Les Cahiers du GERAD ISSN: 0711–2440

Ré-optimisation multi-objectif en temps réel suite à une petite perturbation

R. Hassani, G. Desaulniers,
I. El Hallaoui

G–2018–47

July 2018

La publication de ces rapports de recherche est rendue possible grâce au soutien de HEC Montréal, Polytechnique Montréal, Université McGill, Université du Québec à Montréal, ainsi que du Fonds de recherche du Québec – Nature et technologies.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2018
– Bibliothèque et Archives Canada, 2018

The publication of these research reports is made possible thanks to the support of HEC Montréal, Polytechnique Montréal, McGill University, Université du Québec à Montréal, as well as the Fonds de recherche du Québec – Nature et technologies.

Legal deposit – Bibliothèque et Archives nationales du Québec, 2018
– Library and Archives Canada, 2018

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Ré-optimisation multi-objectif en temps réel suite à une petite perturbation

Rachid Hassani
Guy Desaulniers
Issmail Elhallaoui

GERAD & Department of Mathematics and Industrial Engineering, Polytechnique Montréal (Québec) Canada, H3C 3A7

rachid.hassani@gerad.ca
guy.desaulniers@gerad.ca
issmail.elhallaoui@gerad.ca

July 2018
Les Cahiers du GERAD
G–2018–47

Abstract: Personnel scheduling aims at determining the cheapest work schedules to cover the demand for one or more tasks at each period of a given horizon. During the operations, several minor disruptions such as delays or absences of employees can occur and must be handled in real time without a significant change in the planned schedule. In this paper, we develop a fast re-scheduling heuristic that corrects minor disruptions in a context where employees can be assigned to a wide variety of shifts, starting and ending at different times. This heuristic consists of correcting a disruption by proposing a set of solutions that realize a good compromise between the cost and the number of modifications. The heuristic mainly uses a state graph whose construction and search for solutions corresponding to non-dominated paths, are based on a fundamental and "probabilistic" study. Computational experiments conducted on practical problem instances involving up to 95 employees have shown the effectiveness of the proposed heuristic. It can find all the exact solutions that realize a good compromise cost/modifications, in a search zone set by the employer, in less than one second on average for more than 96 % of the scenarios generated.

Résumé : Les problèmes de gestion de personnel visent à déterminer les horaires de travail les moins coûteux pour couvrir la demande d'une ou plusieurs tâches à chaque période d'un horizon donné. Durant l'opération, plusieurs petites perturbations, comme les retards ou les absences d'employés, surviennent et doivent être traitées en temps réel sans trop modifier l'horaire planifié. Dans cet article, nous développons une heuristique de ré-optimisation rapide qui corrige les petites perturbations dans un contexte où les employés peuvent être assignés à une grande variété de quarts, débutant et s'achevant à divers moments. Cette heuristique consiste à corriger la perturbation en proposant un ensemble de solutions qui réalisent un compromis entre le coût et le nombre de modifications. L'heuristique utilise principalement un graphe d'états dont la construction et la recherche des solutions correspondantes aux chemins non-dominés, sont basées sur une étude fondamentale et "probabiliste". Les tests numériques menés sur des instances de problèmes réels allant jusqu'à 95 employés ont montré l'efficacité de l'heuristique proposée. Celle-ci arrive à trouver toutes les solutions exactes qui réalisent un bon compromis coût/modifications, dans une zone de recherche fixée par l'employeur, en moins d'une seconde en moyenne pour plus de 96% des scénarios générés.

1 Introduction

Les problèmes de gestion d'horaires de personnel sont présents dans de nombreux domaines tels que la vente au détail, la santé, les services postaux, le transport ou la production industrielle. Le processus de gestion d'horaires de personnel peut varier d'un domaine à un autre. Mais, généralement, l'objectif demeure le même. Ce dernier consiste à construire un horaire de travail d'un ensemble d'employés sur un horizon de temps donné pour satisfaire la demande associée à un ensemble de tâches. La demande doit être satisfaite tout en respectant les différentes contraintes de travail liées aux employés, les contraintes syndicales et législatives, et en optimisant un critère, ou plusieurs critères de sélection entre les horaires possibles. Ces critères sont fixés par la nature du domaine (par exemple: les dépenses liées à la masse salariale, la qualité du travail fourni ou les préférences des employés). Trouver un tel horaire constitue un exercice très délicat. Dans ces conditions, l'employeur peut utiliser un logiciel d'optimisation.

Dans la pratique, les problèmes de gestion d'horaires de personnel sont sujets à beaucoup d'incertitude. En effet, les employés peuvent arriver en retard, s'absenter ou bien la demande observée peut également ne pas coïncider avec la demande prévue pour certaines périodes de l'horizon. Ces incidents, appelés petites perturbations, se révèlent petit à petit durant l'opération. Due à ces perturbations, l'horaire courant devient non réalisable. Une ré-optimisation est alors nécessaire. L'aspect "dynamique" qui caractérise l'arrivée de ces perturbations (ensemble de réalisations très grand) rend l'exercice de la ré-optimisation en avance quasi-impossible. D'où la nécessité de faire une ré-optimisation en temps réel. Le nouvel horaire, obtenu après ré-optimisation, doit être proche de l'horaire initial. En effet, on ne peut pas modifier le programme de tous les employés juste parce que l'un d'eux est en retard. La qualité du nouvel horaire sera alors évaluée sur la base de deux critères: le nombre de modifications et le coût engendré par ces modifications.

Dans cet article, nous proposons une heuristique efficace de ré-optimisation multi-objectif en temps réel des horaires de personnel suite à une petite perturbation causé par un incident de la liste de ceux cités ci-dessus. La méthode devra apporter très rapidement à l'employeur des choix de réadaptation des horaires et évaluer les coûts des modifications occasionnées en tenant compte des coûts immédiats (coût de gestion) et des coûts futurs déterministes (impact des modifications sur les horaires futurs et la rémunération des employés). On se place dans un contexte où les quarts de travail débutent, et s'achèvent à divers moments.

1.1 Revue de littérature

Au début des années 1950, Edie [4] a proposé une méthode heuristique dans le but d'optimiser le délai d'attente des véhicules qui se présentent aux postes de péage de New York. Quelques mois plus tard, Dantzig [3] a proposé de résoudre le même problème en utilisant un programme linéaire en nombres entiers, à savoir un problème de recouvrement. Ainsi, il a présenté une modélisation capable de fournir un horaire de personnel pour répondre aux besoins en question à moindre coût. L'étude de Dantzig est devenue une référence puisque, depuis, les chercheurs s'intéressent à l'amélioration de ce modèle pour, entre autres, l'adapter à d'autres problèmes de gestion de personnel. Des études complètes sur ces travaux ont été rédigées [16, 2, 5]. Au début des années 2000, les chercheurs ont commencé à s'intéresser au problème de la ré-optimisation d'un horaire prévu. La plupart des travaux ont abordé le problème de la ré-optimisation associé aux horaires des infirmières qui comprennent un nombre très limité de quarts de travail possibles (par exemple, 7am à 3pm, 3pm à 11pm, et 11pm à 7am). La demande est souvent exprimée par le nombre de quarts de travail requis pour une tâche. Dans ce contexte, une petite perturbation correspond à l'absence d'une infirmière (resp. des infirmières) pour un quart (resp. plusieurs quarts) de travail. Moz et Pato [12, 13] et Bard et Purnomo [1] ont proposé des algorithmes exacts de branchement pour résoudre les variantes de ce problème. Les temps de calcul de ces algorithmes étant démesurément longs, ils sont non applicables en temps réel. Des algorithmes heuristiques qui traitent du même problème ont également été développés: se référer au travail de Pato et Moz [12, 14], Pato et Moz [15], Maenhout et Vanhoucke [11], et Kitada et al. [9, 10]. Même si certains de ces algorithmes se sont avérés rapidement exécutables, ils conviennent davantage au domaine de la santé. Notre domaine d'application concerne la vente au détail. Ce domaine permet généralement beaucoup plus de flexibilité au niveau de l'horaire. En 2016, Gross et al. [7] ont travaillé sur la ré-optimisation des horaires des médecins, exerçant dans un hôpital, après l'absence de l'un d'entre eux en cherchant un compromis entre la qualité du nouvel horaire et la distance de ce dernier de l'horaire

initial. Pour ce faire, un programme linéaire mixte est résolu. Le temps de résolution pouvait aller jusqu'à 21 secondes. À noter que pour aboutir au meilleur compromis, il fallait faire tourner le programme plusieurs fois avec différents choix de paramètres. C'est d'ailleurs cette donnée qui a influencé principalement le temps total nécessaire pour choisir la bonne ré-optimisation.

En 2015, Froger [6] a étudié le domaine de la vente au détail et plus précisément les perturbations à grandes échelles: par exemple, une variation relativement importante de la demande est attendue sur un ou plusieurs jours en raison de mauvais temps ou d'une vente imprévue, etc. Elle a proposé une méthode de résolution exacte, à partir des quarts prévus pour chaque employé, puis a énuméré des quarts transformés selon certaines règles de modification. Enfin, un programme en nombres entiers a été résolu. Ce travail a des similitudes avec le nôtre. Le problème initial d'optimisation du personnel est le même et les modifications qui peuvent être apportées aux quarts de travail planifiés sont également les mêmes. Cependant, les tailles des perturbations différent et le temps disponible pour faire face à une perturbation est beaucoup moins important dans notre cas (quelques secondes contre quelques minutes). Par conséquent, l'approche de solution proposée par Froger [6] ne peut être appliquée à une petite perturbation en temps réel. À notre connaissance, aucune étude n'a été réalisée pour les petites perturbations dans les magasins de vente au détail ou dans des contextes similaires, excepté l'article récent de Hassani et al. [8]. Ils ont quant à eux proposé une heuristique qui utilise principalement l'information duale trouvée lors de la première optimisation. Cette information est actualisée après chaque correction d'une perturbation à l'aide de la méthode de régression M.A.R.S. Cette heuristique retourne des choix d'adaptation qui concernent, juste, le jour où la perturbation a eu lieu de telle façon que l'horaire du reste de la semaine demeure optimal ou quasi-optimal.

1.2 Contributions

Dans cet article, nous développons une heuristique qui se caractérise par l'efficacité et la rapidité. Cette heuristique ré-optimise l'horaire en temps réel une fois qu'une petite perturbation survient pour y remédier, en minimisant les coûts ainsi que le nombre de modifications à réaliser. Contrairement à ce qui été fait dans l'article de Hassani et al. [8], les modifications ici peuvent affecter le jour où la perturbation a eu lieu ainsi que les jours suivants jusqu'à la fin de l'horizon. À l'aide de certaines décisions de base qui permettent de générer tous les horaires possibles, notre heuristique vise à trouver les bonnes séquences, appelées politiques, qui conduisent à des solutions de bonne qualité, appelées solutions non-dominées, obtenues après une optimisation multi-objectif. Un graphe d'états est utilisé pour modéliser le problème. Notre heuristique se base sur une étude théorique de la structure des politiques qui conduisent à de telles solutions et une étude probabiliste qui réduit significativement la taille du graphe d'états considéré. L'heuristique proposée a été testée sur des instances de données réelles impliquant jusqu'à 95 employés. Notre méthode a alors permis de trouver toutes les solutions exactes non-dominées dans 96% des cas de test en moins d'une seconde en moyenne.

1.3 Structure de l'article

Ce document est organisé comme suit. Dans la section 2 on définit les notions que nous allons utiliser par la suite et on formule le problème. Dans la section 3, nous décrivons l'heuristique proposée avec les études théorique et probabiliste sur lesquelles cette heuristique est basée. Dans la section 4, nous présentons et analysons les résultats de nos expériences numériques. Enfin, les conclusions sont tirées dans la section 5.

2 Formulation du problème

Dans cette section, nous commençons par présenter quelques notions de base et définir formellement ce qu'est une petite perturbation.

2.1 Notions générales

On considère le programme linéaire en nombres entiers utilisé par Hassani et al. [8] pour trouver un horaire optimal (ou quasi-optimal) sur un horizon de dates $\mathcal{H}$, pour un ensemble d'employés désigné par $\mathcal{E}$ et pour un ensemble de tâches désigné par $\mathcal{W}$. L'horizon est discrétisé en périodes de u minutes. Soit $\mathcal{P}_{\mathcal{H}}$, l'ensemble

de ces périodes. La demande est exprimée en nombre d'employés nécessaires pour chaque tâche à chaque période. Ce programme se base sur des grands ensembles de quarts proposés, générés en avance par une heuristique, pour chaque employé afin d'affecter les quarts aux employés. La sous-couverture (avoir moins d'employés que la demande à une période donnée) est interdite et la sur-couverture (avoir plus d'employés que la demande à une période donnée) est pénalisée. Le coût d'un horaire x, noté $c(x)$, comporte ces pénalités et aussi les coûts de la main-d'œuvre. La structure de ces coûts assure une équité au niveau du nombre d'heures travaillées dans l'horizon. Soit $\mathcal{X}$ l'ensemble des solutions réalisables de ce programme. Pour une solution réalisable $x \in \mathcal{X}$, on note $\mathcal{S}_x$, l'ensemble des quarts présents dans l'horaire x. Ainsi $\mathcal{S}_x(e,h)$ représente le quart réservé pour l'employé $e \in \mathcal{E}$, le jour $h \in \mathcal{H}$. Par la suite, on caractérisera un quart $s \in \mathcal{S}_x$ par sa période de début $b_s \in \mathcal{P}_\mathcal{H}$, sa période de fin $f_s \in \mathcal{P}_\mathcal{H}$, sa longueur en périodes $l_s (= f_s - b_s + 1)$ et l'employé e_s affecté à ce quart dans l'horaire x. Nous ferons également l'hypothèse que ce quart n'a contribué qu'à une seule tâche $w_s \in \mathcal{W}$. Par abus de langage, on dit qu'un employé est affecté à un quart nul, noté $s_0(h)$, si cet employé est en repos le jour $h \in \mathcal{H}$. Dans ce cas, on a $\mathcal{S}_x(e,h) = s_0(h)$. Soit $x_0 \in \mathcal{X}$ la solution optimale retenue par l'employeur.

Durant l'opération, plusieurs petites perturbations, responsables de la non réalisabilité d'un horaire, surviennent. Ces petites perturbations peuvent résulter de retards ou d'absences d'un ou de plusieurs employés, de l'augmentation/baisse de la demande pour certaines périodes d'une journée donnée, d'imprévus qui ne permettent pas à certains employés de finir leurs heures de travail, etc. On s'intéresse dans ce document aux petites perturbations dues à un retard d'un employé car les autres situations peuvent être assimilées à celle du retard (voir Hassani et al. [8]). Une perturbation due à un retard sera caractérisée par:

- l'employé $\hat{e} \in \mathcal{E}$ en retard;
- le jour $\hat{h} \in \mathcal{H}$ où le retard a eu lieu;
- le quart $\hat{s} = \mathcal{S}_{x_0}(\hat{e}, \hat{h})$ réservé à l'employé $\hat{e}$ le jour $\hat{h}$;
- la durée du retard $\hat{l}$ en périodes ($\hat{l} < l_{\hat{s}}$);
- l'instant $\hat{p} \in \mathcal{P}_\mathcal{H}$ où l'employeur prend connaissance de la perturbation ($\hat{p} \leqslant b_{\hat{s}}$).

Dans ce cas, on parle de la petite perturbation $\hat{X} = (\hat{e}, \hat{h}, \hat{s}, \hat{l}, \hat{p})$. Lorsqu'une telle perturbation survient, l'horaire x_0 devient non réalisable. Ainsi, pour le corriger, une ré-optimisation est nécessaire pour les jours: $\hat{h}, \hat{h}+1, ..., |\mathcal{H}|$. On suppose qu'on ne possède aucune information sur la perturbation $\hat{X}$ avant l'instant $\hat{p}$, un instant qui peut coïncider avec l'instant du début de la perturbation $b_{\hat{s}}$. Ceci l'aspect contraignant de notre ré-optimisation en matière de temps de résolution. Lors de la ré-optimisation, il faut respecter les contraintes initiales (utilisées pour trouver x_0). De plus, le nouvel horaire doit être proche de l'horaire initial x_0 en termes de nombre de modifications utilisées au préalable pour obtenir cet horaire. Le coût ajouté dû à ces modifications doit être le moins élevé possible.

2.2 Décision génératrice, décision élémentaire et politique

On définit les trois décisions d^p, d^a et d^m, appelées décisions génératrices, suivantes:

- $d^p(s_1, s_2)$: consiste à permuter l'affectation des employés pour les quarts s_1 et s_2;
- $d^m(s_1, s_2, r)$: consiste à allonger le quart s_1 de façon à couvrir les r premières, ou dernières, périodes du quart s_2;
- $d^a(s_1, s_2)$: pour $s_1 = s_0(h)$ ou bien $s_2 = s_0(h)$, consiste à ajouter le quart s_2, absent initialement de l'horaire courant, et à l'affecter à l'employé e_{s_1}, et à supprimer le quart s_1.

La proposition 1 montre qu'à l'aide d'un horaire initial, et les décisions d^p, d^m et d^a, on peut générer toutes les solutions de $\mathcal{X}$. On note $\circ$ la relation de composition des décisions d^p, d^m et d^a. À noter que cette relation n'est pas commutative. Algébriquement, l'application de chaque décision, ou une séquence de ces décisions, peut être vue comme un vecteur de transition d entre une solution x et une autre solution $y = x + d$ qui n'est pas forcément réalisable.

Proposition 1 *Soit $x, y \in \mathcal{X}$. Il existe une séquence de décisions génératrices pour passer de x à y.*

Preuve. Soit $\bar{h}_x \in \mathcal{H}$ tel que pour $h < \bar{h}_x$ on a $x(h) = y(h)$ et $x(\bar{h}_x) \neq y(\bar{h}_x)$, avec $x(h)$ la restriction de la solution x à la journée h.

Pour tout employé e tel que $\mathcal{S}_x(e, \bar{h}_x) \neq \mathcal{S}_y(e, \bar{h}_x)$, si $s_0(\bar{h}_x) = \mathcal{S}_x(e, \bar{h}_x)$ alors on applique la décision $d^a(s_0(\bar{h}_x), \mathcal{S}_y(e, \bar{h}_x))$, si $s_0(\bar{h}_x) = \mathcal{S}_y(e, \bar{h}_x)$ alors on applique la décision $d^a(\mathcal{S}_x(e, \bar{h}_x), s_0(\bar{h}_x))$. On note $\vec{d}^a_{\bar{h}_x}$ le vecteur qui représente la séquence de ces décisions (prises jusqu'à présent).

Jusque-là, nous savons que les mêmes employés travaillent le jour $\bar{h}_x$ aux horaires $x^a = x + \vec{d}^a_{\bar{h}_x}$ et y. On se retrouve donc avec le même nombre de quarts dans les deux solutions x^a et y.

Avec la décision génératrice d^m, on peut adapter les longueurs des quarts. Une telle adaptation existe parce que y est réalisable, en particulier $y(\bar{h}_x)$ est réalisable. On note $\vec{d}^m_{\bar{h}_x}$ le vecteur représentant la séquence de ces décisions. Ceci nous amène à une autre solution $x^m = x^a + \vec{d}^m_{\bar{h}_x}$.

Dans les solutions x^m et y, les "mêmes" quarts ne sont pas nécessairement affectés aux mêmes employés. Pour remédier à cela, on applique une séquence de décisions d^p. Avec ces décisions, on peut passer de x à une autre solution z tel que $\bar{h}_z = \bar{h}_x + 1$. D'une façon récursive et analogique, on finit par trouver une solution f telle que $\bar{h}_f = |\mathcal{H}| + 1$, donc par définition $\forall h \leqslant |\mathcal{H}| \quad f(h) = y(h)$ donc $f = y$. □

L'application d'une décision génératrice peut conduire à une solution non réalisable. Dans ce cas, cela impliquera la nécessité d'appliquer d'autres décisions génératrices afin d'obtenir une solution réalisable.

Définition 1 *On appelle décision élémentaire d, toute séquence minimale de décisions génératrices faisant passer d'une solution $x \in \mathcal{X}$ à une solution $y = x + d \in \mathcal{X}$, i.e on se retrouve avec une solution non réalisable si on élimine une décision génératrice de cette séquence. Dans ce cas, on dit que y est une solution voisine de la solution x.*

On note par $\mathcal{D}(x)$ l'ensemble des décisions élémentaires admissibles à partir de la solution x. Chaque décision $d \in \mathcal{D}(x)$ permet de passer à une solution voisine de x, à savoir $y = x + d$. Cette décision génère un nombre de modifications n_d et engendre un coût c_d tel que:

$$c_d = c(y) - c(x).$$

À noter que l'on appelle modification, tout changement apporté à l'horaire d'un employé après la prise d'une séquence de décisions. Lorsque cette séquence apporte plusieurs changements à l'horaire d'un même employé pendant le même jour, ces changements seront comptabilisés par une seule modification (voir exemple 1 ci-bas).

Durant l'opération, lorsqu'une perturbation survient, il faut prendre une décision élémentaire, le plus rapidement possible, afin de rectifier la perturbation. Très souvent, le coût de cette décision est strictement positif. En effet, suite à ces modifications, certains employés tombent en sur-temps ou des nouvelles périodes en sur-couverture se produisent dans le nouvel horaire. Afin de réduire ce coût, il faut prendre une séquence de décisions élémentaires. Cette séquence de décisions génère alors une séquence de solutions réalisables dans le but d'aboutir à un compromis coût/nombre de modifications. Pour ce faire, on donne la définition 2.

Définition 2 *Soit $x \in \mathcal{X}$. On appelle δ une politique admissible de x, toute séquence de décisions élémentaires $(d_1, d_2, \ldots, d_{|\delta|})$ telle que:*

$$\forall i \in \{1, \ldots, |\delta|\} \quad d_i \in \mathcal{D}(x + \sum_{k=1}^{i-1} d_k).$$

On note $\mathcal{P}(x)$, l'ensemble de ces politiques et $\mathcal{X}_x$ l'ensemble des solutions atteignables de la solution x.

Proposition 2 *Pour tout $x \in \mathcal{X}$, on a $\mathcal{X}_x = \mathcal{X}$.*

Preuve. Soit $x \in \mathcal{X}$. Par définition, toute solution de $\mathcal{X}_x$ est réalisable. Alors $\mathcal{X}_x \subseteq \mathcal{X}$. On suppose, par l'absurde, que $\mathcal{X} \nsubseteq \mathcal{X}_x$. Alors, il existe une solution réalisable y telle que $y \notin \mathcal{X}_x$. D'après la proposition 1, il existe une séquence de décisions génératrices pour passer de x à y. On note par $D = d_1^{i_1} \circ d_2^{i_2} \circ d_3^{i_3} \circ d_4^{i_4} \circ d_5^{i_5} \circ ... \circ d_k^{i_k}$ cette séquence avec $i_j \in \{a, m, p\}$ pour tout $j \in \{1, 2, \ldots, k\}$. Soit k_0 le plus petit indice tel que $d_1^{i_1} \circ d_2^{i_2} \circ ... \circ d_{k_0}^{i_{k_0}}$ représente une décision élémentaire qui nous fait passer à une solution réalisable z_0. Soit k_1 l'indice le plus petit tel que $d_{k_0+1}^{i_{k_0+1}} \circ d_{k_0+2}^{i_{k_0+2}} \circ ... \circ d_{k_1}^{i_{k_1}}$ est une décision élémentaire qui nous fait passer à une solution réalisable z_1. Soit k_l le plus grand indice tel que $d_{k_{l-1}+1}^{i_{k_{l-1}+1}} \circ d_{k_{l-1}+2}^{i_{k_{l-1}+2}} \circ ... \circ d_{k_l}^{i_{k_l}}$ est une décision élémentaire aboutissant à une solution z_l. On considère la séquence restante de décision $d_r = d_{k_l+1}^{i_{k_l+1}} \circ d_{k_l+2}^{i_{k_l+2}} \circ ... \circ d_k^{i_k}$. On ne peut plus extraire une décision élémentaire de d_r. Autrement cela contredirait le fait que k_l soit le plus grand indice. D'autre part, d_r représente une séquence de décisions génératrices minimale qui nous conduit de la solution z_l, réalisable par construction, à la solution réalisable y. Ainsi, d_r est une décision élémentaire, alors $y \in \mathcal{X}_x$. Contradiction. Donc $\mathcal{X}_x \subseteq \mathcal{X}$, d'où:

$$\mathcal{X}_x = \mathcal{X}.$$

□

La proposition 2 nous permet de déduire qu'à partir de n'importe quelle solution réalisable, à l'aide d'une politique donnée, on peut atteindre n'importe quelle autre solution réalisable. En particulier, à partir de notre horaire initial x_0, après l'arrivée d'une perturbation $\hat{X} = (\hat{e}, \hat{h}, \hat{s}, \hat{l}, \hat{p})$, on peut atteindre n'importe quel autre horaire réalisable de $\hat{\mathcal{X}}$ tel que $\hat{\mathcal{X}}$ est l'ensemble des solutions réalisables qui corrigent la perturbation $\hat{X}$ et qui respectent les contraintes produites par le temps d'horloge (jusqu'à l'instant $\hat{p}$). On note par $\hat{\mathcal{P}}(x_0)$ l'ensemble des politiques conduisant à ces solutions.

Exemple 1 *On se place dans la situation suivante: l'employé $\hat{e}$ est en retard de 4 périodes. Les horaires des employés e_{s_1} et e_{s_3} sont flexibles contrairement à ceux des employés $e_{\hat{s}}$ et e_{s_2}. En effet, l'employé $e_{\hat{s}}$ ne peut travailler plus de 8 heures par jour et l'employé e_{s_2} doit travailler exactement 8 heures par jour (voir figure 1). Pour cet exemple on a: $\mathcal{D}(x_0) = \{d_1, d_2, d_3, d_4, d_5\}$ tel que:*

$$d_1 = d^a(\hat{s}, s_1, 4), \quad n_{d_1} = 2$$

$$d_2 = d^a(\hat{s}, s_2, 4) \circ d^a(s_2, s_3, 4), \quad n_{d_2} = 3$$

$$d_3 = d^p(\hat{s}, s_2), \quad n_{d_3} = 2$$

$$d_4 = d^p(\hat{s}, s_3), \quad n_{d_4} = 2$$

$$d_5 = d^a(\hat{s}, s_1, 3) \circ d^a(\hat{s}, s_2, 1) \circ d^a(s_2, s_3, 1), \quad n_{d_5} = 4$$

$$d_6 = d^a(\hat{s}, s_1, 2) \circ d^a(\hat{s}, s_2, 2) \circ d^a(s_2, s_3, 2), \quad n_{d_6} = 4.$$

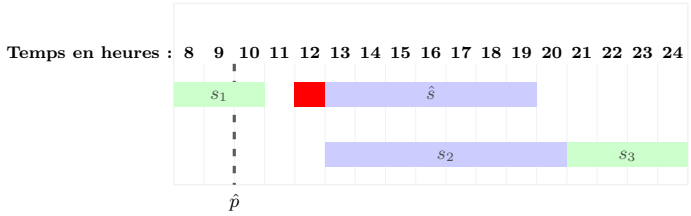

Figure 1: Exemple d'un horaire x_0 perturbé le jour $\hat{h}$.

2.3 Politiques et solutions non-dominées

On considère la fonction φ suivante:

$$\begin{aligned} \varphi : \hat{\mathcal{P}}(x_0) &\longrightarrow \mathbb{R} \times \mathbb{N}^* \\ \delta &\longmapsto (\varphi_c(\delta), \varphi_n(\delta)) \end{aligned}$$

telle que pour $\delta = (d_1, d_2, \ldots, d_{|\delta|}) \in \hat{\mathcal{P}}(x_0)$, on a:

$$\varphi_c(\delta) = \sum_{k=1}^{|\delta|} c_{d_k};$$
$$\varphi_n(\delta) = \sum_{k=1}^{|\delta|} n_{d_k}.$$

Cette fonction donne une valuation, pour toute politique δ de $\hat{\mathcal{P}}(x_0)$, caractérisée par les deux critères $\varphi_c(\delta)$ et $\varphi_n(\delta)$ qui représentent respectivement le coût total et le nombre total de modifications dues à la séquence de décisions élémentaires $(d_1, d_2, \ldots, d_{|\delta|})$. Ces deux critères, par essence sont contradictoires. En effet, plus on admet de modifications, plus on a de chance de faire baisser le coût. On introduit la relation d'ordre strict $\prec$ telle que:

$$\delta_1 \prec \delta_2 \Longleftrightarrow \{\varphi_c(\delta_1) \leqslant \varphi_c(\delta_2) \wedge \varphi_n(\delta_1) < \varphi_n(\delta_2)\} \ \vee \ \{\varphi_c(\delta_1) < \varphi_c(\delta_2) \wedge \varphi_n(\delta_1) \leqslant \varphi_n(\delta_2)\}$$

La relation $\prec$ est clairement une relation d'ordre partiel. Dans ce cas, deux politiques ne sont pas forcément comparables. On introduit alors les concepts de politique dominée ou non-dominée (définition 3). Une politique dominée (resp. non-dominée) nous ramène à une solution dominée (resp. non-dominée). L'ensemble des solutions non-dominées forme ce que l'on appelle la surface de compromis ou front de Pareto noté $\hat{\mathcal{X}}^*$. Le front de Pareto peut contenir un très grand nombre de solutions.

Définition 3 *Une politique δ est dite dominée s'il existe une politique $\delta^0 \in \hat{\mathcal{P}}(x_0)$ telle que $\delta^0 \prec \delta$. Si une telle politique n'existe pas alors la politique δ est dite non-dominée. On note par $\hat{\mathcal{P}}^*$ l'ensemble des politiques non-dominées.*

En pratique, toute solution de $\hat{\mathcal{X}}^*$ avec un coût ou un nombre de modifications assez élevé ne sera jamais adoptée par l'employeur. Pour cette raison, l'employeur fixe une zone de recherche paramétrée par les deux paramètres Φ_c et Φ_n, qui représentent des bornes supérieures respectivement sur les critères φ_c et φ_n. On s'intéresse alors aux politiques $\hat{\mathcal{P}}^*(\Phi_c, \Phi_n) = \{\delta \in \hat{\mathcal{P}}^*(x_0) \mid \varphi_c(\delta) \leqslant \Phi_c \ \wedge \ \varphi_n(\delta) \leqslant \Phi_n\}$ qui ramènent aux solutions non-dominées présentes dans la zone fixée par l'employeur. On note $\hat{\mathcal{X}}^*(\Phi_c, \Phi_n)$, l'ensemble de ces solutions.

3 Méthodologie

Dans la première partie de cette section, on propose le modèle ainsi que le graphe dynamique utilisé. Par la suite nous présentons les propositions sur lesquelles notre heuristique se base avant de décrire cette heuristique de manière détaillée.

3.1 Étude théorique sur les politiques non-dominées

On propose ici un programme mathématique qui nous permet de trouver les politiques recherchées. Ce modèle vise à sélectionner les décisions élémentaires de façon dynamique. Notre modèle utilise les notations et variables suivantes:

- T: le nombre maximal de décisions élémentaires sélectionnées;
- d_t: la représentation vectorielle de la décision élémentaire sélectionnée à l'étape t;

- x_t: la solution trouvée suite à la politique $(d_1, d_2, ..., d_t)$;
- y_d^t: vaut 1 si on choisit la décision $d \in \mathcal{D}(x_t)$, 0 sinon.

Le modèle:

$$\text{Minimiser} \qquad \varphi((d_1, d_2, ..., d_T)) = \left\{ \begin{array}{l} \varphi_c((d_1, d_2, ..., d_T)) \\ \varphi_n((d_1, d_2, ..., d_T)) \end{array} \right\} \tag{1}$$

$$\text{sujet à} : d_t = \sum_{d \in \mathcal{D}(x_t)} y_d^t d, \qquad \forall t \in \{1, ..., T\} \tag{2}$$

$$x_t = x_{t-1} + d_t, \qquad \forall t \in \{1, ..., T\} \tag{3}$$

$$\sum_{d \in \mathcal{D}(x_t)} y_d^t \leqslant 1, \qquad \forall t \in \{2, ..., T\} \tag{4}$$

$$\sum_{d \in \mathcal{D}(x_0)} y_d^1 = 1, \tag{5}$$

$$\varphi_c((d_1, d_2, ..., d_T)) \leqslant \Phi_c, \tag{6}$$

$$\varphi_n((d_1, d_2, ..., d_T)) \leqslant \Phi_n, \tag{7}$$

$$y_d^t \in \{0, 1\}, \qquad \forall t \in \{1, ..., T\}, \forall d \in \mathcal{D}(x_t). \tag{8}$$

La fonction objectif (1) est un vecteur de deux dimensions formé par les deux critères qu'on veut minimiser. Les contraintes (2) nous permettent de choisir, à chaque étape $t \in \{1, ..., T\}$, une décision élémentaire de $\mathcal{D}(x_t)$. La décision élémentaire choisie, à l'étape t, nous permet de passer à une autre solution réalisable x_t à l'aide des contraintes (3). D'autre part, les contraintes (4) imposent qu'à chaque étape $t \in \{2, ..., T\}$ on choisit au plus une décision élémentaire de $\mathcal{D}(x_t)$. Pour l'étape initiale, $t = 1$, on est obligé de choisir une décision élémentaire de $\mathcal{D}(x_0)$ vu que l'horaire initial n'est plus réalisable à cause de la perturbation $\hat{X}$ survenue. Cette situation est modélisée par la contrainte (5). Les contraintes (6) et (7) définissent la zone de recherche fixée par l'employeur. Enfin, les contraintes (8) sont des contraintes d'intégralité sur les différentes variables. On peut représenter ce programme à l'aide d'un graphe d'états orienté. En effet, soit $R(\mathcal{N}, \mathcal{A})$ un réseau avec $\mathcal{N}$ l'ensemble de sommets et $\mathcal{A}$ l'ensemble d'arcs. On associe la solution initiale x_0 au sommet $s \in \mathcal{N}$ source de ce réseau. On note par $t \in \mathcal{N}$ le puits de $R(\mathcal{N}, \mathcal{A})$. Chaque décision élémentaire d_i est associée à un sommet $v_i \in \mathcal{N}$ (et ainsi à une solution). La source s est connectée aux sommets v_i tel que $d_i \in \mathcal{D}(x_0)$. De plus, l'arc (v_i, v_j) existe, si la décision d_j est admissible à partir de la solution trouvée après la prise de la décision d_i. Le seul arc entrant à chaque sommet v_i est associé à un coût c_{d_i} et un nombre de modifications n_{d_i}. À noter que tous les sommets, excepté le sommet s, sont connectés au sommet t par un arc de coût et de nombre de modifications nuls. On partitionne l'ensemble des sommets $\mathcal{N}$ en η ($\eta \leqslant \Phi_n$) sous-ensembles $(\mathcal{N}_i)_{0 \leqslant i \leqslant \eta}$, tel que $\mathcal{N}_i$ contient les sommets associés aux décisions admissibles après la prise de i décisions élémentaires (voir la figure 2). Par abus de langage, on dit qu'une décision élémentaire d est dans $\mathcal{N}_i$, si cette décision est associée à un sommet de $\mathcal{N}_i$. Chaque chemin de s à t représente une politique de $\hat{\mathcal{P}}(\Phi_c, \Phi_n)$. Notre objectif est de trouver les chemins "non-dominés" qui correspondent aux politiques de $\hat{\mathcal{P}}^*(\Phi_c, \Phi_n)$. On note par $\hat{\mathcal{C}}^*(\Phi_c, \Phi_n)$ l'ensemble de ces chemins. On a:

$$|\mathcal{N}| = 2 + \sum_{d_1 \in \mathcal{D}(x_0)} \sum_{d_2 \in \mathcal{D}(x_0 + d_1)} \sum_{d_3 \in \mathcal{D}(x_0 + d_1 + d_2)} ... \sum_{d_\eta \in \mathcal{D}(x_0 + d_1 + .. + d_{\eta-1})} |\mathcal{D}(x_0 + d_1 + .. + d_\eta)|;$$

$$|\mathcal{A}| = 2(\sum_{d_1 \in \mathcal{D}(x_0)} \sum_{d_2 \in \mathcal{D}(x_0 + d_1)} \sum_{d_3 \in \mathcal{D}(x_0 + d_1 + d_2)} ... \sum_{d_\eta \in \mathcal{D}(x_0 + d_1 + .. + d_{\eta-1})} |\mathcal{D}(x_0 + d_1 + .. + d_\eta)|).$$

La taille du graphe de la figure 2 peut être très grande. Donc la recherche des chemins de $\hat{\mathcal{C}}^*(\Phi_c, \Phi_n)$ en temps réel est impossible, voir même la construction de ce graphe. Pour cette raison, nous allons essayer de générer les décisions élémentaires qui ont le plus de chance d'être choisies. Pour cela, nous avons réalisé une étude théorique sur les structures des politiques non-dominées.

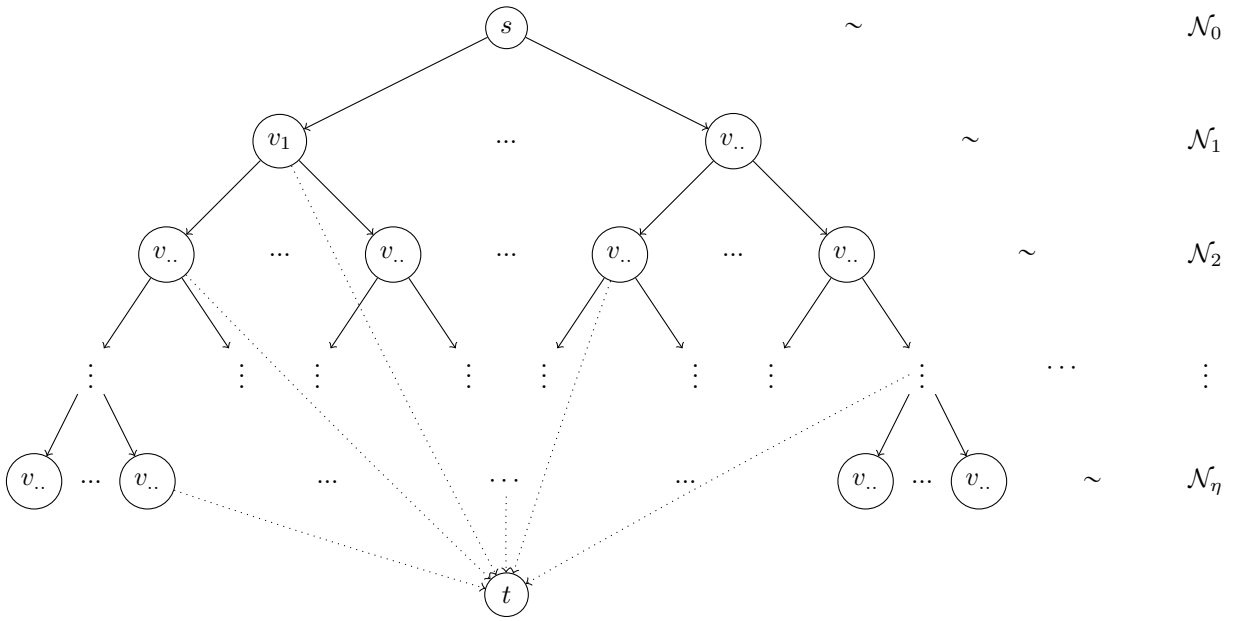

Figure 2: Graphe d'états $R(\mathcal{N}, \mathcal{A})$.

Soit $\delta_i^* \in \hat{\mathcal{P}}^*(\Phi_c, \Phi_n)$ tel que:

$$\forall i \in \{c, n\} \quad \delta_i^* \in \underset{\delta \in \hat{\mathcal{P}}(\Phi_c, \Phi_n)}{\operatorname{argmin}} \ (\varphi_i(\delta)).$$

Par définition, $\delta_i^* \in \hat{\mathcal{P}}(\Phi_c, \Phi_n)$, $\forall i \in \{c, n\}$. Les deux politiques δ_c^* et δ_n^* réduisent significativement l'espace de recherche (figure 3). De plus, les deux solutions correspondantes à ces deux politiques représentent, par la suite, une référence pour l'employeur sur la qualité de la zone de recherche initiale fixée (pour traiter d'autres perturbations similaires). Notre modélisation nous assure que si on restreint notre recherche au premier niveau $\mathcal{N}_1$ du graphe d'états, on peut trouver la politique δ_n^* (proposition 3). Il est un peu plus difficile de trouver δ_c^*. Pour cela, notre heuristique se base sur la proposition 5 afin d'estimer cette politique.

Proposition 3 *Il existe une décision élémentaire* d *de* $\mathcal{N}_1$ *tel que:*

$$\delta_n^* = (d)$$

Preuve. Par l'absurde on suppose que $\delta_n^* = (d_1, d_2, ..., d_k)$ avec $2 \leqslant k \leqslant \eta$. On a $\delta = (d_1, d_2, ..., d_{k-1})$ une politique telle que: $\varphi_n(\delta) < \varphi_n(\delta_n^*)$ (car le critère φ_n est additif). Cela est impossible, donc $k = 1$. □

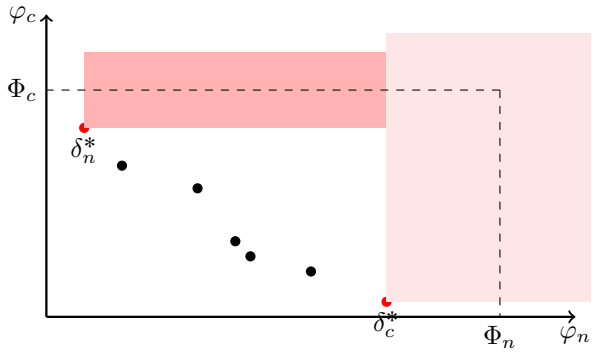

Figure 3: Forme de $\hat{\mathcal{P}}^*(\Phi_c, \Phi_n)$.

Proposition 4 *Soit* $\delta = (d_1, d_2, ..., d_k) \in \hat{\mathcal{P}}^*(\Phi_c, \Phi_n)$. *Si* $k \geqslant 2$ *alors* $\exists\ l_\delta \in \{2, 3, ..., k\}$ *tel que:*

$$c_{d_{l_\delta - 1}} \geqslant 0 \quad et \quad \forall i \geqslant l_\delta \quad c_{d_i} < 0.$$

Preuve. Soit $\delta = (d_1, d_2, ..., d_k) \in \hat{\mathcal{P}}^*(\Phi_c, \Phi_n)$ avec $k \geqslant 2$. Si on suppose par l'absurde que $c_{d_k} \geqslant 0$, alors dans ce cas la politique δ est dominée par la politique $(d_1, d_2, ..., d_{k-1})$. Or $\delta \in \hat{\mathcal{P}}^*(\Phi_c, \Phi_n)$. Contradiction. Donc $c_{d_k} < 0$ alors on peut déduire que l_δ existe. D'une façon itérative, en vérifiant si $c_{d_{k-1}} < 0, c_{d_{k-2}} < 0, ...$, on peut déduire que $l_\delta \in \{2, 3, ..., k\}$. □

Proposition 5 *Soit* $\delta_k = (d_1, d_2, ..., d_k) \in \hat{\mathcal{P}}(x_0)$ *tel que:*

$$c_{d_i} \leqslant \frac{-\varphi_n(d_1, d_2, ..., d_{i-1})}{\Phi_n}\ \varphi_c(d_1, d_2, ..., d_{i-1}), \quad \forall i \geqslant l.$$

Alors:

$$\delta_k \xrightarrow{\text{pour } k-l \text{ assez grand}} \delta_c^*$$

Preuve. Pour chaque $i \in \{l, ..., k\}$ on pose $\delta_i = (d_1, d_2, ..., d_i)$. On a:

$$\begin{aligned} c_{d_i} \leqslant \frac{-\varphi_n(\delta_{i-1})}{\Phi_n}\varphi_c(\delta_{i-1}) \Rightarrow & \varphi_c(\delta_i) \leqslant \varphi_c(\delta_{i-1}) - \frac{\varphi_n(\delta_{i-1})}{\Phi_n}\varphi_c(\delta_{i-1}) \\ \Rightarrow & \varphi_c(\delta_i) \leqslant \varphi_c(\delta_{i-1})\Big(1 - \frac{\varphi_n(\delta_{i-1})}{\Phi_n}\Big) \end{aligned}$$

Donc

$$\varphi_c(\delta_k) \leqslant \varphi_c(\delta_{l-1}) \prod_{i=l}^{k} \Big(1 - \frac{\varphi_n(\delta_{i-1})}{\Phi_n}\Big).$$

D'autre part, on a : $\varphi_n(\delta_1) \leqslant \varphi_n(\delta_2) \leqslant ... \leqslant \varphi_n(\delta_k)$, donc:

$$\varphi_c(\delta_k) \leqslant \varphi_c(\delta_{l-1}) \prod_{i=l}^{k} \Big(1 - \frac{\varphi_n(\delta_{i-1})}{\Phi_n}\Big) \Rightarrow \varphi_c(\delta_k) \leqslant \varphi_c(\delta_{l-1})\Big(1 - \frac{\varphi_n(\delta_{l-1})}{\Phi_n}\Big)^{k-l}$$

Or $\varphi_n(\delta_{l-1}) < \Phi_n$ et $1 - \frac{\varphi_n(\delta_{l-1})}{\Phi_n} < 1$. D'où:

$$\Big(1 - \frac{\varphi_n(\delta_{l-1})}{\Phi_n}\Big)^{k-l} \xrightarrow{\text{pour } k-l \text{ assez grand}} 0$$

et vu que x_0 est optimale alors $\varphi_c(\delta_k) \geqslant 0$. On en déduit que:

$$\delta_k \xrightarrow{\text{pour } k-l \text{ assez grand}} \delta_c^*$$

□

La proposition 4 nous permet de déduire qu'après la prise d'un certain nombre de décisions élémentaires, on aura besoin de générer que des décisions élémentaires de coût négatif, étant donné que seules de telles décisions peuvent conduire à des solutions non-dominées. Algébriquement, cela consiste à générer que des directions descendantes. Cela est très efficace, parce que ça réduira significativement le nombre de décisions élémentaires, donc la taille du graphe.

Il est intéressant de noter que l'existence d'une décision élémentaire dans une politique non dominée peut nous aider à prévoir l'existence d'une ou de plusieurs autre(s) décision(s) dans la même politique. En effet, si une décision amène un employé e_1 à travailler deux heures de plus et un employé e_2 à travailler deux heures de moins, l'employeur tentera alors de rétablir l'équilibre en faisant travailler e_1 (resp. e_2) moins (resp. plus). Pour cela, il est très probable qu'une décision ou des décisions qui ont cet objectif (faire travailler e_1 un peu

moins et e_2 un peu plus) soient prises. Pour une décision élémentaire d (resp. pour une politique δ), on note $\mathcal{E}_d$ (resp. $\mathcal{E}_\delta$) l'ensemble des employés concernés par les modifications entraînées par la décision d (resp. par la politique δ). Soit $\delta = (d_1, d_2, ..., d_k)$ une politique faisant passer de la solution x_0 à une solution réalisable x_δ. On définit la fonction μ_d qui va nous aider à "mesurer" la compatibilité de chaque décision de $\mathcal{D}(x_\delta)$ avec la séquence $\delta = (d_1, d_2, ..., d_k)$:

$$\begin{aligned}\mu_\delta : \mathcal{D}(x_\delta) &\longrightarrow [0,1] \\ d &\longmapsto \underbrace{\sqrt[k]{\frac{|\mathcal{E}_\delta \cap \mathcal{E}_d|}{|\mathcal{E}_\delta \cup \mathcal{E}_d|}}}_{1er\ terme} \cdot \underbrace{\frac{\min\limits_{d' \in \mathcal{D}(x_\delta)} (n_{d'})}{n_d}}_{2e\ terme} \cdot \underbrace{\prod_{e \in \mathcal{E}_\delta \cap \mathcal{E}_d} \frac{\min(N_{x_0}(e), N_{x_\delta + d}(e))}{\max(N_{x_0}(e), N_{x_\delta + d}(e))}}_{3e\ terme}\end{aligned}$$

où $N_{x_0}(e)$ (resp. $N_{x_\delta+d}(e)$) représente le nombre de périodes de travail programmées à l'employé e dans l'horaire x_0 (resp. $x_\delta + d$). Le premier terme mesure l'implication de nouveaux employés dans la mise à jour. Le deuxième terme évalue la consommation d'une décision d par rapport à la consommation minimale possible. Le troisième terme mesure la modification apportée sur la durée de travail totale de chaque employé de $\mathcal{E}_\delta$. La fonction $\sqrt[k]{\cdot}$ sert à faire une pondération favorisant le premier terme. Cette pondération est paramétrée par k (nombre de décisions déjà prises). En effet, plus on prend de décisions, plus d'employés ont des chances d'être impliqués, ce qui entraînera beaucoup de modifications. A cause de la borne Φ_n, on risque de générer un nombre très important de politiques dominées par δ_c^*. On se base sur la proposition 6 afin d'éviter cela, en générant les décisions élémentaires d telles que $\mu_\delta(d)$ est relativement proche de 1.

Proposition 6 *Soit $\delta = (d_1, d_2, .., d_k)$ une politique qui conduit à la solution x_δ.*

$$Si\ \ \exists d \in \mathcal{D}(x_\delta) \mid \mu_\delta(d) = 1\ alors\ (d_1, d_2, ..., d_k, d) \prec \delta^{'}\ pour\ toute\ \delta^{'} = (d_1, \ldots, d_k, d_{k+1}, \ldots).$$

Preuve. On suppose qu'il existe une décision élémentaire de $\mathcal{D}(x_\delta)$ tel que $\mu_\delta(d) = 1$.

On a:

$$\mu_\delta(d) = \sqrt[k]{\frac{|\mathcal{E}_\delta \cap \mathcal{E}_d|}{|\mathcal{E}_\delta \cup \mathcal{E}_d|}} \cdot \frac{\min\limits_{d' \in \mathcal{D}(x_\delta)} (n_{d'})}{n_d} \cdot \prod_{e \in \mathcal{E}_\delta \cap \mathcal{E}_d} \frac{\min(N_{x_0}(e), N_{x_\delta + d}(e))}{\max(N_{x_0}(e), N_{x_\delta + d}(e))} = 1.$$

Or

$$\sqrt[k]{\frac{|\mathcal{E}_\delta \cap \mathcal{E}_d|}{|\mathcal{E}_\delta \cup \mathcal{E}_d|}} \leqslant 1 \quad , \quad \frac{\min\limits_{d' \in \mathcal{D}(x_\delta)} (n_{d'})}{n_d} \leqslant 1 \quad \text{et} \quad \forall e \in \mathcal{E}_\delta \cap \mathcal{E}_d \quad \frac{\min(N_{x_0}(e), N_{x_\delta + d}(e))}{\max(N_{x_0}(e), N_{x_\delta + d}(e))} \leqslant 1$$

donc

$$\frac{|\mathcal{E}_\delta \cap \mathcal{E}_d|}{|\mathcal{E}_\delta \cup \mathcal{E}_d|} = 1 \quad , \quad \frac{\min\limits_{d' \in \mathcal{D}(x_\delta)} (n_{d'})}{n_d} = 1 \quad \text{et} \quad \forall e \in \mathcal{E}_\delta \cap \mathcal{E}_d \quad \frac{\min(N_{x_0}(e), N_{x_\delta + d}(e))}{\max(N_{x_0}(e), N_{x_\delta + d}(e))} = 1$$

$$\Rightarrow \begin{cases} \mathcal{E}_\delta \cap \mathcal{E}_d = \mathcal{E}_\delta \cup \mathcal{E}_d \\ \min\limits_{d' \in \mathcal{D}(x_\delta)} (n_{d'}) = n_d \\ \min(N_{x_0}(e), N_{x_\delta + d}(e)) = \max(N_{x_0}(e), N_{x_\delta + d}(e)) \quad \forall e \in \mathcal{E}_\delta \cap \mathcal{E}_d \end{cases}$$

$$\begin{aligned} &\Rightarrow \begin{cases} \mathcal{E}_\delta = \mathcal{E}_d \\ N_{x_0}(e) = N_{x_\delta + d}(e) \quad \forall e \in \mathcal{E}_\delta \end{cases} \\ &\Rightarrow c(x_0) = c(x_\delta + d) \\ &\Rightarrow \varphi_c((d_1, d_2, \ldots, d_k, d)) = 0. \end{aligned}$$

On suppose par l'absurde qu'il existe une politique $\delta^{'} = (d_1, \ldots, d_k, d_{k+1}, \ldots, d_v)$ $(v \geqslant k+1)$ telle que $\delta^{'} \prec (d_1, d_2, \ldots, d_k, d)$. Donc:

$$\varphi_c((d_1, \ldots, d_k, d_{k+1}, \ldots, d_v)) = 0 \text{ et } \varphi_n((d_1, \ldots, d_k, d_{k+1}, \ldots, d_v)) < \varphi_n((d_1, \ldots, d_k, d))$$

puisque:

$$\begin{aligned}\varphi_n((d_1,\ldots,d_k,d_{k+1},\ldots,d_v)) < \varphi_n((d_1,\ldots,d_k,d)) &\Rightarrow \sum_{i=1}^{v} n_{d_i} < \sum_{i=1}^{k} n_{d_i} + n_d \\ &\Rightarrow \sum_{i=k+1}^{v} n_{d_i} < n_d \\ &\Rightarrow n_{d_{k+1}} + \sum_{i=k+2}^{v} n_{d_i} < n_d\end{aligned}$$

alors cela est impossible, d'où le résultat. □

3.2 Heuristique

On définit dans le tableau ci-dessous les fonctions qui vont nous aider par la suite dans la description de notre heuristique.

`findElementaryDecisions`(x, d, m)	Cette fonction retourne l'ensemble de décisions élémentaires qui ramènent à une solution voisine de la solution x. Chacune d'elles est représentée par une séquence formée d'au plus m décisions génératrices telle que la première décision génératrice dans cette séquence est la décision d.
`correctiveGeneratorDecisions`$(\hat{X})$	Cette fonction retourne l'ensemble de décisions génératrices qui corrigent la perturbation $\hat{X}$ dans l'horaire x_0.
`estimateDelta2`$(\mathcal{N}_1)$	Cette fonction estime la politique δ_c^* en se basant sur la proposition 5.
`dominance`$(\mathcal{C})$	Cette fonction retourne les chemins de $\mathcal{C}$ non-dominés.
`applySequenceDecisions`(δ)	Consiste à appliquer la séquence δ sur l'horaire x_0.
`candidateElementaryDecisions`(x, m)	Cette fonction retourne un ensemble de cardinalité assez important, contenant des décisions élémentaires de taille inférieure à m admissibles à partir de la solution courante x. Cette fonction se base sur les modifications déjà apportées sur l'horaire x_0.
`paths`$(\mathcal{N}_i)$	Cette fonction retourne l'ensemble des chemins de s à t en prenant juste en compte les niveaux $\mathcal{N}_1, \mathcal{N}_2, ..., \mathcal{N}_i$.
`propagate`$(\delta, \Phi_c, \Phi_n, m, l, \epsilon)$	Cette fonction renvoie un ensemble de décisions élémentaires choisies par l'algorithme 2. Cela représentera un sous-ensemble de $\mathcal{N}_{\lvert\delta\rvert+1}$.
`policy`(d)	Consiste à donner la séquence de décisions élémentaires qui conduisent de la solution x_0 à la solution courante trouvée après la prise de la décision élémentaire d.

Considérons une petite perturbation définie par $\hat{X} = (\hat{e}, \hat{h}, \hat{s}, \hat{l}, \hat{p})$. L'algorithme 1 vise à rendre l'horaire x_0 réalisable, en cherchant les chemins non-dominés $\hat{\mathcal{C}}^*(\Phi_c, \Phi_n)$ associés aux politiques non-dominées $\hat{\mathcal{P}}^*(\Phi_c, \Phi_n)$ dans la zone de recherche fixée par l'employeur à l'aide des deux paramètres Φ_c et Φ_n. Dans un premier temps, il faut trouver les décisions génératrices qui corrigent la perturbation $\hat{X}$ (étape 1). Puisque ces décisions peuvent conduire à des horaires non-réalisables, il est nécessaire à l'étape 4 de trouver, si possible, la séquence qui rend chacun de ces horaires réalisables. L'étape 7 nous permet de trouver les chemins non-dominés $\mathcal{C}_2$ associés aux décisions élémentaires du premier niveau $\mathcal{N}_1$. La proposition 3 nous confirme que parmi ces chemins se trouve le chemin associé à la politique δ_n^*. A l'étape 8, on se base sur la proposition 5 pour trouver un chemin $\tilde{c}_1$ associé à la politique qui tend vers δ_c^*. Cela nous aidera par la suite à réduire le nombre des chemins dans $\mathcal{C}^*(\Phi_c, \Phi_n)$. Ensuite, de l'étape 11 à l'étape 19, nous construisons au fur à mesure

Algorithm 1: Heuristique de ré-optimisation

input : $\hat{X}$ petite perturbation;
m taille maximale des décisions élémentaires;
Φ_c, Φ_n paramètres qui définissent la zone de recherche;
l paramètre;
ϵ erreur;
output: $\hat{\mathcal{C}}^*(\Phi_c, \Phi_n)$: ensemble de chemins non-dominés;

1 $D \leftarrow$ `correctiveGeneratorDecisions(`$\hat{X}$`)` ;
2 $\mathcal{N}_1 \leftarrow \emptyset$;
3 **foreach** d *in* D **do**
4 $\quad \mathcal{N}_1 \leftarrow \mathcal{N}_1 \cup$ `findElementaryDecisions(`d,m`)` ;
5 **end**
6 $\mathcal{C}_2 \leftarrow$`paths(`$\mathcal{N}_1$`)`;
7 $\mathcal{C}_2 \leftarrow$`dominance(`$\mathcal{C}_2$`)` ;
8 $\tilde{c}_1 \leftarrow$`estimateDelta2(`$\mathcal{N}_1$`)` ;
9 $\hat{\mathcal{C}}^*(\Phi_c, \Phi_n) \leftarrow \mathcal{C}_2 \cup \{\tilde{c}_1\}$;
10 $i \leftarrow 1$;
11 **while** $\mathcal{N}_i \neq \emptyset$ **do**
12 $\quad$ **foreach** d *in* $\mathcal{N}_i$ **do**
13 $\quad\quad \delta \leftarrow$ `policy(`d`)`;
14 $\quad\quad \mathcal{N}_{i+1} \leftarrow \mathcal{N}_{i+1} \cup$ `propagate(`$\delta, \Phi_c, \Phi_n, m, l, \epsilon$`)`;
15 $\quad$ **end**
16 $\quad \mathcal{C} \leftarrow$`paths(`$\mathcal{N}_{i+1}$`)`;
17 $\quad \hat{\mathcal{C}}^*(\Phi_c, \Phi_n) \leftarrow$`dominance(`$\hat{\mathcal{C}}^*(\Phi_c, \Phi_n) \cup \mathcal{C}$`)`;
18 $\quad i \leftarrow i + 1$;
19 **end**

notre réseau en trouvant, à l'aide de la fonction `propagate`, les nœuds fils (de niveau $\mathcal{N}_{i+1}$) de chaque décision élémentaire d (de niveau $\mathcal{N}_i$). Ces nœuds représentent les décisions élémentaires susceptibles d'être prises après la `politique`(d). Lors de cette construction dynamique, nous cherchons les chemins non-dominés dans le graphe d'états construit jusqu'à présent.

L'algorithme 2 décrit la fonction `propagate`. Cette fonction commence par calculer une borne sur le coût des décisions élémentaires qu'on prendra en considération dans la construction de notre réseau. Cette borne vaut 0 si le nombre de décisions élémentaires qui forment la politique δ (=`politique`(d)) est supérieur à l, et vaut Φ_c sinon. A noter que l représente l'entier positif dont nous avons montré l'existence dans la proposition 4. Après l'application de la politique δ sur l'horaire x_0 (étape 8), on trouve à l'étape 9 un grand ensemble D de décisions élémentaires candidates. Chaque décision élémentaire d de D, passe par deux tests de sélection. A l'étape 11, le premier test consiste à voir si la nouvelle politique (δ, d) est dans l'espace de recherche et si le coût de la décision élémentaire d est bien inférieur à $\bar{c}$. Si d n'est pas éliminée par ce premier test, on calcule $\mu_\delta(d)$. Le deuxième test (étape 13) consiste à vérifier si $\mu_\delta(d) \in [\theta, 1]$, où θ est un paramètre qui représente le seuil d'acceptation d'une décision élémentaire. Ce seuil dépend principalement du reste à consommer de la ressource φ_n. En effet, plus le nombre de modifications est grand (proche de la borne Φ_n) plus le seuil θ tend vers 1. D'après la proposition 6, on peut déduire que, dans ce cas, la politique (δ, d) domine toute politique formée par des décisions élémentaires qui se trouvent dans les branches qui sont issues du chemin associé à la politique δ dans le graphe de la figure 2. Il est donc fort probable qu'une telle politique soit dans $\mathcal{P}^*(\Phi_c, \Phi_n)$. A noter que l'erreur ϵ (voir étape 12) permet de ne pas rejeter une décision élémentaire telle que $\varphi_n((\delta, d)) = \Phi_n$ et $1 - \epsilon < \mu_\delta(d) < 1$. L'erreur ϵ étant assez petit, la politique (δ, d) peut être dans $\mathcal{P}^*(\Phi_c, \Phi_n)$.

Dans notre heuristique, la construction du graphe et la recherche des chemins non-dominés se font en même temps. Le graphe résultant est peu dense. En effet, les nœuds du premier niveau $\mathcal{N}_1$ représentent les décisions élémentaires qui corrigent la perturbation $\hat{X}$, le nombre de ces décisions restant raisonnable. Ces nœuds produisent des nœuds fils dans le niveau suivant, ainsi de suite. Grâce à la fonction `propagate`, lorsqu'on passe d'un niveau à un autre, on devient plus sélectif dans nos choix. Ainsi toutes les décisions élémentaires des niveaux $\mathcal{N}_i$ tel que $l \leqslant i$ représentent des directions de descente. Tout cela a une influence positive sur le temps de calcul.

Algorithm 2: Fonction `propagate`

input : δ politique;
m taille maximale des décisions élémentaires;
Φ_c, Φ_n paramètres qui définissent la zone de recherche;
l paramètre;
ϵ erreur;
output: $D_{|\delta|+1}$: ensemble de décisions élémentaires;

$D_{i+1} \leftarrow \varnothing$;
if $|\delta| < l$ **then**
  $\bar{c} = \Phi_c$
end
else if $|\delta| \geqslant l$ **then**
  $\bar{c} = 0$
end
$x \leftarrow$ `applySequenceDecisions`(δ);
$D \leftarrow$ `candidateElementaryDecisions`(x, m);
foreach d *in* D **do**
  if $\varphi_c(d) < \bar{c}$ *and* $\varphi_n((\delta, d)) \leqslant \Phi_n$ **then**
    $\theta \leftarrow \frac{\varphi_n((\delta,d))}{\Phi_n} - \epsilon$;
    if $\mu_\delta(d) \in [\theta, 1]$ **then**
      $D_{|\delta|+1} \leftarrow D_{|\delta|+1} \cup \{d\}$;
    end
  end
end

4 Résultats numériques

Afin de tester l'efficacité de l'heuristique décrite dans l'algorithme 1, nous avons réalisé des tests expérimentaux sur un ensemble de scénarios de perturbation. Dans ces tests, nous avons adopté une méthode exacte comme une référence sur la qualité des solutions retournées par notre heuristique. Dans la première sous-section, on donne les instances utilisées, ainsi que la procédure utilisée pour générer les scénarios. Dans la deuxième sous-section, après avoir défini la méthode exacte, nous analysons les résultats obtenus.

Tous nos algorithmes ont été implémentés en C++. Par ailleurs, tous les tests ont été exécutés sur une machine Linux équipée d'un processeur Intel Core i7 8 cœurs cadencé à 3,4 GHz et d'une RAM de 16 Go.

4.1 Instances et scénarios de perturbation

Nous avons accès à 7 instances réelles, fournies par notre partenaire industriel. Les caractéristiques de ces instances sont données dans le tableau 1. A savoir que, pour optimiser ces instances, il faut dans un premier temps générer un ensemble de quarts proposés pour chaque employé durant chaque jour de l'horizon. Généralement une heuristique qui fait appel à la génération de colonnes est utilisée pour cela. Dans un second temps, il faut résoudre le programme linéaire en nombres entiers défini dans Hassani et al. [8]. Le solveur utilisé pour résoudre ce programme est CPLEX, version 12.6.1.0.

Table 1: Caractéristiques d'instances.

Instance	Nombre d'employés	Nombre de tâches	Horizon (en jours)
I_1	15	5	7
I_2	25	5	7
I_3	29	6	7
I_4	32	3	7
I_5	47	6	7
I_6	49	7	7
I_7	95	7	7

Pour chaque instance I_k, $k \in \{1, \ldots, 7\}$, on génère 30 scénarios. Chaque scénario est formé par une perturbation $\hat{X} = (\hat{e}, \hat{h}, \hat{s}, \hat{l}, \hat{p})$ qui perturbe l'horaire planifié obtenu pour l'instance I_k. Cette perturbation

est générée comme suit. Tout d'abord, le jour $\hat{h}$ est choisi en utilisant une distribution discrète uniforme sur l'horizon $\mathcal{H}$. Ensuite, on choisit l'employé $\hat{e}$ par une distribution discrète uniforme sur l'ensemble des employés qui travaillent le jour $\hat{h}$. La sélection du jour $\hat{h}$ et l'employé $\hat{e}$ conduit à la sélection du quart $\hat{s}$ (quart affecté à l'employé $\hat{e}$ durant le jour $\hat{h}$). La durée du retard $\hat{l}$, en nombre de périodes, est choisie en utilisant une autre distribution uniforme discrète sur l'ensemble $\{1, \ldots, l_{\hat{s}} - 1\}$. Enfin, la période $\hat{p}$, où l'employeur prend connaissance de la perturbation, est choisie par une distribution uniforme discrète sur l'ensemble $\{b_{\hat{s}} - 8, \ldots, b_{\hat{s}}\}$. Cela consiste à supposer que l'employeur prend connaissance de la perturbation au plus deux heures en avance. A noter qu'avant la validation de chaque scénario, une vérification est faite pour s'assurer que ce scénario ne se répète pas.

4.2 Résultats

La méthode exacte que nous avons adoptée afin d'évaluer l'efficacité de notre heuristique est basée sur le programme utilisé initialement pour trouver l'horaire initial, en ajoutant la contrainte suivante:

$$Hamming(x, x_0) \leqslant 2 \cdot \phi_n.$$

où la fonction $Hamming$ permet de calculer la distance du nouvel horaire x à l'horaire initial x_0. Lorsqu'on modifie le quart d'un employé, cette distance augmente avec 2 unités. $\phi_n \in [2, \Phi_n]$ représente le nombre total des quarts qu'on peut modifier. A noter que la planification de chaque période de l'horizon qui précède $\hat{p}$ est semblable dans les deux horaires x_0 et x. En bref, pour une perturbation donnée, la méthode exacte retourne la solution optimale qui corrige la perturbation avec la modification d'au plus ϕ_n quarts. On note par la suite $\Gamma^E(\phi_n)$ le coût de cette solution optimale, et par $\Gamma^H(\phi_n)$ le coût de la meilleure politique $\delta^* \in \underset{\{\delta \in \hat{\mathcal{P}}(\Phi_c, \Phi_n) | \varphi_n(\delta) \leqslant \phi_n\}}{\operatorname{argmax}} (\varphi_n(\delta))$, trouvée par notre heuristique, avec un nombre de modifications inférieur ou égal à ϕ_n. Dans l'industrie, les solutions retenues comporte peu de modifications car on ne veut pas modifier l'horaire d'un grand nombre d'employés juste parce que l'un d'eux est en retard. Nous avons donc choisi de fixer $\Phi_n = 6$. Ce choix nous permet de prédire qu'une décision élémentaire avec une taille importante ne sera probablement jamais présente dans les politiques de $\hat{\mathcal{P}}(\Phi_c, \Phi_n)$, car une telle décision impliquera généralement plus de quarts et consommera significativement la deuxième ressource φ_n. Pour cette raison, nous ne générons que les décisions élémentaires formées par une ou deux décisions génératrices ($m = 2$). Cela influencera positivement les temps de calcul. En revanche, cela peut théoriquement éliminer quelques solutions réalisables. Cette élimination va principalement concerner les solutions avec un petit nombre de modifications.

Pour comprendre cet effet, considérons l'exemple de la figure 4 qui représente une partie du polyèdre des solutions de $\mathcal{X}(\Phi_c, \Phi_n)$. On suppose qu'il existe une décision élémentaire formée par 4 décisions génératrices qui nous conduit de x à la solution x^*. La solution x^* ne sera donc plus atteignable directement de x pour $m = 2$. Mais on peut arriver à la solution x^* en passant dans un premier temps par la solution voisine y. Avec toute logique, plus le nombre de modifications autorisé est grand, plus nous avons de chance de trouver une séquence de décisions élémentaires qui peut conduire à une solution réalisable qui n'est plus atteignable directement à partir de la solution courante, à cause de l'élimination d'une séquence donnée. Par la suite, si notre heuristique n'arrive pas à trouver une solution réalisable avec moins de ϕ_n modifications, on dit que nous avons un échec.

Sur le tableau 2, on donne le pourcentage d'échecs que nous avons eu pour chaque instance sur l'ensemble des 30 scénarios. Ce pourcentage est donné, pour chaque valeur de $\phi_n \in [2, \Phi_n]$, par $\frac{n^E - n^H}{n^E} \cdot 100$ où n^E (resp. n^H) représente le nombre de scénarios résolubles avec un nombre de modifications inférieur ou égal à ϕ_n par la méthode exacte (resp. par l'heuristique). On peut voir que pour $\phi_n = 2$, nous avons obtenu moins de 4% d'échecs sur les 210 scénarios générés. Sinon pour $\phi_n > 2$, nous avons eu 0 échec (voir tableau 2). Ces résultats soutiennent ce que nous avons expliquer un peu plus haut.

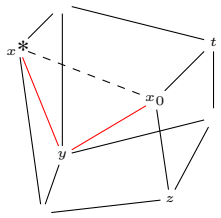

Figure 4: Partie du polyèdre de solutions.

Table 2: Pourcentage d'échecs.

Instance	$\phi_n = 2$			$\phi_n = 3$			$\phi_n = 4$			$\phi_n = 5$			$\phi_n = 6$		
	n^E	n^H	Echec(%)	n^E	n^H	Echec(%)	n^E	n^H	Echec(%)	n^E	n^H	Echec(%)	n^E	n^H	Echec(%)
I_1	30	30	0	30	30	0	30	30	0	30	30	0	30	30	0
I_2	30	30	0	30	30	0	30	30	0	30	30	0	30	30	0
I_3	30	30	0	30	30	0	30	30	0	30	30	0	30	30	0
I_4	30	30	0	30	30	0	30	30	0	30	30	0	30	30	0
I_5	30	29	3.33	30	30	0	30	30	0	30	30	0	30	30	0
I_6	30	25	16.66	30	30	0	30	30	0	30	30	0	30	30	0
I_7	30	28	6.66	30	30	0	30	30	0	30	30	0	30	30	0
Moyenne			3.81			0			0			0			0

Soit $\Gamma^+(\phi_n)$ l'erreur commise par l'heuristique dans la zone de recherche $\{\delta \in \hat{\mathcal{P}}(\Phi_c, \Phi_n) | \varphi_n(\delta) \leqslant \phi_n\}$, donnée par $\Gamma^H(\phi_n) - \Gamma^E(\phi_n)$. La moyenne de cette erreur sur les n_H scénarios générés pour l'instance I_k, où nous n'avons pas eu un échec, est notée par $\Gamma^+_{moy}(I_k, \phi_n)$. On note par $\Gamma^+_{Moy}(\phi_n)$ la moyenne totale des $\Gamma^+_{moy}(I_k, \phi_n)$ sur toutes les instances. Le tableau 3 indique que pour chaque valeur $\phi_n \in [2, \Phi_n]$, l'heuristique arrive à trouver la politique non dominée exacte dans le domaine $\{\delta \in \hat{\mathcal{P}}(\Phi_c, \Phi_n) | \varphi_n(\delta) \leqslant \phi_n\}$ sur plus que 96% des scénarios. A remarquer que Γ^+_{Moy} est plus importante dans la zone $\{\delta \in \hat{\mathcal{P}}(\Phi_c, \Phi_n) | \varphi_n(\delta) \leqslant 2\}$. Cela est attendu, car c'est la zone la plus affectée par l'élimination de certaines solutions réalisables. Cette erreur diminue relativement en passant de la zone $\{\delta \in \hat{\mathcal{P}}(\Phi_c, \Phi_n) | \varphi_n(\delta) \leqslant \phi_n\}$ à la zone $\{\delta \in \hat{\mathcal{P}}(\Phi_c, \Phi_n) | \varphi_n(\delta) \leqslant \phi_n + 1\}$.

Table 3: Pourcentage d'optimalité.

Instance	$\phi_n = 2$		$\phi_n = 3$		$\phi_n = 4$		$\phi_n = 5$		$\phi_n = 6$	
	$OPT(\%)$	Γ^+_{moy}	$OPT(\%)$	Γ^+_{moy}	$OPT(\%)$	Γ^+_{moy}	$OPT(\%)$	Γ^+_{moy}	$OPT(\%)$	Γ^+_{moy}
I_1	100	0	100	0	100	0	100	0	100	0
I_2	100	0	100	0	100	0	100	0	100	0
I_3	96.66	0.16	96.66	0.16	100	0	96.66	0.16	100	0
I_4	93.33	21.81	96.66	16.45	93.33	16.59	93.33	16.73	93.33	16.73
I_5	100	0	100	0	100	0	100	0	100	0
I_6	88	3.37	80	4.08	96	0.39	96	0.39	100	0
I_7	96.42	0.23	100	0	100	0	96.42	0.29	92.86	0.37
Γ^+_{Moy}		3.65		2.95		2.51		2.51		2.44
$OPT(\%)$ moyenne	96.34		96.19		98.47		97.49		98.03	

Très souvent, dans la pratique la correction est faite localement en modifiant l'horaire de deux employés afin de pallier le retard (méthode gloutonne utilisée dans Hassani et al. [8]). Pour cela, on propose une analyse au niveau de la réduction du coût en fonction de ϕ_n. Sur la figure 4, on voit qu'avec 2 modifications, le coût est strictement positif (car la solution x_0 est optimale). Cette correction, en soi, est une perturbation. Alors qu'en augmentant la borne ϕ_n, on voit qu'on peut réduire ce coût.

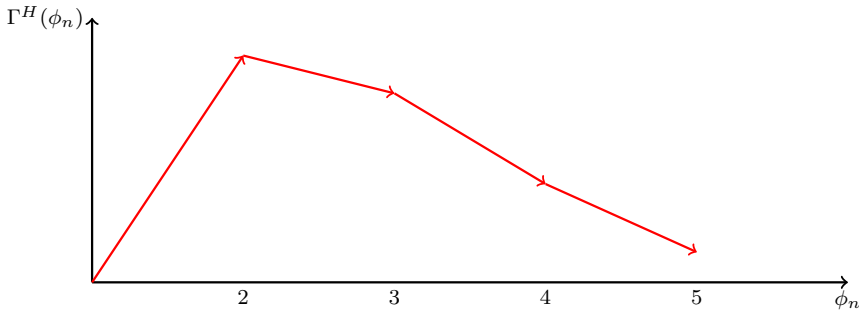

Table 4: Forme générale de $\Gamma^H(\phi_n)$.

On note par $\mathcal{G}(2,\phi_n)$ le pourcentage de coût sauvé de $\Gamma^H(2)$ avec ϕ_n modifications ($\mathcal{G}(2,2)=0$). Ce coût représente le gain apporté avec ϕ_n-2 modifications de plus. Soit $\mathcal{G}_{moy}(2,\phi_n)$ la moyenne de ce pourcentage sur l'ensemble des scénarios générés. Sur le tableau 5, on voit l'évaluation croissante de ce gain avec ϕ_n excepté pour les instances I_1 et I_2. Si on revient au tableau 3, on trouve que notre heuristique produit une solution optimale sur tous les scénarios générés pour ces deux instances. Donc, tout simplement le front de Pareto pour ces deux instances est formé par une seule politique δ^* telle que:

$$\{\delta^*\} = \underset{\delta\in\hat{\mathcal{P}}(\Phi_c,\Phi_n)}{\operatorname{argmin}}\ (\varphi_c(\delta)) = \underset{\delta\in\hat{\mathcal{P}}(\Phi_c,\Phi_n)}{\operatorname{argmin}}\ (\varphi_n(\delta)).$$

Table 5: Les gains $\mathcal{G}_{moy}(2,\phi_n)$.

Instance	$\mathcal{G}_{moy}(2,3)(\%)$	$\mathcal{G}_{moy}(2,4)(\%)$	$\mathcal{G}_{moy}(2,5)(\%)$	$\mathcal{G}_{moy}(2,6)(\%)$
I_1	0	0	0	0
I_2	0	0	0	0
I_3	0	87.86	87.86	89.34
I_4	19.56	35.25	35.25	36.57
I_5	0	90.40	90.40	90.40
I_6	74.41	89.41	99.11	100
I_7	47.41	61.94	64.64	65.08

Dans le tableau 6, on donne, pour chaque instance, le temps moyen de calcul T_{moy} sur les 30 scénarios pris par l'heuristique pour retourner toutes les politiques non-dominées $\hat{\mathcal{P}}^*(\Phi_c,\Phi_n)$. En moyenne, ce temps est de 0.51 seconde. Sur la deuxième colonne du même tableau, on voit que le temps d'exécution maximal T_{max} que nous avons eu est de 6.53 secondes. Cette durée reste très courte par rapport à celle prise par la méthode exacte (quelques minutes), pour former l'ensemble des politiques non-dominées. On voit sur la figure 5 le temps de calcul pour chaque scénario de chaque instance. On peut constater que le nombre de scénarios où le temps de calcul dépasse 2 secondes est de 7 sur les 210 scénarios générés. Donc dans 96.67% des cas, notre heuristique retourne l'ensemble des politiques non-dominées à l'employeur en moins de 2 secondes.

Table 6: Temps de calcul moyen.

Instance	$T_{moy}(s)$	$T_{max}(s)$
I_1	0.02	0.03
I_2	0.02	0.11
I_3	1.12	3.25
I_4	0.47	1.04
I_5	0.56	1.16
I_6	0.51	2.62
I_7	0.84	6.53
Moyenne	0.51	6.53

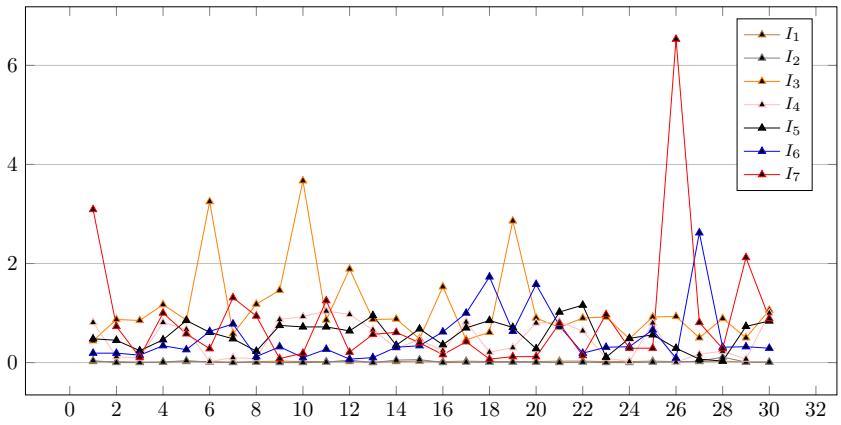

Figure 5: Temps de calcul pour chaque scénario.

5 Conclusion

Dans cet article, nous avons considéré le problème de ré-optimisation d'un horaire du personnel qui doit être résolu en temps réel après l'apparition d'une petite perturbation, dans un contexte où les employés peuvent être assignés à une grande variété de quarts, commençant et se terminant à divers moments. Pour résoudre ce problème, nous avons proposé une heuristique rapide qui vise à corriger la perturbation en proposant un ensemble de solutions qui réalisent un compromis entre le coût et le nombre de modifications. Cette heuristique est basée sur une étude théorique et une étude "probabiliste". Les tests numériques obtenus sur 210 scénarios de perturbation générés pour des instances réelles ont montré l'efficacité de cette heuristique. Celle-ci arrive à détecter les solutions exactes qui ne sont pas dominées, dans une zone de recherche fixée par l'employeur, en moins d'une seconde en moyenne pour plus de 96% de ces scénarios. Les tests ont aussi prouvé que le fait de traiter ce problème en faisant une optimisation multi-objectif, sur le reste de l'horizon en temps réel, peut réduire significativement le coût minimal de la correction locale classique utilisée actuellement, qui consiste uniquement à corrige la perturbation du jour où elle se produit.

Plusieurs pistes de recherche peuvent être poursuivies après ce travail, dont les deux suivantes. Premièrement, l'heuristique de ré-optimisation proposée ne considère que la perturbation actuelle sans anticiper des perturbations futures probables. Intégrer cela dans notre heuristique semble très utile et pourrait mener à la naissance d'une méthode stochastique de ré-optimisation du personnel efficace qui évalue "les coûts futurs attendus". Deuxièmement, on peut profiter du fait que notre heuristique n'utilise aucun solveur commercial pour l'adapter et proposer une heuristique qui trouve des horaires initiaux quasi-optimales en optimisant certains critères fixés par l'employeur (coût, préférence d'employés,...). Évidemment, la contrainte du temps sera relaxée et on doit adapter la mesure μ pour ces nouveaux critères.

References

[1] Bard, J. F., Purnomo, H. W., 2005. Hospital-wide reactive scheduling of nurses with preference considerations. IIE Transactions 37 (7), 589–608.

[2] Burke, E., De Causmaecker, P., Vanden Berghe, G., 2004. The state of the art of nurse rerostering. Journal of Scheduling 7 (6), 441–499.

[3] Dantzig, G. B., 1954. Letter to the editor-a comment on edie's "traffic delays at toll booths". Journal of the Operations Research Society of America 2 (3), 339–341.

[4] Edie, L. C., 1954. Traffic delays at toll booths. Journal of the Operations Research Society of America 2 (2), 107–138.

[5] Ernst, A. T., Jiang, H., Krishnamoorthy, M., Owens, B., Sier, D., 2004. An annotated bibliography of personnel scheduling and rostering. Annals of Operations Research 127 (1–4), 21–144.

[6] Froger, C., 2015. Mise à jour des horaires de personnel travaillant sur des quarts. Master's dissertation, Polytechnique Montréal. In French.

[7] Gross, C. N., Fügener, A., Brunner, J. O., 2016. Online rescheduling of physicians in hospitals. Flexible Services and Manufacturing Journal, p.1–33.

[8] Hassani, R., Guy, D., Issmail, E., 2017. Real-time personnel re-scheduling after a minor disruption. Technical report, HEC Montréal, Montréal. (G–2017–27).

[9] Kitada, M., Morizawa, K., 2013. A heuristic method for nurse rerostering problem with a sudden absence for several consecutive days. International Journal of Emerging Technology and Advanced Engineering 3 (11), 353–361.

[10] Kitada, M., Morizawa, K., Nagasawa, H., 2010. A heuristic method in nurse rerostering following a sudden absence of nurses. In: Proc. 11st Asia Pacific Industrial Engineering & Management Systems Conference. Vol. 6.

[11] Mahenhout, B., Vanhoucke, M., 2011. An evolutionary approach for the nurse rerostering problem. Computers & Operations Research 38 (10), 1400–1411.

[12] Moz, M., Pato, M. V., 2003. An integer multicommodity flow model applied to the rerostering of nurse schedules. Annals of Operations Research 119 (1–4), 285–301.

[13] Moz, M., Pato, M. V., 2004. Solving the problem of rerostering nurse schedules with hard constraints: New multicommodity flow models. Annals of Operations Research 128 (1–4), 179–197.

[14] Moz, M., Pato, M. V., 2007. A genetic algorithm approach to a nurse rerostering problem. Computers & Operations Research 34 (3), 667–691.

[15] Pato, M. V., Moz, M., 2008. Solving a bi-objective nurse rerostering problem by using a utopic pareto genetic heuristic. Journal of Heuristics 14 (4), 359–374.

[16] Van den Bergh, J., Beliën, J., De Bruecker, P., Demeulemeester, E., De Boeck, L., 2013. Personnel scheduling: A literature review. European Journal of Operational Research 226 (3), 367–385.