

UNIVERSITÉ DE MONTRÉAL

HORAIRES PERSONNALISÉS AVEC PRIORITÉ : CONSIDÉRATION
ACCRUE DES EMPLOYÉS JUNIORS

JULIE ST-GERMAIN
DÉPARTEMENT DE MATHÉMATIQUES ET DE GÉNIE INDUSTRIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES
(MATHÉMATIQUES APPLIQUÉES)

MARS 2003



National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisitions et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-81560-9

Canada

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé :

HORAIRES PERSONNALISÉS AVEC PRIORITÉ : CONSIDÉRATION
ACCRUE DES EMPLOYÉS JUNIORS

présenté par : ST-GERMAIN Julie

en vue de l'obtention du diplôme de : Maîtrise ès sciences appliquées

a été dûment accepté par le jury d'examen constitué de :

M. HERTZ Alain, Doctorat ès sciences, président

M. DESAULNIERS Guy, Ph.D., membre et directeur de recherche

M. SOUMIS François, Ph.D., membre et codirecteur de recherche

M. GAMACHE Michel, Ph.D., membre

Remerciements

Je tiens à remercier mon directeur de recherche, M. Guy Desaulniers, pour ses conseils et commentaires, toujours appréciés, ainsi que pour son souci du détail. Je remercie également mon codirecteur, M. François Soumis, pour ses nombreuses idées et suggestions.

Un grand merci à Daniel Villeneuve pour son aide précieuse et ses compétences informatiques. Remerciements aussi à Benoît Rochefort qui a été très habile pour régler les petits pépins informatiques. Merci aussi à Éric Vatri pour m'avoir aidée avec les scripts perl.

Mentionnons aussi la compagnie Ad Opt Technologies qui a créé le logiciel PBS, et qui a bien voulu répondre à certaines de mes questions. Un merci aussi à Heykel Achour pour quelques ajustements apportés au code.

Également, un merci spécial à mes confrères étudiants ainsi qu'à mes collègues de bureau, pour l'entraide, le support moral et les discussions amicales. Malgré les hauts et les bas, j'ai grandement apprécié mon passage au GERAD et l'occasion non négligeable de jouer au tarot tous les midis.

Merci aussi à mes parents pour la confiance et les encouragements, pour m'avoir toujours soutenue dans mes choix, pour les valeurs que vous avez su me transmettre. Et un merci tout particulier à mon cher Sébastien, toi dont la simple présence m'apporte tant.

Résumé

Il existe différentes façons de construire les horaires du personnel en transport aérien. On procède généralement selon une base mensuelle et il s'agit d'attribuer à chaque employé un ensemble d'activités (séquences de vols ou rotations, repos, congés, etc) qui constitueront un horaire valide. Dans le cas qui nous concerne, la construction d'horaires mensuels personnalisés avec priorité s'effectue de façon séquentielle selon l'ordre d'ancienneté des employés de façon à privilégier les plus seniors. On cherche à maximiser les préférences de chacun sans nuire pour autant à la réalisation des tâches résiduelles : chaque employé doit avoir un horaire admissible et tous les vols prévus doivent être couverts.

Le *Preferential Bidding System* est un logiciel conçu par la compagnie Ad Opt Technologies pour répondre en ce sens aux besoins des compagnies aériennes, en particulier Air Canada. Avec ce système, chaque employé indique et pondère ses préférences (ex : congés spéciaux, destinations particulières, etc) ce qui permet d'attribuer un pointage à chaque horaire pour chaque employé. En débutant par les employés les plus seniors, le système cherche alors un horaire de pointage minimal pour chaque employé tout en s'assurant que des horaires admissibles peuvent être définis pour les employés juniors. L'horaire de chaque employé est déterminé à l'aide d'une méthode de génération de colonnes et d'un processus de séparation et évaluation progressive. Lorsque plusieurs horaires de pointage minimal existent, la méthode de résolution standard fixe arbitrairement l'un de ces horaires sans évaluer l'impact potentiel sur les horaires des employés suivants. Les employés plus juniors risquent ainsi d'avoir des horaires de moins bon pointage, ce que nous cherchons à améliorer.

Ce mémoire présente différents algorithmes permettant d'énumérer plusieurs horaires

ayant le pointage optimal pour un employé donné. Diverses approches sont explorées et commentées, le but étant de pouvoir énumérer pour chaque employé plusieurs horaires admissibles ayant le bon pointage. Deux des algorithmes proposés correspondent à des problèmes de plus court chemin avec dominance modifiée : la méthode variera selon que l'on se base sur les coûts ou sur les coûts réduits. L'algorithme basé sur les coûts réduits implique que l'on récupère les variables duales requises, ce qui complique quelque peu le processus. Une autre méthode d'énumération, plus simple que les précédentes, consiste à retenir les colonnes du problème maître ayant le coût désiré. Cette méthode ne permet toutefois pas d'énumérer tous les horaires cherchés.

Une fois que les horaires de pointage optimal sont énumérés, il s'agit d'évaluer lequel de ces horaires semble le plus avantageux pour les employés suivants. Pour ce faire, nous présenterons deux façons d'évaluer la pénalité reliée au choix d'un horaire donné. Le branchement sera modifié en conséquence, afin de choisir l'horaire qui aura été évalué le moins pénalisant pour les employés suivants. Advenant que le choix occasionne un retour en arrière (si non-réalisable), le branchement s'effectuera alors sur le deuxième choix, la liste triée des différentes possibilités ayant été mise en mémoire. Différents tests sont ensuite présentés afin d'évaluer l'impact de la nouvelle méthode proposée ainsi que les temps de calculs requis.

De toutes les analyses effectuées, il ressort que l'algorithme utilisant les colonnes du problème maître semble le plus avantageux, tant en termes de l'amélioration de la qualité des horaires, que du point de vue des temps de calcul requis. Bien que les améliorations ne touchent en moyenne que 7% des employés, il s'agit d'améliorations substantielles quand le système a pour objectif de construire le meilleur horaire pour chaque employé.

Abstract

Monthly crew scheduling is a big challenge in air transportation. It consists in assigning to every crew member a set of activities (pairings, rest periods, days off, training periods) for the month. A pairing is a set of flights beginning and ending at the same airport or base. The *Preferential Bidding System* (PBS), developed by Ad Opt Technologies for Air Canada, does personalized monthly scheduling with priority. It builds schedules sequentially, from the most senior to the most junior employee, in a way to favor the most senior. The goal is to maximize the preferences of each employee, respecting that every employee should have a schedule, and all the pairings should be assigned.

PBS considers the preferences or bids that each employee indicates and weighs (ex: a special day off, a particular destination, etc). These weights allow to give a score to each schedule for each employee. Beginning with the most senior employees, the system searches a schedule with the minimal score for each employee, taking into account that admissible schedules could be defined for the junior employees. The schedule of each employee is found by a method of column generation embedded in a branch-and-bound tree. When there are many schedules having the minimal score, the standard solution method fixes randomly one of these schedules without evaluating the potential impact on the following employees' schedules. This is why junior employees might have schedules with less good scores, the situation that we are trying to improve.

This master thesis presents different algorithms for enumerating schedules having the optimal score for a given employee. Various approaches are explored and discussed.

Two methods are based on modified shortest path problem algorithms: one uses costs, the other uses reduced costs. The method with reduced costs is complicated by the fact that dual variables are needed. Another method, rather simpler, is to retain columns of the master problem having the desired score. This method, however, does not allow to enumerate all schedules.

Once we have enumerated optimal schedules for a given employee, we then have to evaluate which of them seems better for the following employees. We present two methods for evaluating the penalty related to the choice of a given schedule. The sorted list of the possible schedules should be kept in memory. Branching will be modified in order to choose what would have been evaluated as the best schedule for the other employees. If this choice involved backtracking, the branching would then be fixed on the second choice on the list. Finally, tests are presented in order to estimate the impact of the new method and the time required.

After analysis, considering the improvements obtained and the required time, the algorithm using columns of the master problem seems to be the best. Even if the improvements concern in average only 7% of all employees, those improvements are substantial when the system has for goal to build the best schedule for each employee.

Table des matières

REMERCIEMENTS	iv
RÉSUMÉ	v
ABSTRACT	vii
TABLE DES MATIÈRES	ix
LISTE DES TABLEAUX	xiii
LISTE DES FIGURES	xv
CHAPITRE 1 : INTRODUCTION	1
1.1 : Types d'horaires de personnel en transport aérien	3
1.2 : Horaires personnalisés avec priorité en transport aérien	4
1.2.1 : Problématique des employés juniors	4
1.3 : Revue de littérature	5
1.3.1 : Un problème similaire : construction des rotations	6

	x
1.3.2 : Fabrication d'horaires personnalisés : <i>rostering</i>	8
1.3.3 : Fabrication d'horaires personnalisés avec priorité	9
1.3.4 : <i>Preferential Bidding System</i>	10
1.4 : Contribution	12
CHAPITRE 2 : L'APPROCHE PBS	13
2.1 : Description du problème	13
2.1.1 : Description des contraintes	14
2.2 : Modélisation	16
2.2.1 : Réseaux et ressources	18
2.3 : Approche de résolution existante	21
2.3.1 : Coupes et branchement interne	23
2.3.2 : Branchement externe	24
CHAPITRE 3 : ALGORITHMES D'ÉNUMÉRATION DES CHE- MINS DE MÊME POINTAGE	26
3.1 : Notation associée au réseau	26
3.2 : Problème de plus court chemin avec contraintes de ressources	28
3.2.1 : Revue de littérature	29

	xi
3.2.2 : Quelques concepts	30
3.3 : Fonctions de prolongation linéaires	32
3.3.1 : Algorithme basé sur les coûts	33
3.3.2 : Algorithme basé sur les coûts réduits	36
3.4 : Fonctions de prolongation non linéaires	42
3.4.1 : Algorithme basé sur les coûts	44
3.4.2 : Algorithme basé sur les coûts réduits	45
3.4.3 : Algorithme utilisant les colonnes du problème maître	46
CHAPITRE 4 : MODIFICATION DE LA MÉTHODE DE RÉSOLUTION	47
4.1 : Pénalité associée à un horaire	48
4.1.1 : Première méthode	48
4.1.2 : Deuxième méthode	49
4.2 : Branchement multiple sur les horaires d'un employé	50
4.3 : Ajustements dus à l'implantation	53
CHAPITRE 5 : RÉSULTATS OBTENUS	56
5.1 : Description des jeux de données	56

5.2 : Comparaison des méthodes et ajustement des paramètres	58
5.3 : Dénombrement des employés à horaires multiples	60
5.4 : Tests comparatifs	62
5.4.1 : Exposé de la démarche	63
5.4.2 : Algorithme COST	65
5.4.3 : Algorithme MASTER	68
5.4.4 : Sommaire	72
5.5 : Deux exemples plus détaillés	74
5.5.1 : Un exemple de petite taille : YUL-767	75
5.5.2 : Un exemple de taille moyenne : YVR-320	76
5.6 : Temps requis au total	78
CONCLUSION	80
BIBLIOGRAPHIE	84

Liste des tableaux

Tableau 5.1 : Caractéristiques des jeux de données	57
Tableau 5.2 : Comparaison des modèles de dominance	59
Tableau 5.3 : Statistiques sur les chemins - algorithme COST	61
Tableau 5.4 : Statistiques sur les chemins - algorithme MASTER	62
Tableau 5.5 : Résultats comparatifs : 1er jeu de données, algorithme COST	66
Tableau 5.6 : Résultats comparatifs : 2e jeu de données, algorithme COST	67
Tableau 5.7 : Résultats comparatifs : 3e jeu de données, algorithme COST	67
Tableau 5.8 : Résultats comparatifs : 4e jeu de données, algorithme COST	68
Tableau 5.9 : Résultats comparatifs : 5e jeu de données, algorithme COST	69
Tableau 5.10 : Résultats comparatifs : 1er jeu de données, algorithme MAS- TER	70
Tableau 5.11 : Résultats comparatifs : 2e jeu de données, algorithme MAS- TER	70
Tableau 5.12 : Résultats comparatifs : 3e jeu de données, algorithme MAS- TER	71

Tableau 5.13 :Résultats comparatifs : 4e jeu de données, algorithme MAS- TER	71
Tableau 5.14 :Résultats comparatifs : 5e jeu de données, algorithme MAS- TER	72
Tableau 5.15 :Impact partiel : algorithme COST	73
Tableau 5.16 :Impact général : algorithme COST	73
Tableau 5.17 :Impact général : algorithme MASTER	73
Tableau 5.18 :Résultats détaillés 1er jeu de données	75
Tableau 5.19 :Résultats détaillés 2e jeu de données	77
Tableau 5.20 :Temps de résolution	78

Table des figures

Figure 2.1 : Fragment de réseau	20
Figure 2.2 : Fonctionnement de GENCOL	22
Figure 2.3 : Arbre de branchement externe	25
Figure 3.1 : Dominance des étiquettes de plus court chemin avec une seule ressource.	35
Figure 3.2 : Dominance des étiquettes $(\underline{z}_i^l, \underline{T}_i^l)$ avec une seule ressource. .	41
Figure 4.1 : Nouvel arbre de branchement externe.	52

Chapitre 1

Introduction

Établir l'horaire mensuel du personnel navigant d'une compagnie aérienne s'avère un problème particulièrement important. Important d'abord d'un point de vue financier, vu les économies considérables impliquées. En effet, après le carburant, la masse salariale du personnel navigant représente la deuxième dépense en importance sur l'ensemble des coûts d'opérations (Gershkoff, 1989). Pour une grande compagnie aérienne, cela peut représenter des centaines de millions de dollars annuellement. Important ensuite si l'on considère la grande taille et la complexité du problème, tenant compte des multiples contraintes à respecter, entre autres la convention collective des employés et les règles de sécurité aérienne à respecter. Dans le cas des horaires personnalisés avec priorité, l'objectif étant de maximiser la satisfaction des employés (et non de réduire nécessairement les coûts), une résolution efficace signifie un temps de calcul raisonnable (donc économie de temps) et l'attribution de toutes les rotations aux employés (économie financière comparativement à une solution où il faudrait embaucher davantage d'employés pour effectuer toutes les tâches).

L'horaire d'un employé pour le mois courant constitue ce que l'on appelle un bloc mensuel. On veut ainsi attribuer un bloc mensuel à chaque employé, un bloc mensuel étant constitué par une suite de vols, repos, congés, périodes d'entraînement, etc. Ce bloc tient compte des qualifications et des activités pré-assignées de l'employé, s'étale en général sur une période d'un mois et doit respecter les normes de sécurité aérienne ainsi que la convention collective. Déterminer l'horaire mensuel du personnel

navigant signifie donc construire et attribuer les blocs mensuels. En transport aérien, la construction des blocs mensuels constitue le dernier problème à résoudre lors de la planification des opérations. Cette planification procède suivant des étapes bien précises :

1. **Déterminer l'horaire des vols** : Dans un premier temps, l'horaire des vols à couvrir pour la saison courante est établi, tenant compte des demandes du marché, de la disponibilité des heures de décollage et d'atterrissage dans les différentes stations (aéroports), ainsi que de la disponibilité même des avions.
2. **Affectation des avions** : Selon les caractéristiques propres à chaque vol (durée, demande prévue), il s'agit de déterminer quel avion (de type approprié) peut être affecté à chacun de ces vols, considérant des contraintes telles le nombre d'avions disponibles et le temps minimum de connexion entre chaque paire de vols consécutifs pour un même avion. À ce niveau, on construit également les itinéraires d'avions et l'on planifie partiellement l'entretien.
3. **Formation des rotations d'équipage** : On construit des rotations en regroupant plusieurs segments de vol partant et revenant à une même ville (base), chaque rotation étant effectuée par un seul et même équipage. Ce problème se décompose d'abord par flotte, ensuite on résout séparément pour les pilotes et les agents de bord. On doit respecter les contraintes de la convention collective et de la répartition des rotations entre les bases.
4. **Construction et attribution des blocs mensuels** : Il s'agit ensuite de répartir entre les employés qualifiés les différentes rotations devant être couvertes tout en s'assurant de respecter les règles de sécurité aérienne ainsi que la convention collective des employés. On doit également attribuer les périodes de repos, congés et tenir compte des activités pré-assignées (entraînements, congés annuels, ...).

Cette dernière étape est celle sur laquelle nous nous attarderons, soit la fabrication proprement dite de l'horaire du personnel navigant.

1.1 Types d'horaires de personnel en transport aérien

Il existe différents types de fabrication d'horaires mensuels de personnel en transport aérien. Tout dépend, bien sûr, de l'objectif visé ainsi que de la politique particulière des compagnies aériennes.

1. Horaires anonymes : *bidline*

Les horaires sont d'abord construits indépendamment des préférences des employés. Chaque employé choisit ensuite, selon son rang d'ancienneté, l'horaire qu'il préfère parmi ceux qui lui sont admissibles. Ce type d'horaires a été utilisé notamment par British Airways (Wilson 1981) et American Airline (Jones 1989).

2. Horaires personnalisés : On accorde davantage d'importance à construire des blocs mensuels répondant le plus possible aux préférences des employés. Deux procédés sont utilisés.

(a) *rostering* : Les blocs sont construits en tenant compte des activités pré-assignées de chaque employé. On vise à maximiser la satisfaction générale des employés. Pour ce faire, la compagnie définit les critères bonifiés (ex : bonis pour congés à intervalle régulier) entrant dans la construction d'un "bel" horaire. Ce procédé a été utilisé surtout par les grandes compagnies européennes telles Air France (Giafferri et al. 1982, Gontier 1985, Gamache et Soumis 1998), Lufthansa (Glanert 1984), et SwissAir (Tingley 1979).

(b) *preferential bidding* : On construit les blocs en considérant les préférences spécifiques de chacun et en priorisant les employés seniors. Ce mode de fabrication est employé surtout par certaines compagnies nord-américaines telles Air Canada (Gamache et al. 1998) et Air Transat.

Dans la prochaine section, nous nous pencherons plus avant sur ce dernier type de fabrication d'horaires, soit celui qui nous intéresse le plus dans ce travail.

1.2 Horaires personnalisés avec priorité en transport aérien

Le *preferential bidding* est le terme anglais couramment utilisé pour désigner la fabrication d'horaires personnalisés avec priorité. Ce procédé a comme objectif de maximiser la satisfaction de chacun des employés en donnant priorité à ceux ayant le plus d'ancienneté. Pour ce faire, chaque employé aura préalablement indiqué ses différentes préférences au moyen d'un système de pointage. Par exemple, si l'employé désire avoir congé un week-end, il attribuera alors plus de points pour cette activité. Il est également possible d'attribuer un pointage négatif pour une activité jugée indésirable. Le pointage d'un horaire, pour un employé donné, sera ainsi la somme des points sur les différentes activités choisies pour composer cet horaire. Ce pointage dépendra donc des préférences de l'employé.

Les horaires seront ainsi construits séquentiellement selon l'ordre d'ancienneté des employés : il s'agit d'attribuer à chacun un horaire admissible de pointage maximal, en s'assurant que les rotations résiduelles (pas encore attribuées) puissent être effectuées par le personnel résiduel (juniors). La difficulté du problème, lorsque l'on résout pour un employé particulier, réside dans l'obtention d'une solution optimale permettant la faisabilité du problème résiduel pour les employés suivants. Une des méthodes pouvant être employée pour résoudre ce problème de grande taille est la génération de colonnes imbriquée dans un processus de séparation et évaluation progressive (*Branch-and-Bound*). Le présent mémoire s'inscrit dans le contexte de cette approche de résolution.

1.2.1 Problématique des employés juniors

Suivant l'approche ci-haut énoncée, il faut remarquer qu'au fur et à mesure que l'on avance dans la résolution, davantage de rotations sont déjà attribuées, ce qui

signifie un choix davantage restreint pour les employés résiduels, donc un resserrement des horaires possibles. Pour un employé donné, il peut exister plusieurs horaires admissibles de meilleur pointage. Aucun critère particulier ne détermine lequel des horaires de meilleur pointage sera choisi, cette décision est arbitraire. Comme nous l'avons mentionné plus haut, fixer l'horaire de l'employé courant restreint le choix de rotations (non-attribuées) pour les employés suivants. Une analyse (pour l'instant inexistante) des horaires de meilleur pointage permettrait sans doute d'effectuer un choix favorisant davantage les employés juniors.

Le présent mémoire propose donc de modifier la méthode de résolution actuellement utilisée de façon à améliorer la qualité des horaires pour les employés juniors. Pour ce faire, nous préconisons la démarche suivante : énumérer tous les horaires admissibles de meilleur pointage pour un employé donné, analyser ces horaires en fonction des préférences des employés résiduels afin de choisir celui qui favorise le plus ces derniers. Nous espérons ainsi améliorer la qualité des horaires des employés juniors, tout en ne diminuant en rien le pointage des employés seniors.

1.3 Revue de littérature

La fabrication d'horaires personnalisés avec priorité étant relativement récente, il existe à ce jour peu de littérature sur ce sujet. Précisons également que l'on retrouve davantage de littérature concernant la construction des rotations que la fabrication proprement dite des horaires. Ceci est compréhensible puisque des gains monétaires plus importants peuvent être réalisés lors de la construction des rotations. Soulignons de plus que ces deux étapes, la construction des rotations et la fabrication d'horaires, constituent des problèmes de très grande taille. Le problème de fabrication d'horaires peut néanmoins être divisé en plusieurs problèmes indépendants de plus petite taille

(pilotes vs agents de bord, bases, type d'appareil). Le nombre d'horaires possibles pour chacun de ces problèmes demeure toutefois élevé. En effet, Antosik (1978) estime que le problème de 670 rotations et d'une soixante d'employés peut donner 10^{23} solutions. Pour des problèmes d'affectations de 200 à 700 rotations, Ryan (1992) évalue que le nombre de solutions peut s'élever de 10 à 100 millions. La fabrication d'horaires personnalisés est donc considérée comme un problème de grande taille et plus compliqué que la fabrication des horaires anonymes, ce dernier mode de fabrication étant le plus simple puisque les horaires sont construits sans tenir compte des préférences des employés (ce problème ne sera pas considéré dans cette revue).

C'est pourquoi nous proposons d'abord un bref survol des articles concernant la construction des rotations. Ceci sera suivi d'un résumé des travaux portant sur la fabrication d'horaires personnalisés (*rostering*, avec priorité, et plus spécifiquement le *Preferential Bidding System*).

1.3.1 Un problème similaire : construction des rotations

La construction des rotations (*crew pairing problem*) est aujourd'hui un problème de grande taille qui comporte des règles complexes, le nombre de variables atteignant facilement plusieurs millions. Notons également que ce problème possède une structure semblable à celle du problème de fabrication d'horaires. On regroupe des segments de vol pour construire des rotations alors que l'on regroupe des rotations pour constituer des horaires mensuels. Pour ces deux problèmes, il s'agit donc de recouvrir un ensemble de tâches par un ensemble de routes. Vu les similitudes entre le problème de construction des rotations et celui de la fabrication des horaires, nous présentons ici une brève revue concernant quelques méthodes de résolution proposées.

Marsten (1974) ainsi que Marsten et Shepardson (1981) ont formulé le problème de construction des rotations comme un problème de partitionnement de grande taille

et proposé un algorithme de résolution pour ce type de problèmes. L'algorithme résout le problème de partitionnement après que les rotations aient été générées par une heuristique. Une méthode de résolution heuristique basée sur la génération de colonnes a ensuite été développée par Crainic et Rousseau (1987). Une heuristique était alors utilisée pour intégrer dans le processus de résolution des rotations dont la durée était de plus en plus longue.

Une autre approche utilisant la génération de colonnes a été suggérée par Lavoie, Minoux et Odier (1988). L'algorithme du simplexe est utilisé pour résoudre la relaxation linéaire du problème de recouvrement, ceci constituant le problème maître. Le sous-problème utilise ensuite un algorithme de plus court chemin (chaque chemin correspondant à une rotation possible) pour générer les nouvelles colonnes à partir des variables duales obtenues lors de la résolution du problème maître. La solution entière est obtenue en utilisant un processus de séparation et évaluation. Plusieurs se sont ensuite inspirés de ces travaux, mentionnons notamment Desaulniers et al. (1997) qui utilisent un problème de plus court chemin avec contraintes de ressources et résolvent le problème à l'optimalité. L'algorithme de séparation et évaluation progressive est alors basé sur une extension du principe de décomposition de Dantzig-Wolfe (1960) permettant d'évaluer une borne inférieure resserrée. Contrairement aux travaux précédents, les fonctions de coût non-linéaires peuvent ainsi être modélisées. Vance et al. (1997) ont présenté une nouvelle formulation basée sur les services de vol (journées de travail composées de segments de vol, connexions et repos). L'approche de résolution en est une de génération de colonnes où chaque colonne correspond à un ensemble de services de vol qui établit une partition de l'ensemble des segments de vol à couvrir. Cette décomposition selon de "bons" services de vol permet d'obtenir une meilleure borne inférieure pour la solution entière (en fait la relaxation linéaire donne même dans certains cas une solution entière optimale) contrairement à la formulation habituelle de partitionnement. La relaxation linéaire s'avère toutefois plus difficile à résoudre.

1.3.2 Fabrication d'horaires personnalisés : *rostering*

Lors de la fabrication des horaires mensuels, les objectifs vont varier selon la politique visée par chaque compagnie aérienne. On peut prioriser la fabrication d'horaires équitables (même charge de travail pour tous les employés) ou équilibrés (charge de travail répartie sur toute la durée du programme mensuel : Antosik 1978, Glanert 1984). Il est souhaitable également de minimiser l'écart entre les heures de vol d'un employé et la moyenne de son groupe (Nicoletti 1975, Buhr 1978, Tingley 1979, Sarra 1988). Minimiser la masse salariale peut constituer bien sûr un objectif désiré (Nicoletti 1975, Antosik 1978, Sarra 1988). La fonction objectif devrait tenter d'éviter si possible les horaires avec heures supplémentaires ou en deçà du minimum garanti. Certaines compagnies telle Air France (Gamache et al. 1999) cherchent à affecter aux employés réguliers le plus de rotations possibles afin de minimiser les rotations résiduelles.

Le problème de *rostering* peut être modélisé comme un problème de partitionnement généralisé (Ryan et Falkner 1988 ; Ryan 1992). Chaque employé doit avoir un horaire mensuel et chaque rotation doit être couverte par suffisamment d'employés. On peut générer plusieurs horaires mensuels pour chaque employé au moyen d'une heuristique. Il faut résoudre une relaxation linéaire imbriquée dans une méthode de séparation et évaluation progressive (*branch-and-bound*) pour obtenir la solution entière. L'implantation suggérée construit un très grand nombre de colonnes avant de passer à l'obtention de la solution entière du problème de partitionnement, plusieurs des colonnes ainsi générées n'étant pas essentielles à la résolution.

Se basant sur le modèle ci-haut décrit, Gamache et Soumis (1998) ont proposé une méthode de résolution optimale. La relaxation linéaire du problème de partitionnement est résolue par génération de colonnes. Les colonnes sont générées en résolvant

un problème de plus court chemin contraint sur un réseau où les arcs représentent les rotations et périodes de repos possibles pour chaque employé. Par la suite, Gamache et al. (1999) ont introduit des heuristiques dans cette approche afin de résoudre des problèmes de plus grande taille.

1.3.3 Fabrication d'horaires personnalisés avec priorité

Certaines compagnies aériennes ont développé leur propre méthode pour résoudre le problème de fabrication d'horaires personnalisés avec priorité. Citons entre autres Air Canada (Gamache et al 1998), Air Transat, Quantas (Moore, Evans et Ngo 1978), CP Air (Byrne 1988). Vu la complexité du problème, aucune méthode optimale n'a été présentée. Mis à part les travaux de Gamache et Soumis (nous y reviendrons avec le *Preferential Bidding System*), la plupart se base sur une heuristique commune. On établit d'abord des priorités sur les activités pour chaque employé. On construit ensuite séquentiellement, selon l'ordre d'ancienneté, un bon horaire pour chaque employé en choisissant l'activité disponible (non-affectée) la plus désirée, jusqu'à l'obtention d'un horaire admissible. On affecte l'horaire à l'employé courant et on élimine les rotations choisies de l'ensemble résiduel disponible. Afin d'assurer que toutes les rotations puissent être affectées à l'ensemble des employés, on vérifie lors de la construction de chaque horaire que, pour chaque jour, il y a au moins autant d'employés disponibles que de rotations à effectuer. En dernier lieu, lorsque tous les horaires sont construits, on effectue des échanges de rotations entre les horaires de façon à améliorer les pointages.

La façon de procéder ci-haut mentionnée, où l'on juxtapose les unes après les autres les rotations disponibles les plus désirées, ne garantit pas toujours l'obtention d'un horaire admissible, de même que rien n'assure que l'horaire ainsi construit soit le meilleur que l'employé puisse obtenir. Nous aborderons dans la prochaine sous-section

la méthode de résolution proposée par Gamache (1995), davantage efficace que l'heuristique de type glouton que nous venons de décrire.

1.3.4 *Preferential Bidding System*

Le *Preferential Bidding System*, ou système PBS, est un logiciel relativement récent qui a été développé par la compagnie Ad Opt Technologies pour résoudre le problème de fabrication d'horaires personnalisés avec priorité. L'élaboration du modèle initial en revient à Gamache (1995). L'approche est séquentielle. Priorisant la séniorité, on construit et l'on fixe les horaires les uns après les autres en commençant par l'employé le plus senior jusqu'à l'employé le plus junior. La méthode suggérée par Gamache donne un horaire de pointage optimal à chaque employé en considérant les horaires déjà fixés, à partir des rotations résiduelles, optimalité que l'heuristique précédente ne pouvait garantir. Le pointage d'un horaire est défini comme la somme des poids des activités le composant, le poids d'une activité dépendant des préférences spécifiques de l'employé. Il s'agit ici de résoudre pour chaque employé un problème résiduel dont l'objectif est de maximiser le pointage de l'horaire courant, les rotations déjà attribuées aux itérations précédentes étant éliminées de l'ensemble résiduel. Ainsi on construit un horaire pour chaque employé à l'aide d'un programme linéaire en nombres entiers qui affecte les rotations résiduelles en tenant compte des employés restants. La solution optimale ainsi obtenue pour l'employé courant vérifie qu'il existe des solutions (horaires) admissibles pour les employés résiduels. Ceci est résolu par génération de colonnes combinée à un procédé de séparation et évaluation progressive. Davantage de détails seront donnés au chapitre suivant.

Soulignons que les problèmes sont en pratique de très grande taille. C'est pourquoi l'implantation de la méthode est heuristique (branchement sans retour en arrière sauf si non réalisable) bien que Gamache propose une approche exacte.

Diverses améliorations à la méthode ci-haut mentionnée ont ensuite été apportées. Nous nous proposons d'en faire ici un survol.

Afin d'améliorer la vitesse d'exécution et de permettre ainsi la résolution de problèmes de plus grande taille, diverses approches ont été considérées. Jeandroz (2000) a proposé une heuristique utilisant des compteurs dynamiques reliés aux compétences requises par les employés (ex : parler une langue spécifique). Ces compteurs sont intégrés aux réseaux décrivant les chemins (un horaire étant vu comme un chemin). Pour chaque employé, en commençant par le plus senior, on résout un problème de plus court chemin. Ce chemin identifie l'horaire à attribuer à l'employé courant. On arrête cette procédure lorsque les compteurs indiquent que le problème devient très restreint et que l'affectation du meilleur horaire possible à un employé peut entraîner un problème résiduel non-réalisable. Un premier type de compteur évalue l'offre et la demande pour chaque compétence et peut forcer l'employé courant à effectuer une tâche précise dans un intervalle donné, si l'offre égale la demande et si l'employé est qualifié et disponible. Un deuxième type de compteur compile, pour chaque compétence, les crédits de vol des tâches résiduelles et compare avec la disponibilité (en crédits) des employés qualifiés. Ceci détermine le critère d'arrêt de l'heuristique. L'approche développée par Gamache (1995) est ensuite utilisée pour terminer la résolution. Avec cette amélioration, qui réduit considérablement le temps de calcul (un seul plus court chemin pour chaque employé), la complexité est pseudo-polynomiale pour les employés les plus seniors au lieu d'être exponentielle (avec Gencol, le logiciel de génération de colonnes utilisé). Pour les plus gros problèmes (300 employés et plus), le temps de calcul est réduit d'un facteur de 10 à 15. Les améliorations apportées par El Idrissi (2002) permettent de résoudre des problèmes de plus de mille employés avec des temps de calcul raisonnables.

1.4 Contribution

Notre étude vise à améliorer la qualité des horaires des employés juniors s'inscrivant dans le contexte du logiciel *Preferential Bidding System*. Pour ce faire, nous proposons d'abord plusieurs algorithmes d'énumération pour dénombrer les horaires ayant le pointage recherché pour un employé donné. Nous présentons ensuite deux heuristiques pour l'évaluation de la pénalité de ces horaires sur les employés suivants. Ces outils sont utilisés pour modifier la méthode de branchement standard afin qu'elle puisse permettre l'attribution à chaque employé d'un horaire de pointage optimal qui prend en compte des préférences des employés juniors. Finalement, nous exposons les résultats des tests effectués afin d'évaluer l'impact de ce nouveau branchement au niveau de la qualité des horaires des employés juniors de même que des temps de résolution. Ces résultats démontrent que les outils développés permettent d'améliorer de façon significative la qualité des horaires des employés juniors.

Chapitre 2

L'approche PBS

Ce chapitre présente une description générale de l'approche de résolution pour la construction d'horaires personnalisés avec priorité, telle que développée par Gamache et al. (1998). En guise de mise en contexte, nous exposerons d'abord le problème dans son détail.

2.1 Description du problème

Dans le domaine du transport aérien, le problème de fabrication d'horaires personnalisés avec priorité consiste à déterminer, pour un horizon d'un mois, l'horaire de chaque employé (navigant) en tenant compte de ses préférences ainsi que des activités pré-assignées. Nous considérons ici le personnel technique seulement : les pilotes ainsi que les premiers et seconds officiers. Notons que le problème pour les agents de bord s'avère quelque peu différent en ce qu'il est plus gros (davantage d'agents de bord, davantage de tâches à couvrir) et requiert n employés par rotation (au lieu d'un seul pour le personnel technique).

Le problème de grande taille qui nous intéresse peut se séparer en plusieurs sous-problèmes indépendants. D'abord, les problèmes pour les pilotes, les premiers et seconds officiers se résolvent séparément. Une autre ramification s'effectue ensuite selon le type d'appareil, chaque employé technique n'étant qualifié pour voler que sur

un seul type. Enfin, on peut diviser encore les problèmes selon la base à laquelle les employés sont rattachés.

Les rotations à couvrir sont connues, de même que leurs caractéristiques : horaires et destinations des vols composant chaque rotation, qualifications requises par les employés (par exemple, un vol peut requérir qu'au moins un membre du personnel technique parle une langue donnée). Les qualifications et restrictions (s'il y a lieu) de chaque employé sont connues, de même que ses préférences. Chaque employé a précisé ses préférences spécifiques au moyen d'un système de pointage bien précis. Ces préférences sont de nature variée : il peut s'agir d'un congé désiré, d'une destination de prédilection, de ne pas avoir de vol avant certaines heures le matin, ou encore de voler en compagnie d'un autre officier en particulier, etc. Pour une activité désirée, l'employé accordera un pointage plus ou moins élevé selon le degré d'importance, de même pour les activités indésirées (en ce cas, les pointages seront négatifs).

Il s'agit de construire un horaire pour chaque employé, avec comme fonction objectif la maximisation du pointage de cet horaire. Le pointage d'un horaire sera obtenu en sommant le pointage des activités le constituant. En d'autres termes, on cherche à maximiser la satisfaction des employés. Bien entendu, la maximisation du pointage d'un employé peut être en conflit avec la maximisation du pointage d'un autre employé. L'objectif poursuivi sera de maximiser la satisfaction de chaque employé par ordre de séniorité : pour ce faire, on procédera de façon séquentielle.

2.1.1 Description des contraintes

Un horaire admissible doit respecter différentes contraintes que nous nous proposons d'énumérer ici-bas. Il est à noter que les valeurs numériques indiquées ci-bas sont spécifiques à Air Canada.

Contraintes locales. Les contraintes locales régissent la construction de chaque horaire. Ces contraintes proviennent de la convention collective des employés (propre à chaque compagnie aérienne) et des règles de sécurité aérienne :

Crédits de vol : les crédits de vol sont les heures de vol ou l'équivalent pour des activités telles les périodes d'entraînement. La somme S de ces crédits de vol doit être comprise dans un intervalle donné, soit $S \in [v_1, v_2]$. On permet également des horaires réduits avec $S \in [t_1, v_1[$. Les paramètres v_1 et v_2 sont à prédéterminer.

Jours de travail entre deux périodes de repos : le nombre de jours de travail consécutifs entre les périodes de repos doit être d'au plus une certaine borne supérieure, soit $\leq t_2$.

Indice de fatigue : cet indice est évalué par un pointage qui augmente lors de la couverture d'une rotation et diminue lors des périodes de repos. L'indice doit être inférieur à une certaine borne spécifique au début de chaque rotation et ne peut excéder une borne supérieure fixe t_3 durant le mois.

Périodes de repos : chaque employé doit avoir un nombre minimal j de jours de repos durant le mois, avec au moins une séquence particulière (3x72 h, 4x48 h, 2x96 h).

Un *horaire normal* respecte toutes ces contraintes, tandis qu'un *horaire court* contient un nombre de crédits de vol inférieur au seuil minimal v_1 , i.e. $S \in [t_1, v_1]$ (mais toutes les autres contraintes sont respectées). Si on ne peut respecter toutes les contraintes pour un employé donné, on lui assignera alors un *horaire vide* et cet employé fera alors partie du personnel de réserve. Enfin, les *horaires ouverts* contiennent toutes les rotations n'ayant pu être affectées aux employés et n'ont pas à vérifier toutes les contraintes ci-haut mentionnées : ils serviront pour le personnel de réserve.

Contraintes globales. Les contraintes globales concernent la composition de l'ensemble des horaires.

Partition : chaque rotation doit être attribuée à un seul employé.

Affectation : chaque employé se voit assigner un horaire normal, court ou vide.

Horaires courts : la solution ne peut contenir plus de $t_4 = 2$ horaires courts.

Horaires ouverts : la solution ne peut contenir plus de $t_5 = 2$ horaires ouverts.

Temps ouverts : les rotations non couvertes ne peuvent excéder $t_6 = 120$ heures.

Composition des équipages. La composition du personnel technique devant couvrir une rotation doit respecter quelques règles. Par exemple, il ne peut y avoir plus d'un membre d'équipage avec une restriction médicale. Puisque les problèmes pour chaque catégorie d'employés sont résolus en ordre hiérarchique (pilotes, premiers et seconds officiers), les contraintes de composition d'équipage seront imposées aux problèmes des premiers et seconds officiers si besoin est.

Après résolution du problème, chaque employé obtiendra un horaire. Ceci permettra à la compagnie aérienne de connaître la composition de l'équipage et du personnel de réserve affilié à chaque vol.

2.2 Modélisation

Les horaires seront construits séquentiellement selon l'ordre d'ancienneté des employés, il s'agit d'attribuer à chacun un horaire admissible de pointage maximal, s'assurant que les rotations résiduelles puissent être effectuées par le personnel résiduel. Pour chaque employé, il s'agira de résoudre un nouveau problème résiduel.

Nous reprendrons ici la formulation telle que proposée par Gamache et al. (1998). Soit P le problème que nous désirons résoudre et m le nombre d'employés. Soit

$m + 1$ l'indice pour les horaires ouverts, considéré comme l'indice d'un employé fictif. Supposons que des horaires ont déjà été attribués aux $k - 1$ employés les plus seniors, alors nous devons résoudre un problème résiduel en nombres entiers IP^k , $1 \leq k \leq m + 1$ pour trouver l'horaire de meilleur pointage du k^{ieme} employé.

Soit P^k l'ensemble des rotations résiduelles (i.e. qui n'ont pas été attribuées aux employés 1 à $k - 1$) et Ω_e^k l'ensemble des horaires résiduels (i.e. les horaires ne contenant que les rotations résiduelles) pour l'employé e , $k \leq e \leq m + 1$, avec $\Omega^k = \bigcup_{e=k}^{m+1} \Omega_e^k$. Soit également t_4^k , le nombre d'horaires courts résiduels pouvant être construits, avec $t_4^1 = t_4 = 2$ (au plus 2 horaires courts au total). Les paramètres sont définis comme suit :

c_s = pointage de l'horaire s si $s \in \Omega_k^k$, 0 sinon ;

$a_{ps} = 1$ si l'horaire s inclut la rotation p , 0 sinon ;

$l_s = 1$ si l'horaire s est court, 0 sinon ;

d_s = durée de l'horaire s s'il est ouvert, 0 sinon.

Un seul type de variables de décision est utilisé, soit :

$x_s = 1$ si l'horaire s est choisi, 0 sinon.

Au k^{ieme} employé, la formulation mathématique du problème résiduel IP^k s'énonce alors comme suit :

$$(IP^k) \quad \max \sum_{s \in \Omega_k^k} c_s x_s \quad (2.1)$$

sous les contraintes :

$$\sum_{s \in \Omega^k} a_{ps} x_s = 1, \quad \forall p \in P^k \quad (2.2)$$

$$\sum_{s \in \Omega_e^k} x_s = 1, \quad e = k, k+1, \dots, m \quad (2.3)$$

$$\sum_{s \in \Omega^k} l_s x_s \leq t_4^k \quad (2.4)$$

$$\sum_{s \in \Omega_{m+1}^k} x_s \leq t_5 \quad (2.5)$$

$$\sum_{s \in \Omega_{m+1}^k} d_s x_s \leq t_6 \quad (2.6)$$

$$x_s \in \{0, 1\}, \quad \forall s \in \Omega^k. \quad (2.7)$$

Le problème résiduel IP^k admet donc comme objectif de maximiser le pointage de l'horaire de l'employé k (2.1), tout en respectant les contraintes globales : chaque rotation résiduelle doit être couverte par un horaire (2.2) ; chaque employé se voit attribuer un horaire admissible (2.3) ; le nombre d'horaires courts choisis n'excède pas le nombre permis restant t_4^k (2.4) ; le nombre et la durée totale des horaires ouverts respectent les limites maximales imposées t_5 et t_6 (2.5)-(2.6). À ces contraintes globales s'ajoutent les contraintes d'intégrité (2.7).

Ce modèle est du type partitionnement d'ensemble et comporte en pratique un très grand nombre de variables binaires.

2.2.1 Réseaux et ressources

Notons qu'un horaire r peut être représenté par un chemin (allant du noeud source au noeud puits) dans un réseau. En effet, les différentes activités possibles (ex : rotations, repos, tâches fixes) peuvent être représentées au moyen d'arcs dans un réseau général,

arcs sur lesquels sont indiquées les consommations de ressources associées (crédits de vol, indice de fatigue, etc.). Pour chaque problème (employé k) à résoudre, il y aura un réseau particulier créé à partir du réseau global contenant tous les arcs. Un exemple de réseau est illustré à la figure 2.1 ; se référer à Gamache (1995) pour plus de détails sur la conception du réseau.

Ce réseau particulier tiendra compte des préaffectations (les arcs inutiles seront ainsi supprimés) et des préférences de l'employé courant. Les coûts personnalisés correspondant aux préférences de l'employé courant seront indiqués sur chaque arc de ce réseau. De même, puisque certains navigants peuvent indiquer leur préférence pour voler en compagnie d'un officier particulier, les contraintes correspondantes seront intégrées aux réseaux des officiers inférieurs concernés.

Le pointage d'un chemin (horaire) sera la somme des coûts associés aux préférences se trouvant sur les arcs constituant le chemin en question. Trouver l'horaire de pointage maximal pour un employé donné revient alors à résoudre un problème de plus long chemin dans ce réseau, auquel des variables duales auront été ajoutées (explications plus loin), tout en respectant les contraintes de ressources.

Un horaire admissible devra satisfaire différentes contraintes élémentaires :

1. deux rotations dans le même horaire ne peuvent se chevaucher ;
2. si le navigant a une restriction médicale, son réseau ne contiendra pas de rotations attribuées à un officier supérieur avec restriction médicale ;
3. les préaffectations font nécessairement partie de l'horaire.

Les contraintes locales sont représentées au moyen de ressources pour lesquelles des bornes à respecter seront indiquées sur les noeuds. Lorsqu'un chemin passe sur un arc donné, la valeur courante au noeud de départ pour chaque ressource se voit

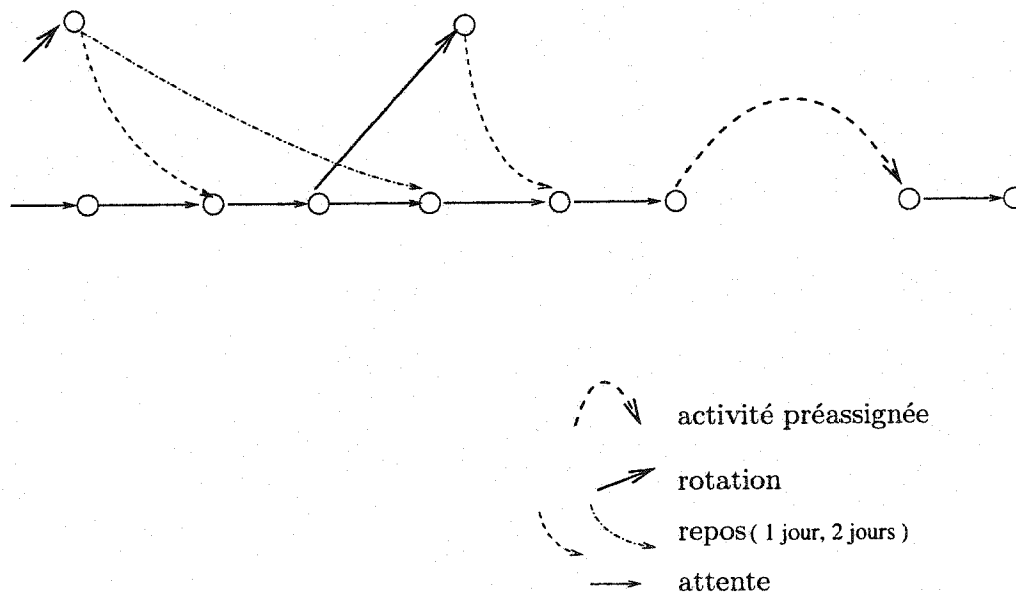


Figure 2.1 – Fragment de réseau

augmentée de la consommation indiquée sur l'arc. Les nouvelles valeurs de ressource devront respecter les bornes indiquées au noeud arrivée. Nous devons considérer les ressources suivantes.

Repos : On veut vérifier que l'employé reçoit le nombre de jours de repos j auxquels il a droit durant le mois. Pour ce faire, une ressource cumule le nombre de jours de repos déjà attribués à l'employé, une autre ressource détermine le patron de repos (3x72 h, 4x48 h, 2x96 h).

Amplitude entre les repos : Entre deux repos, il ne peut y avoir plus de t_2 jours de travail. Une ressource calcule donc le nombre de jours de travail depuis le dernier repos. On vérifie à chaque noeud que cette ressource se situe entre 0 et t_2 . Après chaque repos, on remet la ressource à zéro.

Crédits de vol : Le nombre de crédits de vol accumulés depuis le début du mois est représenté par une ressource. À chaque noeud début de tâche, on vérifie que les limites supérieures modifiées sont respectées, la limite inférieure n'étant vérifiée qu'au noeud puits.

Indice de fatigue : On doit s'assurer que la ressource d'indice de fatigue est inférieure à la borne supérieure permise au début de chaque rotation. En tout temps, l'indice maximal t_3 ne peut être dépassé. Les arcs de rotation font augmenter l'indice courant, tandis que les repos diminuent cet indice.

2.3 Approche de résolution existante

Tel que mentionné plus haut, l'approche de résolution est séquentielle : on résout employé par employé, du plus ancien au plus junior. On doit résoudre à chaque itération un problème de partitionnement en nombres entiers (IP_k). Pour se faire, on utilise le logiciel d'optimisation GENCOL, une implantation informatique d'une approche de génération de colonnes imbriquée dans une méthode de séparation et évaluation progressive. Ce logiciel traitera alternativement un problème maître et un sous-problème jusqu'à l'obtention d'un horaire optimal. Le problème maître correspond au problème résiduel précédemment mentionné. L'objectif est de trouver un horaire optimal pour l'employé courant tout en considérant un sous-ensemble d'horaires réalisables ainsi que les contraintes globales. La stratégie de résolution consistera à relaxer d'abord les contraintes d'intégrité (2.7). Le problème linéairement relaxé (LP_k) sera résolu par génération de colonnes, ces dernières étant générées en résolvant, pour chaque employé, un problème de plus long chemin avec contraintes de ressources (explication plus détaillée au chapitre suivant). La figure 2.2 illustre le fonctionnement de GENCOL.

Pour chaque employé, le sous-problème reçoit les variables duales du problème maître et fournit en retour des horaires réalisables de coût marginal positif. Ces horaires définissent des colonnes qui seront ajoutées au problème maître pour l'itération suivante. On poursuit ainsi jusqu'à ce qu'il n'y ait plus de colonnes à coût marginal positif

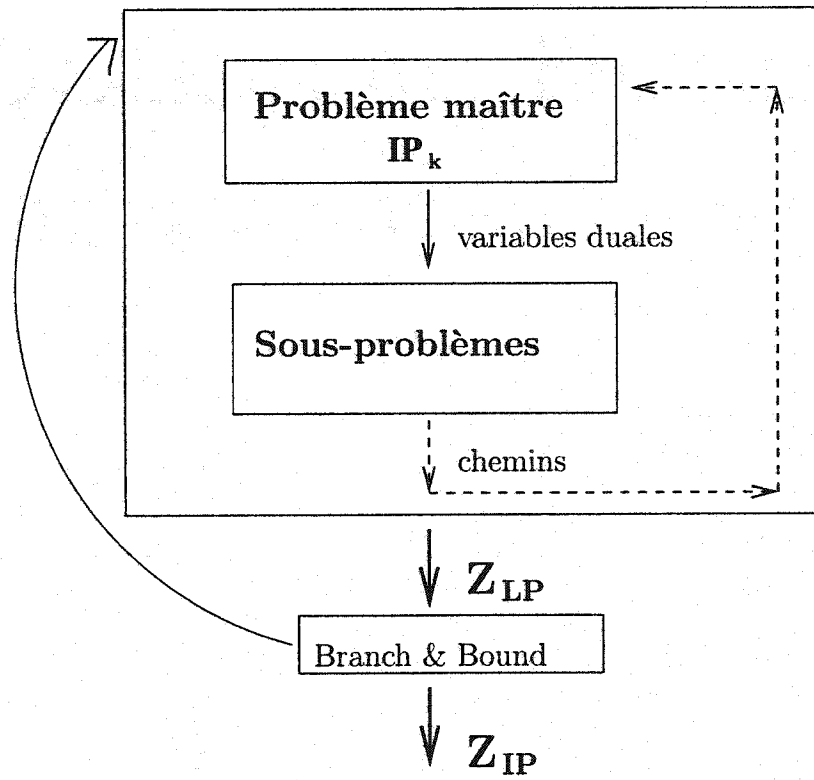


Figure 2.2 – Fonctionnement de GENCOL

pouvant être ajoutées. À partir de la solution relaxée, on obtiendra par la suite une solution entière au moyen d'un processus de séparation et évaluation progressive ou *branch-and-bound*. Ce processus fera appel à la même technique de génération de colonnes à tous les noeuds de branchement pour ajouter, au besoin, de nouveaux horaires.

2.3.1 Coupes et branchement interne

Lorsque l'on résout le problème IP_k , les contraintes d'intégrité sont d'abord relâchées pour les employés suivants. Ceci revient à résoudre un problème mixte MIP_k . Pour l'employé k , la solution de la relaxation linéaire du problème MIP_k peut s'écrire comme une combinaison linéaire de colonnes pouvant avoir des pointages différents. Puisque LP_k est une relaxation linéaire de MIP_k , alors le pointage du meilleur horaire de l'employé k devra être inférieur ou égal au pointage de la solution LP_k pour cet employé, pour une solution optimale en nombres entiers. On réduit le domaine des solutions réalisables en éliminant tous les horaires dont le pointage est supérieur à celui de la relaxation linéaire. Cette réduction est faite en imposant des coupes dans le sous-problème de plus long chemin avec contraintes associé à l'employé courant afin d'interdire la génération de colonnes dont le coût est supérieur à celui de la solution de la relaxation linéaire. Ce type de coupe est imposé au niveau du sous-problème en ajoutant une ressource comptabilisant le pointage total.

Si la solution optimale pour l'employé k contient un seul horaire à pointage maximum après avoir imposé les coupes, cette solution sera alors aussi celle du MIP_k . On peut alors fixer l'horaire de l'employé k et passer au problème MIP_{k+1} .

Cependant, si la solution ainsi trouvée contient une combinaison convexe de plusieurs horaires de même coût, alors on doit imposer une décision de branchement. Il s'agit

d'affecter sur une branche à l'employé k la rotation p qui maximise $\sum_{s \in \Omega_k^*} a_{ps} x_s$, i.e. la rotation de flot maximal dans la solution fractionnaire courante. Le contraire (interdire cette rotation) sera indiqué sur l'autre branche. Ces décisions sont imposées au niveau des sous-problèmes en retirant des réseaux appropriés les arcs représentant la rotation p choisie. On continue d'explorer l'arbre de branchement interne du problème k avec la stratégie du meilleur d'abord jusqu'à l'obtention d'une solution heuristique entière de bonne qualité pour le problème MIP_k . Cette solution sera heuristique puisque la dominance employée dans la résolution des sous-problèmes est inexacte.

2.3.2 Branchement externe

Lors de la résolution du problème de l'employé courant, les contraintes d'intégrité sont relaxées pour les employés suivants, et c'est l'exploration de l'arbre de branchement externe qui permettra d'assurer l'obtention de solutions réalisables entières pour tous. La figure 2.3 illustre le processus de branchement externe.

Si la solution du problème MIP_k est réalisable, on fixe alors l'horaire ainsi trouvé pour l'employé courant et l'on continue en résolvant le MIP_{k+1} , en poursuivant ainsi jusqu'à l'employé m . Par contre, si un MIP_k devient non réalisable, alors on sait qu'un horaire n'aurait pas dû être fixé pour l'un des précédents employés. Afin d'obtenir une solution réalisable, nous devons remonter dans l'arbre. Si IP_{k-1} n'a pas été résolu, on le résout à la place de MIP_{k-1} et l'on revient à MIP_k avec un nouvel horaire fixé pour l'employé $k - 1$. Sinon, si IP_{k-2} n'a pas encore été résolu, on répète à nouveau la procédure. Ceci revient à remonter dans l'arbre de branchement externe où les horaires fixés des employés 1 à k correspondent aux noeuds. Cet arbre de branchement permet d'assurer l'obtention de solutions réalisables entières pour les problèmes IP_1 à IP_{m+1} .

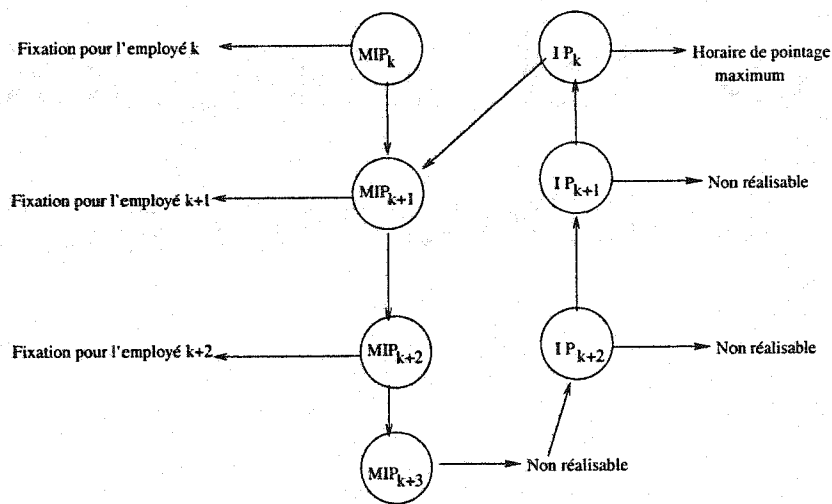


Figure 2.3 – Arbre de branchement externe

Ce type de branchement externe comporte certes des faiblesses. Il s'agit d'une heuristique et le choix de l'horaire pour chaque employé est arbitraire. En effet, s'il existe plusieurs horaires ayant le meilleur pointage, pourquoi en choisir un plutôt qu'un autre ? Ce mémoire tente justement de remédier à ces faiblesses. Dans le cas où il y a plusieurs horaires de meilleur pointage pour un employé donné, nous allons tenter de déterminer parmi ceux-ci lequel favorise davantage les employés suivants. En fait, nous allons trier tous les horaires de meilleur pointage en termes des préférences des employés suivants, et fixer d'abord le choix sur l'horaire le moins préjudiciable à ces derniers. Davantage de détails sont donnés dans les chapitres suivants.

Chapitre 3

Algorithmes d'énumération des chemins de même pointage

Dans l'algorithme modifié de résolution que nous proposerons au chapitre suivant, il faut énumérer tous les horaires d'un pointage donné, ce dernier ne correspondant peut-être pas au pointage maximal pour l'employé courant. Dans ce chapitre, nous considérerons différents algorithmes pour énumérer ces chemins de même pointage. Dans un premier temps, la notation associée au réseau sera présentée. Ceci sera suivi d'un bref résumé concernant les problèmes de plus court chemin avec contraintes de ressources. Ces prémisses devraient permettre de mieux caractériser notre problématique d'énumération de chemins dans un graphe. Le cas avec fonctions de prolongation linéaires sera d'abord abordé, avec les deux algorithmes que nous proposons pour l'énumération dans ce cas. La démarche inhérente à chaque algorithme sera également justifiée et exposée, en s'appuyant sur les concepts de dominance. Enfin, le cas avec fonctions de prolongation non linéaires sera étudié et trois méthodes de résolution seront exposées.

3.1 Notation associée au réseau

Pour l'employé courant k , le problème relaxé est résolu par génération de colonnes et coupes jusqu'à l'obtention d'une solution entière de coût z_{IP} . Soit le pointage P ainsi obtenu pour cet employé.

Pour l'employé k , nous avons un réseau acyclique $G = (N, A)$ comportant $|R|$ ressources où R dénote l'ensemble des ressources. L'ensemble des noeuds, N , est trié en ordre topologique et numéroté de 1 à n . La valeur de la ressource $r \in R$ au noeud $i \in N$ est restreinte à l'intervalle $[a_i^r, b_i^r]$, des temps d'attente étant toutefois permis avant la borne minimale. Pour chaque arc $(i, j) \in A$, t_{ij}^r donne la consommation de la ressource r sur cet arc, c_{ij} étant le coût associé dans le cas linéaire. Soit le flot x_{ij} , ce flot sera égal à 1 si l'horaire considéré inclut l'arc (i, j) , sinon le flot sera nul. Soit également T_i^r , la quantité de ressource consommée r cumulée au noeud i depuis le noeud origine 1, et $\mathbf{T}_i = (T_i^r | r \in R)$, le vecteur regroupant ces quantités au noeud i .

Par conséquent, un chemin entre l'origine 1 et la destination n qui satisfait les contraintes de ressources doit satisfaire les contraintes suivantes :

$$\sum_{j:(1,j) \in A} x_{1j} = 1 \quad (3.1)$$

$$\sum_{i:(i,n) \in A} x_{in} = 1 \quad (3.2)$$

$$\sum_{i:(i,j) \in A} x_{ij} - \sum_{i:(j,i) \in A} x_{ji} = 0, \quad \forall j \in N \setminus \{1, n\} \quad (3.3)$$

$$x_{ij}(f_{ij}^r(\mathbf{T}_i) - T_j^r) \leq 0, \quad \forall (i, j) \in A, \forall r \in R \quad (3.4)$$

$$a_i^r \leq T_i^r \leq b_i^r, \quad \forall i \in N, \forall r \in R \quad (3.5)$$

$$x_{ij} \in \{0, 1\}, \quad \forall (i, j) \in A. \quad (3.6)$$

Les contraintes (3.1) - (3.3) fournissent la structure de chemin entre les noeuds i et n . Les contraintes (3.4) permettent de définir une borne inférieure sur les variables de ressource T_j^r lorsque l'arc (i, j) est utilisé, i.e. lorsque $x_{ij} = 1$. Cette borne inférieure se calcule à l'aide d'une fonction de prolongation $f_{ij}^r(\mathbf{T}_i)$ qui peut être linéaire ou non linéaire. Dans le cas linéaire, $f_{ij}^r(\mathbf{T}_i) = T_i^r + t_{ij}^r$. Les fenêtres de ressources sont données par (3.5) et les contraintes (3.6) restreignent les variables de flot x_{ij} à des valeurs binaires.

Nous voulons énumérer tous les chemins réalisables dans G , allant du noeud source 1 au noeud puits n , dont le coût est égal au pointage P déjà prédéterminé. Le coût d'un chemin se calcule aussi à l'aide de variables $z_{ij}, i \in N$, semblables aux variables de ressource, et de fonction de prolongation $z_{ij}(z_i, \mathbf{T}_i), (i, j) \in A$. Dans le cas linéaire, $z_{ij}(z_i, \mathbf{T}_i) = z_i + c_{ij}$. Les équations suivantes s'ajoutent donc à la formulation (3.1) - (3.6) :

$$x_{ij}(z_{ij}(z_i, \mathbf{T}_i) - z_j) = 0, \quad \forall (i, j) \in A \quad (3.7)$$

$$z_1 = 0 \quad (3.8)$$

Les chemins recherchés sont donc ceux pour lesquels $z_n = P$. Notons en passant que P n'est pas nécessairement le pointage maximal associé à un chemin du réseau G .

3.2 Problème de plus court chemin avec contraintes de ressources

Dans l'énoncé de modélisation au chapitre 2, nous avons présenté le problème de maximisation des pointages des employés, conformément à l'usage théorique courant. Cependant, puisque ce problème a été implanté en pratique comme un problème de minimisation (en inversant les signes sur les arcs dans le réseaux), nous parlerons à partir de maintenant en termes de minimisation. Dans les algorithmes proposés (section 3.3), on utilise l'algorithme de plus court chemin avec contraintes de ressources pour générer les étiquettes des chemins de coût minimal. Il importe donc de bien comprendre cet algorithme. Faisons un survol de la littérature sur le sujet et regardons plus en détails certains éléments.

3.2.1 Revue de littérature

L'énumération des horaires de pointage minimal se fait à l'aide d'un algorithme de plus court chemin (si l'on considère la minimisation des pénalités plutôt que la maximisation des préférences) sur le réseau de l'employé courant. Pour tenir compte des nombreuses contraintes locales à respecter, on ajoute des ressources au réseau (voir détails au chapitre 2). Le problème de plus court chemin avec ressources fut initialement introduit par Desrochers en 1988. D'autres articles ont été écrits par la suite sur le sujet : Desrochers et Soumis (1988), Desrosiers et al. (1995), Desaulniers et al. (1997). Ces articles abordent entre autres les concepts d'étiquettes et de dominance, concepts sur lesquels nous reviendrons en détail plus bas. L'algorithme de plus court chemin énumère ceux-ci à l'aide d'étiquettes et tient compte de la consommation de ressource (un chemin est valide s'il respecte les bornes de ressource en chaque noeud). On considère ici des fonctions de prolongation non décroissantes. Spécifions ainsi que par définition, on ne peut avoir de chemins valides de coût inférieur à celui d'un plus court chemin.

Plus récemment, Vovor (1998) a proposé une version plus générale du problème de plus court chemin avec contraintes de ressources dans un graphe acyclique. Dans cette dernière version, on permet entre autres des fonctions non décroissantes pour l'extension des consommations de ressources. Des algorithmes pseudo-polynomiaux sont proposés, basés sur la programmation dynamique et sur une approche en deux phases. Cette méthode, en deux phases, s'avère très efficace pour traiter les cas de réoptimisation, par exemple si certains noeuds sont supprimés ou fixés, ou encore si les coûts sont modifiés. La première phase s'avère très coûteuse. Cependant, seule la deuxième phase (requérant peu de calculs) est nécessaire dans les cas de réoptimisation.

3.2.2 Quelques concepts

Cette sous-section présente les principaux concepts sur lesquels se base l'algorithme de plus court chemin avec contraintes de ressource. Nous nous référons ici à la version utilisée par le logiciel Gencol pour le cas d'un réseau acyclique (Desrosiers et al., 1995).

Un ensemble de ressources R est utilisé pour modéliser les contraintes locales. Pour un employé courant donné, l'état de ces ressources en un noeud i est représenté par le vecteur $\mathbf{T}_i$ des variables de ressource $T_i^r, r \in R$. Les variables de ressource sont nécessaires pour déterminer les chemins réalisables dans chaque graphe G .

Une étiquette multi-dimensionnelle, $\mathbf{E}_i = (z_i, \mathbf{T}_i)$, est associée à chaque chemin s_i , avec z_i représentant le coût de s_i . Cette étiquette est utilisée pour comparer s_i avec les autres chemins partiels se terminant au noeud i . Une étiquette $\mathbf{E}_i$ associée à un chemin réalisable doit être telle que $T_i^r \in [a_i^r, b_i^r], \forall r \in R$. Les étiquettes non réalisables sont éliminées durant le processus de résolution de l'algorithme. Puisque plusieurs chemins réalisables peuvent se rendre au même noeud, une liste d'étiquettes est ainsi associée à chaque noeud.

Un nouveau chemin partiel est créé en ajoutant un arc à un chemin partiel déjà existant. Les fonctions de prolongation sont nécessaires pour calculer la valeur de l'étiquette associée à ce nouveau chemin. La valeur de cette nouvelle étiquette dépend de la valeur de l'étiquette du chemin déjà existant ainsi que de la consommation de ressource sur l'arc. Il y a ainsi une fonction-type de prolongation pour le coût et une autre pour les variables de ressource. Ces fonctions d'extension type dépendent ainsi de l'arc ajouté au chemin existant. Étant donné l'étiquette $\mathbf{E}_i = (z_i, \mathbf{T}_i)$ associée à un chemin partiel se terminant au noeud i , l'étiquette $\mathbf{E}_j = (z_j, \mathbf{T}_j)$ obtenue en prolongeant le premier chemin avec l'arc (i, j) est déterminée comme suit : $T_j^r =$

$\max\{a_j^r, f_{ij}^r(\mathbf{T}_i)\}$ pour les ressources et $z_j = z_{ij}(z_i, \mathbf{T}_i)$ pour le coût. Il est à noter que le chemin prolongé est valide seulement si $f_{ij}^r(\mathbf{T}_i) \leq b_j^r, \forall r \in R$.

Tel que mentionné plus haut, une liste d'étiquettes est donc associée à chaque noeud. Cependant, ce ne sont pas tous les chemins partiels qui permettront ensuite d'obtenir des plus courts chemins. Une règle de dominance est donc appliquée afin de réduire la taille de la liste d'étiquettes en chaque noeud. Soit $\mathbf{E}_i^1 = (z_i^1, \mathbf{T}_i^1)$ et $\mathbf{E}_i^2 = (z_i^2, \mathbf{T}_i^2)$, deux étiquettes distinctes associées à des chemins se terminant au noeud i . La première étiquette *domine* la seconde, i.e. $\mathbf{E}_i^1 \prec \mathbf{E}_i^2$, si et seulement si $z_i^1 \leq z_i^2$ et $T_i^{1,r} \leq T_i^{2,r}, \forall r \in R$.

Chaque ressource définie ajoute une dimension aux étiquettes. La complexité de l'algorithme ainsi que le temps de résolution augmenteront donc proportionnellement au nombre de ressources sur lesquelles s'effectue la dominance. Un exemple de pseudo-code de l'algorithme est donné ci-dessous (Algorithme 1) ; rappelons que nous considérons toujours le cas acyclique.

Algorithm 1 Plus court chemin avec contraintes de ressources

```

1:  $L_1 = \{(0, (a_1^r)_{r \in R})\}$ 
2: pour tout  $j \in N$  dans un ordre topologique effectuer
3:    $L_j \leftarrow \emptyset$ 
4:   pour tout  $i \mid (i, j) \in A$  effectuer
5:     pour tout  $(z_i, \mathbf{T}_i) \in L_i$  effectuer
6:       si  $f_{ij}^r(\mathbf{T}_i) \leq b_j^r, \forall r \in R$  alors
7:          $L_j \leftarrow L_j \cup \{(z_{ij}(z_i, \mathbf{T}_i), (\max\{a_j^r, f_{ij}^r(\mathbf{T}_i)\})_{r \in R})\}$ 
8:    $L_j \leftarrow EFF(L_j)$ 

```

On prolonge ainsi des chemins réalisables à partir du noeud source 1, selon une méthode de tirage ('pulling'). En chaque noeud j , il y aura ainsi un ensemble L_j d'étiquettes valides. Seules les étiquettes non dominées sont conservées. On utilise une procédure multi-dimensionnelle $EFF(\cdot)$ pour faire la sélection selon la dominance.

Cette procédure est relativement compliquée et requiert des structures de données particulières. Des exemples de dominance sont présentés plus loin, dans le cas simple avec une seule ressource.

Remarquons que, lorsque les fonctions de prolongations $z_{ij}(z_i, \mathbf{T}_i)$ sont linéaires, nous pourrions procéder de même avec un problème de plus long chemin, en inversant le signe des coûts sur les arcs et en résolvant ensuite comme un problème de plus court chemin. Comme nous allons utiliser à la fois des étiquettes provenant d'algorithmes de plus court et plus long chemin, nous utiliserons le souligné pour les étiquettes de plus court chemin $\underline{E}_i = (\underline{z}_i, \underline{\mathbf{T}}_i)$ et le surligné pour les étiquettes de plus long chemin $\overline{E}_i = (\overline{z}_i, \overline{\mathbf{T}}_i)$.

3.3 Fonctions de prolongation linéaires

Dans le cas linéaire, les fonctions de prolongation dans le réseau G s'expriment sous une forme simplifiée : $f_{ij}^r(\mathbf{T}_i) = T_i^r + t_{ij}^r$ pour la consommation de ressources et $z_{ij}(z_i, \mathbf{T}_i) = z_i + c_{ij}$ pour le coût. De telles fonctions permettent de faire marche arrière de $\mathbf{T}_j$ à $\mathbf{T}_i$ puisqu'il s'agit alors d'effectuer une simple soustraction.

Dans cette section, nous proposons deux algorithmes pour énumérer tous les chemins de pointage P pour l'employé k , dans le cas où les fonctions de prolongation sont linéaires. Ces algorithmes utilisent une procédure d'étiquetage arrière, i.e. du noeud n vers le noeud 1. Afin d'éliminer des étiquettes ne pouvant mener à des chemins de longueur P , des bornes de comparaison sont préalablement déduites à partir de la résolution de problèmes de plus court et plus long chemin avec fenêtres de ressources sur G . Le premier algorithme fait appel uniquement aux coûts c_{ij} associés aux arcs de G tandis que le deuxième algorithme repose sur des coûts réduits $\tilde{c}_{ij}$ associés à ces arcs. Afin d'alléger le texte, nous parlerons dans la suite du présent document

de problèmes de plus court et plus long chemins, sous-entendant implicitement qu'il s'agit de problèmes avec contraintes de ressources.

3.3.1 Algorithme basé sur les coûts

Le premier algorithme procède en quatre étapes :

1. Résoudre un problème de plus court chemin entre les noeuds 1 et n du réseau considérant les coûts c_{ij} .
2. Résoudre un problème de plus long chemin entre les noeuds 1 et n du réseau considérant les coûts $-c_{ij}$.
3. Utiliser une procédure d'étiquetage arrière (i.e du noeud n au noeud 1) décrite ci-bas pour trouver tous les chemins de pointage P entre 1 et n dans le réseau G .
4. Retenir les chemins de coût nul.

Les bornes obtenues par l'algorithme de plus court chemin s'expriment, à chaque noeud i , à l'aide de $\underline{ne}_i$ étiquettes triées par ordre lexicographique selon leur coût et consommation de ressources : $(\underline{z}_i^l, \underline{T}_i^l), l = 1, \dots, \underline{ne}_i$, où $\underline{T}_i^l = (\underline{T}_i^{l,r} | r \in R)$. Par conséquent, le coût minimum d'un chemin entre 1 et i est donné par $\underline{z}_i^1$. De même, les bornes obtenues par l'algorithme de plus long chemin s'expriment à chaque noeud i à l'aide de $\overline{ne}_i$ étiquettes triées par ordre lexicographique : $(\overline{z}_i^l, \overline{T}_i^l), l = 1, \dots, \overline{ne}_i$, où $\overline{T}_i^l = (\overline{T}_i^{l,r} | r \in R)$. Ainsi, le coût maximum d'un chemin entre 1 et i est donné par $-\overline{z}_i^1$.

Dans la procédure d'étiquetage arrière, chaque étiquette arrière au noeud i correspond en fait à un chemin de i à n . Pour une étiquette retenue de la passe avant $((\underline{z}_i^l, \underline{T}_i^l)$ ou

$(\underline{z}_i^l, \overline{T}_i^l)$, le coût et les bornes de consommation de ressources sont ceux des chemins du noeud 1 au noeud i pouvant être prolongés par le chemin de i à n associé à l'étiquette arrière courante L , de telle sorte que l'on obtienne un chemin résultant de 1 à n de pointage P .

Les ne_i étiquettes arrière créées au noeud i ont les composantes suivantes :

z_i^L : coût de l'étiquette,

$b_i^{L,r}$: valeur maximale pour la ressource $r \in R$,

N_i^L : noeud prédécesseur utilisé pour créer l'étiquette L .

Notons qu'étant donné les temps d'attente permis, seules les valeurs maximales des ressources $b_i^{L,r}$ sont pertinentes. Il serait incongru de considérer des bornes inférieures $a_i^{L,r}$, d'où la notation clarifiée $b_i^{L,r}$ au lieu de $T_i^{L,r}$.

Dans la procédure d'étiquetage arrière, les étiquettes arrière créées au noeud i sont comparées aux étiquettes $(\underline{z}_i^l, \underline{T}_i^l)$ et $(\overline{z}_i^l, \overline{T}_i^l)$ obtenues par la résolution de problèmes de plus court et plus long chemin selon le principe de dominance suivant.

Par définition, les étiquettes de plus court chemin en chaque noeud j , pour tout chemin allant du noeud 1 au noeud j , déterminent le coût minimal permis pour une consommation de ressource donnée, tandis que les étiquettes de plus long chemin indiquent le coût maximal permis pour une consommation de ressource donnée. Le schéma 3.1 illustre la dominance des étiquettes $(\underline{z}_i^l, \underline{T}_i^l)$ dans le cas simple d'une seule ressource, les étiquettes de plus court chemin délimitant une zone (hachurée) de chemins interdits. Les points aux extrémités des lignes correspondent aux étiquettes de plus court chemin. Cet exemple considère différentes étiquettes (a, b, c) provenant de la procédure d'étiquetage arrière. On constate que l'étiquette a serait interdite (on ne peut avoir un chemin à meilleur coût qu'un plus court chemin), cependant les étiquettes b et c seraient valables.

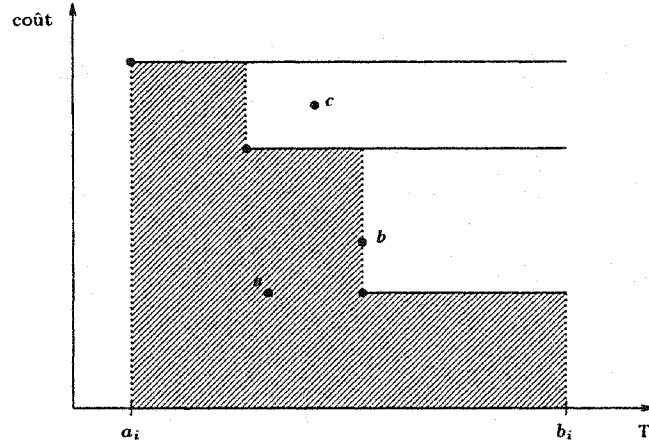


Figure 3.1 – Dominance des étiquettes de plus court chemin avec une seule ressource.

Ainsi, pour qu'un chemin de pointage P qui passe par le noeud i soit réalisable, il faut que les étiquettes $(z_i^L, b_i^{L,r})$ soient dominées par l'ensemble des étiquettes $(z_i^l, \underline{T}_i^l)$, de même que les étiquettes $(-z_i^L, b_i^{L,r})$ doivent être dominées par l'ensemble $(\bar{z}_i^l, \bar{T}_i^l)$, ce qui permettra sans doute d'éliminer des étiquettes superflues en cours de procédure.

Le pseudo-code de la procédure d'étiquetage arrière est fourni dans l'algorithme 2. Dans cette procédure, l'étiquette de départ se situe au dernier noeud n avec comme valeur de coût initiale z_n^1 un pointage égal au pointage final recherché, P , et une consommation de ressource $b_n^{1,r}$ égale à la borne supérieure b_n^r permise au dernier noeud. Une étiquette l se trouvant à un noeud j donné se prolonge en traversant à reculons un arc (i, j) entrant au noeud j pour créer une nouvelle étiquette L comme suit : $z_i^L = z_j^l - c_{ij}$ et $b_i^{L,r} = \min\{b_i^r, b_j^{l,r} - t_{ij}^r\}$. Le coût de la nouvelle étiquette est comparé avec celui des étiquettes $(z_i^l, \underline{T}_i^l)$ et $(\bar{z}_i^l, \bar{T}_i^l)$ en ce noeud, et si ce coût est inférieur au coût $\underline{z}_i^1$ minimal ou encore s'il est supérieur au coût $-\bar{z}_i^1$ maximal, ces cas étant interdits, on élimine L . De même, si pour une ressource $r \in R$, $b_i^{L,r} < a_i^r$, l'étiquette L n'est pas créée. Des tests de dominance sont ensuite effectués, la nouvelle étiquette étant valable seulement elle est dominée par les étiquettes $(z_i^l, \underline{T}_i^l)$ et $(\bar{z}_i^l, \bar{T}_i^l)$. À la toute fin, au noeud source 1, seules les étiquettes de coût $z_1^L = 0$ correspondront

aux chemins de coût P .

Algorithm 2 Étiquetage arrière basé sur les coûts

```

1:  $j = n, L = 0, z_n^1 = P, b_n^{1,r} = b_n^r$ 
2:  $ne_i = 0, \forall i \in N \setminus \{n\}; ne_n = 1$ 
3: tant que  $j > 1$  effectuer
4:   si  $ne_j \neq 0$  alors
5:     pour tout  $i$  tel que  $(i, j) \in A$  effectuer
6:       pour tout  $l = 1, \dots, ne_j$  effectuer
7:          $L = ne_i + 1$  {étiquette courante de  $i$ }
8:          $z_i^L = z_j^l - c_{ij}$ 
9:         si  $z_i^L < \underline{z}_i^1$  ou  $\bar{z}_i^L > -z_i^1$  alors aller à la ligne 6
10:        pour tout  $r \in R$  effectuer
11:           $b_i^{L,r} = \min\{b_i^r, b_j^{l,r} - t_{ij}^r\}$ 
12:          si  $b_i^{L,r} < a_i^r$  alors aller à la ligne 6
13:          si  $(z_i^L, b_i^{L,r})$  est non dominée par les étiquettes  $(\underline{z}_i^l, \underline{T}_i^{l,r})$  ou si  $(-z_i^L, b_i^{L,r})$ 
             est non dominée par les étiquettes  $(\bar{z}_i^l, \bar{T}_i^{l,r})$  alors
14:            aller à la ligne 6
15:             $N_i^L = j$ 
16:             $ne_i \leftarrow ne_i + 1$ 
17:       $j \leftarrow j - 1$ 
    {toutes les étiquettes d'indice  $L$  avec  $z_1^L = 0$  correspondront aux chemins recher-
    chés}

```

3.3.2 Algorithme basé sur les coûts réduits

L'algorithme présenté à la sous-section précédente s'avère relativement simple, mais il se pourrait que le nombre d'étiquettes conservées soit très grand. On pourrait s'interroger à savoir s'il serait possible d'éliminer davantage d'étiquettes en cours de processus. Par exemple, si l'on considérait les étiquettes en termes de coût réduit, est-ce que la dominance ne serait pas alors effectuée de façon plus resserrée, permettant ainsi une élimination plus efficace? C'est cette idée qui sera exploitée dans cette section. Toutefois, elle pourra s'appliquer seulement sous l'hypothèse que la valeur

optimale de la relaxation linéaire du problème z_{LP} et celle du problème z_{IP} sont égales. Dans ce cas, nous allons démontrer que le coût réduit d'un horaire réalisable s pour l'employé courant k est nul ($\bar{c}_s^k = 0$) dans le cas où les coûts réduits sont calculés à partir des variables duales obtenues avec la résolution du problème linéairement relaxé LP .

Quelques rappels des notations et définitions associées s'imposent afin de se remettre en contexte. Soit k , l'employé courant.

MIP_k ou IP_k : problème courant à résoudre possédant une solution optimale Θ_{IP}^k de coût P .

PM_k : problème maître relaxé associé à MIP_k ou IP_k possédant une solution de base optimale Θ_{LP}^k de coût P , i.e. $z_{LP}^k = z_{IP}^k = P$.

Π_{LP}^k : vecteur des variables duales associées à Θ_{LP}^k , obtenu à la dernière itération de génération de colonnes du noeud 0.

θ_s^l : variable de PM_k qui vaut 1 si l'horaire s est choisi pour l'employé $l = k, k + 1, \dots, n$, et 0 sinon.

c_s^l : coût de l'horaire s pour l'employé $l = k, k + 1, \dots, n$, ($c_s^l = 0, \forall l = k + 1, k + 2, \dots, n$).

$\bar{c}_s^l$: coût réduit de l'horaire s pour l'employé $l = k, k + 1, \dots, n$ calculé à la dernière itération de génération de colonnes du noeud 0 en utilisant Π_{LP}^k .

Ω_k^l : l'ensemble des horaires résiduels pour l'employé $l = k, k + 1, \dots, n$.

Le résultat énoncé dans la proposition suivante est à la base de l'algorithme qui sera proposé par la suite.

Proposition : Considérons la notation présentée ci-haut. Si $\bar{\Theta}^k = \{\bar{\Theta}_s^l | l = k, k + 1, \dots, n; s \in \Omega_k^l\}$ est une solution réalisable de PM_k pour laquelle $\exists s \in \Omega_k^k$ tel que

$\theta_s^k = 1$ et $c_s^k = P$, alors le coût réduit $\tilde{c}_s^k$, calculé à partir des variables duales Π_{LP}^k , est nul.

Preuve : Deux cas peuvent se présenter, dépendamment si θ_s^k est une variable de base ou non dans la solution optimale Θ_{LP}^k .

Premièrement, si θ_s^k est une variable de base, alors $\tilde{c}_s^k = 0$ par définition du coût réduit. Deuxièmement, considérons le cas où θ_s^k est une variable hors-base dans la solution Θ_{LP}^k . Rappelons que, pour cette solution,

$$z = z_{LP}^k + \tilde{c}_H \Theta_{LP,H}^k = P + \tilde{c}_H \Theta_{LP,H}^k \quad (3.9)$$

où $\Theta_{LP,H}^k$ est le vecteur des variables hors-base, $\tilde{c}_H$ est le vecteur des coûts réduits associé à ce vecteur, et z est la valeur de la fonction objectif en fonction des variables hors-base. Puisque $\bar{\Theta}^k$ est une solution réalisable de coût P , en fixant les variables du vecteur $\Theta_{LP,H}^k$ aux valeurs $\bar{\Theta}_H^k$ prises dans $\bar{\Theta}^k$, l'équation (3.9) devient

$$P = P + \tilde{c}_H \bar{\Theta}_H^k \implies \tilde{c}_H \bar{\Theta}_H^k = 0.$$

Comme $\tilde{c}_H \geq 0$ et $\bar{\Theta}_H^k \geq 0$, on obtient que chaque terme du produit scalaire est nul. Entre autres, le terme $\tilde{c}_s^k \bar{\theta}_s^k = 0$. Puisque $\bar{\theta}_s^k = 1$, on trouve que $\tilde{c}_s^k = 0$. $\square$

Nous venons ainsi de montrer que pour l'employé courant k , le coût réduit de tout chemin s réalisable de pointage P et associé à la solution optimale Θ doit nécessairement être nul. Cette information nous permettra entre autres d'éliminer toutes les étiquettes ne correspondant pas à des chemins de coût réduit nul.

Une première utilisation de ce fait serait de modifier l'algorithme 1 en retenant ensuite seulement les chemins de 1 à n qui ont un coût réduit nul. Une utilisation encore plus efficace est cependant possible. En effet, il serait plus efficace d'énumérer d'abord

tous les chemins de coût réduit égal à zéro, et de ne retenir à la fin que les solutions de coût total égal à P . Nous tirons ainsi avantage de la dominance de façon nettement plus marquée que si l'on considérait l'énumération de tous les chemins de coût P . Ceci sera illustré à la figure 3.2 plus loin.

Ce second algorithme, version modifiée du premier, se déroule en trois étapes :

1. Résoudre un problème de plus court chemin entre les noeuds 1 et n du réseau en considérant les coûts réduits. Ceci produit des étiquettes $(\underline{z}_i^l, \underline{T}_i^l)$.
2. Utiliser une procédure d'étiquetage arrière pour trouver tous les chemins de coût réduit nul entre 1 et n dans le réseau G .
3. Retenir les chemins de coût réduit nul et de pointage P .

L'algorithme ici se déroule encore selon une procédure d'étiquetage arrière, i.e. du noeud n vers le noeud 1. Afin d'éliminer des étiquettes ne pouvant mener à des chemins de coût réduit nul, on ne retiendra à chaque noeud que les étiquettes dont le coût réduit correspond à celui d'un plus court chemin $(\underline{z}_i^l, \underline{T}_i^l)$. On peut consulter le pseudo-code à la page suivante (Algorithme 3).

Puisque tout chemin de coût réduit nul est ici optimal (minimal), il faut donc nécessairement que le coût réduit de tout chemin allant en chaque noeud ($1 \rightarrow j$) corresponde exactement au coût réduit du plus court chemin ($1 \rightarrow j$) dont la consommation de ressources est comparable. On peut éliminer ainsi toutes les étiquettes arrière dont le coût réduit (déduit de $1 \rightarrow j$) ne correspond pas à celui d'un plus court chemin $(\underline{z}_i^l, \underline{T}_i^l)$. Notons également que les étiquettes calculées ne peuvent dominer celles des plus courts chemins, par définition.

Sur le schéma 3.2, les points aux extrémités des lignes correspondent aux étiquettes de plus court chemin $(\underline{z}_i^l, \underline{T}_i^l)$ créées par le deuxième algorithme. Seules les étiquettes arrière situées sur les paliers sont admissibles, le coût réduit de celles-ci devant nécessairement être celui d'une étiquette $(\underline{z}_i^l, \underline{T}_i^l)$, puisque minimal.

Algorithm 3 Étiquetage arrière basé sur les coûts réduits

- 1: $j = n, L = 0, z_n^1 = P, \tilde{z}_n^1 = 0, b_n^{1,r} = b_n^r$
 - 2: $ne_i = 0, \forall i \in N \setminus \{n\}; ne_n = 1$
 - 3: **tant que** $j > 1$ **effectuer**
 - 4: **si** $ne_j \neq 0$ **alors**
 - 5: **pour tout** i tel que $(i, j) \in A$ **effectuer**
 - 6: **pour tout** $l = 1, \dots, ne_j$ **effectuer**
 - 7: $L = ne_i + 1$ {étiquette courante de i}
 - 8: $z_i^L = z_j^l - c_{ij}, \tilde{z}_i^L = \tilde{z}_j^l - \tilde{c}_{ij}$
 - 9: **pour tout** $r \in R$ **effectuer**
 - 10: $b_i^{L,r} = \min\{b_i^r, b_j^{l,r} - t_{ij}^r\}$
 - 11: **si** $b_i^{L,r} < a_i^r$ **alors** aller à la ligne 6
 - 12: **si** $\tilde{z}_i^L$ ne correspond pas au coût réduit d'une étiquette $(\tilde{z}_i^l, \underline{T}_i^l)$ **alors**
 - 13: aller à la ligne 6
 - 14: Test de dominance sur la consommation des ressources
 - 15: $N_i^L = j$
 - 16: $ne_i \leftarrow ne_i + 1$
 - 17: $j \leftarrow j - 1$
- {toutes les étiquettes d'indice L avec $z_1^L = 0$ et $\tilde{z}_1^L = 0$ correspondront aux chemins recherchés}
-

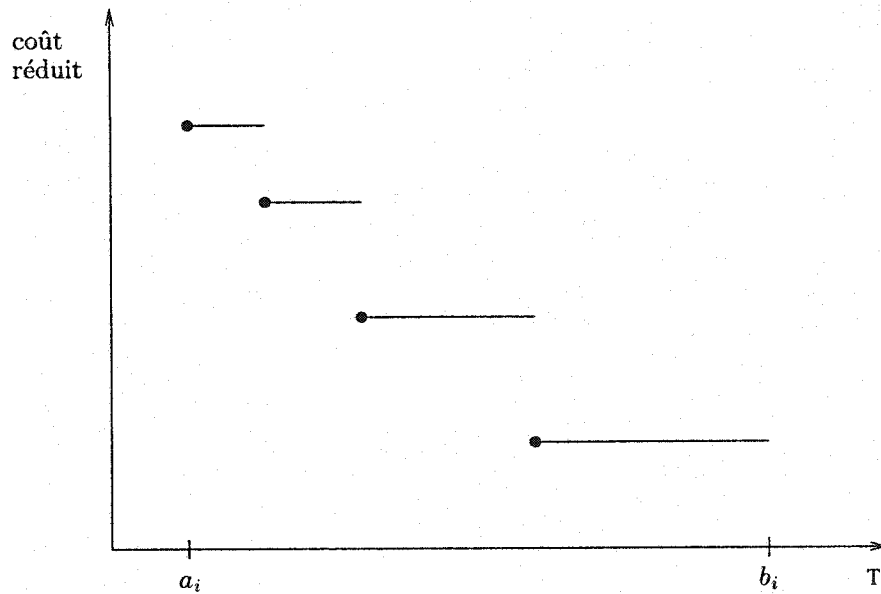


Figure 3.2 – Dominance des étiquettes $(\tilde{z}_i^l, \underline{T}_i^l)$ avec une seule ressource.

Par contre, on ne peut imposer que le coût calculé soit exactement celui du plus court chemin en termes de coût associé à cette consommation de ressources, car les chemins de pointage P ne sont pas nécessairement optimaux. C'est pourquoi procéder avec des étiquettes de coût réduit (au lieu du coût simple) devrait permettre de restreindre davantage le nombre d'étiquettes conservées en chaque noeud. Il pourrait exister des chemins de coût réduit nul dont le pointage serait différent de P . Nous devons donc conserver uniquement à la fin du processus (au noeud 1) les étiquettes dont le pointage sera exactement P .

En guise d'amélioration, on peut combiner les deux algorithmes proposés. Dans la procédure d'étiquetage arrière, il s'agira alors de s'assurer que le coût réduit de l'étiquette corresponde à celui d'un $(\tilde{z}_i^l, \underline{T}_i^l)$, après quoi nous vérifierons (avec le pointage) que l'étiquette ne domine pas celle d'un plus court chemin $(\underline{z}_i^l, \underline{T}_i^l)$ ou d'un plus long chemin $(\bar{z}_i^l, \bar{T}_i^l)$. On obtient ainsi l'algorithme correspondant :

1. Résoudre un problème de plus court chemin entre les noeuds 1 et n du réseau considérant les coûts réduits. Ceci produit des étiquettes $(\underline{z}_i^l, \underline{T}_i^l)$.
2. Résoudre un problème de plus court chemin en considérant les coûts c_{ij} . Ceci produit des étiquettes $(z_i^l, \underline{T}_i^l)$.
3. Résoudre un problème de plus long chemin en considérant les coûts $-c_{ij}$. Ceci produit des étiquettes $(\bar{z}_i^l, \bar{T}_i^l)$.
4. Utiliser une procédure d'étiquetage arrière pour trouver tous les chemins de coût recherché. On utilise la dominance pour éliminer les chemins impossibles.
5. Retenir les chemins de coût réduit nul et de pointage P .

On peut consulter le pseudo-code ci-dessous (Algorithme 4).

3.4 Fonctions de prolongation non linéaires

En pratique, les fonctions de prolongation rencontrées sont généralement non linéaires pour les applications de type PBS, i.e. le système de fabrication d'horaires personnalisés avec priorité que nous étudions. Les fonctions de prolongation peuvent alors varier non linéairement selon les ressources, le coût et le coût réduit. Ainsi, pour les ressources, il y a une fonction de prolongation du genre : $T_j^r = f_{ij}(\mathbf{T}_i)$. De même pour le coût : $z_j = z_{ij}(z_i, \mathbf{T}_i)$ et pour le coût réduit : $\tilde{z}_j = \tilde{z}_{ij}(\tilde{z}_i, \mathbf{T}_i)$. Ces fonctions, $f_{ij}(\mathbf{T}_i)$, $z_{ij}(z_i, \mathbf{T}_i)$ et $\tilde{z}_{ij}(\tilde{z}_i, \mathbf{T}_i)$, sont possiblement non linéaires.

Avec de telles fonctions de prolongation, il devient assez complexe d'effectuer l'étiquetage arrière. En effet, l'inverse de ces fonctions n'étant pas toujours disponible, comment bien calculer le coût des chemins arrière ? On ne voit pas comment utiliser les algorithmes proposés à la section 3.3. Il faut donc considérer des algorithmes faisant appel seulement à des passes avant.

Algorithm 4 Étiquetage arrière, version hybride

```

1:  $j = n, L = 0, z_n^1 = P, \bar{z}_n^1 = 0, b_n^{1,r} = b_n^r$ 
2:  $ne_i = 0, \forall i \in N \setminus \{n\}; ne_n = 1$ 
3: tant que  $j > 1$  effectuer
4:   si  $ne_j \neq 0$  alors
5:     pour tout  $i$  tel que  $(i, j) \in A$  effectuer
6:       pour tout  $l = 1, \dots, ne_j$  effectuer
7:          $L = ne_i + 1$  {étiquette courante de  $i$ }
8:          $z_i^L = z_j^l - c_{ij}, \bar{z}_i^L = \bar{z}_j^l - \tilde{c}_{ij}$ 
9:         si  $z_i^L < \underline{z}_i^1$  ou  $\bar{z}_i^L > -z_i^1$  alors aller à la ligne 6
10:        pour tout  $r \in R$  effectuer
11:           $b_i^{L,r} = \min\{b_i^r, b_j^{l,r} - t_{ij}^r\}$ 
12:          si  $b_i^{L,r} < a_i^r$  alors aller à la ligne 6
13:          si  $\bar{z}_i^L$  ne correspond pas au coût réduit d'une étiquette  $(\underline{z}_i^l, \underline{T}_i^l)$  alors
14:            aller à la ligne 6
15:          si  $(z_i^L, b_i^L)$  est non dominée par les étiquettes  $(\underline{z}_i^l, \underline{T}_i^{l,r})$  ou si  $(-z_i^L, b_i^{L,r})$ 
16:            est non dominée par les étiquettes  $(\bar{z}_i^l, \bar{T}_i^{l,r})$  alors
17:              aller à la ligne 6
18:               $N_i^L = j$ 
19:               $ne_i \leftarrow ne_i + 1$ 
20:    $j \leftarrow j - 1$ 
21: {toutes les étiquettes d'indice  $L$  avec  $z_1^L = 0$  et  $\bar{z}_1^L = 0$  correspondront aux chemins recherchés}

```

3.4.1 Algorithme basé sur les coûts

Une première approche serait de modifier légèrement la dominance sur les coûts c_{ij} dans l'algorithme de plus court chemin. De la sorte, une simple énumération des étiquettes de plus court chemin au noeud puits devrait permettre de retracer tous les horaires réalisables de pointage recherché. Bien que cette méthode soit très simple, elle possède le désavantage d'une explosion combinatoire possible quant au nombre d'étiquettes en chaque noeud. Dans ce cas, on pourra avoir recours à une énumération heuristique pour limiter le nombre d'étiquettes par noeud.

Soit P^{min} , le coût du plus court chemin qui est calculé a priori à l'aide d'un algorithme de plus court chemin standard. Considérons deux étiquettes $E^i = (c^i, T^i)$ et $E^j = (c^j, T^j)$ correspondant à deux chemins partiels se terminant à un même noeud. La dominance habituelle entre ces étiquettes, telle que présentée à la section 3.2, propose d'éliminer E^j si $c^i \leq c^j$ et $T^i \leq T^j$. En effet, c^j et T^j étant supérieurs ou égaux à c^i et T^i , respectivement, et les fonctions de prolongations du coût et des ressources étant non décroissantes, la prolongation de l'étiquette E^j ne peut mener qu'à des étiquettes de coûts et de ressources supérieurs ou égaux aux coût et ressources obtenus en prolongeant de la même façon l'étiquette E^i . Comme on ne recherche habituellement qu'un seul plus court chemin, il devient donc inutile de conserver l'étiquette E^j .

Dans notre cas, on veut trouver tous les chemins de pointage P sachant que $P \geq P^{min}$. Par conséquent, toute étiquette pouvant mener à un chemin ayant un coût excédant d'au plus $P - P^{min}$ le coût du plus court chemin doit être conservée. Il n'est donc pas justifié de rejeter une étiquette E^j en la comparant à une étiquette E^i à moins que $c^i + P - P^{min} < c^j$. Nous proposons donc de remplacer la règle de dominance habituelle par la règle suivante :

Éliminer l'étiquette E^j si $c^i + P - P^{min} < c^j$ et $T^i \leq T^j$.

Par exemple, considérons la situation suivante : le meilleur chemin est de coût $P^{min} = -55$ et la solution optimale est de coût $P = -50$. Si l'on considère les étiquettes E^i et E^j respectivement, de coûts $c^i = -30$ et $c^j = -25$ et de ressources T^i et T^j telles que $T^i \leq T^j$, selon la dominance standard E^j devrait être éliminée, mais elle sera maintenant conservée vu la modification apportée.

En résumé, il y aura trois étapes à respecter :

1. Résoudre un problème de plus court chemin avec la dominance habituelle, ce qui permettra d'obtenir P^{min} .
2. Résoudre un problème de plus court chemin avec la dominance modifiée telle qu'exposée ci-haut.
3. Retenir tous les chemins de pointage P .

Avec la dominance modifiée dans l'algorithme de plus court chemin, on garde ainsi tous les chemins de coût $\leq P$. Il s'agira ensuite d'énumérer au dernier noeud toutes les étiquettes de coût P , ce qui permettra de retracer les chemins associés. En général, dans les exemples étudiés, $P^{min} = P$ ce qui fait qu'on évite les explosions combinatoires.

3.4.2 Algorithme basé sur les coûts réduits

Une autre stratégie à considérer serait d'énumérer plutôt les chemins de coûts réduits nuls, reprenant ainsi le résultat démontré à la section 3.3.2. On effectue d'abord la passe avant de l'algorithme 3, en modifiant toutefois la dominance pour pouvoir conserver tous les chemins de coût réduit nul. Soit deux étiquettes $E^i = (\tilde{c}^i, T^i)$ et $E^j = (\tilde{c}^j, T^j)$ correspondant à deux chemins partiels se terminant à un même noeud. Reprenant le même raisonnement qu'à la section 3.4.1, nous obtenons ainsi la règle de dominance suivante :

Éliminer l'étiquette E^j si $\tilde{c}^i < \tilde{c}^j$ et $T^i \leq T^j$.

Bien sûr, il faudra préalablement récupérer les variables duales entrant dans le calcul des coûts réduits et possédant les propriétés énoncées dans la section 3.3.2. Ce pré-requis est d'ailleurs le point délicat de l'approche basée sur les coûts réduits. Nous y reviendrons au chapitre suivant.

Après avoir obtenu les variables duales, voici les étapes à effectuer :

1. Résoudre un problème de plus court chemin avec la dominance modifiée telle qu'exposée ci-haut.
2. Retenir tous les chemins de coût réduit nul.
3. Calculer le coût associé à chacun des chemins retenus à l'étape 2.
4. Retenir les chemins ayant un coût nul à l'étape 3.

Bien entendu, il ne s'agit ici que d'un exposé sommaire de la démarche à suivre, l'implantation requérant quelques ajustements techniques.

3.4.3 Algorithme utilisant les colonnes du problème maître

Une dernière méthode pour énumérer des horaires de pointage P consiste tout simplement à retenir en mémoire toutes les colonnes du problème maître qui ont le coût désiré P . Pour ce faire, aucun passage dans un algorithme de plus court chemin avec dominance modifiée n'est nécessaire. Cependant, il peut exister d'autres chemins de même coût qui ne font pas partie du problème maître, ces chemins seront donc ignorés.

Chapitre 4

Modification de la méthode de résolution

L'objectif visé par ce mémoire est d'améliorer la qualité des horaires des employés juniors. Par qualité, nous entendons la satisfaction desdits employés, à savoir la minimisation de leur pointage. Nous exposerons ici les modifications en ce sens apportées à la méthode de résolution.

Tel que mentionné précédemment, il peut arriver qu'il existe plusieurs horaires de meilleur pointage pour un même employé. L'usage actuel consiste à fixer un seul de ces horaires pour ensuite poursuivre avec la résolution du problème de l'employé suivant. Mais rien ne garantit que ce choix soit celui qui convienne le mieux pour les employés suivants. Afin de pallier à cette lacune, nous proposons donc d'énumérer tous les horaires de meilleur pointage pour l'employé courant avec les algorithmes proposés au chapitre précédent, à la section 3.4. Cette énumération se fera une fois que le problème maître *MIP* ou *IP* aura été résolu pour l'employé courant. Les horaires énumérés seront ensuite triés selon certains critères de classification ou fonctions de pénalité, lesquelles fonctions seront présentées plus bas. Il s'agira ensuite de fixer en conséquence le choix de l'horaire pour l'employé courant, contrairement à la méthode standard actuelle. Pour ce faire, nous présenterons et expliquerons la nouvelle façon d'effectuer le branchement. Cet exposé sera suivi de quelques précisions concernant les ajustements supplémentaires que nous avons dû apporter lors de l'implantation.

4.1 Pénalité associée à un horaire

Une fois que tous les horaires de meilleur pointage ont été énumérés, ces horaires devront être évalués en fonction des préférences des employés suivants, l'objectif étant de choisir pour l'employé courant k l'horaire de meilleur pointage pénalisant le moins les employés suivants. Différents critères de pénalité peuvent être considérés, que nous décrirons plus bas avec leurs fonctions associées. De plus, nous devons tenir compte de la priorité accordée selon l'ancienneté, ce qui se traduira par des poids décroissants selon l'ordre des employés j suivants, soit les poids $\lambda_j = 10^{M-(j-k)+1}$. Seuls les M employés suivants seront considérés, le poids des autres est alors considéré comme négligeable.

Afin de normaliser selon la satisfaction maximale potentielle de chacun, des problèmes de plus court et de plus long chemin seront résolus pour les M employés suivants considérés, calculés en tenant compte des choix fixés pour les $k-1$ premiers employés. Ceci sera effectué pour déterminer les pointages maximal P_j^{max} et minimal P_j^{min} possibles pour chacun à ce moment de la résolution.

Il faut considérer :

H^k : l'ensemble des horaires à énumérer ;

h : un des horaires ;

ρ_h^k : la pénalité associée à l'horaire $h \in H^k$.

Deux façons d'évaluer la pénalité ρ_h^k sont présentées ci-bas.

4.1.1 Première méthode

On peut se baser sur l'attrait de l'horaire h pour les employés suivants. Si cet horaire est peu convoité, nous faisons l'hypothèse qu'il constituera alors un bon choix

pour l'employé courant. Pour chaque horaire h à évaluer, soit P_j^h le pointage que le j^{ieme} employé obtiendrait avec cet horaire h . Pour que cet horaire soit possible pour l'employé j , il faut nécessairement que $P_j^h \in [P_j^{min}, P_j^{max}]$. Ensuite, ce pointage sera majoré pour avoir un indice du pourcentage de satisfaction maximale atteint.

Tenant compte de tous les éléments précédents, la fonction de pénalité utilisée pour l'horaire h est la suivante :

$$\rho_h^k = \sum_{j=k+1}^{k+M} \lambda_j \frac{P_j^h - P_j^{min}}{P_j^{max} - P_j^{min}}. \quad (4.1)$$

Cette pénalité globale devrait permettre d'évaluer et de choisir pour l'employé k l'horaire qui pénalise le moins les M employés suivants. En effet, moins un horaire pénalise les employés suivants, plus la valeur de sa fonction de pénalité sera grande. Par conséquent, un horaire h_1 sera considéré avant un horaire h_2 si $\rho_{h_1}^k > \rho_{h_2}^k$.

Bien sûr, le critère choisi comme base à cette méthode est très heuristique. On ne tient pas compte de l'impact de la décision sur les horaires des employés suivants. Par conséquent, le choix pour ces derniers devient en effet restreint du fait que l'employé courant sélectionne certaines tâches susceptibles de devenir saturées.

4.1.2 Deuxième méthode

Une autre approche serait d'évaluer quel serait le pointage minimal potentiel des M employés suivants si l'on considère que les tâches constituant l'horaire h seront attribuées à l'employé courant k . Il s'agit d'éliminer temporairement les tâches (rotations) ayant atteint leur limite à couvrir si l'employé k hérite de l'horaire h . On enlève alors temporairement ces tâches saturées du réseau des M suivants, et l'on applique un

algorithme de plus court chemin sur les réseaux ainsi modifiés afin de déterminer le meilleur pointage maintenant possible pour chacun. Cette méthode possède un avantage très intéressant. En effet, s'il n'y a pas de plus court chemin possible pour un des M suivants suite à l'attribution de h , cela signifie que ce chemin h doit être rejeté puisqu'il ne permet pas de solution réalisable pour l'un des employés suivants, ce qui évite par la suite certains retours en arrière dans l'arbre de branchement externe.

Cette méthode permet donc de voir l'impact de la décision courante (fixer l'horaire h pour l'employé k) sur les employés suivants. Cependant, si le résultat est très valide pour évaluer la pénalité encourue pour l'employé suivant (P_{k+1}), on perd de la précision lorsque l'on évalue $P_{k+2}, P_{k+3}, \dots$, puisque l'on devrait considérer la décision de $k+1$ pour évaluer $k+2$, et ainsi de suite...

Si l'horaire h est valide, on procédera au calcul normalisé de la pénalité, tel que présenté à l'équation (4.1), sauf que P_j^h représente ici le pointage minimal que l'employé j obtiendrait si h est choisi pour l'employé k . En ce cas, nous aurions tout intérêt à ce que la valeur de ρ_h^k soit la plus petite possible puisqu'il faut tendre à pouvoir obtenir ensuite un pointage le plus près possible du minimum désiré (P_{min}) pour les suivants.

4.2 Branchement multiple sur les horaires d'un employé

Nous avons décrit au chapitre précédent comment s'effectuait l'énumération des différents horaires de même pointage pour un employé donné, et nous venons d'exposer à la section précédente le calcul des pénalités associées au choix de chacun de ces horaires. Tous ces horaires sont alors triés en ordre croissant de pénalité, l'horaire désavantageant le moins les employés suivants sera alors choisi et fixé le premier.

Parmi les horaires de meilleur pointage dénombrés pour l'employé courant k , on veut maintenant choisir et fixer l'horaire qui pénalise le moins les employés suivants. Les modifications apportées en ce sens au branchement seront décrites ci-dessous. Le lecteur peut se référer au chapitre 2 en ce qui concerne le branchement standard. Le problème IP (2.1) comporte des contraintes d'intégrité globale (2.7). Rappelons que l'on résout généralement un problème mixte MIP où les contraintes d'intégrité globale sont relaxées pour les employés suivants. La section 2.3 décrivait le processus de résolution existant.

Au lieu de sélectionner arbitrairement un horaire parmi ceux de meilleur pointage, on les retiendra tous et l'on tente d'abord de fixer celui qui est le moins désiré par les employés résiduels d'après la fonction de pénalité. Cette décision étant fixée, on passe à la résolution de l'employé suivant. Advenant que l'on ne puisse alors trouver de solution au MIP , au lieu de remonter immédiatement et de résoudre alors à l'intégrité globale, on explore plutôt le chemin suivant en termes de pénalités. Ce n'est qu'après avoir exploré tous les chemins de meilleur pointage et dans le cas où aucune solution ne semble alors possible pour l'employé suivant que l'on remontera dans l'arbre et que l'on solutionnera alors à l'intégrité globale.

La figure 4.1 illustre le nouvel arbre de branchement. Les noeuds en trait plein représentent des problèmes MIP_k tandis que le noeud en pointillé est résolu à l'intégrité globale IP_k . Les noeuds seront explorés dans l'ordre indiqué.

Dans cet exemple, la résolution du problème MIP pour l'employé k donne plusieurs horaires de meilleur pointage. Ces horaires sont triés selon la pénalité croissante pour les employés suivants. Le choix est d'abord fixé sur celui (noeud 2) pénalisant le moins les employés suivants, mais la résolution du MIP correspondant n'admet pas d'horaire possible pour l'employé $k + 1$. C'est pourquoi on remonte dans l'arbre et l'on choisit plutôt le deuxième meilleur horaire (même pointage, mais pénalise un

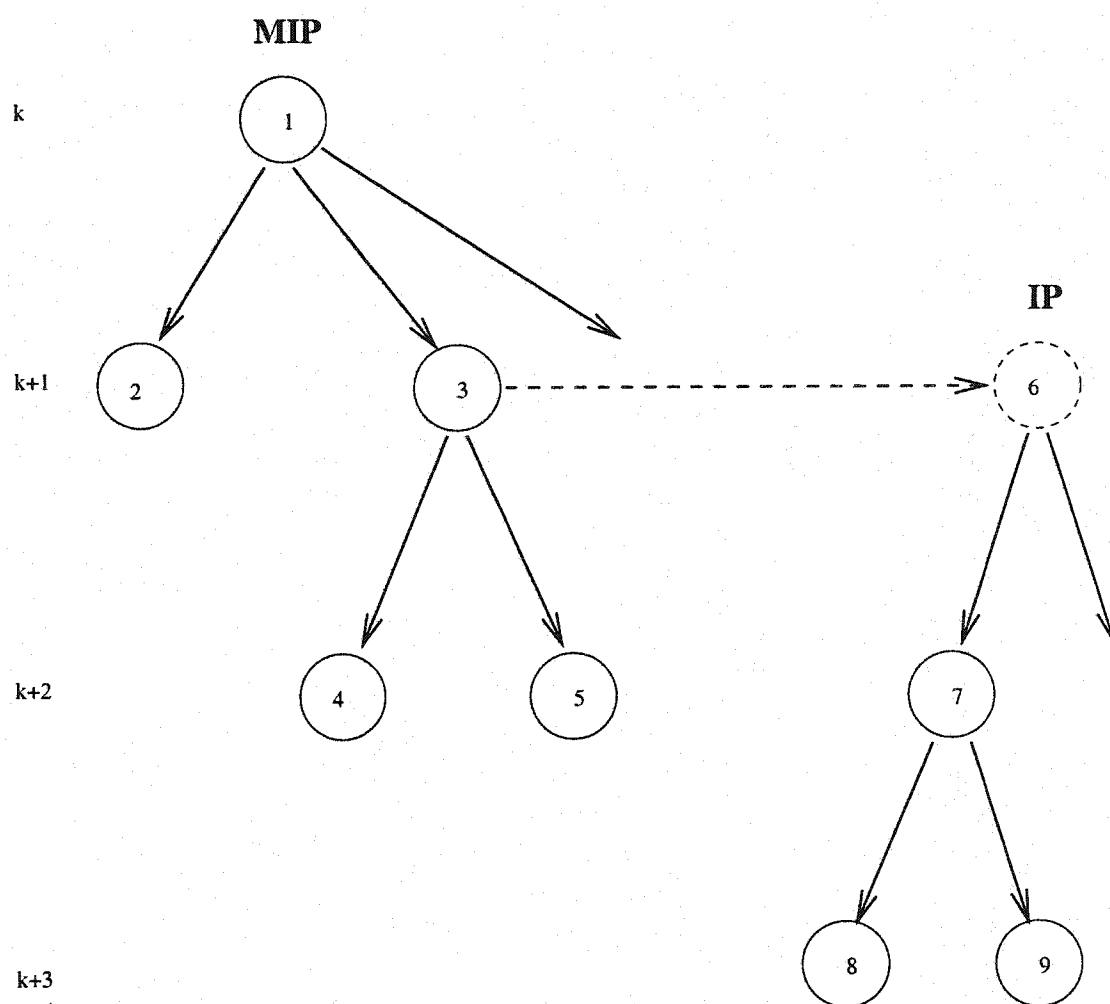


Figure 4.1 – Nouvel arbre de branchement externe.

peu plus les employés suivants) correspondant ici au noeud 3. En fixant cet horaire et en résolvant le MIP, deux horaires possibles sont alors obtenus pour l'employé $k + 1$. Le MIP du premier de ces horaires (noeud 4) ne permet pas d'horaire pour l'employé $k + 2$, on remonte alors dans l'arbre et l'autre horaire de meilleur pointage pour l'employé $k + 1$ sera exploré (noeud 5). Un nouveau MIP est alors résolu, et puisqu'aucun horaire n'est possible pour l'employé $k + 2$, on remonte au noeud 3. Puisque le MIP correspondant avait déjà été résolu et que tous les horaires énumérés ont été testés, le problème sera alors traité à l'intégrité globale IP (noeud 6). La résolution IP donne plusieurs horaires de meilleur pointage et l'on explore d'abord le plus avantageux pour les juniors (noeud 7). Ce nouvel horaire est ensuite fixé et l'on résout le MIP correspondant à ce choix pour l'employé $k + 2$. On obtient ainsi des horaires admissibles de meilleur pointage et, de même que précédemment, on commence par fixer le choix sur le moins pénalisant d'entre eux. Le MIP associé (noeud 8) ne permettant pas d'horaire possible pour l'employé $k + 3$, on remonte au niveau précédent et l'on fixe le choix sur l'horaire suivant (noeud 9). Et la résolution se poursuit ainsi.

Il faut préciser que l'implantation choisie est heuristique s'il existe beaucoup d'horaire de même pointage. En ce cas, on en garde seulement un certain nombre, incluant celui qui aurait été retenu par la méthode standard puisqu'il admet une solution réalisable au problème MIP ou IP qui l'a généré.

4.3 Ajustements dus à l'implantation

Les algorithmes présentés au chapitre précédent requièrent quelques modifications au code original PBS, lesquelles sont susceptibles d'altérer les résultats dans certains cas.

a) Cas de l'algorithme basé sur les coûts

Tel que mentionné précédemment, nous avons utilisé une dominance réduite dans le problème de plus court chemin, afin d'énumérer tous les chemins de coût entre P et P^{min} . Cette méthode est relativement simple, mais possède le désavantage de générer un très grand nombre d'étiquettes, ce qui peut devenir problématique. De plus, un paramètre dans PBS restreint le nombre d'étiquettes par noeud, nous avons dû hausser la limite permise. De plus, des règles de dominance plus restrictives sont utilisées pour l'énumération des horaires de pointage P . Il se peut toutefois que l'on ne puisse générer aucune étiquette ayant le coût désiré, dû à un problème de mémoire (limite atteinte).

b) Cas de l'algorithme basé sur les coûts réduits - ajout de coupes

Nous devons modifier la méthode de résolution dans le cas où l'algorithme d'énumération avec coûts réduits est utilisé afin d'obtenir les variables duales pertinentes. En effet, lors du processus d'évaluation et séparation progressive, les variables duales sont constamment modifiées lorsque l'on passe d'un noeud de branchement (donc d'un sous-problème) à l'autre. Évidemment, il se pourrait fort bien que la solution entière retenue ne soit pas celle obtenue lors de la dernière itération et, par conséquent, les variables duales correspondantes auront été remplacées en mémoire par de nouvelles. Il faudrait s'assurer d'obtenir les conditions requises pour que les variables duales soient valides lors de l'appel de notre algorithme, soit que ces variables duales correspondent aux variables duales découlant d'une solution primale d'une relaxation linéaire ayant P comme valeur optimale. Une façon d'obtenir les conditions requises en tout temps serait de retourner au noeud zéro et d'utiliser une coupe pour garantir une valeur optimale de P .

La stratégie suggérée pour obtenir les variables duales recherchées est la suivante.

- Après avoir obtenu la solution optimale $z_{IP}^k = P$, on remonte au noeud zéro.
- On résout ensuite un noeud de branchement qui aura été dupliqué, en imposant

une coupe dans le sous-problème, telle que la valeur de la solution ne pourra être inférieure à la valeur optimale retenue précédemment : on impose $z^k \geq P$.

- On obtient ainsi une solution optimale z_{LP}^k et les variables duales correspondantes, Π_{LP}^k , qui seront notées.

Chapitre 5

Résultats obtenus

La méthode, son principe et son implantation, ayant été décrite dans les chapitres précédents, il importe maintenant d'en valider l'efficacité et la pertinence. Pour ce faire, nous comparerons les résultats obtenus selon cette nouvelle méthode avec les originaux. Le pointage des employés juniors est-il vraiment meilleur qu'auparavant ? À cet effet, des tests comparatifs sont proposés sur différents jeux de données. Dans un premier temps, les jeux de données utilisés seront présentés afin de bien saisir le contexte pratique. Ceci sera suivi d'une comparaison des différents modèles et paramètres de dominance pouvant être utilisés dans l'algorithme de plus court chemin avec dominance modifiée, afin de déterminer lequel de ces modèles s'avère le plus approprié. On présente ensuite le résultat de l'énumération simple avec les méthodes retenues, c'est-à-dire le nombre de chemins énumérés ainsi que leurs distributions. Le plan des tests est ensuite exposé, la démarche se voulant la plus complète et structurée possible. Finalement, les résultats comparés seront présentés, commentés et analysés. Des tests plus généraux seront suivis d'exemples plus détaillés. Enfin, les temps de calcul seront montrés.

5.1 Description des jeux de données

Les tests sont effectués sur de vraies données provenant de la compagnie Air Canada. La version utilisée du logiciel PBS est d'ailleurs celle adaptée pour Air Canada, tel que développée par la compagnie Ad Opt Technologies.

Les problèmes considèrent un horizon d'un mois (décembre, en l'occurrence), pour lequel il s'agit d'affecter au personnel de vol toutes les rotations d'un type d'appareil. Les rotations débutent et se terminent toujours à une même base d'attachement. Nous avons les bases suivantes : Montréal (YUL), Toronto (YYZ), Vancouver (YVR), Winnipeg (YWG). Et nous considérons des appareils de type 320, 340, 767. Ainsi, le problème dénoté YUL-767 consiste à attribuer aux employés des rotations partant et se terminant à Montréal, sur des appareils de type 767. Cette notation sera utilisée tout au long du présent chapitre.

Le tableau 5.1 présente les caractéristiques permettant d'évaluer la taille de chaque problème étudié. Le nombre total d'employés est d'abord indiqué, suivi du nombre d'employés pour lesquels un horaire ou bloc complet est demandé (ces derniers sont surnommés *blokholder*, d'où l'appellation BH). Il est à noter que les algorithmes proposés ne seront pertinents qu'appliqués à ces employés BH, puisque les autres employés ne demandent pas d'horaire (ils sont en réserve). On indique également le nombre de rotations à attribuer. Selon ces facteurs, il s'agit ici de problèmes de petite et moyenne tailles. Il est à noter que tous les tests ont été effectués sur le même processeur, un Sunfire 4800 avec 4 cpu 900MHz et 8 GB de mémoire.

Tableau 5.1 – Caractéristiques des jeux de données

	YUL	YVR		YWG	YYZ
	767	320	767	320	340
Nombre d'employés	40	68	113	56	98
Nombre d'employés BH	26	37	84	42	71
Nombre de rotations	223	413	627	343	566

5.2 Comparaison des méthodes et ajustement des paramètres

Lorsque l'on fait appel à l'algorithme de plus court chemin, différents modèles de dominance peuvent être utilisés. Nous allons tester l'incidence de ce choix lorsque l'on utilise l'algorithme de plus court chemin avec dominance modifiée pour l'énumération dans les cas avec coûts (COST) et avec coûts réduits (DUAL). L'objectif ici recherché est de pouvoir énumérer, pour chaque employé, au moins un horaire ayant le meilleur coût, et ce dans un temps de calcul raisonnable.

Trois modèles différents seront testés sur deux jeux de données (les temps de calcul étant trop longs pour les autres jeux de données), soit le dernier modèle utilisé pour résoudre le problème *MIP* associé à un employé (ce modèle propose bien souvent une dominance sur peu de ressources), le modèle 3RES où la dominance s'effectue sur trois ressources, et un nouveau modèle implanté NEW où la dominance s'effectue sur toutes les ressources possibles.

Les résultats obtenus sont présentés dans le tableau 5.2. Pour chaque problème (YUL-767 et YVR-320), en plus du temps d'énumération requis et du nombre d'employés pour lesquels aucun chemin n'est énuméré (nuls, # cas), le tableau indique également le nombre moyen de chemins, en excluant les cas où aucun chemin n'a été trouvé ainsi que les extrêmes trop écartés de la médiane.

En comparant les modèles de dominance les uns aux autres, il va sans dire que choisir le dernier modèle ne semble pas judicieux : bien que le temps d'énumération soit alors très rapide, le nombre d'employés où l'on ne parvient pas à énumérer de chemins est trop important. On remarque qu'en resserrant la dominance, on parvient à énumérer des chemins dans plus de cas. En jouant avec d'autres paramètres (espace-mémoire permis pour les étiquettes de plus court chemin, par exemple), on pourrait

Tableau 5.2 – Comparaison des modèles de dominance

algo	modèle	YUL-767			YVR-320		
		temps (sec)	nuls # cas	moyenne # chemins	temps (sec)	nuls # cas	moyenne # chemins
COST	dernier	16	3	2	28	10	15
	3RES	756	0	5	2883	1	22
	NEW	1218	0	5	4834	0	21
DUAL	dernier	22	6	2	2344	10	21
	3RES	12660	3	4	22899	8	9
	NEW	19341	2	4	32808	6	9

éventuellement parvenir à énumérer pour tous les employés (aucun cas nul), mais les temps de calcul seraient probablement encore plus longs. Le choix du modèle 3RES avec dominance sur trois ressources semble être un compromis très acceptable.

Comparons maintenant les résultats des deux algorithmes. Le nombre de chemins énumérés en moyenne est plus faible dans le cas de l'algorithme avec coûts réduits (DUAL). De plus, si l'on utilise cet algorithme plutôt que celui avec coûts (COST), on remarque un plus grand nombre d'employés n'ayant aucun chemin énuméré avec le coût recherché. Le fait de ne pouvoir toujours énumérer au moins un chemin de coût désiré s'avère dérangeant, il y a lieu de s'interroger à ce sujet. Rappelons toutefois que l'espace-mémoire réservé aux étiquettes est limité dans l'algorithme de plus court chemin, et lorsque celui-ci est atteint en cours de résolution, on ne peut prolonger toutes les étiquettes jusqu'au bout.

Nous constatons de plus que la méthode DUAL requiert un grand temps de calcul comparativement à la méthode COST, et ce peu importe le modèle de dominance utilisé. On peut soupçonner que la recherche des variables duales qui utilise une coupe au niveau des sous-problèmes requiert une bonne partie du temps de calcul supplémentaire. De plus, la méthode COST parvient plus souvent à énumérer les

chemins. Considérant que les temps de calcul iront en s'accroissant pour les autres problèmes, les arguments précédents nous permettent de rejeter la méthode DUAL puisque beaucoup moins prometteuse que la méthode COST.

La méthode DUAL étant maintenant rejetée, tous les résultats qui suivront seront ceux obtenus avec une ancienne version du code (qui ne permet pas d'utiliser la coupe requise pour la méthode DUAL). Nous avons remarqué que, pour des raisons inconnues, cette ancienne version est plus rapide que la version permettant les coupes. Puisque cette dernière version ne comporte aucun avantage, il n'y a pas d'intérêt particulier à reprendre tous les tests avec celle-ci.

5.3 Dénombrement des employés à horaires multiples

Tel que justifié à la section précédente, l'algorithme COST utilise désormais le modèle de dominance 3RES avec trois ressources dans son algorithme de plus court chemin avec dominance modifiée. On peut vérifier le potentiel des méthodes COST et MASTER (algorithme utilisant les colonnes du problème maître) en dénombrant le nombre de chemins énumérés. Pour ce faire, les tests effectués consistent à résoudre les problèmes selon la méthode standard et à activer les algorithmes d'énumération sans évaluer de pénalités sur les chemins, puisque nous ne voulons pas modifier le branchement standard. Nous voulons comparer le nombre de chemins potentiels énumérés, pour des employés se trouvant avec un même choix de rotations possibles, ce qui implique que les employés précédents doivent avoir choisi les mêmes rotations dans chaque cas puisque nous voulons comparer pour des états semblables.

Les résultats pour l'algorithme COST sont présentés dans le tableau 5.3. Pour chaque problème étudié, nous indiquons d'abord le nombre d'employés pour lequel plus d'un

chemin était dénombré (# employés avec multi-choix) selon l'algorithme d'énumération considéré. Sont ensuite indiqués respectivement le nombre d'employés pour lesquels aucun chemin n'était énuméré et pour lesquels un seul chemin était trouvé. Le pourcentage d'employés ayant plus d'un horaire dénombré est ensuite présenté, déterminé en rapport avec le nombre d'employés BH. La distribution du nombre de chemins énumérés pour chaque employé ayant plus d'un chemin est ensuite présentée, suivie du nombre maximal de chemins trouvés. Le nombre de chemins en moyenne est ensuite indiqué, cette moyenne ne considérant que les employés ayant plus d'un chemin. La moyenne se voulant la plus significative possible, une ou deux valeurs trop éloignées de la médiane ont été écartées pour le calcul de la moyenne pour les cas indiqués par un astérisque dans le tableau.

Tableau 5.3 – Statistiques sur les chemins - algorithme COST

	YUL	YVR		YWG	YYZ
	767	320	767	320	340
# employés avec multi-choix	7	21	43	18	35
# employés avec 0 chemin	0	1	11	1	6
# employés avec 1 chemin	19	15	30	23	29
% employés avec multi-choix	26.9	56.8	51.2	42.9	50.0
# cas avec 2-99 chemins	6	19	36	14	30
# cas avec 100-999 chemins	0	0	3	4	4
# cas avec 1000-9999 chemins	0	0	3	0	1
# cas avec ≥ 10000 chemins	1	2	1	0	0
Nombre maximal de chemins	11291	22038	132865	414	2310
Nombre de chemins en moyenne	6.5*	12.7*	296.1*	55.4	38.2*
Nombre d'itérations requises	246	465	746	486	537
Temps d'énumération (sec)	34	131	26782	3603	14571

À titre comparatif, nous présentons dans le tableau 5.4 le nombre de chemins potentiels tels que déterminé en retenant toutes les colonnes du problème maître (algorithme MASTER) ayant le coût désiré. Tous les employés avaient alors au moins deux chemins de pointage optimal.

Tableau 5.4 – Statistiques sur les chemins - algorithme MASTER

	YUL	YVR		YWG	YYZ
	767	320	767	320	340
# employés avec multi-choix	25	37	83	41	70
# cas avec 2-99 chemins	25	35	80	41	70
# cas avec 100-200 chemins	0	2	3	0	0
Nombre maximal de chemins	16	119	150	51	91
Nombre de chemins en moyenne	5.2	16.9	12.7	5.0	8.0
Temps d'énumération (sec)	13	65	344	59	129

Cette dernière méthode a l'avantage d'être très rapide, mais nous ne pouvons dénombrer ainsi que les colonnes du problème maître correspondant à des chemins ayant le coût désiré. De nombreux chemins peuvent avoir le coût recherché sans pour autant correspondre à une colonne du problème maître. Il serait intéressant de comparer laquelle des deux méthodes d'énumération s'avère la plus intéressante pour déterminer l'horaire le moins pénalisant pour les employés suivants.

5.4 Tests comparatifs

Tel qu'énoncé précédemment, l'objectif est d'énumérer les différents horaires de pointage optimal pour l'employé courant, en évaluer la pénalité encourue par les employés suivants, et fixer ensuite le choix de telle sorte que les employés suivants seront le moins désavantagés possible. En ce sens, le but de la présente section est d'étudier

l'impact des deux algorithmes proposés (COST et MASTER), et ce de la façon la plus juste possible. Dans un premier temps, la démarche sera exposée afin de bien comprendre le processus d'évaluation. Les résultats obtenus avec chaque algorithme seront ensuite présentés pour chaque jeu de données, le tout sera suivi d'un sommaire des résultats.

5.4.1 Exposé de la démarche

Le processus de comparaison s'avère particulier puisque nous ne pouvons parler en termes de pourcentages nets d'amélioration. En effet, le pointage de chaque employé est relatif puisqu'il dépend des pondérations octroyées par l'employé concerné aux différentes tâches. Chaque employé pouvant pondérer à sa guise, un pointage bas pour un employé donné peut exprimer une satisfaction tout aussi élevée qu'un pointage très haut pour un autre employé, l'important étant que chacun obtienne le pointage maximal qui lui est propre, selon ses pondérations choisies.

De plus, il importe de bien comprendre que l'objectif total du problème n'est pas la somme des pointages (sur les employés). Étant donné une solution de pointages $(p_1, \dots, p_n)$, l'objectif consiste à trouver la solution dont le vecteur de pointages est maximal en ordre lexicographique. Considérons deux vecteurs de pointages $(p_1, \dots, p_n)$ et $(q_1, \dots, q_n)$ tels que $p_i = q_i$ pour $i = 1, \dots, j$. Ce que l'on veut vérifier s'énonce comme suit : si les horaires 1 à $j - 1$ sont identiques pour ces deux solutions et que l'horaire j est différent, bien que de même pointage, quel est l'impact de la décision courante j sur le restant des employés par rapport à l'ordre lexicographique des pointages ? En d'autres termes, on veut vérifier quel vecteur de pointage est le meilleur selon l'ordre lexicographique. Ce que nous proposons, c'est de regarder l'impact de la nouvelle décision sur les cinq employés suivants seulement.

Nous allons considérer des problèmes de tailles variables, mais le principe sera toujours le même. Pour un problème donné et une méthode d'énumération donnée (COST ou MASTER), les employés présentant plus d'un horaire de meilleur pointage auront préalablement été identifiés à l'aide de notre algorithme d'énumération et tous les pointages originaux auront été enregistrés dans un fichier de référence.

Le programme principal s'exécutera plusieurs fois consécutives, en concentrant notre algorithme sur un seul employé spécifique à chaque fois. Pour une exécution du programme, on demandera à notre algorithme de considérer l'énumération pour un seul employé à horaires multiples, de calculer les pénalités sur tous ces horaires et de brancher sur le meilleur choix. On notera l'impact sur le pointage des cinq employés suivants. Par exemple, si nous identifions n employés $e_1, e_2, \dots, e_n$ ayant plusieurs horaires de meilleur pointage, alors le programme principal sera exécuté n fois. La j -ème fois, on énumère et modifie le branchement pour l'employé e_j uniquement. On procède successivement ainsi, de l'employé e_1 jusqu'à l'employé e_n .

On regardera dans quelle proportion on améliore ou détériore le pointage des employés suivants, en regardant dans chaque cas quel est l'impact sur le premier des cinq employés suivants dont le pointage sera modifié par rapport à son pointage standard : Quel est le premier des employés suivants pour lequel on observe un impact ? Est-ce mieux ou pire que la solution standard ? Nous observerons ainsi jusqu'au cinquième employé suivant, la fonction de pénalité perdant progressivement de sa pertinence lorsque l'on s'éloigne de l'employé courant. Par exemple, si un des horaires dénombrés pour l'employé k ne modifie pas le pointage de l'employé $k + 1$, mais améliore celui de l'employé $k + 2$, on considère ceci comme une amélioration, peu importe ce qu'il advient des employés $k + 3$ et suivants. Mentionnons que tous les pourcentages présentés seront en fonction du nombre d'employés dénombrés à horaires multiples, et non en fonction du nombre total d'employés BH.

5.4.2 Algorithme COST

Cette section présente les résultats obtenus pour les tests comparatifs avec l'algorithme COST, et cela pour tous les jeux de données étudiés. Pour un problème donné, le tableau associé indique l'impact des nouvelles décisions sur les horaires des employés suivants (considérant le premier changement de pointage observé) : le pourcentage d'améliorations (% mieux), de dégradations (% pire) ou de retours en arrière (% retours). En effet, il peut arriver qu'une solution donnée entraîne que l'un des employés suivants ne puisse avoir d'horaire admissible, auquel cas la dernière décision doit être éliminée et l'on doit remonter au niveau précédent dans l'arbre de branchement externe, ce qui ne dégrade pas nécessairement la qualité de la solution mais requiert davantage de temps de résolution. Précisons que l'algorithme s'applique à chaque fois sur un nouvel employé courant k . Le tableau compile ainsi les impacts obtenus par les employés directement suivant $(k + 1)$, deuxième suivant $(k + 2)$, et ce jusqu'au cinquième suivant à chaque fois.

Sauf avis contraire, nous présentons les résultats obtenus lorsque la deuxième méthode d'évaluation et tri des chemins (voir chapitre 4) est utilisée. Dans les cas où la différence était jugée pertinente, les résultats obtenus avec la première méthode proposée seront également présentés ou commentés.

1er jeu de données. Si l'on considère le problème YUL-767 (tableau 5.5), l'utilisation de l'algorithme n'occasionne aucun changement quant au pointage des employés suivants. Cependant, on observe dans 3 cas (42.9%) des retours en arrière pour l'employé suivant. Peu de changements étaient observés suite à ces retours. Mentionnons qu'il s'agissait ici d'un problème de petite taille, où peu de marge de manoeuvre était permise.

Les résultats étaient sensiblement les mêmes indépendamment de la fonction de pénalité utilisée.

Tableau 5.5 – Résultats comparatifs : 1er jeu de données, algorithme COST

YUL-767						
	k+1	k+2	k+3	k+4	k+5	total
% mieux	0	0	0	0	0	0
% pire	0	0	0	0	0	0
% retours	42.9	0	0	0	0	42.9

2e jeu de données. Pour le problème YVR-320 (tableau 5.6), parmi les 21 employés à horaires multiples pour lesquels l'algorithme s'appliquait, 5 (23.8%) voyaient une amélioration pour l'employé suivant ($k+1$), 1 (4.8%) voyait une amélioration pour le 4e suivant, tandis que l'on observait des retours dans 6 cas (28.6%) pour l'employé suivant. Pour tous les autres cas, le pointage des employés suivants demeurerait inchangé.

On parvient donc à améliorer la solution dans 28.6% des cas, ce qui répond bien au but premier recherché. Toutefois, on observe également un fort pourcentage de retours conséquemment aux nouveaux branchements. Suite à ces six retours, l'incidence générale sur les employés suivants se dénombre comme suit : 3 meilleurs, 2 pires et 1 égal ensuite. Cependant, mieux vaudrait éviter un tel procédé qui peut s'avérer coûteux en temps de résolution.

Quelques différences ont été observées selon la fonction de pénalité utilisée. Notons toutefois qu'il y avait trois employés au lieu de cinq qui voyaient une amélioration pour le suivant, tandis que le nombre de retours augmentait à neuf dans le cas où l'on utilisait la première fonction de pénalité.

Tableau 5.6 – Résultats comparatifs : 2e jeu de données, algorithme COST

YVR-320						
	k+1	k+2	k+3	k+4	k+5	total
% mieux	23.8	0	0	4.8	0	28.6
% pire	0	0	0	0	0	0
% retours	28.6	0	0	0	0	28.6

3e jeu de données. Dans le cas du problème YVR-767, il faut noter que pour 45.2% des employés à horaires multiples, tous les horaires étaient de pénalités égales et donc sans incidence. Sinon, tel que montré au tableau 5.7, on note une amélioration dans 12 % des cas, et des retours dans 19.1% des cas. Le pourcentage de retours est donc très important. De plus, parmi les cas où la solution demeurerait inchangée pour les cinq employés suivants, on observait un très fort pourcentage de retours pour les employés ultérieurs, ce qui n'apparaît pas dans le tableau ci-contre. Il faut cependant remarquer qu'un cas de retour était déjà présent avec la résolution originale. Aucun test ne fût effectué avec l'autre fonction de pénalité, vu les temps de calcul considérables avec ce problème.

Tableau 5.7 – Résultats comparatifs : 3e jeu de données, algorithme COST

YVR-767						
	k+1	k+2	k+3	k+4	k+5	total
% mieux	4.8	4.8	2.4	0	0	12.0
% pire	0	0	0	0	0	0
% retours	14.3	4.8	0	0	0	19.1

4e jeu de données. Les résultats obtenus avec le problème YWG-320 tels que présentés au tableau 5.8 nous donnent 22.4 % d'améliorations. Toutefois, on observe un très fort taux de retours (61.2 %), et ces retours améliorent les pointages suivants dans seulement deux cas (non considérés comme améliorations dans le tableau).

Si l'on utilise l'autre fonction de pénalité, les résultats sont moins bons : moins d'améliorations (16.7 %), 1 cas pire (5.6 %), et davantage de retours (66.7 %).

Tableau 5.8 – Résultats comparatifs : 4e jeu de données, algorithme COST

YWG-320						
	k+1	k+2	k+3	k+4	k+5	total
% mieux	5.6	5.6	5.6	0	5.6	22.4
% pire	0	0	0	0	0	0
% retours	55.6	5.6	0	0	0	61.2

5e jeu de données. Soulignons la situation particulière dans le cas du problème YYZ-340 (tableau 5.9) puisqu'il y avait déjà des retours avec la résolution originale. Le tableau en étant un de comparaison avec les solutions originales, nous avons répertorié uniquement les nouveaux cas de retours occasionnés par les nouveaux choix. On observe un bon nombre de cas (15 sur 35) pour lesquels tous les horaires ont la même valeur de pénalité. Par conséquent, aucune modification ne sera alors apportée. Sur les 35 employés dénombrés à horaires multiples, 8.6% verront une amélioration pour les suivants, tandis que l'on observe un certain pourcentage de retours en arrière, soit 28.6%.

Quelques différences avec l'autre fonction de pénalité : 14.3% de cas avec amélioration, 5.7% de cas avec dégradation du pointage, et 20% de cas avec retours. Aucun cas avec horaires de pointages tous égaux.

5.4.3 Algorithme MASTER

Avec cet algorithme, nous dénombrons au moins deux horaires de pointage désiré pour chaque employé. Il serait intéressant d'observer quels sont les résultats obtenus

Tableau 5.9 – Résultats comparatifs : 5e jeu de données, algorithme COST

YYZ-340						
	k+1	k+2	k+3	k+4	k+5	total
% mieux	0	0	5.7	2.9	0	8.6
% pire	0	0	0	0	0	0
% retours	28.6	0	0	0	0	28.6

avec cet algorithme, seront-ils meilleurs qu'auparavant ? Remarquons qu'avec cet algorithme, il arrive très souvent que tous les horaires énumérés pour un employé donné aient la même valeur de pénalité (peu importe la fonction de pénalité utilisée). Dans de tels cas, les horaires sont considérés de valeur égale et le choix arbitraire n'est pas effectué, le branchement s'effectue alors sur l'horaire obtenu avec la méthode originale.

Les pourcentages indiqués dans les tableaux sont calculés en fonction du nombre d'employés ayant plusieurs horaires possibles, et non en fonction du nombre de cas où le branchement était modifié. Tel que précisé à la sous-section précédente, nous présentons dans les tableaux les résultats obtenus en utilisant la deuxième méthode de tri.

1er jeu de données. Pour le problème YUL-767 présenté au tableau 5.10, les horaires énumérés avaient tous la même valeur de pénalité pour la majorité des employés. Seuls deux d'entre eux présentaient des horaires de pénalités variées. Aucune amélioration n'a été obtenue parmi ces deux cas considérés, l'un d'eux occasionnait même un moins bon résultat.

Les différences observées si l'on utilise l'autre fonction de pénalité : quatre employés ont des horaires de pénalités variées, comparativement à seulement deux cas précédemment. On observe un retour arrière pour l'employé k+1 dont l'horaire est modifié, au lieu de la dégradation de son pointage. Sinon, tous les résultats étaient semblables.

Tableau 5.10 – Résultats comparatifs : 1er jeu de données, algorithme MASTER

YUL-767						
	k+1	k+2	k+3	k+4	k+5	total
% mieux	0	0	0	0	0	0
% pire	4.0	0	0	0	0	4.0
% retours	0	0	0	0	0	0

2e jeu de données. Avec le problème YVR-320 (tableau 5.11), davantage d'employés présentent des possibilités de modifications : 13 employés ont des horaires de pénalités variées, et 24 employés présentent des horaires considérés tous égaux. Sur les 37 employés dénombrés à horaires multiples, une amélioration pour les suivants est observée dans 5 cas (16.2 %), tandis que l'on observe un retour dans un seul cas (2.7 %). Suite à ce retour en arrière, les résultats étaient d'ailleurs améliorés pour les employés suivants.

Si l'on utilise plutôt l'autre fonction de pénalité, on observe alors une amélioration dans 13.5 % des cas, ainsi que 10.8 % de retours, ce qui apparaît beaucoup moins intéressant.

Tableau 5.11 – Résultats comparatifs : 2e jeu de données, algorithme MASTER

YVR-320						
	k+1	k+2	k+3	k+4	k+5	total
% mieux	13.5	0	0	2.7	0	16.2
% pire	0	0	0	0	0	0
% retours	2.7	0	0	0	0	2.7

3e jeu de données. Le problème YVR-767 dénombrait 83 employés à horaires multiples selon l'algorithme MASTER. Parmi ceux-ci, 20 employés avaient des horaires avec pénalités variées. Tel que présenté au tableau 5.12, une amélioration est observée

dans 4.8 % des cas (4 employés), de même que l'on observe de nouveaux retours dans une égale proportion. Il est à noter qu'un retour en arrière était déjà observé avec la méthode de résolution standard, il s'agit donc d'un problème difficile à résoudre. Vu les grands temps de calcul requis, les tests n'ont été effectués qu'avec la deuxième fonction de pénalité.

Tableau 5.12 – Résultats comparatifs : 3e jeu de données, algorithme MASTER

YVR-767						
	k+1	k+2	k+3	k+4	k+5	total
% mieux	2.4	1.2	1.2	0	0	4.8
% pire	0	0	0	0	0	0
% retours	4.8	0	0	0	0	4.8

4e jeu de données. Pour le problème YWG-320, sur les 41 employés dénombrés à horaires multiples, 10 ont un choix d'horaires de pénalités variées. Tel que montré au tableau 5.13, sur les 41 employés, une amélioration est observée dans 9.6 % des cas, tandis que l'on observe des retours dans 7.2 % des cas.

Pour ce problème, on obtient sensiblement les mêmes résultats en utilisant l'une ou l'autre des fonctions de pénalité.

Tableau 5.13 – Résultats comparatifs : 4e jeu de données, algorithme MASTER

YWG-320						
	k+1	k+2	k+3	k+4	k+5	total
% mieux	4.8	2.4	0	0	2.4	9.6
% pire	0	0	0	0	0	0
% retours	4.8	2.4	0	0	0	7.2

5e jeu de données. Pour le problème YYZ-340, sur les 70 employés dénombrés à horaires multiples, 13 ont des pénalités variées. Tel que présenté au tableau 5.13, on

note dans quatre cas (5.7 %) une amélioration pour les suivants, et un seul retour (1.4 %). Il faut noter que des retours étaient déjà présents avec la méthode standard, alors seuls les nouveaux retours ont été notés dans le tableau.

Avec l'autre fonction de pénalité, davantage d'employés (25 vs 13) présentent des horaires de pénalités variées. Cependant, dans deux cas les nouveaux résultats des suivants seront pires que l'original, et nous aurons également un cas supplémentaire de retour.

Tableau 5.14 – Résultats comparatifs : 5e jeu de données, algorithme MASTER

YYZ-340						
	k+1	k+2	k+3	k+4	k+5	total
% mieux	0	0	4.3	0	1.4	5.7
% pire	0	0	0	0	0	0
% retours	1.4	0	0	0	0	1.4

5.4.4 Sommaire

Dans l'ensemble, avec l'algorithme COST, les nouvelles décisions améliorent dans une certaine proportion les pointages des employés juniors. Cependant, on observe un très grand nombre de retours en arrière en contrepartie, ce qui augmente les temps de résolution. À titre comparatif, le tableau 5.15 présente un sommaire des résultats obtenus avec les différents problèmes, pour lesquels nous avons uniquement modifié le branchement des employés ayant plus d'un horaire énuméré. Ce tableau représente donc en quelque sorte dans quelle proportion l'utilisation de notre algorithme s'avère avantageuse pour les employés considérés.

Rappelons que tous les pourcentages indiqués dans les tableaux étaient calculés en proportion du nombre d'employés pour lequel plus d'un horaire avaient été préalablement énumérés. Alors que l'on dénombrait plus d'un horaire pour chaque employé

Tableau 5.15 – Impact partiel : algorithme COST

	YUL-767	YVR-320	YVR-767	YWG-320	YYZ-340	moyenne
% mieux	0	28.6	12.0	22.4	8.6	14.3
% pire	0	0	0	0	0	0
% retours	42.9	28.6	19.1	61.2	28.6	36.1

BH avec l'algorithme MASTER, il serait intéressant d'indiquer le pourcentage général d'amélioration obtenu avec l'algorithme COST, sur l'ensemble des employés BH, afin de pouvoir faire une comparaison plus juste. En effectuant la conversion pour obtenir les pourcentages sur l'ensemble des employés BH, on obtient pour l'algorithme COST les résultats par rapport à l'ensemble des employés BH, tels que présentés au tableau 5.16. Ce tableau indique donc l'impact général qu'aurait l'utilisation de notre algorithme sur l'ensemble des employés BH.

Tableau 5.16 – Impact général : algorithme COST

	YUL-767	YVR-320	YVR-767	YWG-320	YYZ-340	moyenne
% mieux	0	13.5	6.0	9.8	4.3	6.7
% pire	0	0	0	0	0	0
% retours	12.0	16.2	9.6	26.8	14.3	15.8

Tableau 5.17 – Impact général : algorithme MASTER

	YUL-767	YVR-320	YVR-767	YWG-320	YYZ-340	moyenne
% mieux	0	16.2	4.8	9.6	5.7	7.3
% pire	4.0	0	0	0	0	0.8
% retours	0	2.7	4.8	7.2	1.4	3.2

Les améliorations obtenues sont assez similaires avec l'algorithme MASTER, tel qu'indiqué au tableau 5.17. De plus, les retours en arrière apparaissent dans une proportion moindre, ce qui s'avère moins coûteux en temps de résolution.

Il ressort que des améliorations sont possibles, et cela ne devrait pas coûter trop cher en temps de calcul avec l'algorithme MASTER. L'algorithme COST occasionne plus de retours en arrière que MASTER, ce qui entraînerait plus de temps de calcul. En fait, il faudrait éventuellement comparer les temps de résolution. En guise d'exemple, on peut d'abord se référer aux temps requis lors des énumérations simples (cf. tableaux 5.2 et 5.3). Le passage dans la fonction de pénalité rajoute également un certain temps de calcul, sans compter bien sûr que les temps de résolution sont considérablement augmentés lorsque l'on modifie en plus les choix de branchement, occasionnant ainsi de nombreux retours en arrière tels que présentés plus haut.

5.5 Deux exemples plus détaillés

Il serait pertinent de s'interroger à savoir s'il est effectivement possible d'améliorer les solutions de base générées avec la méthode standard PBS. Pour certains problèmes (de taille raisonnable), nous avons ainsi jugé intéressant de tester toutes les possibilités de choix, pour voir dans quelle proportion nous pouvons effectivement espérer améliorer les résultats. Pour ce faire, on regarde successivement l'impact de la décision pour chacun des chemins énumérés, jusqu'à concurrence de dix chemins par employé. En plus d'indiquer si une amélioration est possible pour un des cinq employés suivants, on regarde à titre comparatif ce qu'il en est du premier choix tel que déterminé selon notre fonction de pénalité. Il va de soi que ce processus (tester toutes les possibilités, jusqu'à concurrence de dix, pour chaque employé) peut s'avérer très long. C'est pourquoi ce test particulier ne put être exécuté pour tous les problèmes considérés plus haut. Analysons ce qui arrive dans le cas où l'on utilise l'algorithme COST.

5.5.1 Un exemple de petite taille : YUL-767

Le problème en question étant de petite taille et ne requérant pas un grand temps de résolution, on peut fort bien effectuer l'analyse approfondie. Notons que les cas où un retour arrière occasionne par la suite de meilleurs pointages pour les suivants n'est pas considéré comme une amélioration, puisque nous évaluons selon la première conséquence rencontrée. Le tableau 5.18 présente les résultats ainsi compilés. La première colonne permet d'identifier tous les employés pour lesquels plus d'un chemin ont été dénombrés à l'aide de l'algorithme COST. La seconde colonne indique le nombre de chemins ayant été ainsi dénombrés et ici successivement testés (jusqu'à concurrence de dix). La troisième colonne permet de constater si un des chemins pouvait donner lieu à une amélioration pour un des cinq employés suivants. Enfin, la quatrième et dernière colonne présente l'impact effectif lorsque l'on utilise l'algorithme, ce qui correspond en d'autres termes à l'impact du premier choix tel que déterminé par la fonction de pénalité.

Tableau 5.18 – Résultats détaillés 1er jeu de données

YUL-767 Algorithme COST			
employé	#chemins	amélioration possible ?	impact effectif
5	2	non	nil
6	24	non	nil
11	2	oui : $k+2$	nil
13	6	non	<i>retour $k+1$</i>
14	11291	non	<i>retour $k+1$</i>
18	3	non	<i>retour $k+1$</i>
21	2	non	nil

D'après ce tableau, on observe qu'il était rarement possible d'améliorer la solution standard, sauf dans un cas. L'utilisation de notre fonction de pénalité ne faisait pas ressortir ce choix en premier. Bien sûr, il s'agit d'un très petit exemple, à partir

duquel on ne peut tirer de généralités. Précisons également que nous n'explorons pas au-delà du dixième chemin trié, alors une amélioration est peut-être possible dans les cas avec un grand nombre de chemins, mais nous avons décidé de ne pas tous les tester.

Si on explore plutôt les choix énumérés par la méthode MASTER, on observe également que la seule amélioration possible était aussi en $k+2$ (deuxième employé suivant) par rapport à l'employé 11. Il s'agit de la seule amélioration possible parmi tous les choix d'horaires dénombrés, et tous les employés avaient plus d'un horaire possible avec cette méthode.

5.5.2 Un exemple de taille moyenne : YVR-320

Ce problème de taille moyenne est assez représentatif, voyons alors la pertinence d'utiliser l'algorithme COST. Le tableau 5.19 présente les résultats ainsi obtenus.

En utilisant l'algorithme COST, on réussit ici à améliorer la solution lorsque possible. La fonction de pénalité choisie semble donc assez pertinente. Dans 7 cas (sur 21), il était donc possible d'améliorer le pointage des employés suivants. Toutefois, vu le grand nombre de nouveaux cas de retours arrière occasionné par la modification du branchement original, il y a lieu de remettre sérieusement en question l'idée d'utiliser cet algorithme.

Si on utilise plutôt l'algorithme MASTER, dans 8 cas (sur 37), il était possible d'améliorer le pointage des suivants, ce qui fut effectivement choisi et obtenu dans 6 cas. Remarquons que 2 des 8 cas d'amélioration potentielle comportaient des retours pour certains des employés ultérieurs.

Tableau 5.19 – Résultats détaillés 2e jeu de données

YVR-320 Algorithme COST			
employé	#chemins	amélioration possible ?	impact effectif
0	37	oui : k+4	k+4 meilleur
4	8	non	nil
6	2	non	nil
8	8	oui : k+1	k+1 meilleur
11	2	oui : k+1	k+1 meilleur
16	5	non	nil
17	2	non	nil
18	3	oui : k+1	k+1 meilleur
19	39	non	nil
20	6	non	nil
22	6	oui : k+4	k+4 meilleur
24	22038	non	<i>retour k+1</i>
25	2	non	nil
26	3	non	<i>retour k+1</i>
27	14822	non	<i>retour k+1</i>
28	68	non	<i>retour k+1</i>
30	3	non	<i>retour k+1</i>
31	5	non	<i>retour k+1</i>
32	5	oui : k+1	k+1 meilleur
36	6	non	nil
37	32	oui : k+1	k+1 meilleur

5.6 Temps requis au total

Les algorithmes proposés devraient améliorer la qualité des horaires des employés résiduels, sans pour autant requérir un trop grand temps de calcul. Les temps de calcul requis (en secondes) par la méthode de résolution standard (non modifiée) sont présentés dans le tableau 5.20 ci-bas. À titre comparatif, sont ensuite présentés les temps de calcul requis par les passages dans les algorithmes COST et MASTER, de même que les temps totaux de résolution lorsque l'on résout les problèmes en entier en modifiant le branchement pour prendre les horaires tels que dénombrés et triés selon la nouvelle méthode. Pour amoindrir légèrement le temps de calcul, nous avons limité le calcul des pénalités à mille horaires par employé. Dans un cas, ce choix a occasionné une erreur de résolution. De plus, il faut noter que nous avons trié les horaires en utilisant ici la deuxième fonction de pénalité (voir chapitre 4). Sans doute que les temps de calcul seraient moindres avec la première fonction de pénalité. Tous les temps sont indiqués en secondes.

Tableau 5.20 – Temps de résolution

	YUL	YVR		YWG	YYZ
	767	320	767	320	340
Temps original (sec)	13	65	344	59	129
Algo COST (sec)	113	605	27073	-	13646
Temps total COST (sec)	132	661	27828	-	13858
Algo MASTER (sec)	77	181	293	44	1088
Temps total MASTER (sec)	332	229	1229	115	1240

Selon ce tableau, l'algorithme MASTER s'avère plus rapide, sauf pour le plus petit problème. Considérant les temps présentés, l'algorithme COST consacre beaucoup de temps à l'énumération et à l'évaluation des chemins : on passe davantage de temps dans l'algorithme qu'en résolution propre. C'est le contraire avec l'algorithme

MASTER, où le temps de résolution propre est augmenté considérablement. Peu de temps est passé en comparaison dans l'algorithme (énumération et calcul de pénalité).

Vu l'augmentation des temps de résolution avec la nouvelle méthode, une alternative potentiellement intéressante serait de diminuer le nombre d'employés suivants qui sont pris en considération dans le calcul des pénalités. Il serait intéressant de reprendre la démarche et de tester ce qu'il advient lorsque l'on considère seulement les trois employés suivants (au lieu des cinq suivants) ou encore si l'on ne tient compte que du suivant. De plus, on pourrait calculer les pénalités sans normaliser par les plus courts et plus longs chemins des employés suivants, sauvant ainsi du temps de calcul. Quant au fait que nous avons restreint le nombre de chemins à évaluer, il serait bon de faire varier cette limite.

Conclusion

La construction d'horaires personnalisés avec priorité vise à attribuer à chaque employé un horaire maximisant sa satisfaction, de façon à assigner toutes les rotations prévues pour le mois courant. Le *Preferential Bidding System*, un logiciel développé par Ad Opt Technologies, construit les horaires de façon séquentielle, selon l'ordre d'ancienneté des employés, afin de privilégier les plus seniors. Advenant qu'il existe plusieurs horaires de pointage optimal pour un employé donné, le choix se fait sans évaluer l'impact sur les horaires des employés suivants.

La question que nous nous sommes posée est la suivante : Serait-il possible d'énumérer les différents horaires de pointage optimal pour un employé donné, de les évaluer et de choisir en conséquence celui qui pénalise le moins les employés suivants ? L'objectif visé étant d'en arriver à améliorer la qualité des horaires des employés plus juniors, en des temps de calcul raisonnables, si possible.

Synthèse des méthodes retenues

Concernant l'énumération, trois algorithmes principaux ont été retenus :

- **Algorithme COST** : On modifie le problème de plus court chemin afin de retenir tous les horaires ayant le coût recherché.
- **Algorithme DUAL** : On doit d'abord s'assurer de récupérer les bonnes variables duales, ensuite on modifie le problème de plus court chemin afin de retenir tous les horaires de coût réduit nul.
- **Algorithme MASTER** : On retient toutes les colonnes du problème maître qui

ont le coût désiré.

Deux méthodes d'évaluation ont été suggérées afin d'estimer l'incidence d'un horaire donné à l'employé courant. En d'autres termes, on veut évaluer la pénalité encourue par les employés suivants si l'employé courant choisit un horaire donné. La première fonction de pénalité se base sur l'attrait que peut représenter l'horaire pour les employés suivants. La seconde fonction enlève du réseau des employés suivants les tâches qui seront saturées si l'horaire considéré est choisi, et calcule en conséquence une nouvelle valeur pour le plus court chemin. Vu le peu de différence observée selon que l'on utilise l'une ou l'autre de ces fonctions, nous avons surtout présenté les résultats obtenus avec la seconde fonction.

Analyse des résultats

Suite à une première analyse, la méthode DUAL a été rejetée vu le grand nombre de cas où l'on ne parvenait pas à énumérer un seul horaire de pointage désiré. Pour ce faire, il aurait sans doute fallu resserrer la dominance ou encore augmenter l'espace-mémoire, deux solutions requérant de plus grands temps de calcul, et ces temps étaient déjà relativement élevés avec cette méthode. En conséquence, nous avons choisi d'étudier plus attentivement les deux autres méthodes. Concernant le nombre d'horaires de pointage désiré, on en énumère davantage en utilisant l'algorithme COST. Cependant, l'algorithme MASTER permet d'énumérer des horaires pour tous les employés dans des temps raisonnables, même si le nombre d'horaires en moyenne est moins élevé. Énumérer au moins un horaire pour tous aurait requis davantage de temps de calcul avec la méthode COST, mais le taux de réussite semble très acceptable malgré cela.

Les tests comparatifs ont par la suite permis de constater un grand nombre de retours en arrière lorsque l'on utilise l'algorithme COST, cet impact négatif ayant davantage d'occurrences que les améliorations. L'algorithme MASTER permet d'obtenir davantage de cas avec améliorations que de cas avec retours, ce qui est très positif. Bien que les améliorations ne touchent en moyenne que 7% des employés, il s'agit d'améliorations substantielles quand le système a pour objectif de construire le meilleur horaire pour chaque employé.

Si l'on considère deux exemples de façon plus détaillée, en testant successivement toutes les possibilités énumérées jusqu'à concurrence de dix par employé, il en ressort peu de possibilités d'amélioration par rapport aux décisions originales. Lorsque de telles améliorations sont possibles, le meilleur horaire correspond presque toujours à celui qui a été sélectionné grâce à notre fonction de pénalité. Ce dernier résultat vient valider en quelque sorte la pertinence de la fonction choisie.

Enfin, les temps de calcul sont considérablement plus élevés lorsque l'on utilise l'algorithme COST, où beaucoup de temps de calcul semble passé dans la fonction de pénalité.

Suggestions et améliorations proposées

De toutes les analyses ci-haut mentionnées, il appert clairement que l'algorithme MASTER semble le meilleur parmi ceux proposés, du moins en ce qui concerne les problèmes étudiés. Cet algorithme permet d'améliorer de façon significative la qualité des horaires des employés juniors. Il requiert toutefois plus de temps de calcul que la méthode standard.

Voici quelques suggestions afin de diminuer les temps de calcul, concernant la fonction de pénalité :

- Évaluer l'impact sur les trois employés suivants au lieu des cinq suivants.
- Ne pas normaliser la pénalité par les valeurs des plus courts et plus longs chemins de chacun des employés suivants. Une alternative intéressante serait de normaliser uniquement par les valeurs de plus court chemins.
- Restreindre davantage le nombre de chemins évalués, par exemple, en évaluer seulement 500.

Ces quelques suggestions seraient à étudier, en veillant à ne pas détériorer la valeur des horaires ainsi choisis.

Bibliographie

ANTOSIK, J.L. (1978). Automatic Monthly Crew Assignment, a New Approach. *AGIFORS Symposium Proceedings* 18, 369–402.

BUHR, J. (1978). Four Methods for Monthly Crew Assignment - A Comparison of Efficiency. *AGIFORS Symposium Proceedings* 18, 403–430.

BYRNE, J. (1988). A Preferential Bidding System for Technical Aircrew. *AGIFORS Symposium Proceedings* 28, 87–99.

CRAINIC, T.G. et ROUSSEAU, J.M. (1987). The Column Generation Principle and the Airline Crew Scheduling Problem. *INFOR* 39 (2), 136–151.

DANTZIG, G.B. et WOLFE, P. (1960). Decomposition Principle for Linear Programs. *Operations Research* 8, 101–111.

DESAULNIERS, G., DESROSIERS, J., DUMAS, Y., MARC S., RIOUX, B., SOLOMON M.M. et SOUMIS F. (1997). Crew Pairing at Air France. *European Journal of Operational Research* 97, 245–259.

DESROCHERS, M. (1988). An algorithm for the shortest path problem with resource constraints. Cahiers du GERAD G-88-27, École des Hautes Études Commerciales, Canada.

DESROCHERS, M. et SOUMIS F. (1988). A Generalized Permanent Labeling Algorithm for the Shortest Path Problem with Time Windows. *INFOR* 26, 191–212.

DESROSIERS, J., DUMAS, Y., SOLOMON, M.M. et SOUMIS, F. (1995). Time Constrai-

ned Routing and Scheduling. *Network Routing*, M.O. Ball et al. (eds), Handbooks in Operations Research and Management Science 8, 35–139.

EL IDRISSE, T. (2002). *Amélioration de la méthode des compteurs pour la construction des blocs mensuels personnalisés d'agents de bord*. Mémoire de maîtrise, École Polytechnique de Montréal, Canada.

GAMACHE, M. (1995). *Fabrication d'horaires mensuels pour les membres d'équipages en transport aérien*. Thèse de doctorat, École Polytechnique de Montréal, Canada.

GAMACHE, M. et SOUMIS, F. (1998). A Method for Optimality Solving the Rostering Problem. G. Yu, ed., *Operations Research in the Airline Industry*, Kluwer, Boston, 124–157.

GAMACHE, M. SOUMIS, F., DESROSIERS, J., VILLENEUVE, D. et GÉLINAS, É (1998). The Preferential Bidding System at Air Canada. *Transportation Science* 32, 246–255.

GAMACHE, M. SOUMIS, F., MARQUIS, G. et DESROSIERS, J. (1999). A Column Generation Approach for Large Scale Aircrew Rostering Problems. *Operations Research* 47(2), 247–262.

GERSHKOFF, I. (1989). Optimizing Flight Crew Schedules. *Interfaces* 19, 29–43.

GIAFFERRI, C., HAMON, J.P. et LENGLINE, J.G. (1982). Automatic Monthly Assignment of Medium-Haul Cabin Crew. *AGIFORS Symposium Proceedings* 22, 69–95.

GLANERT, W. (1984). A Timetable Approach to the Assignment of Pilots to Rotations. *AGIFORS Symposium Proceedings* 24, 369–391.

GONTIER, T. (1985). Longhaul Cabin Crew Assignment. *AGIFORS Symposium Proceedings* **25**, 49–66.

JEANDROZ, P. (2000). *Heuristique pour la construction de blocs mensuels personnalisés d'agents de bord*. Mémoire de maîtrise, École Polytechnique de Montréal, Canada.

JONES, R.D. (1989). Development of an Automated Crew Bid Generation System. *Interfaces* **19**, 44–51.

LAVOIE, S., MINOUX, M. et ODIER, E. (1988). A New Approach for Crew Pairing Problems by Column Generation with an Application to Air Transportation. *European Journal of Operational Research* **35**, 45–58.

MARSTEN, R. (1974). An Algorithm for Large Set Partitioning Problems. *Management Science* **20** (5), 774–787.

MARSTEN, R. et SHEPARDSON, F. (1981). Exact Solution of Crew Scheduling Problems Using the Set Partitioning Model : Recent Successful Applications. *Network* **11**, 165–177.

MOORE, R., EVANS, J. et NGO, H. (1978). Computerized Tailored Blocking. *1978 AGIFORS Symposium Proceedings* **24**, 343–361.

NICOLETTI, B. (1975). Automatic Crew Rostering. *Transportation Science* **9**, 33–42.

RYAN, D.M. et FALKNER, J.C. (1988). On the Integer Properties of Scheduling Set Partitioning Models. *European Journal of Operational Research* **35**, 442–456.

RYAN, D.M. (1992). The Solution of Massive Generalized Set Partitioning Problems in Air Crew Rostering. *Journal of the Operational Research Society* **43**, 459–467.

SARRA, D. (1988). The Automatic Assignment Model. *AGIFORS Symposium Proceedings* **28**, 23–37.

TINGLEY, G.A. (1979). Still Another Solution Method for the Monthly Aircrew Assignment Problem. *AGIFORS Symposium Proceedings* **19**, 143–203.

VANCE, P.H., BARNHART, C., JOHNSON, E.L. et NEMHAUSER, G.L. (1988). Airline Crew Scheduling : A New Formulation and Decomposition Algorithm. *Operations Research* **45**, 188–200.

VOVOR, T. (1997). *Problèmes de chemins bicritères ou avec contraintes de ressource : algorithmes et applications*. Thèse de doctorat, École Polytechnique de Montréal, Canada.

WILSON, B.J. (1981). BA's Regular O.R. Crew Planing Models for the 1980's. *1981 AGIFORS Symposium Proceedings* **21**, 257–270.