l'avons vu dans la modélisation PL, les contraintes de repos isolé, de séquence cible et de répartition des repos sont flexibles : on s'auto-

rise à ne pas les respecter mais il faut minimiser le nombre de violations. En PPC, on ajoute une contrainte dite *d'objectif* pour limiter le nombre de fois où les contraintes flexibles sont violées. Si on ne considère qu'un type de contrainte flexible à la fois, la contrainte d'objectif ne comptera que les violations de ce type. Inversement, la contrainte d'objectif pourra intégrer, avec des importances relatives égales ou différentes, plusieurs types de contraintes flexibles. Une procédure itérative permet ensuite d'améliorer une solution en diminuant progressivement la borne de la contrainte d'objectif.

1. ***Minimum de jours consécutifs de travail a ou de repos b :***

   Cette contrainte modélise à la fois l'interdiction de repos isolé et la contrainte de séquence cible.

   A chaque instanciation d'une variable par une vacation «travail», cette contrainte vérifie que, dans les domaines du jour précédent ou du jour suivant, à condition qu'ils ne soient pas encore instanciés, une étiquette «travail» (M/S/N) est disponible et ceci dans la limite de a. De même pour les repos dans la limite de b. Avec $b = 2$, on retrouve l'interdiction de repos isolés. Avec $a = 6$ et $b = 3$, on impose d'avoir au moins 6 jours de travail successifs, contrainte qui devient « exactement » 6 jours de travail successifs du fait de la limite légale à 6 et on aura au moins 3 repos qui suivent. Cette contrainte rappelle donc la notion de séquence cible définie dans l'approche PL.

2. ***Répartition des repos :***

   Comme en PL, on peut imposer un nombre maximum de semaines consécutives sans repos le jour j (n_j). On utilise une contrainte **CtCount**. La somme des repos le jour j sur (n_j+1) semaines consécutives doit être au moins égale à 1. Pour chaque n_j, il y a $i_{\max}$ contraintes.

3.3.4 La stratégie de résolution

L'ordre dans lequel on instancie les variables ainsi que celui de présentation des valeurs dans les domaines sont déterminants pour l'efficacité de la résolution. La construction de la stratégie est donc une étape essentielle dans la modélisation d'un problème par la PPC. Après une série de tests, nous avons défini la stratégie suivante :

1. ***choix des variables :*** Ici, nous avons opté pour la fonction standard de Ilog Solver **CtGenerate** qui prend en compte les variables non instanciées dans l'ordre où elles se présentent, c'est-à-dire pour nous un ordre chronologique.

2. ***choix de l'ordre du domaine :*** de multiples expérimentations ont permis de définir un ordre de présentation des variables dans le domaine qui semble donner les meilleurs résultats : « S, M, N, R ». Ce choix

empirique est toutefois trés lié à la table de besoins et pourrait s'avérer inefficace pour une table de besoins sensiblement différente.

3. ***choix de la valeur :***

   Lorsque le domaine d'une variable comporte au moins deux valeurs, on recourt à l'heuristique suivante tendant à privilégier le placement d'étiquettes « Travail » :

   (a) Si l'étiquette Repos est présente dans le domaine, la supprimer.[2]

   (b) Si la variable précédente est instanciée en Repos,

      i. si possible choisir une vacation différente de celle ayant débuté la séquence précédente, dans l'ordre du domaine ;

      ii. sinon, accepter la même étiquette ;

   (c) Sinon

      i. si possible choisir la même étiquette que celle affectée à la variable précédente ;

      ii. sinon, prendre la première valeur disponible selon l'ordre du domaine.

4 Comparaison des deux approches

4.1 Modèles et Résolution

4.1.1 Les variables

Les approches PL et PPC sont basées sur le même modèle mais la nature même des techniques de résolution entraîne quelques disparités, d'abord au niveau des variables de décision. Dans l'approche PL, on définit une variable par jour et par étiquette alors que dans l'approche PPC, on ne définit qu'une variable par jour, les différentes étiquettes étant représentées par le domaine.

4.1.2 Les contraintes

Notons d'abord que l'écriture des contraintes ne pose pas de difficulté majeure dans chacune des deux approches, même si les contraintes logiques de type « si...alors » s'expriment plus naturellement en PPC. Il suffit toutefois d'un certain niveau d'expertise pour formaliser ces contraintes également en PL. Il existe cependant quelques nuances entre les contraintes des deux modèles.

2. Notons qu'en cas d'indétermination, il reste au moins une vacation de type « travail » dans le domaine. Les Repos ne sont affectés que si la propagation des contraintes a réduit un domaine à cette seule étiquette.

D'une part, du fait de la définition des variables, la contrainte de cohérence imposant en PL une seule étiquette par jour n'est pas nécessaire en PPC.

D'autre part, on utilise en PPC des contraintes d'anticipation afin d'aider à la résolution. En effet, la construction visiblement séquentielle d'un cycle en PPC suggère l'ajout de contraintes permettant de guider vers les bons choix en éliminant de mauvais. Cette démarche n'est pas classique en PL où l'ajout de contraintes supplémentaires nuit généralement à la résolution.

4.1.3 L'optimisation

La PPC est une approche qui vise davantage à déterminer des solutions réalisables qu'à optimiser un objectif. Il est possible de suppléer à cette faiblesse en introduisant les contraintes d'objectif et un processus itératif autour de la stratégie de résolution. Cette démarche est toutefois délicate à mettre en œuvre. D'une part, il faut fixer les bornes de ces contraintes. Or, plus la borne est loin de la valeur optimale, plus il faudra d'itérations pour l'obtenir. Inversement, si la borne est inférieure à la valeur optimale de l'objectif à minimiser, il s'avèrera très couteux en temps de calcul de montrer qu'il n'existe pas de solution avec cette valeur. D'autre part, il n'y a pas d'interprétation directe des bornes en particulier lorsque plusieurs contraintes flexibles interviennent en parallèle et sont d'importance relative différente.

4.1.4 La stratégie

Enfin, les stratégies de résolution sont différentes. En PL, on utilise une procédure classique de séparation et évaluation pré-définie par le solver utilisé (ici OSL). En PPC, il faut construire une stratégie propre au problème traité, la plus efficace possible. Or cette étape peut être longue et déroutante dans la mesure où les réactions du solver sont parfois contre-intuitives. Il faut donc effectuer de nombreux tests sur différentes stratégies avant d'en déterminer une qui paraît fournir les meilleurs temps de réponse.

Ces quelques remarques sur les différences de modélisation entre les deux approches étant faites, comparons maintenant les résultats des tests en termes de temps de calcul et de valeur de la solution optimale obtenue.

4.2 Les résultats

Les deux approches PL et PPC ont été testées pour construire une grille de longueur 12 semaines à partir de deux tables de besoins (charge 1 correspond à celle donnée dans la table 1, charge 2 est légèrement différente).

Rappelons les contraintes rigides respectées dans la résolution:

- une seule vacation par jour;
- couverture exacte de la table de besoins;

- maximum 6 jours de travail successifs ;

- enchaînements interdits : Soir-Matin ; Nuit-Matin ; Nuit-Soir.

D'autres contraintes sont éventuellement ajoutées selon les tests. Nous nous sommes limités dans ces expérimentations à ne considérer qu'une seule contrainte flexible : celle concernant les repos isolés. L'objectif représente donc le nombre de repos isolés placés dans le cycle soit la somme des variables d_i^1 (cf. section 3.2.3). Les résultats de ces tests sont donnés dans le tableau 4.

Contraintes Supplémentaires	**Rés. PL**			**Rés. PPC**	
	Obj	CPU	nb sol	Obj	CPU
Charge 1			*x/y*		
	0	270"	4/10	0	0,2"
$n_j = 12$	0	156"	4/10	0	0,1"
$n_j = 8$	0	230"	3/10	0	6,2"
$n_j = 7$	0	154"	4/10	0	0,4"
$n_j = 6$	0	179"	5/10	0	0,2"
$n_3 = 4$ et autres $n_j = 12$	0	1606"	8/10	0	0,3"
$n_j = 8$ et M-N interdit	0	565"	5/10	0	7,2"
SC 6/3	0	3112"	7/10	0	97"
SC 6/3 et M-N interdit	0	832"	8/10	0	14"
SC 6/3, M-N interdit et $n_j = 8$	4	1128"	7/7	0	171"
SC 6/3, M-N interdit, $n_6, n_7 = 6$, $n_j = 12$	–	>3600"	–	0	2506"
Charge 2					
$n_j = 12$	0	454"	4/10	0	12"
$n_j = 8$	0	34"	1/10	0	1"

SC 6/3 = on ajoute la contrainte sur les séquences cibles ;
$n_j = 8$ = on impose un repos le jour j au maximum toutes les 8 semaines ;
M-N interdit = on ajoute l'enchaînement interdit Matin-Nuit

TAB. 4 - *Comparaison des deux approches*

Les tests en PL ont été effectués sur Station Risc 6000 modèle 250, avec la librairie de résolution OSL. Nous avons utilisé les techniques de prétraitement disponibles dans OSL ainsi que les procédures de séparation et évaluation pour traiter les contraintes d'intégrité. L'exploration intégrale de l'arbre par séparation et évaluation pouvant être très coûteuse en temps de calcul, nous avons limité le nombre de solutions entières à explorer (correspond au y de la colonne «nb sol»). Lorsque le solver obtient une solution entière optimale, il s'arrête. Le nombre de solutions entières explorées (x de la même colonne) garantit donc l'optimalité de la solution obtenue s'il est strictement inférieur au nombre maximum autorisé (y). S'il est égal, il peut exister une meilleure

solution et il faudrait prolonger le test pour le vérifier. Les tests en PPC ont été effectués sur station Sun Sparc 2 avec la librairie d'Ilog Solver.

On observe, à la lecture du tableau 4, que l'objectif est égal à 0 dans tous les tests (à l'exception des deux derniers cas de la charge 1 où l'optimum n'est pas atteint). Cela signifie qu'aucun repos isolé n'a été placé et peut s'expliquer par la proportion satisfaisante du nombre de repos comparé au nombre total de vacations. Rappelons que le nombre de repos a été calculé selon les normes de l'entreprise (3 repos pour 9 vacations) et que les tables de besoins vérifient cette proportion. On peut également noter que l'ajout de contraintes supplémentaires n'a eu d'influence que sur les temps de calcul et non sur la valeur de l'objectif. Ceci est sans doute dû à la nature et au niveau de difficulté des contraintes ajoutées et ne devrait plus se vérifier dans des cas plus contraints.

Les résultats présentés dans le tableau 4 montrent des temps de calcul de 30 à 1000 fois plus rapides pour la PPC. Bien que l'utilisation de différents processeurs pour chaque approche fragilise cette comparaison, elle ne saurait expliquer un tel écart. On peut donc conclure à la nette supériorité de l'approche PPC selon un critère de rapidité d'exécution.

Il faut cependant relativiser cette conclusion, tant au niveau de la qualité des cycles que des temps de calcul.

D'une part, du fait de la séquentialité de la résolution en PPC, on observe des grilles dont la qualité se dégrade entre les premières semaines et les dernières. En effet, au début de la construction, les possibilités de choix sont nombreuses car les variables sont peu contraintes ; plus on avance dans la construction, plus les contraintes réduisent les domaines des variables et donc moins la grille est harmonieuse.

D'autre part, la contrainte d'objectif en PPC autorise un certain nombre de violations de la (ou les) contrainte(s) flexible(s), nombre décrémenté au cours des itérations successives. Or on constate que la PPC place systématiquement toutes les violations autorisées, même si une solution sans violation de cette (ou ces) contrainte(s) existe. Ceci signifie donc la construction de « mauvais » cycles lors des premières itérations.

Par ailleurs, les temps de calcul sont d'autant plus longs que la borne initiale de la contrainte d'objectif est loin de l'optimum puisque cela entraîne autant d'itérations. Or, dans le cadre de ces expérimentations où la valeur optimale de l'objectif est 0, si la borne de la contrainte d'objectif est fixée à 0 dès la première itération, il est normal que le calcul soit rapide. La PPC se trouve donc ici dans une situation particulièrement favorable, ce qui pourrait expliquer son efficacité.

Enfin, on pourrait s'intéresser à la question de "robustesse" des modèles présentés. Nous avons déjà évoqué la forte influence d'un resserrement de contraintes sur les temps de calcul, aussi bien avec l'approche PL que PPC. Il faudrait aussi étudier l'effet de la modélisation et l'insertion de *nouvelles contraintes* sur le comportement des deux approches. Selon Caseau *et al.*

(1995), certains types de contraintes pourraient être trés invalidants pour la PPC alors que la PL réagirait mieux. Il serait donc intéressant, dans des travaux futurs, de déterminer pour quels types de contraintes et dans quelles situations la PPC reste plus performante et quelles sont, au contraire, les contraintes « difficiles » pour la PPC et plus simples à intégrer et gérer en PL.

Remerciements

Nous tenons à remercier le service de Recherche Opérationnelle de la compagnie **Air France** *ainsi que la société de services* **Eurodécision** *avec lesquelles nous avons coopéré pour cette recherche et qui ont mis à notre disposition leur matériel informatique et outils de calculs. Les tests de PL ont été réalisés au sein d'Eurodécision et ceux de PPC chez Air France.*

Références

Bailey, J. (1985). Integrated days-off and shift personnel scheduling. *Computers and Industrial Engineering*, **9**(4), 395–404.

Baker, K. R. (1974). Scheduling a full-time workforce to meet cyclic staffing requirements. *Management Science*, **20**, 1561–1568.

Baker, K. R. (1976). Workforce allocation in cyclical scheduling problems: A survey. *Operations Research Quarterly*, **27**, 155–167.

Baker, K. R. et Magazine, M. J. (1977). Workforce scheduling with cyclic demands and days-off constraints. *Management Science*, **24**, 161–167.

Bartholdi, J. J. et Ratliff, H. D. (1978). Unnetworks with applications to idle time scheduling. *Management Science*, **24**(8), 850–858.

Bechtold, S. E. et Showalter, M. J. (1987). A methodology for labor scheduling in a service operating system. *Decision Sciences*, **18**(1), 89–107.

Bechtold, S. E., Brusco, M. J., et Schowalter, M. J. (1991). A comparative evaluation of labor tour scheduling methods. *Decision Sciences*, **22**, 683–699.

Blais, J., Lamont, J., et Rousseau, J. (1990). The Hastus vehicle and manpower scheduling systems at the Société de Transport de la Communauté Urbaine de Montréal. *Interfaces*, **20**, 26–42.

Brownel, W. S. et Lowerre, J. M. (1976). Scheduling of workforce required in continuous operations under alternative labor policies. *Management Science*, **22**, 597–605.

Brusco, M. J. et Jacobs, L. W. (1993). A simulated annealing approach to the cyclic staff-scheduling problem. *Naval Research Logistics*, **40**, 69–84.

Buffa, E. S., Cosgrove, M. J., et Luce, B. J. (1976). An integrated work shift scheduling system. *Decision Sciences*, **7**, 620–630.

Burns, R. N. et Carter, M. W. (1985). Work force size and single shift schedules with variable demands. *Management Science*, **31**(5), 599–607.

Burns, R. N. et Koop, G. J. (1987). A modular approach to optimal multiple-shift manpower scheduling. *Operations Research*, **35**(1), 100–110.

Caseau, Y., Guillo, P. Y., et Levenez, E. (1995). A deductive and object-oriented approach to a complex scheduling problem. *Journal of Intelligent Information Systems*, **4**, 149–166.

Charnes, A. et Cooper, W. W. (1977). Goal programming and multiple objective optimizations. *European Journal of Operational Research*, **1**, 39–54.

Desrochers, M. et Soumis, F. (1989). A column generation approach to the urban transit crew scheduling problem. *Transportation Science*, **23**(1), 1–13.

Easton, F. F. et Rossin, D. F. (1991). Equivalent alternate solutions for the tour scheduling problem. *Decision Sciences*, **22**, 985–1001.

Glover, F. et McMillan, C. (1986). The general employee scheduling problem. *Computers and Operations Research*, **13**(14), 563–593.

Henderson, W. B. et Berry, W. L. (1976). Heuristic methods for telephone operator shift scheduling: An experimental analysis. *Management Science*, **22**, 1372–1380.

Jacques, P. (1995). Solving a manpower planning problem using constraint programming. *Belgian Journal of Operations Research, Statistics and Computer Science*, **33**(4), 77–85.

Jacquet-Lagrèze, E. et Meziani, R. (1988). Linear programming and interactivity: a manpower scheduling DSS. In *IFORS 88*, volume Operational Research'87, pages 176–189.

Jarrah, A. I. Z., Bard, J. F., et deSilva, A. H. (1994). Solving large-scale tour scheduling problems. *Management Science*, **40**(9), 1124–1144.

Keith, E. (1979). Operator scheduling. *AIIE Transactions*, **11**, 37–41.

Lavoie, S., Minoux, M., et Odier, E. (1988). A new approach for crew pairing problems by column generation with application to air transportation. *European Journal of Operational Research*, **35**, 45–58.

Mabert, V. A. et Watts, C. A. (1982). A simulation analysis of tour-shift construction procedures. *Management Science*, **28**(5), 520–532.

Martello, S. et Toth, P. (1986). A heuristic approach to the bus driver scheduling problem. *European Journal of Operational Research*, **24**, 106–117.

Miller, H., Pierskalla, W., et Rath, G. (1976). Nurse scheduling using mathematical programming. *Operations Research*, **24**, 857–869.

Morris, J. G. et Showalter, M. J. (1983). Simple approaches to shift, days-off and tour scheduling problems. *Management Science*, **29**(8), 942–950.

Rousseau, J. M. et Lessard, R. (1983). Scheduling yearly modification of pilot/aircraft assignements in an airline. Working paper 471, Département d'informatique et de recherche opérationnelle, Université de Montréal.

Rubin, J. (1973). A technique for the solution of massive set covering problems, with applications to airline crew scheduling. *Transportation Sciences*, **7**, 34–48.

Segal, M. (1974). The operator-scheduling problem: A network-flow approach. *Operations Research*, **22**, 808–823.

Smith, L. (1976). The application of an interactive algorithm to develop cyclical rotational schedules for nursing personnel. *INFOR*, **14**, 53–70.