Optimisation des problèmes de plannings (Scheduling). Application de la méthode de génération de colonnes

by

Benoumelaz Farouk

Submitted to the Department de mathématique
in partial fulfillment of the requirements for the degree of

Magister

at the

UNIVERSITÉ KHENCHELA

16 Septembre 2012

© Jury

Pr. Melkemi Khaled	Université de Biskra	President
Dr. Djeffal Lakhdar	Université de Batna	Rapporteur
Dr. Mokhtari Zoheir	Université de Biskra	Examinateur
Dr. Ajroud Nacer	Université de Khanchela	Examinateur

Signature of Author .....
Department de mathématique
16 Septembre 2012

Accepted by .....

Table des matières

Introduction

La confection automatisée d'horaires de personnel est un sujet trés actuel et de nombreux chercheurs (euses) étudient ce problème. Que le personnel considéré soit du domaine hospitalier, ferroviaire ou aérien, etc., les secteurs d'application sont nombreux. Dans le présent mémoire, nous considérons le problème de confection d'horaires des personnes, abrégé par PSP (Personnel Scheduling Problem). D'ailleurs, il est important de préciser que l'utilisation d'un logiciel pour la résolution du PSP permet d'économiser du temps et de l'argent à L'etablissement En effet, cette tâche est habituellement accomplie par la personnes-chef de l'unité considérée et nécessite environ l'équivalent d'une semaine de travail pour chaque horizon . La méthode proposée dans ce mémoire occupe une place importante dans le domaine de la recherche opérationnelle et plus précisément, dans celui de la programmation mathématique. En effet, la technique que nous avons choisie est celle de la méthode de génération de colonnes couplée avec un algorithme de branch-and-bound afin d'obtenir une méthode exacte, soit l'algorithme de séparation et d'évaluation progressive (branch and price). L'algorithme est décomposé en deux niveaux : le problème maître et le problème auxiliaire. Le premier problème utilise la méthode de génération de colonnes pour résoudre la relaxation continue d'un programme linèaire en nombres entiers afin d'obtenir une borne inférieure sur la valeur optimale de ce dernier. Le deuxième problème est un algorithme de plus court chemin dans un graphe d'états et a la tâche de générer de nouvelles variables de coût réduit négatif lors de l'application de la méthode de génération de colonnes.

Le mémoire est organisé de la manière suivante :

Dans Le chapitre I , nous introduisans les notions d'optimisation et les notions liées au probléme de planification d'oraires Dans ce chapitre a pour but de rappeler comment résoudre un problème d'optimisation avec maximisation ou minimisation d'une fonction objectif fidelement à une certain nombre de contraints

Le chapitre II expose tout d'abord la modélisation retenue du problème par un programme linéaire mixte .

Le chapitre III est consacré à un modèle particulièrement important de la méthode de

génération de colonnes : la résolution du sous problème de confection d'horaires despersonnes (ou pricing subproblem) qui constitue dans ce cas un problème de recherche de chemins sur un réseau tridimensionnel dynamique avec contraintes additionnelles. L'algorithme pour résoudre ce problème est au coeur de la méthode de résolution et est développé dans ce chapitre.

Le chapitre IV regroupe la description de la mise en oeuvre expérimentale dans une première partie et l'analyse et l'exploitation des résultats de cette mise en oeuvre dans une seconde partie. Certains algorithmes annexes de traitement des données seront évoqués dans cette partie.

Chapitre 1

Généralités sur les problèmes d'optimisation

1.1 Introduction

Tout problème d'optimisation comporte une étape essentielle : la modélisation mathématique. Elle consiste en trois étapes :

1. Identification des variables de décisions , ce sont les paramètres sur lesquels l'utilisateur peut agir pour faire évoluer le système considéré.

2. Définition d'une fonction coût (appelée fonction objectif) permettant d'évaluer l'état du système (ex : rendement, performance,...).

3. Description des contraintes imposées aux variables de décision. Le problème d'optimisation consiste alors à déterminer les variables de décision conduisant aux meilleures conditions de fonctionnement du système (ce qui revient à minimiser ou maximiser la fonction coût).

1.1.1 Formulation et analyse d'un problème d'optimisation

Notions de minimum, maximum

On distinguera les notions de minimum et de maximum . Ces notions sont des prérequis pour les démonstrations des résultats d'existence et d'unicité d'extrema d'une fonction donnée.

Définition 1 *(Minorant/Majorant) Soit E un sous-ensemble de R.*

$m \in R \cup \{-\infty, +\infty\}$ est un minorant de E ssi $\forall x \in E\ m \leq x$:

$M \in R \cup \{-\infty, +\infty\}$ est un majorant de E ssi $\forall x \in E\ M \geq x$:

1.1.2 Description d'un problème d'optimisation

Formulation

On considère le problème d'optimisation suivant :

$$\min_{x \in R^n} f(x) \text{ sous la contrainte : } x \in X : \tag{1.1}$$

où X est un sous-ensemble de R^n. La fonction $f : X \subset R^n \to R$, appelée fonction objectif, est supposée au moins différentiable. L'ensemble X est appelé ensemble des contraintes.

Le problème (1.1) est dit réalisable si $X \neq \phi$.

Un cas fréquent en optimisation est celui où X est défini par des égalités et des inégalités :

$X = \{x \in R^n : h(x) = 0; g(x) \leq 0\}$

où $h : R^n \to R^p$ et $g : R^n \to R^q$ sont supposées continues. L'écriture "$h(x) = 0$" représente p contraintes d'égalité :

$h_i(x) = 0; i = 1; ...p$; et de même "$g(x) \leq 0$" représente q contraintes d'inégalités :

$g_i(x) \leq 0; i = 1; ....q$:

Dans ce cas, on dit qu'il s'agit d'un problème d'optimisation à n variables de décisions, p contraintes d'égalités et q contraintes d'inégalités.

Résoudre le problème (1.1) revient à chercher des points de minimum local (ou global, c'est encore mieux !) .

Dans toute la suite, on considère $f : A \subset R^n \to R$ où A est un sous-ensemble de R^n[29]

1.2 Définitions

Définition 2 *Si* $x^* \in A$ *est tel que*

$\forall x \in A, f(x^*) \leq f(x)$

alors on dit que f *admet un minimum sur* A *en* x^**. On note*

$f(x^*) = \min_{x \in A} f(x)$

et $x^* = \arg\min_{x \in A} f(x)$

Attention au vocabulaire : $f(x^*)$ est le minimum de f sur A ; f admet un minimum en x^* et x^* réalise le minimum de f sur A.

Remarque 3 *Le minimum d'une fonction, s'il existe, est unique. Il peut cependant être atteint en plusieurs points différents.En effet, le minimum s'il existe est nécessairement la borne inférieure m de l'ensemble des valeurs prises par la fonction f sur l'ensemble A. Il existe donc toujours une suite (x_n)de points de A telle que la suite $(f(x_n))$ converge vers m. Une telle suite (x_n) est dite suite minimisante. Le question de l'existence d'un minimum pour f revient alors à savoir si la suite (x_n) converge avec une limite dans A de sorte que la borne inférieure d l'ensemble des valeurs prises par f sur A soit atteinte.*

Définition 4 *Si* $x^* \in A$ *est tel que*

$\forall x \in A, f\ (x^*) \geq f\ (x)$

alors on dit que f *admet un maximum sur* A *en* x^* *. On note*

$f(x^*) = \max_{x \in A} f\ (x)$

et $x^* = \arg\max_{x \in A} f(x)$

Définition 5 *Si f admet en x^* un minimum ou un maximum, on dit qu'elle admet un optimum en x^**

Remarque 6 *dans la suite, on parlera que de minimum, les énoncés concernant les maxima pourront être déduits facilement.*

Définition 7 *On dit que f admet un minimum relatif (ou local) sur A en x^* ssi il existe un voisinage V de x^* dans A tel que f admette un minimum absolu sur V en x^* .*

Définition 8 *On appelle probème de minimisation :*

" Trouver $x^* \in \Omega$ *tel que* $f(x^*) = \min_{x \in A} f\ (x)$"

Remarque 9 *Un problème de minimisation n'admet pas nécessairement de solution :*

exemple : $x^* = \arg\min_{x \in [1,+\infty)} \frac{1}{x}$ *n'admet pas de solution.*

De plus s'il admet une solution, elle peut ne pas être unique :

$x^* = \underset{x\in[0,4\pi]}{\arg\min} \cos x$ *admet* π *et* 3π *pour solutions.*

Vocabulaire :

Si $A = R^n$, on parle d'optimisation sans contraintes (ou libre), sinon il s'agit d'optimisation sous contrainte (ou liée).

$A = \{x \in R^n | g_1(x) = 0, ..., g_p(x) = 0\}$ contraintes d'égalités

$A = \{x \in R^n | h_1(x) \leq 0, ..., h_q(x) \leq 0\}$ contraintes d'inégalités

On peut aussi mélanger les deux types de contraintes :

$A = \{x \in R^n | g_1(x) = 0, ..., g_p(x) = 0$ et $h_1(x) \leq 0, ..., h_q(x) \leq 0\}$

On parle alors de contraintes égalités–inégalités. C'est le seul type d'optimisation liée que nous étudierons.

Définition 10 *On considère un problème d'optimisation avec contraintes de type inégalité et* x^* *une solution de problème*

$$x^* = \underset{x\in A}{\arg\min}\ f(x)(A = \{x \in R^n | h_1(x) \leq 0, ..., h_q(x) \leq 0\})$$

Définition 11 *Si* $h_i(x^*) = 0$*, on dit que la contrainte* $h_i(x) \leq 0$ *est serrée en* x^* *(ou saturée, ou encore active).Dans le cas contraire* $(h_i(x^*) < 0)$*, on dit que la contrainte est inactive.*

1.3 Optimisation des problèmes sans contrainte

Nous nous intéressons dans cette partie à la conception de méthodes numériques pour la recherche des points $x \in R^n$ qui réalisent le minimum d'une fonction $f : R^n \to R$:

$(P) \quad \min_{x\in R^n} f(x)$

où f est supposée au moins différentiable. On parle d'optimisation sans contrainte.

1.3.1 Condition suffisante d'existence d'un point minimum

Avant d'étudier la solution (ou les solutions) de (P) , il faut s'assurer de leur existence.

Définition 12 *Une application* $f : X \subset \mathbb{R}^n \to \mathbb{R}$ *est dite infinie à l'infini (ou coercive) ssi* $\forall A \in \mathbb{R}; \exists\ R > 0 \forall x \in X; [\| x \| \geq R \Longrightarrow f(x) \geq A]$

On note : $\lim_{\|x\| \to +\infty} f(x) = +\infty$.

1.3.2 méthodes de résolution

Méthode de Newton locale

Pour construire les méthodes de gradient, nous avons remplacé f par son approximation linéaire au voisinage de l'itéré courant. Nous avons vu que ces méthodes ne sont pas très performantes, en partie parce qu'elles ne tiennent pas compte de la courbure qui est une information de second ordre.Méthode de Newton locale

Principe

Supposons maintenant que f est de classe C^2 et remplaçons f au voisinage de l'itéré courant x_k par son développement de Taylor de second ordre :

$$f(y) \sim q(y) = f(x_k) + \bigtriangledown f(x_k)^\intercal \succ (y - x_k) + \tfrac{1}{2}(y - x_k) \succ H[f](x_k)(y - x_k); [6]$$

où la valeur $f(x_k)$, le gradient $\bigtriangledown f(x_k)$ et la matrice hessienne $H[f](x_k)$ sont détaailler à[29].

On choisit alors comme point x_{k+1} le minimum de la quadratique q lorsqu'il existe et est unique, ce qui n'est le cas que si $H[f](x_k)$est définie positive. Or le minimum de q est réalisé par x_{k+1} solution de :$\bigtriangledown q(x_{k+1}) = 0$, soit :ou encore, en supposant que $H[f](x_k)$ est définie positive :

$$x_{k+1} = x_k - H[f](x_k)^{-1} \bigtriangledown f(x_k) : ..... \tag{1.2}$$

On reconnaît dans la formule (1.2) les itérations de la méthode de Newton vue en cours d'analyse numérique, appliquée ici à la résolution de l'équation :$\bigtriangledown f(x) = 0$.La méthode ne doit cependant jamais être appliquée en utilisant une inversion de la matrice Hessienne (qui peut être de très grande taille et mal conditionnée) mais plutôt en utilisant :

$H[f](x^k)d_k = - \bigtriangledown f(x_k) : x_{k+1} = x_k + d_k$ où d_k est l'unique solution du système linéaire :
$H[f](x^k)d_k = - \bigtriangledown f(x_k)$
d_k est appelée direction de Newton.

Cette méthode est bien définie si à chaque itération, la matrice hessienne $H[f](x_k)$est définie positive : ceci est vrai en particulier au voisinage de la solution x^* cherchée si on suppose que $H[f](x^*)$ est définie positive (par continuité de $H[f]$).

1.4 Optimisation des problème avec contraintes

Cette partie est une courte introduction à l'optimisation sous contraintes. On s'intéresse à la résolution de problèmes d'optimisation de la forme :

$\min_{\text{sc } x\in X} f(x)$
où X est un sous-ensemble non vide de R^n.

1.4.1 Condition suffisante d'existence d'un point minimum

La première question est celle de l'existence du point de minimum global de la fonction f sur Ω . Il existe principalement deux théorèmes qui permettent de répondre à cette question. Le premier affirme l'existence d'un point de minimum lorsque l'ensemble des contraintes est fermé borné. Le second est son équivalent pour les ensembles de contraintes fermés mais non bornés.

On considère $f : \Omega \to R^p$ où Ω est un ouvert. Le problème qui nous intéresse est le suivant :

$(Q) \qquad x^* = \arg\min_{x\in K} f(x)$

avec $K \subset \Omega$. L'ensemble K désigne l'ensemble des paramètres admissibles c'est- à-dire l'ensemble des états qui satisfaisant aux contraintes alors que Ω est l'ensemble de définition du côut f .

La première remarque est la suivante : si x^* solution du problème (Q) est dans l'intèrieur de K, tous les états voisins de x^* satisfaisant la contrainte .

Théorème 13 *Soit x^* solution du problème (Q). Si x^*est à l'intérieur du domaine K, alors*

$\nabla f(x^*) = 0$

La preuve est la même qu'en l'absence de contraintes, car on peut faire des petites variations dans toutes les directions autour de x^*.

Par contre, comme le montre l'exemple suivant, la condition d'annulation du gradient n'est pas nécessaire en présence de contraintes :

Conditions nécessaires d'ordre 1

Contraintes de type égalité

Dans ce paragraphe, on suppose que l'ensemble des contraintes K est donné par

$K = \{x \in R^n | g_1(x) = 0, ..., g_p(x) = 0\}$

Pour chaque i alors g_i désigne une fonction de x qui traduit une contrainte. Ici, on suppose que le nombre de contraintes est exactement p. On notera $g(x)$ le vecteur$(g_1(x), ..., g_p(x))^\intercal \in R_p$.

Définition 14 *On appelle Lagrangien du problème de minimisation (Q) l'application L de $\Omega \times R_p$ valeurs dans R définie par :*

$\forall (x, \wedge) \in \Omega \times R_p, L(x, \wedge) = f(x) + \wedge^\intercal g(x)$

Ou encore

$\forall x \in \Omega \ \forall \lambda_{1,...,\lambda_p} \in R, L(x, \lambda_{1,...,\lambda_i}) = f(x) + \sum_{i=1}^{p} \lambda_i g_i(x)$

Remarque 15 *Les nombres $\lambda_1, ..., \lambda_i$ sont appelés multiplicateurs de Lagrange.*

Remarque 16 *Le théorème des extrema liés s'interpréte simplement dans le cas où l'ensemble des contraintes est le cercle unité de R^2.*

En effet, minimiser une fonction $f(x, y)$ sous la contrainte $x^2+y^2 = 1$ revient à minimiser la fonction $j(\theta) = f(cos\theta, sin\theta)$ pour θdans R. On est donc ramené un problème sans contrainte. La condition d'optimalité s'écrit

$j\prime(\theta^*) = 0 \Longleftrightarrow -sin\theta^* \frac{\delta f}{\delta x}(cos\theta^*, sin\theta^*) + cos\theta^* \frac{\delta f}{\delta x}(cos\theta^*, sin\theta^*) = 0.$

En interprétant cette dernière comme l'annulation d'un déterminant, elle montre que les vecteurs

$\nabla f(cos\theta^*, sin\theta^*) et (cos\theta^*, sin\theta^*)^\intercal$ *sont colinéaires. Or ce dernier vecteur n'est autre*

que $\frac{1}{2}\nabla g(cos\theta^*, sin\theta^*) og(x, y) = x^2 + y^2 - 1$ *définit la contrainte.*

Ainsi, il existe un nombre réel μ tel que

$\nabla f(cos\theta^*, sin\theta^*) = \mu \nabla g(\cos\theta^*, sin\theta^*),$

il suffit de poser $\lambda = -\mu$ pour retrouver le résultat annoncé.

Remarque 17 *Il est crucial de vérifier la condition d'indépendance linéaire des gradients des fonctions définissant les contraintes.Cette condition est naturelle car elle écarte le cas de*

contraintes redondantes. Considérons en effet la minimisation de $x_1 + x_2^2$ *sous la contrainte* $x_1^3 - x_2^2 = 0$ *dans le cas d'une unique contrainte, l'indèpendance signifie la non annulation. On voit facilement que* $(0,0)$ *est l'unique solution. Pourtant on n à pas de condition d'ordre* 1 *:*

$$\nabla_x L(x,\lambda) = \begin{bmatrix} 1 \\ 2x_2 \end{bmatrix} + \lambda \begin{bmatrix} 3x_1^2 \\ 2x_2 \end{bmatrix}$$

En point $x = (0,0)$*, la condition d'ordre* 1 *fournit les deux ègalitès* $1 = 0$ *et* $0 = 0$*, ce qui est impossible.*

Contraintes de type inègalitè

Dans ce paragraphe, on suppose que l'ensemble des contraintes K est donnè par

$K = \{x \in R^n | h_1(x) \leq 0, ..., h_q(x) \leq 0\}$

On notera $h(x)$ le vecteur$(h_1(x), ..., h_q(x))^\intercal \in R^q$.

Définition 18 *On appelle Lagrangien du problème de minimisation* (Q) *la fonction* $L : \Omega \times R_q \to R$ *définie par.*

$\forall (x, M) \in \Omega \times R_q, L(x, M) = f(x) + M^\intercal h(x)$

Ou encore

$\forall x \in \Omega \ \forall \mu_1, ..., \mu_q \in R, L(x, \mu_1, ..., \mu_q) = f(x) + \sum_{i=1}^{i=q} \mu_i h_i(x)$

Le résultat suivant fournit la condition nécessaire d'ordre 1 *dans le cas des contraintes de type inégalités*

Remarque 19 *Si on a affaire à un problème de maximisation, on considère(- f) pour se ramener à un problème de minimisation, plutôt que de changer le signe des multiplicateurs.*

Remarque 20 *En dimension* 2*, la condition de qualification des contraintes peut s'interpréter. Ainsi les vecteurs gradients de contraintes serrées aux points considérés sont situés dans un même demi-plan strict.*

Contraintes mixtes égalités–inégalités

Ici l'ensemble contrainte est donné par

$K = \{x \in R^n | g_1(x) = 0, ..., g_p(x) = 0 \text{ et } h_1(x) \leq 0, ..., h_q(x) \leq 0\}$

Le Lagrangien s'écrit

$\forall(x, \wedge, M) \in R^n \times R^p \times R^q, L(x, \wedge, M) = f(x) + \wedge^\intercal g(x) + M^\intercal h(x)$

Les conditions nécessaires d'ordre 1 est donnée par le thèoréme suivant :

Théorème 21 *Soit x^* une solution de (Q). On suppose que les fonctions $f, g_1, ..., g_p$ et h_1,..., h_q sont de classe C^1. On fait, de plus, l'hypothèse de qualification des contraintes :*

$\exists d \in R^n, d^\intercal \nabla g_i(x^*) = 0$ *et* $h_i(x^*) = 0 \Longrightarrow d^\intercal \nabla h_i(x^*) < 0$

Les vecteurs $\nabla g_1(x^*), ..., \nabla g_p(x^*)$ sont linèairement indépendants.

Alors il existe des réels λ_1, . . . , λ_p, $\mu_1, ..., \mu_q$ tels que

(i) $\quad \mu_i \geq 0$

(ii) $\quad \mu_i h_i(x^*) = 0$

(iii) $\quad g_i(x^*) = 0$

(iv) $\quad \nabla f(x^*) + \sum_{i=1}^{p} \lambda_i \nabla g_i(x^*) + \sum_{i=1}^{q} \mu_i \nabla h_i(x^*) = 0$

preuve .voir[29]

1.4.2 méthodes de résolution

il existe plusieurs méthodes de résolution par exemple :

1.5 programmation linéaire

1.5.1 introduction à la programmation linéaire

la programmation linéaire est une branche de l'optimisation permettant de résoudre de nombreux problèmes écconomiques et industriels. Elle traite de maniére générale d'un problème d'allocation de resource limitées parmi des activités (tâches) concurrentes et ce d'une façon optimale.

L'adjectif linéaire indique que tout les fonctions mathématiques de ce modele sont linéaires tandis que le terme "programmation" signifie essentiellement planification.

1.5.2 Définition et propriétés :

On veut résoudre le probleme suivant dit programme linéaire sous forme stantard,

$$\inf_{x\in\mathbb{R}^n} cx$$

$$\text{tel que } \ Ax = b, x \geq 0 \tag{1.3}$$

ou A une matrice de taille $m \times n$,$b \in \mathbb{R}^n$, $c \in \mathbb{R}^n$ et la contrainte $x \geq 0$ signifie que toutes les composantes de x sont positives ou nulles. Dans tout ce qui suit on supposera que $m \leq n$ et que la range de A est exactement m .En effet,si $rg(A) \prec m$, certaines linges de A sont liées et deux possibilités se présentent : soit les contraintes (correspodantes à ces lignes) sont incompatibles, soit elles sont redondantes et on peut donc éliminer les lignes inutiles.

Le problème(1.3) semble être un cas particulier de programme linéaire puisque les contraintes d'inégalités sont seulement de type $x \geq 0$. Il n'en est rien, et tout programme linéaire de type

$$\inf_{x\in\mathbb{R}^n \ \text{telque} \ Ax\geq b \ , \ \ A'x=b} cx$$

peut se mettre sous la forme stantard (1.3) quite à changer la taille des données. En effet, remarquons tout d'abord que les contraintes d'inégalité $A'x \leq b', A'x \geq b'$.On peut donc restreindre au cas suivant(qui ne contient que des contraintes d'inégalité).

$$\inf_{x\in\mathbb{R}^n \ \text{telque} \ Ax\geq b} cx \tag{1.4}$$

Dans (1.4) on peut remplacer la contrainte d'inégalité en introduisant de nouvelle variables, dites d'écarts $\lambda \in \mathbb{R}^m$.La contrainte d'inégalté $Ax \geq b$ est alors équivalente à $Ax = b + \lambda$ avec $\lambda \geq 0$.Ansi (1.4) est équivalent à

$$\inf_{(x,\lambda)\in\mathbb{R}^{n+m} \ \text{telque} \ Ax=b+\lambda,\lambda\geq 0} cx \tag{1.5}$$

Finalment, si on décompose chaque composante de x en partie positive et négative, c'est-dire si on pose $x = x^+ - x^-$ avec $x^+ = max(0, x)$ et $x^- = min(0, x)$ (1.4) est équivalent à

$$\inf_{(x^+,x^-,,\lambda)\in\mathbb{R}^{2n+m} \ \text{telque} \ Ax^+-Ax^-=b+\lambda,x^+\geq 0,x^-\geq 0,\lambda\geq 0} c(x^+ - x^-) \tag{1.6}$$

qui est bien sous forme stantard (mais avec plus de variables). Il n'y a donc aucune perte de généralité à etudier le programme linéaire standard (1.3)

Définition 22 *L'ensemble* $X_{ad} = \{x \in \mathbb{R}^n \text{ tel que } Ax = b, x \geq 0\}$*est appelé ensemble des solutions admissibles. On appelle sommet ou point extrémal de* X_{ad} *tout point* $\bar{x} \in X_{ad}$ *qui ne peut pas se décomposer en combinaison (non triviale) de deux autres points de* X_{ad},*c'est-à -dire que, s'il existe* $y, z \in X_{ad}$ *et* $\theta \in]0,1[$*telle que* $\bar{x} = \theta y + (1-\theta)z$,*alors* $y = z = \bar{x}$

Remarque 23 *Le vocabulaire de l'optimisation est trompeur pour les néophytes. On appelle solution(admissible)un vecteur qui satisfait les contraintes.Par contre, un vecteur qui atteint le minimum de (1.3) est appelé solution optimale(ou point de minimum). On vérifie facilement que l'ensemble* X_{ad} *est un polyédre . (Rappelons qu 'un polyédre est un intersection finie de demi -espaces de* $\mathbb{R}^n$.*) Ses points extrémaux sont donc les sommets de ce polyédre. Lorsque* X_{ad} *est vide ,par convention on note que*

$\inf cx = -\infty$

$x \in \mathbb{R}^n$ *tel que* $Ax = b$ $x \geq 0$

Lemme 24 *Il existe au moins une solution optimale (ou point de minimum) du programme linéaire standard (1.6) si est seulement si la valeur du minimum est fini*

$-\infty \prec \inf cx \prec +\infty$

$x \in \mathbb{R}^n$ *tel que* $Ax = b,\ x \geq 0$

Définition 25 *On appelle base associée à* (1.6) *une base de* $\mathbb{R}^n$ *formée de m colonnes de A On note B cette base qui est une sous- matrice de A , carrée d'ordre m inversible. Aprés permutation de ses colonnes on peut écrire A sous la forme*(B, N) *ou N une matrice de taille* $m \times (n-m)$. *De la même façon on peut décomposer x en* (x_B, x_N) *de sorte qu' on a* $Ax = Bx_B + Nx_N$. *Les composantes du* x_B *sont appelées variables de base et celles de* x_N *variable hors base . Une solution basique (on de base) est un vecteur* $x \in X_{ad}$ *tel que* $x_N = 0$. *si en plus l'une des composantes de* x_B *est nulle, on dit que la solution basique est dégénérer. La solution basique correspondant à celle sommet de* X_{ad}.

Lemme 26 *les sommets du polyédre X_{ad} sont éxactementes solutions basiques.*

Proposition 27 *S'il existe une solution optimale du programme linéaire stantard (1.6) alors il existe une solution optimale basique*

1.5.3 Simplexes

Suposons que $D \subset \mathbb{R}^n$ est un ensemble compact et convexe. En plus, soit S_0 un simplexe de n dimension et contenant D. S_0 est l'ensemble initial de l'algorithme1(l'algorithme de Branch -and -Bound de base qu'on décrit par la suite)

Définition 28 *Soit S un n-simplexe avec l'ensemble des sommets $V(S) = \{v^0, ...v^n\}$ .on choisit un point $w \in S$,tel que $w \notin V(S)$.avec sa représentation unique suivante :*

$$W = \sum \lambda_i v^i, \lambda_i \geq 0 \ (i = 0, ...n), \sum \lambda_i = 1$$

Définition 29 *Alors la construction du sous- simplexe $S(i, w)$ (pour tout i tel que $\lambda_i \succ 0$) se fait à partir du simplexe que S en remplaçant le i^{eme} sommet par w, autrement dit :*

$S(i, w) = conv\{v^0, v^{i-1}, w, v^{i+1}, ...v^n\}$ .Cette partition est appelée " la subdivision radiale"

Proposition 30 *l'ensemble des sous- simplexes $S(i, w)$ peut être construit à partir d'un n-simplexe S selon une subdivision radiale arbitraire qui forme une partition de S ,c.à.d.., $S = \cup S(i, w)$ et $S(i, w) \cap S(j, w) = \delta S(i, w) \cap \delta S(j, w) i \neq j$.*

Définition 31 *On appelle un diamétre d'un polytope P et on le note $\delta(P)$ la valeur suivante :*

$\delta(P) = \max\{\| x - y\| : x, y \in P\}$

Pour un simplexe S $\delta(S)$ représente la longeur de la plus longue arête de S

Algorithme du simplexe

Solution de base admissible

soit le problème programmation linéaire (p) stantard .

$$(p)\begin{cases} \min\ f(x) \\ AX = b \\ x \geq 0 \end{cases}$$

Hypothèse : $n \geq m$ (plus de variables que d'équations) et $rg(A) = m$ (pas d'équation inutile). Donc aprés réarrangement des vecteurs, on peut écrire

$A = (a^{i_1}, ..., a^{i_m}, a^{i_{m+1}}, ..., a^{i_n})$

avec les m premiers vecteurs indépendants c.a.d.

$A = B + N$ avec $B(m, m)$ de rang m. D'où

$Ax = b$ ssi $(B + N) \begin{pmatrix} x_b \\ x_n \end{pmatrix} = b$

ce qui donne

$x_B = B^{-1}b + B^{-1}(-x_N)$

Définition 32 *B est appelée une base et $x_B = B^{-1}b$ la solution de base associée à B*

Si $x_B \geq 0$ alors $(x_B, 0)$ est une solution admissible de P.

Deux idées à retenir pour la suite :

Définition 33 – *une solution de base admissible est un sommet du polyhèdre défini par les contraintes.*

– *Le simplexe va faire passer d'une solution de base admissible à une autre qui améliore la fonction objectif.*

Solution de base admissible et polytope des contraintes

On considère que des polyhèdres situé dans l'orthant positif ($x_i \geq 0$ pour $i = 1, ..., n$). Les équations de définitions sont donc : des équations d'hyperplans et les contraintes $x_i \geq 0$.

Premier résultat : on suppose que $Ax = b, x \geq 0$ avec $rang(A) = m$ définit un ensemble borné. Alors c'est un polytope de R^{n-m}.

Deuxième résultat : x_B est une solution de base admissible ssi x_B correspond à un sommet du polytope associé.

Le polytope associé à

$$Ax = b$$

$$x \geq 0$$

donne $x_B = B^{-1}b + B^{-1}N(x_N)$ avec $x_B, x_N \geq 0$, c.a.d.

$B^{-1}N(x_N) \leq B^{-1}b$

$x_N \geq 0$

ce qui donne bien un polytope de $\mathbb{R}^{n-m}$.

Réciproquement, étant donné un polytope de $\mathbb{R}^{n-m}$. dans le premier orthant, ses équations de définition sont :

$x_l \geq 0$ pour $l = 1, ..., n-m$

$\sum_{l=1}^{l=n-m} a_{il}x_l \leq bi$ pour $i = 1, ..., n.$

d'ou

$x_k \geq 0$ pour $k = 1, ..., m$

$\sum_{l=1}^{n-m} a_{i,l}x_l + x_n = bi$ pour $i = 1, ..., n.$

B donnée, on associe à x le $\widehat{x}$ de $\mathbb{R}^{n-m}$ obtenu en ne gardant que les valeurs $x_i \notin B$. On admettra que si (C) est un polytope et (P) est le problème standard associé, alors

1. x solution de base admissible de(P)

2. $\widehat{x}$ le sommet correspondant est un sommet de (C)

Amélioration de la fonction objectif

On rajoute aux calculs précédents la fonction objectif.

$A = B + N$ avec $B(m, m)$ de rang m et $c = (c_B, c_N)$. ce qui donne

$Max\ z = c_B(x_B + c_N x_N)$

$x_B = B^{-1}b + B^{-1}N(-x_N)$

Après remplacement

max $z = c_B B^{-1}b - B^{-1}Nx_N + c_N x_N$

Le premier terme s'écrit $z_0 = c_B B^{-1}b$ et on écrit le second terme comme $\sum_{j \notin B}(z_j - c_j)(-x_j)$.On a la forme tableau

$$\begin{pmatrix} z \\ x_1 \\ . \\ . \\ x_m \end{pmatrix} = \begin{pmatrix} z \\ b1 \\ . \\ . \\ b_m \end{pmatrix} \begin{pmatrix} -x_{m+1} & . & -x_j & . & -x_n \\ & . & z_j - c_j & . & . \\ & & \alpha_{1,j} & . & . \\ & & & . & . \\ & & & . & . \\ & & \alpha_{m,j} & . & . \end{pmatrix}$$

Le pivotage consiste à

1. Choisir une variable à faire rentrer dans B (une colonne j)

2. Choisir une variable à faire sortir de$B$ (une ligne r)

3. Effectuer le pivotage

Il faut de plus

* Garder l'admissibilite de la base

* Ameliorer la fonction objectif (au moins ne pas la dégrader)

La valeur de la fonction objectif dans la nouvelle base aprés pivotage sur $\alpha_{r,j}$ est

$z_0 = z_0 - (z_j - c_j)b_r/\alpha_{r,j}$

On veut $z\prime_0 \geq z_0$

Pour maintenir l'admissibilité on a choisi $\alpha_{r,j} > 0$ et on sait que $b_r \succeq 0$,

donc il faut et il suffit que :

$(z_j - c_j) \prec 0$

1.6 Méthode Branch -and-bound

Branch-and -Bound (appelée parfois dans la litérateur francophone méthode de partition et d'élimination et notèe brièvement $B \ \& \ B$) est un algorithme assez général qui joue un rôle trés important dans la théorie d'optimisation globale. L'idée général de la méthode est de composer le problème primaire en sous problèmes parallèles(Branching) qui peut être graduellement plus facile à manipuler, puis évaluer les bornes inférieurs et supérieures(Bonding) des valeurs des solutions optimales sur ces problèmes secondaires. Par conséquent, le procédé Branch-and-Bound peut être représenté sous forme d'un arbre : le probleme initial est situé comme racine de cet arbre , et les branches sont les sous problèmes hièrarchiquement construits par l'algorithme.

cette construction est régie par la stratégie de la recherche qui déterminera la suite de solutions des problèmes secondaires. La séquence de la décomposition et la recherche de la solution, continue jusqu'à ce qu'il puisse vérifier que l'un ou l'autre sur la branche indiquée ne peut pas apporter une meilleure solution que la solution du condidat sortant (Trouvé déjà par le procédé Branch-and-Bound)

Notons que Branch-and-Bound a été à l'origine dévelopée pour résoudre des problèmes de la programmation linéaire entière[28]. Plus tard,cet algorithme a été appliqué avec succés dans des problèmes trés difficiles en optimisation globale comme :la minimisation concave; minimisation des fonctions lipchitziennes[40] ; et aussi dans la minimisation de la défférence de deux fonctions convexes (optimisation d.c)[35], etc.

Dans ce qui suit, on expose l'algorithme de base et les conditions de convergence.

Le procédé de l'algorithme branch-and- bound :

Définitions et préliminaires

Considérons le problème d'optimisation globale :

$\min_{x\in D} f(x)$,
où $f : A \rightharpoonup \mathbb{R}, D \subset A \subset \mathbb{R}^n$.

Définition 34 *soit P un polyèdre dans $\mathbb{R}^n$,telle que $intP \neq \phi$,et I un ensemble fini d' indices,la famille$\{P_i : i \in I\}$de sous polyèdres$(intP_i \neq \phi)$est dite partition de P, si*

$P = \bigcup_{i\in I} P_i et\ P_i \cap P_j = \partial p_i \cap \partial P_j \forall i, j \in I, i \neq j$
où ∂p_i est la frontière de P_i

Définition 35 *soit M un élément de la partition d'un polyèdre$P(D \subset P)$*

-Si $M \cap D = \phi$ M est est dit infaisable

-Si $M \cap D \neq \phi$ M est dit faisable.

-Sinon M est dit incertain.

L'algoriththme1 (algorithme Branch-and-Bound de base)

Etape: 0 Initialisation :

choisir

$D \subseteq P_0$

$S_{P_0} \subset D$

$-\infty \prec \beta_0 \leq \min f(D)$

$\begin{cases} P_0 \text{ est le plus petit polyèdre qui contient } D \text{ si ce dernier n}'\text{est pas un polyèdre.} \\ S_{P_0} \text{est un ensemble de points choisis dans l}'\text{ensemble faisable} \end{cases}$

Poser

$\text{Þ}_0 = \{P_0\}$

$\alpha_0 = \min f(S_{P_0})$

$\beta_0 = \beta(P_0)$ un minirant de$f$ sur $P_0 \cap D$ est choisir

$x_0 \in \arg\min f(S_{P_0})$ i,e : $f(x_0) = \alpha_0$

Si $\alpha_0 - \beta_0 \leq \varepsilon$,alors arrêter et poser $\alpha_0 = \min f(D)$

Etape k : $(k = 1, 2, ...)$

Au début de chaque étape k, on a la partition actuelle Þ_{k-1}de sous -ensembles de P_0 qui reste aprés élimination De plus, pour tout $P \in \text{Þ}_{k-1}$on a un ensemble de points

$S_P \subseteq D \cap P$ et des bornes $\beta(P), \alpha(P)$ vérifiant :

$\begin{cases} \beta(P) \leq \min f(P \cap D) \leq \alpha(P), \text{si } P \text{ est faisable} \\ \beta(P) \leq \min f(D), \text{si } P \text{ est incertain} \end{cases}$

on a aussi, les bornes $\beta_{k-1,}\alpha_{k-1}$qui vérifient f : $\beta_{k-1} \leq \min f(D) \leq \alpha_{k-1}$.

on a aussi un point $x^{k-1} \in D$ telque :

$f(x^{k-1}) = \alpha_{k-1}$

k.1.Eliminer tout sous ensemble $P \in P_{k-1}$qui vérifiie :

$\beta(P) \geq \alpha_{k-1}$

Considérer $\Re_k$ la classe des sous ensembles restants de P_{k-1} .

k.2 Sélectionner une classe d' ensembles non vide $\Im_k \subset \Re_k$ et construire une nouvelle partition

k.3 Eliminer chaque élément de $P \in \Im_k$ qui vérifie :

$P \cap X = \phi$

Poser $P_k^{'}$ la classe de tous les ensembles restants de $\Im_k$.

k.4 Déterminer pour chaque $P \in P_k^{'}$ un ensemble $S_P \subseteq P \cap D$ et un nombre $\beta(p)$ tels que :

$$\begin{cases} \beta(p) \leq \min f(p \cap D^{\circ}) \leq \alpha(p), \text{si } P \text{ est faisables} \\ \beta(p) \leq \min f(p), \text{si } P \cap D \text{ est incertain.} \end{cases}$$

de plus il faut voir :

$\beta(p) \geq \beta(p^{'})$

$S_p \cap S_{p'} = \phi$

pour tout ensemble $P^{'} \supset P$ de P_{K-1}

poser $\alpha(P) = \min f(S_P)$

k.5 Soit $P_K = \{\Re_k - \Im_k\} \cup P_K^{'}$

calculer

$\alpha_k = \min \{\alpha(p) : p \in P_K\}$

$\beta_k = \min \{\beta(p) : p \in P_K\}$

Si $\alpha_k \prec +\infty$, alors déterminer $x^k \in D$ tel que :

$f(x_k) = \alpha_k$

k.6 Si $\alpha_k - \beta_k \leq \varepsilon$ alors arrêter et poser $\alpha_k = \min f(D)(x^k$est la solution optimale)

Si non, poser $k = k + 1$

Aller a L'etape k.

Remarque 36 1) *pour que un élément P de la partition P_k, sera supprimmé à l 'étape k il faut qu'il soit un élément "sondé",c-à -d il doit vérifier la condition*

$\beta(P) \geq \alpha_{k-1}$.*Alors le critêre d'arrêt $\alpha_k = \beta_k$ signifie que tous les éléments de la partition sont sondés.*

2) *l'étape K est exécutée seulement s'il reste des éléments de la partition $\Re_k$; alors il suffit d'éxiger que $\Re_k \subset \Im$, c-à-d chaque élément sondé de la partition peut être encore divisé.en général, l'opération de subdivision est définie seulement sur une certaine famille $\Im$ de sous ensembles de $\mathbb{R}^n$ (par exemple simplexs, rectangles ou cônes d'un certain type).*

3)*Dans létape k.4 on peut évidement remplacer n'importe quel $P \in P_k^{'}$ par un plus petit ensemble*

$\widetilde{P} \subset P$*tele que* $\widetilde{P} \in \Im$,$\widetilde{P} \cap D = M \cap D$

4) *pour chaque partition P, S_p est une collection de point faisable incluse dans P, elle peut être rénovée au long de l'exécution de l'algorithme B & $B.$, Au cours de l'algorithme, l'ensemble*

S_p toujours fini. Alors, le procédé ci dessus est également défini dans le cas ou les ensembles S_P et $\beta(P)$ pour que $\{\alpha_k\} = \{f(x_k)\}$soit une suite décroissante,$\{\beta_k\}$une suite croissante et $\alpha_k \geq \min f(D) \geq \beta_k$,afin que la différence $\alpha_k - \beta_k$ peut mesurer approximativement la meilleure solution optimale x^k à l'étape k.

Pour une tolérance donnée $\varepsilon \succ 0$,l'algorithme va s'arrêter dés que $\alpha_k - \beta_k \prec \varepsilon$

Et puisque $\{\alpha_k\}$a une monotonie croissante et $\{\beta_k\}$ une monotonie décroissante,alors les limites

$\alpha = \lim_{n \longrightarrow \infty} \alpha_k$ te$\beta = \lim_{n \longrightarrow \infty} \beta_k$ vont exister,et par réccurence, ils vont satisfaire $\alpha_k \geq \min f(D) \geq \beta_k$.

L'algorithme est dit fini si $\alpha_k = \beta_k$ à une certaine étape, tandis qu'il est convergent si $\alpha_k - \beta_k \to 0$,c-à-d $\lim_{X_K \longrightarrow +\infty} f(X_K) = \beta = \min f(D)$.

Conditions d'arrêt et de convergence

Si le procédé s'achève à l'itération k , alors évidement, x^k est la solution optimale et α_k est la valeur optimale de la fonction objectif $(\alpha_k = f(x^k))$. Cependant , on ne peut pas garantir l'arrêt de l'algorithme en un nombre fini d'itérations, on doit donc établir des condition qui assurent que tout point d'accumulation de la suite $\left\{x^k\right\}$ est une solution optimale du probleme (P). Notons d'abord que si l'algorithme est infini, alors il doit générer au moins une suite" extraite"$\{P_K\}$ d'ensembles P_{Kq} issue des partitions successivement raffinées et vérifiant $P_{K_{q+1}} \subset P_{K_q}$ .Ceci est une conséquence immédiate du fait que, à chaque itération k , la classe P_k contient seulement un nombre fini d'ensembles issue de la partition. Dans l'arbre qui représente la procédure branch- and- bound la suite infinie extraite correspond à une branche dont les noeuds sont $P_{K_q}(q = 1, 2, 3, ...)$.

Théorème 37 *Si pour toute suite infinie $\{P_K\}, P_{K_{q+1}} \subset P_{K_q}(q = 1, 2, 3, ...)$.d'ensembles de partitions successivement raffinées, et pour toutes les bornes à l'itération k_p on a :*

$\lim_{q \longrightarrow \infty} (\alpha_{k_q} - \beta_{k_q}) = \lim_{q \longrightarrow \infty} (\alpha_{k_q} - \beta(P_{k_q})) = 0$

alors

$\beta = \lim_{k \longrightarrow \infty} \beta_k = \lim_{k \longrightarrow \infty} f(x^k) = \lim_{k \longrightarrow \infty} \alpha_k = \alpha$

et chaque point d'accumulation x^* *de la suite* $\left\{x^k\right\}$ *est une solution optimale de min* $\{f(x) : x \in D\}$

Supposons que $\left\{x^k\right\}$une suite infinie, puisque D est un compact (un polytope) et $\left\{x^k\right\} \subset D$, alors la suite $\left\{x^k\right\}$ posséde un point d'accumulation de$\left\{x^k\right\}$, donc il existe une sous-suite appartenant à $\left\{x^k\right\}$ qui converge vers x^*. Par construction des ensembles P_K cette sous-suite doit contenir une sous suite infinie$\left\{x^{k_q}\right\}$ telleque , $P_{K_{q+1}} \subset P_{K_q}(q = 1, 2, 3, ...)$.et par la continuité de f sur le compact D ona :

$\lim\limits_{k \longrightarrow \infty} f(x^{k_q}) = f(x^*)$.

Notons $f^* = \{f(x) : x \in D\}$.la suite des bornes inférieures $\{\beta_k\}$ satisfait les condition $\beta_{k+1} \succ \beta_k$ et$\beta_k \prec f^*$ alors sa limite existe, posons $\beta = \lim\limits_{k \longrightarrow \infty} \beta_k$ .En plus, la suite des bornes supérieures$\{\alpha_k\}$satisfait $\alpha_{k+1} \prec \alpha_k$ et $\alpha_k \succ f$ alors elle converge vers une limite finie $\alpha(\alpha = \lim\limits_{k \longrightarrow \infty} \alpha_k)$. On déduit donc que :

$\beta \leq f^* \leq \lim\limits_{k \longrightarrow \infty} f(x^*) = f(x^*) = \alpha$,

mais avec les conditions qui sécrivent aussi sous forme

$\lim\limits_{k \longrightarrow \infty} \alpha_{k_q} = \lim\limits_{k \longrightarrow \infty} \beta_{k_q}$,

et

$\begin{cases} \lim\limits_{k \longrightarrow \infty} \alpha_{k_q} = \lim\limits_{k \longrightarrow \infty} \alpha_k \text{ car } \left\{\alpha_{k_q}\right\} \text{est une sous- suite} \{\alpha_k\} \\ \lim\limits_{k \longrightarrow \infty} \beta_{k_q} = \lim\limits_{k \longrightarrow \infty} \beta_k \text{ car } \left\{\beta_{k_q}\right\} \text{est une sous- suite} \{\beta_k\} \end{cases}$

On déduit que

$\lim\limits_{k \longrightarrow \infty} \alpha_k = \lim\limits_{k \longrightarrow \infty} \beta_k$.

En regardant les différentes étapes de l'algorithme 1 , on voit qu 'il se base sur deux étapes trés importantes :

-La subdivision de l'ensemble faisable et les ensembles de partition.

-Le calcul des bornes inférieures et sépérieures de la fonction f sur les sous-ensembles de partitions.

Alors la question qui se pose est de voir comment choisir la stratégeie de subdivision et la façon de trouver les minimiseurs de la fonction objectif.

Les ensembles de partitions typiques et leurs raffinements

Comme on a indiqué auparavant, les ensembles de partitions P sont tout simplement des polytopes ou des polyédres comme les simplexes S, les rectangles(les pavés) R ou les cônes polyédriques C . Et pour chaque polytope on a une façon de générer ses ensembles de partitions et le calcul des minimiseurs de la fonction objectif sur ces ensembles.

Définition 38 .

1.7 La planification d'horaires de travail

Dans ce paragraphe,on met en scène la problèmatique de la planification des horaires dans un contexte général et sa complexité au quotidien dans les entreprises. En effet, la question de l'aménagement du temps de travail et de ses enjeux préoccupe toute société ou établissement actif ce qui a incité les chercheurs à proposer des méthodes et des techniques pour aider à gérer au mieux les horaires de travail. Pour cela nous définissons les différents types de plannings dans différents domaines de travail et plus particulièrement dans le domaine de santé.

1.7.1 La problèmatique de la planification d'horaires de travail :

La planification d'horaires de travail est un processus trés complexe, qui vise à organiser des activités humaines(principalement de travail) dans le temps et à optimiser l'utilisation des ressources, de façon à couvrir un besoin exprimé par une charge de travail prévisionnelle sous diverses contraintes. Elle aboutit à des programmes définissant les horaires de travail et de repos de la force de travail .

Pour mieux cerner ce qui est la planification et la complexité à sa réalisation, on s'intéresse à un ensemble de questions :

1.7.2 Qu'est ce que la planification ?

La planification est un instrument de gestion dont l'objectif est d'aboutir à des programmes permettant d'organiser et planifier le travail des salariés afin de rester pérenne dans

l'économie globale. Ceci passe par la détermination des capacités de tout un chacun et par le recensement des activités futures et des besoins en personnel.

La planification vise à affecter les ressources humaines pour chaque intervalle de temps sur un horizon donné, de telle manière que les besoins par intervalle soient couverts et que les différentes contraintes soient satisfaites.

1.7.3 Qu'est ce qu'un planning ?

Les plannings sont des calendriers de travail, où figurent à la fois le temps, l'affectation du personnel, les jours et les heures de travail, et les congés et repos. En effet, ils apparaissent dans les cas suivants :

· Si le travail doit être assuré pendant plus d'une journée, il faut prévoir la succession de plusieurs personnes sur le même poste dans la journée. Un outil d'aide est nécessaire lorsque le nombre de postes dépasse la quinzaine, par exemple gérer les absences imprévues des salariés.

· Si le travail doit être assuré pendant plus de 35 heures par semaine, un outil automatique devient indispensable lorsque le nombre de postes dépasse la trentaine pour gérer la succession de plusieurs personnes dans la semaine, ainsi que les absences imprévues.

Les plannings peuvent être utilisés pour planifier les horaires de présences du personnel ou les tâches effectuées par le personnel :

- Planning des horaires de présence :

- ce type de planning est utilisé pour prévoir les horaires de présence du personnel sans préciser les tâches journalières à effectuer soit pour des raisons de sécurité, soit pour une meilleure souplesse .

- Planning des tâches :

- ce type de planning est utilisé dans les entreprises à haute technicité, comportant plusieurs métiers et compétences distincts, où il est souhaitable d'affecter le personnel en fonction des tâches. Ce qui exige une décomposition fine des opérations et le repérage des tâches que chaque personne est capable d'accomplir.

Les plannings peuvent être journaliers (spécifiant les pauses et périodes de travail de la journée de chaque employé), hebdomadaires (utilisés pour une paie hebdomadaire), mensuels (utilisés pour le calcul des coûts pour les besoins de la paie mensuelle) ou annuels (permettant de gérer les congés annuels des employés).

Selon leur spécificité et les branches d'activités concernées, les plannings portent différents noms. Un planning spécifiant les programmes de travail de chaque employé nominativement sur un horizon (un intervalle de temps où un planning est élaboré) d'un mois est appelé tableau de service. Lorsque le planning représente les programmes de travail et de repos non nominatifs sur un nombre entier de semaines, on parle de grille de travail.

Certains plannings sont cycliques, s'ils reflètent une certaine périodicité des horaires individuels c'est à dire si au bout d'une durée D (mesurée généralement en semaine), le salarié retrouve son planning de départ. Autrement, ils sont dits acycliques c'est à dire ils sont différents chaque semaine.

1.7.4 A Quoi sert un planning ?

Depuis le début des années 80, la gestion des ressources humaines à été reconnue comme une activité stratégique pour l'entreprise. Avec cette reconnaissance, l'intérêt d'élaborer des plannings s'est vu accroître de plus en plus car ils permettent :

- aux entreprises exerçant une activité continue ou quasi-continue de répartir convenablement leur personnel (compagnies aériennes, entreprises de transports, hôpitaux, etc...),

- aux entreprises cherchant à se rendre plus accessibles à la clientèle d'étaler les horaires d'ouverture (grands magasins, banques, etc...),

- à toutes les entreprises de surmonter leur exigences de productivité et de mieux gérer les présences et absences de leur personnel.

Les situations où un planning est utile son nombreuses. Elles justifient l'existence de différentes formes de plannings dans un même système : plannings à court, moyen et à long terme.[40]

1.7.5 Comment est évalué un planning ?

Pour que les plannings élaborés soient satisfaisants, ils doivent vérifier un ensemble de contraintes et établir un meilleur compromis entre les différents acteurs (exemple : le chef d'entreprise, le planificateur, le commercial, le syndicaliste et le salarié).

Lorsque les différentes solutions alternatives sont connues, une négociation se déroule de la manière suivante : chaque acteur donne son opinion. Les points d'accord sont très vite expédiés et les points litigieux sont débattus. Et des solutions de compromis sont dégagées.

Les difficultés de négociation augmentent avec le nombre d'acteurs et le nombre de solutions alternative. L'aspect combinatoire (pour l'élaboration des plannings) rend d'autant plus difficile la négociation, car les opinions sont plus difficiles à formuler [36]. Les moyens informatiques apportent une aide certaine notamment dans l'acquisition et la confrontation des données individuelles.

1.7.6 Qui peut se charger de l'élaboration d'un planning ?

Dans la plupart des entreprises, cette tâche peut être centralisée ou déléguée à des cadres de l'entreprise appelés planificateurs.

Le planificateur doit prendre la décision qui correspond le mieux aux préférences des différents acteurs, justifier son choix, car son expérience de la tâche fait de lui un interlocuteur privilégié pour évaluer rapidement et effectuer des jugements de l'orientation à donner à la recherche de solutions de meilleur qualité afin d'aboutir à un choix pertinent.

Une collaboration réussie doit permettre au planificateur de participer efficacement à l'élaboration des plannings. La génération automatique des planning joue un rôle primordial.

1.7.7 Différents types de plannings :

Dans la construction de plannings d'horaires de travail, si créer un planning optimisé d'une journée est aisé, mais créer un bon planning pour un mois ou une année est beaucoup plus complexe. En plus de la complexité combinatoire du problème, il faut tenir compte de la diversité des contraintes applicables et qui sont souvent contradictoires. Pour ce qui suit, on évoquera les différents types de plannings et les approches utilisées pour réaliser ces types de plannings.

Types de plannings dans le domaine de la santé :

Les plannings dans le domaines de la santé sont des calendriers de travail où figurent à la fois le temps, et l'affectation des personnels (jours et horaires de travail, congés et repos). Ils sont établis au niveau de chaque équipe, ils sont à la fois une tâche, un document d'organisation du travail, et un élément contribuant à la gestion administrative du personnel. Cette tâche est parmi les plus difficiles et les plus délicates. Difficile parce qu 'elle repose sur la recherche de solutions combinatoire, répond à des contraintes multiples, remise en cause de manière fréquente par l'absentéisme et délicate car elle impose toujours une négociation avec les acteurs (médecins, infirmiers) de l'équipe et la direction du service de soins et l'administration. Les documents établis sont des calendriers sur lesquels on inscrit les affectations des médecins et des infirmiers ; ils sont généralement des tableaux à double entrée avec en ligne le personnel et en colonne le temps.

Plusieurs méthodes ont été utilisées dans la littérature spécialisée pour gérer ce type de plannings telles que la programmation par contraintes, la recherche locale (recuit simulé, tabou), les algorithmes évolutionnaires et d'autres méthodes.

1.8 théorie de base

1.8.1 Introduction

Le problème de la confection d'horaires est un problème ardu dans la réalisation a la main est une tache draconienne qui peut mobilisé plusieurs personnes plusieurs jours de travail sans oublié, que toutes modification des données du problème peut complètement remettre en cause la solution trouvée. Dans une grande partie des premiers travaux, le problème de planification des horaires du personnel soignants a été traité en tant que problème d'optimisation 'OP '. Il renferme les méthodes dites exactes. Ces méthodes sont celles ayant comme objectif de trouver une solution de valeur minimale ou maximale et de prouver mathématiquement que cette valeur est optimale. En d'autres termes, la meilleure solution trouvée par ces méthodes doit être celle ayant la meilleure valeur parmi l'ensemble de toutes les solutions possibles du problème considéré.

La méthode de programmation mathématique est une approche exacte pour les problèmes

$$MaxZ = \sum_{i=1}^{L} \sum_{j=1}^{J_i} C_{ij} X_{ij}$$

$$\sum_{i=1}^{R} \sum_{j=1}^{J_i} A_{ij} X_{ij} \geq N_{SDE} \quad ; \quad (SDE : Soignants\ Diplômes\ d'Etat)$$

$$\sum_{i=1}^{L} \sum_{j=1}^{J_i} A_{ij} X_{ij} \geq N_{TOT} \quad ; \quad (TOT = SDE + AS, avec\ AS : Aides\ Soignants)$$

$$\sum_{j=1}^{J_i} X_{ij} = 1; .i = 1,2..., L$$

$$X_{ij} \in \{0,1\}, .i = 1,2,..., L.; .j = 1,2,..J_i$$

d'optimisation combinatoire. Curieusement, peu d'auteurs ont fait appel à une méthode exacte pour résoudre le problème en utilisant les techniques traditionnelles de programmation linéaire, programmation en nombre entier, programmation par objective.

1.8.2 Modélisations et techniques de Résolution du problème

La méthode de Warner [39]est parmi les premières à s'intéresser aux plannings non cycliques. La résolution du problème permet d'écrire un programme mathématique, où on associe à chaque soignant un ensemble de variables, chaque variable correspondant à un planning individuel potentiel. L'objectif est de maximiser la satisfaction collective des soignants. Elle se fait en deux étapes. La première étape consiste à trouver une solution réalisable. La deuxième étape essaye d'améliorer la solution en utilisant la méthode des ''blocs pivotants".

Chaque modèle de travail représente un bloc. Le modèle mathématique est :

Avec :

N_{SDE} un vecteur de dimension finie, spécifiant les contraintes de charge en terme de nombre minimum de SDE requis pour chaque tranche.

N_{TOT} un vecteur similaire pour les charges totales (SDE +AS)

$$MinZ = \sum_{j=1}^{J} C_j X_j + \sum_{i=1}^{7} \sum_{l=1}^{L_i} \alpha_{il} U_l$$

$$\sum_{j=1}^{J} A_{ij} X_j - \sum_{k=1}^{K_i} X_{ik} = 0 \quad pour \;\; i = 1...7$$

$$\sum_{k=1}^{K_i} A_{lk} X_{ik} - O_{il} + U_{il} = D_{il} \quad ; \quad pour$$

$$\sum_{j=1}^{J} X_j = W_{total}$$

$$X_{ik}, X_j \in N^*$$

C_{ij} la valeur relative du modèle j pour chaque soignant i, exprimant le poids affecté au modèle j.

Bailey et Field[3] ont formulé l'intégration des deux sous problèmes (affectations des tranches horaires [22] et affectation des jours de repos en un seul problème en le formulant comme un programme linéaire en nombre entier, mais en considérant que le nombre de personnel est illimité. la nécessité d'intégrer ces deux sous problèmes a été reporté aussi dans [4]. L'originalité de cette méthode est de prendre en compte le nombre de personnel limité. La formulation du problème est donnée comme suit :

Où :

A_{lk} : est une matrice $(L_i * K_i)$; $a_{lk} = 1$ si le modèle k est appliqué pendant la période $l, 0$ sinon

D_{il}:personnel requis pour la période l du jour i

K_i : nombre de modèles de travail possibles pour les jours i.

L_i : nombre de périodes considérées le jour i.

X_{ik} : nombre de personnel affecté au modèle k, le jour i.

O_{il} : Surplus du personnel pendant la période l le jour i.

$$W = max\left\{\left\lfloor \frac{10D + 4E}{7} \right\rfloor, \left\lfloor \frac{BE}{B - A} \right\rfloor\right\}$$

U_{il} : manque du personnel ; coût de manque du personnel ;

X_j : effectif affecté au modèle de travail j (modèle de jour de repos)

C_j : coût d'un soignant travaillant selon le modèle j

A_{ij} : Une matrice $(7 * J)$;$a_{ij} = 1$ si le modèle j est de travail le jour i, 0 sinon.

W_{total} : l'effectif intervenant dans le planning

L'objectif du problème intégré est de minimiser le coût hebdomadaire affecté aux modèles de travail plus les coûts journaliers de manque de personnel.

La première contrainte, posée sur chaque jour de la semaine, exprime le fait que le nombre de personnel X_j déterminé par le problème de jours de repos, soit égal au nombre de personnel total X_{ik} déterminé par l'affectation des tranches horaires.

La deuxième contrainte exprime le fait que le nombre de personnel disponible chaque période par jour est égal à celui requis, en tenant compte du manque de personnel et de sureffectif.

La troisième contrainte exprime le fait que le nombre de personnel soit limité.

HUNG [19] utilise une méthode qui commence par calculer le nombre minimum de personnel nécessaire, ensuite il applique une heuristique pour générer un planning cyclique satisfaisant toutes les contraintes. Le modèle se base sur les contraintes suivantes :

- Chaque soignant travaille7 jours pendant une suite de deux semaines
- un jour de travail est composé en P tranches horaires
- D_j soignants au minimum sont requis pour la tranche horaire j pour les jours de la semaine, et E_j pour les week-ends
- Affectation à chaque employé au moins A week-ends libres parmi B week-ends.

$D = \sum_{j=1}^{p} D_j$

et $E = \sum_{j=1}^{p} E_j$

La borne minimum est calculée comme suit :

La procédure commence par affecte les week-ends libres aux soignants, puis les jours libres

$$x_{ij} = \begin{cases} 0 \text{ Si le soignant i travaille le jour } j \\ 1 \text{ Sinon} \end{cases}$$

$$\sum_{i=1}^{N} x_{ij} \leq N - Min_j$$

additionnels, les tranches horaires des week-ends, les tranches horaires de la semaine et enfin elle finit par affecter les extra tranches horaires aux soignants libres.

CHEN ET YOUNG[10] ont abordé le problème d'affectation des tours en utilisant les systèmes experts hybrides, combinant les systèmes experts avec la programmation par objectif. Notons d'abord que la programmation par objectif a été aussi utilisé par Arthur et Ravindram [1] ainsi que Ozkarahan et Bailey [32] .Leurs méthodes permettent d'intégrer les deux sous problèmes de l'affectation des jours de repos ensuite de l'affectation des tranches horaires en un même problème avec introduction des déviations dans les contraintes. Chen et Young traitent le problème en deux étapes : la première étape consiste à résoudre le sous problème d'affectation des jours de repos par la méthode de programmation par objectif.

La variable :

Pour assurer la présence d'un nombre minimum de soignants par jour, la contrainte suivante est écrite :

où N représente le nombre de soignants et le nombre minimum requis pour le jour j. Dans la deuxième étape un système expert est utilisé pour l'affectation des soignants aux différentes tranches horaires. Rappelons qu'un système expert à besoin de connaissance et d'expertises sur le domaine étudié. La plus importante phase dans l'établissement d'un système expert est l'acquisition des

connaissances car chaque conclusion, fournie par le système, dépend du contenu de sa base de connaissances. La représentation des données est faite par un système à base de règles, une règle a la forme suivante : SI Condition ALORS Conclusion 1 Sinon Conclusion 2.

En 2004 J.P DOYON a utilisé la méthode de génération de colonne hybridé ave la méthode de Branch and Bound pour la confection d'horaires du service des infirmiers dans l'hôpital de

Montréal au canada. Celui-ci à présenté la formulation mathématique du problème en deux niveaux sans tenir compte des différents niveaux de qualification des infirmiers. C'est-à-dire la modélisation mathématique présentée ne tient pas compte des différents grades des infirmiers pour avoir un planning adéquat et flexible. Ainsi que lors de l'application de la méthode de résolution du problème l'auteur n'a pas pris en considération les domaines de définitions des préférences pour le gain du temps et la simplicité de la résolution

Récemment en 2007 olivier Richard a étudié le problème de confection d'horaires mais dans le domaine Aérien en utilisant la méthode de génération de colonne hybridée avec la technique de branch and bound. Cette technique à prouver son efficacité et à donné son importance pour la résolution des problèmes de plannings

Chapitre 2

Déscription et modélisation du problème

2.1 Présentation du problème de confection d'horaires

Chaque établissement doit produire à plusieurs reprises des plannings d'affectation pour sont personnel. L'élaboration correct et valide du programme du perssonnel, a un grand impact sur la qualité de travail, le recrutement de ce personnel, le développement des budgets et de diverses autres fonctions. Des programmes peuvent être produits manuellement par le chef de service de chaque unité fonctionnelle, mais ce problème reste toujours difficile. Une vue générale peut être trouvée dans[6]

Le role ou la raison principale du problème est surtout de garantir la continuité des services et la qualité du travail quotidiennement. Habituellement, les chefs de services consomment et dépensent une quantité substantielle de temps pour réaliser le programme d'affectation et particulièrement lorsque les demandes individuelles de personnel sont nombreuses. Le problème d'établissement de programme implique de produire des affectation pendant une période finie (semaine, tout les quinze jours,mensuelle) sujet à une variété de contraintes dures et souples telle que le règlement légal, la politique d'affectaion du personnel, les préférences et beaucoup d'autres conditions qui peuvent être spécifiques à l'établissement.

2.1.1 L'objectif

L'objectif consiste à affecter à chaque personne un horaire qui respecte ses contraintes intrinsèques. On nommera par la suite ces horaires comme suit : horaires individuels. Ensuite, l'ensemble de ces horaires individuels pris ensemble doit satisfaire les contraintes decharge ou de quotas. Le terme horaire général sera utilisé par la suite pour désigner un ensemble d'horaires individuels.

Ainsi, l'objectif est de trouver un horaire général satisfaisant toutes les contraintes(contraintes intrinsèques et contraintes globales ou de quotas). Chaque horaire général généré devra être réalisable. Mais en plus, parmi l'ensemble de tous les horaires généraux réalisables, il suffit de trouver le meilleur, soit celui qui minimise la fonction objectif.

(Notations relatives aux données du problème :)

$K = \{k_1, k_2, ..., k_{|K|}\}$, l'ensemble des personnes.

$D = \{d_1, d_2, ..., d_{|D|}\}$, l'ensemble des jours de l'horizon courant, où d_1 et $d_{|D|}$ sont respectivement le premier et le dernier jour de l'horizon.

$T = \{t_1, t_2, ..., t_{|T|}\}$, l'ensemble des quarts de travail.(Shifts types)

$D_t = \{(d, t) : d \in D, t \in T\}$, l'ensemble des paires jour - période.

$\Lambda = \{\ell_1, \ell_2, ..., \ell_{|\Lambda|}\}$, l'ensemble des compétences des personnes.

$T_k \subseteq T$, l'ensemble des quarts de travail possibles pour $k \in K$.

$D_{T_k} = \{(d, t) : d \in D, t \in T_k\}$, l'ensemble des affectations possibles pour $k \in K$.

(Q_L) Contrainte de quotas définie sur$(d, t) \in D_t$ et $L \in L_{dt}$ et associée au triplet (q_L, q_L^-, q_L^+) (respectivement quota cible, déficit et surplus maximaux acceptés).

$L_{dt} = \{L_1, L_2, ..., L_{|L_{dt}|}\}$, chaque $L \in L_{dt}$ est tel que $L \subseteq \Lambda$ et est associé à une contrainte (Q_L) définie sur $(d, t) \in D_T$.

$k \subseteq \Lambda$, l'ensemble des compétences de la personne $k \in K$.

O_k et $F_k \subseteq D_{T_k}$, respectivement l'ensemble des affectations obligatoires et interdites pour la personne $k \in K$.

u^0 Premier noeud de l'arbre de recherche de solution.

$(P^u)PLNE$ du noeud u de l'arbre de recherche.

$S^u \subseteq S$, l'ensemble des solutions (entières) du $PLNE(P^u)$.

$S_k^u \subseteq S_k$, l'ensemble des horaires valides de la perssone $k \in K$ pour le $PLNE(P^u)$.

$RFRC$ Ryan-Foster Respect Combination.

$I_k^u \subseteq \{0, 1, ..., 2^{|Cuk|} - 1\}$, l'ensemble des $RFRC$ possibles pour la perssone $k \in K$ pour le $PLNE$ (P^u).

$K_{I_k^u} = \{(k, c) : k \in K, c \in I_k^u\}$. Une $RFRC$ $c \in I_k^u$ d'une persson $k \in K$ au $PLNE$ (P^u) est notée par $(k, c) \in K_{I_k^u}$

$K_{D_{Tk}} = \{(k, d, t) : k \in K, (d, t) \in D_{Tk}\}$. Un branchement binaire est noté par $(k, d, t) \in K_{D_{T_k}}$

$K_{D^2_{T_k}} = \{(k, d_i, t_i, d_j, t_j) : k \in K, (d_i, t_i, d_j, t_j) \in D^2_{T_k}\}$. Un branchement $Ryan - Foster$ est noté par $(k, d_i, t_i, d_j, t_j) \in K_{D^2_{T_k}}$

O_k^{cu} et F_k^{cu} Respectivement l'ensemble des affectations obligatoires et interdites définissant $(k, c) \in K_{I_k^u}$.

$S_k^{cu} \subseteq S_k^u$, l'ensemble des horaires respectant la combinaison $(k, c) \in K_{I_k^u}$ du $PLNE(P^u)$.

z_u^i Valeur optimale du problème (R^{ui}).

x_u^i Solution primale-duale réalisable (optimale) de valeur z_u^i du problème(R^{ui}).

(R^{un}) Dernier problème réduit considéré par GC au problème (R^u).

z_u^n Valeur optimale du problème (R^{un}). Utilisé par le test $z_u^n \neq z_u^*$.

x_u^n Solution optimale de valeur z_u^n du probléme(R^{un}).

2.1.2 Notation relatives à la modélisation et à la solution du problème

(P) Formulation du problème sous forme d'un Programme Linèaire en Nombres Entiers$(PLNE)$.

S L'ensemble des solutions (entières) du $PLNE(P)$

S_k L'ensemble des horaires valides d'une perssone $k \in K$ pour le $PLNE(P)$.

z^* Meilleure valeur connue du $PLNE(P)$. Initialement, $z^* = +\infty$.

p_{ks} Coefficient indiquant le poids de l'horaire $s \in S_k$ de la perssone $k \in K$

$(p^-), (p^+)$ Pénalités accordése par unité de déficit et de surplus avec le quota requis de chaque contrainte de quotas.

x_{ks} Variable de décision égale à 1 si l'horaire $s \in S_k$ est affecté à l'infirmière $k \in K$ dans l'horizon courant et à 0 sinon.

a_{ksdt} Variable égale à 1 indiquant si l'affectation $(d,t) \in D_{Tk}$ a été affectée à l'infirmière $k \in K$ dans l'horaire $s \in S_k$ et à 0 sinon.

$(e_L^-)(e_L^+)$ Variables d'écart respectivement de déficit et de surplus associée à la contrainte de quotas (Q_L), où $L \in L_{dt}$ et $(d,t) \in D_T$.

λ_L Variable duale associée à la contrainte de quotas (Q_L), où $L \in L_{dt}$ et $(d,t) \in D_T$.

μ_k Variable duale associée à la contrainte de partionnement de la perssone $k \in K$.

c_{ks} Coût réduit de l'horaire $s \in S_k$ dela perssone $k \in K$.

GC Technique de génération de colonne

(R^{ua}) Relaxation continue résultant de l'ajout de variables artificielles à (R^u).

a_k Variable artificielle créer pour une perssone $k \in K$.

a_L Variable artificielle créer pour la contrainte de quotas (Q_L).

n Nombre d'itérations lors de l'application de GC au problème .

Notations relatives au problème auxiliaire et à une persson $k \in K$:

$G_k = (V_k, E_k)$, le graphe associé à la persson k, où $V_k = \{v_0, v_1, ..., v_n\}$et $E_k \subseteq \{(v_i, v_{j)} : v_i, v_j \in V_k, v_i \neq v_j\}$.

v_0 et v_n Respectivement le noeud source et le noeud puits du graphe G_k.

D_i Ensemble des états réalisables au noeud $v_i \in V_k$.

$R(x_j)$ Ensemble des états qui ont donné naissance à l'état $x_j \in D_j$ au noeud $v_j \in V_k$.

$\hat{C}_k = (\widehat{V_k}, \hat{E}_k)$ le graphe d'états obtenu à partir du graphe G_k

$\widehat{V_k} = \{v_i{\cdot}x_i : v_i \in V_k, x_i \in D_i\}$ est l'ensemble des noeuds d'état de $\hat{C}_k$.

$\hat{E}_k = \{(v_i{\cdot}x_i, v_j{\cdot}x_j) : v_i,\ v_{\ j} \in V_k,\ x_i \in D_i,\ x_j \in D_j,\ x_i \in R(x_j)\}$ est l'ensemble des arcs d'états de $\hat{C}_k$.

c_{ij} Coût associé à l'arc $(v_i, v_j) \in E_k$ et aux arcs d'état $(v_i{\cdot}x_i, v_j{\cdot}x_j) \in \hat{E}_k$.

$\hat{G}_{ki}^{cu}$ Graphe d'états représentant la combinaison $(k,c) \in K_{I_k^u}$ du $PLNE(P^u)$ et les variables duales de l'itération courante i de GC.

PA La phase , du problème auxiliaire, appliquée au graphe $\hat{G}_{ki}^{cu}$ pour obtenir un ensemble, noté PA $(\hat{G}_{ki}^{cu})$, de variables $s \in S_k^{cu}$ hors base et de coût réduit négatif $(c_{ks} < 0)$.

$P_k^{cu}(v_j{\cdot}x_j)$ Pour $v_j{\cdot}x_j \in \widehat{V_k}$, l'ensemble des sous-chemins $v_0{\cdot}x_0 v_j{\cdot}x_j$ qui sont réalisables pour la combinaison $(k,c) \in K_{I_k^u}$ du $PLNE(P^u)$.

P_k^{cu} L'ensemble de tous les chemins $v_0{\cdot}x_0v_n{\cdot}x_n$, où $v_n{\cdot}x_n \in \widehat{V_k}$

est un puits quelconque, qui sont réalisables pour la combinaison $(k,c) \in K_{I_k^u}$ du $PLNE(P^u)$.

2.2 Modélisation mathématique

Nous allons dans cette section proposer la modélisation qui a servi de base à l'outil développé pendant les travaux de recherche. Cette modélisation avait été proposée par [38] , [6] et [26]. Nous établissons des modifications pour répondre à un niveau de raffinement plus poussé dans la modélisation et la résolution du problème. Nous présenterons donc, dans un premier temps, les données dont nous disposons pour résoudre le problème, puis la modélisation mathématique que nous avons retenue.

En appliquant l'idée de la décomposition de $Dantzig - Wolf$ (décrit en annexe 1), nous décomposons le modèle mathématique en deux parties : Un problème maître et un problème auxiliaire.

2.2.1 Problème Maître

Données du problème

Pour générer les horaires les établissements doivent fournir les données suivantes :

. Soit N_i le nombre de soignants telle que i représente l'indice appartenant à l'ensemble fini I des soignants. $K = \left\{k_1, k_2, k_3 ... k_{|k|}\right\}$.

. Soit D une période ou l'horizon de planification composé de plusieurs jours ordonnés d. $D = \left\{d_1, d_2, ... d_{|D|}\right\}$.

. Soit T_t les quarts de travail que l'on peut affecter dans l'horaire des personnes. Un quart de travail est le nombre d'heures consécutives durant lesquelles la personne doit être présent à l'établissement.

. Un ensemble de niveau de qualification (compétences) $\Lambda = \left\{l_1, l_2, ... l_{|\Lambda|}\right\}$ qui servira à la fois à caractériser les personnes et à exprimer les contraintes de quotas. On notera par L_k l'ensemble des compétences d'une personne k et L un ensemble de l'ensemble des compétences distincts exprimés pour les contraintes de quotas

. Les contraintes de charges ou de quotas, qui spécifient que l'on doit avoir Q_{dtl} personnes d'un ensemble de compétences L présents le jour d pour effectuer les quarts de travail T_t. Ces quotas cibles sont spécifiés avec une fenêtre ou une marge de réalisabilité [Q_{dtl} $+Q_{dtl-\max-deficite}$, $Q_{dtl} + Q_{dtl\,\max-surplus}$]

. Des pénalités $P^{+}etP^{-}$ pour le dépassement (overstaffing) et le manque de personnel (understaffing) permettent de contrôler le respect des quotas cibles Q_{dtl} .

Variables du problème

Le but du problème est la génération d'un planning réalisable pour chaque personne de telle sorte que l'ensemble de ces plannings satisfasse les contraintes de charges (quotas). La technique de génération de colonnes basée sur la méthode de branch and bound $B\&B$ permet de générer un grand nombre d'horaires réalisables pour chaque personne k notés par S. On note par S_{ks} le s ^{ème} horaire ou planning réalisable généré pour le personne k considéré. A chacun de ces S_{ks} est affecté un poids dépendant des préférences et de la contribution de cet horaire à la satisfaction des contraintes de charges (On note ce poids par P_{ks}). On définit de plus :

. Les variables auxiliaires suivantes :

$a_{dtksL} = \begin{cases} 1 \text{ si le quart } T_t \text{ est affecté le jour d dans l'horaire } s \text{ du la personne } k \\ 0 \qquad\qquad \text{si non} \end{cases}$

Les variables du problème Maître sont :

$x_{ks} = \begin{cases} 1 \text{ si l'horaire } s \text{ du la personne } k \text{ est conservé dans la solution optimale} \\ 0 \text{ si non} \end{cases}$

on note par e^{-}_{dtl} et e^{+}_{dtl} les variables d'écart pour chaque contrainte de quotas Q_{dtl} comptabilisé dans l'horaire générale.

Contraintes du problème Maître

Les contraintes intervenant dans ce problème sont les contraintes globales. Elles concernent les plannings (horaires) pris dans leur ensemble : ces contraintes stipulent que pour chaque jour, pour chaque période de travail, on doit satisfaire la contraintes de quota.

. Le respect des contraintes de quotas :

$\forall d \in D, \forall t \in T, \sum_{k}\sum_{s} a_{dtksL}.x_{ks} + e^{-}_{dtL} - e^{+}_{dtL} = Q_{dtL}$

A ces contraintes sont associées des variables duales notés par $\lambda_{dtl.}$

. Le respect de la fenêtre maximale de quota (qui donne les bornes sur les variables d'écart) :

$e^-_{dtl} \in \{0, ...Q_{dtl\max - deficit}\}$

$e^+_{dtl} \in \{0, ...Q_{dtl\max - surplus}\}$

. L'existence d'un et un seul horaire (planning) pour chaque personne :

$\forall k \in k, \sum_s x_{ks} = 1$

$x_{ks} \in \{0, 1\}\,; k \in K \quad s \in S$

2.2.2 Formulation du problème Maître

Le problème maître peut se formuler de la façon suivante :

$\min(\sum_k \sum_s p_{ks}.x_{ks} + \sum_L (p^+.e^+_L + p^-.e^-_L)$

$\forall d \in D, \forall t \in T, \sum_k \sum_s a_{dtksL}.x_{ks} + e^-_{dtL} - e^+_{dtL} = Q_{dtL}$

$s^-_{dtl} \in \{0, ...Q_{dtl\max - deficite}\}$

$e^+_{dtl} \in \{0, ...Q_{dtl\max - surplus}\}$

$\forall k \in k, \sum_k x_{ks} = 1$

$x_{ks} \in \{0, 1\}\,; k \in K \quad s \in S$

Le calcul du poids affecté à un horaire S_{ks} dans la fonction objectif du problème maître se fait de la façon suivante :

. Soit θ_i une constante qui définit l'ancienneté du personne k. Cette ancienneté a été normalisée par rapport à l'ancienneté la plus grande parmi tous les personnes considérés.

.la personne peut exprimer, pour chaque affectation potentielle, une préférence ou aversion notée par f_{jk} (on aura $f_{jk} > 0$ pour une préférence et $f_{jk} < 0$ pour une aversion). Si pour une affectation donnée aucune préférence n'est exprimée, on a alors $f_{jk} = 0$.

. Une distance sera introduite pour respecter les contraintes de ratios. Pour chaque type de quart de travail , on dispose d'une fourchette [$\min_t$, $\max_t$] dans laquelle on doit se situer. Pour le vérifier on calcule :

$ratio_T = \frac{nombre\ de\ quarts\ affectés\ det\ ype\ t}{nombre\ de\ quarts\ \text{de travaile}\ affectés\ de\ type\ t} * 100$

Dans le cas où l'on se situe en dehors de l'intervalle, on applique la pénalité suivante :

$$p_{ratio_T} = \begin{cases} 0 \text{ si } \min_t \leq ratio_T \leq \max_t \\ (1/100) * \mid ratio_T - M_t \mid si \text{ non ;avec } M_t = \begin{cases} \min_t si\ ratio_T \prec \min_t \\ \max_t \quad \text{si } ratio_T \succ \max_t \end{cases} \end{cases}$$

Le coût P_{ks} d'un horaire H_{ks} est calculé de la façon suivante :

$$P_{ks} = \sum_d \sum_t \theta_k * a_{dtksl}.(f_{dt} + g_{dt}) + \sum_T p_{ratio_T}$$

Pour plus de détails sur le calcul du poids P_{ks} d'un planning (horaire) donné, nous renvoyons le lecteur à [[38]

2.3 Problème auxiliaire

En programmation linéaire, les coefficients de la fonction objectif finale représente le coût réduit des plannings. Chaque colonne du problème maître ramifié (restreint), qui n'est pas dans la base, a un coût réduit dont la valeur peut être calculée par la formule suivante :

$C_{ks} = P_{ks} - \mu_{kL} - \sum_d \sum_t a_{dtksL} * \lambda_{dtL}$ *telle* que

λ_{dtL} :Sont des variables duales associées aux contraintes de type

$\forall d \in D, \forall t \in T, \sum_k \sum_s a_{dtksL} * x_{ks} + e^-_{dtL} - e^+_{dtL} = Q_{dtL}$

μ_{kL} :sont des variables duales associées aux contraintes de type $\forall k \in k, \sum_k x_{ks} = 1$

Du fait que chaque colonne du problème maître correspond à un planning réalisable pour une personne k, si nous voulons trouver une colonne qui va entrer dans la base de ce problème et dont le coût réduit est le plus petit, nous devons résoudre un problème auxiliaire pour chaque plage horaire qui vise à trouver un planning réalisable dont le coût réduit est le plus petit. Cela consiste à résoudre le problème auxiliaire en décidant des valeurs de telle sorte que C_{ks} soit le plus petit possible.

Avant de décrire le modèle mathématique du problème, nous commençons par une description des données ce qui permettra de définir les contraintes intervenant dans ce problème (contraintes individuelles exprimées pour chaque personne) et la modélisation retenue pour la résolution.

2.3.1 Contraintes

On distingue deux types de contraintes : Contraintes dures et contraintes souples.

Les contraintes dures doivent -être satisfaites obligatoirement pour obtenir un horaire réalisable. Ces contraintes sont en générale :

1- La liste des quarts de travail que peut effectuer la personne k.

2- Une liste d'affection obligatoires ou interdites qui doivent être prises en compte dans tous les horaires potentiels générés pour la personne k.

3- La charge de travail que doit recevoir la personne sur des périodes données. Cette charge sera exprimée en heures.

4- Les règles d'enchaînement des différents types de quart de travail. En effet chaque quart de travail est caractérisé par un type (jour, soir, nuit et/ou repos) et des contraintes doivent – être respectées pour l'enchaînement de ces différents types de quart de travail dans l'horaire à produire pour la personne considéré.

Les contraintes souples peuvent être ou non satisfaites. Elles n'interviendront que sous forme de pénalités introduites dans le coût d'un horaire. Ces contraintes sont :

1- La contrainte contrôlant les ratios entre les différents types de quart.

En effet il est préférable que l'horaire pris dans sa globalité soit relativement balancé en terme de types de quarts (jours, nuit, soir et/ou repos)

2- Les préférences et les aversions qui sont exprimées par la personne. Celles-ci peuvent – être positives ou négatives quant aux affectations qui lui seront attribuées pour l'horizon courant.

2.3.2 Modélisation du problème auxiliaire

Selon les variables duales λ_{dtL} et μ_{kL} de l'itération courante de GC, le problème auxiliaire a la tâche de générer des horaires de cout réduit négatif CR^{-}. Afin d'accomplir efficacement cette tâche, nous modélisons le problème en utilisant l'approche graphique, ensuite deux algorithmes $APA1$ et $APA2$ sont utilisés et qui consistent respectivement en définition de graphe d'état (vecteur de ressources) et la génération des horaires de cout réduit négatif CR^{-}.

Construction du graphe

Il s'agit de construire un graphe acyclique $G_k = (V_k, E_k)$ ($Figure 2.1$) composé d'un ensemble de nouds $\mathrm{v}_k = \{v_0 ... v_n\}$ et d'un ensemble d'arcs $E_k \subseteq \{(v_p, v_l) v_p, v_l \in v_l v_p \neq v_l\}$

pour chaque personne k dont le parcours est contrôlé par des vecteurs de ressource. Les nœuds servent à la mise à jour des valeurs de ressources et les arcs servent à autoriser ou à refuser un passage suivant la valeur courante de la ressource. Ce graphe a un nœud source v_0 et un nœud puits v_n.

Les règles de base retenue par cette construction sont :

. Un nœud v_l représente pour la personne considéré une affectation potentielle d'un quart de travail t pour une journée précise d. Cette affectation est définie par le triplet

(d, t, L) et elle est en bijection avec les variables a_{dtksL} d'un horaire s. Un tel nœud est crée si la personne k a au moins une compétence qui lui permet d'avoir un apport, grâce à l'affectation (d, t), dans la contrainte de quotas Q_L.

Un arc (v_{l1}, v_{l2}) sera présent dans le graphe s_k si la personne peut réaliser l'affectation A_{d2t2} après avoir réalisé l'affectation A_{d1t1}. Là encore ce sont les contraintes dures qui contrôlent la construction des arcs.

Définition des nœuds

Un nœud v_l représente une affectation possible pour une personne considéré, notée par A_{dt}. L'ensemble de nœud V_k est défini selon les contraintes intrinsèques du personne k. En effet, une affectation A_{dt} est associée à un nœud $v \in v_k$ si et seulement si, la personne k peut avoir cette affectation. Dans le cas contraire, aucun nœud n'est crée pour celle-ci.

Les règles utilisées pour décider de l'existence d'un nœud donné sont :

- Le jour d doit appartenir à l'horizon de planification,

- Le quart de travail t doit appartenir à la liste des quarts réalisable pour la personne considéré.

- L'affectation A_{dt} ne doit pas appartenir à la liste des affectations interdites pour la personne considérée.

Si ces trois règles sont satisfaites alors le nœud v_l qui représente l'affectation A_{dt} appartiendra au graphe correspondant au problème auxiliaire pour la personne k considéré.

Pour le nœud source du graphe, il existe deux possibilités. Si on dispose de l'horaire réalisé pendant la période précédant l'horizon de planification courant, la source représente la dernière affectation de cet horizon. Sinon les valeurs initiales seront égales à 0.

Construction des arcs

Il existe un *arc* dans le graphe entre deux sommets v_{l1} (affectation A_{d1t1}) et v_{l2} (affectation A_{d2t2}) si l'enchaînement des affectations représentées par les deux sommets est possible par rapport aux contraintes dures du personne considérée. Les règles utilisées pour décider de l'existence d'un $arc(v_{l1}, v_{l2})$ sont :

- Le jour d_1 doit être différent du jour d_2. En effet la convention collective des personnes interdit de recevoir deux affectations le même jour d.

- Les règles d'enchaînement des quarts de travail doivent être respectées. Par exemple si d_2 est le lendemain de d_1 alors T_{td2} doit appartenir à la liste des quarts de travail réalisables le lendemain du quart de travail T_{td1} . Si c'est le contraire, alors doit appartenir à des quarts de travail réalisable après le quart de travail T_{td1}.

- Si d_2 n'est pas le lendemain de d_1 alors le nombre de jours situés entre d_1 et d_2 doit –être compris entre le nombre minimum et maximum des jours consécutifs non travaillés.

- S'il y a des fins de semaines complètes non travaillées entre d_1 et d_2, le nombre de ces fins de semaine doit être compris entre le nombre minimum et le nombre maximum de fins de semaine consécutives non travaillées que la personne peut recevoir.

Remarques

Si on ne dispose d'aucune information concernant l'horizon précédent, on crée un nœud source virtuel dans le graphe. Dans ce cas les règles faisant intervenir le jour d_1 sont modifiées. A la place du jour d_1 on substitue dans ces règles la veille du premier jour de l'horizon courant.

. Pour le puits, les règles faisant intervenir le jour d_2 sont modifiées. A la place du jour d_2 , on utilise le lendemain du dernier jour de l'horizon courant pour vérifier la valeur des différentes règles.

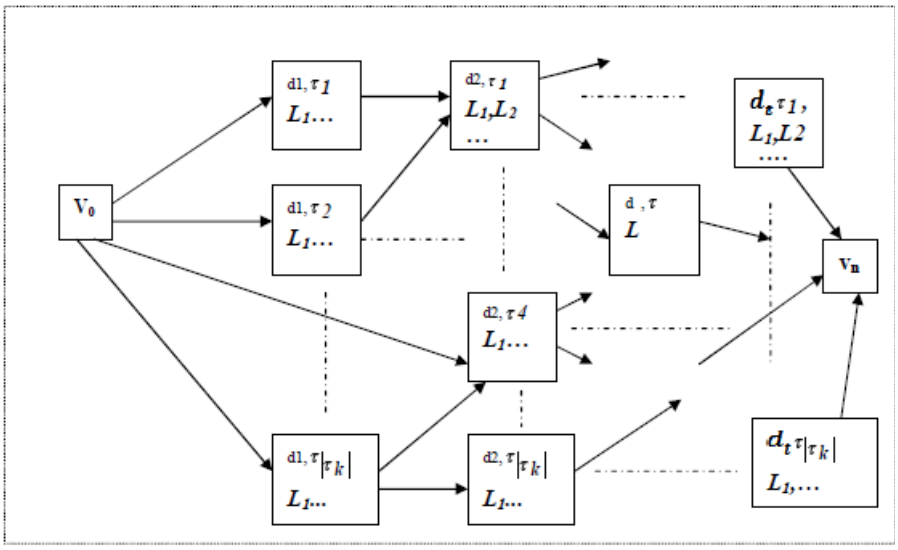


Figure 2.1 Graphe $G_k = (V_k, E_k)$

Description du graphe d'état $\hat{G_k}$

Pour définir le graphe d'état $\hat{G_k}$ à partir du grapheG_k , nous appliquons l'algorithme $APA1$. Cet $APA1$ consiste à calculer pour chaque nœud $v_l \in \hat{v_k}$ l'ensemble de ses états réalisables D_l. Par la suite, pour chaque état x_l du nœud $v_l \in \hat{v_k}$ un ensemble de vecteurs de ressources $R(x_l)$ est défini. Tout état $x_h \in R(x_L)$ est un état réalisable au nœud correspondant $v_h \in v_k$, où $(v_h, v_l)\hat{G_k}$ et $v_h \in D_h$. La dimension d'un état x_l (vecteur de ressource) est égale au nombre de contraintes dures (intrinsèques) associées aux personnes. Autrement dit, chaque ressource correspond à une contrainte dure. Un état x_l d'un nœud $v_l \in v_k$ représente le résultat de chacune de ces ressources obtenu à ce nœud après avoir parcouru un sous chemin $v_0...v_l$.

Dans ce graphe d'état, un nœud ne représente plus une affectation A_{dt}, mais un état réalisable x_l. De plus, un arc ne représente plus la transition entre une paire d'affectation (A_{d1t1}, A_{d2t2}), mais une transition entre deux états réalisables. Un tel arc existe si et seulement si, il est possible de transiter entre ces deux états.

A la fin de l'exécution de l'algorithme, l'ensemble $R(x)$ contient tout les états réalisables à partir desquels l'état x est né. Par conséquent, il est alors possible de définir l'ensemble des

chemins de v_0 ...$_{v_n}$ (donc l'ensemble des horaires $s \in H_k$) grâce à l'ensemble des états réalisables finaux D_n et aux ensemble de $R(x_n)$ de chaque état $x_n \in D_n$. Cet ensemble d'états réalisables correspond à celui de nœud puits v_n.

Finalement, le graphe d'état $\hat{G_k}$ est défini par l'ensemble d'états D_n de tous les nœuds $v_l \in \hat{v_k}$ et par les ensembles d'états $R(x_l)$ associés à tous les états réalisables $x_l \in D_l$.

Pour la description complète de l'algorithme $APA1$ et la modélisation des ressources dans le graphe G_k, nous renvoyons le lecteur à[38] et [25].

Figure 2.2 Graphe $\hat{G_k} = (\hat{v_k}, \hat{E_k})$

Soit un personne . Le graphe d'état $\hat{G_k}$ associé à chaque k est définie par :

- L'ensemble des noeuds $\hat{v}_k = \{v_d / v_d \in V_k\}$
- L'ensemble des arcs $\hat{E_k} = \{(v_l, v_h)$ / le nœud $v_l \in V_k$ et le nœud $v_h \in V_k$ ont respectivement les états $x_l \in D_k$ *et* $x_h \in D_h$

Le coût d'un arc d'état $(v_l, v_h) \in \hat{E_k}$ associé au graphe d'état est identique à celui de l'arc correspondant $(v_l, v_h) \in E_k$ du graphe G_k. On note ce coût par C_{lh}.

Soit Le graphe d'état $\hat{G_k}$ associé à chaque personne $k \in K$.

Le sous chemin d'état noté par $v_0...^p...v_h$ où $v_0 \in V_k$ et $v_h \in V_k$ est définie comme suit :

pour $k = 1, 2.... \mid p \mid -1, p = v_1^{'}, v_2^{'}, ...v_{\mid p \mid}^{'}$ $v_1^{'} = v_0, v_{\mid p \mid}^{'} = v_h$ et $(v_k^{'}, v_{k+1}^{'}) \in \hat{E_k}$ Le coût d'un sous chemin d'état $C(p)$ est donné par la somme des coûts de ses arcs.

$$C(p) = \sum_{d=1}^{|p|-1} C_{d(d+1)}$$

Description du graphe d'état G_k^{cu}

A chaque itération courante de GC,l'algorithme $APA2$ permet de générer des horaires *de* CR^- suivant les variables duales λ_{dtL} et μ_k associées aux contraintes de quotas et aux contraintes de partition. Pour tenir compte de ces variables duales, des prétraitements doivent-être effectués sur le graphe d'état $\hat{G_k} = (\hat{v_k}, \hat{E_k})$ afin d'obtenir un nouveau graphe d'état G_k^{cu} qui sera utilisé par $APA2$ pour déterminer les horaires. Ces prétraitements consistent à redéfinir les coûts C_{lh} associés aux arcs $(v_l, v_h) \in \hat{E_k}$ du graphe d'état $\hat{G_k}$, où $v_l = (d_{l,T_1})$, $v_h = (d_{h,T_h})$ et $v_h \neq v_n$(v_n est le nœud puits) de la manière suivantes :

Le premier traitement consiste à inclure toutes les affectations obligatoires (d,t) et à les exclure de chaque chemin $v_0...v_n$ obtenu par l'algorithme $APA2$ (G_k^{cu}). Pour obtenir de tel chemin, il suffit de redéfinir le cout C_{lh} d'un arc $(v_l, v_{h)} \in \hat{E}_k$:

$$C_{lh}\begin{cases} +\infty \text{ si } v_h \neq v_n \text{ et si } \exists v' \in V_k \text{ tq : } v'=(d',t') \text{ est une affectation obligatoire } (\mathrm{d}',\mathrm{t}') \neq (\mathrm{d}_h,\mathrm{t}_h) \text{ et } \mathrm{d}_l \\ +\infty \text{ si } v_h = v_n \text{ et si } \exists v' \in V_k \text{ tq : } \mathrm{v}'=(\mathrm{d}',\mathrm{t}') \text{ est une affectation obligatoire et } \mathrm{d}_l \prec d \\ +\infty \text{ si } v_h \neq v_n \text{ et si } (d_h,t_h) \text{est une affectation non obligatoire} \\ C_{lh} \text{ si non} \end{cases}$$

Pour un nœud v_h d'état x_h, on note par $P_k^{cu}(v_h) \subseteq \{p : v_0..^p..._v_n\}$l'ensemble des sous chemin $v_0..^p..._v_n$ réalisable. Ainsi en considérant tous les puits $v_n \in V_k$, l'ensemble des chemins $\mathrm{v}_0\_\_\mathrm{v}_n$ réalisables est noté par $P_k^{cu} \cup_{v_n \in V_k} P_k^{cu}$ et à chacun des chemins $p \in P_k^{cu}$ correspond un unique horaire s$\in H_{ks}(\mid P_k^{cu} \mid = \mid H_k^{cu} \mid)$ et nous avons :

$$\forall s \in H_k^{cu} \left\{ \begin{array}{c} \forall (d,t) \text{ affectation interdite, } a_{ksdtl} = 0 \\ \forall (d,t) \text{ affectation obligatoire, } a_{ksdtl} = 1 \end{array} \right\}$$

Un deuxième prétraitement est appliqué pour redéfinir encore une fois le cout c_{lh} associés aux arcs $(v_l, v_{h)} \in \hat{E}_k$ pour que le coût $C(p)$ appliqué à un chemin d'état $v_0\_^p\_v_n$ ( ) correspond au coût réduit C_{ks} de l'horaire $s \in H_k^{cu}$ associé à ce chemin. Ainsi pour chaque arc dont l'extrémité initiale est un nœud source : (v_0, v_h), la variable duale $\mu_k$est incluse dans la redéfinition du coût C_{lh}. Aussi les variables duales λ_L des contraintes de quotas sont aussi incluses. Enfin le coût C_{lh} de chaque arc$(v_l, v_{h)} \in \hat{E}_k$ reste inchangé car aucune affectation n'est associé au nœud puits $v_N \in V_k$. Les coûts C_{lh} associés aux arcs$(v_l, v_{h)} \in \hat{E}_k$, où $\mathrm{v}_l = (d_l, t_l)$, $\mathrm{v}_h = (d_h, t_h)$ et $v_h \neq v_n$(v_n est le nœud puits) sont calculés comme suit :

$$C_{lh} = \begin{cases} C_{lh} - \mu_k - \sum\sum(a_{ksdhT_hL}, \lambda_L) : \text{ si } \mathrm{v}_l = v_0 \\ C_{lh} - \sum\sum(a_{ksdhT_hL}, \lambda_L) \text{si } \mathrm{v}_l \neq v_0 \text{ et si } \mathrm{v}_h \neq v_n \\ C_{lh} \text{ si } \mathrm{v}_h = v_n \end{cases}$$

L.Algorithme APA2

Une fois le nouveau graphe d'état G_k^{cu}est obtenu, l'algorithme $APA2$ est appliqué sur ce dernier afin de générer des colonnes de coûts réduits négatifs hors base notés par CR^-, soit des coûts négatifs réduits C_{ks}. L'ensemble des colonnes ainsi obtenu est noté par$\mathrm{PA}(G_k^{cu}) \subseteq \{s \in H_k^{cu}, c_{ks} \prec 0\}$:

Pour un nœud $v_h \in \hat{v}_k$d'état x_h, on note par :

-$C(v_h)$ le coût optimal de tous les sous chemins $v_0 \_^{p} \_ v_h$ *t*elle que :

$C(v_h) = Min\{C(p) : p \in P_k^{cu}(v_h)\}$

$p(v_h) \in \hat{v}_k$ le nœud d'état prédécesseur de v_h dans le sous cheminp$'$ $P_k^{cu}(v_h)$ qui est de coût optimal $C(v_h)$.

*L'algorithme APA*2 permet de calculer pour chaque nœud $v_h \in \hat{v}_k$ les valeurs $c(v_h)$ et $p(v_h)$. Ces valeurs sont définies par [7]] :

$$c(v_h) = \left\{ \begin{array}{c} 0 \text{ si } v_h = v_0 \\ \min\left\{c(v_l) + c_{lh}, v_l \in v_k, (v_l, v_h) \in \hat{E}_k\right\} \end{array} \right\}$$

$$p(v_h) = \left\{ \begin{array}{c} 0 \text{ si } v_h = v_0 \\ Arg\min\left\{c(v_l) + c_{lh}, v_l \in v_k, (v_l, v_h) \in \hat{E}_k\right\} \end{array} \right\}$$

Définition 39 *Soit le graphe G_k^{cu} d'état , de nœud puits $v_n \in v_k$. Le chemin optimal $v_0 \_^{p} \_ \_ v_n$, où $p \in P_k^{cu}(v_h)$ de coût optimal $c(v_n)$ est définie par :*

pour z=1,2...$\mid p \mid -1, p = v_1^{'}, v_2^{'}, ...v_{\mid p \mid}^{'}$ et $v_z^{'} = p(v_{z+1}^{'})$.

L'horaire $s \in H_k^{cu}$ associe au chemin optimal de ce puits est retourné par l'algorithme $APA2$ si et seulement son coût C_{ks} est négatif (rappelons que, le coût réduit C_{ks} est donné par $C(v_n)$).

L.Algorithme $APA2$

-soit G_k^{cu}=($\hat{v}_k$, $\hat{E}_k$)) le graphe d'état

-v_0 nœud source, v_n nœud puits

-$c(v_0)$ (le nœud v_0 d'état x_0)

- pour tout $v_0 \in \hat{v}_k$d'état faire $x_0^{'}$ $faire$

. $c(v_0) \longleftarrow +\infty$

- pour $v_h = v_1, v_2, \ldots v_n$ faire

- pour tout $v_h \in \hat{v}_k$ faire

- $c(v_h) \longleftarrow +\infty$

Pour tout $v_l \in \hat{v}_k$ tel que $(v_l, v_h) \in \hat{E}_k$ faire

Si $C(v_l) + C_{lh} \prec C(v_h)$ alors

$C(v_h) \longleftarrow C(v_l) + C_{lh}$

$p(v_h) \longleftarrow v_l$

pour tout puits $v_n \in \hat{v}_k$ tel que $c(v_n) \prec 0$ faire

-Construire ou définir le chemin $p \in P_k^{cu}(v_n)$ qui est de coût optimal $C(v_n)$ (voir définition)

. Ajouter s à l'ensemble PA, où s est l'unique horaire associé au chemin p

Chapitre 3

Méthode de résolution

Ce chapitre explique la méthode de résolution utilisée pour résoudre le problème. Les algorithmes les plus fréquemment utilisés pour résoudre les problèmes de programmation en nombres entière sont des algorithmes de séparation et évaluation progréssive appelé Branch and Price.

Le problème à résoudre comporte plusieurs difficultés dont trois sont primordiales. Premièrement, il existe un nombre exponentiel d'horaires individuels (colonnes) pour chaque personne . Ensuite, un horaire général est réalisable si et seulement si un seul horaire (individuel) est affecté à chaque personne et si l'ensemble de ces horaires respecte les contraintes de quotas. Troisièmement, parmi tous les horaires généraux réalisables, il faut trouver celui qui est de valeur optimale.

3.1 Branch and Price

La technique de branch and price est un algorithme qui consiste à parcourir l'ensemble de solutions (entières) du $PLNE$ (P) de façon intelligente et non explicite. Ceci se fait en partitionnant cet ensemble, qui est noté par S , en plusieurs sous-ensembles à travers un arbre de recherche de solution.[**?**]. Dans cet arbre, chaque noeud u correspond à un $PLNE(P^u)$, où l'ensemble des solutions réalisables de ce même problème est noté par $S^u \subseteq S$. Si nécessaire, une méthode de séparation sera appliquée au problème (P^u) afin de créer deux problèmes fils notés (P^v) et (P^w). Cette méthode de séparation consiste à ajouter une contrainte de branchement

différente à chacun de ces $PLNE$ afin de partitionner l'ensemble de solutions S^u du problème (P^u) en deux sous-ensembles disjoints : S^v et S^w. À chaque noeud u de l'arbre, la relaxation continue (R^u) du $PLNE$ (P^u) est résolue afin d'obtenir une borne inférieure sur la valeur de toutes les solutions de ce problème en nombres entiers. Le problème (R^u) est identique au problème (P^u),mais contrairement à ce dernier qui n'accépte que des solutions avec des variables de valeur entière, le problème (R^u) permet à ces mêmes variables des valeurs réels. Par conséquent, le problème (P^u) étant plus contraint que sa relaxation (R^u), la valeur optimale z_u^* de cette dernière est une borne inférieure sur la valeur de toutes les solutions de l'ensemble S^u. D'ailleurs, lorsque la solution x_u^n , qui est obtenue aprés la résolution du problème (R^u) et qui est de valeur optimale z_u^*, est entière, le problème (P^u) est résolu . Dans ce cas, il est possible de mettre à jour la valeur de la meilleure solution du problème (P) actuellement connue, qu'on note par z^*. Cette valeur est aussi une borne supérieure sur la valeur optimale de ce même problème.

Voici les principales composantes de l'algorithme du branch and price :

– méthode de génération de colonnes,

– méthode de pricing (résolution du problème auxiliaire),

– méthodes de séparation,

– algorithmes d'exploration de l'arbre de recherche,

– conditions pour couper un noeud,

Premièrement, la méthode de génération de colonnes est utilisée pour résoudre le problème (R^u) au noeud u. Cette méthode marie l'algorithme du Simplexe avec une méhode nommée Pricing qui a la tâche de générer de nouvelles colonnes de coût réduit négatif CR^-. Ensuite, lorsque la solution ainsi obtenue est fractionnaire, une méthode de séparation est appliquée au problème (P^u) et consiste à partitionner son ensemble de solutions (S^u) en deux sous-ensembles disjoints. Ceci a pour conséquence d'éliminer la réalisabilité de la solution fractionnaire x_u^n pour les deux nouveaux problèmes(P^v) et (P^w), qui sont les fils de (P^u), et possiblement d'atteindre une solution entière dans ces problèmes ou plus profondément dans l'arbre de recherche. Troisièmement,on applique l'un des deux algorithmes pour l'exploration de l'arbre de recherche :

1) recherche en profondeur d'abord DFS

2) recherche avec le meilleur des deux fils d'abord $BSFS$

Enfin, aprés avoir résolu le problème (R^u), si sa valeur optimale z_u^* est telle que $\lceil z_u^* \rceil \geq z^*$, il est alors inutile de résoudre le problème (P^u) en partitionnant son ensemble de solution S^u. Cette condition pour couper un noeud u sera développée au cours de ce chapitre.

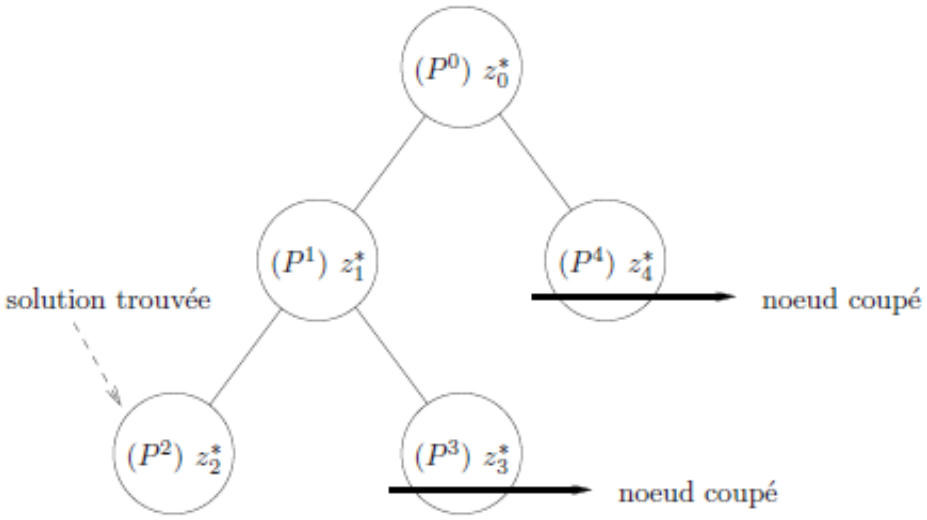
Plusieurs stratégies peuvent être ajoutées à l'algorithme du branch and price pour diminuer son temps de traitement. La première est la stratégie du branchement prématuré. Elle consiste à arrêter prématurément l'optimisation du problème (R^u) au noeud u lorsque nous considérons qu'il est inutile et (ou) trop long de connaître sa valeur optimale z_u^*. La deuxième stratégie est la méthode d'arrondi. Elle consiste à trouver une solution entière, donc un horaire général et réalisable, à l'aide des variables déjà existantes. En plus de celles-ci, d'autres stratégies sont présentées par Desaulniers [11].

Nous pouvons voir à la figure 3.1 un exemple simple d'arbre de recherche. On peut supposer que les problèmes (P^i) ont été explorés par ordre croissant de leur indice i et que chaque z_i^*, qui est la valeur optimale de la relaxation continue (R^i), a été atteinte. Aprés la résolution du problème (R^2), on obtient la première solution entière x_2^n de valeur z_2^* Ensuite, nous obtenons au problème (R^3) la solution fractionnaire x_3^n de valeur z_3^*, où $\lceil z_3^* \rceil \geq z_2^*$. Dans ce cas, remarquons que $\lceil z_3^* \rceil$ est une borne inférieure sur la valeur de toutes solutions du problème (P^3) et que z_2^* est une borne supérieure sur la valeur optimale du problème (P). Les mêmes remarques s'appliquent au problème (P^4), où $\lceil z_4^* \rceil \geq z_2^*$. C'est pour ces raisons que les deux problèmes (P^3)et (P^4) ont été coupés.

Finalement, dans ce chapitre nous allons présenter en détail chaque composante de l'algorithme du branch and price. Nous commençons par décrire la relaxation continue

(R) du problème (P). Deuxièmement, nous allons expliquer la méthode de génération de colonnes, et la méthode Pricing. Ensuite, deux méthodes de séparation possibles seront présentées : règle de branchement binaire simple et règle de branchement Ryan-Foster.[35] Quatrièmement, les deux parcours possibles de l'arbre de recherche seront expliqués : le DFS et le $BSFS$. Finalement, nous expliquerons pourquoi notre algorithme de branch and price est exhaustif, c'est à dire qu'il calculera la valeur optimale z^* du problème(P).

figure 3.1 Exemple d'arbre de recherche.



3.1.1 Présentation du problème (R)

Comme nous l'avons mentionné à la section précédente, à chaque problème (P^u) de l'arbre de recherche correspond une relaxation continue notée (R^u) . Dans cette relaxation, les contraintes d'intégralité sur les valeurs des variables x_{ks} , s_L^- et s_L^+ ont été assouplies afin de leur permettre des valeurs réelles. De façon plus précise, voici les modifications qui sont apportées dans la formulation du problème (P) afin de définir la relaxation continue (R) :

– $x_{ks} \in \{0,1\}$est remplacé par $0 \leq x_{ks} \leq 1$,

– $e_L^- \in \{0,1,...,Q_L^-\}$ est remplacé par $0 \leq e_L^- \leq Q_L^-$,

–$e_L^+ \in \{0,1,...,Q_L^+\}$ est remplacé par $0 \leq e_L^+ \leq Q_L^+$.

Ces modifications sont les mêmes à apporter au $PLNE(P^u)$ afin de définir sa relaxation continue (R^u).Donc le probléme Maitre devient .

$$min(\sum_{k\in K}\sum_{s\in Sk} p_{ks}x_{ks} + \sum_{(d,t)\in D_T}\sum_{L\in L_{dt}}(p^-e_L^- + p^+e_L^+))$$

$s.l.c.$:

$$\sum_{k\in K}\sum_{s\in S_k} a_{ksdt}x_{ks} + e_L^- - e_L^+ = Q_L \qquad (d,t) \in D_T, L \in L_{dt}$$

$$\sum_{s\in S_k} x_{ks} = 1 \qquad k \in K$$

$$0 \leq x_{ks} \leq 1 \qquad k \in K, s \in S_k$$

$0 \leq e_L^- \leq Q_L^-$ $\qquad (d,t) \in D_T, L \in L_{dt}$

$0 \leq e_L^+ \leq Q_L^+$ $\qquad (d,t) \in D_T, L \in L_{dt}$

De plus, la variable duale associée à une contrainte de quotas (Q_L), où $L \in L_{dt}$ et $(d,t) \in D_T$, et celle associèe à la contrainte de partionnement d'un personne $k \in K$

sont respectivement notées λ_L et μ_k.

3.1.2 Méthode de génération de colonnes GC

Historique de la méthode de génération de colonnes

En 1958, Ford et Fulkerson [27] suggérèrent de résoudre un problème de multi-flots sans en énumérer explicitement toutes les variables. En 1960, Dantzig et Wolf[17] développèrent cette idée en décrivant un processus ajoutant au fur et à mesure les colonnes nécessaires à la résolution d'un programme linéaire. Enfin, en 1961, cette technique que l'on allait appeler génération de colonnes était utilisée pour la première fois par Gilmore et Gomory [33][34]pour résoudre un problème de découpe (Cutting Stock Problem), qui reste le problème de référence de cette technique. La génération de colonnes a depuis permis de résoudre un grand nombre de programmes linéaires comportant un nombre important de variables. L'extension de la technique à la résolution des programmes linéaires en nombres entiers, initiée par Desrosiers et al. [23] qui combinèrent la technique de génération de colonnes à un processus de branch-and-bound, a encore accru la popularité de cette méthode qui a depuis été employée avec succès dans un grand nombre de champs d'application : problèmes de routage, de découpes, de coloration de graphes, de détermination de plannings, etc. (voir Lübbecke et Desrosiers[**?**] ou Wilhelm [23] pour des revues récentes des applications de la génération de colonnes). D'une manière générale, la méthode de la génération de colonnes est d'ailleurs très efficace pour résoudre des programmes linéaires en nombres entiers.

Principe de la génération de colonnes

La génération de colonnes est initialement une méthode de résolution des programmes linéaires de grande taille. Si le programme linéaire admet une solution optimale, il en existe une dans laquelle un grand nombre de variables ont une valeur nulle (les variables hors base de l'algorithme du simplexe). Le principe général de la génération de colonnes est alors de ne pas

énumérer explicitement toutes les variables, mais de travailler à partir de leurs coûts réduits pour ne garder que des variables potentiellement intéressantes. Ainsi, lorsque la méthode du simplexe recherche à chaque itération une variable de coût réduit négatif pour la faire entrer dans la base, ce qui améliorera la valeur de la fonction objectif, la génération de colonnes va de la même manière chercher des variables de coût réduit négatif à chaque itération, mais sans énumérer explicitement ces variables.

Résolution de programmes linéaires

On cherche à résoudre un programme linéaire appelé problème maître (Master Problem), on résout tout d'abord un programme linéaire obtenu en ne considérant qu'une restriction des variables, programme appelé problème maître restreint (Restricted Master Problem ou RMP). Cette résolution permet d'obtenir les valeurs prises à l'optimal par les variables du problème dual du problème maître restreint. Ces valeurs associées à chaque contrainte du problème maître restreint sont appelées valeurs duales. Un sous problème nommé sous problème de tarification (pricing subproblem) ou oracle va alors chercher à partir de ces valeurs duales, une ou plusieurs variables qui ne sont pas encore dans le sous ensemble de variables considérées et qui ont un coût réduit négatif. Ces variables sont ajoutées au RMP qui est résolu à nouveau. C'est un processus itératif qui continue tant que le sous problème trouve des nouvelles variables à coût réduit négatif. Lorsqu'il n'en trouve aucune nouvelle, la dernière solution optimale du RMP est la solution optimale du problème maître. La génération de colonnes menée jusqu'à son terme est donc une méthode exacte.

Résolution de programmes linéaires en nombres entiers

La méthode peut être combinée à un processus de branch-and-bound pour résoudre un programme linéaire en nombres entiers. La méthode branch-and-bound classique consiste à résoudre une succession de programmes linéaires obtenus en ajoutant progressivement des contraintes au problème initial de manière à obtenir une solution comportant de moins en moins de valeurs non entières. Chaque sous programme linéaire peut alors être résolu par génération de colonnes en prenant soin de bien effectuer à chaque noeud de l'arbre de branch and- bound le processus de génération de colonnes de manière à ne pas oublier de variables potentiellement intéressantes.

Dans l'algorithme de branch and price, la méthode de génération de colonnes est utilisée pour résoudre la relaxation continue (R^u) du problème (P^u) au noeud u de l'arbre de recherche, et devra trouver une base primale-duale réalisable .

Encore une fois, nous devons introduire de nouvelles notations. Soit le $PLNE$ (P^u) du noeud u , un personne $k \in K$, le problème restreint $(\dot{R^{ui}})$ de l'itération i de GC et une $RFRC$ $(k, c) \in K_{I_k^u}$.[9] Notons par

– $\mathrm{S}_k^u \subseteq S_k$, l'ensemble des horaires de la personne k qui sont réalisables au $PLNE$ (P^u) ;

– $\dot{S_k^{ui}} \subseteq \mathrm{S}_k^u$, l'ensemble des horaires de la personne k qui sont inclus dans le problème restreint $(\dot{R^{ui}})$

– $\mathrm{S}_k^{cu} \subseteq \mathrm{S}_k^u$, l'ensemble des horaires de la personne k qui respecte la $RFRC$ (k, c) du $PLNE$ (P^u).

Soit un horaire $s \in S_k^u$. Cet horaire est tel que $s \in \dot{S_k^{ui}}$ si et seulement s'il a été généré par le problème auxiliaire à une itération antérieure à l'itération i ou à un $PLNE$ antérieur au $PLNE$ (P^u).

Description de l'algorithme

Essentiellement, la méthode GC utilise un algorithme itératif . Nous ajoutons une fonction qui apparemment très nécessaire pour accélérer la convergence de la solution a ce GC. Cette fonction permet de sélectionner les personnes auxquelles on souhaite déterminer leurs horaires sans difficultés, c'est à dire les personnes ayant un nombre minimale de contraintes de préferences. Puis à chaque itération i, le problème restreint (R^{ui}) est résolu par la méthode du Simplexe. Le résultat obtenu est une base, qui est primale-duale réalisable à $(\dot{R^{ui}})$, notée $\dot{x_u^i}$ et de valeur optimale $\dot{z_u^i}$. Ensuite, la méthode Pricing est appelée pour générer des colonnes de cout réduit négatif hors base noté par CR^- , soit des variables de coûts réduits négatifs qui ne sont pas déjà dans le problème restreint $(\dot{R^{ui}})$. L'ensemble de colonnes ainsi obtenu est noté $PA(\hat{G}_{ki}^{cu}) \subseteq \{s \in S_k^{cu} : s \notin \dot{S_k^{ui}} \text{ et } c_{ks} < 0\}$,où $(k, c) \in K_{I_k^u}$

Ensuite, si $PA(\hat{G}_{ki}^{cu}) \neq \emptyset$, $alorsla\ solution\ \dot{x_u^i}$ n ést donc pas primale-duale réalisable pour $(\dot{R^u})$ et l'algorithme continue. L'ensemble de colonnes $PA(\hat{G}_{ki}^{cu})$ de CR^- est ajouté au problème (R^{ui}) afin de définir le nouveau problème restreint :$(\dot{R^{u(i+1)}}) \leftarrow (\dot{R^{ui}}) \oplus PA(\hat{G}_{ki}^{cu})$.

En d'autres termes :$\dot{S_k^{u(i+1)}} \leftarrow \dot{S_k^{ui}} \oplus PA(\hat{G}_{ki}^{cu})$. Une nouvelle itération est alors effectuée

pour le traitement de ce nouveau problème restreint.

Enfin, si $PA(\hat{G}^{cu}_{ki}) = \emptyset$, cela signifie que la condition d'optimalité $\forall (k,c) \in K_{I^u_k}$, $PA(\hat{G}^{cu}_{ki}). = \emptyset$ est vérifiée. Autrement dit, il n'existe pas de colonnes CR^- hors bases, alors x^i_u est une base primale-duale réalisable pour le problème restreint (R^{ui}) mais aussi pour le problème (R^u). Donc, $\dot{z^i_u} = \dot{z^*_u}$ est la valeur optimale de ce dernier problème et GC est terminé

Finalement, les ensembles de variables $PA(\hat{G}^{cu}_{ki})$ et la notation $(k,c) \in K_{I^u_k}$ seron dans l'algorithme de Pricing. Au même moment, le fondement de la précédente condition, qui est nécessaire et suffisante à l'optimalité d'une solution, sera expliqué.

Observations importantes

Ainsi, en considérant les itérations 1 à n de GC modifiée, la série de n triplets suivants est générée $[((\dot{R^{u1}}), \dot{x^1_u}, \dot{z^1_u}), ((\dot{R^2_u}), \dot{x^2_u}, \dot{z^2_u}), ..., ((\dot{R^n_u}), \dot{x^n_u}, \dot{z^n_u})]$. Plusieurs observations sont importantes

1. chacune des solutions $(\dot{x^i_u}, \dot{z^i_u})$ est primale-duale réalisable au problème restreint $(\dot{R^{ui}})$.

2. pour chacune des itérations i, la valeur de la fonction objectif est au moins aussi bonne que celle de l'itération précédente : $\dot{z^i_u} \leq \dot{z^{i-1}_u}$. Et s'il n'y a pas de dégénérescence, nous avons $\dot{z^i_u} < \dot{z^{i-1}_u}$.

3. considérons une paire d'itérations successives$(i-1, i)$, où $\dot{z^i_u} \prec \dot{z^{i-1}_u}$. Alors, la solution primale-duale ($\dot{x^{i-1}_u}$, $\dot{z^{i-1}_u}$)du problème $(\dot{R^{(i-1)}_u})$ n'est plus duale réalisable au problème suivant $(\dot{R^{ui}})$, mais elle demeure primale réalisable pour ce même problème.

La troisième observation s'explique grâce à la relation entre un problème primal et son dual. L'ajout de nouvelles colonnes au problème primal signifie l'ajout de nouvelles lignes à son dual. Par conséquent, une solution qui est réalisable à ces deux problèmes perd sa réalisabilité duale aprés l'ajout de nouvelles lignes (contraintes) au problème dual. C'est par cette explication que la solution ($\dot{x^{n-1}_u}, \dot{z^{n-1}_u}$), qui est primale-duale réalisable au problème $(\dot{R^{(i-1)}_u})$, n'est plus duale réalisable mais demeure primale réalisable au problème suivant $(\dot{R^{ui}})$.

Description formelle de l'algorithme

La méthode GC modifiée est explicitement donnée par l'algorithme 42. Elle retourne le dernier triplet qu'elle a créé, soit celui de l'itération n qui est composé du problème restreint

($\dot{R^{un}}$), de la base $\dot{x_u^n}$ et de sa valeur $\dot{z_u^n}$. Elle retourne aussi la valeur optimale $\dot{z_u^*}$, qui débute l'algorithme avec une valeur de $+\infty$. Toutefois, si un branchement prématuré est décidé, cette valeur n'est pas atteinte à l'itération n et le test $z_u^n \stackrel{?}{=} z_u^*$ est donc faux. Il est important de souligner que ce test sera fréquemment utilisé dans plusieurs composantes de l'algorithme de branch and price pour déterminer si la valeur optimale $\dot{z_u^*}$ du problème ($\dot{R^u}$) a été atteinte par GC modifiée

Aussi, il est important de souligner qu'une condition nécessaire à l'application de GC modifiée pour la résolution de la relaxation continue ($\dot{R^u}$) doit être satisfaite. En effet, il est indispensable que ce problème soit réalisable avant de faire appel à cette méthode [2].

Dans la prochaine section, nous décrivons la méthode Pricing, qui génére l'ensemble de colonnes hors bases $PA(\hat{G}_{ki}^{cu})$ de CR^- d'une combinaison quelconque $(k,c) \in K_{I_k}^u$ et qui vérifie la condition d'optimalité $(\forall (k,c) \in K_{I_k}^u, PA(\hat{G}_{ki}^{cu}) = \emptyset)$

Algorithme 40 ***Méthode de génération de colonnes GCmodifiée***

Méthode de génération de colonnes pour résoudre le problème linèaire (R^u) , qui correspond à (R^u) du problème (P^u) au noeud u.

GC modifiée ($\dot{R^{u1}}$) : problème initial) $\Longrightarrow$ ($\dot{R^{un}}$)), $\dot{x_u^n}, \dot{z_u^n}, \dot{z_u^*}$) :

$\alpha_k = MIN(\mid p_k \mid)$

$\dot{z_u^*} \leftarrow +\infty$

tant que $(i = 0 \curlyvee PA(\hat{G}_{ki}^{cu}) \neq \emptyset)$ **effectuer**

$i \leftarrow i + 1$

$(\dot{x_u^i}, \dot{z_u^i}, [\lambda_L]^i, [\mu_k]^i) \leftarrow$ **Simplexe** ($\dot{R^{un}}$)

$PA(\hat{G}_{ki}^{cu}) \leftarrow$ Pricing ($K_{I_k}^u$, $[\lambda_L]^i$,$[\mu_k]^i$) **{Gen. de var. CR$^-$}**

si $PA(\hat{G}_{ki}^{cu}) \neq \emptyset$ **alors**

si branchement Prematuré ($\dot{z_u^i}$) **alors**

$n \leftarrow i$

Aller à (*)

$(\dot{R^{u(i+1)}}) \leftarrow (\dot{R^{ui}}) \oplus PA(\hat{G}_{ki}^{cu})$

$n \leftarrow i$

$\dot{z_u^*} \leftarrow \dot{z_u^i}$ { La valeur optimale $\dot{z_u^*}$ a été atteinte :$\dot{z_u^n} \stackrel{?}{=} \dot{z_u^*}$ est vraie

(*)**retourner**$((\dot{R^{un}})), \dot{x_u^n}, \dot{z_u^n}, \dot{z_u^*})$

L'algorithme Pricing

La méthode Pricing a pour tâche de générer des colonnes de CR^- qui sont hors bases lors de l'itération courante i de GC modifiée. Les variables duales de chaque contrainte de quotas et de chaque contrainte de partitionnement de cette itération doivent être fournies. Elles sont respectivement notées par les vecteurs $[\lambda_L]^i$et $[\mu_k]^i$ et sont obtenues par la méthode du Simplexe.

L'algorithme parcourt l'ensemble K des personnes afin de trouver au moins une colonne CR^- hors base. Par contre, lorsqu'un personne $k \in K$ est traitée, il faut considérer les différentes façons de respecter simultanément toutes les contraintes de branchement $Ryan - Foster$. Une telle façon ou $RFRC$ est notée $(k, c) \in K_{I_k}^u$. Alors, pour chacune de ces combinaisons, la phase du problème auxiliaire est appelée dans le but de trouver une ou plusieurs colonnes hors base qui respectent la définition de cette combinaison et qui sont de coûts réduits négatifs selon les variables duales $[\lambda_L]^i$et $[\mu_k]^i$.

Afin de bien comprendre l'algorithme Pricing, nous commençons par introduire le lecteur au concept de $RFRC$. À l'aide d'exemples et du développement d'un algorithme nommé calculer I_k^u .

Ensuite, la phase du problème auxiliaire est aussi introduite. Nous terminons en décrivant les deux résultats pouvant être obtenus par l'application de Pricing.

Introduction au concept de RFRC[26])

il y a deux possibilités pour le respect d'une contrainte de branchement Ryan-Foster. Par conséquent, le nombre total de combinaisons des deux possibilités pour le respect de chaque contrainte de branchement Ryan-Foster est de $2^{|C_k^u|}$, où $k \in K$, u est le noeud courant et $C_k^u \leftarrow C_{k=}^u \cup C_{k\neq}^u$. Les deux ensembles de contraintes de branchement Ryan-Foster $C_{k=}^u$ et $C_{k\neq}^u$ et pour le moment le lecteur doit bien noter que $|C_k^u|$représente le nombre total de ces contraintes.

Par contre, ce ne sont pas toutes les combinaisons mentionnées ci-haut qui sont possibles. En effet, certaines d'entres elles peuvent contenir une ou plusieurs contradictions.

Pour l'instant, notons par$I_k^u \subseteq \{0, 1, ..., 2^{|C_k^u|}-1\}$ l'ensemble des indices des combinaisons qui

sont sans contradiction pour $k \in K$ et pour le problème (P^u). De plus, on note par $(k,c) \in K_{I_k^u}$, où $K_{I_k^u}^u = \{(k,c) : k \in K, c \in I_k^u\}$, la combinaison $c \in I_k^u$ de la personne $k \in K$ au problème(P^u).

l'algorithme qui calcule I_k^u dont les tâches sont les suivantes :

(1) calculer l'ensemble I_k^u

(2)pour chaque $c \in I_k^u$, calculer les ensembles O_k^{cu} et F_k^{cu} . Ces deux ensembles représentent respectivement l'ensemble des affectations obligatoires et l'ensemble des affectations interdites définissant la combinaison $(k,c) \in K_{I_k^u}$.

Nous verrons plus loin que l'application de calculer I_k^u pour chaque personne $k \in K$ et pour le problème (P^u) est un pré-traitement effectué au début de GC modifiée.

3.2 problème auxiliaire

3.2.1 Introduction

L'ensemble PA a pour objectif de générer des horaires de CR^-qui respectent la définition d'une combinaison $(k,c) \in K_{I_k^u}$ au noeud u, en plus des contraintes intrinsèques à la personne $k \in K$. Cette seconde phase est un algorithme de plus court chemin appliqué au graphe d'état $\hat{G}_{ki}^{cu}$ et cette application est notée $PA(\hat{G}_{ki}^{cu})$. L'ensemble de colonnes hors base de CR^-, qui est obtenu par cette application, est aussi noté $PA(\hat{G}_{ki}^{cu}) \subseteq S_k$.

Ensuite, le graphe d'état $\hat{G}_{ki}^{cu}$, qui représente la combinaison $(k,c) \in K_{I_k^u}$ et qui tient compte des variables duales $[\lambda_L]^i$ et $[\mu_k]^i$ pour le calcul des coûts réduits C_{ks} des horaires $s \in S_k^{cu}$, est défini à l'aide du graphe d'´etat $\hat{G}_k$. En effet, deux prétraitements doivent être appliqués au graphe d'état $\hat{G}$ pour définir $\hat{G}_{ki}^{cu}$:

1) modification du graphe d'état$\hat{G}_k$ selon les contraintes définies par O_k^{cu} et F_k^{cu}

2) modification du graphe d'état $\hat{G}_k$ pour la prise en compte des variables duales $[\lambda_L]^i$ et $[\mu_k]^i$.

Enfin, il est théoriquement impossible pour les colonnes obtenues par $PA(\hat{G}_{ki}^{cu})$ d'être déjà incluses dans le problème restreint $(\dot{R^{ui}})$.

Autrement dit, cet ensemble est tel que $PA(\hat{G}_{ki}^{cu}) \subseteq \{s \in S_k^{cu} : s \notin S_k^{ui} \text{ et } c_{ks} < 0\}$. Aussi, mentionnons que $PA(\hat{G}_{ki}^{cu}) = \emptyset$ si et seulement si $\{s \in S_k^{cu} : \text{s} \notin \dot{S_k^{ui}} \text{ et } C_{ks} < 0\} = \emptyset$.

Résultats possibles de Pricing

Aprés l'application de Pricing au triplet $(K_{I_k^u}, [\lambda_L]^i, [\mu_k]^i\ )$, les 2 résultats suivants sont possibles :

– soit que $\exists (k,c) \in K_{I_k^u}$ tel que $PA(\hat{G}_{ki}^{cu}) \neq \emptyset$

– ou soit que $\forall (k,c) \in K_{I_k^u}$, $PA(\hat{G}_{ki}^{cu}) = \emptyset$.

Le premier résultat signifie que la solution courante $\dot{x_u^i}$ de l'algorithme GC modifiée n'est pas une solution optimale du problème ($\dot{R^u}$) associé au probléme P^u. En effet, il a été prouvé que pour une combinaison $(k,c) \in K_{I_k^u}$, il existe au moins une colonne $s \in S_k^{cu}$ hors base ($s \notin \dot{S_k^{ui}}$) et de coût réduit négatif ($C_{ks} < 0$), donc $\dot{x_u^i}$ n'est pas primale-duale réalisable au probléme ($\dot{R^u}$).

Le deuxième résultat signifie qu'aucune colonne de CR^- hors base n'existe pour chaque combinaison $(k,c) \in K_{I_k^u}$ et selon la valeur des variables duales $[\lambda_L]^i$ et $[\mu_k]^i$

Dans ce cas, la valeur courante $\dot{z_u^i}(\dot{z_u^n})$ de l'algorithme GC modifiée est la valeur optimale du problème $((\dot{R^u})(\dot{z_u^n}) \stackrel{?}{=} \dot{z_u^*}$ est vraie).

Finalement, nous pouvons voir la méthode Pricing décrite de façon formelle à l'algorithme 42

Algorithme 41 .

L'algorithme Pricing où i est l'itération courante de GC modifiée**.**

Pricing ($K_{I_k^u}$: **ensemble de** $RFRC$,$[\lambda_L]^i, [\mu_k]^i$: **variables duales**)

$\Rightarrow PA(\hat{G}_{ki}^{cu}) \subseteq S_k^{cu}$ **pour tout** $(k,c) \in K_{I_k^u}$ effectuer

$\bullet G_{ki}^{cu} \leftarrow G_k$

• Modifier $\hat{G}_{ki}^{cu}$ selon les ensembles O_k^{cu} et F_k^{cu} {Pré-traitement 1}

• Modifier $\hat{G_{ki}^{cu}}$ selon les variables duales $[\lambda_L]^i et [\mu_k]^i$ {Pré-traitement 2}

• Appliquer$PA(\hat{G}_{ki}^{cu})$

si $PA(\hat{G}_{ki}^{cu}) \neq \emptyset$ **alors**

• **retourner** $PA(\hat{G}_{ki}^{cu})$ $\{\exists (k,c) \in K_{I_k^u}$ tq. $PA(\hat{G}_{ki}^{cu}) \neq \emptyset$ $\}$.

• **retourner** $\emptyset$ $\{\forall (k,c) \in K_{I_k^u}, PA(\hat{G_{ki}^{cu}}) = \emptyset$ $\}$

3.2.2 Séparation disjointe et complète

Une séparation appliquée au noeud u consiste à partitionner l'ensemble de solutions S^u du $PLNE(P^u)$ en deux sous-ensembles disjoints : $S^v \subseteq S^u$ et $S^w \subseteq S^u$.S^vet S^w sont respectivement l'ensemble de solutions du problème (P^v) au noeud et v du problème (P^w)au noeud w.

Notons par$[s_{k_1}, s_{k2}, ..., s_{k_{|K|}}] \in S$, où pour $k \in K$ $s_k \in S_k$, un horaire général réalisable quelconque. Alors, pour aider à la compréhension de ce mémoire mais surtout pour préparer notre argumentation concernant l'exhaustivité de notre algorithme de branch and price, nous devons introduire deux nouvelles définitions.voir[2]

3.2.3 Règles de branchement Ryan-Foster

Deux méthodes de séparation sont possibles pour le partionnement de l'ensemble S^u : règle de branchement $Ryan - Foster$. La règle de branchement choisie est effectuée sur le problème (P^u) afin de définir ses deux

problèmes fils (P^v) et (P^w)et s'applique à $k \in K$, qui est nommée "personne de branchement". De plus, la contrainte de branchement qui est ajoutée à l'un des problèmes est la négation de celle qui est ajoutée à l'autre, mais elles concernent toutes les deux cette infirmière.

Avant de définir les deux règles de branchement, il est important d'aviser le lecteur qu'elles sont toutes les deux définies selon le problème restreint (R^{un}) et non pas (R^u).C'est pour cette raison que les deux variables s_1 et $s_2 \in S_k$ (pour $k \in K$), qui sont choisies pour définir l'un ou l'autre de ces branchements, doivent être telles que s_1 et $s_2 \in S_k^{un}$ et non pas seulement s_1 et $s_2 \in S_k^u$ .voir[2]

Chapitre 4

Résultats numériques

ce chapitre présente des résultats numériques pour quelques instances de la confection d'horaires des personnes. Le but de cette recherche est de développer un modèle mathématique pour une application réelle. Par conséquent, ce chapitre a pour but principal de montrer que le modèle développé au chapitre 2, en conjonction avec la méthode de résolution choisie, permet de résoudre des instances de la confection d'horaires des personnes afin d'obtenir une solution de bonne qualité en des temps raisonnables.

4.1 Présentation d'un exemple

Maintenant, nous présentons un exemple pour le problème PSP. Nous commençons par énumérer ses données. Nous décrivons ensuite tous les horaires possibles pour chacune de ses personnes et nous terminons en décrivant toutes ses solutions possibles, incluant sa solution optimale.

4.1.1 Les données de l'exemple

Nous ne présentons pour le présent exemple que les données du premier niveau. Rapidement, voici ces données :

$K = \{k_1, k_2, k_3\}, D = \{d_1, d_2, d_3\}, T = \{t_1\}$ et $\Lambda = \{\ell_1, \ell_2, \ell_3, \ell_4\}$. Les caractéristiques des personnes sont décrites ci-dessous.

– Pour $k_1, T_{k_1} = \{t_1\}$ et $k_1 = \{\ell_1, \ell_2\}$.

– Pour $k_2, T_{k_2} = \{t_1\}$ et $k_2 = \{\ell_3\}$.

– Pour $k_3, T_{k_3} = \{t_1\}$ et $k_3 = \{\ell_4\}$.

Les contraintes de quotas sont définies par le tableau 4.1. Il ya 4 contraintes de quotas (Q_{L_1}) et (Q_{L_2}) définies sur (d_3, t_1) sont liées par une contrainte $(\widehat{Q}_{d_3t_1})$ qui est aussi définie sur cette paire. Notons que (Q_{L_1}) et (Q_{L_2}) ont toutes les deux un quota cible de 1 mais elles acceptent respectivement un déficit et un surplus tous deux de valeur maximale égale à 1. Par contre, grâce à la contrainte $(\widehat{Q}_{d_3t_1})$, soit que les quotas obtenus pour les contraintes (Q_{L_1}) et (Q_{L_2}) sont tous les deux de valeur 1 ou soit qu'ils sont respectivement de valeur 0 et 2. En effet, la contrainte $(\widehat{Q}_{d_3t_1}))$, à laquelle sont associés l'ensemble de compétences $\widehat{L} = \{\ell_1, \ell_2, \ell_3, \ell_4\}$ et le triplet $(2, 0, 0)$, oblige un quota final de 2 et n'accepte ni déficit ni surplus sur ce quota cible, qui représente le total des quotas obtenus pour les contraintes (Q_{L_1}) et (Q_{L_2}).

4.1.2 Les horaires existants pour les trois personnes.

Même si le problème PSP ne limite pas à un le nombre de compétences par personne, une affectation $(d, t) \in D_{T_k}$ d'un personne $k \in K$ et d'un horaire $s \in S_k$

Tableau .4.1 : Les contraintes de quotas définies sur l'horizon.

	d_1	d_2	d_3
t_1	$(Q_{L1}) : (\{\ell_1\}, (1, 0, 0))$	$(Q_{L1}) : (\{\ell_3, \ell_4\}, (2, 1, 0))$	$(Q_{L1}) : (\{\ell_1, \ell_3\}, (1, 1, 0))$ $(Q_{L2}) : (\{\ell_2, \ell_4\}, (1, 0, 1))$ $(\widehat{Q}_{d_3t_1}) : \{\ell_1, \ell_2, \ell_3, \ell_4\}, (2, 0, 0, 0)$

S_k (noté par a_{ksdt}) ne peut être comptabilisée que dans une seule contrainte de quotas définie sur cette même paire. Considérons deux contraintes $(Q_{L'})$ et $(Q_{L''})$ définies sur la paire (d, t), où $L' et L'' \in L_{dt}$ et $L' \neq L''$, et supposons que $L' \cap \Lambda_k \neq \Phi$ et $L'' \cap \Lambda_k \neq \Phi$. Nonobstant les multiples compétences de lapersonne k qui permettent à l'affectation a_{ksdt} de couvrir ces deux contraintes, cette même affectation doit être comptabilisée soit dans $(Q_{L'})$ ou soit dans $(Q_{L''})$ et non pas simultanément dans les deux.

Nous pouvons voir au tableau 4.2 . tous les horaires possibles pour les trois personnes k_1, k_2 et k_3. Remarquons que les horaires s_1 et s_2 de la personne k_1 sont différents uniquement

par cause des affectations (d_3, t_1, L_1)et (d_3, t_1, L_2) qui leur sont respectivement associées. De façon plus précise, ces deux affectations auront respectivement un apport dans les contraintes (Q_{L1}) et (Q_{L2}), qui sont toutes les deux définies sur la paire (d_3, t_1). En effet, si PSP était le problème considéré, la personne k_1 aurait seulement l'horaire composé des affectations (d_1, t_1) et (d_3, t_1) de valide.

4.1.3 Les solutions possibles

Dans cet exemple, nous avons 8 horaires généraux possibles. Par contre, seulement 3 d'entres eux respectent les contraintes de quotas : ($k_1 : s_1, k_2 : s_2, k_3 : s_2$),

Tableau 4.2 : Les horaires possibles pour les personnes k_1, k_2 et k_3.

k	Λ_k	s_k	$co\hat{u}t$	d_1	d_2	d_3
k_1	$\{\ell_1, \ell_2\}$	s_1	0	(t_1, L_1)		(t_1, L_1)
		s_2	0	(t_1, L_1)		(t_1, L_2)
k_2	$\{\ell_3\}$	s_1	1		(t_1, L_1)	(t_1, L_1)
		s_2	0		(t_1, L_1)	
k_3	$\{\ell_4\}$	s_1	2		(t_1, L_1)	(t_1, L_2)
		s_2	0			

($k_1 : s_2, k_2 : s_1, k_3 : s_1$) et ($k_1 : s_2, k_2 : s_2, k_3 : s_2$). Les valeurs de ces trois horaires généraux sont respectivement 1, 3 et 3. Par conséquent, la valeur optimale du problème est 1 et ($k_1 : s_1, k_2 : s_2, k_3 : s_2$) est une solution optimale. Pour chacun des ces horaires, un tableau décrit la comptabilisation des contraintes de quotas.

Tableau 4.3. : Comptabilisation des quotas pour l'horaire ($k_1 : s_1, k_2 : s_2, k_3 : s_2$) de valeur 1.

	d_1	d_2	d_3
t_1	$(Q_{L1}) : 1$	$(Q_{L1}) : 1$	$(Q_{L1}) : 1$ $(Q_{L2}) : 1$ $(\widehat{Q}_{d_3 t_1}) : 2$

Tableau 4.4 Comptabilisation des quotas pour l'horaire ($k_1 : s_2, k_2 : s_1, k_3 : s_1$) de valeur 3.

	d_1	d_2	d_3
t_1	$(Q_{L1}) : 1$	$(Q_{L1}) : 2$	$(Q_{L1}) : 1$ $(Q_{L2}) : 1$ $(\widehat{Q}_{d_3t_1}) : 2$

Tableau 4.5 : Comptabilisation des quotas pour l'horaire $(k_1 : s_2, k_2 : s_2, k_3 : s_2)$ de valeur 3.

	d_1	d_2	d_3
t_1	$(\mathrm{Q}_{L1}) : 1$	$(\mathrm{Q}_{L1}) : 1$	$(\mathrm{Q}_{L1}) : 0$ $(\mathrm{Q}_{L2}) : 2$ $(\widehat{Q}_{d_3t_1}) : 2$

4.2 Résultats et analyse

Au chapitre 2, nous avons présentés les deux parcours possibles de l'arbre de recherche et, au chapitre 3, nous avons définis les deux règles de branchement qui peuvent être utilisées lors de ces deux algorithmes. Ces parcours sont le DFS et le $BSFS$ et ces deux méthodes de séparation sont le branchement binaire et le branchement $\mathbf{Ryan - Foster}$.

1.$\mathbf{DFS - 1}$: parcours DFS avec l'application de branchements binaire seulement.

2. $DFS - 2$: parcours DFS avec l'application de branchements $Ryan - Foster$ et de branchements binaire.

3. $BSFS - 1$: parcours $BSFS$ avec l'application de branchements binaire seulement.

4. $BSFS - 2$: parcours $BSFS$ avec l'application de branchements $Ryan - Foster$ et de branchements binaire.

Premièrement, pour chacun des 3 jeux de données présentés à la section précédente, nous allons décrire les résultats que nous avons obtenus par l'une des 4 versions de l'algorithme de branch and price énumérées ci-dessus. Aussi, nous allons donner la ou les raisons pour lesquelles cette version de Iris a été choisie pour résoudre le jeux de données considéré.

Ensuite, pour chacun des jeux de données, nous allons présenter les résultats obtenus à l'aide de deux tableaux.

Le premier tableau donne un aperçu de l'arbre de recherche résultant de l'algorithme de branch and price. Il est composé des données d´ecrites ci-dessous.

1. Le nombre de noeuds

– explorés (pour lesquels au moins un des deux fils a été défini et résolu),

– coupés,

– feuilles (pour lesquels la solution optimale est entière) ,

– total.

2. Le nombre de branchements pour $Ryan - Foster$. Le deuxième tableau donne quelques informations sur la première solution trouvée et quelques-unes des meilleures solutions trouvées durant l'exécution du programme Iris. Pour chaque solution, les informations suivantes sont données :

1. son ordre (première, deuxième, etc.),

2. la profondeur et l'ordre du noeud où elle a été trouvée,

3. le temps durant lequel elle a été trouvée,

4. sa valeur,

5. si elle est entière (oui ou non),

6. si c'est une solution optimale (oui ou non).

De plus, nous donnons la valeur de la relaxation continue du noeud racine, notée par z^*_{u0} . Rappelons que $\lceil z^*_{u0} \rceil$ est une borne inférieure sur la valeur optimale du problème .

Pour chaque exécution du programme Iris, un temps maximal de 8 heures est imposé. Ce temps inclus le pré-traitement nécessaire à la construction des graphes G_k et à leur application à $PA_{\Phi 1}\ (G_k)$. Pour les 3 jeux de données, le temps requis pour ce pré-traitement est d'environ 8 minutes. Notons aussi que la profondeur et l'ordre du noeud racine, noté par u_0, sont tous les deux de valeur 0. De plus, précisons qu'une solution qui n'est pas entiére est générée par la méthode d'arrondi .

Conclusion

Dans ce mémoire, nous avons proposé la modélisation sous forme mathématique d'un problème reéll pour le résoudre. Nous décomposons le modèle mathématique en deux parties :un proléme maitre et un problème auxiliaire.Nous présenterons dans un premier temps, les données dont nous disposons pour résoudre le problème, puis la modélsation mathématique que nous avons retenue.

Ainsi,l'objectif est de trouver un planning général satisfaisant toutes les contraintes (contraintes intrinsèques et contraintes de quotas). Chaque horaire général généré

devra être réalisable. Mais en plus, parmi l'ensemble de tous les plannings généraux réalisables, il suffit de trouver le meilleur, soit celui qui minimise la fonction objectif.

-Nombre de boucles de génération de colonnes

On a vu lors des tests que la convergence de l'algorithme est très rapide. Les deux premières boucles améliorent significativement la valeur de la fonction objectif et celle de la borne inférieure. Les deux dernières affinent principalement la connaissance de la borne inférieure. En résumé on consacre environ un tiers du temps de calcul global à déterminer plus précisément la borne inférieure sans amélioration de la fonction objectif. Cette connaissance de la borne inférieure est précieuse mais dans un contexte où le temps de calcul est précieux, peut être pourrait-on en économiser lors de cette phase. Il faudrait effectuer pour cela des tests beaucoup nombreux sur des instances différentes du problème d'optimisation afin de pouvoir tirer des lois générales sur la convergence de l'algorithme. On pourrait alors fixer un seuil d'arrêt pour l'écart connu en sachant que l'écart réel est probablement plus faible.

- Etude théorique et pratique du caractère fractionnaire de la solution relaxée

Les solutions relaxées obtenues pour toutes les instances testées étaient remarquablement

peu fractionnaires, ce qui a permis leur résolution en nombres entiers sans utiliser le processus de branch-and-price avec un résultat très proche de la borne inférieure. S'il est assuré que ce processus est valide dans le cas testé, on ne peut tout à fait exclure la possibilité que d'autres instances auraient une solution relaxée très fractionnaire. Deux approches complémentaires sont alors possibles pour lever cette incertitude. La première, expérimentale, consiste à multiplier les instances de tests. En revanche les plannings étaient similaires et peut être faudrait-il tester des instances faisant également varier ce facteur.

La deuxième approche est plus théorique : il est possible dans certains cas de montrer que les contraintes d'un programme linéaire représentent des facettes du polyèdre défini par l'enveloppe convexe des solutions en nombres entiers. Dans ce cas, la résolution du problème relaxé sera peu fractionnaire. Une telle étude en complément des tests supplémentaires permettrait d'améliorer la compréhension des résultats obtenus. Elle permettrait également de distinguer les contraintes qui jouent un rôle important dans le caractère fractionnaire ou entier des solutions relaxées. Une autre formulation des contraintes de capacité (capacité instantanée par exemple) conduirait à des solutions différentes, peut être plus fractionnaires.

- Processus d'obtention d'une solution entière Au cours du processus de génération de colonnes, un grand nombre de plannings sont calculées pour la résolution du problème relaxé. Ainsi même si on obtenait une solution relaxée très fractionnaire nécessitant la création d'un arbre de branch-and-price, le nombre de plannings à construire pour résoudre chaque problème fils serait très restreint. Etant donné que la résolution des sous programmes linéaires , la résolution de chaque sous problème par génération de colonnes ou l'obtention d'une bonne borne inférieure serait très rapide. Par ailleurs, le processus d'obtention d'une solution entière n'est donc pas limitatif de la méthode. Pour le branchement, la prise en compte des caractéristiques des items et l'utilisation de diverses techniques pour réduire le problème permettent de réduire la taille de l'arbre de branchement et d'obtenir une solution optimale entière plus rapidement . Les résultats obtenus sur les instances testées sont impressionnants et suggèrent un développement plus approfondi de cette technique.

Bibliographie

[1] ARTHUR. A, RAVINDRAN A. A multiple objective nurse scheduling using contraint logic programing, AAA/IAAI.1999, pp 838-843

[2] DOYON Jean PH. Algorithme de branch and price pour la confection d'oraires d'infirmières _université de montréal Avril 2004

[3] BAILEY. J, Integrated days off on chift personnel scheduling. Computers and IE 9(4) :395-404,97

[4] BAKER K.R, Workforce allocation in cyclical scheduling problèmes : A survey. Operation Research, 5(3) :327,1995

[5] BENOUMELAZ F, L. DJEFFAL, S. ABED, Optimisation des problémes de planning, SMA 2012, annaba 08 -09-MARS 2012

[6] BERGE C.Hypergraphes-Combinatoire des ensembles finis. Gauthiers Villars,France,1987.

[7] BOURDAIS, S. 2004. Génération automatique d'horaires en milieu hospitalier. Master's thesis, École Polytechnique de Montréal.

[8] CHAN Yew Chéong, Peter, « La planification du personnel : acteurs,actions et termes multiples pour une planification opérationnelle des personnes », Thèse de doctorat, Institut IMAG, Université Joseph Fourier-Grenoble, 1 octobre 2002.

[9] CHVÀTAL, V. 1983. Linear Programming. Freeman.

[10] CHEN J.Gand YOUNG T.W. Hybrid expert-system approach to nurse scheduling.Computer in nursing,11(4) :183,1993.

[11] DESAULNIERS, G., DESROSIERS, J., et SOLOMON, M. 1999. Accelerating strategies in column generation methods for vehicle routing and crew scheduling problems.

[12] DJEFFAL L. Elaboration de méthodes avancées pour les problémes de plannings Cas des Hopitaux.Thése de doctorat d'état préparée à l'Université D'Artois Laboratoire LGI2 A Béthune France et Univesité H L Batna Algérie 2007

[13] DJEFFAL. L, DJEFFAL. EL.A, F. BENOUMELAZ. A mixed integer linear programming model branch and bound for nurse sheduling problem. The second International Conference on Complex System. CISC'11 jijel Algérie, 2011.

[14] FORD. L.R, FULKERSON. D. R.1958. A suggested computation for maximal multi-commodity network flows. Management Science, pages $97 - 101$

[15] GEORGES WEIL,KAMEL HEUS , Patrice François, « Gymnaste : Aide à l'élaboration des roulements infirmiers . Du traitement des absences au management participatif », Laboratoire TIMC, SILM , CHU de Grenoble , Université Joseph Fourier-Grenoble,1994.

[16] GENDRON B. « Adaptation d'un modèle mathématique de génération de colonnes pour les horaires de médecins », MIC2001- 4th Metaheuristics Internationa conference,2001

[17] G.B.DANTZIG, P. WOLF, Decomposition principle for linear programs, Operational Research, 8 p.100-111, 1960

[18] HUNG R.Multiple shifte workforce scheduling under the 3-4 workweek with different weekend labour requirements.Management Science,$40(2)280 - 284, 1994$

[19] HUNG R. Hospital Nurse Scheduling, journal of nursing administration $25(7/8)$ $199521 - 35$

[20] HUNG R. Multiple shift Workforce cheduling under the 3-4 workweek with different weekday and week -end labour requirements. Management Science,40(2)280-284,1994

[21] Hôpital Royal Victoria. Convention collective du personnel infirmier.

[22] ISKEN M.W and HANCOCK W.M.A heuristic approach to nurse schedilung in hospital units wits non- stationary, urgent demand, and fixed staff size. Journal of so society for Healh Systems, 2(2) :24-41,1990

[23] J.DESROSIERS , F. SOUMI, M. DESROCHERS, Routing with time windows by column generation, Networks, 14 p.545-565, 1984

[24] J.DESROSIERS, M.E.LUBBECKE , A primer in column generation, Les cahiers du GERAD, G-2004-02, Janvier 2004

[25] JAUMARRD B and al. A genéralized linear programing model for Nurse sheduling Problème. european journal of Operational Research 107,p1 – 8,1998

[26] LABIT, P. 2000. IRIS : Amélioration d'une méthode de génération de colonnes pour la confection d'horaires d'infirmières. Master's thesis, École Polytechnique de Montréal.

[27] L.R. Ford, D.R. Fulkerson, A suggested computation for maximal multicommodity network flows, Management Science, 5 p. 97-101, 1958

[28] L.G.MITTEN Branch-and -Bound Méthods : Général Formilation and porprerties.opeations Research,18,24,341970

[29] MARC DAMBRINE et arc Dambrine et Grégory Vial - "Généralités sur les problèmes d'optimisation " Décembre2007..

[30] M.E. Lübbecke, J. Desrosiers, Selected topics in column generation, Les cahiers du GERAD, G-2002-64 2002, revisé Octobre 2004

[31] NEMHAUSER, G.L., et WOLSEY, L.A. 1988. Integer and Combinatorial Optimization. Wiley-Interscience Series in Discrete Mathematics and Optimization.Wiley-Interscience.

[32] OZKARAHAN I. and BAILEY J.E.Goal programing model subsystem of the flexible nurse scheduling support systhem.AIIE Trasactions 20(3) :306-316,1988

[33] P.C.GILMORE, R.E. GOMORY, A linear programming approach to the cutting stock problèm Part II, Operational Research, 11 p.863-888, 1963

[34] P.C. GILMORE, R.E. GOMORY, A linear programming approach to the cutting stock problem, Operational Research, 9 p.849-859

[35] R.HORST and P.M.PARADOLS Handbook of global Optimisation,Kluwer,Dordrechet 1994

[36] REMY-ROBERT, ALEXANDRE JOSEPH, « Systèmes interactifs d'aide à lélaboration de plannings de travail de personne» l , Thèse de doctorat, Laboratoire TIMC, Institut IMAG , Université Joseph Fourier-Grenoble, 07 novembre 2003.

[37] RYAN, D.M., et FOSTER, J.C. 1981. An interger programming approach to scheduling. Computer Scheduling of Public Transport, pages 269 – 280

[38] VOVOR T.Oroblème de chemins bicritès ou avec contraintes de ressource : algorithme et appllications. thèse de doctorat, Ecole polytechnique de montréal, 1997

[39] WARNER D.M scheduling nursing personnel according to nursing

[40] ZIADI. A, "Méthodes d'optimisation Global". Université de Setif -2008