SUR LA PLANIFICATION DE TOURNÉES DE VÉHICULES SCOLAIRES

THÈSE N^{O} 3085 (2004)

PRÉSENTÉE À LA FACULTÉ SCIENCES DE BASE

Institut de mathématiques

SECTION DE MATHÉMATIQUES

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Michela SPADA

ingénieure mathématicienne diplômée EPF
de nationalité suisse et originaire d'Onex (GE)

acceptée sur proposition du jury:

Prof. Th. Liebling, directeur de thèse
Prof. D. de Werra, rapporteur
Prof. A. Hertz, rapporteur
Prof. M. Widmer, rapporteur
Dr Ph. Wieser, rapporteur

Lausanne, EPFL
2004

à mes parents,

à Eric

Je remercie le Professeur Thomas Liebling de m'avoir accueillie au sein de son groupe et de m'avoir guidée dans le choix du sujet ainsi que tout au long de cette thèse. Un grand merci au Docteur Michel Bierlaire qui a contribué à l'échafaudage de mon travail et qui a aussi officié en tant que président du jury lors de mon examen. Je remercie également les Professeurs Dominique de Werra, Alain Hertz et Marino Widmer, ainsi que le Docteur Philippe Wieser d'avoir accepté de faire partie de mon jury et d'avoir consacré de leurs précieux instants à la lecture de cette thèse.

Je suis reconnaissante à Alessandra et Eric qui ont patiemment relu ce document et contribué à sa clarté. J'envoie aussi un grand *tack* à Emma pour son aide lors de la traduction anglaise de la version abrégée.

J'ai partagé pendant plusieurs année mon bureau avec Frank, au pragmatisme inné, dont j'ai apprécié sa constante bonne humeur et sa collaboration. J'ai également eu beaucoup de plaisir à travailler avec Jean-François à la rigueur mathématique inégalée, avec Kim et ses traditionnels *piccoli caffè* matinaux et surtout avec Christine ma partenaire et co-championne du monde de coinche. Comment ne pas mentionner les autres membres du ROSO que j'ai côtoyés et qui ont contribué à la bonne ambiance au sein de notre groupe et pour certains d'entre eux à la génération d'évènements surprenants : Schumi, Krisprolls, Gogo, Papy, Marco, Gégé, Nicolas, Jaf, Tophe, Lionel, Alain, Mats, Sébastien et Stefano.

Je tiens à chaleureusement remercier mes parents, ma sœur et mes amis du soutien qu'ils m'ont apporté. Mes derniers remerciements, et non les moindres, vont à Eric, pour ses perpétuels encouragements.

Version abrégée

Un service de bus scolaires a pour objectif d'emmener des enfants de la maison à l'école et *vice versa*. L'organisation et la planification d'un tel service s'avèrent longues et fastidieuses si elles sont effectuées manuellement. Afin d'apporter une solution à ce problème combinatoire complexe, diverses heuristiques permettant de construire des planifications de tournées de véhicules scolaires de bonne qualité pour des problèmes de petite et de grande taille sont proposées dans ce travail.

Nous avons modélisé le problème de la planification de tournées de véhicules scolaires en introduisant une mesure de performance correspondant au niveau de service fourni aux enfants. Cette mesure de qualité centrée sur les enfants se démarque des précédents travaux sur le sujet, dans lesquels l'évaluation d'une solution dépend généralement du nombre de bus utilisés ou des longueurs des trajets effectués par les véhicules. Par ailleurs, nos planifications permettent à un véhicule de transporter simultanément des enfants devant se rendre en des écoles différentes, ce qui à notre connaissance n'a été envisagé qu'une seule fois dans la littérature.

Deux problèmes ont été considérés dans ce travail, l'un consistant à maximiser la qualité de service moyenne des enfants et l'autre à maximiser la qualité du plus mauvais niveau de service fourni. Ces problèmes se laissent exprimer sous la forme de programmes linéaires à variables binaires, mais leur taille est bien trop importante pour envisager une résolution exacte à l'aide des logiciels et machines actuels. Une méthode constructive est proposée pour l'obtention d'une planification admissible qui, par la suite, est optimisée à l'aide de diverses heuristiques : le recuit simulé, la recherche tabou et la recherche à voisinage variable. Notons qu'au cours de l'optimisation, la visite de planifications violant la capacité des véhicules est autorisée. Nos heuristiques, dans leur variante explorant le domaine des solutions non admissibles, s'avèrent plus efficaces que celles se limitant au domaine admissible. Ces méthodes nous ont également permis de produire, pour des instances réelles, des planifications de bien meilleure qualité que celles effectivement utilisées par les écoles en question.

Une planification idéale devrait permettre d'optimiser les deux fonctions objectif mentionnées ci-dessus. Afin de s'approcher d'une telle solution, une méthode à deux phases est proposée, la première consistant à maximiser le minimum des niveaux de service et la seconde à maximiser le niveau de service moyen sans réduire le minimum obtenu durant la première phase.

Par ailleurs, une extension de nos méthodes adaptée à la gestion des problèmes de grande taille a également été développée. L'approche par décomposition proposée permet de réduire considérablement les temps de calcul sans détériorer la qualité des planifications et se révèle indispensable pour la résolution de problèmes où le nombre d'enfants à transporter est important.

Notons finalement qu'une interface utilisateur a été développée. Elle permet de visualiser, d'analyser et de modifier les planifications générées par nos heuristiques. Il est ainsi possible d'intégrer des contraintes supplémentaires non modélisées ou d'adapter une solution en cas de modification légère des données du problème.

Abstract

The purpose of school bus services is to carry children from their homes to school and back. Scheduling and routing such services manually can be a long and tedious task, as it gives rise to complex combinatorial problems. To tackle those, we propose various heuristics in order to build good quality schedules of school bus routes for small and large size problems.

The school bus routing and scheduling problem is modeled by introducing a performance measure corresponding to the service level provided to the children. This quality measure focusing on the children differs from previous approaches to the subject, in which the solution evaluation generally depends on the number of buses used or the lengths of the routes of the vehicles. In addition, our schedules allow the vehicles simultaneously to carry children going to different schools, which to our knowledge has only been treated once in the literature.

Two problems are considered: one consisting in maximizing the average quality of children service level and another to maximize the quality of the worst service level provided to a child. These problems can be formulated as binary integer programs, but their size is too large to consider an exact resolution with current software and computers. A constructive method is proposed to obtain a feasible schedule which is optimized thereafter with various heuristics: simulated annealing, tabu search and variable neighborhood search. Let us note that, during optimization, the use of schedules violating the vehicle capacity is authorized. The heuristics employed, in their variant exploring infeasible solution set, are more effective than those that are confined to the feasible solution set. These methods also yield much higher quality schedules for real data sets than those actually used by the schools in question.

An ideal schedule should allow to simultaneously optimize both objective functions mentioned above. In order to approach such a solution, a two phase method is proposed. The first phase consists in maximizing the minimum service level and the second in maximizing the average service level without reducing the minimum obtained during the first phase.

We also developed an extension of our methods based on decomposition adapted to large scale problems. The proposed decomposition approach allows to considerably reduce computing time without giving up quality and is essential in order to solve problems with a large number of children.

Let us finally note that a graphical user interface was developed, allowing to visualize, analyze and modify schedules generated by the heuristics. It is thus possible to integrate additional constraints that have not been modeled, or to adapt a solution in case of a small modification to the problem data.

Table des matières

CHAPITRE 1

Introduction et revue de la littérature

1.1. Introduction au problème de la planification de tournées de véhicules scolaires

L'organisation du transport scolaire est courante dans des pays où de petites communes procèdent à une centralisation des infrastructures scolaires : c'est notamment le cas en Suisse, en Italie ou en France. Cette concentration a pour conséquence que certains écoliers doivent se rendre dans une école éloignée de leur domicile. Les localités concernées se voient dans l'obligation d'organiser elles-mêmes un réseau de transport privé et il leur incombe, chaque année, de planifier l'horaire et le trajet des bus prenant en charge les écoliers, c'est-à-dire les emmenant de la maison à l'école et *vice versa*. Selon le nombre d'enfants, le nombre d'écoles et le nombre de bus concernés, ce travail peut devenir extrêmement long et fastidieux s'il est fait manuellement, ceci pour un résultat de qualité souvent médiocre. C'est pourquoi l'usage de techniques de résolution automatisées ou d'aide à la décision s'avère particulièrement utile dans ce domaine. De tels outils permettent également d'étudier l'impact qu'aurait l'adjonction de bus sur la qualité du service fourni et sur les coûts supportés par les communes ou les conséquences qu'aurait une modification des regroupements scolaires (comme le déplacement d'une classe d'une école à une autre) ou de l'ensemble des écoles concernées sur la faisabilité de la planification.

À première vue, cette problématique s'apparente au «problème du voyageur de commerce» (*traveling salesman problem*), c'est-à-dire la construction d'un circuit hamiltonien de longueur minimale sur un ensemble de lieux donnés, *i.e.* visiter chaque lieu une et une seule fois. C'est un problème complexe d'optimisation combinatoire. Cependant, le problème de la planification de tournées de véhicules scolaires est bien plus difficile. En effet, supposons que l'on ait un petit nombre d'enfants, habitant tous en des lieux différents, à emmener en une unique école à l'aide d'un seul véhicule. Le trajet à construire doit vérifier les contraintes suivantes :

- ▷ il faut prendre en charge tous les enfants ;
- ▷ il faut d'abord passer prendre les enfants avant de les emmener à l'école ;
- ▷ il faut que les enfants arrivent à l'école à l'heure ;
- ▷ on ne peut transporter plus d'enfants qu'il n'y a de places dans le véhicule (il va donc peut-être falloir faire plusieurs trajets pour emmener tous les enfants à l'école).

Or, dans la planification de tournées de véhicules scolaires, on ne se limite pas à un seul bus. De même, le nombre d'écoles est généralement bien supérieur à un et leurs heures de début des cours ne sont pas toutes identiques. Comme de nombreux problèmes, la planification de tournées de véhicules scolaires peut être considérée de deux points de vue : celui des communes et celui des enfants. Les communes visent à générer une planification ne leur coûtant pas trop cher, tout en assurant une qualité de service acceptable. Les enfants quant à eux ne s'intéressent qu'à la qualité du service (ce service étant supposé gratuit), ce qui correspond au fait de ne pas avoir un trajet trop long et de ne pas avoir à attendre trop longtemps dans la cour d'école entre leur arrivée et l'heure de début des cours. Le nombre de bus à disposition étant généralement fixé, il est plus judicieux de considérer le problème du point de vue des enfants.

N'oublions pas que le problème de planification de tournées de véhicules scolaires consiste à emmener les enfants de la maison à l'école, mais également de les ramener chez eux à la fin des cours. Malheureusement, il est rarement possible de simplement parcourir en sens inverse la tournée qui emmène les enfants de la maison à l'école pour les ramener à la maison. En effet, les heures de fin de cours ne sont généralement pas échelonnées comme celles de début des cours. Il est donc vraiment important de générer des planifications efficaces, dans les deux sens, car parfois, lorsqu'aucune structure n'est mise en place pour accueillir les enfants pour le repas de midi, certains d'entre eux effectuent le trajet entre la maison et l'école ou *vice versa* quatre fois par jour (*i.e.* on les ramène à la maison à la mi-journée). Les techniques que nous avons développées permettent de déterminer des planifications de tournées de véhicules scolaires de bonne qualité en des temps relativement rapides.

Le problème étudié dans ce document n'est pas propre au milieu scolaire. On retrouve ce type de problématique dans d'autres domaines. En certaines régions, un service régulier de transports publics n'est pas rentable : peu de personnes l'utilisent du fait de sa basse fréquence et de la faible densité de population. Il peut alors s'avérer avantageux d'instaurer un système de «réservation de course» avec de petits véhicules pouvant transporter une dizaine de personnes. Les personnes utilisant ce service réservent une course en indiquant le trajet désiré et l'heure de départ ou d'arrivée. Un client peut, par exemple, demander à être emmené à la gare pour l'heure de départ du train qu'il doit prendre. Le planificateur doit faire en sorte d'effectuer le plus grand nombre de trajets tout en respectant les *desiderata* des clients. Pour satisfaire à un grand nombre de demandes, il convient de prendre en charge des courses différentes en même temps, c'est-à-dire transporter simultanément des personnes avec des origines et des destinations différentes, tout en ne violant pas la capacité des véhicules à disposition. On retrouve également le problème de la planification de tournées de véhicules scolaires chez certaines entreprises qui n'ont peu ou pas de places de parc ou qui sont soucieuses de l'environnement. Elles organisent alors des transports en commun pour leur employés.

Le problème de planification de tournées de véhicules scolaires s'apparente également à des problèmes de tournées de véhicules classiques, tels que les livraisons régulières de marchandises d'un fournisseur vers ses clients (transport de journaux, de médicaments, de pain, *etc.*). Certains problèmes ont également une composante de «récupération» d'une partie de ce qui a été livré, comme les verres vides consignés lors de la distribution de boissons. Les variantes sont très nombreuses et certaines d'entre elles n'ont pas encore été abordées mathématiquement.

1.2. Revue de la littérature

1.2.1. Variations sur le problème du voyageur de commerce

Le problème «originel» de la grande famille des problèmes de tournées de véhicules est le problème du voyageur de commerce évoqué pour la première fois par Menger en 1932 [Men32]. Ce problème a été très largement étudié (voir [LLKS85]), mais il reste néanmoins difficile. En effet, malgré les études approfondies de sa structure polyédrique, les méthodes exactes ne permettent de résoudre que des instances de taille modeste. De nombreuses heuristiques de résolution ont également été développées pour s'approcher des solutions optimales des problèmes de plus grande taille. Il existe de nombreuses variations sur le problème du voyageur de commerce, dont on propose ci-dessous une liste non exhaustive, ordonnée par ordre croissant de complexité.

Une première extension au problème du voyageur de commerce est celui «des» voyageurs de commerce, *i.e.* plusieurs véhicules sont à disposition pour visiter l'ensemble des clients. On le

nomme «problème de tournées de véhicules» (*vehicle routing problem*) (voir [Lap92]). Il a été mentionné pour la première fois par Dantzig et Ramser [DR59] en 1959.

Dans de nombreux contextes, il est important de tenir compte du fait que les véhicules utilisés ont des capacités finies. Lorsque l'on introduit cette contrainte, on parle alors de «problème de tournées de véhicules avec capacité» (*capacited vehicle routing problem*). De nombreuses études ont été effectuées sur ce problème, afin d'améliorer sa résolution par des méthodes exactes (voir [CFN85], [CH93], [NP98]). De même, des méthodes heuristiques de résolution ont été développées pour des instances de plus grande taille (voir [CW64], [BSL95], [GWKC98]).

Il est courant d'imposer que le voyageur de commerce visite chacun de ses clients dans un intervalle de temps donné : avant l'ouverture de son échoppe, par exemple, s'il vend des quotidiens. On nomme cette classe de problèmes, «problèmes de tournées de véhicules avec fenêtres de temps» (*vehicle routing problem with time windows*). Pullen et Webb [PW67], ainsi que Knight et Hofer [KH68] ont été les premiers à s'y intéresser.

Un autre paramètre dont on peut tenir compte est la livraison et le ramassage de marchandises chez les divers clients (*vehicle routing problem with pickup and delivery* [DDI+98]), telles que des bouteilles de lait, de boissons non alcoolisées ou de bière : on livre les bouteilles pleines et on récupère les vides (voir [GW87], [EMK94]).

Un problème proche du problème de la planification de tournées de véhicules scolaires est le problème de la «course à la demande» (*static dial-a-ride problem* ou *door-to-door problem*) (voir [JOPW86]). Il s'agit typiquement d'organiser un transport réalisé par des véhicules aux caractéristiques particulières, par exemple des véhicules pouvant accueillir des chaises roulantes (voir [BGKK99]). Ces véhicules fonctionnent comme une flotte de taxis, à la différence près qu'il faut réserver les courses à l'avance, *i.e.* indiquer l'origine et la destination du trajet ainsi que l'heure à laquelle il faut l'effectuer. Toutes les demandes étant connues à l'avance, il est possible de planifier les trajets des véhicules de manière à réaliser un maximum de courses (voir [CL03]) (notons qu'il existe également une version dynamique de ce problème). Le problème de la course à la demande se différencie du problème de la planification de tournées de véhicules scolaires par le fait que deux personnes partant de lieux différents, mais se rendant au même endroit sont généralement prises en charge séparément, la capacité des véhicules étant souvent très petite.

Remarquons que toute combinaison de ces variantes du problème du voyageur de commerce est envisageable. Les problèmes typiques de tournées de véhicules, ainsi que plusieurs méthodes permettant de s'approcher de leurs solutions optimales sont énumérées et décrites dans l'ouvrage de Toth et Vigo [TV02].

1.2.2. Le problème de la planification de tournées de véhicules scolaires

Une description générale du «problème de la planification de tournées de véhicules» (*vehicle routing and scheduling problem*) se trouve dans l'article de Bodin *et al.* [BGAB83]. Pour le cas particulier des véhicules scolaires, différentes approches sont proposées dans la littérature. Elles diffèrent au niveau de la manière de décomposer le problème, sur les hypothèses de modélisation et en matière d'algorithmes de résolution.

Décomposition du problème. Schématiquement, on distingue deux décompositions du problème, l'une basée sur les écoles et l'autre sur les enfants. L'approche basée sur les écoles consiste à résoudre un sous-problème pour chaque école. Ceci implique que l'*occupation mixte* des bus n'est pas autorisée, *i.e.* des enfants devant se rendre dans des écoles différentes ne sont pas autorisés à voyager en même temps dans le même bus. Cette approche est traitée par de

nombreux auteurs, Clarke et Wright [CW64], Newton et Thomas [NT69], Angel *et al.* [ACNW72], Bennett et Gazis [BG72], Rosenkrantz *et al.* [RSL74], Bodin et Berman [BB79], Gavish et Shlifer [GS79], Desrosiers *et al.* [DFR+81], Nygard *et al.* [NSW82], ainsi que Thangiah et Nygard [TN92]. L'approche basée sur les enfants consiste à ajouter successivement les enfants, dans une solution partielle du problème initial. Cette approche est plus flexible, car des occupations mixtes des bus sont autorisées. Elle a été proposée par Braca *et al.* [BBPSL97]. Dans la méthodologie présentée dans ce document, on construit une première solution selon l'approche basée sur les écoles, puis on suit Braca *et al.* [BBPSL97] en autorisant des occupations mixtes des bus au cours de l'optimisation de la solution initiale.

Modélisation du problème. La plupart des auteurs ont pour objectif de diminuer les coûts, ce qui revient à réduire le nombre de bus (voir [CW64], [BB79], [DFR+81], [SB84], [BBPSL97]) ou à minimiser une somme pondérée du nombre de bus et du temps total des trajets des bus (voir [GS79], [TN92]). Dans ce document, une fonction objectif diamétralement opposée a été choisie. On optimise explicitement le *niveau de service* fourni aux enfants. Plus précisément, on cherche à minimiser la perte de temps que subissent les enfants par rapport à une solution où chaque enfant aurait son propre véhicule à disposition. Une telle approche n'est pas courante dans la littérature, où le niveau de service apparaît souvent dans les contraintes. À notre connaissance, la seule référence où l'on considère la durée totale des trajets effectués par les enfants (*i.e.* la somme des temps passés dans les véhicules) dans la fonction objectif est un article de Bennett et Gazis [BG72]. Ces auteurs cherchent à minimiser la somme des temps de trajet parcourus par les bus et ceux effectués par les enfants. Bodin et Berman [BB79], Gavish et Shlifer [GS79], Desrosiers *et al.* [DFR+81], Braca *et al.* [BBPSL97], ainsi que Thangiah et Nygard [TN92] incluent une borne supérieure sur le temps de trajet de chaque enfant dans les contraintes du problème. Desrosiers *et al.* [DFR+81], Swersey et Ballard [SB84] et Braca *et al.* [BBPSL97] introduisent quant à eux une fenêtre de temps sur l'arrivée des enfants à l'école. Desrosiers *et al.* [DFR+81] ajoutent également une borne supérieure sur le temps d'attente entre l'arrivée à l'école et l'heure de début des cours. Braca *et al.* [BBPSL97] imposent une heure au plus tôt à laquelle un enfant peut être pris par un bus.

Algorithmes de résolution. Considérons dans un premier temps les approches basées sur les écoles. Le problème est décomposé en deux sous-problèmes : premièrement la génération de tournées pour chacune des écoles individuellement ; deuxièmement la planification horaire des véhicules qui effectuent plusieurs de ces tournées l'une après l'autre et visitent donc plus d'une école. Newton et Thomas [NT69] traitent un problème avec une seule école. Ils déterminent une solution du problème du voyageur de commerce visitant toutes les maisons et la découpent en tournées réalisables, chacune d'elles étant effectuée par un seul véhicule. Angel *et al.* [ACNW72] partitionnent les arrêts et génèrent des tournées sur chaque élément de la partition (*cluster first-route second*). Par la suite certains éléments de la partition sont fusionnés. Bennett et Gazis [BG72] s'intéressent à un problème ne comportant qu'une seule école. Ils utilisent la méthode proposée par Clarke et Wright [CW64], affectant à chaque maison une (ou plusieurs, selon le nombre d'enfants à transporter) tournée(s) directe(s) dépôt-maison-école. Ces tournées sont par la suite fusionnées de manière à respecter la capacité des véhicules et à minimiser le temps total de trajet des bus ou la somme pondérée du temps total de trajet des bus et le temps total que les enfants passent dans le bus. Pour terminer, ils tentent d'améliorer la solution par l'application de la procédure Lin-3-opt (voir [Lin65]). Bodin et Berman [BB79] utilisent l'approche «routage en premier-partition en second» (*route first-cluster second*), en s'inspirant de Newton et Thomas [NT69] : pour chaque école, ils commencent par déterminer une bonne solution au problème du voyageur de commerce visitant l'école et les maisons où habitent des enfants devant se rendre

en cette école à l'aide d'un Lin-3-opt (voir [Lin65]), puis partitionnent la solution en plusieurs tournées de manière à satisfaire les contraintes de capacité des véhicules et les contraintes de temps de trajet maximum des enfants. Afin de minimiser le temps de trajet total des bus, des tournées de *différentes écoles* sont par la suite concaténées. Desrosiers *et al.* [DFR+81] résolvent séquentiellement trois problèmes. Premièrement, pour chaque école, ils créent des tournées. Pour ce faire, ils utilisent des adaptations des méthodes de Newton et Thomas [NT69], Clarke et Wright [CW64] et la technique d'insertion proposée par Rosenkrantz *et al.* [RSL74] (*cluster first-route second*), suivies d'un Lin-2-opt. Deuxièmement, ils déterminent l'horaire des écoles (heures de début et de fin des cours), dans des fenêtres de temps données, à l'aide d'une méthode de génération de colonnes, de sorte qu'un véhicule puisse effectuer le plus grand nombre de tournées possible. Enfin les tournées sont concaténées de manière à en minimiser le nombre et l'horaire des bus est déterminé. Gavish et Shlifer [GS79] utilisent une approche différente, résolvant le problème pour chaque école à l'aide d'une approche d'énumération par séparation et évaluation dans laquelle une séquence de problèmes d'affectation sont résolus, tout en vérifiant que chaque enfant ne passe pas plus qu'un certain nombre de minutes dans le bus. Une fois des tournées admissibles générées, elles sont fusionnées afin de minimiser le nombre de bus et les coûts. La planification horaire des véhicules est formulée comme un problème d'affectation modifié. Swersey et Ballard [SB84] proposent diverses méthodes pour fusionner un ensemble donné de tournées de manière à minimiser le nombre de véhicules nécessaires, ainsi que pour en déterminer l'horaire. Ils expriment le problème comme un programme non linéaire à variables mixtes et l'approchent par un programme mathématique en nombres entiers. Ils étudient et modifient ce dernier pour obtenir une formulation pour laquelle une relaxation lagrangienne et linéaire fournit de bonnes solutions. Thangiah et Nygard [TN92] utilisent une approche «partition en premier-routage en second» (*cluster first-route second*) pour résoudre le problème avec une seule école. Ils déterminent la meilleure partition des maisons, au sens de la fonction objectif, par une méthode génétique et améliorent les tournées construites sur chaque élément de la partition par une méthode d'optimisation locale.

Braca *et al.* [BBPSL97], utilisant l'approche basée sur les enfants, résolvent le problème en son entier, c'est-à-dire qu'ils ne dissocient pas le problème de génération de tournées du problème de planification. Ils commencent par choisir aléatoirement un enfant et construisent une tournée initiale entre sa maison et son école. Puis, ils y insèrent itérativement un enfant et donc la paire (maison, école) associée de manière gloutonne : on insère en priorité les paires qui minimisent la longueur totale de la tournée et qui n'induisent pas de violation de contrainte. Une fois construite, la tournée n'est pas retouchée, sauf si elle ne permet de transporter qu'un petit nombre d'enfants. Lorsque plus aucune insertion de paire (maison, école) admissible n'est possible, une nouvelle tournée est alors créée.

CHAPITRE 2

Formulation du problème

Dans ce chapitre, les notations utilisées sont décrites et le problème de planification de tournées de véhicules scolaires est formulé comme un programme mathématique. On ne s'attarde que sur un sens du problème, à savoir celui consistant à emmener les enfants de la maison à l'école. La planification du trajet des enfants de l'école à la maison s'avère similaire au cas présenté et la description des quelques différences fait l'objet de la section 2.7.

2.1. Notations

Soit $G = (\mathcal{V}, \mathcal{E}, \ell)$ un réseau simple, où $\mathcal{V}$ désigne l'ensemble des sommets, $\mathcal{E}$ l'ensemble des arêtes et où ℓ est une fonction associant un coût non négatif à chaque arête (typiquement, le temps de trajet pour parcourir une arête). On note $d(i, j)$ le temps de parcours sur un plus court chemin dans G entre les sommets i et j. On note $\mathcal{H} = \{h_1, \dots, h_H\} \subseteq \mathcal{V}$ l'ensemble des sommets (appelés *maisons*) où les enfants montent dans le bus. Il s'agit en fait de l'arrêt de bus le plus proche du domicile de l'enfant. De manière similaire, on note $\mathcal{S} = \{s_1, \dots, s_S\} \subseteq \mathcal{V}$ l'ensemble des sommets où les *écoles* sont situées. Une classe de l'école s_i commence les leçons à l'heure o_{s_i}. Il est courant que différents niveaux scolaires, dont les cours commencent à des heures différentes, soient enseignés dans le même établissement scolaire. Dans ce cas, on attribue à l'établissement scolaire en question autant de sommets que de différentes heures de début des cours. Ces sommets ont les mêmes coordonnées géographiques et présentent un temps de parcours nul entre eux.

On note $N_{\mathcal{C}}$ le nombre total d'enfants devant se rendre d'une maison à une école. Dans ce modèle, on impose qu'un groupe d'enfants ayant les mêmes caractéristiques, *i.e.* la même maison et la même école fassent le même trajet, en même temps et dans le même bus. Ceci simplifie la formulation avec une perte de flexibilité modérée, dans le sens où cette situation est parfois requise pour des raisons de non-discrimination. Pour la clarté du document on appellera dorénavant un tel groupe d'enfants simplement un *enfant*. Soit $\mathcal{C}$ l'ensemble de ces (groupes d') enfants. On note $n(c)$ le nombre «réel» d'enfants correspondant aux (groupe d') enfants $c \in \mathcal{C}$. À chaque enfant $c \in \mathcal{C}$ correspond un sommet maison $h(c)$ et un sommet école $s(c)$. Dans un bus, chaque enfant occupe une certaine place, notée $q(c)$. Pour chaque école $s_i \in \mathcal{S}$, on note $\mathcal{C}(s_i) \subset \mathcal{C}$ l'ensemble $\{c \in \mathcal{C} \mid s(c) = s_i\}$ des enfants devant se rendre en s_i. On note $\mathcal{H}(s_i) \subset \mathcal{H}$ l'ensemble $\{h(c) \in \mathcal{H} \mid c \in \mathcal{C}(s_i)\}$, *i.e.* l'ensemble des maisons des enfants devant se rendre en l'école s_i.

Pour transporter les enfants, B bus sont à disposition. Un bus $b \in \mathcal{B} = \{1, \dots, B\}$ a une capacité Q_b (mesurée à l'aide de la même unité que $q(c)$, la place qu'occupe un enfant $c \in \mathcal{C}$) et est typiquement un nombre de sièges. L'occupation $q(c)$ peut être fractionnaire, car il existe des bus qui ne sont pas constitués de sièges individuels mais de bancs. Dans ce cas, les places ne sont donc pas bien définies et ces véhicules peuvent transporter plus de jeunes enfants que d'adolescents. Dans ce document, on fait l'hypothèse qu'il n'existe pas d'enfant $c \in \mathcal{C}$ tel que la place qu'il occupe $q(c)$ soit plus grande que le plus grand des véhicules à disposition, *i.e.* $\{c \in \mathcal{C} \mid q(c) > \max_{b \in \mathcal{B}} Q_b\} = \varnothing$.

Une solution $\mathcal{P}$ du problème de planification de tournées de véhicules scolaires consiste à spécifier pour chaque bus $b \in \mathcal{B}$, une tournée $\mathcal{T}_b = \left(a_\alpha^b\right)_{\alpha=1}^{n_b}$, constituée d'une liste de n_b arrêts notés a_α^b, avec $\alpha \in \{1, \dots, n_b\}$. Chaque arrêt a_α^b est un quadruplet

$$a_\alpha^b = (v_\alpha^b, t_\alpha^b, \overline{\mathcal{C}}_\alpha^b, \underline{\mathcal{C}}_\alpha^b), \tag{2.1}$$

où $v_\alpha^b \in \{1, \dots, |\mathcal{V}|\}$ désigne le numéro du sommet correspondant à l'arrêt, $t_\alpha^b \in \mathbb{R}$ est l'heure à laquelle le bus se trouve en cet arrêt, $\overline{\mathcal{C}}_\alpha^b \subset \mathcal{C}$ est l'ensemble des enfants pris par le bus b à l'arrêt α et $\underline{\mathcal{C}}_\alpha^b \subset \mathcal{C}$ est l'ensemble des enfants déposés par le bus à cet arrêt. En chaque arrêt, au moins un des deux ensembles $\overline{\mathcal{C}}_\alpha^b$ et $\underline{\mathcal{C}}_\alpha^b$ est non vide. Cependant, une planification doit vérifier certaines contraintes supplémentaires énoncées dans la section 2.3. On note Ω l'ensemble de toutes les planifications de véhicules admissibles, *i.e.* vérifiant les contraintes de la section 2.3.

Pour des raisons de sécurité, on admet que chaque enfant n'effectue son trajet qu'avec un seul bus, *i.e.* aucun changement de bus n'est autorisé. Le bus transportant l'enfant c dans la planification de véhicules $\mathcal{P}$ est noté $\beta^{\mathcal{P}}(c)$.

2.2. Mesures de performance

Idéalement, avec un bus par enfant, chaque enfant pourrait être conduit directement de sa maison à son école, pour l'heure de début des cours. C'est ce que l'on appelle la *solution taxi*. Comme le nombre de bus à disposition est limité, chaque enfant se voit imposer un temps de trajet supplémentaire, car le bus doit également faire monter d'autres enfants sur son chemin en direction de l'école. Pour chaque enfant, on définit le *supplément de temps de trajet* comme la différence entre le temps de trajet qu'il effectue dans le bus et la durée du plus court chemin entre sa maison et son école. De plus, comme les bus doivent effectuer plusieurs tournées, certains enfants arriveront à l'école trop tôt, c'est-à-dire avant l'heure de début des cours. Le temps que l'enfant passe à l'école en attendant que les cours ne commencent est appelé le *temps d'attente*. On appelle *perte de temps* d'un enfant la somme du supplément de temps de trajet et du temps d'attente. Plus loin, on associe deux mesures de performance à une planification $\mathcal{P}$, toutes deux fonction de la perte de temps des enfants.

Pour l'arrêt a_α^b et l'enfant c, on définit

$$\overline{\delta}_{c\alpha}^b = \begin{cases} 1 & \text{si } c \in \overline{\mathcal{C}}_\alpha^b \textit{ i.e. } \text{si le bus } b \text{ prend l'enfant } c \text{ à l'arrêt } a_\alpha^b \\ 0 & \text{sinon,} \end{cases} \tag{2.2}$$

et

$$\underline{\delta}_{c\alpha}^b = \begin{cases} 1 & \text{si } c \in \underline{\mathcal{C}}_\alpha^b \textit{ i.e. } \text{si le bus } b \text{ dépose l'enfant } c \text{ à l'arrêt } a_\alpha^b \\ 0 & \text{sinon.} \end{cases} \tag{2.3}$$

L'heure de départ σ_c de l'enfant c est donnée par

$$\sigma_c = \sum_{b=1}^{B} \sum_{\alpha=1}^{n_b} \overline{\delta}_{c\alpha}^b t_\alpha^b. \tag{2.4}$$

On note qu'un seul terme de cette somme est non nul, car aucun des enfants n'effectue de changements de bus au cours de son trajet. De manière similaire, l'heure d'arrivée τ_c de l'enfant c est définie par

$$\tau_c = \sum_{b=1}^{B} \sum_{\alpha=1}^{n_b} \underline{\delta}_{c\alpha}^b t_\alpha^b. \tag{2.5}$$

Par conséquent, la perte de temps $f(c)$ de l'enfant c est donnée par

$$f(c) = \left(o_{s(c)} - \sigma_c\right) - d\left(h(c), s(c)\right), \tag{2.6}$$

où $o_{s(c)}$ est l'heure de début des cours de l'école $s(c)$. L'équation (2.6) est la différence entre l'heure de début des cours et l'heure de départ de l'enfant c, à laquelle on soustrait le temps du plus court trajet possible entre sa maison et son école. Nous sommes maintenant en mesure d'exprimer la première fonction objectif $\mathcal{F}_1$, la *somme totale* des pertes de temps des enfants :

$$\mathcal{F}_1 = \sum_{c \in \mathcal{C}} f(c) n(c). \tag{2.7}$$

La seconde fonction objectif que l'on définit est $\mathcal{F}_2$, le *maximum* des pertes de temps de tous les enfants :

$$\mathcal{F}_2 = \max_{c \in \mathcal{C}} f(c). \tag{2.8}$$

On cherche à minimiser l'une ou l'autre de ces fonctions objectif sur toutes les planifications Ω admissibles. Minimiser $\mathcal{F}_1$, la somme totale des pertes de temps, revient à minimiser la moyenne des pertes de temps des enfants. Minimiser $\mathcal{F}_2$, le maximum des pertes de temps, tend à répartir équitablement les pertes de temps entre les enfants. La fonction objectif $\mathcal{F}_1$ permet d'identifier les enfants pénalisant le problème à résoudre et donc entravant la réalisation d'une planification de bonne qualité. On peut, peut-être, envisager pour eux un autre moyen de transport. Dans une planification optimisée, ces enfants ont une grande perte de temps. La fonction objectif $\mathcal{F}_2$ est plus équitable entre les enfants et est donc généralement plus appréciée.

2.3. Contraintes

Les contraintes suivantes doivent être vérifiées. Premièrement chaque enfant doit arriver à l'école à l'heure, c'est-à-dire

$$\tau_c \leq o_{s(c)}, \quad \forall c \in \mathcal{C}. \tag{2.9}$$

Deuxièmement, on impose que chaque enfant soit transporté par un seul bus. Il faut donc que les enfants ne montent dans un bus $b \in \mathcal{B}$ qu'à leur maison et n'en descendent qu'à leur école, ce que l'on peut exprimer comme suit :

$$v^b_\alpha = h^v(c), \quad \forall c \in \overline{\mathcal{C}}^b_\alpha, \alpha \in \{1, \dots, n_b\}, b \in \mathcal{B} \tag{2.10}$$

$$v^b_\alpha = s^v(c), \quad \forall c \in \underline{\mathcal{C}}^b_\alpha, \alpha \in \{1, \dots, n_b\}, b \in \mathcal{B} \tag{2.11}$$

où $h^v(c) \in \{1, \dots, |\mathcal{V}|\}$ désigne le numéro du sommet $h(c) \in \mathcal{V}$ et $s^v(c) \in \{1, \dots, |\mathcal{V}|\}$ désigne le numéro du sommet $s(c) \in \mathcal{V}$.
Il faut également que chaque enfant $c \in \mathcal{C}$ ne soit transporté qu'une seule fois, c'est-à-dire que

$$\sum_{b=1}^{B} \sum_{\alpha=1}^{n_b} \overline{\delta}^b_{c\alpha} = 1, \quad \forall c \in \mathcal{C}, \tag{2.12}$$

$$\sum_{b=1}^{B} \sum_{\alpha=1}^{n_b} \underline{\delta}^b_{c\alpha} = 1, \quad \forall c \in \mathcal{C} \tag{2.13}$$

et qu'il descende du bus dans lequel il est monté, c'est-à-dire

$$\sum_{\alpha=1}^{n_b} \left(\overline{\delta}^b_{c\alpha} - \underline{\delta}^b_{c\alpha}\right) = 0 \quad \forall b \in \mathcal{B}, c \in \mathcal{C}. \tag{2.14}$$

De plus, l'heure de départ de la maison doit précéder l'heure d'arrivée à l'école :

$$\sigma_c \leq \tau_c, \quad \forall c \in \mathcal{C}. \tag{2.15}$$

Les bus ne peuvent transporter plus d'enfants que leur capacité ne le leur permet. En chaque arrêt a_α^b de chaque bus $b \in \mathcal{B}$, l'inégalité suivante doit donc être vérifiée :

$$\sum_{k=1}^{\alpha} \sum_{c=1}^{|\mathcal{C}|} q(c) \left(\overline{\delta}_{ck}^b - \underline{\delta}_{ck}^b \right) \leq Q_b, \quad \forall \alpha \in \{1, \dots, n_b\}, b \in \mathcal{B}. \tag{2.16}$$

Pour chaque tournée, l'horaire doit être tel que le bus puisse se déplacer tout en respectant les temps de trajet minimum donnés, c'est-à-dire

$$t_{\alpha+1}^b - t_\alpha^b \geq d(v_\alpha^b, v_{\alpha+1}^b), \quad \forall \alpha \in \{1, \dots, n_b\}, b \in \mathcal{B}. \tag{2.17}$$

2.4. Programme mathématique

En rassemblant les fonctions objectif et contraintes énoncées ci-dessus, on obtient le programme mathématique suivant :

$$\begin{array}{llcll}
\min & z & = & \mathcal{F}(\mathcal{P}) & & \\
\text{s.c.} & \displaystyle\sum_{b=1}^{B} \sum_{\alpha=1}^{n_b} \overline{\delta}_{c\alpha}^b & = & 1, & \forall c \in \mathcal{C} & \\
& \displaystyle\sum_{b=1}^{B} \sum_{\alpha=1}^{n_b} \underline{\delta}_{c\alpha}^b & = & 1, & \forall c \in \mathcal{C} & \\
& \displaystyle\sum_{\alpha=1}^{n_b} \left(\overline{\delta}_{c\alpha}^b - \underline{\delta}_{c\alpha}^b \right) & = & 0, & \forall b \in \mathcal{B}, c \in \mathcal{C} & \\
& \displaystyle\sum_{b=1}^{B} \sum_{\alpha=1}^{n_b} \overline{\delta}_{c\alpha}^b t_\alpha^b & = & \sigma_c, & \forall c \in \mathcal{C} & (2.4) \\
& \displaystyle\sum_{b=1}^{B} \sum_{\alpha=1}^{n_b} \underline{\delta}_{c\alpha}^b t_\alpha^b & = & \tau_c, & \forall c \in \mathcal{C} & (2.5) \\
& \displaystyle\sum_{k=1}^{\alpha} \sum_{c=1}^{|\mathcal{C}|} q(c) \left(\overline{\delta}_{ck}^b - \underline{\delta}_{ck}^b \right) & \leq & Q_b, & \forall \alpha \in \{1, \dots, n_b\}, b \in \mathcal{B} & \\
& \tau_c & \leq & o_{s(c)}, & \forall c \in \mathcal{C} & \\
& \sigma_c & \leq & \tau_c, & \forall c \in \mathcal{C} & \\
& t_{\alpha+1}^b - t_\alpha^b & \geq & d(v_\alpha^b, v_{\alpha+1}^b), & \forall \alpha \in \{1, \dots, n_b\}, b \in \mathcal{B} & \\
& v_\alpha^b & = & h^v(c), & \forall c \in \overline{\mathcal{C}}_\alpha^b, \alpha \in \{1, \dots, n_b\}, b \in \mathcal{B} & (2.10) \\
& v_\alpha^b & = & s^v(c), & \forall c \in \underline{\mathcal{C}}_\alpha^b, \alpha \in \{1, \dots, n_b\}, b \in \mathcal{B} & (2.11) \\
& \overline{\delta}_{c\alpha}^b, \ \underline{\delta}_{c\alpha}^b & \in & \{0, 1\}, & \forall c \in \mathcal{C}, \alpha \in \{1, \dots, n_b\}, b \in \mathcal{B} & \\
& t_\alpha^b & \in & \mathbb{R}, & \forall \alpha \in \{1, \dots, n_b\}, b \in \mathcal{B} & \\
& v_\alpha^b & \in & \{1, \dots, |\mathcal{V}|\}, & \forall \alpha \in \{1, \dots, n_b\}, b \in \mathcal{B} &
\end{array}$$

où $\mathcal{F} \in \{\mathcal{F}_1, \mathcal{F}_2\}$ avec

- ▷ $\mathcal{F}_1 = \sum_{c \in \mathcal{C}} f(c) n(c)$ et $\mathcal{F}_2 = \max_{c \in \mathcal{C}} f(c)$;
- ▷ $\mathcal{P} = \{\mathcal{T}_b\}_{b=1}^B = \{\left(a_\alpha^b\right)_{\alpha=1}^{n_b}\}_{b=1}^B = \{(v_\alpha^b, t_\alpha^b, \overline{\mathcal{C}}_\alpha^b, \underline{\mathcal{C}}_\alpha^b)_{\alpha=1}^{n_b}\}_{b=1}^B$,
  $\overline{\mathcal{C}}_\alpha^b = \{c \in \mathcal{C} \mid \overline{\delta}_{c\alpha}^b = 1\}$ et $\underline{\mathcal{C}}_\alpha^b = \{c \in \mathcal{C} \mid \underline{\delta}_{c\alpha}^b = 1\}$.

2.5. Linéarisation

Le programme mathématique défini ci-dessus est un programme non linéaire à variables mixtes. La non-linéarité provient des contraintes (2.4) et (2.5), des contraintes (2.10) et (2.11), ainsi que de la fonction objectif $\mathcal{F}_2$. Ce programme peut être transformé en un programme linéaire en nombres entiers à l'aide de techniques standard, au prix de l'introduction d'un nombre non négligeable de variables et de contraintes.

Pour la fonction objectif $\mathcal{F}_2$, on obtient l'expression suivante :

$$\min_{\mathcal{P}\in\Omega} \quad z = \max_{c\in\mathcal{C}} f(c).$$

Pour linéariser cette expression, il suffit d'introduire une nouvelle variable m réelle, que l'on posera supérieure à tous les $f(c)$ (avec $c \in \mathcal{C}$) et que l'on cherchera à minimiser.

$$\min_{\mathcal{P}\in\Omega} \quad z = \max_{c\in\mathcal{C}} f(c) \quad \Longleftrightarrow \quad \begin{array}{lll} \min\limits_{\mathcal{P}\in\Omega} & z = m & \\ \text{s.c.} & m \geq f(c), & \forall c \in \mathcal{C} \\ & m \in \mathbb{R} & \end{array} \tag{2.18}$$

Cette linéarisation fait donc apparaître une variable réelle et $|\mathcal{C}|$ contraintes supplémentaires.

Intéressons-nous maintenant aux contraintes (2.4) et (2.5). Pour des variables binaires x, y et z, on a l'équivalence

$$xy = z \quad \Longleftrightarrow \quad \begin{cases} x + y - 1 \leq z \\ y \geq z \\ x \geq z \end{cases} \tag{2.19}$$

Cependant, les contraintes (2.4) et (2.5) ne sont pas du type présenté ci-dessus, mais il est possible de s'y ramener. En effet, elles sont de la forme

$$\sigma = \delta t, \tag{2.20}$$

où $\delta \in \{0, 1\}$ et la variable $t \in \mathbb{R}$ représente une heure. Si l'on discrétise la variable t à la minute, ce qui est réaliste dans le cadre d'un horaire de bus, on a alors que $t \in \{0, \dots, T\}$, où l'intervalle $[0, T]$ représente la plage horaire sur laquelle on désire construire la planification. Ainsi

$$\begin{aligned} \sigma &= \delta t \\ &= \delta \sum_{i=0}^{\lfloor \log_2(T) \rfloor} 2^i t_i, \qquad \text{avec } t_i \in \{0, 1\} \\ &= \sum_{i=0}^{\lfloor \log_2(T) \rfloor} 2^i z_i, \qquad \text{où } z_i = \delta t_i, \forall i \in \{0, \dots, \lfloor \log_2(T) \rfloor\} \end{aligned}$$

Les variables δ, z_i et t_i ($i \in \{0, \dots, \lfloor \log_2(T) \rfloor\}$) étant binaires, il est possible de linéariser chaque équation $z_i = \delta t_i$ en utilisant l'équivalence (2.19). Ainsi, chaque contrainte de type (2.20), engendre $\lfloor \log_2(T) \rfloor + 1$ variables z_i binaires et $3(\lfloor \log_2(T) \rfloor + 1)$ contraintes supplémentaires. Watters [Wat67] est l'auteur de cette transformation.

Pour un enfant c donné, une contrainte (2.4) est formée d'une somme de $B \cdot n_b$ produits de type δt. Il en va de même pour la contrainte (2.5). Ainsi l'ensemble des contraintes (2.4) et (2.5) peuvent être linéarisées en les remplaçant par $2(|\mathcal{C}| \cdot B \cdot n_b(\lfloor \log_2(T) \rfloor + 1))$ variables binaires et $2 \cdot 3(|\mathcal{C}| \cdot B \cdot n_b(\lfloor \log_2(T) \rfloor + 1))$ contraintes linéaires.

Occupons-nous maintenant des contraintes (2.10) et (2.11). Pour des variables binaires x et y_i, avec $i \in \{1, \dots, N\}$, on a l'équivalence

$$\sum_{i=1}^{N} x y_i = 0 \quad \Longleftrightarrow \quad x + y_i \leq 1, \quad \forall i \in \{1, \dots, N\}. \tag{2.21}$$

Il est possible de ramener les contraintes (2.10) et (2.11) à la forme ci-dessus. En effet, pour tout $b \in \mathcal{B}$ et pour tout $\alpha \in \{1, \dots, n_b\}$ on a

$$v_\alpha^b = h^v(c), \quad \forall c \in \overline{\mathcal{C}}_\alpha^b, \quad \Longleftrightarrow \quad \overline{\delta}_{c\alpha}^b (v_\alpha^b - h^v(c)) = 0, \quad \forall c \in \mathcal{C}, \tag{2.22}$$

car $\overline{\mathcal{C}}_\alpha^b = \{c \in \mathcal{C} \mid \overline{\delta}_{c\alpha}^b = 1\}$.

Les variables $\overline{\delta}_{c\alpha}^b$ sont binaires, alors que l'expression $v_\alpha^b - h^v(c)$ prend des valeurs dans l'ensemble $\{1 - h^v(c), \dots, |\mathcal{V}| - h^v(c)\}$, avec $h^v(c) \in \{1, \dots, |\mathcal{V}|\}$. Ainsi la contrainte à linéariser est de la forme

(2.23)

$$\begin{aligned} 0 &= \delta w, && \text{avec } \delta \in \{0, \dots, 1\}, w \in \{1 - h^v(c), \dots, |\mathcal{V}| - h^v(c)\} \\ &= \delta \sum_{i=0}^{\lfloor \log_2(|\mathcal{V}|-1) \rfloor} 2^i (w_i^+ - w_i^-), && \text{avec } w_i^+, w_i^- \in \{0, 1\} \\ &= \sum_{i=0}^{\lfloor \log_2(|\mathcal{V}|-1) \rfloor} (2^i \delta w_i^+ - 2^i \delta w_i^-), && \text{avec } w_i^+, w_i^- \in \{0, 1\}. \end{aligned}$$

Les variables δ, w_i^+ et w_i^- (avec $i \in \{0, \dots, \lfloor \log_2(|\mathcal{V}| - 1) \rfloor\}$) étant binaires, il est possible de linéariser l'équation (2.23) et par conséquent la contrainte (2.22) en utilisant l'équivalence (2.21). Néanmoins, sachant que w prend des valeurs dans l'ensemble $\{1 - h^v(c), \dots, |\mathcal{V}| - h^v(c)\}$, il est nécessaire d'ajouter deux contraintes pour restreindre les valeurs que peuvent prendre les variables w_i^+ et w_i^- :

$$\sum_{i=0}^{\lfloor \log_2(|\mathcal{V}|-1) \rfloor} 2^i (w_i^+ - w_i^-) \leq |\mathcal{V}| - h^v(c) \tag{2.24}$$

et

$$\sum_{i=0}^{\lfloor \log_2(|\mathcal{V}|-1) \rfloor} 2^i (w_i^+ - w_i^-) \geq 1 - h^v(c). \tag{2.25}$$

On en conclut que chaque contrainte (2.22) peut être remplacée par $2(\lfloor \log_2(|\mathcal{V}|-1) \rfloor + 1)$ variables binaires et $2(\lfloor \log_2(|\mathcal{V}| - 1) \rfloor + 1) + 2$ contraintes supplémentaires. Comme il y a $B \cdot n_b \cdot |\mathcal{C}|$ contraintes (2.10) et autant de contraintes (2.11), elles peuvent être linéarisées en les remplaçant par $2(B \cdot n_b \cdot |\mathcal{C}|(2\lfloor \log_2(|\mathcal{V}| - 1) \rfloor + 2))$ variables binaires et $2(B \cdot n_b \cdot |\mathcal{C}|(2\lfloor \log_2(|\mathcal{V}| - 1) \rfloor + 4))$ contraintes.

On voit donc qu'il est possible d'exprimer notre problème comme un programme linéaire à variables binaires. Malheureusement, en termes de variables et de contraintes, la taille de cette formulation devient bien trop grande pour une résolution exacte à l'aide des logiciels et ordinateurs actuels. Par exemple, pour le problème réel des communes de Savigny et Forel décrit dans la section 4.1.1 (34 sommets, 90 enfants et 4 bus), le programme non linéaire (respectivement linéaire) en nombres entiers est constitué de 21 840 (respectivement 453 600) variables et 22 560 (respectivement 821 460) contraintes. Le problème de la planification de tournées de véhicules scolaires étant une extension du problème du voyageur de commerce, qui est NP-complet (voir

[GJ79]), nous avons choisi de développer et de comparer diverses heuristiques de recherche locale, comme le recuit simulé, la recherche tabou (voir chapitres 3 et 5) et la recherche à voisinage variable (voir section 5.4).

2.6. Autre modélisation

La modélisation proposée n'a pas pour objectif de construire des tournées de bus au sens usuel du terme, c'est-à-dire partir d'un dépôt, ne visiter les sommets qu'une et une seule fois et terminer sa course au dépôt, tout en respectant les capacités des véhicules, ainsi que les contraintes relatives aux précédences et aux heures de début des cours. En effet, dans ce document, on construit des chemins ouverts (voir [CPR94]) débutant en une maison et se terminant en une école. De plus on autorise un véhicule à visiter plusieurs fois le même sommet. Néanmoins, il est possible de transformer ce problème de manière à le formuler comme un problème classique de tournées de véhicules.

La transformation consiste à attribuer à chaque enfant $c \in \mathcal{C}$ un sommet maison propre ainsi qu'un sommet école propre. Plus précisément, si des enfants (se rendant dans des écoles différentes) prennent le bus au même arrêt, le sommet maison est dupliqué pour en avoir un par enfant (voir figure 2.1). On a le cas symétrique de duplication des sommets école lorsque des enfants prennent le bus en des arrêts différents mais se rendent à la même école.

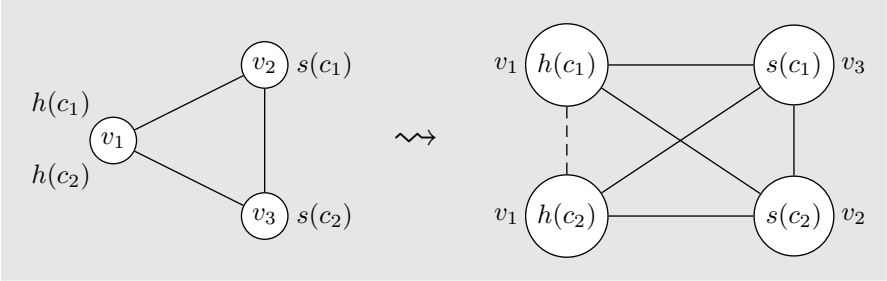


FIG. 2.1 – Chaque enfant $c \in \mathcal{C}$ a un sommet maison propre ainsi qu'un sommet école propre.

On obtient un nouveau réseau $G' = (\mathcal{V}', \mathcal{E}', \ell')$, où $\mathcal{V}'$ est l'ensemble des sommets, $\mathcal{E}'$ l'ensemble des arêtes et ℓ' une fonction associant un coût non négatif à chaque arête. L'ensemble des sommets est la réunion de trois ensembles disjoints : $\mathcal{V}' = \mathcal{V}'_0 \cup \mathcal{V}'_{\mathcal{H}} \cup \mathcal{V}'_{\mathcal{S}}$. L'ensemble $\mathcal{V}'_0 = \{v'_0\}$ ne contient que le dépôt v'_0 où sont situés tous les véhicules. L'ensemble $\mathcal{V}'_{\mathcal{H}} = \{v'_1, \dots, v'_{|\mathcal{C}|}\}$ contient les sommets maisons de chacun des $|\mathcal{C}|$ enfants. L'ensemble $\mathcal{V}'_{\mathcal{S}} = \{v'_{|\mathcal{C}|+1}, \dots, v'_{2|\mathcal{C}|}\}$ contient les sommets écoles de chacun des $|\mathcal{C}|$ enfants. Pour permettre la construction de tournées transportant en même temps des enfants avec des origines et destinations diverses, le réseau G' doit être complet. Le poids de chaque arête $[v'_i, v'_j] \in \mathcal{E}'$ correspond au temps de trajet sur un plus court chemin dans G pour se rendre de v'_i à v'_j, pour tout $v'_i \neq v'_j$, avec $v'_i, v'_j \in \mathcal{V}' \setminus \{v'_0\}$. Le poids de chaque arête $[v'_0, v']$, pour tout $v' \in \mathcal{V} \setminus \{v'_0\}$, vaut 0.

Nous illustrons, dans la figure 2.2, la transformation d'un réseau initial G en un nouveau réseau G', ainsi qu'une planification $\mathcal{P}$ dans G et G'. Supposons que l'on doive transporter 3 enfants, $\mathcal{C} = \{c_1, c_2, c_3\}$ avec les caractéristiques suivantes :

$$\begin{array}{lll} h(c_1) = v_1, & s(c_1) = v_4, & q(c_1) = 1, \\ h(c_2) = v_1, & s(c_2) = v_3, & q(c_2) = 1, \\ h(c_3) = v_2, & s(c_3) = v_4, & q(c_3) = 1, \\ o_{s(c_1)} = 9, & o_{s(c_2)} = 9, & o_{s(c_3)} = 9. \end{array}$$

Pour transporter ces enfants, un seul bus est à disposition ($B = 1$) de capacité $Q_1 = 3$.
On donne $\mathcal{P}$ une planification admissible du problème ci-dessus, représentée dans la figure 2.2 :

$$\mathcal{P} = \{\mathcal{T}_1\} = \{(a_\alpha)_{\alpha=1}^4\} = \{(v_\alpha, t_\alpha, \overline{\mathcal{C}}_\alpha, \underline{\mathcal{C}}_\alpha)_{\alpha=1}^4\}$$

$$\begin{aligned} a_1 &= (v_1, 1, \{c_1, c_2\}, \varnothing\} \\ a_2 &= (v_2, 4, \{c_3\}, \varnothing\} \\ a_3 &= (v_3, 7, \varnothing, \{c_2\}\} \\ a_4 &= (v_4, 9, \varnothing, \{c_1, c_3\}\} \end{aligned}$$

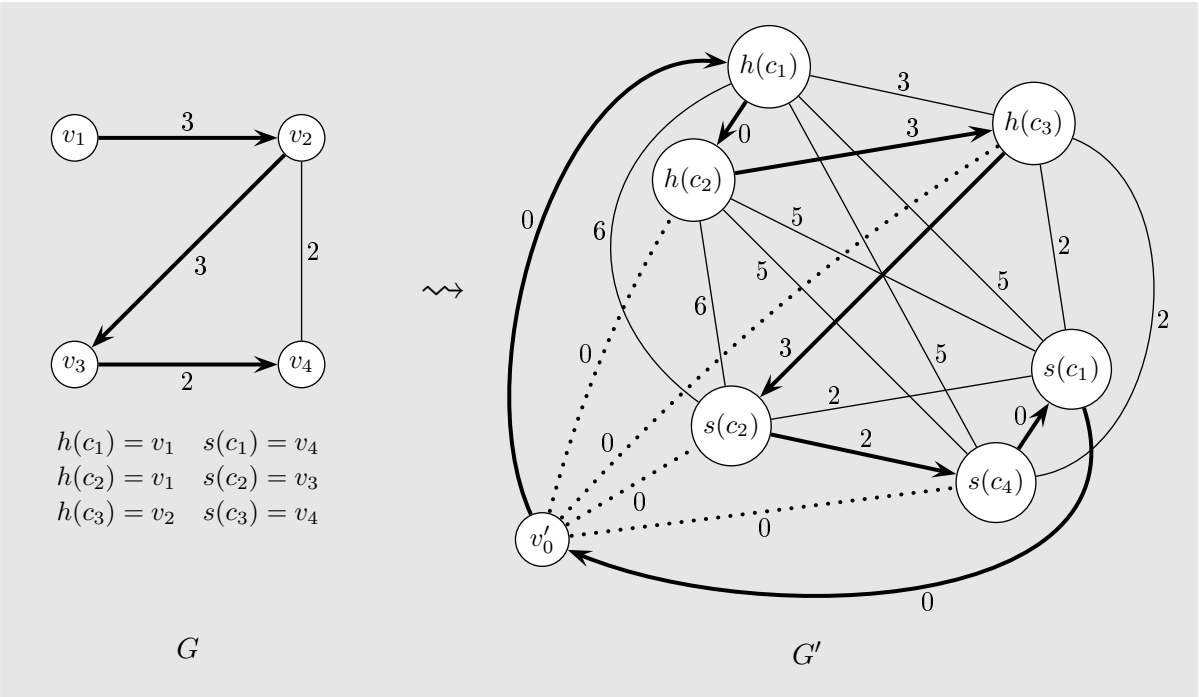


FIG. 2.2 – Transformation du réseau G de la modélisation de type «chemins ouverts» en G' pour une modélisation de type «tournées de véhicules» et représentation de la planification $\mathcal{P}$.

Cette modélisation a un prix, car elle augmente sensiblement le nombre de sommets et la densité du réseau. Dans ce cas particulier, on passe de $|\mathcal{V}| = 4$ sommets et $|\mathcal{E}| = 4$ arêtes à $|\mathcal{V}'| = 7$ sommets et $|\mathcal{E}'| = 21$ arêtes.

On a ainsi montré que l'on peut voir le problème de la planification de tournées de véhicules scolaires comme un problème classique de tournées de véhicules, auquel il convient d'ajouter des contraintes de précédence entre les sommets, des contraintes de capacité et des contraintes d'heure de visite au plus tard en certains sommets.

2.7. Trajet des enfants de l'école à la maison

Le problème consistant à ramener les enfants de l'école à la maison, nommé *retour*, est très similaire au problème présenté dans les sections 2.1 à 2.4, nommé *aller*. Schématiquement, il suffit d'inverser les rôles des maisons et des écoles. Cependant, au niveau des écoles, on note que l'on n'a pas d'heure d'arrivée au plus tard, mais une heure de départ au plus tôt, *i.e.* l'heure de fin des cours. La définition de la perte de temps des enfants est également légèrement modifiée. Le temps d'attente d'un enfant correspond au temps qu'un enfant attend dans la cour d'école entre la fin des cours et le moment où il monte dans un bus. Quant au supplément de temps

de trajet, il reste inchangé. Notons que si l'on a une planification admissible pour le problème aller, il ne suffit généralement pas de la renverser pour obtenir une bonne solution du problème retour. Renverser une planification consiste à faire monter dans le bus les enfants en l'arrêt où ils descendent et *vice versa*, ainsi qu'à faire parcourir au bus le trajet en sens inverse. Comme l'échelonnement des heures de début des cours n'est pas forcément identique à celui des heures de fin des cours, on ne dispose d'aucune garantie sur la qualité d'une telle planification *renversée*. Des résultats numériques pour le problème retour sont présentés dans la section 4.5.

CHAPITRE 3

À la recherche de planifications efficaces

Dans ce chapitre, nous décrivons comment obtenir une ou plusieurs planifications efficaces. Dans un premier temps, une planification, dont la principale raison d'être est son admissibilité, est construite. Étant généralement de mauvaise qualité, cette solution est ensuite améliorée à l'aide de diverses heuristiques telles que le recuit simulé et la recherche tabou.

3.1. Construction d'une planification admissible

L'idée générale de la méthode est de commencer par construire une planification vérifiant toutes les contraintes du problème, sauf éventuellement celle concernant le nombre B de bus disponibles (ce processus comprend deux étapes décrites dans les sections 3.1.1 et 3.1.2). Dans un deuxième temps, les tournées obtenues sont concaténées afin de ramener leur nombre à B (voir section 3.1.3).

3.1.1. Génération de chemins

La première étape de notre construction consiste à générer un ensemble de chemins constitués d'une succession de maisons se terminant par une école. Un exemple d'une telle construction est représenté dans la figure 3.1.

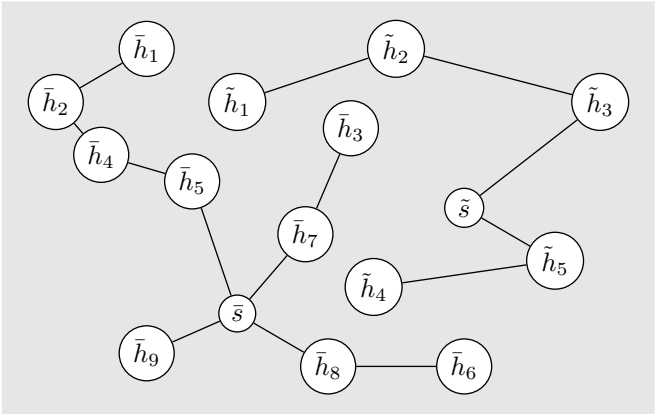


FIG. 3.1 – Représentation de six chemins vers deux écoles. Les maisons $\bar{h}$ (respectivement $\tilde{h}$) des enfants devant se rendre en l'école $\bar{s}$ (respectivement $\tilde{s}$) sont triées par ordre décroissant de leur temps de trajet jusqu'à $\bar{s}$ (respectivement $\tilde{s}$).

Avant de construire ces chemins, nous allons décrire leurs attributs. On note $\mathcal{U} = (v_1, \dots, v_n)$ un *chemin* correspondant à une liste de sommets $v_j \in \{1, \dots, |\mathcal{V}|\}$ dont les $n-1$ premiers sont des maisons et le dernier est une école. On définit le *temps de trajet total* du chemin $\mathcal{U}$ par

$$L(\mathcal{U}) = \sum_{j=1}^{n-1} d(v_j, v_{j+1}). \tag{3.1}$$

Posons $Q = \min_{b \in \mathcal{B}} Q_b$, la capacité du plus petit bus à disposition. Nous allons construire des chemins satisfaisant la contrainte

$$\sum_{c \in \mathcal{C}_{\mathcal{U}}} q(c) \leq Q, \tag{3.2}$$

où

$$\mathcal{C}_{\mathcal{U}} = \{c \in \mathcal{C} \mid h(c) \in \{v_1, \dots, v_{n-1}\}, s(c) = v_n\},$$

c'est-à-dire permettant de transporter au plus le nombre d'enfants qu'il est possible de mettre dans le plus petit bus à disposition. Par la suite, ces chemins seront transformés en tournées (au sens de leur définition en page 8) qu'il sera possible d'effectuer en utilisant n'importe quel véhicule.

Le procédé de construction des chemins est le suivant : pour chaque école $s \in \mathcal{S}$, on construit un ou plusieurs chemins satisfaisant la contrainte de capacité (3.2). Plus précisément, pour chaque école $s \in \mathcal{S}$, les maisons associées $\mathcal{H}(s)$ sont triées dans l'ordre décroissant de leur temps de trajet jusqu'à s. On crée un chemin en y insérant successivement des maisons, dans cet ordre, aussi longtemps que la contrainte (3.2) est vérifiée. S'il n'est pas possible d'y placer toutes les maisons de $\mathcal{H}(s)$, un nouveau chemin est créé de la même manière. On note N le nombre total de chemins obtenus. La construction de ces N chemins $\mathcal{U}_i = (v_1^i, \dots, v_{n_i}^i)$ est détaillée dans l'algorithme 3.1.

Remarque 3.1. *Afin de simplifier l'écriture, on identifiera dorénavant les sommets avec leur numéro ; par exemple, on utilisera indifféremment un sommet $s \in \mathcal{S} \subseteq \mathcal{V}$ ou son numéro $s^v \in \{1, \dots, |\mathcal{V}|\}$, pour se référer à une école.*

Algorithme 3.1 Création de N chemins

Résultat : $\{\mathcal{U}_i\}_{i=1}^N$

$i \leftarrow 0$

Pour tout $s \in \mathcal{S}$ **faire** (∗ pour chaque école ∗)

$\widehat{\mathcal{H}} \leftarrow$ liste des maisons $\mathcal{H}(s)$ triées dans l'ordre décroissant de leur temps de trajet à s

Tant que $\widehat{\mathcal{H}}$ est non vide **faire**

$i \leftarrow i + 1$, $\mathrm{occ}(\mathcal{U}_i) \leftarrow 0$, $\mathcal{U}_i \leftarrow (\ )$ (∗ création d'un chemin vide ∗)

Pour toute maison $h \in \widehat{\mathcal{H}}$ (considérées dans l'ordre donné) **faire**

Soit $c \in \mathcal{C}$ l'unique enfant tel que $h(c) = h$ et $s(c) = s$.

Si $(\mathrm{occ}(\mathcal{U}_i) + q(c)) \leq Q$ **alors** (∗ vérification de la capacité ∗)

$\mathcal{U}_i \leftarrow$ **InsertionSommet**$(\mathcal{U}_i, h)$ (∗ à la meilleure position ∗)

retirer h de la liste $\widehat{\mathcal{H}}$, $\mathrm{occ}(\mathcal{U}_i) \leftarrow \mathrm{occ}(\mathcal{U}_i) + q(c)$

$\mathcal{U}_i \leftarrow$ **InsertionÉcole**$(\mathcal{U}_i, s)$ (∗ insertion de l'école ∗)

$N \leftarrow i$

Retourner $\{\mathcal{U}_i\}_{i=1}^N$

L'insertion d'un sommet dans un chemin est basée sur le principe d'augmentation minimale du temps de trajet total du chemin (voir équation (3.1)). Cette procédure est décrite dans l'algorithme 3.2. Pour terminer, à chaque chemin $\mathcal{U}_i = (v_1, \dots, v_{n_i-1})$, avec $i \in \{1, \dots, N\}$, il convient d'ajouter le sommet correspondant à l'école, $s(i)$. Le sens de parcours du chemin étant jusqu'à présent indifférent dans le cadre de notre construction, l'insertion de l'école peut se faire à chacune de ses deux extrémités, de manière à obtenir un chemin de temps de trajet total minimal. Si nécessaire, l'ordre de parcours du chemin est donc inversé de manière à ce que l'école apparaisse en dernier. Cette dernière étape est détaillée dans l'algorithme 3.3.

Algorithme 3.2 InsertionSommet Insertion du sommet v dans le chemin $\mathcal{U}$

Donnée : v et $\mathcal{U}$ contenant m sommets

Résultat : un chemin contenant $m+1$ sommets

Si $m = 0$ **alors** (∗ chemin vide ∗)

Retourner $\mathcal{U}' \leftarrow (v)$

Sinon

$\mathcal{U}_0 \leftarrow (v, v_1, \dots, v_m)$ (∗ insertion en début de chemin ∗)

Pour i **de** 1 **à** m **faire**

$\mathcal{U}_i \leftarrow (v_1, \dots, v_i, v, v_{i+1}, \dots, v_m)$ (∗ insertion après la position i ∗)

$k \leftarrow \operatorname{argmin}_{j \in \{0,\dots,m\}} L(\mathcal{U}_j)$ (∗ plus court chemin ∗)

Retourner $\mathcal{U}_k$

Algorithme 3.3 InsertionÉcole Insertion de l'école s dans le chemin $\mathcal{U}$

Donnée : s et $\mathcal{U} = (v_1, \dots, v_{n-1})$

Résultat : un chemin contenant n sommets

$\mathcal{U}_f \leftarrow (v_1, \dots, v_{n-1}, s)$ (∗ insertion en fin de chemin ∗)

$\mathcal{U}_d \leftarrow (v_{n-1}, \dots, v_1, s)$ (∗ insertion en début de chemin ∗)

Si $L(\mathcal{U}_f) \leq L(\mathcal{U}_d)$ **alors**

Retourner $\mathcal{U}_f$

Sinon

Retourner $\mathcal{U}_d$

3.1.2. Génération de tournées

Afin de transformer les chemins construits en tournées (au sens de leur définition en page 8), il convient de leur associer les ensembles d'enfants montant et descendant en chaque arrêt, ainsi

qu'un horaire. La i^{e} tournée $\mathcal{T}_i$, associée au chemin $\mathcal{U}_i = (v_1^i, \dots, v_{n_i}^i)$, est décrite par la suite des arrêts $(a_\alpha^i)_{\alpha=1}^{n_i}$, où a_α^i est le quadruplet $(v_\alpha^i, t_\alpha^i, \overline{\mathcal{C}}_\alpha^i, \underline{\mathcal{C}}_\alpha^i)$ (voir (2.1)) défini comme suit :

- ▷ $t_{n_i}^i = o_{s(i)}$, *i.e.* le bus est planifié pour arriver exactement à l'heure de début des cours ;
- ▷ $t_{n_i-\alpha}^i = t_{n_i-\alpha+1}^i - d(v_{n_i-\alpha}^i, v_{n_i-\alpha+1}^i)$, $\forall \alpha \in \{1, \dots, n_i - 1\}$, *i.e.* tout déplacement est effectué en utilisant un plus court chemin ;
- ▷ $\overline{\mathcal{C}}_\alpha^i = \{c \in \mathcal{C} \mid h(c) = v_\alpha^i, s(c) = s(i)\}$, $\forall \alpha \in \{1, \dots, n_i - 1\}$, *i.e.* seuls les enfants se rendant en l'école $s(i)$ montent dans le bus ;
- ▷ $\overline{\mathcal{C}}_{n_i}^i = \varnothing$, *i.e.* aucun enfant ne monte dans le bus à l'école ;
- ▷ $\underline{\mathcal{C}}_\alpha^i = \varnothing$, $\forall \alpha \in \{1, \dots, n_i - 1\}$, *i.e.* aucun enfant ne descend en une maison ;
- ▷ $\underline{\mathcal{C}}_{n_i}^i = \bigcup_{\alpha=1}^{n_i-1} \overline{\mathcal{C}}_\alpha^i$, *i.e.* tous les enfants montés dans le bus descendent à l'école.

3.1.3. Concaténation des tournées

La planification en N tournées, obtenue par l'application de la procédure décrite ci-dessus, n'est admissible que si $N \leq B$, où B est le nombre de bus réellement disponibles. Dans le cas contraire, nous allons concaténer les tournées deux à deux jusqu'à en obtenir exactement B. La concaténation de deux tournées $\mathcal{T}_j = (a_\alpha^j)_{\alpha=1}^{n_j}$ et $\mathcal{T}_k = (a_\alpha^k)_{\alpha=1}^{n_k}$ (l'ordre est important) peut conduire à deux situations : soit il est possible de les parcourir l'une après l'autre sans changement d'horaire, soit il est nécessaire d'avancer l'horaire de la première pour pouvoir effectuer la seconde dans les temps. Le premier cas se produit lorsque, entre l'heure $t_{n_j}^j$ de fin de la tournée $\mathcal{T}_j$ et l'heure t_1^k de début de la tournée $\mathcal{T}_k$, il y a assez de temps pour se rendre du dernier sommet $v_{n_j}^j$ de la tournée $\mathcal{T}_j$ au premier sommet v_1^k de la tournée $\mathcal{T}_k$, c'est-à-dire si

$$t_{n_j}^j + d(v_{n_j}^j, v_1^k) \leq t_1^k.$$

D'une manière générale, on mesure l'impact de la concaténation des tournées $\mathcal{T}_j$ et $\mathcal{T}_k$ par la grandeur

$$w_{jk} = t_1^k - t_{n_j}^j - d(v_{n_j}^j, v_1^k).$$

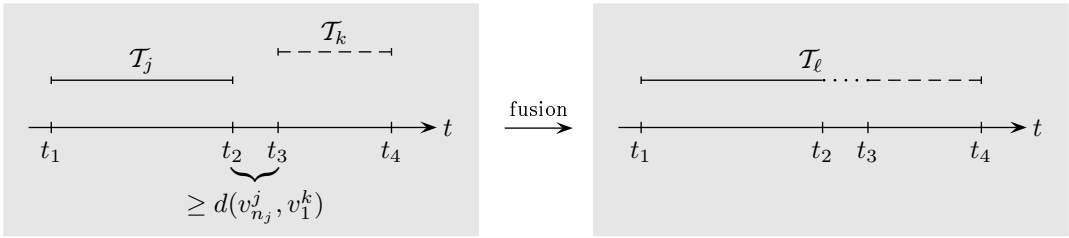
Si w_{jk} est positif, cela signifie que l'heure de début de la tournée $\mathcal{T}_j$ ne serait pas affectée par la fusion. Dans ce cas, w_{jk} représente un temps d'attente pour le conducteur effectuant d'abord la tournée $\mathcal{T}_j$ puis la tournée $\mathcal{T}_k$ (voir figure 3.2(a)). Si w_{jk} est négatif, cela signifie qu'en cas de fusion, il faudrait avancer l'heure de début de la tournée $\mathcal{T}_j$ pour arriver dans les temps à l'école $s(k)$ de la tournée $\mathcal{T}_k$. Dans ce cas, $-w_{jk}$ indique de combien de minutes il faudrait anticiper l'horaire de toute la tournée $\mathcal{T}_j$ (voir figure 3.2(b)).

Lors de la concaténation de tournées, le premier objectif est d'effectuer une fusion qui ne modifie pas l'heure de début des tournées (rappelons que l'on cherche à minimiser la perte de temps des enfants) et de minimiser accessoirement le temps d'attente du conducteur effectuant successivement les tournées en question. Plus concrètement, partant de l'ensemble de tournées $\Psi = \{\mathcal{T}_1, \dots, \mathcal{T}_N\}$ résultant des procédures de création de tournées des sections 3.1.1 et 3.1.2, on définit les deux ensembles suivants :

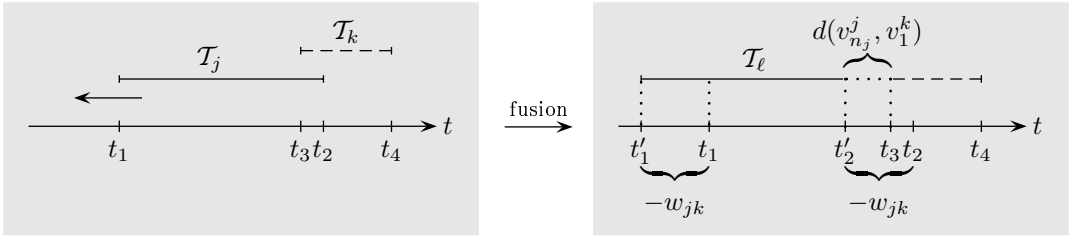
$$\Psi_+^2 \leftarrow \{(\mathcal{T}_r, \mathcal{T}_s) \in \Psi^2 \mid r \neq s, t_1^k - t_{n_j}^j - d(v_{n_j}^j, v_1^k) \geq 0\}$$

et

$$\Psi_-^2 \leftarrow \{(\mathcal{T}_r, \mathcal{T}_s) \in \Psi^2 \mid r \neq s, t_1^k - t_{n_j}^j - d(v_{n_j}^j, v_1^k) < 0\}.$$



(a) $w_{jk} \geq 0$



(b) $w_{jk} < 0$

FIG. 3.2 – Les deux cas pouvant se présenter lors de la concaténation des tournées $\mathcal{T}_j$ et $\mathcal{T}_k$ en une nouvelle tournée $\mathcal{T}_\ell$.

Si l'ensemble Ψ^2_+ est non vide, on choisit une paire (j, k) telle que

$$w_{jk} \quad = \min_{(\mathcal{T}_r, \mathcal{T}_s) \in \Psi^2_+} w_{rs},$$

c'est-à-dire minimisant le temps d'attente du chauffeur. Si Ψ^2_+ est vide, cela signifie qu'il va falloir avancer l'heure de la première tournée choisie pour la concaténation. Parmi l'ensemble de tournées Ψ, la concaténation qui minimise les changements d'horaire sera choisie. Dans ce cas, on choisit donc une paire (j, k) de tournées telle que

$$w_{jk} \quad = \max_{(\mathcal{T}_r, \mathcal{T}_s) \in \Psi^2_-} w_{rs}.$$

Les tournées $\mathcal{T}_j$ et $\mathcal{T}_k$ sont alors concaténées en une tournée $\mathcal{T}_\ell$. Le nouvel ensemble de tournées devient $\Psi \setminus \{\mathcal{T}_j, \mathcal{T}_k\} \cup \{\mathcal{T}_\ell\}$ et sa taille diminue d'une unité. Ce procédé est réitéré tant que Ψ contient plus de B tournées. La procédure de concaténation de deux tournées données est décrite dans l'algorithme 3.5. Le processus de sélection des tournées à fusionner est détaillé dans l'algorithme 3.4. L'ensemble de tournées finalement obtenu constitue une planification admissible (*i.e.* vérifiant toutes les contraintes énoncées dans la section 2.3).

3.1.4. Cas particulier

Il subsiste un cas particulier qui n'a pas encore été considéré : l'existence d'au moins un enfant $c \in \mathcal{C}$ tel que $q(c) > Q$. Dans ce cas, les méthodes proposées ci-dessus ne permettent pas de trouver une planification admissible. Pour y remédier, on construit une planification initiale en n'utilisant qu'un sous-ensemble des véhicules :

$$\overline{\mathcal{B}} = \{b \in \mathcal{B} \mid Q_b \geq \max_{c \in \mathcal{C}} q(c)\} \subset \mathcal{B}.$$

Algorithme 3.4 Réduction éventuelle du nombre de tournées

Donnée : $\Psi = \{\mathcal{T}_1, \dots, \mathcal{T}_N\}$

Résultat : une planification admissible d'au plus B tournées

Tant que $|\Psi| > B$ **faire**

$\Psi^2_+ \leftarrow \{(\mathcal{T}_r, \mathcal{T}_s) \in \Psi^2 \mid r \neq s, t^k_1 - t^j_{n_j} - d(v^j_{n_j}, v^k_1) \geq 0\}$

$\Psi^2_- \leftarrow \{(\mathcal{T}_r, \mathcal{T}_s) \in \Psi^2 \mid r \neq s, t^k_1 - t^j_{n_j} - d(v^j_{n_j}, v^k_1) < 0\}$

Si $\Psi^2_+ \neq \varnothing$ **alors**

$(j,k) \leftarrow \underset{(\mathcal{T}_r,\mathcal{T}_s)\in\Psi^2_+}{\operatorname{argmin}} \; t^s_1 - t^r_{n_r} - d(v^r_{n_r}, v^s_1)$

$w_{jk} \leftarrow t^k_1 - t^j_{n_j} - d(v^j_{n_j}, v^k_1)$

Sinon

$(j,k) \leftarrow \underset{(\mathcal{T}_r,\mathcal{T}_s)\in\Psi^2_-}{\operatorname{argmax}} \; t^s_1 - t^r_{n_r} - d(v^r_{n_r}, v^s_1)$

$w_{jk} \leftarrow t^k_1 - t^j_{n_j} - d(v^j_{n_j}, v^k_1)$

$\mathcal{T}_\ell \leftarrow$ **Concaténer**$(\mathcal{T}_j, \mathcal{T}_k, w_{jk})$

$\Psi \leftarrow \Psi \setminus \{\mathcal{T}_j, \mathcal{T}_k\} \cup \{\mathcal{T}_\ell\}$

Retourner Ψ

Algorithme 3.5 Concaténer Concaténation des tournées $\mathcal{T}_j$ et $\mathcal{T}_k$

Donnée : $\mathcal{T}_j = (a^j_\alpha)^{n_j}_{\alpha=1}$, $\mathcal{T}_k = (a^k_\alpha)^{n_k}_{\alpha=1}$, w_{jk}

Résultat : $\mathcal{T}_\ell = (a^\ell_\alpha)^{n_\ell}_{\alpha=1}$

Si $w_{jk} < 0$ **alors** (∗ décalage de la première tournée ∗)

$t^j_\alpha \leftarrow t^j_\alpha + w_{jk}$ pour chaque $\alpha \in \{1, \dots, n_j\}$

$a^\ell_\alpha \leftarrow a^j_\alpha$ pour $\alpha \in \{1, \dots, n_j\}$

$a^\ell_\alpha \leftarrow a^k_{\alpha - n_j}$ pour $\alpha \in \{n_j + 1, \dots, n_j + n_k\}$.

Retourner $(a^\ell_\alpha)^{n_\ell}_{\alpha=1}$, où $n_\ell = n_j + n_k$

La planification initiale obtenue comprend donc $|\mathcal{B}| - |\overline{\mathcal{B}}|$ bus vides (rappelons que, compte tenu de l'hypothèse énoncée en page 7 sur la taille des enfants, l'ensemble $\overline{\mathcal{B}}$ n'est jamais vide). Ces véhicules vides seront remplis dans une seconde phase, lors de l'amélioration de la planification, à l'aide des méthodes décrites dans la section suivante.

3.2. Amélioration des planifications

Dans cette section, nous allons décrire des méthodes permettant d'améliorer la qualité d'une planification (obtenue par exemple en appliquant la procédure décrite dans la section précédente). Cette seconde phase revient à minimiser les fonctions objectif $\mathcal{F}_1$ ou $\mathcal{F}_2$ sous les contraintes (2.9) à (2.17). Précisons que ces méthodes s'appliquent à toute planification admissible, ou violant éventuellement la contrainte (2.16) sur la capacité des véhicules.

Plusieurs heuristiques de recherche locale ont été implémentées et comparées : un recuit simulé décrit dans la section 3.2.4 et une recherche tabou décrite dans la sections 3.2.5, chacune de ces heuristiques ayant été testée dans des versions explorant des voisinages constitués uniquement de planifications admissibles ou comprenant également des planifications non admissibles. Les performances de ces méthodes sont analysées et discutées au chapitre 4.

3.2.1. Recherche locale

Les méthodes de recherche locale constituent une approche standard pour la recherche de solutions aux problèmes d'optimisation combinatoire. De telles techniques s'articulent autour d'une structure de voisinage sur un ensemble de planifications envisageables. Cette structure peut être représentée par un graphe orienté, dont les sommets correspondent aux planifications, chaque planification étant reliée à ses voisines par un arc du graphe. Une planification voisine $\mathcal{P}'$ de $\mathcal{P}$ est obtenue en apportant des modifications locales à $\mathcal{P}$. Notons que si $\mathcal{P}'$ est voisine de $\mathcal{P}$, $\mathcal{P}$ n'est pas forcément voisine de $\mathcal{P}'$. Il est cependant souhaitable que ce graphe soit quasi-fortement connexe pour que, partant d'une planification initiale quelconque, il soit toujours possible d'atteindre une meilleure planification. Appliquer une heuristique de recherche locale revient à explorer ce graphe de voisinage selon une stratégie donnée. Dans la section 3.2.2, on propose deux structures de voisinage et dans les sections 3.2.4 et 3.2.5 deux stratégies d'exploration du graphe de voisinage : une variante du recuit simulé (voir [KGV83], [RTL86] et [AK89]) avec réchauffement cyclique et une recherche tabou (voir [Glo86], [Han86] et [HJ87]).

3.2.2. Structures de voisinage

Le voisinage d'une planification $\mathcal{P}$ est un ensemble de planifications obtenues en déplaçant un enfant c du bus qui le transporte $\beta^{\mathcal{P}}(c)$ dans un autre bus $b \in \overline{\mathcal{B}}^{\mathcal{P}}(c)$, où

$$\overline{\mathcal{B}}^{\mathcal{P}}(c) = \{b \in \mathcal{B} \mid b \neq \beta^{\mathcal{P}}(c), Q_b \geq q(c)\}.$$

Une fois l'enfant c retiré de $\mathcal{T}_{\beta^{\mathcal{P}}(c)}$, on élimine les détours éventuels que le bus $\beta^{\mathcal{P}}(c)$ effectuait pour cet enfant. L'insertion de l'enfant c dans b est faite de manière à allonger le moins possible le temps total de trajet de la tournée $\mathcal{T}_b$. On définit le *temps total de trajet* d'une tournée $\mathcal{T}_i = (a^i_\alpha)_{\alpha=1}^{n_i}$ par

$$L(\mathcal{T}_i) = t^i_{n_i} - t^i_1. \tag{3.3}$$

Si l'insertion de l'enfant c est faite de manière à ne violer aucune contrainte, on parle de *voisinage admissible* $\mathcal{N}_f(\mathcal{P})$. Si l'insertion de l'enfant c est faite de manière à tenir compte de toutes les contraintes du problème, sauf éventuellement celle concernant la capacité du bus b (2.16), on parle de *voisinage non admissible* $\mathcal{N}_i(\mathcal{P})$. Par conséquent, on a $\mathcal{N}_f(\mathcal{P}) \subseteq \mathcal{N}_i(\mathcal{P})$. Remarquons que, quelle que soit la planification $\mathcal{P}$, il est toujours possible d'insérer n'importe quel enfant c dans n'importe quel bus $b \in \overline{\mathcal{B}}^{\mathcal{P}}(c)$. En effet, bien que ce ne soit généralement pas judicieux, il est toujours possible d'insérer l'enfant c en début de tournée, ce qui n'affecte en rien le reste de la planification.

L'insertion d'un enfant c dans la tournée du bus b est effectuée en trois étapes :

- ▷ insertion de l'arrêt correspondant à l'école : $a_s = (s(c), t_s, \varnothing, \{c\})$;
- ▷ insertion de l'arrêt correspondant à la maison : $a_h = (h(c), t_h, \{c\}, \varnothing)$;
- ▷ calcul du nouvel horaire de la tournée (en particulier détermination de t_h et de t_s).

Détaillons tout d'abord ce dernier point. Si l'on est en présence d'une tournée (visitant plusieurs écoles) sans horaire établi, l'horaire qui lui est affecté doit être «calé à droite», c'est-à-dire que l'on va faire en sorte que les écoles soient visitées le plus tard possible, quitte à générer un temps d'attente pour le chauffeur (rappelons que l'on cherche à minimiser la perte de temps des enfants). Le calcul de l'horaire «calé à droite» d'une tournée est décrit dans l'algorithme 3.6 et nécessite $O(|\mathcal{C}|)$ opérations dans le pire des cas.

Algorithme 3.6 CalculHoraire Calcul de l'horaire «calé à droite» d'une tournée

Donnée : $\mathcal{T} = (a_\alpha)_{\alpha=1}^n$, où $a_\alpha = (v_\alpha, t_\alpha, \overline{\mathcal{C}}_\alpha, \underline{\mathcal{C}}_\alpha)$ pour tout $\alpha \in \{1, \dots, n\}$

$t_n \leftarrow o_{v_n}$ (∗ fin de la tournée en l'heure de début des cours de l'école v_n ∗)

Pour α **de** 1 **à** n **faire** (∗ calcul de l'horaire à rebours ∗)

$t_{n-\alpha} \leftarrow t_{n-\alpha+1} - d(v_{n-\alpha+1}, v_{n-\alpha})$

Si $v_{n-\alpha} \in \mathcal{S}$ **et** $t_{n-\alpha} > o_{v_{n-\alpha}}$ **alors** (∗ le sommet courant est une école ∗)

$t_{n-\alpha} \leftarrow o_{v_{n-\alpha}}$ (∗ on remonte l'horaire pour arriver à l'heure ∗)

Retourner $\mathcal{T}$

Pour l'insertion des arrêts correspondant à l'école et à la maison de l'enfant c à placer, deux cas principaux sont à distinguer : soit le bus b est vide, soit il ne l'est pas.

Le bus est vide : la tournée $\mathcal{T}_b$ va desservir uniquement l'enfant c. Elle ne sera donc constituée que de deux arrêts, les arrêts correspondant à la maison et à l'école :

$$\mathcal{T}_b = (a_1, a_2) = (a_h, a_s).$$

Son horaire est calculé à l'aide de l'algorithme 3.6.

Le bus n'est pas vide : partant de la fin de la tournée, on commence par déterminer quel est le dernier arrêt α après lequel il est possible d'insérer l'arrêt correspondant à l'école a_s, de manière à ce que l'arrivée en $s(c)$ puisse avoir lieu avant le début des cours, *i.e.* le dernier arrêt α de la tournée tel que

$$t_\alpha^b + d(v_\alpha^b, s(c)) \leq o_{s(c)}.$$

L'ensemble des emplacements candidats à l'insertion de l'arrêt école s'étend du début de la tournée jusqu'à «juste après» l'arrêt α. Pour chacun de ces placements potentiels de l'arrêt école, l'arrêt maison correspondant peut être placé à n'importe quelle position précédente dans la tournée. Toutefois, lorsque l'on utilise le voisinage admissible $\mathcal{N}_f$, un tel placement des arrêts maison et école n'est envisageable que si la contrainte de capacité (2.16) reste vérifiée en tout arrêt compris entre ceux-ci. Pour chacune de ces possibilités, l'horaire de la tournée résultante est calculé à l'aide de l'algorithme 3.6 et l'on retient finalement la solution qui minimise le temps total de trajet au sens de l'expression (3.3).

Cas particuliers : si l'arrêt i situé juste avant ou après le point d'insertion de l'école est tel que $v_i = s(c)$, l'arrêt a_s n'a pas besoin d'être inséré dans la tournée et il suffit de modifier $\underline{\mathcal{C}}_i$ en $\underline{\mathcal{C}}_i \cup \{c\}$. De même, si l'arrêt j situé juste avant ou après le point d'insertion de la maison est tel que $v_j = h(c)$, l'arrêt a_h n'a pas besoin d'être inséré dans la tournée et il suffit de modifier $\overline{\mathcal{C}}_j$ en $\overline{\mathcal{C}}_j \cup \{c\}$.

La détermination d'une planification voisine requiert $O(|\mathcal{C}|^3)$ opérations dans le pire des cas. En effet, le nombre de placements à considérer de l'école et de la maison d'un enfant c dans une tournée est de l'ordre de $O(|\mathcal{C}|^2)$. De plus, pour chacune de ces possibilités, il faut vérifier que la contrainte sur la capacité du bus est respectée sur toute la durée de la tournée (lorsque l'on utilise le voisinage admissible $\mathcal{N}_f$) et calculer un nouvel horaire «calé à droite», ce qui nécessite $O(|\mathcal{C}|)$ opérations.

Dans nos heuristiques, l'enfant déplacé c est choisi aléatoirement selon la loi de probabilité :

$$P(\text{choisir } c) = \frac{f(c)^\lambda}{\sum_{c\in\mathcal{C}} f(c)^\lambda}, \quad \forall c \in \mathcal{C}, \tag{3.4}$$

où $f(c)$ est la perte de temps de l'enfant c, définie par (2.6) et $\lambda \in [0,1]$ est un paramètre d'optimisation. Une valeur de λ nulle signifie que l'on assigne la même probabilité à tous les enfants. Dans les autres cas, on favorise le déplacement d'enfants présentant une grande perte de temps. Le bus d'arrivée b est choisi aléatoirement de manière équiprobable sur l'ensemble $\overline{\mathcal{B}}^{\mathcal{P}}(c)$.

Notons également que le nombre de planifications envisageables, *i.e.* le nombre de sommets du graphe de voisinage, est bien plus grand que la simple affectation enfant/bus ($|\mathcal{C}|\cdot B$). En effet, à un véhicule b et à un certain sous-ensemble d'enfants correspondent une multitude de tournées $\mathcal{T}_b$, vérifiant toutes les contraintes sauf éventuellement la contrainte de capacité (2.16). On peut aussi remarquer que la relation de voisinage n'est pas symétrique, *i.e.* $\mathcal{P}' \in \mathcal{N}(\mathcal{P}) \not\Rightarrow \mathcal{P} \in \mathcal{N}(\mathcal{P}')$. En effet, considérons une planification $\mathcal{P}$ telle qu'il existe un enfant c qu'il est possible de placer à un autre endroit à l'intérieur de la même tournée, tout en en réduisant le temps total de trajet. Soit $\mathcal{P}'$ une planification voisine de $\mathcal{P}$ obtenue en déplaçant l'enfant c dans un autre bus. La planification $\mathcal{P}$ ne sera pas voisine de $\mathcal{P}'$, car lors du «retour» de l'enfant c dans le bus initial, il ne sera pas inséré au même endroit que dans la planification de départ $\mathcal{P}$.

Remarquons que si $\mathcal{P}$ est une planification admissible, son voisinage admissible $\mathcal{N}_f(\mathcal{P})$ est uniquement constitué de planifications admissibles. Si par contre, $\mathcal{P}$ n'est pas admissible (*i.e.* viole la contrainte de capacité (2.16)), $\mathcal{N}_f(\mathcal{P})$ contient également des planifications violant la capacité, mais ce voisinage ne comprend que des planifications pour lesquelles cette violation n'est pas augmentée par rapport à celle de $\mathcal{P}$. Plus précisément, soit

$$\mathrm{occ}^b_\alpha(\mathcal{P}) = \sum_{k=1}^{\alpha}\sum_{c=1}^{|\mathcal{C}|} q(c)\left(\overline{\delta}^b_{ck} - \underline{\delta}^b_{ck}\right)$$

l'occupation du bus b en l'arrêt α. La *violation totale de capacité* de la planification $\mathcal{P}$ est définie par

$$V(\mathcal{P}) = \sum_{b=1}^{B}\sum_{\alpha=1}^{n_b} \max\{0, \mathrm{occ}^b_\alpha(\mathcal{P}) - Q_b\}.$$

Si la planification $\mathcal{P}$ n'est pas admissible, son voisinage admissible $\mathcal{N}_f(\mathcal{P})$ ne comprend que des planifications $\mathcal{P}'$ (obtenues en déplaçant un enfant dans un autre bus) telles que $V(\mathcal{P}') \leq V(\mathcal{P})$.

3.2.3. Restauration de l'admissibilité

Quelle que soit la stratégie d'exploration du graphe de voisinage adoptée, le but est de trouver une planification admissible efficace. Or, il est clair que des planifications non admissibles ont tendance à avoir une meilleure valeur de fonction objectif que celles qui sont admissibles. Par conséquent, afin de restaurer l'admissibilité, on ajoute à la fonction objectif $\mathcal{F} \in \{\mathcal{F}_1, \mathcal{F}_2\}$ une pénalité en fonction de la violation de capacité que présente la planification. On redéfinit donc les deux fonctions objectif $\mathcal{F}_1$ et $\mathcal{F}_2$ comme suit :

$$\mathcal{F}_1(\mathcal{P}) = \sum_{c \in \mathcal{C}} f(c)n(c) + \kappa \sum_{b=1}^{B} \max\{0, \mathrm{occ}^b_{\max} - Q_b\}, \tag{3.5}$$

$$\mathcal{F}_2(\mathcal{P}) = \max_{c \in \mathcal{C}} f(c) + \kappa \sum_{b=1}^{B} \max\{0, \mathrm{occ}^b_{\max} - Q_b\}, \tag{3.6}$$

où $\kappa > 1$ est un paramètre d'optimisation et

$$\mathrm{occ}^b_{\max} = \max_{\alpha \in \{1, \dots, n_b\}} \left\{ \sum_{k=1}^{\alpha} \sum_{c=1}^{|\mathcal{C}|} q(c) \left(\overline{\delta}^b_{ck} - \underline{\delta}^b_{ck} \right) \right\} \tag{3.7}$$

est la charge maximale du bus $b \in \mathcal{B}$ sur tous les arrêts $\alpha \in \{1, \dots, n_b\}$. Dans nos heuristiques, le paramètre κ est tout d'abord initialisé à une valeur donnée $\kappa_0 > 1$. Puis, à chaque fois que l'on visite une planification non admissible, κ est multiplié par κ_0. Lorsque l'on retombe sur une planification admissible, la valeur de κ est réinitialisée à κ_0.

3.2.4. Recuit simulé

Le recuit simulé puise ses racines dans la thermodynamique. Cette méthode est inspirée du phénomène physique de refroidissement lent d'un corps en fusion qui le conduit à un état solide de basse énergie. Dans un tel processus, la température est lentement abaissée par paliers, de manière à ce qu'à chacun d'entre eux le corps atteigne un équilibre thermodynamique. Pour les matériaux, cette basse énergie se manifeste par l'obtention d'une structure régulière. Lorsque la température est proche de zéro, seules les transitions vers un état d'énergie plus faible et donc plus stable sont possibles. Metropolis *et al.* [MRR+53] ont proposé un algorithme simulant l'évolution d'un solide en fusion vers un équilibre thermodynamique. À partir de là, Kirkpatrick *et al.* [KGV83], ainsi que Černý [Č85] ont indépendamment adapté cette stratégie à l'optimisation combinatoire. Le recuit simulé peut être vu comme une marche aléatoire sur le graphe de voisinage.

Dans notre cas, partant d'une planification $\mathcal{P}$, une planification $\mathcal{P}'$ est choisie aléatoirement dans le voisinage $\mathcal{N}_i(\mathcal{P})$ avec probabilité φ et dans le voisinage $\mathcal{N}_f(\mathcal{P})$ avec probabilité $(1 - \varphi)$. Cependant, on ne se déplace de $\mathcal{P}$ en $\mathcal{P}'$ qu'avec probabilité

$$p = \min \left\{ 1, \exp\left(-\frac{\mathcal{F}(\mathcal{P}') - \mathcal{F}(\mathcal{P})}{T} \right) \right\}, \tag{3.8}$$

où $T > 0$ est un paramètre d'optimisation appelé *température*. En d'autres termes, on effectue toujours un mouvement améliorant la solution et l'on accepte de visiter une planification de moins bonne qualité avec probabilité $\exp\left(-\frac{\mathcal{F}(\mathcal{P}')-\mathcal{F}(\mathcal{P})}{T}\right)$. Plus la température T est élevée, plus il est probable qu'une planification candidate moins bonne que la planification courante soit acceptée. Au début du déroulement de l'heuristique, la température T_0 est fixée à une valeur élevée, de manière à permettre de s'échapper d'un éventuel optimum local. La température est ensuite périodiquement diminuée (toutes les μ itérations) afin d'intensifier la recherche. Lorsque le

recuit simulé «gèle», c'est-à-dire lorsque plus aucune planification voisine n'est acceptée pendant γ_{max} itérations, on remonte la température T à sa valeur initiale T_0. Par conséquent, T oscille entre une diminution et une augmentation de la probabilité d'accepter des planifications de moins bonne qualité. C'est ce que l'on appelle un *réchauffement cyclique*.

Initialement, la probabilité φ de choix du voisinage $\mathcal{N}_i$ est fixée à $\varphi_0 \in [0, 1)$. Lorsqu'une planification non admissible est visitée, la valeur du paramètre φ est diminuée ; plus précisément, φ est multiplié par un facteur constant $\psi \in (0, 1)$. Lorsque l'on retombe sur une planification admissible, la valeur de φ est réinitialisée à φ_0. Dans le cas où $\varphi_0 = 0$, seules des planifications admissibles sont visitées, le recuit simulé n'utilise que le voisinage $\mathcal{N}_f$ et l'on se trouve en présence d'un «recuit simulé admissible». Dans le cas contraire, des planifications non admissibles sont également visitées, le recuit simulé utilise les deux voisinages $\mathcal{N}_f$ et $\mathcal{N}_i$ et il s'agit alors d'un «recuit simulé non admissible».

Lorsqu'il n'y a pas eu d'amélioration de la meilleure planification admissible rencontrée parmi les dernières $\eta_{\max}$ planifications voisines évaluées, le recuit simulé se termine. On applique alors la procédure d'optimisation Lin-2-opt (voir [Lin65]) sur la meilleure planification trouvée. Cette procédure consiste à considérer toutes les paires d'arcs d'un chemin et à effectuer, tant qu'il en existe, les échanges permettant de réduire sa longueur. La figure 3.3 représente un de ces échanges.

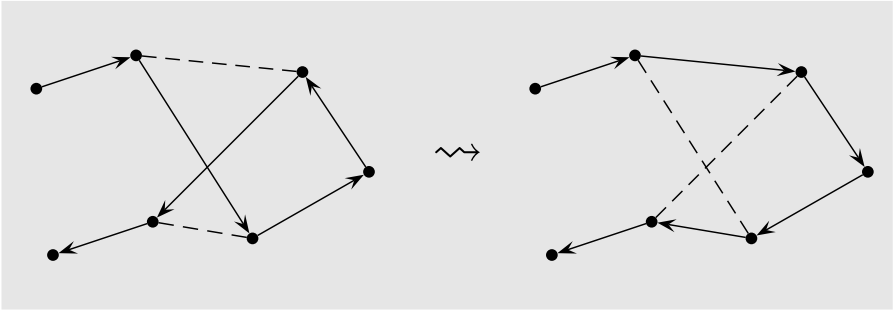


FIG. 3.3 – Application de la procédure Lin-2-opt : échange de deux arcs.

Dans le cas particulier de notre problème, on applique la procédure Lin-2-opt à chaque tournée $\mathcal{T}_b^*$ de la meilleure planification $\mathcal{P}^*$ rencontrée lors de l'application du recuit simulé. Ceci revient à considérer B sous-problèmes. Chaque sous-problème correspond à un seul bus $b \in \mathcal{B}$ de capacité Q_b et les enfants à considérer sont ceux transportés dans la tournée $\mathcal{T}_b^*$, *i.e.* l'ensemble $\mathcal{C}_b = \{c \in \mathcal{C} \mid \beta^{\mathcal{P}^*}(c) = b\}$. Lors de l'application du Lin-2-opt, certains échanges d'arcs pourraient engendrer une tournée non admissible. Ainsi, on ne retiendra que les échanges menant à des tournées admissibles et l'on n'effectuera que ceux permettant de réduire la valeur de la fonction objectif $\mathcal{F}$. L'application du Lin-2-opt à une planification est décrite dans l'algorithme 3.8.

La description générale du recuit simulé développé dans ce travail se trouve dans l'algorithme 3.7. Les paramètres d'optimisation sont :

- ▷ la température initiale T_0 ;
- ▷ le coefficient ω de diminution de la température ;
- ▷ le nombre μ d'itérations consécutives avec la même valeur de la température ;
- ▷ le coefficient de pénalité κ_0 associé à la fonction objectif $\mathcal{F}$;
- ▷ le paramètre λ de la loi de probabilité (3.4) pour le choix de l'enfant à déplacer ;

Algorithme 3.7 Recuit Simulé

Donnée : $\{\mathcal{T}_b\}_{b=1}^{B}$, T_0, ω, μ, κ_0, λ, φ_0, ψ, γ_{max}, η_{max}, $\mathcal{F} \in \{\mathcal{F}_1, \mathcal{F}_2\}$

1 : $T \leftarrow T_0$, $\eta \leftarrow 0$, $\gamma \leftarrow 0$, $\kappa \leftarrow \kappa_0$, $\varphi \leftarrow \varphi_0$, $n \leftarrow 0$, $\mathcal{P} \leftarrow \{\mathcal{T}_b\}_{b=1}^{B}$, $\mathcal{P}^* \leftarrow \mathcal{P}$

2 : **Tant que** $\eta < \eta_{max}$ **faire**

3 : $n \leftarrow n + 1$ (∗ nombre d'itérations ∗)

4 : Choisir aléatoirement un enfant c selon la loi (3.4) et uniformément un bus $b \in \overline{\mathcal{B}}^{\mathcal{P}}(c)$.

5 : **Si** $\varphi \leq (x \sim U[0,1))$ **alors**

6 : $\mathcal{N} \leftarrow \mathcal{N}_i$

7 : **Sinon**

8 : $\mathcal{N} \leftarrow \mathcal{N}_f$

9 : Soit $\mathcal{P}'$ la planification voisine de $\mathcal{P}$ obtenue en insérant c dans b (selon le voisinage $\mathcal{N}$).

10 : **Si** $\mathcal{F}(\mathcal{P}') < \mathcal{F}(\mathcal{P})$ **alors** (∗ trouvé une meilleure planification ∗)

11 : $\mathcal{P} \leftarrow \mathcal{P}'$, $\gamma \leftarrow 0$

12 : **Sinon**

13 : **Si** $\exp\left(-\dfrac{\mathcal{F}(\mathcal{P}') - \mathcal{F}(\mathcal{P})}{T}\right) > (y \sim U[0,1))$ **alors**

14 : $\mathcal{P} \leftarrow \mathcal{P}'$, $\gamma \leftarrow 0$ (∗ moins bonne planification acceptée ∗)

15 : **Sinon**

16 : $\gamma \leftarrow \gamma + 1$

17 : **Si** $\mathcal{P}'$ est admissible **alors**

18 : $\kappa \leftarrow \kappa_0$, $\varphi \leftarrow \varphi_0$

19 : **Sinon si** $\gamma = 0$ **alors** (∗ déplacement vers une planification non admissible ∗)

20 : $\kappa \leftarrow \kappa \cdot \kappa_0$, $\varphi \leftarrow \psi \cdot \varphi$

21 : **Si** $\mathcal{P}'$ est admissible **et** $\mathcal{F}(\mathcal{P}') < \mathcal{F}(\mathcal{P}^*)$ **alors**

22 : $\mathcal{P}^* \leftarrow \mathcal{P}'$, $\eta \leftarrow 0$ (∗ meilleure planification ∗)

23 : **Sinon**

24 : $\eta \leftarrow \eta + 1$

25 : **Si** $(n \mod \mu = 0)$ **alors**

26 : $T \leftarrow \omega \cdot T$ (∗ diminution de la température ∗)

27 : **Si** $(\gamma > \gamma_{max})$ **alors**

28 : $T \leftarrow T_0$, $\gamma \leftarrow 0$ (∗ gel ∗)

29 : **Retourner Lin-2-opt**$(\mathcal{P}^*)$ (∗ optimisation ∗)

Algorithme 3.8 Lin-2-opt Optimisation d'une planification par la procédure Lin-2-opt

Donnée : $\{\mathcal{T}_b\}_{b=1}^{B} = \{(a_\alpha^b)_{\alpha=1}^{n_b}\}_{b=1}^{B}$

Pour tout $b \in \mathcal{B}$ **faire** (∗ chaque tournée ∗)

amélioration ← **vrai**

Tant que amélioration **faire**

amélioration ← **faux**

Pour toute paire d'arcs distincts de $\mathcal{T}_b$ **faire**

Soit $\overline{\mathcal{T}}_b$ la planification obtenue en inversant la paire d'arcs comme sur la figure 3.3.

Si $\overline{\mathcal{T}}_b$ vérifie toutes les contraintes pour $\{c \in \mathcal{C} \mid \beta^{\overline{\mathcal{P}}}(c) = b\}$ **alors**

Si $\mathcal{F}(\{\overline{\mathcal{T}}_b\}) < \mathcal{F}(\{\mathcal{T}_b\})$ **alors**

$\mathcal{T}_b \leftarrow \overline{\mathcal{T}}_b$

amélioration ← **vrai**

Retourner $\{\mathcal{T}_b\}_{b=1}^{B}$

- ▷ la probabilité φ_0 de choisir un voisin dans le voisinage non admissible $\mathcal{N}_i$;
- ▷ le coefficient ψ de diminution de la probabilité de choisir le voisinage non admissible ;
- ▷ le nombre maximum γ_{max} d'itérations consécutives pendant lesquelles on admet que la planification courante ne soit pas changée (*i.e.* contrôle du «gel» du recuit simulé) ;
- ▷ le nombre maximal η_{max} d'évaluations de planifications consécutives sans amélioration de la meilleure planification rencontrée.

Les résultats des tests effectués à l'aide de cette heuristique sont présentés au chapitre 4.

3.2.5. La recherche tabou

La recherche tabou, développée indépendamment par Glover [Glo86] et Hansen [Han86], est une autre méthode de recherche locale. La stratégie d'exploration du graphe de voisinage est la suivante : partant d'une planification $\mathcal{P}$, on se déplace vers la meilleure planification voisine $\mathcal{P}'$. Ce critère de choix autorisant une dégradation de la fonction objectif, il semble capable de permettre d'éviter que l'algorithme ne reste piégé dans un optimum local. Il existe cependant un risque que la méthode *cycle*. En effet, ayant quitté un optimum local pour une moins bonne solution et, compte tenu du fait que l'on choisit la meilleure solution voisine, il est probable que l'on revienne sur nos pas à l'itération suivante. Pour contourner ce problème, il suffirait de conserver les dernières solutions visitées et d'interdire d'y retourner pendant quelques itérations. Le cyclage de la méthode serait ainsi évité et la recherche serait dirigée vers des régions encore non explorées du graphe de voisinage. Cependant, comme il est généralement impossible, pour une question d'espace mémoire, de stocker un ensemble de planifications interdites, on va se limiter à mémoriser le *mouvement* inverse de celui qui nous a permis de passer de $\mathcal{P}$ à $\mathcal{P}'$. Plus précisément, si le passage de la planification $\mathcal{P}$ à la planification $\mathcal{P}'$ a été produit par le

déplacement de l'enfant c du bus b dans le bus b', le mouvement inverse (c, b) est introduit dans une liste de taille fixe, nommée *liste tabou*. Les mouvements de cette liste sont interdits pendant ζ itérations (où ζ est un paramètre d'optimisation appelé *longueur* de la liste tabou).

Mémoriser dans la liste tabou uniquement des mouvements et non des planifications conduit non seulement à l'interdiction de revenir vers des planifications récemment visitées, mais aussi à celle d'atteindre un ensemble de planifications n'ayant pas été considérées jusqu'ici. En effet, comme cela a déjà été mentionné (voir p. 25), le couple (enfant, bus) ne suffit pas à décrire une planification. Pour corriger ce défaut, il est primordial de définir un critère permettant d'accepter tout de même, dans certains cas, un mouvement tabou. On parle alors de critère d'*aspiration*. La fonction d'aspiration la plus simple et la plus couramment utilisée consiste à lever le statut tabou d'un mouvement s'il permet de visiter une planification meilleure que la meilleure planification rencontrée jusqu'alors.

Partant d'une planification $\mathcal{P}$ (admissible ou violant éventuellement la contrainte (2.16)), on se déplace donc vers la meilleure planification voisine. Cependant, pour des raisons d'efficacité, la détermination de la meilleure solution voisine ne se fait pas sur le voisinage entier de $\mathcal{P}$, mais uniquement sur une partie de celui-ci, *i.e.* un *sous-voisinage*. Plus concrètement, on ne va évaluer que ν planifications voisines (choisies aléatoirement) pour déterminer celle qui est la meilleure. Avec probabilité φ, on se déplace donc vers la meilleure planification voisine $\mathcal{P}'$ du sous-voisinage non admissible $\mathcal{N}'_i(\mathcal{P}) \subseteq \mathcal{N}_i(\mathcal{P})$ et avec probabilité $(1-\varphi)$ vers la meilleure planification voisine $\mathcal{P}'$ du sous-voisinage admissible $\mathcal{N}'_f(\mathcal{P}) \subseteq \mathcal{N}_f(\mathcal{P})$, avec $|\mathcal{N}'_f(\mathcal{P})| = |\mathcal{N}'_i(\mathcal{P})| = \nu$. Initialement, la probabilité φ de choix du voisinage $\mathcal{N}_i$ est fixée à $\varphi_0 \in [0, 1)$. Lorsqu'une planification non admissible est visitée, la valeur du paramètre φ est diminuée : plus précisément, φ est multiplié par un facteur constant $\psi \in (0, 1)$. Lorsque l'on retombe sur une planification admissible, la valeur de φ est réinitialisée à φ_0. Dans le cas où $\varphi_0 = 0$, seules des planifications admissibles sont visitées, la recherche tabou n'utilise que le voisinage $\mathcal{N}_f$ et l'on se trouve en présence d'une «recherche tabou admissible». Dans le cas contraire, des planifications non admissibles sont également visitées, la recherche tabou utilise alors les deux voisinages $\mathcal{N}_f$ et $\mathcal{N}_i$ et il s'agit d'une «recherche tabou non admissible».

Lorsqu'il n'y a pas eu d'amélioration de la meilleure planification admissible rencontrée parmi les dernières $\eta_{\max}$ planifications voisines évaluées, la recherche tabou se termine. À nouveau, on applique la procédure Lin-2-opt (voir l'algorithme 3.8) à la meilleure planification trouvée.

La description générale de la recherche tabou développée dans ce travail se trouve dans l'algorithme 3.9. Les paramètres d'optimisation sont :

- ▷ la taille ν des sous-voisinages considérés ;
- ▷ la longueur ζ de la liste tabou ;
- ▷ le coefficient κ_0 de pénalité associé à la fonction objectif $\mathcal{F}$;
- ▷ le paramètre λ de la loi de probabilité (3.4) pour le choix de l'enfant à déplacer ;
- ▷ la probabilité φ_0 de choisir un voisin dans le voisinage non admissible $\mathcal{N}_i$;
- ▷ le coefficient ψ de diminution de la probabilité de choisir le voisinage non admissible ;
- ▷ le nombre maximal η_{max} d'évaluations de planifications consécutives sans amélioration de la meilleure planification rencontrée.

Les résultats des tests effectués à l'aide de cette heuristique sont présentés au chapitre 4.

Algorithme 3.9 RechercheTabou

Donnée : $\{\mathcal{T}_b\}_{b=1}^{B}$, ν, ζ, κ_0, λ, φ_0, ψ, η_{max}, $\mathcal{F} \in \{\mathcal{F}_1, \mathcal{F}_2\}$

$\eta \leftarrow 0$, $\mathcal{L} \leftarrow (\ )$, $\kappa \leftarrow \kappa_0$, $\varphi \leftarrow \varphi_0$, $\mathcal{P} \leftarrow \{\mathcal{T}_b\}_{b=1}^{B}$, $\mathcal{P}^* \leftarrow \mathcal{P}$

Tant que $\eta < \eta_{max}$ **faire**

Si $\varphi \leq (x \sim U[0,1))$ **alors**

$\mathcal{N} \leftarrow \mathcal{N}_i$

Sinon

$\mathcal{N} \leftarrow \mathcal{N}_f$

Pour $i \in \{1, \ldots, \nu\}$ **faire**

Choisir aléatoirement un enfant c_i selon la loi (3.4) et uniformément un bus $b_i \in \overline{\mathcal{B}}^{\mathcal{P}}(c_i)$.

Soit $\mathcal{P}'_i$ la planification voisine de $\mathcal{P}$ obtenue en insérant c_i dans b_i (selon $\mathcal{N}$).

$i^* \leftarrow \text{argmin}_{i \in \{1,\ldots,\nu\}}\{\mathcal{F}(\mathcal{P}'_i)\}$ (∗ meilleure planification voisine ∗)

$\mathcal{P}' \leftarrow \mathcal{P}'_{i^*}$, $c \leftarrow c_{i^*}$, $b \leftarrow b_{i^*}$

Si $\mathcal{P}'$ est admissible **et** $\mathcal{F}(\mathcal{P}') < \mathcal{F}(\mathcal{P}^*)$ **alors** (∗ meilleure planification rencontrée ∗)

$\mathcal{P}^* \leftarrow \mathcal{P}'$, $\mathcal{P} \leftarrow \mathcal{P}'$, $\eta \leftarrow 0$ (∗ avec éventuellement la fonction d'aspiration ∗)

Sinon

$\eta \leftarrow \eta + \nu$ (∗ ν planifications voisines évaluées ∗)

Si $(c, b) \notin \mathcal{L}$ **alors** (∗ mouvement non tabou ∗)

$\mathcal{P} \leftarrow \mathcal{P}'$

Si $\mathcal{P}'$ est admissible **alors**

$\kappa \leftarrow \kappa_0$, $\varphi \leftarrow \varphi_0$

Sinon

$\kappa \leftarrow \kappa \cdot \kappa_0$, $\varphi \leftarrow \psi \cdot \varphi$

$\mathcal{L} \leftarrow \mathcal{L} + (c, b)$ (∗ mise à jour de la liste tabou ∗)

Si $|\mathcal{L}| > \zeta$ **alors**

effacer le plus ancien élément de $\mathcal{L}$.

Retourner Lin-2-opt$(\mathcal{P}^*)$ (∗ optimisation ∗)

3.2.6. Complexité des heuristiques

La complexité des heuristiques dépend essentiellement de l'effort à fournir pour déterminer une planification voisine $\mathcal{N}(\mathcal{P})$ de $\mathcal{P}$. Comme nous l'avons déjà relevé (voir p. 25), dans le pire des cas $O(|\mathcal{C}|^3)$ opérations sont nécessaires pour la construction d'une planification voisine. Le nombre de planifications à évaluer étant aléatoire (mais supérieur ou égal à η_{max}), on ne peut estimer le nombre d'opérations nécessaires pour déterminer une planification de bonne qualité.

CHAPITRE 4

Application des algorithmes

Dans ce chapitre, nous décrivons les instances que nous avons considérées, l'évolution des performances de nos heuristiques au cours du temps ainsi que leurs temps d'exécution. Pour comparer l'efficacité des heuristiques en question, la notion de *profil de performance* est utilisée. Quelques résultats concernant le problème *retour* sont également présentés.

4.1. Présentation des instances testées

Afin d'analyser l'efficacité des heuristiques développées, nous les avons testées pour les deux problèmes de planification de tournées de véhicules scolaires énoncés dans le chapitre 2. La seule différence existant entre ces deux problèmes est la fonction objectif $\mathcal{F} \in \{\mathcal{F}_1, \mathcal{F}_2\}$. Rappelons que la fonction objectif $\mathcal{F}_1$ (voir (3.5)) vise à minimiser la somme des pertes de temps des enfants, alors que $\mathcal{F}_2$ (voir (3.6)) a pour but de minimiser le maximum des pertes de temps des enfants. Une instance de chacun de ces deux problèmes est définie notamment par un réseau $G = (\mathcal{V}, \mathcal{E}, \ell)$, un ensemble d'écoles $\mathcal{S}$, un ensemble d'enfants $\mathcal{C}$ à transporter d'une maison à une école, ainsi qu'un ensemble de bus (de différents types) à disposition. Les instances testées, construites sur des données réelles ou à partir de données fictives, sont décrites ci-dessous.

4.1.1. Données réelles

La première instance réelle considérée, que l'on référencera plus loin par $R_{34}(4)$, correspond au problème de Savigny et de Forel, deux communes vaudoises. Le réseau en question, représenté sur la figure 4.1, contient 34 sommets, comprenant 26 maisons et 12 écoles : quatre écoles enfantines dont l'horaire est 8h30 - 11h10, six écoles primaires dont l'horaire est 8h10 - 11h35 et deux écoles secondaires dont l'horaire est 7h35 - 12h05. Remarquons que les écoles ne sont positionnées qu'en 8 lieux géographiques. Le diamètre du graphe, *i.e.* le temps de trajet maximum entre deux sommets, est de 18 minutes. Pour l'année scolaire 1997-1998, quatre bus ont été utilisés pour transporter 274 enfants réels (79 pour l'école enfantine, 162 pour l'école primaire et 33 pour l'école secondaire), qui correspondent dans notre modélisation à 91 (groupes d') enfants (21 pour l'école enfantine, 54 pour l'école primaire et 16 pour l'école secondaire). La capacité de chaque bus est de 210 unités, chaque enfant en secondaire occupant 10 unités, contre 7 en enfantine ou en primaire. En d'autres termes, chaque bus peut transporter 21 enfants de secondaire ou 30 d'enfantine ou de primaire. Cette différence s'explique par l'utilisation de véhicules contenant des bancs et non des sièges individuels.

La seconde instance réelle considérée, que l'on appellera ci-dessous $R_{151}(11)$, nous a été soumise par l'École Nouvelle de la Suisse Romande, une école privée de Chailly (quartier de Lausanne). Le réseau correspondant, représenté sur la figure 4.2 contient 151 sommets, incluant 97 maisons et 2 écoles : une école enfantine dont l'horaire est 9h00 - 12h35 et une école primaire dont l'horaire est 8h20 - 12h35. Ces deux écoles se situent au même endroit. Le diamètre du graphe est de 59 minutes. Pour l'année scolaire 2002-2003, onze bus ont été utilisés pour transporter 146 enfants réels (68 pour l'école enfantine et 78 pour l'école primaire), qui correspondent dans notre

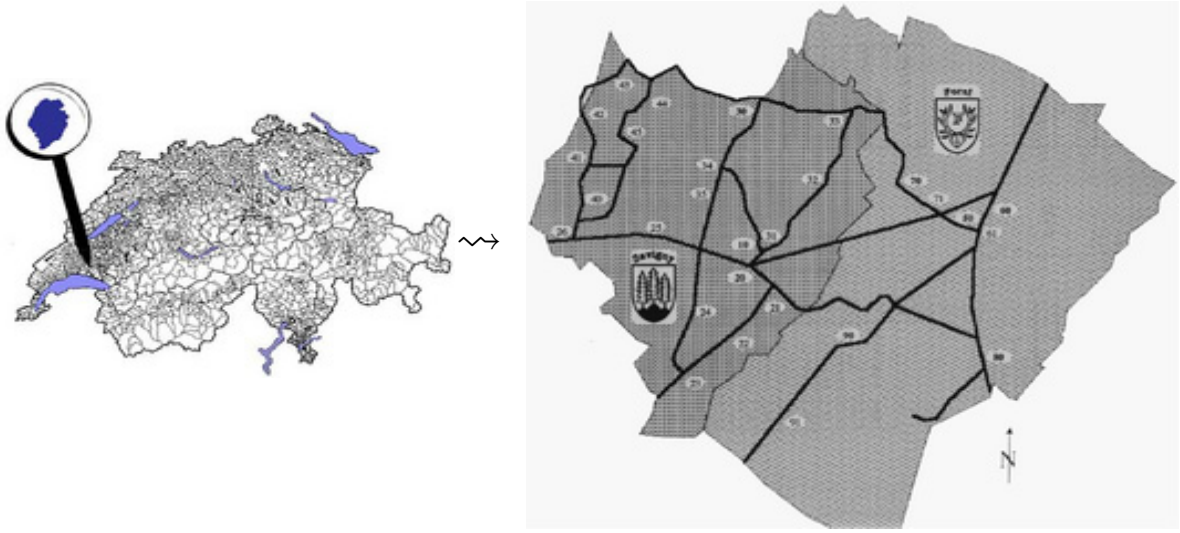


FIG. 4.1 – Réseau de Savigny et de Forel.

modélisation à 107 (groupes d') enfants (50 pour l'école enfantine et 57 pour l'école primaire). Parmi les bus utilisés, on en compte un d'une capacité de 4 unités, un d'une capacité de 7 unités, un d'une capacité de 8 unités, deux d'une capacité de 13 unités, quatre d'une capacité de 14 unités et deux d'une capacité de 15 unités. Chaque enfant, qu'il soit en classe enfantine ou primaire, occupe une unité dans chacun des bus à disposition.

FIG. 4.2 – Lieux desservis par l'École Nouvelle de la Suisse Romande.

Les caractéristiques de ces deux instances réelles sont résumées dans la table 4.1.

				Enfants en classe			
	$\lvert\mathcal{H}\rvert$	$\lvert\mathcal{S}\rvert$	$\lvert\mathcal{C}\rvert$	enfantine	primaire	secondaire	$N_{\mathcal{C}}$
$R_{34}(4)$	26	12	91	21	54	16	274
$R_{151}(11)$	97	2	107	50	57	0	146

TAB. 4.1 – Caractéristiques des instances réelles.

4.1.2. Données fictives

Dans cette section, une méthode permettant de générer des données fictives ressemblant à des données réelles est proposée. Les paramètres considérés pour caractériser et construire de telles instances sont :

- ▷ le réseau, décrit par le nombre de sommets et leur degré maximum ;
- ▷ la vitesse autorisée sur le réseau, *i.e.* le temps nécessaire pour effectuer un nombre de mètres donné ;
- ▷ les types d'écoles, leur nombre et leurs positions ;
- ▷ le nombre d'enfants ainsi que leurs maisons et leurs écoles ;
- ▷ les types de véhicules disponibles et leur nombre.

Le réseau : soit V le nombre de sommets désiré. Les V sommets sont générés uniformément dans un carré de dimensions $[0, V/\xi] \times [0, V/\xi]$, où le paramètre ξ permet d'obtenir des distances euclidiennes correspondant à des temps de parcours (en minutes) proches de la réalité. Les arêtes du réseau sont construites à partir d'une triangulation de Delaunay (voir [PS85]) sur l'ensemble des sommets. On note $G = (\mathcal{V}, \mathcal{E}, \ell)$ le réseau obtenu, où $\mathcal{V}$ est l'ensemble des sommets, avec $|\mathcal{V}| = V$, $\mathcal{E}$ est l'ensemble des arêtes et ℓ est la fonction associant à chaque arête le temps de trajet pour la parcourir (ce temps correspond à la distance euclidienne). Notons que l'on suppose que chaque arête peut être parcourue dans les deux sens. Tant que le réseau contient un sommet $v \in \mathcal{V}$ de degré supérieur à un certain paramètre $\delta \in \mathbb{N}^*$, une arête incidente à ce sommet est retirée. Cette arête est choisie aléatoirement selon la loi de probabilité :

$$P(\text{choisir } e \in \mathcal{E}(v)) = \frac{\ell(e)}{\sum_{e \in \mathcal{E}(v)} \ell(e)}, \tag{4.1}$$

où $\mathcal{E}(v)$ désigne l'ensemble des arêtes incidentes au sommet $v \in \mathcal{V}$. En outre, en utilisant la même loi de probabilité (4.1), un nombre aléatoire d'arêtes supplémentaires sont retirées, tout en veillant à ce que le degré de tout sommet reste strictement supérieur à zéro. Ce processus d'élimination d'arêtes est décrit dans l'algorithme 4.1.

Au terme de l'application de l'algorithme 4.1, le réseau obtenu n'est plus forcément connexe. Soit $\mathcal{K} = \{k_1, \dots, k_K\}$ l'ensemble des composantes connexes du réseau, où $k_i \subset \mathcal{V}$. On définit la *distance minimale entre deux composantes connexes* k_i et k_j par

$$D(k_i, k_j) = \min_{v_i \in k_i, v_j \in k_j} d(v_i, v_j).$$

Tant que le réseau n'est pas connexe, on ajoute une arête de distance minimale permettant de relier deux composantes. Ce processus est décrit dans l'algorithme 4.2.

Choix des écoles : soit $\mathcal{M} = \{m_1, \dots, m_M\}$ l'ensemble des types d'écoles (par exemple, enfantine, primaire et secondaire). À chaque type d'école $m_i \in \mathcal{M}$, on associe une proportion p_i de sommets du réseau correspondant à une école de ce type. Ainsi pour tout $i \in \{1, \dots, M\}$,

Algorithme 4.1 Élimination aléatoire d'arêtes

Donnée : $G = (\mathcal{V}, \mathcal{E}, \ell)$, $\delta \in \mathbb{N}^*$ le degré maximum des sommets

Résultat : $G = (\mathcal{V}, \mathcal{E}, \ell)$ non forcément connexe

Pour tout $v \in \mathcal{V}$ **faire**

Tant que $deg(v)$[1] $> \delta$ **faire** (∗ réduction du degré à une valeur $\leq \delta$ ∗)

Choisir aléatoirement une arête $e \in \mathcal{E}(v)$ selon la loi (4.1).

$\mathcal{E} \leftarrow \mathcal{E} \setminus \{e\}$

Tant que $deg(v) > (x \sim U[1, \delta])$ **faire** (∗ retrait d'arêtes supplémentaires ∗)

Choisir aléatoirement une arête $e \in \mathcal{E}(v)$ selon la loi (4.1).

$\mathcal{E} \leftarrow \mathcal{E} \setminus \{e\}$

Retourner $G = (\mathcal{V}, \mathcal{E}, \ell)$

Algorithme 4.2 Rétablissement de la connexité

Donnée : $G = (\mathcal{V}, \mathcal{E}, \ell)$, $\mathcal{K} = \{k_1, \ldots, k_K\}$ l'ensemble des composantes connexes de G

Tant que $|\mathcal{K}| > 1$ **faire**

$(i, j) \leftarrow \underset{i,j \in \{1,\ldots,|\mathcal{K}|\}, i \neq j}{\operatorname{argmin}} D(k_i, k_j)$ (∗ deux composantes les plus proches ∗)

$[v_r, v_s] \leftarrow \underset{v_r \in k_i, v_s \in k_j}{\operatorname{argmin}} d(v_r, v_s)$ (∗ deux sommets les plus proches ∗)

$\mathcal{E} \leftarrow \mathcal{E} \cup \{[v_r, v_s]\}$, $k_\ell \leftarrow k_i \cup k_j$, $\mathcal{K} \leftarrow \mathcal{K} \setminus \{k_i, k_j\} \cup \{k_\ell\}$

Retourner $G = (\mathcal{V}, \mathcal{E}, \ell)$

$\lceil p_i \cdot |\mathcal{V}| \rceil$ sommets sont choisis aléatoirement, afin d'y placer une école de type m_i, tout en veillant à ce que deux écoles de même type ne soient pas situées côte à côte. Ce procédé est détaillé dans l'algorithme 4.3.

Placement des enfants : chaque sommet v du réseau qui n'est pas une école peut *a priori* être considéré comme une maison. Pour chaque paire maison-type d'école, on détermine le nombre d'enfants habitant dans cette maison et devant se rendre dans une école du type en question. Ce nombre est obtenu en tirant uniformément une valeur dans une liste d'observations réelles Γ_i associée à un type d'école $m_i \in \mathcal{M}$. Une liste Γ_i contient toutes les valeurs observées pour les différentes maisons d'un réseau réel, chaque valeur correspondant au nombre d'enfants d'une maison se rendant dans une école de type m_i. Notons que toute liste Γ_i peut contenir la valeur 0 (s'il existe une maison, dans le réseau réel observé, à partir de laquelle aucun enfant ne se rend dans une école de type m_i) et qu'une même valeur peut y apparaître plusieurs fois. Après avoir déterminé le nombre d'enfants de chaque niveau scolaire habitant dans chacune des maisons, on

[1] $deg(v)$ est le degré du sommet v, *i.e.* le nombre d'arêtes incidentes à v (rappelons que le réseau est supposé simple).

Algorithme 4.3 Placement des écoles sur le réseau

Donnée : $G = (\mathcal{V}, \mathcal{E}, \ell)$, $\mathcal{M} = \{m_1, \dots, m_M\}$ l'ensemble des types d'écoles, p_i la proportion d'écoles de type m_i, pour $i \in \{1, \dots, M\}$

Résultat : $\mathcal{S}_i$ l'ensemble des écoles de type m_i, pour tout $i \in \{1, \dots, M\}$

Pour tout $i \in \{1, \dots, M\}$ **faire**

$\mathcal{V}' \leftarrow \mathcal{V}$

Pour j **de** 1 **à** $\lceil p_i \cdot |\mathcal{V}| \rceil$ **faire**

Choisir uniformément un sommet v dans $\mathcal{V}'$.

$\mathcal{S}_i \leftarrow \mathcal{S}_i \cup \{v\}$, $\mathcal{V}' \leftarrow \mathcal{V}' \setminus \{v, Adj(v)^1\}$ (∗ pas côte à côte ∗)

Si $\mathcal{V}' = \varnothing$ **alors**

$j \leftarrow \lceil p_i \cdot |\mathcal{V}| \rceil$ (∗ passer au type d'école suivant ∗)

Retourner $\mathcal{S}_i$ pour tout $i \in \{1, \dots, M\}$

attribue à tout enfant l'école la plus proche de son lieu d'habitation. Ce processus est détaillé dans l'algorithme 4.4.

Algorithme 4.4 Placement des enfants

Donnée : $\mathcal{S}_i$ l'ensemble des écoles de type m_i, Γ_i la liste des observations réelles de nombres d'enfants se rendant d'une maison à une école de type m_i, $i \in \{1, \dots, M\}$

$\mathcal{S} \leftarrow \bigcup_{i=1}^{M} \mathcal{S}_i$, $\mathcal{H} \leftarrow \varnothing$, $\mathcal{C} \leftarrow \varnothing$, $N_{\mathcal{C}} \leftarrow 0$

Pour tout $m_i \in \mathcal{M}$ **faire**

$\mathcal{V}' \leftarrow \mathcal{V} \setminus \mathcal{S}$ (∗ aucun enfant n'habite dans une école ∗)

Pour tout $v \in \mathcal{V}'$ **faire** (∗ pour toute maison ∗)

Choisir uniformément un nombre γ d'enfants dans la liste Γ_i.

Si $\gamma \neq 0$ **alors**

Soit c un nouvel enfant avec

$h(c) \leftarrow v$, $s(c) \leftarrow \operatorname{argmin}_{s \in S_i}\{d(v, s)\}$, $n(c) \leftarrow \gamma$ (∗ école la plus proche ∗)

$\mathcal{H} \leftarrow \mathcal{H} \cup \{v\}$, $\mathcal{C} \leftarrow \mathcal{C} \cup \{c\}$, $N_{\mathcal{C}} \leftarrow N_{\mathcal{C}} + n(c)$ (∗ ensemble des enfants ∗)

Retourner $\mathcal{H}$, $\mathcal{S}$, $\mathcal{C}$, $N_{\mathcal{C}}$

[1] $Adj(v)$ est l'ensemble des sommets adjacents au sommet v.

Données générées : pour la génération des données fictives, les divers paramètres ont été fixés de manière à s'approcher de certaines caractéristiques de l'instance réelle $R_{34}(4)$. Ainsi, les valeurs suivantes ont été retenues :

- ▷ $\xi = 1.8$;
- ▷ le degré maximum des sommets $\delta = 5$;
- ▷ l'ensemble des types d'écoles $\mathcal{M} = \{$enfantine, primaire, secondaire$\}$;
- ▷ les proportions d'écoles de chaque type $(p_1, p_2, p_3) = (1/10, 1/5, 1/15)$;
- ▷ $\Gamma_1 = (0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 2, 2, 3, 3, 3, 4, 6, 6, 6, 9, 13, 15)$;
- ▷ $\Gamma_2 = (0, 1, 1, 2, 2, 2, 2, 2, 3, 4, 4, 5, 6, 6, 7, 7, 8, 10, 10, 10, 14, 17, 19, 20)$;
- ▷ $\Gamma_3 = (0, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 3, 3, 8)$.

De plus, on supposera qu'un enfant d'école enfantine ou primaire occupe 7 unités dans un bus, alors qu'un enfant de secondaire en occupe 10. Les bus utilisés ont une capacité de 210 unités et peuvent donc transporter 30 enfants d'enfantine ou de primaire et 21 enfants de secondaire.

Ayant fixé les paramètres ci-dessus, il ne reste qu'à choisir le nombre de sommets V et le nombre de bus B. Cinq jeux de données fictives correspondant à des réseaux constitués de 11, 20, 32, 52 et 105 sommets, nommés F_{11}, F_{20}, F_{32}, F_{52} et F_{105}, ont été générés. Les caractéristiques de ces jeux de données sont reportées dans la table 4.2. En guise d'illustration, le réseau du jeu de données F_{52} est représenté sur la figure 4.3.

				Enfants en classe			
	$\lvert\mathcal{H}\rvert$	$\lvert\mathcal{S}\rvert$	$\lvert\mathcal{C}\rvert$	enfantine	primaire	secondaire	$N_{\mathcal{C}}$
F_{11}	10	4	17	6	5	6	86
F_{20}	16	8	32	19	14	8	111
F_{32}	25	11	47	15	20	12	250
F_{52}	47	19	102	35	39	28	570
F_{105}	95	37	190	60	78	52	916

TAB. 4.2 – Caractéristiques des jeux de données fictives.

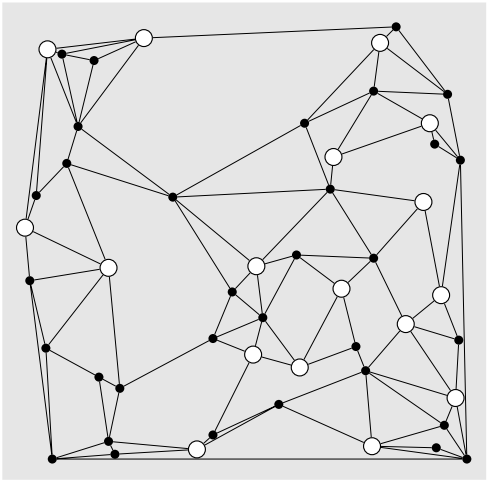


FIG. 4.3 – Réseau fictif de 52 sommets où les écoles sont représentées en blanc.

4.1.3. Instances considérées

Pour chacun des deux problèmes de planification de tournées de véhicules scolaires, *i.e.* l'un avec la fonction objectif $\mathcal{F}_1$ et l'autre avec la fonction objectif $\mathcal{F}_2$, des instances sur les jeux de données R_{34}, R_{151}, F_{11}, F_{20}, F_{32}, F_{52} et F_{105} ont été construites avec différents nombres de bus à disposition. Dans les résultats présentés ci-dessous, on note $F_V(B)$ l'instance obtenue à partir du jeu de données F_V pour un nombre de bus B. Par exemple, l'instance sur le réseau fictif F_{11} avec deux bus à disposition est notée $F_{11}(2)$. Un total de 18 instances par problème, énumérées dans la table 4.3 ont été considérées.

	F_{11}	F_{20}	F_{32}	F_{52}	F_{105}	R_{34}	R_{151}
B	2 3 4 5	2 3 4 5	3 6	9 13	20 25	4 5 6	11

TAB. 4.3 – Instances sur lesquelles les heuristiques ont été testées.

4.2. Paramètres utilisés

Les heuristiques développées au chapitre 3 ont été utilisées pour générer des planifications efficaces pour nos deux problèmes et pour les diverses instances décrites dans la section 4.1.3. Quatre heuristiques ont été considérées : le recuit simulé admissible, noté Sim $\mathcal{N}_f$, le recuit simulé non admissible, noté Sim $\mathcal{N}_i$, la recherche tabou admissible, notée Tab $\mathcal{N}_f$ et la recherche tabou non admissible, notée Tab $\mathcal{N}_i$.

Les recuits simulés ont été testés avec quatre températures initiales différentes et cinq germes distincts du générateur de nombres pseudo-aléatoires[1]. Les recherches tabou ont été testées avec deux longueurs différentes de liste tabou et dix germes distincts du générateur de nombres pseudo-aléatoires. Par conséquent, chaque heuristique a été exécutée avec vingt jeux de paramètres, pour un total de 2 880 exécutions (2 problèmes $\times$ 18 instances $\times$ 4 heuristiques $\times$ 20 jeux de paramètres) sur un Pentium 4 cadencé à 2GHz. Après quelques tests, les valeurs suivantes des paramètres d'optimisation ont été retenues pour les recuits simulés :

- ▷ température initiale : $T_0 \in \begin{cases} \{250,\ 500,\ 750,\ 1\,000\} & \text{si } \mathcal{F} = \mathcal{F}_1 \\ \{5, 6, 7, 8\} & \text{si } \mathcal{F} = \mathcal{F}_2\,; \end{cases}$
- ▷ coefficient de diminution de la température : $\omega = 0.93$;
- ▷ nombre d'itérations consécutives avec la même température : $\mu = 10$;
- ▷ coefficient de pénalité associé à la fonction objectif $\mathcal{F}$: $\kappa_0 = 1.02$;
- ▷ paramètre de la loi de probabilité (3.4) pour le choix de l'enfant à déplacer : $\lambda = 1$;
- ▷ probabilité de choisir un voisin dans $\mathcal{N}_i$: $\varphi_0 = \begin{cases} 0 & \text{pour Sim } \mathcal{N}_f \\ 0.7 & \text{pour Sim } \mathcal{N}_i\,; \end{cases}$
- ▷ coefficient de diminution de la probabilité φ_0 de choisir le voisinage non admissible : $\psi = 0.95$;
- ▷ nombre maximum d'itérations consécutives pendant lesquelles il est autorisé que la planification courante ne soit pas changée : $\gamma_{max} = 200$;
- ▷ nombre maximum d'évaluations de planifications consécutives sans amélioration de la meilleure planification rencontrée : $\eta_{max} = 7\,500$.

[1] Le générateur utilisé est MRG32k3a de L'Écuyer [L'É99].

De même, les valeurs suivantes des paramètres d'optimisation ont été retenues pour les recherches tabou :

- ▷ taille des sous-voisinages considérés : $\nu = \begin{cases} 10 & \text{si } |\mathcal{V}| \in \{11, 20\} \\ 18 & \text{sinon} \end{cases}$;
- ▷ longueur de la liste tabou : $\zeta \in \{7, 10\}$;
- ▷ coefficient de pénalité associé à la fonction objectif $\mathcal{F}$: $\kappa_0 = 1.02$;
- ▷ paramètre de la loi de probabilité (3.4) pour le choix de l'enfant à déplacer : $\lambda = 1$;
- ▷ probabilité de choisir un voisin dans $\mathcal{N}_i$: $\varphi_0 = \begin{cases} 0 & \text{pour Tab } \mathcal{N}_f \\ 0.7 & \text{pour Tab } \mathcal{N}_i \end{cases}$;
- ▷ coefficient de diminution de la probabilité φ_0 de choisir le voisinage non admissible : $\psi = 0.95$;
- ▷ nombre maximum d'évaluations de planifications consécutives sans amélioration de la meilleure planification rencontrée : $\eta_{max} = 7\,500$.

Les principaux résultats des expériences effectuées sont présentés et analysés dans les sections suivantes.

4.3. Comparaison des algorithmes : profils de performance

Afin d'analyser la qualité globale des heuristiques et de les comparer entre elles, une variante de la méthode des *profils de performance* proposée par Dolan et Moré [DM02] a été développée.

4.3.1. La méthode de Dolan et Moré

Dolan et Moré ont introduit la méthode des *profils de performance* de manière à évaluer et comparer l'efficacité d'algorithmes pour des problèmes d'optimisation continue. Soit $\mathcal{I}$ un ensemble d'instances d'un problème à résoudre et $\mathcal{A}$ un ensemble d'algorithmes d'optimisation locale applicables à ce problème. Pour une instance, chacun de ces algorithmes, partant du même point initial, peut soit converger vers un optimum local (le même pour tous les algorithmes), soit diverger. Supposons que l'on veuille comparer les temps de convergence nécessaires aux différents algorithmes pour les diverses instances. On nomme *indice de performance* le critère que l'on veut comparer, ici $t_{i,a}$, le temps de calcul nécessaire pour que l'algorithme $a \in \mathcal{A}$ converge vers un optimum local pour l'instance $i \in \mathcal{I}$. Si l'algorithme a diverge pour cette instance i, on pose $t_{i,a} = \infty$. Le *coefficient de performance* de l'algorithme a pour la résolution de l'instance i est défini par

$$r_{i,a} = \frac{t_{i,a}}{\min\limits_{\tilde{a} \in \mathcal{A}} \{t_{i,\tilde{a}}\}}.$$

Ce coefficient est le rapport entre l'indice de performance d'un algorithme pour une instance donnée et le meilleur indice (*i.e.* le temps minimum de convergence) sur tous les algorithmes considérés. Notons que cette méthode permet également d'intégrer les échecs (divergence), ce que d'autres moyens de comparaison ne permettent généralement pas de faire (par convention, on pose $\frac{\infty}{\infty} = 1$). Pour tout $\chi \in [1, \infty)$, on définit

$$\rho_a(\chi) = \frac{1}{|\mathcal{I}|} \cdot \big|\{i \in \mathcal{I} \mid r_{i,a} \leq \chi\}\big|,$$

la proportion des instances pour lesquelles le coefficient de performance de l'algorithme a est inférieur au *seuil* χ. En d'autres termes, $\rho_a(\chi)$ est la proportion des instances $i \in \mathcal{I}$ pour lesquelles l'algorithme a converge en au plus $\chi \cdot \min_{\tilde{a} \in \mathcal{A}} \{t_{i,\tilde{a}}\}$ unités de temps. La fonction $\rho_a : [1, \infty) \mapsto [0, 1]$ est appelée *profil de performance* de l'algorithme a. Notons que cette fonction est non

décroissante et que $\rho_a(1)$ est la proportion d'instances de $\mathcal{I}$ pour lesquelles l'algorithme a s'avère au moins aussi efficace (au sens de l'indice de performance choisi) que tous les autres algorithmes de $\mathcal{A}$.

Nous allons adapter cette notion de profil de performance de manière à analyser l'influence du choix des paramètres d'optimisation sur le bon fonctionnement de nos heuristiques et déterminer si une heuristique se démarque de toutes les autres en termes de construction de solutions efficaces pour le problème de la planification de tournées de véhicules scolaires.

4.3.2. Influence des paramètres des heuristiques

D'une manière générale, pour étudier l'impact du choix des paramètres d'une heuristique sur la production de planifications efficaces pour un ensemble d'instances $\mathcal{I}$, il convient de comparer les valeurs des fonctions objectif trouvées par chaque heuristique et chaque jeu de paramètres. L'indice de performance naturel est par conséquent la valeur de la fonction objectif obtenue par une heuristique pour un jeu donné de paramètres d'optimisation, après un nombre fixé d'évaluations de planifications candidates que l'on nomme *budget d'évaluations*. On note

$$\mathcal{A}_a = \{a(\pi_1), \dots, a(\pi_{N_a})\}, \tag{4.2}$$

l'ensemble des N_a algorithmes obtenus en associant, à une même heuristique a, N_a jeux de paramètres d'optimisation distincts $\pi_1, \dots, \pi_{N_a}$. L'indice de performance $f^n_{i,a(\pi_j)}$ de l'heuristique a sur l'instance i avec le j^{e} jeu de paramètres et pour un budget d'évaluations n est défini comme la valeur de la fonction objectif obtenue après n évaluations de planifications par l'algorithme $a(\pi_j)$ sur l'instance $i \in \mathcal{I}$. Le coefficient de performance de l'algorithme $a(\pi_j)$ pour la résolution de l'instance $i \in \mathcal{I}$ avec un budget d'évaluations de valeur n est défini par

$$r^n_{i,a(\pi_j)} = \frac{f^n_{i,a(\pi_j)}}{\min\limits_{k \in \{1,\dots,N_a\}} \{f^n_{i,a(\pi_k)}\}}. \tag{4.3}$$

Soulignons le fait que l'on ne cherche pas à comparer les différentes heuristiques entre elles, mais uniquement à évaluer l'effet des différents jeux de paramètres utilisés sur les planifications produites par une heuristique donnée. Le profil de performance de l'algorithme $a(\pi_j) \in \mathcal{A}_a$ est donc

$$\rho^n_{a(\pi_j)}(\chi) = \frac{1}{|\mathcal{I}|} \cdot \left|\{i \in \mathcal{I} \mid r^n_{i,a(\pi_j)} \leq \chi\}\right|. \tag{4.4}$$

Dans ce cas, $\rho^n_{a(\pi_j)}(1)$ se laisse interpréter comme la proportion des instances pour lesquelles le jeu de paramètres π_j a permis de trouver la meilleure planification (parmi celles obtenues avec un algorithme de $\mathcal{A}_a$) avec l'heuristique a et un budget d'évaluations égal à n.

Pour clarifier la notion de profil de performance, nous allons construire une telle fonction à partir d'un exemple. Dans la figure 4.4(a), les valeurs de fonction objectif des planifications obtenues pour quatre instances par une heuristique avec six jeux de paramètres (par conséquent six algorithmes) pour un budget d'évaluations donné sont reportées. Dans un premier temps, le coefficient de performance pour chaque algorithme et chaque instance est calculé. Cette opération revient à diviser chaque ligne du tableau par la valeur minimale de la ligne. Les coefficients de performance obtenus sont présentés dans la figure 4.4(b).

Pour obtenir le profil de performance de l'algorithme $a(\pi_j)$, il faut, pour toute valeur du seuil $\chi \in [1, \infty)$, calculer la proportion des coefficients de performance de valeur inférieure ou égale à χ. Pour cet exemple, on va discrétiser χ au dixième, c'est-à-dire que l'on va considérer les valeurs de χ dans l'ensemble $\{1, 1.1, 1.2, 1.3, \dots\}$. Dans la table 4.4, on trouve les valeurs des profils de

Instances \ Algorithmes	$a(\pi_1)$	$a(\pi_2)$	$a(\pi_3)$	$a(\pi_4)$	$a(\pi_5)$	$a(\pi_6)$
I_1	21	26	21	22	25	24
I_2	3	3	5	4	4	3
I_3	14	13	15	14	12	15
I_4	10	11	7	7	9	7

(a) Indices de performance $f^n_{i,a(\pi_j)}$

Instances \ Algorithmes	$a(\pi_1)$	$a(\pi_2)$	$a(\pi_3)$	$a(\pi_4)$	$a(\pi_5)$	$a(\pi_6)$
I_1	1	1.24	1	1.05	1.19	1.14
I_2	1	1	1.67	1.33	1.33	1
I_3	1.17	1.08	1.25	1.17	1	1.25
I_4	1.43	1.57	1	1	1.29	1

(b) Coefficients de performance $r^n_{i,a(\pi_j)}$

FIG. 4.4 – Partant des valeurs de la fonction objectif obtenues par l'application des algorithmes de $\mathcal{A}_a$, les coefficients de performance sont calculés en normalisant les indices de performance de chaque instance par la meilleure valeur trouvée (cases blanches).

performance de chaque algorithme en fonction de χ. Par exemple, déterminer $\rho^n_{a(\pi_4)}(1.3)$ revient à effectuer le calcul suivant :

$$\rho^n_{a(\pi_4)}(1.3) = \frac{1}{|\mathcal{I}|} \cdot \left|\{i \in \mathcal{I} \mid r^n_{i,a(\pi_4)} \leq 1.3\}\right| = \frac{1}{4} \cdot \left|\{\mathcal{I}_1, \mathcal{I}_3, \mathcal{I}_4\}\right| = \frac{3}{4}.$$

Les calculs n'ont été effectués que jusqu'à $\chi = 1.7$, car aucun coefficient de performance de la figure 4.4(b) ne dépasse cette valeur. Ainsi, on a $\rho^n_{a(\pi_j)}(\chi) = 1$, pour tout $a(\pi_j) \in \mathcal{A}_a$ et tout seuil $\chi \geq 1.7$. Les graphes des profils de performance sont représentés sur la figure 4.5.

χ	1	1.1	1.2	1.3	1.4	1.5	1.6	1.7
$a(\pi_1)$	2/4	2/4	3/4	3/4	3/4	4/4	4/4	4/4
$a(\pi_2)$	1/4	2/4	2/4	3/4	3/4	3/4	4/4	4/4
$a(\pi_3)$	2/4	2/4	2/4	3/4	3/4	3/4	3/4	4/4
$a(\pi_4)$	1/4	2/4	3/4	3/4	4/4	4/4	4/4	4/4
$a(\pi_5)$	1/4	1/4	2/4	3/4	4/4	4/4	4/4	4/4
$a(\pi_6)$	2/4	2/4	3/4	4/4	4/4	4/4	4/4	4/4

TAB. 4.4 – Valeur de la fonction profil de performance $\rho^n_{a(\pi_j)}(\chi)$ pour chaque algorithme $a(\pi_j)$ et quelques valeurs du seuil χ.

Application à nos problèmes. Pour nos deux problèmes de planification de tournées de véhicules scolaires, une comparaison a été effectuée pour chacun des ensembles d'algorithmes suivants :

$$\begin{aligned}
\mathcal{A}_1 &= \{\text{Sim } \mathcal{N}_f(\pi_1), \ldots, \text{Sim } \mathcal{N}_f(\pi_{20})\}, \\
\mathcal{A}_2 &= \{\text{Sim } \mathcal{N}_i(\pi_1), \ldots, \text{Sim } \mathcal{N}_i(\pi_{20})\}, \\
\mathcal{A}_3 &= \{\text{Tab } \mathcal{N}_f(\pi_1), \ldots, \text{Tab } \mathcal{N}_f(\pi_{20})\}, \\
\mathcal{A}_4 &= \{\text{Tab } \mathcal{N}_i(\pi_1), \ldots, \text{Tab } \mathcal{N}_i(\pi_{20})\}.
\end{aligned}$$

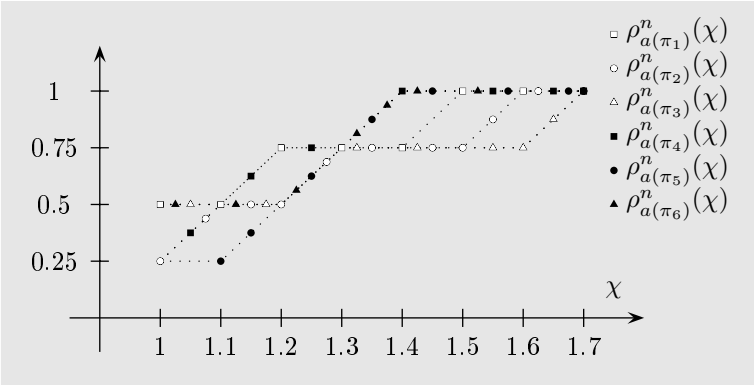
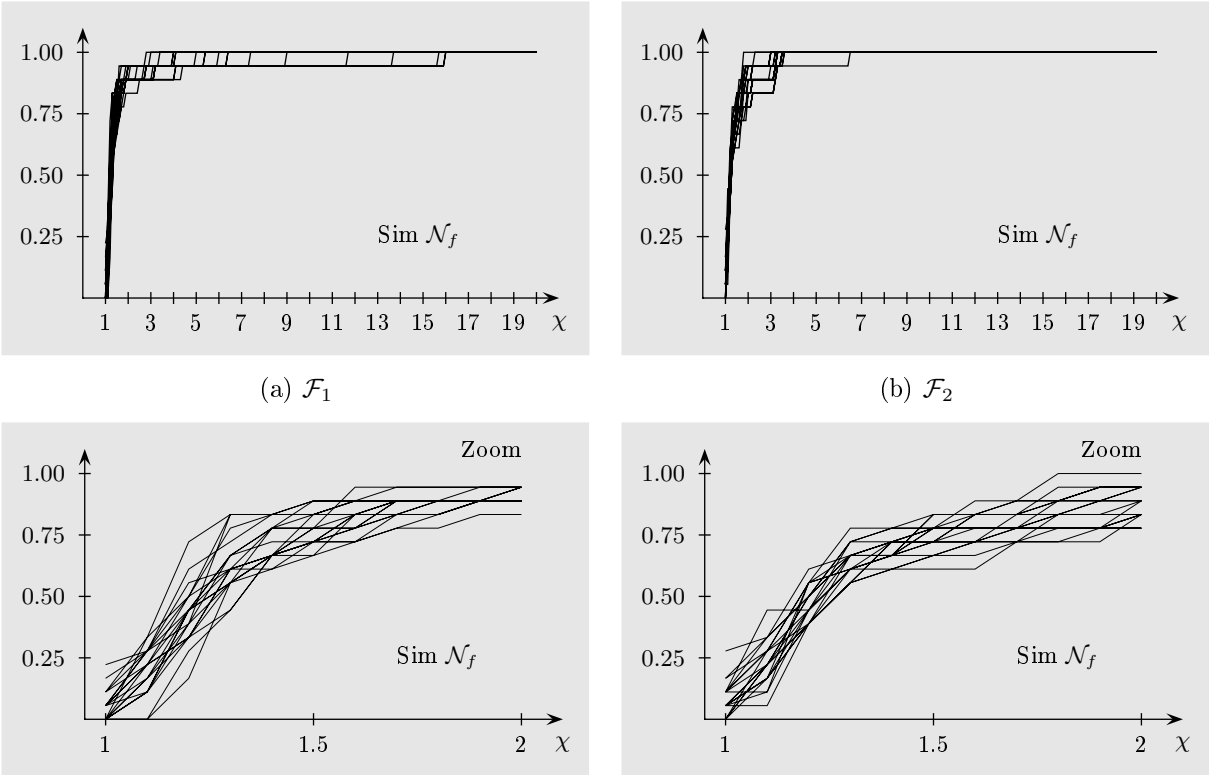


FIG. 4.5 – Profil de performance des algorithmes $a(\pi_1)$, $a(\pi_2)$, $a(\pi_3)$, $a(\pi_4)$, $a(\pi_5)$ et $a(\pi_6)$.

Notons que les jeux de paramètres des heuristiques Sim $\mathcal{N}_f$, Sim $\mathcal{N}_i$, Tab $\mathcal{N}_f$ et Tab $\mathcal{N}_i$ sont naturellement distincts. Ils sont cependant tous notés π_j afin de ne pas alourdir la notation. Les instances considérées sont celles de la section 4.1.3 :

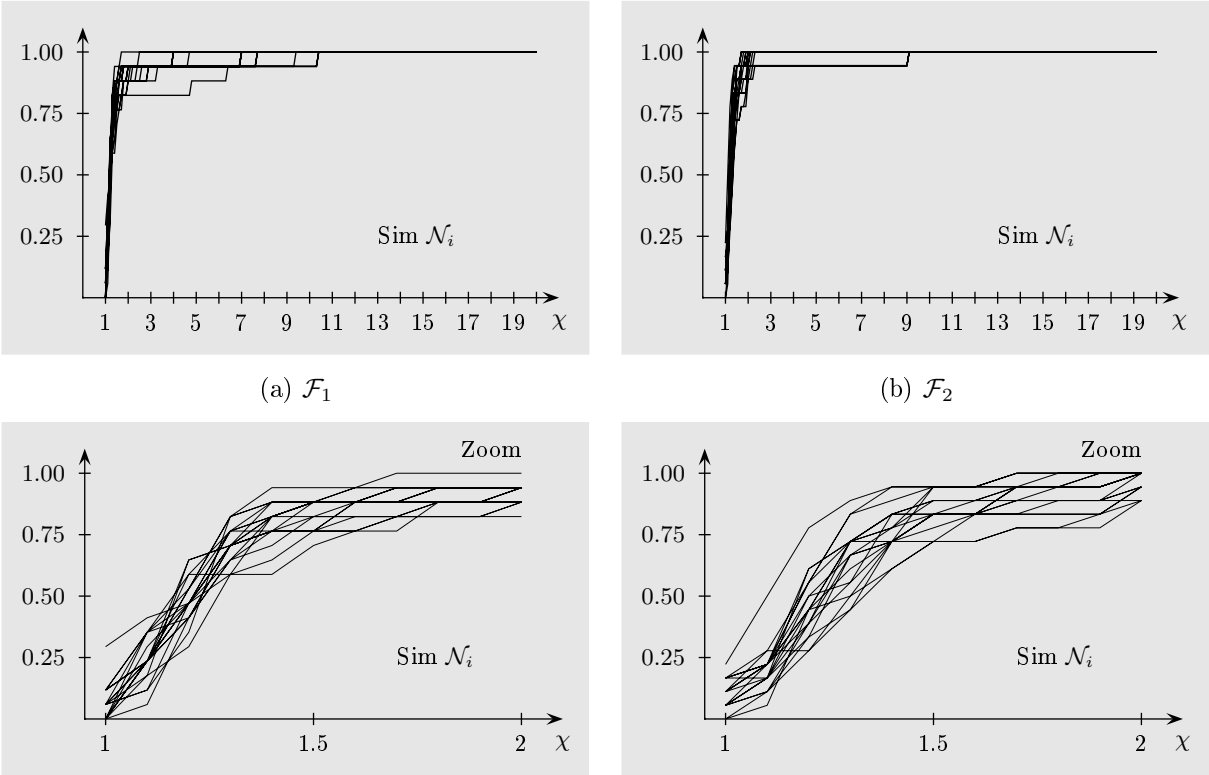
$$\begin{aligned}\mathcal{I} \;=\; &\{F_{11}(2), F_{11}(3), F_{11}(4), F_{11}(5), F_{20}(2), F_{20}(3), F_{20}(4), F_{20}(5), F_{32}(3), F_{32}(6),\\ &F_{52}(9), F_{52}(13), F_{105}(20), F_{105}(25), R_{34}(4), R_{34}(5), R_{34}(6), R_{151}(11)\}.\end{aligned} \tag{4.5}$$

Pour chaque ensemble $\mathcal{A}_1$, $\mathcal{A}_2$, $\mathcal{A}_3$ et $\mathcal{A}_4$ le profil de performance $\rho^n_{a(\pi_j)}(\chi)$ des vingt algorithmes correspondants est représenté dans les figures 4.6 à 4.9, pour un même budget d'évaluations de $n = 30\,000$ planifications visitées. Chaque profil de performance a été généré pour chaque ensemble d'algorithmes et chacune des fonctions objectif $\mathcal{F}_1$ et $\mathcal{F}_2$.



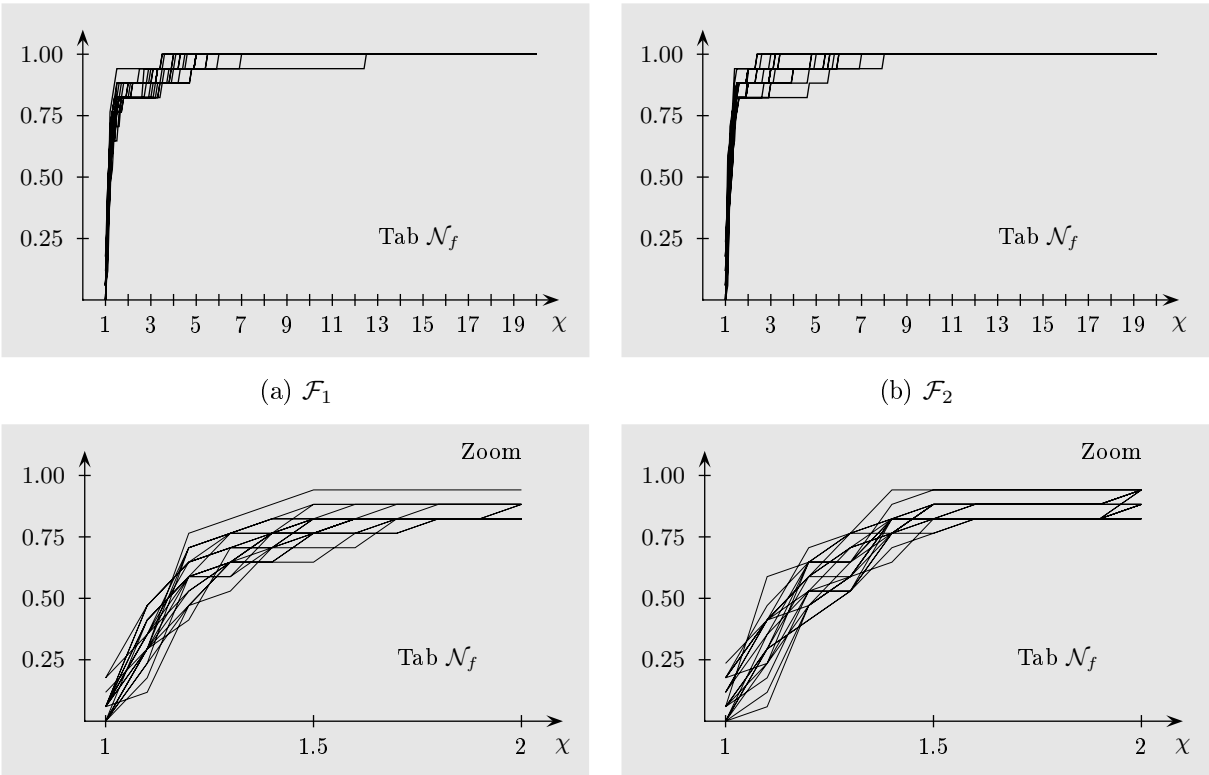
(a) $\mathcal{F}_1$ (b) $\mathcal{F}_2$

FIG. 4.6 – Profils de performance de $\mathcal{A}_1 = \{\text{Sim } \mathcal{N}_f(\pi_1), \ldots, \text{Sim } \mathcal{N}_f(\pi_{20})\}$ pour $n = 30\,000$.



(a) $\mathcal{F}_1$ (b) $\mathcal{F}_2$

FIG. 4.7 – Profils de performance de $\mathcal{A}_2 = \{\text{Sim } \mathcal{N}_i(\pi_1), \ldots, \text{Sim } \mathcal{N}_i(\pi_{20})\}$ pour $n = 30\,000$.



(a) $\mathcal{F}_1$ (b) $\mathcal{F}_2$

FIG. 4.8 – Profils de performance de $\mathcal{A}_3 = \{\text{Tab } \mathcal{N}_f(\pi_1), \ldots, \text{Tab } \mathcal{N}_f(\pi_{20})\}$ pour $n = 30\,000$.

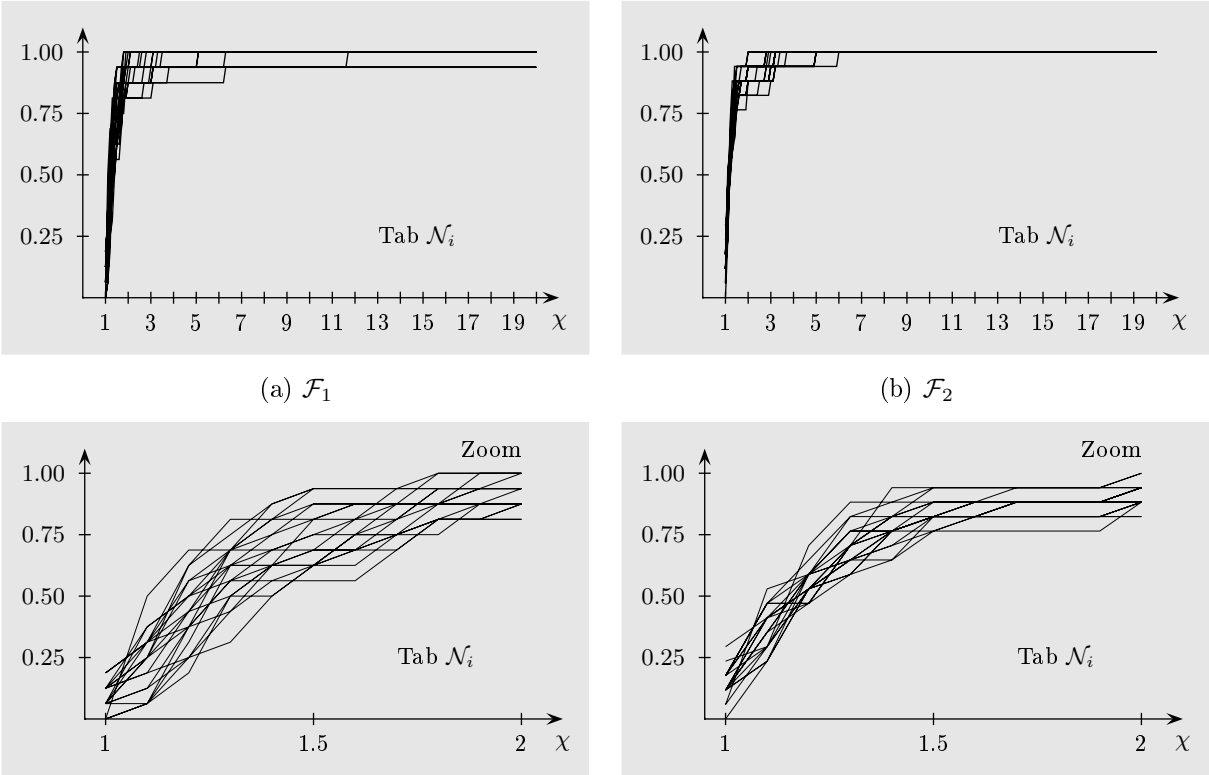


FIG. 4.9 – Profils de performance de $\mathcal{A}_4 = \{\text{Tab } \mathcal{N}_i(\pi_1), \ldots, \text{Tab } \mathcal{N}_i(\pi_{20})\}$ pour $n = 30\,000$.

Dans les figures 4.6 à 4.9, on constate que pour des valeurs de χ prises dans l'intervalle $[1, 1.5]$ la pente des profils de performance est très élevée et que dans la plupart des cas, la valeur de la fonction objectif est à environ au plus deux fois la meilleure valeur trouvée, *i.e.* $\rho_{a(\pi_j)}(2) \approx 1$ pour chaque algorithme. Comme il y a peu de courbes pour lesquelles $\rho_{a(\pi_j)}(1) = 0$, on en conclut qu'il y a peu de jeux de paramètres ne permettant pas d'égaler la meilleure solution trouvée pour au moins une instance de $\mathcal{I}$.

De plus, la plus grande valeur de $\rho_{a(\pi_j)}(1)$ sur tous les algorithmes de $\mathcal{A}_a$, $a \in \{1, \ldots, 4\}$ est de l'ordre de 0.25. Pour aucune heuristique, il n'existe donc de jeu de paramètres permettant de trouver les meilleures planifications sur plus du quart des instances.

On remarque que les graphes des profils de performance de l'heuristique Tab $\mathcal{N}_i$ avec la fonction objectif $\mathcal{F}_1$ (voir figure 4.9(a) zoom) sont plus espacés que pour les autres heuristiques. Ceci révèle une plus grande sensibilité aux paramètres d'optimisation choisis.

Toutefois, pour chaque ensemble d'algorithmes $\mathcal{A}_1$, $\mathcal{A}_2$, $\mathcal{A}_3$ et $\mathcal{A}_4$, on note que les profils de performance obtenus sont relativement semblables. Ainsi, pour une instance donnée, rien ne nous permet de dire qu'un jeu de paramètres, plutôt qu'un autre, va nous permettre de trouver une bonne planification. Il paraît donc raisonnable de faire une moyenne sur les divers jeux de paramètres pour chaque heuristique, plutôt que de comparer chaque algorithme séparément. Une telle moyenne nous permettra de confronter les quatre heuristiques Sim $\mathcal{N}_f$, Sim $\mathcal{N}_i$, Tab $\mathcal{N}_f$ et Tab $\mathcal{N}_i$ entre elles, indépendamment du jeu de paramètres utilisé.

4.3.3. Comparaison de l'efficacité des diverses heuristiques

Supposons que l'on veuille comparer différentes heuristiques réunies dans un ensemble

$$\bar{\mathcal{A}} = \{a_1, a_2, \dots\}.$$

Pour chaque heuristique $a \in \bar{\mathcal{A}}$, rappelons que $\mathcal{A}_a$ désigne l'ensemble des algorithmes obtenus en associant des jeux de paramètres différents à l'heuristique a (voir équation (4.2)). Finalement, on définit

$$\mathcal{A} = \bigcup_{a \in \bar{\mathcal{A}}} \mathcal{A}_a,$$

l'ensemble de tous les algorithmes construits à partir des diverses heuristiques considérées. Comme pour l'analyse précédente, l'indice de performance $f^n_{i,a(\pi_j)}$ est la valeur de la fonction objectif $\mathcal{F}$ obtenue après un budget de n évaluations par l'algorithme $a(\pi_j)$ pour l'instance $i \in \mathcal{I}$. Le coefficient de performance de l'algorithme $a(\pi_j)$ pour la résolution de l'instance $i \in \mathcal{I}$ est défini par

$$r^n_{i,a(\pi_j)} = \frac{f^n_{i,a(\pi_j)}}{\min\limits_{a(\pi) \in \mathcal{A}} \{f^n_{i,a(\pi)}\}}.$$

Cette fois-ci, on compare donc l'indice de performance de l'algorithme $a(\pi_j)$ non seulement avec tous les algorithmes issus de la même heuristique a (*i.e.* l'ensemble $\mathcal{A}_a$) mais également avec tous les autres algorithmes contenus dans l'ensemble $\mathcal{A}$. Pour $\chi \in [1, \infty)$, le profil de performance de l'heuristique $a \in \bar{\mathcal{A}}$ devient naturellement

$$\rho^n_a(\chi) = \frac{1}{|\mathcal{I}| \cdot |\mathcal{A}_a|} \cdot \left| \left\{ (i, a(\pi_j)) \in \mathcal{I} \times \mathcal{A}_a \mid r^n_{i,a(\pi_j)} \leq \chi \right\} \right|. \tag{4.6}$$

Ainsi, $\rho^n_a(1)$ est la proportion des expériences (sur toutes les instances et algorithmes associés à l'heuristique a) pour lesquelles la solution obtenue est au moins aussi bonne que pour tout autre algorithme de $\mathcal{A}$.

Nous allons à nouveau illustrer la construction d'un profil de performance sur un exemple. La figure 4.10(a) contient les valeurs de fonction objectif des planifications obtenues pour quatre instances par deux heuristiques avec chacune trois jeux de paramètres (donc six algorithmes) après un budget d'évaluations donné. À nouveau, le coefficient de performance de chaque instance et de chaque algorithme est calculé. Cette opération revient à diviser chaque ligne du tableau par la valeur minimale de la ligne. Les coefficients de performance obtenus sont présentés dans la figure 4.10(b).

Comme précédemment, on a calculé pour tout χ discrétisé au dixième les valeurs de $\rho^n_a(\chi)$, mais cette fois pour une heuristique $a \in \bar{\mathcal{A}}$. On a par exemple

$$\begin{aligned}
\rho^n_{a_2}(1.1) &= \frac{1}{|\mathcal{I}| \cdot |\mathcal{A}_{a_2}|} \cdot \left| \left\{ (i, a_2(\pi_j)) \in \mathcal{I} \times \mathcal{A}_{a_2} \mid r^n_{i,a_2(\pi_j)} \leq 1.1 \right\} \right| \\
&= \frac{1}{4 \cdot 3} \cdot \left| \left\{ (\mathcal{I}_1, a_2(\pi_1)), (\mathcal{I}_2, a_2(\pi_3)), (\mathcal{I}_3, a_2(\pi_2)), (\mathcal{I}_4, a_2(\pi_1)), (\mathcal{I}_4, a_2(\pi_3)) \right\} \right| \\
&= \frac{5}{12}.
\end{aligned}$$

Toutes ces valeurs sont reportées dans la table 4.5 et le graphe des profils de performance est donné dans la figure 4.11.

Instances	Heuristique a_1			Heuristique a_2		
	$a_1(\pi_1)$	$a_1(\pi_2)$	$a_1(\pi_3)$	$a_2(\pi_1)$	$a_2(\pi_2)$	$a_2(\pi_3)$
I_1	21	26	21	22	25	24
I_2	3	3	5	4	4	3
I_3	14	13	15	14	12	15
I_4	10	11	7	7	9	7

(a) Indices de performance $f^n_{i,a(\pi_j)}$

Instances	Heuristique a_1			Heuristique a_2		
	$a_1(\pi_1)$	$a_1(\pi_2)$	$a_1(\pi_3)$	$a_2(\pi_1)$	$a_2(\pi_2)$	$a_2(\pi_3)$
I_1	1	1.24	1	1.05	1.19	1.14
I_2	1	1	1.67	1.33	1.33	1
I_3	1.17	1.08	1.25	1.17	1	1.25
I_4	1.43	1.57	1	1	1.29	1

(b) Coefficients de performance $r^n_{i,a(\pi_j)}$

FIG. 4.10 – Partant des valeurs de la fonction objectif obtenues par l'application des algorithmes de $\mathcal{A}$, les coefficients de performance sont calculés en normalisant les indices de performance de chaque instance par la meilleure valeur trouvée (cases blanches).

χ	1	1.1	1.2	1.3	1.4	1.5	1.6	1.7
a_1	5/12	6/12	7/12	9/12	9/12	10/12	11/12	12/12
a_2	4/12	5/12	8/12	10/12	12/12	12/12	12/12	12/12

TAB. 4.5 – Valeur de la fonction profil de performance $\rho^n_a(\chi)$ pour les deux heuristiques et quelques valeurs du seuil χ.

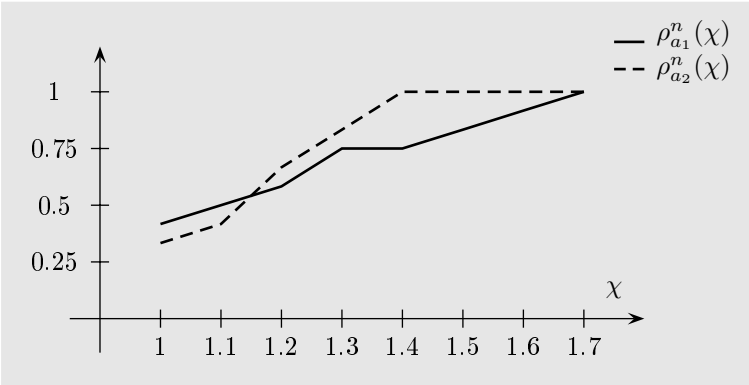
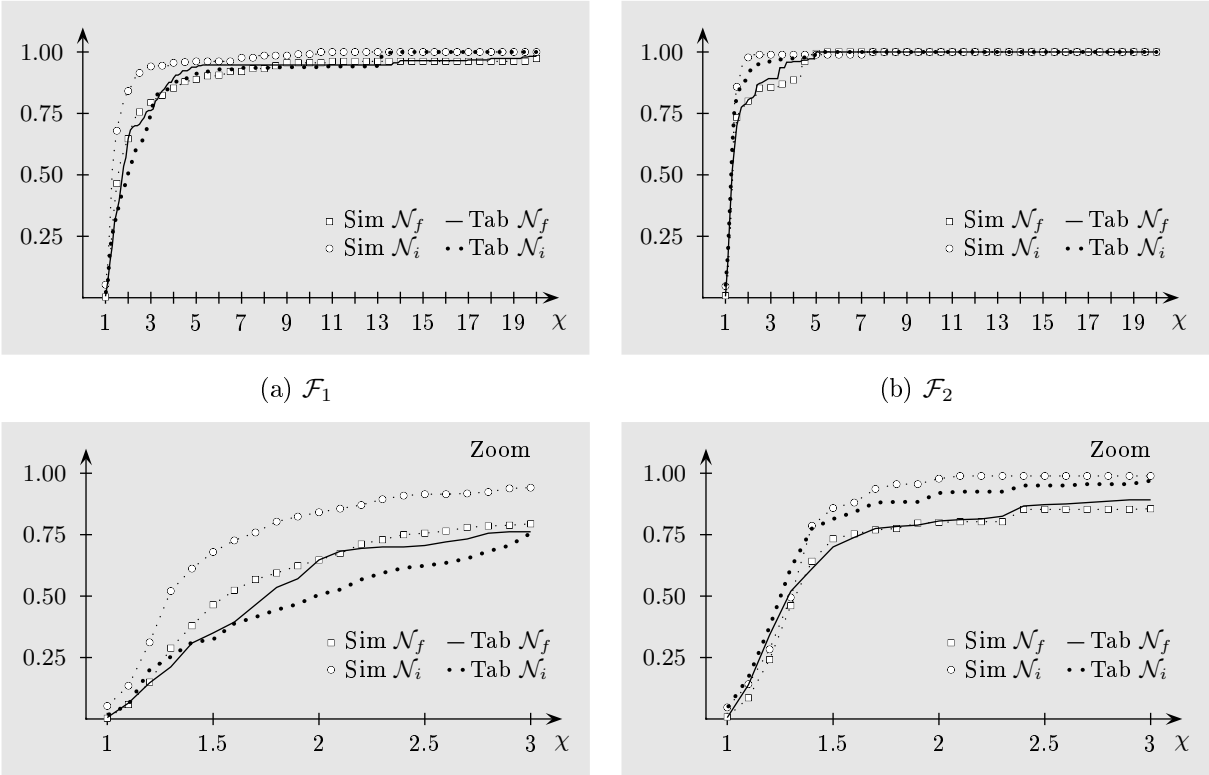


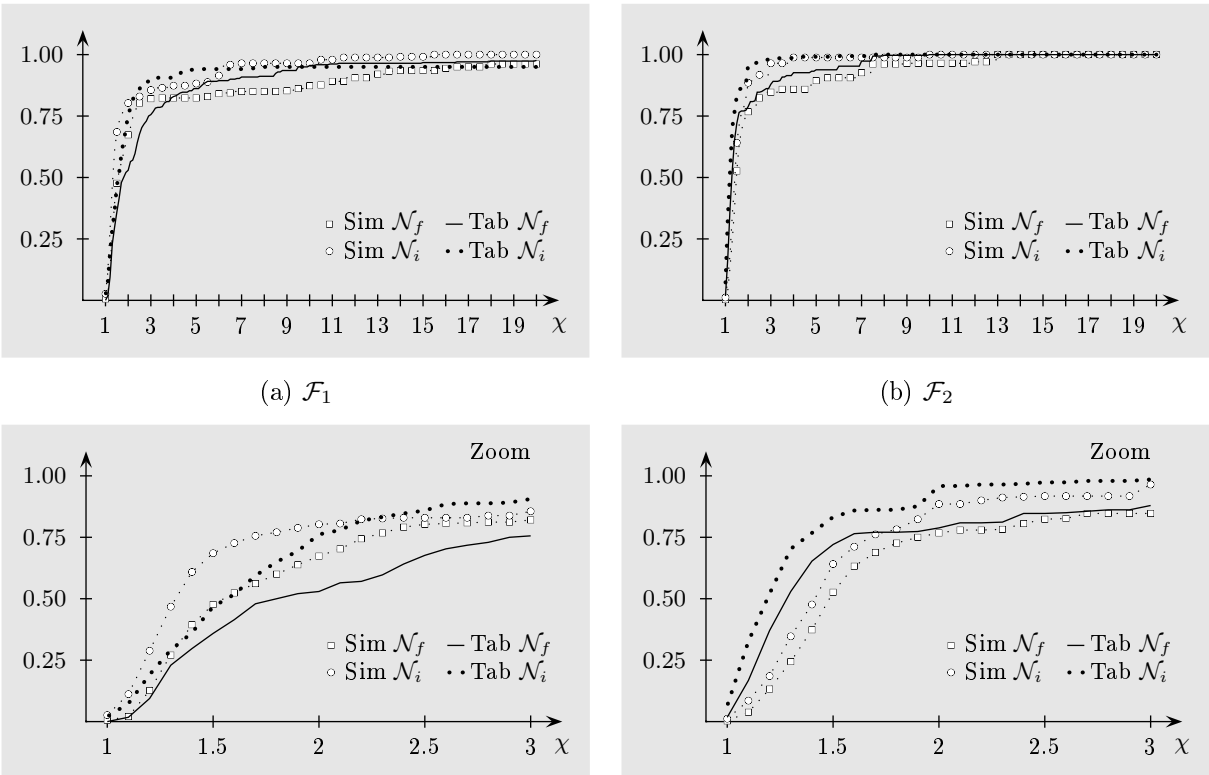
FIG. 4.11 – Profil de performance des heuristiques a_1 et a_2.

Application à nos problèmes. Pour les tests effectués, les quatre heuristiques comparées sont $\bar{\mathcal{A}} = \{\text{Sim } \mathcal{N}_f, \text{Sim } \mathcal{N}_i, \text{Tab } \mathcal{N}_f, \text{Tab } \mathcal{N}_i\}$ et l'ensemble $\mathcal{I}$ des instances considérées est celui défini par (4.5). Sur les figures 4.12 à 4.14, la valeur de $\rho^n_a(\chi)$ est représentée en fonction du seuil χ pour chaque heuristique $a \in \bar{\mathcal{A}}$, pour les fonctions objectif $\mathcal{F}_1$ et $\mathcal{F}_2$, ainsi que pour différents budgets d'évaluations $n \in \{500, 5\,000, 30\,000\}$. Finalement, des profils de performance basés non pas sur un budget d'évaluations, mais en fonction des valeurs des planifications obtenues à l'arrêt des heuristiques (pour un critère d'arrêt $\eta_{max} = 7\,500$) sont représentés sur la figure 4.15.



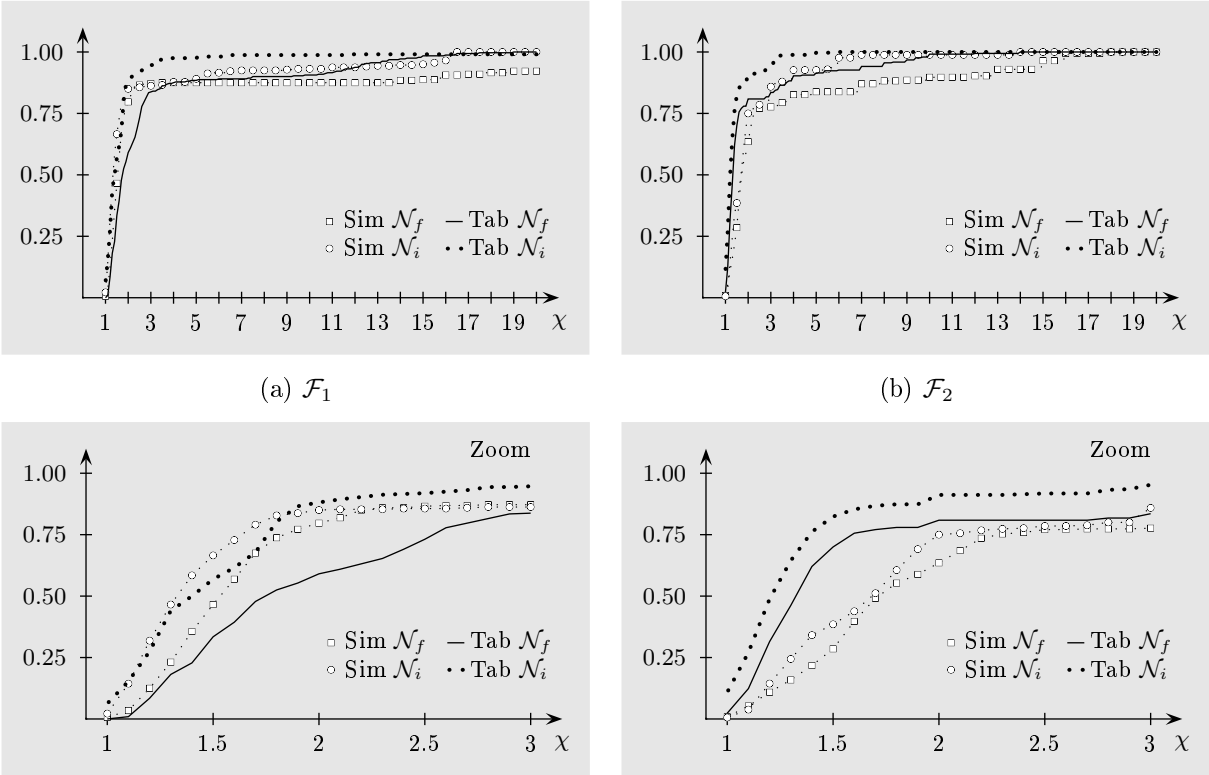
(a) $\mathcal{F}_1$ (b) $\mathcal{F}_2$

FIG. 4.12 – Profils de performance des heuristiques de l'ensemble $\bar{\mathcal{A}}$ pour $n = 500$.



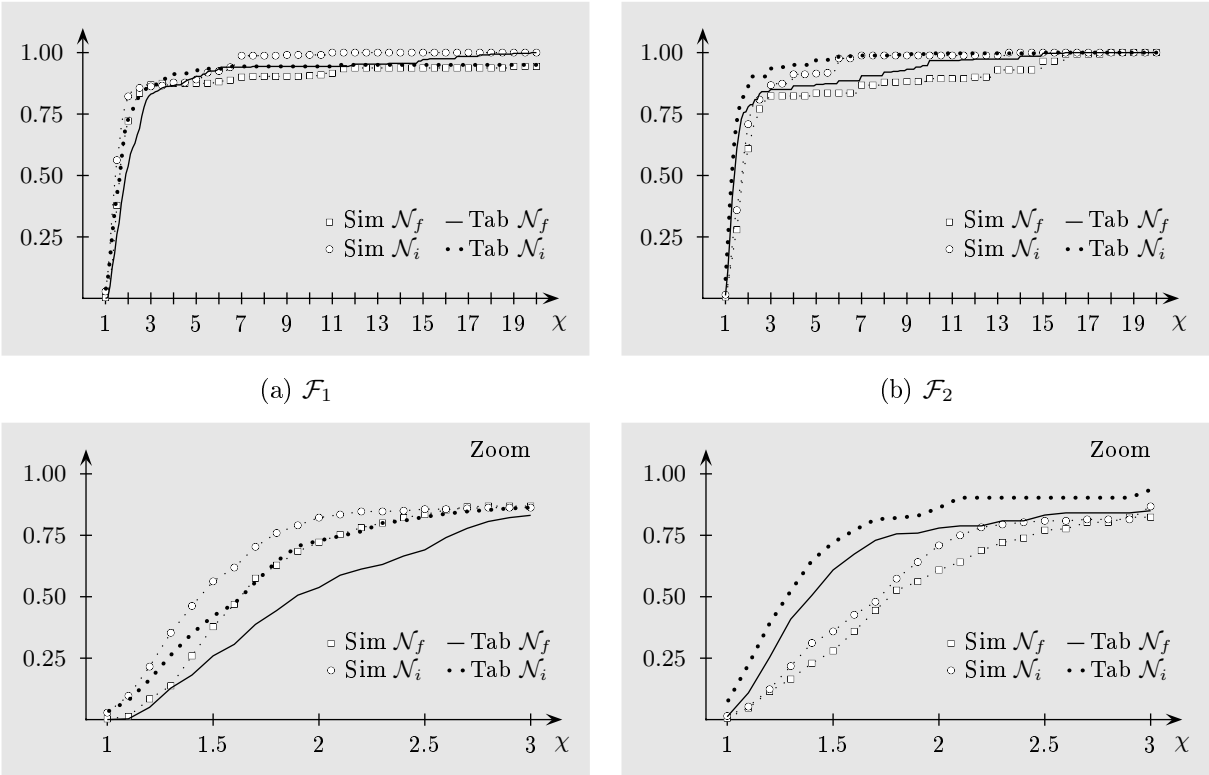
(a) $\mathcal{F}_1$ (b) $\mathcal{F}_2$

FIG. 4.13 – Profils de performance des heuristiques de l'ensemble $\bar{\mathcal{A}}$ pour $n = 5\,000$.



(a) $\mathcal{F}_1$ (b) $\mathcal{F}_2$

FIG. 4.14 – Profils de performance des heuristiques de l'ensemble $\bar{\mathcal{A}}$ pour $n = 30\,000$.



(a) $\mathcal{F}_1$ (b) $\mathcal{F}_2$

FIG. 4.15 – Profils de performance des heuristiques de l'ensemble $\bar{\mathcal{A}}$ pour $\eta_{max} = 7\,500$.

Ces profils de performance permettent notamment d'observer les éléments suivants :

- ▷ la valeur du profil de performance $\rho_a^n(\chi)$ d'une heuristique a en $\chi = 1$ n'est autre que la proportion des expériences impliquant l'heuristique a (chaque expérience correspondant à l'application d'un algorithme de l'ensemble $\mathcal{A}_a$ sur une des instances considérées) pour lesquelles on a obtenu une solution au moins aussi bonne que pour tout autre algorithme de $\mathcal{A}$;
- ▷ le fait qu'un profil de performance atteigne la valeur 1 rapidement (*i.e.* pour une petite valeur de χ) indique que l'heuristique correspondante a de grandes chances de permettre de construire une planification pas trop éloignée de la meilleure solution que l'on pourrait obtenir en utilisant un algorithme de $\mathcal{A}$ (avec le même budget d'évaluations).
- ▷ si le profil de performance d'une heuristique a_1 *domine* celui d'une autre heuristique a_2 (*i.e.* $\rho_{a_1}^n(\chi) \geq \rho_{a_2}^n(\chi)$, $\forall \chi \in [1, \infty)$), alors a_1 offre de meilleures garanties que a_2 sur l'obtention d'une solution d'une certaine qualité (du moins pour un budget de n évaluations).

D'une manière générale, on constate que le profil de performance de la version non admissible de nos deux heuristiques domine celui de la variante admissible correspondante. Il s'agit d'un résultat auquel on pouvait s'attendre, dans la mesure où les méthodes non admissibles utilisent de plus larges voisinages et sont donc plus flexibles.

Analysons tout d'abord les profils de performance de la fonction objectif $\mathcal{F}_1$ (voir figures 4.12(a) à 4.15(a)). En les observant en $\chi = 1$, on conclut que pour un petit budget d'évaluations ($n = 500$), le recuit simulé non admissible fournit généralement des planifications plus efficaces que la recherche tabou (voir figure 4.12(a)). En revanche, à partir d'au plus 30 000 évaluations, la tendance s'inverse et la recherche tabou non admissible fournit de meilleures garanties que le recuit simulé non admissible (voir figure 4.14(a)). De plus, on remarque que, pour les versions admissibles des heuristiques, on obtient systématiquement une valeur nulle en $\chi = 1$, ce qui signifie qu'elles ne trouvent jamais la meilleure planification pour aucune des instances. Nous conseillons donc d'utiliser l'une des deux heuristiques non admissibles pour optimiser ce problème. Si le temps disponible pour une optimisation est très limité, le recuit simulé non admissible semble plus robuste. En effet, pour un budget de 500 ou 5 000 évaluations, son profil de performance domine celui des autres heuristiques (voir figures 4.12(a) et 4.13(a)). Par contre, si l'on peut se permettre un plus grand budget d'évaluations ($n = 30\,000$), l'utilisation de la recherche tabou non admissible est également envisageable, les profils des deux heuristiques non admissibles s'entremêlant.

Pour la fonction objectif $\mathcal{F}_2$ (voir figures 4.12(b) à 4.15(b)), les meilleurs résultats proviennent de la recherche tabou non admissible quel que soit le budget d'évaluations. Cette supériorité du tabou non admissible se manifeste sur l'entier de chacun des profils (*i.e.* pour toute valeur de χ). La différence est légère pour un petit budget d'évaluations (voir figures 4.12(b) et 4.13(b)), mais devient plus marquée pour un grand nombre d'évaluations (voir figure 4.14(b)). La recherche tabou non admissible est donc l'heuristique à conseiller pour l'optimisation de $\mathcal{F}_2$.

Les profils de performance obtenus avec le critère d'arrêt $\eta_{max} = 7500$ (voir figure 4.15) présentent des comportements similaires aux profils de performance établis pour un grand budget d'évaluations. Cette observation nous conforte dans le type de critère d'arrêt choisi. Rappelons qu'il correspond à terminer une optimisation lorsqu'il n'y a pas eu d'amélioration de la meilleure planification rencontrée durant les dernières η_{max} solutions visitées.

Une explication de la supériorité de la recherche tabou pour la fonction objectif $\mathcal{F}_2$ tient au fait qu'un grand nombre de planifications différentes présentent la même valeur de la fonction

objectif. En effet, avec $\mathcal{F}_2$, seule la plus grande perte de temps est considérée. Par conséquent, les heuristiques ne sont pas en mesure de faire la différence entre plusieurs planifications voisines de même valeur et sont donc plus facilement amenées à cycler aléatoirement sur ces dernières. Soit $\mathcal{P}$ une planification et $\mathcal{P}' \in \mathcal{N}(\mathcal{P})$ une planification voisine, telles que $\mathcal{F}_2(\mathcal{P}) = \mathcal{F}_2(\mathcal{P}')$. Avec le recuit simulé, le déplacement vers une planification voisine est toujours effectué en cas d'amélioration stricte de la planification courante, et avec probabilité

$$\exp\left(-\frac{\mathcal{F}_2(\mathcal{P}') - \mathcal{F}_2(\mathcal{P})}{T}\right)$$

dans le cas contraire. Dans le cas présent, cette probabilité vaut toujours 1, ce qui peut facilement mener au cyclage du recuit simulé. Une modification du recuit simulé permettant de remédier à ce comportement est proposée dans la section 5.1. Ce problème est beaucoup moins prononcé pour la recherche tabou, car la présence d'une liste tabou nous préserve, du moins partiellement, de tels phénomènes de cyclage. Cette supériorité de la recherche tabou sur le recuit simulé a déjà été observée par exemple pour la résolution de problèmes de satisfaction d'un maximum de contraintes (voir [HP98]). Cependant, on note que l'observation inverse a également été constatée pour des problèmes tels que l'ordonnancement de tâches (voir [GDL92]) ou le placement de cellules standard sur des puces VLSI (voir [Kur98]).

En conclusion, pour le problème de planification de tournées de véhicules scolaires, lorsque la fonction objectif est $\mathcal{F}_1$, nous conseillons le recuit simulé non admissible si le temps imparti pour l'optimisation est court et la recherche tabou non admissible dans le cas contraire. Lorsque la fonction objectif est $\mathcal{F}_2$, nous recommandons l'utilisation de la recherche tabou non admissible.

4.4. Analyse des planifications obtenues

4.4.1. Application des heuristiques aux instances réelles

Afin d'évaluer l'efficacité des heuristiques développées, comparons les planifications obtenues sur les instances réelles $R_{34}(4)$ et $R_{151}(11)$ avec celles actuellement utilisées par les communes de Savigny et de Forel et par l'École Nouvelle de la Suisse Romande. Dans cette optique, on introduit la notion de *nombre d'enfants réels* (il ne s'agit donc pas de groupes d'enfants) *présentant une perte de temps* (voir (2.6)) *de valeur t pour une planification donnée*

$$N_{\mathcal{C}}(t) = \sum_{c \in \mathcal{C}, f(c)=t} n(c).$$

La planification utilisée par Savigny et Forel en 1997-1998 pour l'instance $R_{34}(4)$ est notée $\mathcal{P}_{act}$. Notons que pour trois enfants, leur planification ne respecte pas la contrainte interdisant aux enfants de changer de bus (voir (2.12), (2.13) et (2.14)). On note $\mathcal{P}_{\mathcal{F}_1}$ (respectivement $\mathcal{P}_{\mathcal{F}_2}$) la meilleure planification produite entre toutes les heuristiques exécutées pour la fonction objectif $\mathcal{F}_1$ (respectivement $\mathcal{F}_2$). Les caractéristiques de ces trois planifications sont reportées dans la table 4.6. Remarquons que l'on a préféré indiquer la moyenne des pertes de temps, $\mathcal{F}_1/N_{\mathcal{C}}$ ($N_{\mathcal{C}}$ étant le nombre total d'enfants réels) plutôt que la valeur de $\mathcal{F}_1$, cette notion étant d'une interprétation plus intuitive.

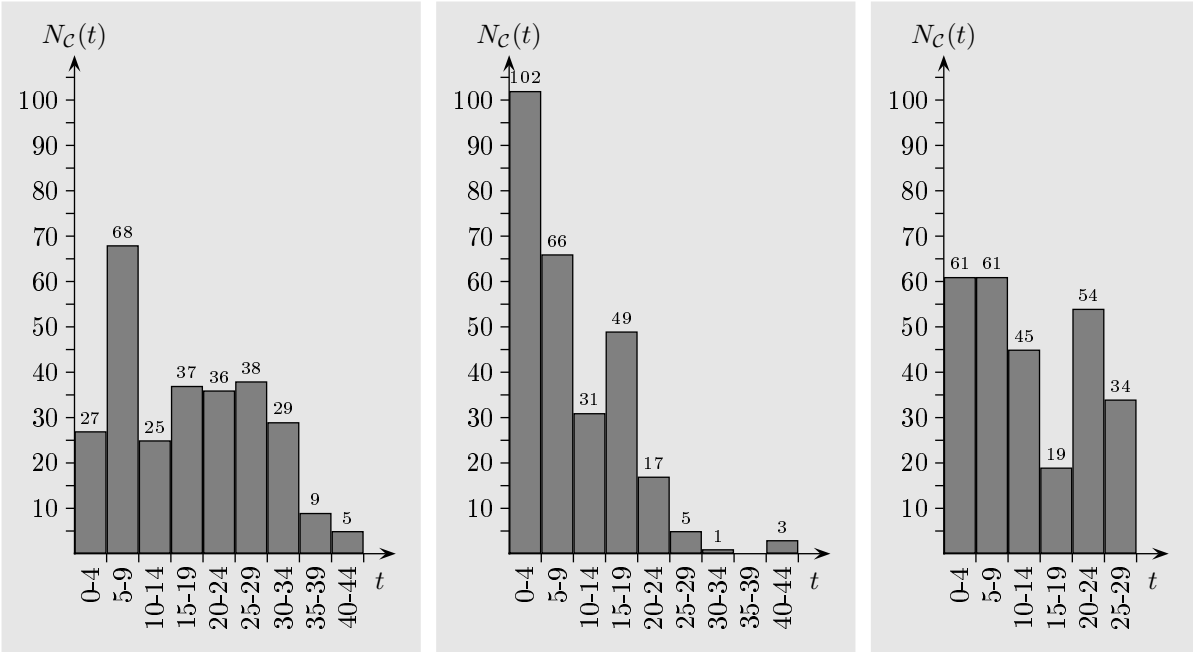
On constate une amélioration significative des principaux critères de comparaison par rapport à la solution actuelle. La solution $\mathcal{P}_{\mathcal{F}_1}$ présente une perte de temps moyenne de 9.1 minutes par enfant, alors qu'elle est de 16.9 minutes dans la planification réelle utilisée et la solution $\mathcal{P}_{\mathcal{F}_2}$ permet de réduire la plus grande perte de temps de 42 à 29 minutes. De plus, on remarque que l'optimisation de $\mathcal{F}_1$ permet aussi d'améliorer $\mathcal{F}_2$ et *vice versa*. On observe également que le

	$\mathcal{P}_{act}$	$\mathcal{P}_{\mathcal{F}_1}$	$\mathcal{P}_{\mathcal{F}_2}$
$\mathcal{F}_1/N_{\mathcal{C}}$ [min.]	16.9	9.1	12.4
$\mathcal{F}_2$ [min.]	42	40	29
$\min_{c\in\mathcal{C}} f(c)$ [min.]	0	0	0
Départ le plus tôt	6h49	7h12	6h59
$N_{\mathcal{C}}(\min_{c\in\mathcal{C}} f(c))$	22	47	42
$N_{\mathcal{C}}(\max_{c\in\mathcal{C}} f(c))$	1	3	12

TAB. 4.6 – Comparaison pour l'instance $R_{34}(4)$ de la planification actuellement utilisée par les communes de Savigny et Forel avec les meilleures planifications produites par les heuristiques.

nombre d'enfants atteignant le minimum des pertes de temps est plus grand pour $\mathcal{P}_{\mathcal{F}_1}$ que pour $\mathcal{P}_{\mathcal{F}_2}$, mais qu'il est surtout plus grand que pour $\mathcal{P}_{act}$.

Afin d'analyser la structure de ces trois solutions, la distribution des pertes de temps des enfants est représentée dans la figure 4.16. On remarque que la solution obtenue pour la fonction objectif $\mathcal{F}_1$ contient une importante proportion d'enfants présentant une petite perte de temps, ainsi qu'une minorité pour laquelle la perte de temps est importante. Par contre, pour la fonction objectif $\mathcal{F}_2$, on constate une tendance à l'uniformisation des pertes de temps, qui ne semble toutefois pas s'être établie au détriment de la majorité.



(a) $\mathcal{P}_{act}$ (b) $\mathcal{P}_{\mathcal{F}_1}$ (c) $\mathcal{P}_{\mathcal{F}_2}$

FIG. 4.16 – Distribution des pertes de temps de trois planifications pour l'instance $R_{34}(4)$.

Effectuons les mêmes comparaisons pour l'instance $R_{151}(11)$. La planification utilisée en 2002-2003 par l'École Nouvelle de la Suisse Romande pour l'instance $R_{151}(11)$ est notée $\mathcal{P}_{act}$. On note $\mathcal{P}_{\mathcal{F}_1}$ (respectivement $\mathcal{P}_{\mathcal{F}_2}$) la meilleure planification produite entre toutes les heuristiques exécutées pour la fonction objectif $\mathcal{F}_1$ (respectivement $\mathcal{F}_2$). Les caractéristiques de ces trois planifications sont reportées dans la table 4.7.

On constate à nouveau des améliorations significatives par rapport à la solution actuellement utilisée. On note que le maximum des pertes de temps pour la planification $\mathcal{P}_{\mathcal{F}_1}$ est un peu moins bon que celui de la planification $\mathcal{P}_{act}$, mais que le gain sur la moyenne des pertes de temps est par contre de 59.5%. Dans ce cas particulier, la planification à proposer à l'École Nouvelle de la Suisse Romande est sans aucun doute $\mathcal{P}_{\mathcal{F}_2}$. En effet, cette solution offre une diminution de 56% de la perte de temps moyenne et permet de réduire de 30% le maximum des pertes de temps.

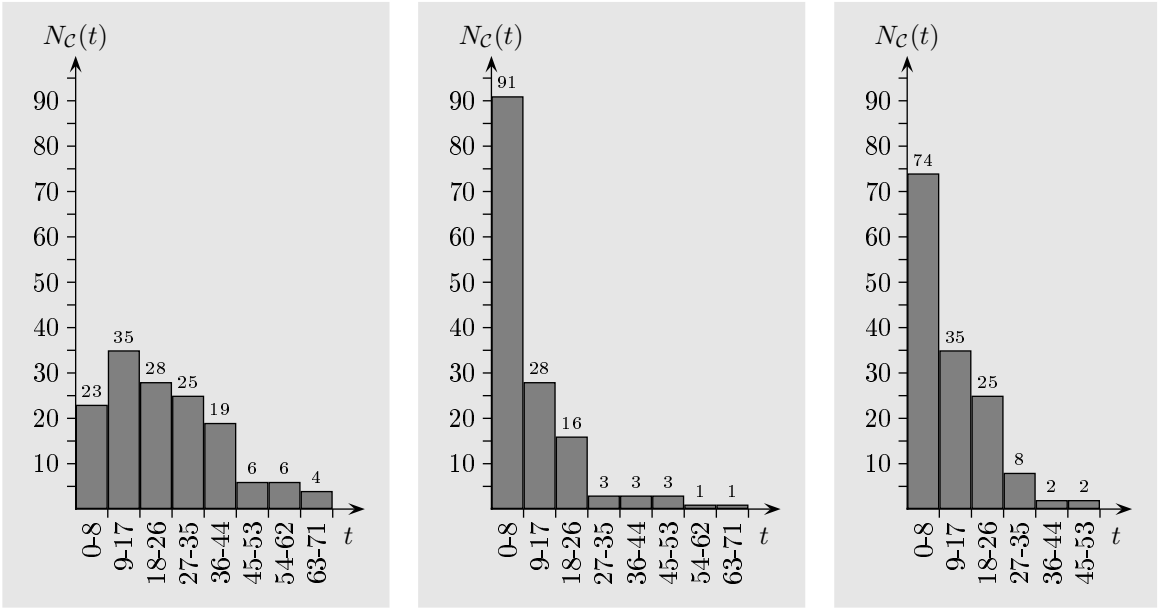
	$\mathcal{P}_{act}$	$\mathcal{P}_{\mathcal{F}_1}$	$\mathcal{P}_{\mathcal{F}_2}$
$\mathcal{F}_1/N_{\mathcal{C}}$ [min.]	24.2	9.8	10.6
$\mathcal{F}_2$ [min.]	69	71	48
$\min_{c\in\mathcal{C}} f(c)$ [min.]	0	0	0
Départ le plus tôt	6h52	7h11	7h09
$N_{\mathcal{C}}(\min_{c\in\mathcal{C}} f(c))$	12	28	27
$N_{\mathcal{C}}(\max_{c\in\mathcal{C}} f(c))$	1	1	2

TAB. 4.7 – Comparaison pour l'instance $R_{151}(11)$ de la planification actuellement utilisée par l'École Nouvelle de la Suisse Romande avec les meilleures planifications produites par les diverses heuristiques.

La distribution des pertes de temps des enfants pour ces trois planifications est représentée sur la figure 4.17. Comme précédemment, on remarque que $\mathcal{P}_{\mathcal{F}_1}$ contient une importante proportion d'enfants présentant une petite perte de temps. En revanche, on constate que l'histogramme de $\mathcal{P}_{\mathcal{F}_2}$ pour l'instance $R_{151}(11)$ est nettement moins équilibré que celui obtenu pour l'instance $R_{34}(4)$. La planification $\mathcal{P}_{\mathcal{F}_2}$ de l'instance $R_{151}(11)$ que nous avons pu obtenir ressemble à une solution où $\mathcal{F}_2$ aurait dans un premier temps été minimisée, puis, sans changer la valeur de $\mathcal{F}_2$, l'autre fonction objectif $\mathcal{F}_1$ aurait été optimisée. Notons que cette seconde optimisation pourrait également être envisagée sur la planification $\mathcal{P}_{\mathcal{F}_2}$ de l'instance $R_{34}(4)$. Une approche consistant à déterminer la planification minimisant la somme des pertes de temps parmi toutes celle présentant le même maximum des pertes de temps est décrite dans la section 5.3.

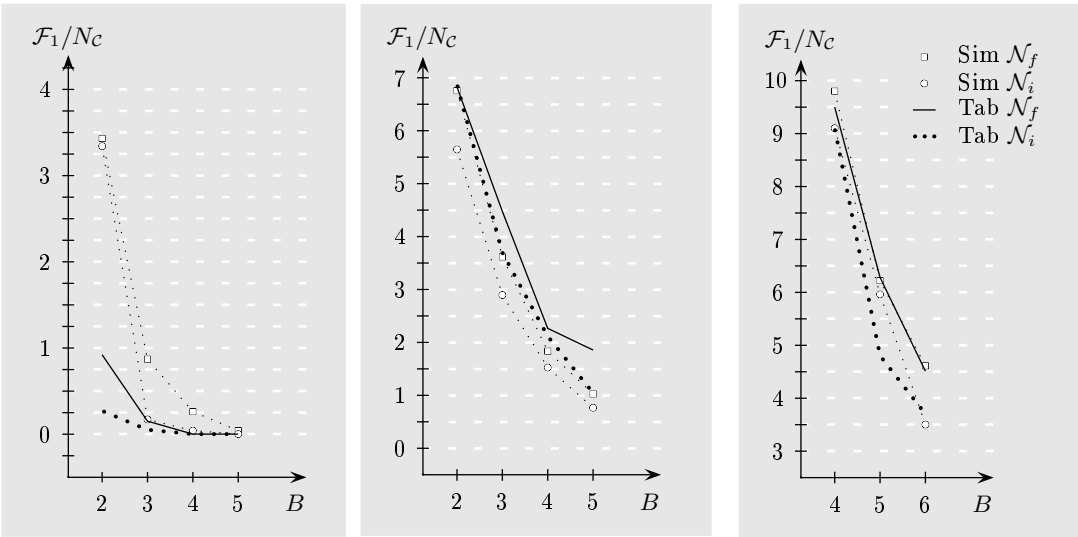
4.4.2. Évolution de la qualité des planifications en fonction du nombre de bus

L'effet d'une variation du nombre de bus disponibles sur la qualité des planifications est une information économique importante, car elle permet d'estimer l'amélioration du niveau de service que l'investissement d'un bus supplémentaire offrirait. Les figures 4.18 et 4.19 illustrent (pour les fonctions objectif $\mathcal{F}_1$ et $\mathcal{F}_2$), l'effet du nombre de bus disponibles sur la qualité des meilleures planifications que l'on peut obtenir à l'aide des diverses heuristiques (pour un budget de 50 000 évaluations) sur les jeux de données F_{11}, F_{20} et R_{34}.



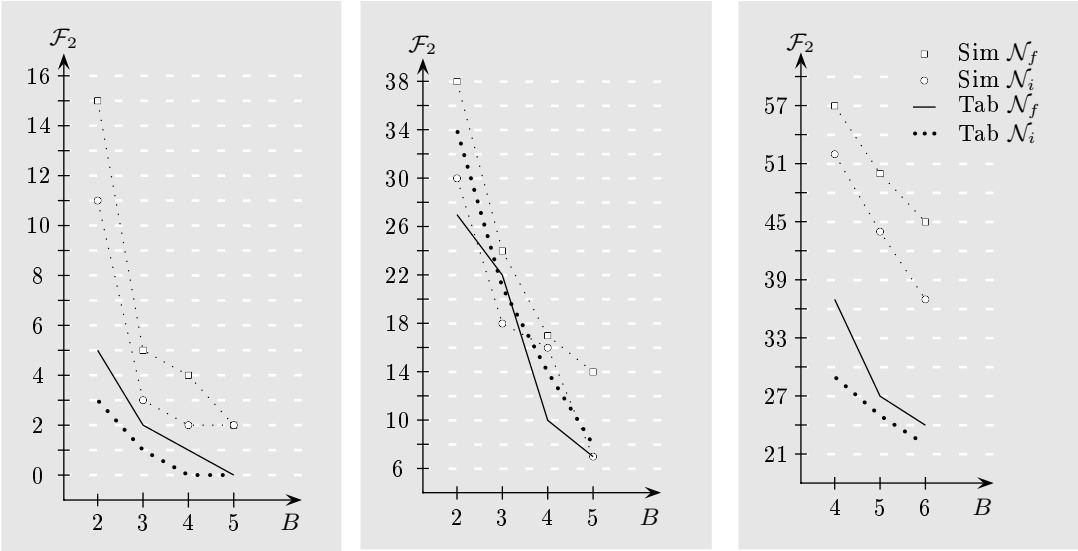
(a) $\mathcal{P}_{act}$ (b) $\mathcal{P}_{\mathcal{F}_1}$ (c) $\mathcal{P}_{\mathcal{F}_2}$

FIG. 4.17 – Distribution des pertes de temps de trois planifications pour l'instance $R_{151}(11)$.



(a) F_{11} (b) F_{20} (c) R_{34}

FIG. 4.18 – Évolution des pertes de temps moyennes (en minutes) des meilleures planifications produites par les diverses heuristiques pour les instances basées sur les jeux de données F_{11}, F_{20} et R_{34}, en fonction du nombre de bus disponibles.



(a) F_{11} (b) F_{20} (c) R_{34}

FIG. 4.19 – Évolution des pertes de temps maximales (en minutes) des meilleures planifications produites par les diverses heuristiques pour les instances basées sur les jeux de données F_{11}, F_{20} et R_{34}, en fonction du nombre de bus disponibles.

Si le nombre de bus à disposition est limité, l'ajout d'un bus supplémentaire peut avoir un grand impact sur la qualité des planifications obtenues. C'est par exemple le cas lors du passage de 4 à 5 bus pour l'instance réelle R_{34}, qui permettrait d'obtenir des pertes de temps moyennes nettement plus petites. Par contre lorsque le nombre de bus est déjà suffisant, l'apport d'un nouveau bus n'a qu'un effet marginal sur les valeurs des fonctions objectif. C'est par exemple le cas lors du passage de 3 à 4 bus pour F_{11}.

4.4.3. Planifications initiales, optimisées et «optimales»

La qualité des planifications initiales générées par la procédure décrite dans la section 3.1, celle des planifications optimisées à l'aide de nos heuristiques, ainsi que celle de solutions «optimales» sont comparées dans cette section. Les tables 4.8 (pour la fonction objectif $\mathcal{F}_1$) et 4.9 (pour la fonction objectif $\mathcal{F}_2$) contiennent, pour un ensemble d'instances, les valeurs des solutions suivantes.

- ▷ La planification initiale obtenue par la méthode décrite dans la section 3.1.
- ▷ La meilleure planification obtenue par l'application de l'heuristique Sim $\mathcal{N}_i$. Pour chaque instance, sur les vingt exécutions (jeux de paramètres) du recuit simulé non admissible effectuées, la planification présentant la meilleure fonction objectif a été retenue. Notons que le critère d'arrêt est fixé à $\eta_{max} = 7\,500$ évaluations sans amélioration.
- ▷ La meilleure planification obtenue par l'application de l'heuristique Tab $\mathcal{N}_i$. Comme pour le recuit simulé non admissible, pour chaque instance on a retenu la planification de meilleure valeur de fonction objectif produite par l'un des vingt jeux de paramètres, avec un critère d'arrêt fixé à $\eta_{max} = 7\,500$ évaluations sans amélioration.
- ▷ La planification «optimale» : ne connaissant naturellement pas la valeur de la solution optimale, nous l'avons estimée à l'aide de la meilleure planification obtenue en laissant tourner chacune de nos heuristiques avec un budget de 50 000 évaluations. Cette solution

nous permettra d'évaluer l'efficacité des planifications produites par nos heuristiques, ainsi que la pertinence du critère d'arrêt choisi.

	$F_{20}(2)$	$F_{20}(3)$	$F_{52}(9)$	$F_{52}(13)$	$F_{105}(20)$	$F_{105}(25)$	$R_{34}(4)$	$R_{34}(5)$	$R_{34}(6)$	$R_{151}(11)$
Planifications initiales	19.3	31.6	30.3	22.2	30.2	23.1	41.1	25.5	17.7	140.9
Planifications optimisées Sim $\mathcal{N}_i$	5.9	2.9	8.8	4.7	12.8	8.4	9.1	5.5	4.7	16.5
Planifications optimisées Tab $\mathcal{N}_i$	7.2	4.4	10.0	4.4	14.0	8.7	9.3	4.8	4.3	10.9
Planifications «optimales»	5.7	2.9	8.3	4.2	11.1	7.2	9.1	4.8	3.5	9.8

TAB. 4.8 – Pertes de temps moyennes (en minutes) des planifications initiales, optimisées par les heuristiques Sim $\mathcal{N}_i$ et Tab $\mathcal{N}_i$, ainsi que des planifications «optimales».

	$F_{20}(2)$	$F_{20}(3)$	$F_{52}(9)$	$F_{52}(13)$	$F_{105}(20)$	$F_{105}(25)$	$R_{34}(4)$	$R_{34}(5)$	$R_{34}(6)$	$R_{151}(11)$
Planifications initiales	49	39	114	65	135	97	141	80	80	660
Planifications optimisées Sim $\mathcal{N}_i$	34	18	70	51	97	81	52	48	37	82
Planifications optimisées Tab $\mathcal{N}_i$	37	21	51	37	98	78	34	28	23	55
Planifications «optimales»	27	18	51	32	93	74	29	25	22	48

TAB. 4.9 – Pertes de temps maximales (en minutes) des planifications initiales, optimisées par les heuristiques Sim $\mathcal{N}_i$ et Tab $\mathcal{N}_i$, ainsi que des planifications «optimales».

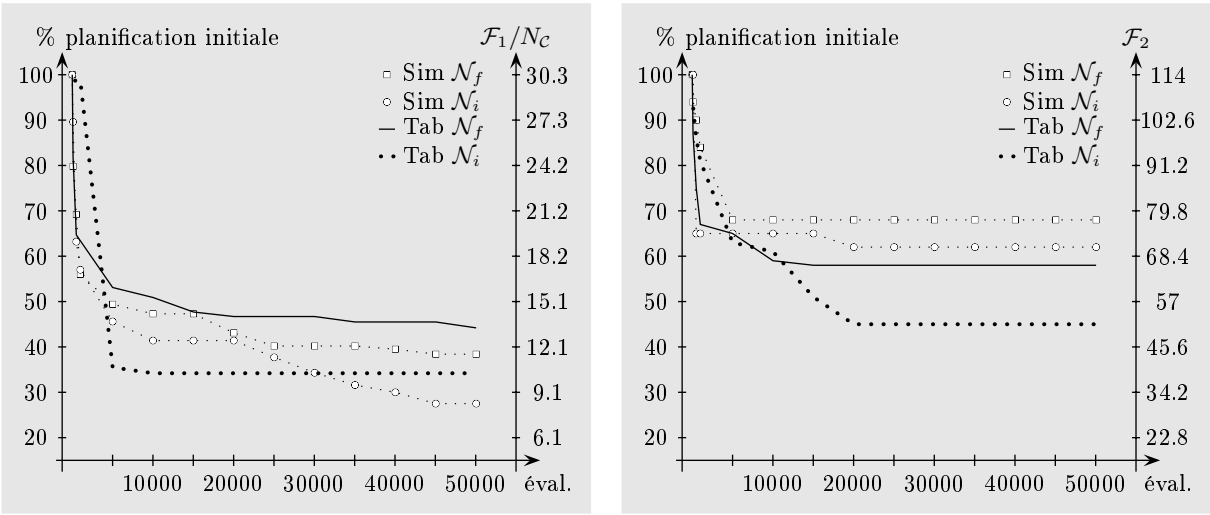
Dans les tables 4.8 et 4.9, on remarque que les planifications initiales obtenues par la méthode décrite dans la section 3.1 présentent des valeurs assez éloignées des solutions optimisées, en particulier pour l'instance $R_{151}(11)$. Rappelons que la construction d'une planification admissible consiste en la création et la concaténation de chemins permettant de transporter au plus le nombre d'enfants qu'il est possible de mettre dans le plus petit bus à disposition. Or, pour l'instance $R_{151}(11)$, la capacité du plus petit bus est de 4 places, soit bien inférieure à celle des 10 autres bus disponibles (voir page 34). D'une manière générale, pour chacune des instances, que la fonction objectif à minimiser soit $\mathcal{F}_1$ ou $\mathcal{F}_2$, on constate qu'il est possible d'obtenir une nette amélioration des planifications initiales. On en conclut que les planifications admissibles construites à l'aide de la méthode de la section 3.1 ne sont généralement pas de très bonne qualité et nécessitent une phase d'optimisation.

Pour la fonction objectif $\mathcal{F}_1$, on observe que les planifications obtenues par l'application du recuit simulé non admissible et de la recherche tabou non admissible produisent des résultats

proches des planifications «optimales». On en déduit donc que le type de critère d'arrêt choisi (stagnation de la meilleure planification trouvée) semble convenir pour ce problème. De plus, les meilleures planifications produites par nos heuristiques, avec les jeux de paramètres proposés, semblent provenir équitablement du recuit simulé et de la recherche tabou.

Pour la fonction objectif $\mathcal{F}_2$, on constate que l'écart entre les planifications optimisées et «optimales» est plus marqué, ce qui n'est guère surprenant compte tenu du fait que ce deuxième problème est plus difficile à optimiser. En général, comme nous l'avons déjà mentionné dans la section 4.3.3, la recherche tabou non admissible semble produire de meilleures planifications pour ce problème que le recuit simulé non admissible. À notre avis, les écarts importants constatés entre les planifications produites par le recuit simulé non admissible et les planifications «optimales» pour les instances basées sur les jeux de données F_{52}, R_{34} et R_{151} ne sont pas imputables au critère d'arrêt, mais plutôt au phénomène de cyclage du recuit déjà discuté en page 51.

Pour l'instance $F_{52}(9)$, on trouve sur la figure 4.20 l'évolution des valeurs des planifications rencontrées par nos quatre heuristiques (pour chaque heuristique, les paramètres produisant la meilleure solution ont été retenus) au cours du déroulement de nos expériences. On peut y lire le pourcentage de réduction de la valeur de la fonction objectif par rapport à la planification initiale, ainsi que la valeur de la fonction objectif, en fonction du nombre d'évaluations de planifications candidates. On constate que, partant de la planification initiale obtenue par la méthode de la section 3.1, toutes nos heuristiques présentent le même type de comportement : des améliorations significatives durant les premières itérations, puis une phase de décélération se terminant par une stagnation des résultats. Ce phénomène a été observé sur toutes les instances testées et il nous conforte dans le choix d'un critère d'arrêt basé sur la stagnation de la valeur de la meilleure planification trouvée.



(a) $\mathcal{F}_1$ (b) $\mathcal{F}_2$

FIG. 4.20 – Évolution de la qualité des planifications obtenues pour l'instance $F_{52}(9)$.

4.4.4. Nombre d'évaluations et temps de calcul

Les tests ont été effectués sur un Pentium 4 cadencé à 2GHz. Pour un choix d'instances, la table 4.10 contient le nombre moyen d'évaluations effectuées, sur les vingt jeux de paramètres,

pour chaque problème et chacune des heuristiques. Les temps moyens d'exécution correspondants sont donnés dans la table 4.11.

		F_{20}				F_{52}		F_{105}	
B		2	3	4	5	9	13	20	25
$\mathcal{F}_1$	Sim $\mathcal{N}_f$	14 989	15 289	15 440	16 923	27 090	19 762	12 991	17 928
	Sim $\mathcal{N}_i$	14 567	16 012	16 311	17 553	15 884	16 450	17 908	18 859
	Tab $\mathcal{N}_f$	11 996	15 143	20 201	18 124	22 842	26 006	26 358	28 418
	Tab $\mathcal{N}_i$	20 680	20 577	22 661	19 010	10 425	13 232	12 122	19 318
$\mathcal{F}_2$	Sim $\mathcal{N}_f$	10 371	10 901	9 955	8 509	9 262	10 252	11 282	11 472
	Sim $\mathcal{N}_i$	12 364	10 998	10 630	12 996	9 554	9 837	14 293	12 599
	Tab $\mathcal{N}_f$	14 396	12 467	11 684	9 374	14 931	16 563	20 913	23 413
	Tab $\mathcal{N}_i$	10 305	10 723	11 615	11 382	13 143	20 667	22 524	21 677

		R_{34}			R_{151}
B		4	5	6	11
$\mathcal{F}_1$	Sim $\mathcal{N}_f$	18 280	17 825	25 852	23 761
	Sim $\mathcal{N}_i$	17 138	15 238	26 833	24 714
	Tab $\mathcal{N}_f$	26 063	28 411	34 836	37 773
	Tab $\mathcal{N}_i$	18 752	16 980	34 490	34 403
$\mathcal{F}_2$	Sim $\mathcal{N}_f$	10 847	17 903	8 948	8 648
	Sim $\mathcal{N}_i$	10 820	11 772	8 780	8 605
	Tab $\mathcal{N}_f$	22 982	24 531	9 369	8 742
	Tab $\mathcal{N}_i$	18 856	27 342	9 609	8 387

TAB. 4.10 – Nombre total (en moyenne sur les vingt jeux de paramètres) d'évaluations effectuées avec le critère d'arrêt $\eta_{max} = 7\,500$.

Globalement, on constate que le nombre d'évaluations de planifications est plus petit pour la fonction objectif $\mathcal{F}_2$ que pour $\mathcal{F}_1$. Sachant que le problème $\mathcal{F}_2$ est plus difficile, ce phénomène illustre qu'avec le problème $\mathcal{F}_1$, nos heuristiques se laissent moins facilement piéger dans un minimum local.

En outre, on constate que plus le nombre de bus à disposition est grand, plus les temps de calcul semblent courts, même si le nombre d'évaluations augmente. Par exemple, pour l'instance $R_{34}(4)$ résolue avec le recuit simulé admissible pour la fonction objectif $\mathcal{F}_1$, le nombre moyen d'évaluations (18 280) est inférieur au nombre moyen d'évaluations pour l'instance $R_{34}(6)$ avec la même heuristique (25 852), alors que le temps de calcul pour l'instance $R_{34}(4)$ (47.94 minutes) est supérieur à celui de l'instance $R_{34}(6)$ (32.82 minutes). Ce comportement peut s'expliquer simplement : plus la tournée d'un bus est longue, plus il faut de temps pour trouver le meilleur point d'insertion d'un enfant à déplacer. Par conséquent, pour un même jeu de données, plus il y a de bus à disposition, plus les tournées sont courtes et plus le temps de calcul (par évaluation) est bref.

Prolongeant ce raisonnement, si pour deux instances le rapport entre le nombre d'enfants et le nombre de bus est similaire, alors le jeu de données contenant le plus d'enfants à transporter devrait mener aux calculs les plus longs. Par exemple, on constate que les instances suivantes (classées dans l'ordre croissant du nombre d'enfants) présentent un rapport d'environ 46 enfants réels par bus de 30 places : $F_{20}(2)$, $R_{34}(6)$, $F_{52}(13)$ et $F_{105}(20)$. Pour la fonction objectif $\mathcal{F}_2$, la

		F_{20}				F_{52}		F_{105}	
B		2	3	4	5	9	13	20	25
$\mathcal{F}_1$	Sim $\mathcal{N}_f$	7.79	5.31	3.54	3.44	54.23	30.91	27.78	39.37
	Sim $\mathcal{N}_i$	6.88	5.04	4.30	4.42	18.47	24.36	71.89	43.35
	Tab $\mathcal{N}_f$	2.61	2.26	2.06	1.66	19.97	13.67	35.99	28.09
	Tab $\mathcal{N}_i$	4.32	2.85	2.35	1.87	6.77	5.70	7.99	13.91
$\mathcal{F}_2$	Sim $\mathcal{N}_f$	4.66	3.02	1.82	1.34	16.87	10.18	26.72	25.72
	Sim $\mathcal{N}_i$	6.11	2.94	1.92	2.20	19.70	11.30	37.74	25.42
	Tab $\mathcal{N}_f$	2.72	1.55	1.11	0.67	12.74	9.67	15.87	16.06
	Tab $\mathcal{N}_i$	2.30	1.28	1.13	0.83	6.99	9.19	17.89	12.87

		R_{34}			R_{151}
B		4	5	6	11
$\mathcal{F}_1$	Sim $\mathcal{N}_f$	47.94	27.50	32.82	21.72
	Sim $\mathcal{N}_i$	42.37	24.65	45.93	20.37
	Tab $\mathcal{N}_f$	16.39	13.61	14.35	5.74
	Tab $\mathcal{N}_i$	9.53	7.75	9.21	5.92
$\mathcal{F}_2$	Sim $\mathcal{N}_f$	26.27	21.84	9.02	8.52
	Sim $\mathcal{N}_i$	21.95	15.65	8.90	5.47
	Tab $\mathcal{N}_f$	13.78	8.65	3.17	1.42
	Tab $\mathcal{N}_i$	10.85	11.14	2.77	1.43

TAB. 4.11 – Temps moyen (en minutes) d'exécution des heuristiques (en moyenne sur les vingt jeux de paramètres) avec $\eta_{max} = 7\,500$.

résolution par le recuit simulé non admissible prend 6.11 minutes pour $F_{20}(2)$, 8.90 minutes pour $R_{34}(6)$, 11.30 minutes pour $F_{52}(13)$ et 37.74 minutes pour $F_{105}(20)$. Ces durées étant croissantes, notre supposition est vérifiée pour cet ensemble d'instances.

Afin de pousser plus loin l'observation de l'évolution du temps de calcul en fonction du nombre d'enfants à transporter, des instances de plus grande taille (présentant toujours un rapport de 46 enfants réels par bus de 30 places) ont été générées en utilisant la méthode décrite dans la section 4.1.2. Le tableau 4.12 et la figure 4.21 montrent comment évolue le temps de calcul requis pour construire et évaluer une planification voisine en fonction du nombre d'enfants à transporter. Les valeurs représentées ont été estimées sur la base d'un échantillon de 200 évaluations.

$\lvert\mathcal{C}\rvert$	17	32	47	91	102	190	401	566	963	1935	2378
temps de calcul	0.0087	0.0098	0.022	0.047	0.054	0.083	0.33	0.58	2.07	5.71	9.63

TAB. 4.12 – Estimation du temps de calcul (en secondes) nécessaire pour créer et évaluer une planification voisine en fonction du nombre d'enfants à transporter.

La croissance observée est telle que, même en supposant que la résolution d'une instance de grande taille ne nécessite pas un plus grand nombre d'évaluations de planifications qu'une de celles que nous avons optimisées, il n'est pas réaliste d'espérer générer en un temps raisonnable une planification de bonne qualité pour un problème comprenant plusieurs milliers d'enfants. Notons qu'un problème de planification de tournées de véhicules scolaires dans une métropole

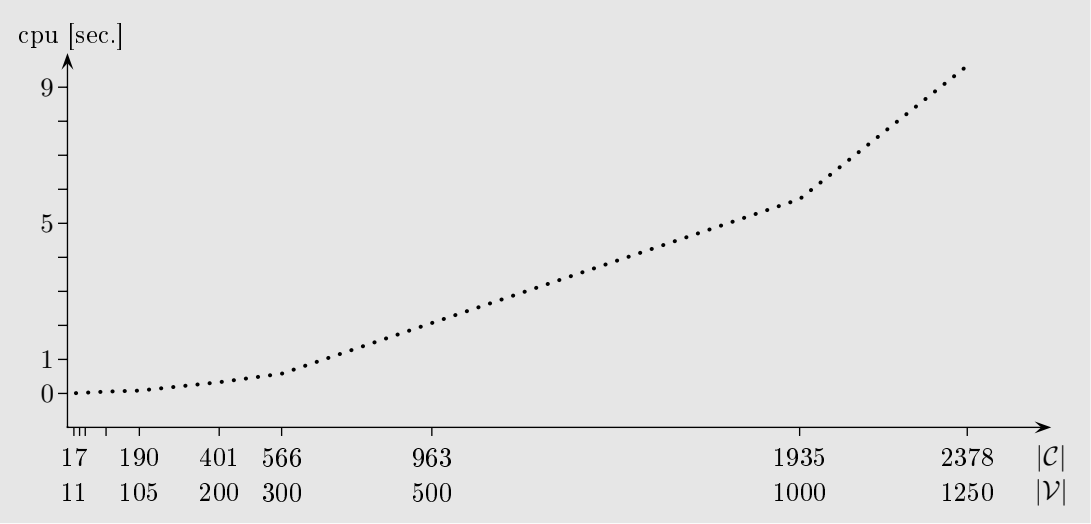


FIG. 4.21 – Estimation de l'évolution du temps de calcul (en secondes) nécessaire pour la création et l'évaluation d'une planification voisine en fonction du nombre d'enfants à transporter (la seconde échelle indique le nombre de sommets du réseau correspondant).

ou une grande région géographique peut être abordé à l'aide d'une approche par décomposition en sous-problèmes. Une technique adaptée à la résolution de problèmes de grande taille est développée au chapitre 6.

4.4.5. Améliorations produites par l'application de la procédure Lin-2-opt

Les heuristiques proposées dans ce document sont toutes basées sur le même schéma (voir algorithmes 3.7 et 3.9) : application d'un recuit simulé ou d'une recherche tabou, puis utilisation de la procédure Lin-2-opt. Pour étudier l'impact du Lin-2-opt sur l'efficacité des planifications produites par les méthodes de recherche locale, les pourcentages d'amélioration moyens (sur toutes les heuristiques et tous les jeux de paramètres) sont reportés dans la table 4.4.5. On constate que l'effet de cette procédure n'est pas négligeable, notamment pour $\mathcal{F}_1$.

	$F_{11}(2)$	$F_{11}(3)$	$F_{20}(2)$	$F_{20}(3)$	$F_{32}(3)$	$F_{32}(6)$	$F_{52}(9)$	$F_{52}(13)$	$F_{105}(20)$	$F_{105}(25)$
$\mathcal{F}_1$	5.35	4.0	1.26	2.0	2.45	5.15	4.11	6.37	3.26	4.24
$\mathcal{F}_2$	0.2	0	0.58	0.6	0.4	0.25	0.66	1.2	1.45	1.85

	$R_{34}(4)$	$R_{34}(5)$	$R_{34}(6)$	$R_{151}(11)$
$\mathcal{F}_1$	3.09	4.95	5.93	5.51
$\mathcal{F}_2$	0.54	0.58	0.7	0.53

TAB. 4.13 – Pourcentages d'amélioration moyens obtenus par l'application du Lin-2-opt sur les planifications construites à l'aide des méthodes de recherche locale.

Notons cependant, qu'il ne serait pas envisageable d'appliquer la procédure Lin-2-opt à chaque évaluation de planification. En effet, cette procédure étant relativement coûteuse, les temps de calcul s'en trouveraient fortement augmentés. En outre, le fait de transiter par des planifications d'efficacité moyenne peut parfois éviter aux heuristiques de rester bloquées dans un optimum local. Cette constatation a déjà été effectuée lors de la comparaison des planifications produites par le voisinage admissible $\mathcal{N}_f$ et non admissible $\mathcal{N}_i$ (voir page 50).

4.5. Planifications du problème retour

Le problème *retour* décrit dans la section 2.7 revient à résoudre le problème consistant à ramener les enfants de l'école à leur maison. Les heures de début et de fin des cours des instances basées sur les réseaux R_{34}, F_{11}, F_{20}, F_{32}, F_{52} et F_{105} sont reportées dans la table 4.14, alors que celles de l'instance $R_{151}(11)$ sont données dans la table 4.15. On remarque que pour les deux problèmes (l'aller et le retour), l'ordre dans lequel les enfants doivent être transportés est inversé.

Aller			Retour		
secondaire	7h35		enfantine	11h10	
		35 minutes			25 minutes
primaire	8h10		primaire	11h35	
		20 minutes			30 minutes
enfantine	8h30		secondaire	12h05	

TAB. 4.14 – Horaires des instances basées sur les réseaux R_{34}, F_{11}, F_{20}, F_{32}, F_{52} et F_{105}.

Aller			Retour		
primaire	8h20		enfantine	12h35	
		40 minutes			0 minutes
enfantine	9h00		primaire	12h35	

TAB. 4.15 – Horaire de l'instance $R_{151}(11)$.

Pour déterminer une planification du problème retour, il est donc envisageable de considérer la meilleure solution du problème aller et de la renverser. Plus précisément, générer une *planification renversée* revient à effectuer les opérations suivantes sur une planification du problème aller :

- ▷ faire monter dans le bus les enfants à l'arrêt où ils descendaient ;
- ▷ faire descendre du bus les enfants à l'arrêt où ils montaient ;
- ▷ faire parcourir au bus le trajet en sens inverse ;
- ▷ calculer pour chaque tournée un horaire «calé à gauche», c'est-à-dire faire en sorte que les écoles soient visitées le plus tôt possible (calcul par une méthode analogue à l'algorithme 3.6).

Bien évidemment, la planification renversée n'est admissible pour le problème retour que si les enfants à transporter sont les mêmes qu'à l'aller. Bien que rien ne garantisse l'efficacité de cette

planification renversée, un échelonnement des heures de fin des cours similaire à celui des heures de début des cours peut néanmoins laisser présager de la bonne qualité d'une telle solution.

Afin d'évaluer l'efficacité des planifications renversées, la méthode de la section 3.1 a été adaptée pour construire une planification admissible du problème retour. De plus, les heuristiques du chapitre 3 ont été modifiées (l'objectif, décrit en section 2.7, étant légèrement différent pour le problèmes retour) afin d'optimiser les deux types de planifications initiales disponibles (renversée et construite). Quelques résultats concernant la qualité des planifications construites, renversées et optimisées sont présentés pour diverses instances dans les tables 4.16 (pour la moyenne des pertes de temps $\mathcal{F}_1/N_{\mathcal{C}}$) et 4.17 (pour le maximum des pertes de temps $\mathcal{F}_2$). Les planifications renversées considérées sont le fruit du renversement des meilleures planifications trouvées pour le problème aller, toutes heuristiques confondues. Chaque planification optimisée est la meilleure planification produite par nos heuristiques en partant de la meilleure des deux solutions initiales considérées.

	$F_{20}(2)$	$F_{20}(3)$	$F_{20}(4)$	$F_{20}(5)$	$F_{52}(9)$	$F_{52}(13)$	$F_{105}(20)$	$F_{105}(25)$
Planifications construites	33	14.8	10.3	8.4	33.8	20.9	31.3	24.4
Planifications renversées	6.4	3.3	1.6	0.8	7.9	3.7	11.8	7.3
Planifications optimisées	6.2	3.2	1.5	0.7	7.7	3.0	8.8	6.1

	$R_{34}(4)$	$R_{34}(5)$	$R_{34}(6)$	$R_{151}(11)$
Planifications construites	47.9	30.4	23.8	85
Planifications renversées	8.0	5.0	4.0	29.8
Planifications optimisées	7.8	4.6	3.8	22.1

TAB. 4.16 – Valeurs (en minutes) des pertes de temps moyennes, $\mathcal{F}_1/N_{\mathcal{C}}$, des planifications construites, renversées et optimisées pour le problème retour.

En observant les tables 4.16 et 4.17, on constate que les planifications renversées sont de très bonne qualité, en particulier pour les instances basées sur les réseaux F_{20}, F_{52}, F_{105} et R_{34}. En revanche, pour l'instance $R_{151}(11)$, l'écart entre les planifications renversées et optimisées est plus important. Dans la mesure où l'échelonnement des heures de début et de fin des cours est dans ce cas nettement différent (voir table 4.15), ce résultat n'est guère surprenant. Néanmoins, d'une manière générale, les planifications renversées paraissent bien meilleures que les planifications construites et, l'effort d'optimisation restant à fournir semblant limité, nous recommandons de les utiliser comme solution initiale de nos heuristiques pour le problème retour.

	$F_{20}(2)$	$F_{20}(3)$	$F_{20}(4)$	$F_{20}(5)$	$F_{52}(9)$	$F_{52}(13)$	$F_{105}(20)$	$F_{105}(25)$
Planifications construites	75	37	34	34	110	69	120	120
Planifications renversées	27	20	16	7	87	52	98	73
Planifications optimisées	27	20	12	7	86	50	90	73

	$R_{34}(4)$	$R_{34}(5)$	$R_{34}(6)$	$R_{151}(11)$
Planifications construites	163	106	106	299
Planifications renversées	34	29	25	104
Planifications optimisées	34	26	24	82

TAB. 4.17 – Valeurs (en minutes) des pertes de temps maximales, $\mathcal{F}_2$, des planifications construites, renversées et optimisées pour le problème retour.

CHAPITRE 5

Amélioration des heuristiques de base

Afin d'améliorer les heuristiques présentées dans le chapitre 3, quelques variations ont été envisagées. La première concerne le recuit simulé, lorsque l'on cherche à minimiser le maximum des pertes de temps. La seconde se situe au niveau de la manière de choisir l'enfant à déplacer. Une approche permettant de réduire à la fois le maximum et la moyenne des pertes de temps est également proposée. Finalement, une méthode de recherche à voisinage variable est décrite. Dans chaque cas, les résultats de quelques expériences numériques sont discutés.

5.1. Recuit simulé pour le problème avec la fonction objectif $\mathcal{F}_2$

Comme nous l'avons mentionné en page 51, le recuit simulé est moins efficace que la recherche tabou lorsqu'il s'agit d'optimiser le problème avec la fonction objectif $\mathcal{F}_2$, le maximum des pertes de temps. En effet, il est courant que plusieurs planifications voisines présentent le même maximum des pertes de temps, alors qu'elles sont en fait sensiblement différentes. Dans une telle situation, il est judicieux d'envisager d'autres critères que la seule indication du maximum des pertes de temps pour décider si un mouvement doit être effectué ou non. Dans la section 5.1.1, l'utilisation de trois critères secondaires est proposée. Les résultats correspondants sont présentés dans la section 5.1.2.

5.1.1. Critères discriminants

Dans l'algorithme du recuit simulé, le fait de se déplacer vers une planification voisine dépend de la probabilité (3.8). Nous avons déjà observé que si deux planifications voisines $\mathcal{P}$ et $\mathcal{P}' \in \mathcal{N}(\mathcal{P})$ sont telles que $\mathcal{F}_2(\mathcal{P}) = \mathcal{F}_2(\mathcal{P}')$, alors cette probabilité de déplacement vaut toujours 1. Nous allons donc modifier cette probabilité de la manière suivante : si $\mathcal{F}_2(\mathcal{P}) = \mathcal{F}_2(\mathcal{P}')$, alors le fait d'accepter ou non la solution voisine $\mathcal{P}'$ dépend de la probabilité

$$\min\left\{1, \exp\left(-\frac{\mathcal{G}(\mathcal{P}') - \mathcal{G}(\mathcal{P})}{T^{\mathcal{G}}}\right)\right\},$$

où $\mathcal{G}$ est un nouveau critère discriminant et $T^{\mathcal{G}} > 0$ est la température correspondante. Notons que ce dernier paramètre d'optimisation est fixé à une certaine valeur initiale $T_0^{\mathcal{G}}$.

Cette nouvelle règle d'acceptation d'une solution voisine revient à remplacer les lignes 10 à 16 de l'algorithme 3.7 par les instructions détaillées dans l'algorithme 5.1. De même, la nouvelle mise à jour de la température de l'algorithme du recuit simulé revient à remplacer les lignes 25 à 28 de l'algorithme 3.7 par les instructions décrites dans l'algorithme 5.2.

Trois critères discriminants $\mathcal{G}$ ont été considérés :

- ▷ la somme des pertes de temps : $\mathcal{G}_1 = \mathcal{F}_1$;
- ▷ le nombre d'enfants atteignant le maximum des pertes de temps :

$$\mathcal{G}_2 = \left|\left\{c \in \mathcal{C} \mid f(c) = \max_{c' \in \mathcal{C}} f(c')\right\}\right| ;$$

▷ la deuxième plus grande perte de temps :

$$\mathcal{G}_3 = \max_{c \in \mathcal{C}'} f(c), \text{ où } \mathcal{C}' = \{c' \in \mathcal{C} \mid f(c') < \max_{c \in \mathcal{C}} f(c)\}.$$

Algorithme 5.1 Critère d'acceptation d'une solution pour le recuit simulé

Si $\mathcal{F}(\mathcal{P}') < \mathcal{F}(\mathcal{P})$ **alors** (∗ trouvé une meilleure planification ∗)

$\mathcal{P} \leftarrow \mathcal{P}'$, $\gamma \leftarrow 0$

Sinon

Si $\mathcal{F}(\mathcal{P}') = \mathcal{F}(\mathcal{P})$ **alors**

Si $\mathcal{G}(\mathcal{P}') < \mathcal{G}(\mathcal{P})$ **alors** (∗ critère discriminant ∗)

$\mathcal{P} \leftarrow \mathcal{P}'$, $\gamma \leftarrow 0$

Sinon

Si $\exp\left(-\frac{\mathcal{G}(\mathcal{P}') - \mathcal{G}(\mathcal{P})}{T^{\mathcal{G}}}\right) > (y \sim U[0,1))$ **alors**

$\mathcal{P} \leftarrow \mathcal{P}'$, $\gamma \leftarrow 0$ (∗ moins bonne planification selon $\mathcal{G}$ acceptée ∗)

Sinon

Si $\exp\left(-\frac{\mathcal{F}(\mathcal{P}') - \mathcal{F}(\mathcal{P})}{T}\right) > (y \sim U[0,1))$ **alors**

$\mathcal{P} \leftarrow \mathcal{P}'$, $\gamma \leftarrow 0$ (∗ moins bonne planification acceptée ∗)

Sinon

$\gamma \leftarrow \gamma + 1$

Algorithme 5.2 Mise à jour des températures

Si $(n \mod \mu = 0)$ **alors**

$T \leftarrow \omega \cdot T$, $T^{\mathcal{G}} \leftarrow \omega \cdot T^{\mathcal{G}}$ (∗ diminution des températures ∗)

Si $(\gamma > \gamma_{max})$ **alors**

$T \leftarrow T_0$, $T^{\mathcal{G}} \leftarrow T_0^{\mathcal{G}}$, $\gamma \leftarrow 0$ (∗ gel ∗)

5.1.2. Résultats obtenus

Ces critères discriminants n'ont été appliqués qu'au recuit simulé non admissible. On note Sim $\mathcal{N}_i(\mathcal{G}_j)$ l'heuristique du recuit simulé non admissible avec la fonction objectif discriminante $\mathcal{G}_j$, avec $j \in \{1, 2, 3\}$. Les paramètres d'optimisation utilisés pour ces trois heuristiques sont les

mêmes que ceux décrits dans la section 4.2. Les valeurs suivantes des températures initiales ont été retenues :

$$(T_0, T_0^{\mathcal{G}}) \in \begin{cases} \{(5,250),(6,500),(7,750),(8,1\,000)\} & \text{pour } \mathcal{G} = \mathcal{G}_1, \\ \{(5,5),(6,6),(7,7),(8,8)\} & \text{pour } \mathcal{G} = \mathcal{G}_2 \text{ et } \mathcal{G} = \mathcal{G}_3. \end{cases}$$

Ainsi, un total de vingt jeux de paramètres (quatre couples $(T_0, T_0^{\mathcal{G}})$ et cinq germes distincts du générateur de nombres pseudo-aléatoires) ont été testés avec chacune des heuristiques. Pour un ensemble d'instances, la table 5.1 contient la perte de temps maximale obtenue pour les solutions suivantes.

- ▷ La meilleure planification obtenue par l'application de l'heuristique Sim $\mathcal{N}_i$ à la solution initiale générée par la procédure décrite dans la section 3.1. Pour chaque instance, sur les vingt exécutions (jeux de paramètres) du recuit simulé non admissible effectuées, la planification présentant la meilleure fonction objectif a été retenue. Notons que le critère d'arrêt a été fixé à $\eta_{max} = 7\,500$ évaluations de planifications sans amélioration.
- ▷ La meilleure planification obtenue par l'application de chacune des trois heuristiques Sim $\mathcal{N}_i(\mathcal{G}_j)$, $j \in \{1,2,3\}$. Comme pour le recuit simulé non admissible, pour chaque instance et chaque heuristique on a retenu la planification de meilleure valeur de fonction objectif construite à l'aide d'un des vingt jeux de paramètres, avec un critère d'arrêt fixé à $\eta_{max} = 7\,500$ évaluations de planifications sans amélioration.
- ▷ La meilleure planification obtenue par l'application de l'heuristique Tab $\mathcal{N}_i$. Comme ci-dessus, pour chaque instance on a retenu la planification de meilleure valeur de fonction objectif construite à l'aide d'un des vingt jeux de paramètres, avec un critère d'arrêt fixé à $\eta_{max} = 7\,500$ évaluations de planifications sans amélioration.

	$F_{32}(3)$	$F_{32}(6)$	$F_{52}(9)$	$F_{52}(13)$	$F_{105}(20)$	$F_{105}(25)$	$R_{34}(4)$	$R_{34}(5)$	$R_{34}(6)$
Sim $\mathcal{N}_i$	54	28	70	51	97	81	52	48	37
Sim $\mathcal{N}_i(\mathcal{G}_1)$	52	24	62	49	**94**	**78**	43	37	33
Sim $\mathcal{N}_i(\mathcal{G}_2)$	58	26	59	52	95	79	43	34	28
Sim $\mathcal{N}_i(\mathcal{G}_3)$	**46**	24	63	46	**94**	79	47	33	32
Tab $\mathcal{N}_i$	53	**19**	**51**	**37**	98	**78**	**34**	**28**	**23**

TAB. 5.1 – Pertes de temps maximales (en minutes) des solutions obtenues par les heuristiques Sim $\mathcal{N}_i$, Sim $\mathcal{N}_i(\mathcal{G}_1)$, Sim $\mathcal{N}_i(\mathcal{G}_2)$, Sim $\mathcal{N}_i(\mathcal{G}_3)$ et Tab $\mathcal{N}_i$.

On constate que l'application des trois variantes Sim $\mathcal{N}_i(\mathcal{G}_j)$, $j \in \{1,2,3\}$ permet d'obtenir des planifications de meilleure qualité que la méthode Sim $\mathcal{N}_i$. L'heuristique Sim $\mathcal{N}_i(\mathcal{G}_2)$ semble la moins efficace des trois. En effet, à l'aide de cette méthode, il arrive même que l'on obtienne une planification de moins bonne qualité que celle produite par l'heuristique Sim $\mathcal{N}_i$ (voir $F_{32}(3)$ et $F_{52}(13)$). Toutefois, d'une manière générale, l'ajout d'un critère discriminant au recuit simulé non admissible permet d'approcher la qualité des solutions produites par la recherche tabou non admissible, pour le problème avec la fonction objectif $\mathcal{F}_2$.

5.2. Loi de probabilité pour le choix de l'enfant à déplacer

Déterminer une planification dans le voisinage admissible ($\mathcal{N}_f$) ou non admissible ($\mathcal{N}_i$) d'une solution revient à choisir un enfant c selon la loi de probabilité

$$P(\text{choisir } c) = \frac{f(c)^\lambda}{\sum_{c\in\mathcal{C}} f(c)^\lambda}, \quad \forall c \in \mathcal{C}$$

et à le déplacer dans un bus qui ne le transporte pas. Cette loi de probabilité présente un défaut : les enfants dont la perte de temps est nulle ($f(c) = 0$) ne sont jamais choisis, sauf si l'on pose $\lambda = 0$ (dans ce cas, tous les enfants ont la même probabilité de se faire déplacer, ce qui n'est pas souhaitable). Pour que la probabilité de déplacer les enfants sans perte de temps soit non nulle, nous proposons d'ajouter une constante $m > 0$ à la perte de temps de chaque enfant. La nouvelle loi de probabilité à utiliser pour le choix de l'enfant c à déplacer est donc :

$$P(\text{choisir } c) = \frac{(f(c)+m)^\lambda}{\sum_{c\in\mathcal{C}} (f(c)+m)^\lambda}, \quad \forall c \in \mathcal{C}. \tag{5.1}$$

Des tests ont été effectués sur quatre instances avec l'heuristique Tab $\mathcal{N}_i$ pour $\lambda \in \{0.5, 1, \sqrt{2}\}$ et $m \in \{0, 0.1, 1\}$. Pour chaque paire de paramètres (λ, m), dix germes distincts du générateur pseudo-aléatoire ont été utilisés ainsi que les valeurs suivantes des paramètres d'optimisation :

- ▷ taille des sous-voisinages considérés : $\nu = 18$;
- ▷ longueur de la liste tabou : $\zeta = 10$;
- ▷ coefficient de pénalité associé à la fonction objectif $\mathcal{F}$: $\kappa_0 = 1.02$;
- ▷ probabilité de choisir un voisin dans $\mathcal{N}_i$: $\varphi_0 = 0.7$;
- ▷ coefficient de diminution de la probabilité φ_0 de choisir le voisinage non admissible : $\psi = 0.95$;
- ▷ nombre maximum d'évaluations de planifications consécutives sans amélioration de la meilleure planification rencontrée : $\eta_{max} = 7\,500$.

Les valeurs des meilleures planifications obtenues sont reportées dans la table 5.2 pour la fonction objectif $\mathcal{F}_1$ et dans la table 5.3 pour la fonction objectif $\mathcal{F}_2$.

		$F_{32}(3)$	$F_{32}(6)$	$F_{52}(9)$	$F_{52}(13)$
$\lambda = 0.5$	$m = 0$	14.76	2.50	11.29	5.01
	$m = 0.1$	12.19	2.10	9.75	4.12
	$m = 1$	**11.68**	1.68	9.47	4.95
$\lambda = 1$	$m = 0$	14.22	2.89	9.95	4.62
	$m = 0.1$	12.11	2.02	**8.58**	4.46
	$m = 1$	11.86	**1.61**	8.76	**4.05**
$\lambda = \sqrt{2}$	$m = 0$	12.11	3.01	8.67	4.73
	$m = 0.1$	12.19	2.10	9.75	4.12
	$m = 1$	11.82	2.13	10.13	4.15

TAB. 5.2 – Pertes de temps moyennes (en minutes) des planifications obtenues avec l'heuristique Tab $\mathcal{N}_i$ en utilisant la loi de probabilité (5.1) pour le choix de l'enfant à déplacer.

Bien que ces résultats présentent d'importantes fluctuations, l'utilisation d'une valeur non nulle du paramètre m semble être un choix judicieux. En effet, avec $m > 0$ (et pour toute valeur

		$F_{32}(3)$	$F_{32}(6)$	$F_{52}(9)$	$F_{52}(13)$
$\lambda = 0.5$	$m = 0$	57	20	60	47
	$m = 0.1$	**49**	21	60	41
	$m = 1$	53	18	62	48
$\lambda = 1$	$m = 0$	53	19	**56**	46
	$m = 0.1$	55	**17**	**56**	43
	$m = 1$	50	**17**	58	43
$\lambda = \sqrt{2}$	$m = 0$	54	19	57	**38**
	$m = 0.1$	**49**	21	60	41
	$m = 1$	55	20	59	39

TAB. 5.3 – Pertes de temps maximales (en minutes) des planifications obtenues avec l'heuristique Tab $\mathcal{N}_i$ en utilisant la loi de probabilité (5.1) pour le choix de l'enfant à déplacer.

de λ) on obtient des planifications de meilleure (ou de même) qualité dans 21 cas sur 24 avec $\mathcal{F}_1$ et dans 12 cas sur 24 avec $\mathcal{F}_2$. Au niveau du paramètre λ, nous recommandons d'en rester à la valeur $\lambda = 1$.

5.3. Réduction du maximum et de la somme des pertes de temps

Lors de la recherche d'une bonne solution au problème de la planification de tournées de véhicules scolaires, deux fonctions objectif ont été considérées, l'une mesurant le maximum et l'autre la somme des pertes de temps. Dans la mesure du possible, il serait toutefois intéressant de construire des planifications «minimisant» ces deux grandeurs simultanément. Rappelons que dans le chapitre 4, le résultat obtenu pour l'instance $R_{151}(11)$, pour le problème de la minimisation du maximum des pertes de temps, présentait une distribution des pertes de temps similaire à une planification minimisant la somme des pertes de temps (voir figure 4.17). Ainsi, nous avons déjà obtenu, de manière fortuite, une solution «minimisant» conjointement $\mathcal{F}_1$ et $\mathcal{F}_2$. Dans cette même optique, une méthode permettant de tenir compte simultanément des deux fonctions objectif est proposée ci-dessous.

5.3.1. Voisinages et fonction objectif

L'idée principale consiste à procéder en deux étapes. Le problème de la minimisation du maximum des pertes de temps étant plus difficile que celui de la minimisation de la somme des pertes de temps, il semble plus judicieux de s'attaquer d'abord au problème dont la fonction objectif est $\mathcal{F}_2$, puis de déterminer, parmi toutes les planifications présentant la plus petite valeur connue Θ de la fonction objectif $\mathcal{F}_2$, celle qui minimise $\mathcal{F}_1$.

L'heuristique développée pour la seconde étape mentionnée ci-dessus est une modification de la recherche tabou décrite dans la section 3.2.5. La différence principale réside dans la notion d'admissibilité et donc dans la structure de voisinage utilisée. Dans ce nouveau contexte, une planification $\mathcal{P}$ est considérée comme admissible si elle vérifie toutes les contraintes énoncées dans la section 2.3 et si $\mathcal{F}_2(\mathcal{P}) \leq \Theta$. Par conséquent, pour qu'une solution $\mathcal{P}'$ appartienne au voisinage admissible $\mathcal{N}_f(\mathcal{P})$ d'une planification $\mathcal{P}$, il faut non seulement que la violation de capacité de $\mathcal{P}'$ ne soit pas augmentée par rapport à celle de $\mathcal{P}$, mais également que $\mathcal{F}_2(\mathcal{P}') \leq \mathcal{F}_2(\mathcal{P})$. De manière similaire, une solution $\mathcal{P}'$ du voisinage non admissible $\mathcal{N}_i(\mathcal{P})$ d'une planification $\mathcal{P}$ est

une planification pour laquelle la contrainte de capacité des bus peut être violée et pour laquelle il est envisageable d'avoir $\mathcal{F}_2(\mathcal{P}') > \Theta$.

Tout comme au chapitre 3, la modification suivante de la fonction objectif $\mathcal{F}_1$ permet de restaurer l'admissibilité des planifications rencontrées :

$$\mathcal{F}_1(\mathcal{P}) = \sum_{c \in \mathcal{C}} f(c)n(c) + \kappa \sum_{b=1}^{B} \max\{0, \mathrm{occ}^b_{\max} - Q_b\} + \vartheta \max\{0, \mathcal{F}_2(\mathcal{P}) - \Theta\}. \tag{5.2}$$

Précisons que $\kappa > 1$ et $\vartheta > 1$ sont des paramètres d'optimisation et que $\mathrm{occ}^b_{\max}$ (voir (3.7)) désigne la charge maximale du bus $b \in \mathcal{B}$ sur l'ensemble des arrêts correspondants. Dans nos heuristiques, le paramètre κ est tout d'abord initialisé à une valeur donnée $\kappa_0 > 1$. Puis, à chaque fois que l'on visite une planification non admissible au niveau de la capacité des bus, κ est multiplié par κ_0. Lorsque l'on retombe sur une planification pour laquelle la contrainte sur la capacité des bus est satisfaite, la valeur de κ est réinitialisée à κ_0. De manière analogue, le paramètre ϑ est tout d'abord initialisé à une valeur donnée $\vartheta_0 > 1$. Puis, à chaque fois que l'on visite une planification de valeur $\mathcal{F}_2$ supérieure à Θ, ϑ est multiplié par ϑ_0. Lorsque l'on retombe sur une planification présentant un maximum des pertes de temps inférieur ou égal à Θ, la valeur de ϑ est réinitialisée à ϑ_0.

Compte tenu du fait que la recherche tabou se comporte généralement mieux que le recuit simulé pour la minimisation de la fonction objectif $\mathcal{F}_2$, c'est l'heuristique Tab $\mathcal{N}_i$ qui a été retenue pour être adaptée au problème considéré. De plus, la loi de probabilité (5.1) a été utilisée pour le choix des enfants à déplacer. Cette nouvelle méthode nommée «recherche tabou max-somme non admissible» et notée $\mathrm{Tab}_{\mathcal{F}_2\mathcal{F}_1}\mathcal{N}_i$ est décrite dans l'algorithme 5.3. Notons que si l'on rencontre en cours d'optimisation une planification $\mathcal{P}$ présentant une plus petite valeur de $\mathcal{F}_2$ que celle de la meilleure planification connue, *i.e.* telle que $\mathcal{F}_2(\mathcal{P}) < \Theta$, alors on met à jour la valeur de Θ en la posant égale à $\mathcal{F}_2(\mathcal{P})$. Ainsi, en un certain sens, l'heuristique $\mathrm{Tab}_{\mathcal{F}_2\mathcal{F}_1}\mathcal{N}_i$ permet non seulement de réduire la somme des pertes de temps sans en augmenter le maximum, mais également de minimiser le maximum des pertes de temps. Précisons cependant que cette méthode a été développée pour être appliquée à des planifications déjà optimisées par rapport à la fonction objectif $\mathcal{F}_2$. Notons également que la procédure Lin-2-opt appliquée en fin d'algorithme 5.3 a aussi été modifiée de manière à ce que la planification résultante conserve une perte de temps maximale inférieure ou égale à Θ.

5.3.2. Résultats obtenus

La recherche tabou max-somme non admissible a été testée avec deux valeurs de m, deux longueurs différentes de liste tabou et cinq germes distincts du générateur de nombres pseudo-aléatoires, c'est-à-dire avec vingt jeux de paramètres. Les paramètres d'optimisation utilisés sont :

- ▷ taille des sous-voisinages considérés : $\nu = 18$;
- ▷ longueur de la liste tabou : $\zeta \in \{7, 10\}$;
- ▷ coefficient de pénalité de la violation de capacité associé à la fonction objectif $\mathcal{F}_1$: $\kappa_0 = 1.02$;
- ▷ coefficient de pénalité de la violation de la valeur du maximum des pertes de temps Θ associé à la fonction objectif $\mathcal{F}_1$: $\vartheta_0 = 1.02$;
- ▷ paramètre de la loi de probabilité (5.1) pour le choix de l'enfant à déplacer : $\lambda = 1$;
- ▷ constante ajoutée à la perte de temps de chaque enfant dans la loi de probabilité (5.1) utilisée pour le choix de l'enfant à déplacer : $m \in \{0.1, 0.5\}$;

Algorithme 5.3 Recherche Tabou Max-Somme

Donnée : $\{\mathcal{T}_b\}_{b=1}^{B}$, ν, ζ, κ_0, ϑ_0, λ, m, φ_0, ψ, η_{max}, $\mathcal{F}_1, \mathcal{F}_2$

$\eta \leftarrow 0$, $\mathcal{L} \leftarrow (\ )$, $\kappa \leftarrow \kappa_0$, $\vartheta \leftarrow \vartheta_0$, $\varphi \leftarrow \varphi_0$, $\mathcal{P} \leftarrow \{\mathcal{T}_b\}_{b=1}^{B}$, $\Theta \leftarrow \mathcal{F}_2(\mathcal{P})$, $\mathcal{P}^* \leftarrow \mathcal{P}$

Tant que $\eta < \eta_{max}$ **faire**

Si $\varphi \leq (x \sim U[0,1))$ **alors**

$\mathcal{N} \leftarrow \mathcal{N}_i$

Sinon

$\mathcal{N} \leftarrow \mathcal{N}_f$

Pour $i \in \{1, \ldots, \nu\}$ **faire**

Choisir aléatoirement un enfant c_i selon la loi (5.1) et uniformément un bus $b_i \in \overline{\mathcal{B}}^{\mathcal{P}}(c_i)$.

Soit $\mathcal{P}'_i$ la planification voisine de $\mathcal{P}$ obtenue en insérant c_i dans b_i (selon $\mathcal{N}$).

$i^* \leftarrow \text{argmin}_{i \in \{1,\ldots,\nu\}}\{\mathcal{F}_1(\mathcal{P}'_i)\}$ (∗ meilleure planification voisine ∗)

$\mathcal{P}' \leftarrow \mathcal{P}'_{i^*}$, $c \leftarrow c_{i^*}$, $b \leftarrow b_{i^*}$

Si ($\mathcal{P}'$ est admissible) **et** ($\mathcal{F}_2(\mathcal{P}') \leq \Theta$) **et** ($\mathcal{F}_1(\mathcal{P}') < \mathcal{F}_1(\mathcal{P}^*)$) **alors** (∗ meilleure plan. ∗)

$\mathcal{P}^* \leftarrow \mathcal{P}'$, $\mathcal{P} \leftarrow \mathcal{P}'$, $\eta \leftarrow 0$ (∗ avec éventuellement la fonction d'aspiration ∗)

Si $\mathcal{F}_2(\mathcal{P}') < \Theta$ **alors**

$\Theta \leftarrow \mathcal{F}_2(\mathcal{P}')$ (∗ amélioration du maximum des pertes de temps ∗)

Sinon

$\eta \leftarrow \eta + \nu$ (∗ ν planifications voisines évaluées ∗)

Si $(c, b) \notin \mathcal{L}$ **alors** (∗ mouvement non tabou ∗)

$\mathcal{P} \leftarrow \mathcal{P}'$

Si ($\mathcal{P}'$ est admissible) **et** ($\mathcal{F}_2(\mathcal{P}') \leq \Theta$) **alors**

$\kappa \leftarrow \kappa_0$, $\vartheta \leftarrow \vartheta_0$, $\varphi \leftarrow \varphi_0$

Sinon

$\kappa \leftarrow \kappa \cdot \kappa_0$, $\vartheta \leftarrow \vartheta \cdot \vartheta_0$, $\varphi \leftarrow \psi \cdot \varphi$

$\mathcal{L} \leftarrow \mathcal{L} + (c, b)$ (∗ mise à jour de la liste tabou ∗)

Si $|\mathcal{L}| > \zeta$ **alors**

effacer le plus ancien élément de $\mathcal{L}$.

Retourner Lin-2-opt$(\mathcal{P}^*)$

- ▷ probabilité de choisir un voisin dans $\mathcal{N}_i$: $\phi_0 = 0.7$;
- ▷ coefficient de diminution de la probabilité ϕ_0 de choisir le voisinage non admissible : $\psi = 0.95$;
- ▷ nombre maximum d'évaluations de planifications consécutives sans amélioration de la meilleure planification rencontrée : $\eta_{max} = 7\,500$.

La recherche tabou max-somme non admissible a été appliquée aux meilleures planifications obtenues par l'une de nos méthodes du chapitre 3 (après 50 000 évaluations de planifications) pour le problème avec la fonction objectif $\mathcal{F}_2$. Les résultats obtenus pour différentes instances sont présentés dans la table 5.4.

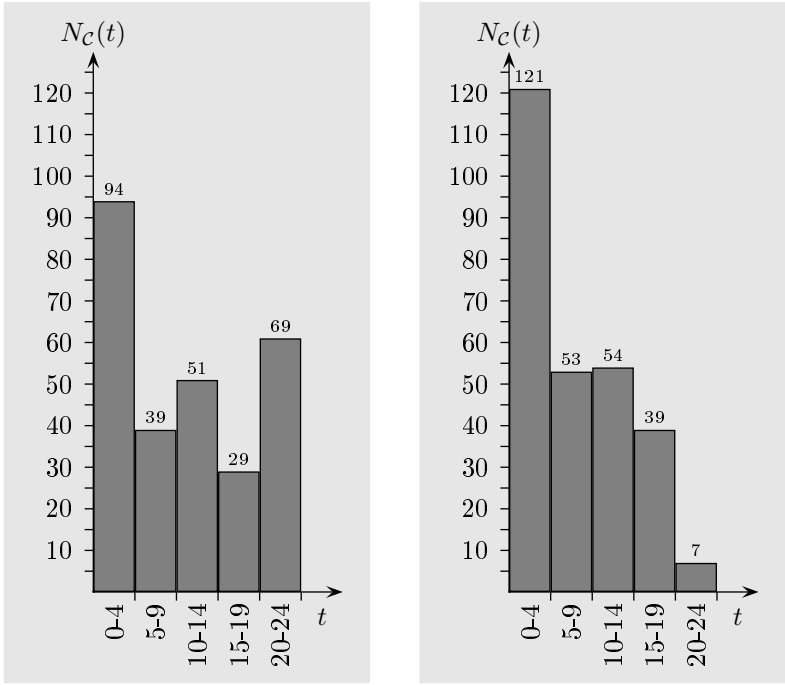
		$F_{20}(2)$	$F_{20}(3)$	$F_{32}(3)$	$F_{32}(6)$	$F_{52}(9)$	$F_{52}(13)$
Avant optimisation	$\mathcal{F}_1/N_\mathcal{C}$	10.75	7.07	21.14	6.58	20.93	13.82
	$\mathcal{F}_2$	27	21	47	17	58	41
Après optimisation	$\mathcal{F}_1/N_\mathcal{C}$	10.48	4.48	20.66	5.35	17.83	7.95
	$\mathcal{F}_2$	27	20	47	17	58	39
Pourcentage d'amélioration	$\mathcal{F}_1/N_\mathcal{C}$	2.5%	36.6%	2.3%	18.7%	14.8%	42.5%
	$\mathcal{F}_2$	0%	4.8%	0%	0%	0%	4.9%

		$F_{101}(20)$	$F_{101}(25)$	$R_{34}(4)$	$R_{34}(5)$	$R_{34}(6)$
Avant optimisation	$\mathcal{F}_1/N_\mathcal{C}$	40.36	28.92	12.43	11.66	10.58
	$\mathcal{F}_2$	103	84	29	27	24
Après optimisation	$\mathcal{F}_1/N_\mathcal{C}$	14.42	14.16	10.36	8.47	7.26
	$\mathcal{F}_2$	101	81	29	26	24
Pourcentage d'amélioration	$\mathcal{F}_1/N_\mathcal{C}$	64.3%	51%	16.7%	27.4%	31.4%
	$\mathcal{F}_2$	1.9%	3.6%	0%	3.7%	0%

TAB. 5.4 – Pertes de temps moyennes et maximales (en minutes) des planifications avant et après l'application de l'heuristique Tab$_{\mathcal{F}_2\mathcal{F}_1}\mathcal{N}_i$ et pourcentages d'amélioration correspondants.

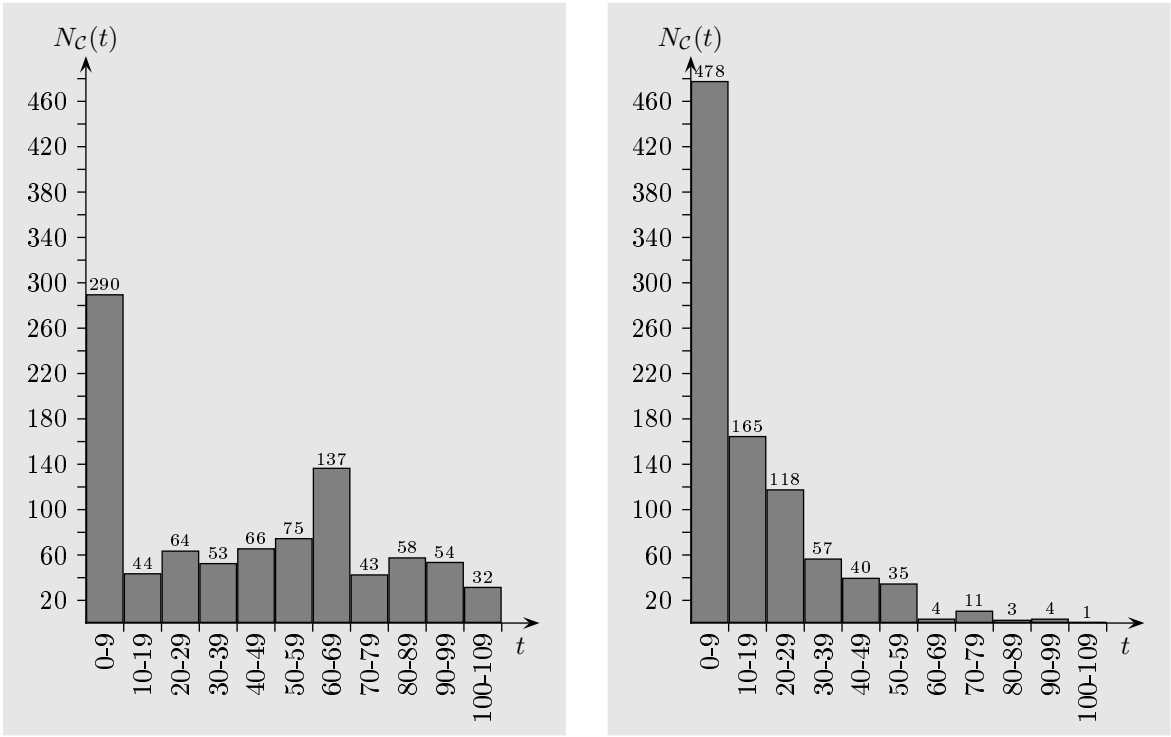
On constate que l'amélioration de la moyenne des pertes de temps varie entre 2.3% et 64.3%. De plus, dans 5 cas sur 11, non seulement la moyenne des pertes de temps a été améliorée, mais également le maximum des pertes de temps. La nouvelle structure de voisinage permet de visiter un grand nombre de planifications présentant le même maximum des pertes de temps Θ et, le cas échéant, de sortir d'un optimum local pour la fonction objectif $\mathcal{F}_2$.

La distribution des pertes de temps des enfants est représentée dans les figures 5.1 et 5.2 pour les instances $R_{34}(6)$ et $F_{105}(20)$. On note $\mathcal{P}_{\mathcal{F}_2}$, la meilleure planification obtenue à l'aide des méthodes décrites dans le chapitre 3 (après 50 000 évaluations de planifications) et $\mathcal{P}_{\mathcal{F}_2\mathcal{F}_1}$ la meilleure planification obtenue par l'application de la recherche tabou max-somme non admissible utilisant $\mathcal{P}_{\mathcal{F}_2}$ comme planification initiale. Notons que pour l'instance $R_{34}(6)$ la valeur de Θ est restée inchangée à 24 minutes, alors que pour l'instance $F_{105}(20)$ sa valeur est passée de 103 à 101 minutes. On observe l'effet attendu sur l'allure des histogrammes, c'est-à-dire un tassement sur la gauche des pertes de temps des enfants.



(a) $\mathcal{P}_{\mathcal{F}_2}$ (b) $\mathcal{P}_{\mathcal{F}_2\mathcal{F}_1}$

FIG. 5.1 – Distribution des pertes de temps des planifications obtenues pour l'instance $R_{34}(6)$.



(a) $\mathcal{P}_{\mathcal{F}_2}$ (b) $\mathcal{P}_{\mathcal{F}_2\mathcal{F}_1}$

FIG. 5.2 – Distribution des pertes de temps des planifications obtenues pour l'instance $F_{105}(20)$.

5.4. La recherche à voisinage variable

La recherche à voisinage variable est une métaheuristique nécessitant diverses structures de voisinage ainsi qu'une bonne méthode de recherche locale. Comme nous avons déjà développé deux méthodes de recherche locale précédemment, il nous a semblé intéressant d'étudier l'impact que pourrait avoir une recherche à voisinage variable sur nos méthodes.

5.4.1. Description de la méthode

La recherche à voisinage variable a été introduite en 1997 par Mladenović et Hansen (voir [MH97] et [HM03]). Elle se base sur deux observations simples :

- ▷ un optimum local par rapport à une structure de voisinage ne l'est pas forcément par rapport à une autre ;
- ▷ un optimum global est un optimum local par rapport à toutes les structures de voisinage possibles.

Dotée de diverses structures de voisinage et d'une méthode de recherche locale, la recherche à voisinage variable consiste à périodiquement recommencer la recherche locale en partant d'une nouvelle solution, dès que l'optimisation stagne en un optimum local. Afin d'explorer d'autres bassins de convergence, ce nouveau point initial est choisi dans l'un des voisinages de la meilleure solution connue. Les k_{max} structures de voisinage $\mathcal{N}_k$, $k = \{1, \ldots, k_{max}\}$ utilisées par la recherche à voisinage variable sont classées par ordre croissant des modifications qu'elles apportent à une solution donnée (notons qu'il n'est pas toujours évident de définir un tel ordre). Ainsi partant de la solution courante $\mathcal{P}$, on génère aléatoirement une solution voisine $\mathcal{P}_1 \in \mathcal{N}_1(\mathcal{P})$. Si l'optimum local $\mathcal{P}^*$ résultant de l'application de la recherche locale sur $\mathcal{P}_1$ est de moins bonne qualité que $\mathcal{P}$, alors on recommence le processus avec un voisinage qui modifie un peu plus la solution courante, *i.e.* on applique la méthode locale à $\mathcal{P}_2 \in \mathcal{N}_2(\mathcal{P})$ et ainsi de suite pour $k = 1, \ldots, k_{max}$. Si l'optimum local $\mathcal{P}^*$ résultant de l'application de la méthode de recherche locale sur la solution $\mathcal{P}_{k_{max}} \in \mathcal{N}_{k_{max}}(\mathcal{P})$ n'est pas de meilleure qualité que la solution $\mathcal{P}$, alors la métaheuristique est interrompue. En revanche, si au cours de l'optimisation on obtient un optimum local $\mathcal{P}^*$ de meilleure qualité que la planification courante $\mathcal{P}$, on recommence la recherche à partir de $\mathcal{P}^*$ avec le premier voisinage. La structure de base d'une recherche à voisinage variable (voir [MH97]) pour un problème de minimisation est détaillée dans l'algorithme 5.4.

Précisons que si la méthode de recherche locale utilise une structure de voisinage, celle-ci ne fait pas forcément partie des voisinages $\mathcal{N}_1, \ldots, \mathcal{N}_{k_{max}}$. Notons que l'algorithme 5.4 est une méthode de descente, mais qu'il est envisageable de modifier le critère de mise à jour de la solution courante, de manière à se déplacer avec une certaine probabilité vers une solution de moins bonne qualité, ou au contraire d'opter pour la meilleure solution voisine sur les k_{max} voisinages. Bien entendu, d'autres variantes et extensions de cette méthode sont également possibles (voir [HM03]).

5.4.2. Voisinages utilisés

Pour la résolution de nos problèmes, quatre structures de voisinage ont été envisagées et sont présentées ci-dessous. Les deux premières sont basées sur la même idée que dans le chapitre 3, c'est-à-dire déplacer un enfant d'un bus vers un autre. Les deux autres sont des modifications des tournées elles-mêmes.

- ▷ $\mathcal{N}_1$: voisinage admissible $\mathcal{N}_f$ décrit en section 3.2.2.

Algorithme 5.4 Recherche à voisinage variable classique

Donnée : $\mathcal{P}$ une solution initiale, $\mathcal{N}_1, \ldots, \mathcal{N}_{k_{max}}$ structures de voisinage

$k \leftarrow 1$

Tant que $k \leq k_{max}$ **faire**

Générer aléatoirement une solution voisine $\mathcal{P}' \in \mathcal{N}_k(\mathcal{P})$.

$\mathcal{P}^* \leftarrow$ **RechercheLocale**$(\mathcal{P}')$

Si $\mathcal{F}(\mathcal{P}^*) < \mathcal{F}(\mathcal{P})$ **alors**

$\mathcal{P} \leftarrow \mathcal{P}^*$, $k \leftarrow 1$

Sinon

$k \leftarrow k + 1$

Retourner $\mathcal{P}$

- ▷ $\mathcal{N}_2$: comme pour le voisinage $\mathcal{N}_f$, un enfant c est choisi selon la loi de probabilité (5.1) et est déplacé dans un bus b tiré aléatoirement et uniformément dans $\overline{\mathcal{B}}^{\mathcal{P}}(c)$, *i.e.* l'ensemble des bus ne transportant pas c dans la planification $\mathcal{P}$. On procède de la manière suivante pour déplacer l'enfant c dans le bus b : son école $s(c)$ est ajoutée en fin de tournée et sa maison $h(c)$ est insérée de manière à minimiser le temps total de trajet de la tournée au sens de l'expression (3.3), tout en vérifiant la contrainte de capacité (2.16). L'horaire de la tournée résultante est calculé à l'aide de l'algorithme 3.6. En général, l'utilisation de ce voisinage implique des modifications plus importantes de la planification que l'utilisation du voisinage $\mathcal{N}_1$. En effet, la plupart des enfants transportés par le bus b voient leur perte de temps augmenter, car l'ajout d'une nouvelle école en fin de tournée peut impliquer que leur arrivée à l'école doive être avancée. Ce voisinage a été proposé par Walter ([Wal04]).
- ▷ $\mathcal{N}_3$: ce voisinage consiste à choisir aléatoirement et uniformément deux tronçons de tournées associés à deux bus différents et à les échanger. On appelle *tronçon d'une tournée* $\mathcal{T}_b$ une suite d'arrêts $(a^b_{\alpha_1}, \ldots, a^b_{\alpha_2})$ telle que

$$\mathrm{occ}^b_{\alpha_1 - 1}(\{\mathcal{T}_b\}) = 0 \text{ et } \mathrm{occ}^b_{\alpha_2}(\{\mathcal{T}_b\}) = 0,$$

  i.e. le bus est vide en commençant et en terminant cette sous-tournée. L'algorithme 5.5 permet d'énumérer l'ensemble des tronçons d'une tournée et l'algorithme 5.6 détaille la construction d'une planification $\mathcal{P}'$ voisine de $\mathcal{P}$ selon ce voisinage.
- ▷ $\mathcal{N}_4$: ce voisinage consiste à choisir aléatoirement et uniformément un bus $b \in \mathcal{B}$ et à le vider entièrement. Tout enfant c transporté par le bus b en question est déplacé dans un des bus de l'ensemble $\overline{\mathcal{B}}^{\mathcal{P}}(c)$ en utilisant le voisinage admissible $\mathcal{N}_f$. Parmi les $B - 1$ bus envisageables, on choisit celui pour lequel on obtient la meilleure planification. La construction d'une planification $\mathcal{P}'$ voisine de $\mathcal{P}$ selon ce voisinage est détaillée dans l'algorithme 5.7.

Algorithme 5.5 Tronçons Énumération des tronçons d'une tournée

Donnée : $\mathcal{T}_b = \left(a_\alpha^b\right)_{\alpha=1}^{n_b}$

$\Upsilon_b \leftarrow \varnothing$, tronçon $\leftarrow$ vrai

Pour α **de** 1 **à** n_b **faire**

Si tronçon **alors**

$a_{\alpha_1} \leftarrow a_\alpha$, tronçon $\leftarrow$ faux (∗ premier arrêt du tronçon ∗)

Si $\mathrm{occ}_\alpha^b(\mathcal{T}_b) = 0$ **alors**

tronçon $\leftarrow$ vrai, $\Upsilon_b \leftarrow \Upsilon_b \cup (a_{\alpha_1}, \ldots, a_\alpha)$ (∗ dernier arrêt du tronçon ∗)

Retourner Υ_b

Algorithme 5.6 Voisinage $\mathcal{N}_3$: échange de deux tronçons

Donnée : $\{\mathcal{T}_b\}_{b=1}^B = \left\{\left(a_\alpha^b\right)_{\alpha=1}^{n_b}\right\}_{b=1}^B$

Tirer aléatoirement et uniformément deux bus $b, b' \in \mathcal{B}$ $(b \neq b')$.

$\Upsilon_b \leftarrow$ **Tronçons**$(\mathcal{T}_b)$, $\Upsilon_{b'} \leftarrow$ **Tronçons**$(\mathcal{T}_{b'})$,

Tirer aléatoirement et uniformément deux tronçons $(a_{\alpha_1}^b, \ldots, a_{\alpha_2}^b) \in \Upsilon_b$ et $(a_{\alpha_1'}^{b'}, \ldots, a_{\alpha_2'}^{b'}) \in \Upsilon_{b'}$

$\mathcal{T}_b \leftarrow (a_1^b, \ldots, a_{\alpha_1-1}^b, a_{\alpha_1'}^{b'}, \ldots, a_{\alpha_2'}^{b'}, a_{\alpha_2+1}^b, \ldots, a_{n_b}^b)$ (∗ échange des tronçons ∗)

$\mathcal{T}_{b'} \leftarrow (a_1^{b'}, \ldots, a_{\alpha_1'-1}^{b'}, a_{\alpha_1}^b, \ldots, a_{\alpha_2}^b, a_{\alpha_2'+1}^{b'}, \ldots, a_{n_{b'}}^{b'})$

$\mathcal{T}_b \leftarrow$ **CalculHoraire**$(\mathcal{T}_b)$, $\mathcal{T}_{b'} \leftarrow$ **CalculHoraire**$(\mathcal{T}_{b'})$

Retourner $\{\mathcal{T}_b\}_{b=1}^B$

5.4.3. Heuristique implémentée

Contrairement à la version classique de l'algorithme présentée ci-dessus, nous avons choisi de ne pas arrêter la recherche une fois tous les voisinages utilisés, mais de ne l'interrompre qu'après avoir épuisé un budget d'évaluations de planifications $\bar{\eta}_{max}^{rvv}$ donné. Une telle adaptation nécessite de disposer d'une longue suite de voisinages. Nous avons donc opté pour la séquence

$$\mathcal{N}_1^1, \ldots, \mathcal{N}_{k_{max}}^1, \mathcal{N}_1^2, \ldots, \mathcal{N}_{k_{max}}^2, \mathcal{N}_1^3, \ldots, \mathcal{N}_{k_{max}}^3, \ldots$$

où la notation $\mathcal{N}_k^j$ désigne le fait d'utiliser j fois le voisinage $\mathcal{N}_k$ (avant d'effectuer la recherche locale). Précisons que lors de l'utilisation répétée d'un même voisinage, on introduit une «liste tabou» de mouvements, réinitialisée après les j applications du voisinage $\mathcal{N}_k$, afin que ces derniers ne s'annulent pas, c'est-à-dire :

- ▷ pour les voisinages $\mathcal{N}_1$ et $\mathcal{N}_2$, si l'on a retiré un enfant c du bus b, alors on interdit de le replacer dans son bus initial b ;
- ▷ pour le voisinage $\mathcal{N}_3$, si l'on a échangé des tronçons des bus b et b', alors on interdit d'inverser à nouveau des tronçons de ces mêmes bus ;

Algorithme 5.7 Voisinage $\mathcal{N}_4$: vider un bus

Donnée : $\{\mathcal{T}_b\}_{b=1}^{B}$, $\mathcal{F} \in \{\mathcal{F}_1, \mathcal{F}_2\}$

$\mathcal{P} \leftarrow \{\mathcal{T}_b\}_{b=1}^{B}$

Choisir aléatoirement et uniformément un bus $b \in \mathcal{B}$. (∗ bus à vider ∗)

Pour tout $c \in \mathcal{C}$ tel que $\beta^{\mathcal{P}}(c) = b$ **faire**

objectif $\leftarrow \infty$

Pour tout $b' \in \overline{\mathcal{B}}^{\mathcal{P}}(c)$ **faire**

Soit $\mathcal{P}'$ la planification voisine de $\mathcal{P}$ obtenue en insérant c dans b' selon $\mathcal{N}_f$.

Si $\mathcal{F}(\mathcal{P}') <$ objectif **alors**

$\mathcal{P}^* \leftarrow \mathcal{P}'$, objectif $\leftarrow \mathcal{F}(\mathcal{P}')$

$\mathcal{P} \leftarrow \mathcal{P}^*$

Retourner $\mathcal{P}$

- ▷ pour le voisinage $\mathcal{N}_4$, si un bus b a été choisi pour être vidé, il ne pourra pas l'être à nouveau.

Afin de comparer les résultats de notre recherche à voisinage variable avec ceux obtenus à l'aide des méthodes décrites au chapitre 3, nous allons accorder à chaque type de méthode le même budget d'évaluations de planifications. Notons que pour la recherche à voisinage variable, la plus grande partie du coût (en termes d'évaluations de planifications) se situe au niveau de la méthode de recherche locale. Cependant, il nous faut également tenir compte de l'effort à fournir pour construire une solution voisine $\mathcal{P}' \in \mathcal{N}_k^j(\mathcal{P})$. Pour $k \in \{1, 2, 3\}$, la génération d'une planification voisine $\mathcal{P}' \in \mathcal{N}_k^j(\mathcal{P})$ revient à évaluer j planifications. En revanche, pour le voisinage $\mathcal{N}_4^j$, cette valeur est bien plus élevée. En effet, la construction d'une solution voisine selon le voisinage $\mathcal{N}_4^j$ revient à choisir successivement j bus différents et à les vider. Pour tout enfant à déplacer, on essaie de le placer dans les $B - 1$ bus ne le transportant pas, pour ne finalement retenir que la meilleure solution. Ainsi le nombre de planifications à évaluer est de l'ordre de $(B - 1)$ fois le nombre d'enfants à déplacer.

La méthode de recherche locale choisie est la recherche tabou non admissible décrite dans l'algorithme 3.9. Toutefois, les modifications suivantes lui ont été apportées :

- ▷ un paramètre $\bar{\eta}_{max}^{loc}$ a été ajouté, correspondant au nombre maximum d'évaluations de planifications à ne pas dépasser au cours d'une recherche locale ;
- ▷ une variable $\bar{\eta}^{rvv}$, permettant de compter le nombre total de planifications évaluées par notre recherche à voisinage variable, est mise à jour pendant la recherche tabou non admissible.

La recherche à voisinage variable implémentée, notée $\text{Rvv}_{\text{Tab } \mathcal{N}_i}$, est détaillée dans l'algorithme 5.8.

Algorithme 5.8 Recherche à voisinage variable

Donnée : $\{\mathcal{T}_b\}_{b=1}^{B}$, ν, ζ, κ_0, λ, φ_0, ψ, $\bar{\eta}^{rvv}_{max}$, η_{max}, $\bar{\eta}^{loc}_{max}$, $\mathcal{F} \in \{\mathcal{F}_1, \mathcal{F}_2\}$, $\mathcal{N}_1, \ldots, \mathcal{N}_{k_{max}}$

$\bar{\eta}^{rvv} \leftarrow 0$, $n \leftarrow 1$, $\mathcal{P} \leftarrow \{\mathcal{T}_b\}_{b=1}^{B}$

Tant que $\bar{\eta}^{rvv} < \bar{\eta}^{rvv}_{max}$ **faire**

$k \leftarrow ((n-1) \mod k_{max}) + 1$, $j \leftarrow \left\lfloor \frac{n-1}{k_{max}} \right\rfloor + 1$

$\mathcal{P}_0 \leftarrow \mathcal{P}$

Pour i **de** 1 **à** j **faire** (∗ j fois le voisinage $\mathcal{N}_k$ ∗)

Générer une planification $\mathcal{P}_i$ voisine de $\mathcal{P}_{i-1}$ selon le voisinage $\mathcal{N}_k$.

Si $k \neq 4$ **alors**

$\bar{\eta}^{rvv} \leftarrow \bar{\eta}^{rvv} + 1$

Sinon

$\bar{\eta}^{rvv} \leftarrow \bar{\eta}^{rvv} + (B-1) \sum_{\alpha=1}^{n_{b_i}} |\overline{\mathcal{C}}^{b_i}_{\alpha}|$ (∗ bus b_i vidé ∗)

$\mathcal{P}^* \leftarrow$ **RechercheTabou**$(\mathcal{P}_j, \nu, \zeta, \kappa_0, \lambda, \varphi_0, \psi, \eta_{max}, \bar{\eta}^{loc}_{max}, \bar{\eta}^{rvv}, \mathcal{F})$

Si $\mathcal{F}(\mathcal{P}^*) < \mathcal{F}(\mathcal{P})$ **alors**

$\mathcal{P} \leftarrow \mathcal{P}^*$, $n \leftarrow 1$

Sinon

$n \leftarrow n + 1$

Retourner $\mathcal{P}$

5.4.4. Valeurs des meilleures planifications obtenues

L'algorithme 5.8 a été testé sur nos deux problèmes avec vingt jeux de paramètres (dix germes distincts du générateur pseudo-aléatoire et deux longueurs de liste tabou). Les valeurs sélectionnées des paramètres d'optimisation sont :

- ▷ taille des sous-voisinages considérés : $\nu = 18$;
- ▷ longueur de la liste tabou : $\zeta \in \{7, 10\}$;
- ▷ coefficient de pénalité associé à la fonction objectif $\mathcal{F}$: $\kappa_0 = 1.02$;
- ▷ paramètre de la loi de probabilité (5.1) pour le choix de l'enfant à déplacer : $\lambda = 1$;
- ▷ constante ajoutée à la perte de temps de chaque enfant dans la loi de probabilité (5.1) utilisée pour le choix de l'enfant à déplacer : $m = 0$;[1]

[1] Bien que l'on ait constaté en section 5.2 que l'on pouvait espérer de meilleurs résultats avec $m > 0$, la valeur du paramètre m a été fixée à zéro. Ce choix nous permettra de comparer les résultats de notre recherche à voisinage variable avec ceux obtenus à l'aide des méthodes décrites dans le chapitre 3.

- ▷ probabilité de choisir un voisin dans $\mathcal{N}_i$: $\varphi_0 = 0.7$;
- ▷ coefficient de diminution de la probabilité φ_0 de choisir le voisinage non admissible : $\psi = 0.95$;
- ▷ nombre maximum d'évaluations de planifications consécutives sans amélioration de la meilleure planification rencontrée dans la recherche locale : $\eta_{max} = 3\,000$;
- ▷ nombre maximum d'évaluations de planifications autorisé pendant une recherche locale : $\bar{\eta}^{loc}_{max} = 5\,000$;
- ▷ nombre maximum d'évaluations de planifications autorisé durant tout l'algorithme de recherche à voisinage variable : $\bar{\eta}^{rvv}_{max} = 50\,000$.

La table 5.5 contient, pour différentes instances, les pertes de temps moyennes et maximales obtenues pour les solutions suivantes.

- ▷ La meilleure planification obtenue par l'application de l'heuristique Tab $\mathcal{N}_i$ à la solution initiale générée selon la procédure décrite dans la section 3.1. Pour chaque instance, sur les vingt exécutions (jeux de paramètres) de la recherche tabou non admissible effectuées, la planification présentant la meilleure fonction objectif a été retenue. Notons que le critère d'arrêt a été fixé à un budget de 50 000 évaluations de planifications.
- ▷ La meilleure planification obtenue par l'application de l'heuristique $\text{Rvv}_{\text{Tab}\ \mathcal{N}_i}$. Comme pour la recherche tabou non admissible, pour chaque instance on a retenu la meilleure planification construite à l'aide d'un des vingt jeux de paramètres considérés, toujours avec un critère d'arrêt fixé à $\bar{\eta}^{rvv}_{max} = 50\,000$ évaluations de planifications.

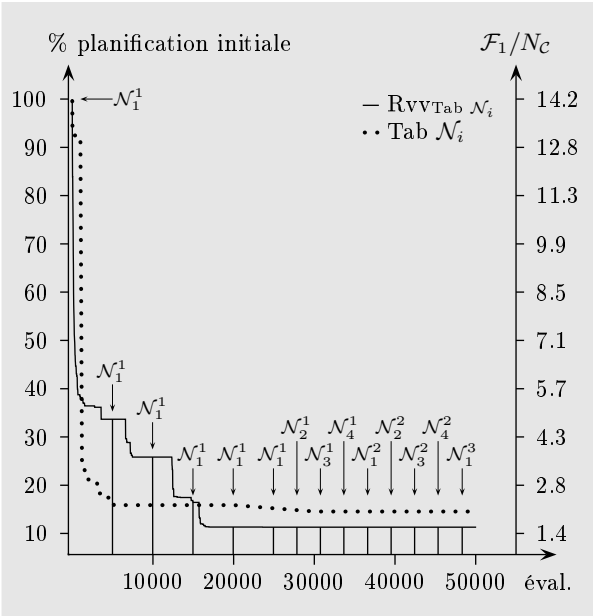
		$F_{32}(3)$	$F_{32}(6)$	$F_{52}(9)$	$F_{52}(13)$
$\mathcal{F}_1/N_{\mathcal{C}}$	Tab $\mathcal{N}_i$	13.1	2.1	10.0	4.2
	$\text{Rvv}_{\text{Tab}\ \mathcal{N}_i}$	11.9	1.6	9.2	3.5
	% d'amélioration	9.2%	23.8%	8%	16.7%
$\mathcal{F}_2$	Tab $\mathcal{N}_i$	53	19	51	37
	$\text{Rvv}_{\text{Tab}\ \mathcal{N}_i}$	51	17	50	35
	% d'amélioration	3.8%	10.5%	2.0%	5.4%

		$F_{105}(20)$	$F_{105}(25)$	$R_{34}(4)$	$R_{34}(5)$	$R_{34}(6)$
$\mathcal{F}_1/N_{\mathcal{C}}$	Tab $\mathcal{N}_i$	12.0	7.9	9.1	4.8	4.3
	$\text{Rvv}_{\text{Tab}\ \mathcal{N}_i}$	12.5	7.7	8.9	4.7	3.9
	% d'amélioration	-4.2%	2.5%	2.2%	2.1%	9.3%
$\mathcal{F}_2$	Tab $\mathcal{N}_i$	93	74	29	25	22
	$\text{Rvv}_{\text{Tab}\ \mathcal{N}_i}$	93	72	29	24	23
	% d'amélioration	0%	2.7%	0%	4%	-4.5%

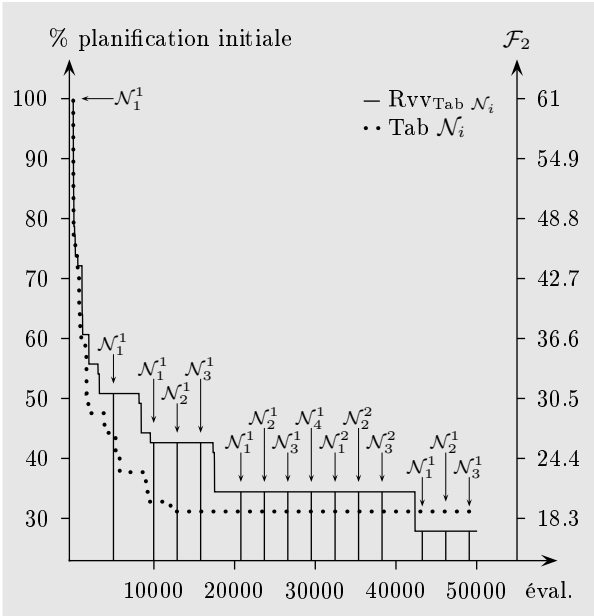
TAB. 5.5 – Pertes de temps moyennes et maximales (en minutes) obtenues avec les heuristiques Tab $\mathcal{N}_i$ et $\text{Rvv}_{\text{Tab}\ \mathcal{N}_i}$ après 50 000 évaluations de planifications et pourcentages d'amélioration correspondants.

En comparant les résultats obtenus en utilisant l'heuristique $\text{Rvv}_{\text{Tab}\ \mathcal{N}_i}$ et l'heuristique Tab $\mathcal{N}_i$ (qui n'est autre que la méthode de recherche locale de notre recherche à voisinage variable) pour un même budget de 50 000 évaluations de planifications, on constate dans la plupart des cas une amélioration des valeurs des meilleures planifications construites par notre recherche à voisinage variable.

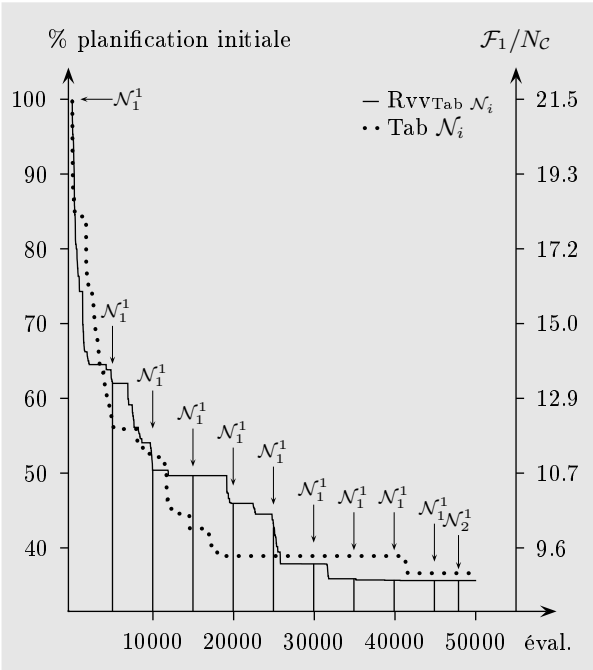
La figure 5.3 montre, pour nos deux fonctions objectif, l'évolution au cours du temps de la qualité des meilleures planifications obtenues pour les instances $F_{32}(6)$ et $F_{101}(25)$. On constate



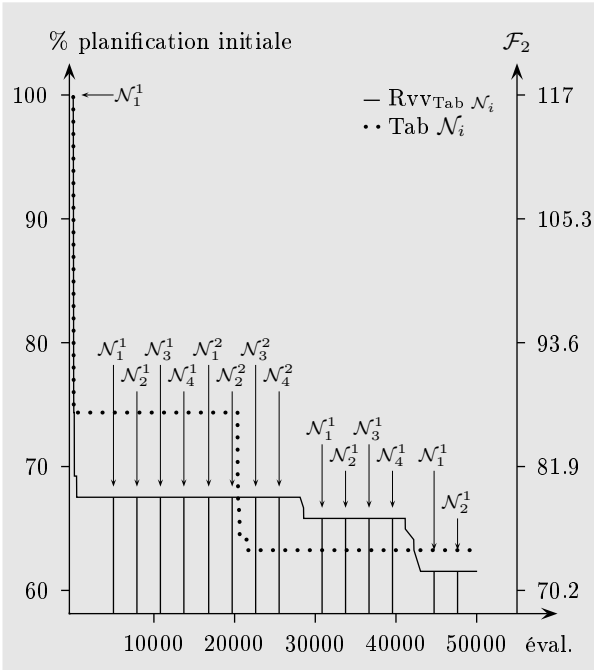
(a) $\mathcal{F}_1$ pour $F_{32}(6)$



(b) $\mathcal{F}_2$ pour $F_{32}(6)$



(c) $\mathcal{F}_1$ pour $F_{105}(25)$



(d) $\mathcal{F}_2$ pour $F_{105}(25)$

FIG. 5.3 – Évolution, en fonction du nombre d'évaluations de planifications, de la qualité (en minutes) des meilleures planifications obtenues à l'aide des heuristiques RVVTab $\mathcal{N}_i$ et Tab $\mathcal{N}_i$ pour les instances $F_{32}(6)$ et $F_{105}(25)$. Les traits verticaux correspondent à la réinitialisation de la recherche locale et à un éventuel changement de voisinage.

que l'heuristique Tab $\mathcal{N}_i$ est généralement plus efficace que l'heuristique $\mathrm{Rvv}_{\mathrm{Tab}\ \mathcal{N}_i}$ durant la partie initiale de l'optimisation. Par contre, avec la fonction objectif $\mathcal{F}_2$, l'utilisation de différents voisinages semble permettre à l'heuristique $\mathrm{Rvv}_{\mathrm{Tab}\ \mathcal{N}_i}$ de sortir plus facilement d'un minimum local. Avec la fonction objectif $\mathcal{F}_1$, les différents voisinages ne sont que peu utilisés. En effet, durant la phase initiale de l'optimisation, la recherche à voisinage variable utilise essentiellement le voisinage $\mathcal{N}_1$ qui est justement l'un des deux voisinages utilisés par l'heuristique Tab $\mathcal{N}_i$ (dans ce cas, les heuristiques $\mathrm{Rvv}_{\mathrm{Tab}\ \mathcal{N}_i}$ et Tab $\mathcal{N}_i$ sont donc grossièrement équivalentes).

En conclusion, la recherche à voisinage variable semble permettre d'obtenir des planifications de bonne qualité, du moins après un grand nombre d'évaluations de planifications. Toutefois, si le temps à disposition pour l'optimisation est limité, il semble plus judicieux d'opter pour l'une des méthodes de recherche locale du chapitre 3.

CHAPITRE 6

Décomposition des problèmes de grande taille

Dans ce chapitre, afin de réduire le temps de calcul nécessaire à la résolution de problèmes de grande taille, nous proposons une approche basée sur la décomposition du problème en sous-problèmes. Chaque sous-problème comprend un certain nombre de bus ainsi qu'un sous-ensemble d'enfants à transporter et est traité à l'aide d'une des méthodes développées au chapitre 3. Toutefois, durant l'optimisation, la décomposition évolue par le biais de déplacements d'enfants d'un sous-problème à un autre. Le comportement de cette heuristique est étudié et les résultats obtenus sont comparés à ceux des méthodes présentées dans les chapitres précédents.

6.1. Décomposition en sous-problèmes

Les méthodes proposées dans les chapitres 3 et 5 permettent de résoudre en des temps de calcul raisonnables des problèmes de la taille de ceux que l'on rencontre dans des «groupements scolaires», *i.e.* des écoles regroupées au sein d'une même petite zone géographique. Comme nous l'avons constaté dans la section 4.4.4, cette approche ne permet pas de traiter en un temps raisonnable des problèmes de grande taille. Il convient donc d'étudier une méthode permettant d'accélérer la construction de planifications de bonne qualité pour des problèmes que l'on peut rencontrer dans une métropole ou une grande région.

6.1.1. Choix du type de décomposition

Les caractéristiques d'une approche idéale par décomposition sont :

- ▷ la réduction du temps de calcul global ;
- ▷ l'obtention d'une planification d'une qualité proche de celle que l'on pourrait obtenir (beaucoup plus lentement) sans décomposer le problème.

Temps de calcul. Cette mesure dépend notamment des caractéristiques suivantes.

- ▷ Comme nous l'avons discuté en page 58, le nombre d'enfants à transporter par bus est un facteur influençant le nombre moyen d'arrêts d'une tournée et par conséquent le temps de calcul nécessaire pour évaluer une planification voisine.
- ▷ Le nombre d'enfants total à transporter joue également un rôle. En effet, comme l'illustre la figure 4.21, même lorsque le rapport entre la place qu'occupent les enfants et la somme des capacités des bus est constant, le temps de calcul évolue de manière super-linéaire en fonction du nombre d'enfants à transporter.
- ▷ D'une manière générale, il semble judicieux d'équilibrer la taille des sous-problèmes. Schématiquement il est généralement plus rapide de résoudre deux problèmes de taille moyenne que de traiter un petit et un grand problème.

Qualité de la solution. La qualité de la planification globale, constituée de la réunion des planifications obtenues après avoir résolu chacun des sous-problèmes, dépend de divers facteurs.

- ▷ Idéalement, la place totale qu'occupent les enfants affectés à un sous-problème doit être proportionnée par rapport à la somme des capacités des bus correspondants. On évite ainsi que certains bus effectuent de multiples trajets et que des enfants doivent partir très tôt de chez eux.
- ▷ Les enfants se rendant généralement à l'école la plus proche de leur maison, il est raisonnable de travailler avec des zones géographiques connexes.
- ▷ Il convient également de définir des sous-problèmes présentant approximativement la même taille. En effet, un sous-problème de grande taille, même s'il n'est pas constitué judicieusement (par exemple au niveau du découpage en zones géographiques), nous offre une certaine flexibilité pour déterminer une solution acceptable. En revanche, pour un sous-problème de petite taille la marge de manœuvre est généralement plus limitée.

Caractéristiques souhaitées. Au vu des considérations ci-dessus, nous allons nous efforcer de construire une décomposition en sous-problèmes présentant les caractéristiques suivantes :

(1) pour chaque sous-problème un rapport similaire entre la place totale qu'occupent les enfants et la somme des capacités des bus ;

(2) approximativement le même nombre d'enfants dans chaque sous-problème ;

(3) des sous-problèmes correspondant à des régions connexes.

Typiquement deux approches de décomposition du problème sont envisageables :

- ▷ par rapport aux écoles, *i.e.* tous les enfants devant se rendre en un certain sous-ensemble d'écoles géographiquement proches sont rassemblés en un même sous-problème ;
- ▷ par rapport aux enfants, *i.e.* les enfants habitant en un certain sous-ensemble de maisons géographiquement proches sont rassemblés en un même sous-problème.

Comparons ces deux approches au niveau des trois caractéristiques souhaitées, mentionnées ci-dessus.

(1) Obtention de rapports similaires entre les places qu'occupent les enfants et la somme des capacités des bus affectés à chaque sous-problème : le critère est plus difficile à atteindre pour la décomposition par rapport aux écoles que par rapport aux enfants. En effet, la répartition des enfants est plus aisée, si l'on considère chaque enfant séparément que si l'on affecte en bloc tous les enfants se rendant en une même école.

(2) Pour des mêmes questions de granularité, il est également plus facile de générer des sous-problèmes comprenant approximativement le même nombre d'enfants lorsque l'on utilise une décomposition par rapport aux enfants que par rapport aux écoles, en particulier en présence d'écoles aux effectifs disparates (voir figure 6.1).

(3) Au niveau de la génération de sous-problèmes correspondant à des régions géographiques connexes, les deux approches sont équivalentes.

Notons que la planification optimale du problème entier peut toujours s'exprimer comme une décomposition par rapport aux enfants, mais généralement pas comme une décomposition par rapport aux écoles. La décomposition par rapport aux enfants semble donc plus judicieuse et c'est celle que nous avons adoptée.

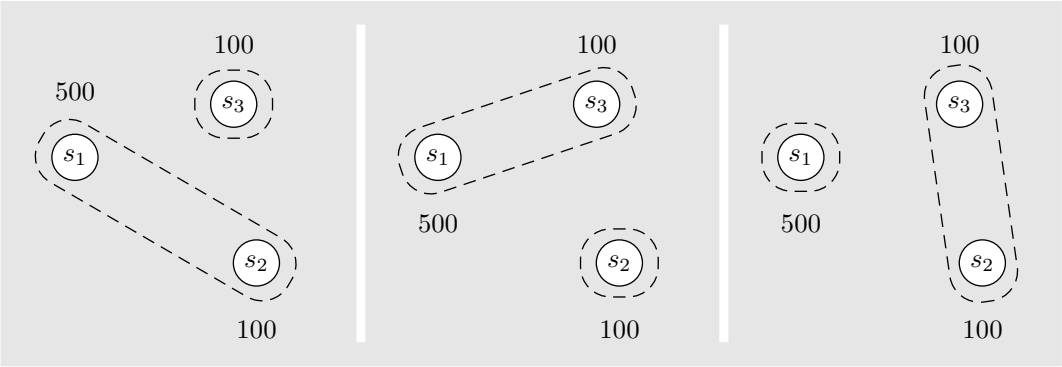


FIG. 6.1 – Exemple typique de répartition problématique des enfants en deux sous-problèmes, en cas d'utilisation d'une décomposition par rapport aux écoles.

Nombre de sous-problèmes. Le nombre idéal de sous-problèmes à considérer dépend du nombre d'enfants à transporter et du nombre de bus disponibles. *A priori*, ce nombre peut varier entre 1 et B, le nombre de bus disponibles (un bus étant affecté à une et une seule région). Notons que si le nombre de sous-problèmes est égal à 1 ou B, alors on se retrouve dans le cas d'un problème sans décomposition. Il convient donc de faire varier le nombre de sous-problèmes entre 2 et $\lfloor B/2 \rfloor$, la moitié du nombre de bus disponibles.

Le nombre optimal de sous-problèmes à considérer dépend de l'instance à résoudre, mais on peut tout de même présager que si le nombre de sous-problèmes est trop petit, la réduction du temps de calcul risque d'être limitée et que si le nombre de régions est trop important, la qualité de la solution pourrait souffrir d'un cloisonnement trop radical. Une étude concernant l'influence du nombre de régions sur la qualité des planifications obtenues est présentée en section 6.6.3.

6.1.2. Notation

Rappelons quelques notations :

- ▷ $G = (\mathcal{V}, \mathcal{E}, \ell)$ est un réseau simple, où $\mathcal{V}$ désigne l'ensemble des sommets, $\mathcal{E}$ l'ensemble des arêtes et où ℓ est une fonction associant un coût non négatif à chaque arête ;
- ▷ $d(i, j)$ est le temps de parcours sur un plus court chemin dans le réseau G entre les sommets i et j ;
- ▷ $\mathcal{H} \subseteq \mathcal{V}$ désigne l'ensemble des maisons ;
- ▷ $\mathcal{S} \subseteq \mathcal{V}$ désigne l'ensemble des écoles ;
- ▷ $\mathcal{C}$ désigne l'ensemble des enfants à transporter ;
- ▷ $q(c)$ désigne la place occupée par l'enfant c dans un bus ;
- ▷ $\mathcal{B}$ désigne l'ensemble des bus disponibles, avec $B = |\mathcal{B}|$.

On introduit également les nouvelles notations suivantes :

- ▷ D désigne le nombre de sous-problèmes à construire ;
- ▷ $\mathcal{G}_i \subset \mathcal{C}$ désigne le *groupe d'enfants* à transporter pour le sous-problème $i \in \{1, \ldots, D\}$ (on suppose $\mathcal{G}_i \neq \varnothing$) ;
- ▷ $\mathcal{D} = \{\mathcal{G}_1, \ldots, \mathcal{G}_D\}$ est une *décomposition* : il s'agit d'une partition de l'ensemble des enfants ;
- ▷ $g(c) \in \{1, \ldots, D\}$ désigne l'indice du groupe contenant l'enfant c ;

- ▷ $Q(\mathcal{G}_i)$ désigne la place qu'occupe un groupe d'enfants $\mathcal{G}_i \subset \mathcal{C}$ dans un bus :

$$Q(\mathcal{G}_i) = \sum_{c \in \mathcal{G}_i} q(c)\,;$$

- ▷ $\mathcal{B}_i \subset \mathcal{B}$ (avec $\mathcal{B}_i \neq \varnothing$) désigne l'ensemble des bus affectés au i^{e} sous-problème ;
- ▷ on appelle *région* l'ensemble $\mathcal{V}_{\mathcal{G}_i} \subseteq \mathcal{V}$ des sommets correspondant aux maisons et aux écoles des enfants du groupe $\mathcal{G}_i$, *i.e.*

$$\mathcal{V}_{\mathcal{G}_i} = \{h(c) \mid c \in \mathcal{G}_i\} \cup \{s(c) \mid c \in \mathcal{G}_i\}, \quad \forall i \in \{1, \dots, D\}.$$

  Notons que deux régions distinctes $\mathcal{V}_{\mathcal{G}_i}$ et $\mathcal{V}_{\mathcal{G}_j}$ ne sont pas forcément disjointes. En effet, lorsque des enfants appartenant à deux groupes différents se rendent en une même école, le sommet correspondant est commun aux deux régions.

6.1.3. Algorithme de décomposition

Dans cette section nous présentons une approche permettant de construire une décomposition en D sous-problèmes initiaux. Nous allons procéder en trois étapes :

- ▷ partitionner les sommets en D zones géographiques connexes ;
- ▷ répartir équitablement les bus entre les zones géographiques ;
- ▷ modifier cette répartition initiale, tout en maintenant les zones connexes, de manière à ce que le rapport entre la place qu'occupent les enfants et la somme des capacités des bus associés à une zone soit similaire pour chacun des D sous-problèmes.

Étape I. Pour construire D zones géographiques $\{\mathcal{W}_1, \dots, \mathcal{W}_D\} \subset V$ connexes, D sommets de $\mathcal{V}$ particuliers (à partir desquels nous allons faire croître des régions) sont choisis. Ces sommets particuliers sont sélectionnés l'un après l'autre, de manière à maximiser la distance au sommet particulier (déjà choisi) le plus proche.

Dans une seconde phase, parmi les sommets de $\mathcal{V}$ (non encore affectés à une des zones) adjacents à une des zones $\{\mathcal{W}_1, \dots, \mathcal{W}_D\}$, un sommet v est choisi aléatoirement et uniformément. Ce sommet est rattaché à l'ensemble de sommets $\mathcal{W}_i$ le plus proche, où la *distance d'un sommet v adjacent à un ensemble de sommets $\mathcal{W}_i$* est définie par

$$\Delta(v, \mathcal{W}_i) = \min_{w \in \mathcal{W}_i} \ell(v, w).$$

Cette seconde phase se termine lorsque chaque sommet de $\mathcal{V}$ a été affecté à une zone. On obtient finalement une répartition initiale des enfants en D zones géographiquement connexes. Cette première étape de la décomposition est détaillée dans l'algorithme 6.1.

Étape II. Afin d'équilibrer autant que possible la taille des sous-problèmes, les bus sont distribués de manière à ce que la somme de leurs capacités soit approximativement la même pour chaque sous-problème. La construction des ensembles $\mathcal{B}_i$ ($i \in \{1, \dots, D\}$) est effectuée à l'aide d'une heuristique constructive, en partant d'ensembles $\mathcal{B}_i$ initialement vides. Plus précisément, les bus sont considérés dans l'ordre décroissant de leur capacité et affectés à l'ensemble $\mathcal{B}_i$ dont la somme des capacités est la plus petite. La répartition obtenue n'est pas forcément optimale, mais s'avère suffisante pour les besoins de notre problème. Le détail de cette répartition est donné dans l'algorithme 6.2. Notons que si tous les bus sont de même capacité, le nombre de bus affectés à chaque région est le même à au plus une unité près.

Algorithme 6.1 Étape I de la décomposition : création de zones géographiques connexes

Donnée : $G = (\mathcal{V}, \mathcal{E}, \ell)$, D, $\mathcal{C}$

$(w_1, w_2) \leftarrow \underset{v_i, v_j \in \mathcal{V}}{\operatorname{argmax}}\, d(v_i, v_j)$ (∗ deux sommets les plus éloignés ∗)

$\mathcal{W}_1 \leftarrow \{w_1\}$, $\mathcal{W}_2 \leftarrow \{w_2\}$, $\mathcal{V} \leftarrow \mathcal{V} \setminus \{w_1, w_2\}$

Pour i **de** 3 **à** D **faire** (∗ choix des autres sommets particuliers ∗)

$w_i \leftarrow \underset{v_i \in \mathcal{V}}{\operatorname{argmax}} \min_{j \in \{1, \dots, i-1\}} d(v_i, w_j)$

$\mathcal{W}_i \leftarrow \{w_i\}$, $\mathcal{V} \leftarrow \mathcal{V} \setminus \{w_i\}$

Tant que $\mathcal{V} \neq \varnothing$ **faire** (∗ affectation des sommets aux zones ∗)

Choisir aléatoirement et uniformément $v \in \mathcal{V}$ adjacent à un sommet de $\bigcup_{i=1}^{D} \mathcal{W}_i$.

$i \leftarrow \underset{j \in \{1, \dots, D\}}{\operatorname{argmin}} \Delta(v, \mathcal{W}_j)$ (∗ ensemble le plus proche de v ∗)

$\mathcal{W}_i \leftarrow \mathcal{W}_i \cup \{v\}$, $\mathcal{V} \leftarrow \mathcal{V} \setminus \{v\}$

Pour i **de** 1 **à** D **faire** (∗ création des groupes d'enfants ∗)

$\mathcal{G}_i \leftarrow \{c \in \mathcal{C} \mid h(c) \in \mathcal{W}_i\}$

Retourner $\mathcal{D} \leftarrow \{\mathcal{G}_1, \dots, \mathcal{G}_D\}$

Algorithme 6.2 Étape II de la décomposition : répartition équilibrée des bus

Donnée : $\mathcal{B}$, D

Pour tout $i \in \{1, \dots, D\}$ **faire**

$\mathcal{B}_i \leftarrow \varnothing$

Soit $\mathcal{L}_{\mathcal{B}} = (b'_1, \dots, b'_B)$ la liste des bus $b \in \mathcal{B}$ ordonnée par ordre décroissant de leur capacité.

Pour j **de** 1 **à** B **faire**

Attribuer le bus b'_j à l'ensemble $\mathcal{B}_i$ dont la somme des capacités est minimale.

Retourner $\{\mathcal{B}_i\}_{i=1}^{D}$

Étape III. Il reste encore à modifier la répartition initiale construite durant l'étape I, de manière à ce que le nombre d'enfants dans chaque groupe soit approximativement proportionnel à la somme des capacités des bus associés au sous-problème correspondant. Afin que la répartition soit équitable, il convient de faire en sorte que *l'excès de charge dans le i^e sous-problème*, grandeur définie par

$$E(\mathcal{G}_i) = Q(\mathcal{G}_i) - Q(\mathcal{C}) \cdot \frac{\sum_{b \in \mathcal{B}_i} Q_b}{\sum_{b \in \mathcal{B}} Q_b}, \tag{6.1}$$

soit aussi proche de zéro que possible[1].

Cette troisième étape consiste donc à déplacer des maisons d'une région présentant un excès de charge positif vers une région où l'excès de charge est négatif. Soit $\mathcal{G}_i$ un groupe avec $E(\mathcal{G}_i) < 0$ (on considère les groupes dans l'ordre croissant de leur excès de charge). Pour toute maison $h \in \mathcal{H} \setminus \mathcal{V}_{\mathcal{G}_i}$, on teste le transfert de $h \in \mathcal{V}_{\mathcal{G}_j}$ de la région $\mathcal{V}_{\mathcal{G}_j}$ vers la région $\mathcal{V}_{\mathcal{G}_i}$. Ce changement n'est effectué que si les régions obtenues $\mathcal{V}_{\mathcal{G}_j} \setminus \{h\}$ et $\mathcal{V}_{\mathcal{G}_i} \cup \{h\}$ restent connexes et si le maximum des valeurs absolues des excès de charge des sous-problèmes i et j diminue. Lorsque plusieurs maisons permettent une telle amélioration, on ne retient que celle qui est la plus proche d'un des sommets de $\mathcal{V}_{\mathcal{G}_i}$. Cette troisième étape est interrompue lorsqu'il n'existe plus de mouvement satisfaisant les conditions ci-dessus. Ce processus est détaillé dans l'algorithme 6.3.

6.2. Résolution d'un problème décomposé

Comme pour la résolution d'un problème sans décomposition, nous allons dans un premier temps construire une planification admissible pour chacun de nos sous-problèmes puis essayer de les améliorer.

6.2.1. Construction d'une planification admissible

Pour construire une solution initiale admissible de notre problème, il suffit de générer D planifications en appliquant la méthode décrite en section 3.1 aux D sous-problèmes. Ces derniers étant indépendants (chaque enfant et chaque bus ne faisant partie que d'un et un seul sous-problème), il suffit de considérer la réunion des solutions admissibles des D sous-problèmes pour obtenir une planification admissible du problème en son entier, *i.e.*

$$\mathcal{P} = \bigcup_{i=1}^{D} \{\mathcal{T}_b\}_{b \in \mathcal{B}_i}.$$

6.2.2. Amélioration des planifications

Dans cette section, nous présentons une approche permettant d'améliorer la qualité d'une planification (obtenue par exemple en appliquant la procédure ci-dessus). Cette méthode agit cycliquement à deux niveaux, celui du problème global et celui des sous-problèmes.

- ▷ Problème global : remise en question de la délimitation des sous-problèmes, *i.e.* des groupes d'enfants $\mathcal{G}_1, \ldots \mathcal{G}_D$;
- ▷ Sous-problèmes : optimisation indépendante de chaque sous-problème par l'une des méthodes développées aux chapitres 3 et 5.

[1]Selon certains auteurs, il s'agirait d'un problème de réduction de discrépance.

Algorithme 6.3 Étape III de la décomposition : équilibrage des sous-problèmes

Donnée : $G = (\mathcal{V}, \mathcal{E}, \ell)$, $\mathcal{D}$, $\mathcal{C}$, $\{\mathcal{B}_i\}_{i=1}^{D}$

continuer $\leftarrow$ vrai

Tant que continuer **faire**

continuer $\leftarrow$ faux

Soit σ la permutation ordonnant les partitions par ordre croissant des excès de charge (6.1).

$j \leftarrow 1$, distMin $\leftarrow \infty$

Tant que $\neg$ continuer **et** $j \leq D$ **faire**

$i \leftarrow \sigma^{-1}(j)$ (∗ j^{e} plus petit excès de charge ∗)

Pour tout $h \in \mathcal{H} \setminus \mathcal{V}_{\mathcal{G}_i}$ **faire**

Soit k la région telle que $h \in \mathcal{V}_{\mathcal{G}_k}$.

Si ($\mathcal{V}_{\mathcal{G}_i} \cup \{h\}$ est connexe) **et** ($\mathcal{V}_{\mathcal{G}_k} \setminus \{h\}$ est connexe) **alors**

$\Delta_h \leftarrow Q\left(\{c \in \mathcal{C} \mid h(c) = h\}\right)$ (∗ place occupée par les enfants à déplacer ∗)

excèsMax $\leftarrow \max\left\{|E(\mathcal{G}_i)|, |E(\mathcal{G}_k)|\right\}$

Si ($|E(\mathcal{G}_i) + \Delta_h| \leq$ excèsMax) **et** ($|E(\mathcal{G}_k) - \Delta_h| \leq$ excèsMax) **alors**

Si $\max\limits_{v \in \mathcal{V}_{\mathcal{G}_i}} d(h, v) <$ distMin **alors** (∗ excès de charge amélioré ∗)

distMin $\leftarrow \max\limits_{v \in \mathcal{V}_{\mathcal{G}_i}} d(h, v)$, $h^* \leftarrow h$

Si distMin $\neq \infty$ **alors** (∗ il existe un mouvement améliorant (6.1) ∗)

$\mathcal{G}_k \leftarrow \mathcal{G}_k \setminus \{c \in \mathcal{C} \mid h(c) = h^*\}$, $\mathcal{G}_i \leftarrow \mathcal{G}_i \cup \{c \in \mathcal{C} \mid h(c) = h^*\}$

continuer $\leftarrow$ vrai

Sinon

$j \leftarrow j + 1$

Retourner $\mathcal{D} = \{\mathcal{G}_1, \ldots, \mathcal{G}_D\}$

6.2.2.1. Détermination d'une planification voisine. Partant d'une planification $\mathcal{P}$, la génération d'une solution voisine $\mathcal{P}''$ comprend d'une part la modification de la décomposition et d'autre part la construction de planifications associées aux nouveaux sous-problèmes. Dans un premier temps, la décomposition $\mathcal{D}$ est modifiée et une planification intermédiaire $\mathcal{P}'$ est obtenue. Dans un second temps, les sous-problèmes modifiés correspondants sont optimisés, pour finalement aboutir à une nouvelle planification $\mathcal{P}''$.

Le processus de remise en cause de la décomposition en sous-problèmes, consiste à déplacer des enfants d'une région à une autre. Cette phase est utile à au moins deux points de vue :

- ▷ dans notre décomposition initiale, tous les enfants habitant en une même maison font forcément partie du même sous-problème. Le déplacement d'enfants individuels offre une plus grande flexibilité et permet d'assouplir cette limitation initiale ;
- ▷ si la décomposition courante n'est pas adéquate, le cloisonnement du problème en sous-problèmes indépendants peut nous empêcher de nous approcher d'un solution optimale.

Plus concrètement, nous allons choisir aléatoirement $\varsigma \in \mathbb{N}^*$ enfants distincts à déplacer en utilisant la loi de probabilité (5.1), *i.e.* plus un enfant présente une importante perte de temps, plus la probabilité est grande qu'il soit choisi. Soit c un enfant à déplacer dans un autre groupe $\mathcal{G}_i$, avec $i \in \{1, \dots, D\} \setminus \{g(c)\}$. Pour des questions géographiques, nous allons placer cet enfant dans le groupe $\mathcal{G}_i$ (avec $i \neq g(c)$) le plus proche de sa maison $h(c)$, où la *distance d'un enfant c à un groupe $\mathcal{G}_i$* est définie par

$$d(c, \mathcal{G}_i) = \min_{v \in \mathcal{V}_{\mathcal{G}_i}} d(h(c), v). \tag{6.2}$$

L'enfant c doit ensuite être affecté à l'un des bus associés à son nouveau groupe $\mathcal{G}_i$. Il semble judicieux de choisir le bus $b \in \mathcal{B}_i$ dont la tournée passe au plus près de l'école $s(c)$, où la *distance d'un enfant c à une tournée $\mathcal{T}_b = \{a_\alpha^b\}_{\alpha=1}^{n_b}$* est définie par

$$d(c, \mathcal{T}_b) = \min_{\alpha \in \{1, \dots, n_b\}} d(s(c), v_\alpha^b). \tag{6.3}$$

L'enfant c doit encore être placé dans la tournée du bus choisi en utilisant le voisinage admissible $\mathcal{N}_f$. La planification obtenue après ς déplacements d'enfants est notée $\mathcal{P}'$. Elle diffère de $\mathcal{P}$ au niveau des sous-problèmes touchés par les déplacements d'enfants (on note $\overline{\mathcal{D}} \subseteq \mathcal{D}$ l'ensemble des groupes modifiés). Ce processus de modification de la décomposition en sous-problèmes est décrit dans l'algorithme 6.4.

La construction de la planification $\mathcal{P}''$ revient à optimiser les sous-problèmes modifiés, *i.e.* ceux correspondant aux groupes de l'ensemble $\overline{\mathcal{D}}$. L'optimisation de chacun de ces sous-problèmes est effectuée à l'aide de la recherche tabou non admissible décrite dans la section 3.2.5.

6.2.2.2. Règle d'acceptation d'une planification voisine. Partant d'une planification $\mathcal{P}$, on obtient une solution voisine $\mathcal{P}''$ selon le processus décrit ci-dessus. Toutefois, on ne se déplace de $\mathcal{P}$ vers $\mathcal{P}''$ qu'avec probabilité

$$\min\left\{1, \exp\left(-\frac{\mathcal{F}(\mathcal{P}'') - \mathcal{F}(\mathcal{P})}{T}\right)\right\}, \tag{6.4}$$

où $T > 0$ est un paramètre de température initialisé à la valeur T_0. Tout comme pour le recuit simulé décrit en section 3.2.4, la température est diminuée toutes les μ itérations et est réinitialisée à T_0, lorsque depuis γ_{max} itérations plus aucune planification voisine $\mathcal{P}''$ n'a été acceptée.

Cette méthode est en fait un recuit simulé admissible avec utilisation de la recherche tabou non admissible pour l'optimisation des sous-problèmes modifiés lors de la construction d'une planification voisine. Notons que, lors de la première itération de l'heuristique, la décomposition initiale construite à l'aide de la méthode présentée dans la section 6.1.3 n'est pas remise en question (*i.e.* $\mathcal{P}' = \mathcal{P}$).

De nombreuses variantes de ce processus sont envisageables. Il serait par exemple possible d'utiliser le recuit simulé non admissible ou une recherche à voisinage variable pour l'optimisation des sous-problèmes, de visiter des solutions voisines non admissibles ou encore de modifier le critère d'acceptation d'une solution voisine.

Algorithme 6.4 ModifieDécomposition

Donnée : $\mathcal{P}$, $\mathcal{D}$, λ, m, ς

$\mathcal{P}_0 \leftarrow \mathcal{P}$, $\overline{\mathcal{D}} \leftarrow \varnothing$

Pour i **de** 1 **à** ς **faire**

Choisir aléatoirement un enfant c_i (avec $c_i \neq c_j$, pour tout $i > j$) selon la loi (5.1).

Déterminer le groupe $\mathcal{G}_j$ (avec $j \neq g(c)$) le plus «proche» de c_i, au sens de (6.2).

Déterminer le bus $b_i \in \mathcal{B}_j$ le plus «proche» de c_i, au sens de (6.3).

Soit $\mathcal{P}_i \in \mathcal{N}_f(\mathcal{P}_{i-1})$ la planification voisine obtenue en insérant c_i dans le bus b_i.

$\mathcal{G}_{g(c_i)} \leftarrow \mathcal{G}_{g(c_i)} \setminus \{c_i\}$, $\mathcal{G}_j \leftarrow \mathcal{G}_j \cup \{c_i\}$

$\overline{\mathcal{D}} \leftarrow \overline{\mathcal{D}} \cup \{\mathcal{G}_{g(c_i)}, \mathcal{G}_j\}$ (∗ ensemble des sous-problèmes modifiés ∗)

Retourner $(\mathcal{P}_\varsigma, \overline{\mathcal{D}})$

La méthode de décomposition en sous-problèmes implémentée, notée $\mathrm{Dec}_{\mathrm{Tab}\ \mathcal{N}_i}$, est détaillée dans l'algorithme 6.5. Les paramètres d'optimisation sont :

- ▷ la température initiale T_0 ;
- ▷ le coefficient ω de diminution de la température ;
- ▷ le nombre μ d'itérations consécutives sans changement de température ;
- ▷ le nombre ς d'enfants à changer de groupe lors de la modification de la décomposition ;
- ▷ le paramètre λ de la loi de probabilité (5.1) pour le choix de l'enfant à déplacer ;
- ▷ la constante m ajoutée à la perte de temps de chaque enfant dans la loi de probabilité (5.1) pour le choix de l'enfant à déplacer ;
- ▷ le nombre maximum γ_{max} d'itérations consécutives pendant lesquelles on admet que la planification courante ne soit pas changée (*i.e.* contrôle du «gel» du recuit simulé) ;
- ▷ le nombre maximum η_{max} d'évaluations de planifications consécutives sans amélioration de la meilleure planification rencontrée ;
- ▷ la taille ν des sous-voisinages considérés ;
- ▷ la longueur ζ de la liste tabou ;
- ▷ le coefficient κ_0 de pénalité associé à la fonction objectif $\mathcal{F}$;
- ▷ la probabilité φ_0 de choisir un voisin dans le voisinage non admissible $\mathcal{N}_i$;
- ▷ le coefficient ψ de diminution de la probabilité de choisir le voisinage non admissible ;
- ▷ le nombre maximum η_{max}^{loc} d'évaluations de planifications consécutives sans amélioration de la meilleure planification rencontrée dans la recherche tabou non admissible.

Algorithme 6.5 Heuristique pour les problèmes de grande taille

Donnée : $\{\mathcal{T}_b\}_{b=1}^{B}$, $\mathcal{D}$, T_0, ω, μ, ν, ζ, κ_0, λ, m, ς, φ_0, ψ, γ_{max}, η_{max}, η_{max}^{loc}, $\mathcal{F} \in \{\mathcal{F}_1, \mathcal{F}_2\}$

$T \leftarrow T_0$, $\eta \leftarrow 0$, $\gamma \leftarrow 0$, $\kappa \leftarrow \kappa_0$, $\varphi \leftarrow \varphi_0$, $\mathcal{P} \leftarrow \{\mathcal{T}_b\}_{b=1}^{B}$, $\mathcal{P}^* \leftarrow \mathcal{P}$, $\mathcal{P}' \leftarrow \mathcal{P}$, $\overline{\mathcal{D}} \leftarrow \mathcal{D}$, $n \leftarrow 0$

Tant que $\eta < \eta_{max}$ **faire**

$n \leftarrow n + 1$ (∗ nombre d'itérations ∗)

Si $n \neq 1$ **alors** (∗ 1ère itération, ne pas modifier les régions initiales ∗)

$(\mathcal{P}', \overline{\mathcal{D}}) \leftarrow$ **ModifieDécomposition**$(\mathcal{P}, \mathcal{D}, \lambda, m, \varsigma)$

$\{\mathcal{T}_b'\}_{b \in \mathcal{B}} \leftarrow \mathcal{P}'$

Pour tout $\mathcal{G}_i \in \overline{\mathcal{D}}$ **faire** (∗ réoptimiser les sous-problèmes de $\overline{\mathcal{D}}$ ∗)

$\{\mathcal{T}_b''\}_{b \in \mathcal{B}_i} \leftarrow$ **RechercheTabou** $(\{\mathcal{T}_b'\}_{b \in \mathcal{B}_i}, \nu, \zeta, \kappa_0, \lambda, \varphi_0, \psi, \eta_{max}^{loc}, \mathcal{F})$

$\mathcal{P}'' \leftarrow \bigcup_{\mathcal{G}_i \in \overline{\mathcal{D}}} \{\mathcal{T}_b''\}_{b \in \mathcal{B}_i} \bigcup_{\mathcal{G}_i \notin \overline{\mathcal{D}}} \{\mathcal{T}_b'\}_{b \in \mathcal{B}_i}$

Si $\mathcal{F}(\mathcal{P}'') < \mathcal{F}(\mathcal{P})$ **alors** (∗ trouvé une meilleure planification ∗)

$\mathcal{P} \leftarrow \mathcal{P}''$, $\gamma \leftarrow 0$

Sinon

Si $\exp\left(-\frac{\mathcal{F}(\mathcal{P}'') - \mathcal{F}(\mathcal{P})}{T}\right) > (y \sim U[0,1))$ **alors**

$\mathcal{P} \leftarrow \mathcal{P}''$, $\gamma \leftarrow 0$ (∗ moins bonne planification acceptée ∗)

Sinon

$\gamma \leftarrow \gamma + 1$

Si $\mathcal{F}(\mathcal{P}'') < \mathcal{F}(\mathcal{P}^*)$ **alors**

$\mathcal{P}^* \leftarrow \mathcal{P}''$, $\eta \leftarrow 0$ (∗ meilleure planification ∗)

Sinon

$\eta \leftarrow \eta + 1$

Si $(n \mod \mu = 0)$ **alors**

$T \leftarrow \omega \cdot T$ (∗ diminution de la température ∗)

Si $(\gamma > \gamma_{max})$ **alors**

$T \leftarrow T_0$, $\gamma \leftarrow 0$ (∗ gel ∗)

Retourner Lin-2-opt$(\mathcal{P}^*)$ (∗ optimisation ∗)

6.3. Présentation des instances testées

Ne disposant pas de problème réel de grande taille, diverses instances fictives ont été générées afin de tester l'approche par décomposition présentée ci-dessus.

6.3.1. Données fictives

La méthode décrite en section 4.1.2 a été utilisée pour générer des jeux de données fictives. Les valeurs des paramètres ont cependant été adaptées de manière à obtenir des jeux de données ressemblant à ce que pourrait être un problème réel dans une métropole. La densité des écoles a donc été réduite par rapport à celle des instances de petite taille du chapitre 4 et la valeur du paramètre ξ a été augmentée de manière à ce qu'il soit possible de traverser notre métropole en environ deux heures. Les valeurs des paramètres retenues sont donc les suivantes :

- ▷ $\xi \in \{4, 5, 8\}$;
- ▷ le degré maximum des sommets $\delta = 5$;
- ▷ l'ensemble des types d'écoles $\mathcal{M} = \{$enfantine, primaire, secondaire$\}$;
- ▷ les proportions d'écoles de chaque type $(p_1, p_2, p_3) = (1/20, 1/25, 1/40)$;
- ▷ $\Gamma_1 = (0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 2, 2, 3, 3, 3, 4, 6, 6, 6, 9, 13, 15)$;
- ▷ $\Gamma_2 = (0, 1, 1, 2, 2, 2, 2, 2, 3, 4, 4, 5, 6, 6, 7, 7, 8, 10, 10, 10, 14, 17, 19, 20)$;
- ▷ $\Gamma_3 = (0, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 3, 3, 8)$.

Par ailleurs, on suppose à nouveau qu'un enfant d'école enfantine ou primaire occupe 7 unités dans un bus, alors qu'un enfant de secondaire en occupe 10. Les bus utilisés ont tous une capacité de 210 unités et peuvent donc transporter 30 enfants d'enfantine ou de primaire et 21 enfants de secondaire.

Ayant fixé les paramètres ci-dessus, il ne reste qu'à choisir le nombre de sommets V et le nombre de bus B. Quatre jeux de données fictives correspondant à des réseaux constitués de 263, 388, 516 et 1036 sommets, nommés F_{263}, F_{388}, F_{516} et F_{1036}, ont été générés. Les caractéristiques de ces jeux de données sont reportées dans la table 6.1. En guise d'illustration, le réseau du jeu de données F_{388} est également représenté sur la figure 6.2. Ces grandeurs sont à mettre en perspective avec les valeurs des tables 4.1 et 4.2 (ainsi que les figures 4.1, 4.2 et 4.3), qui montrent les caractéristiques correspondantes pour les instances réelles et fictives étudiées au chapitre 4.

				Enfants en classe			
	$\lvert\mathcal{H}\rvert$	$\lvert\mathcal{S}\rvert$	$\lvert\mathcal{C}\rvert$	enfantine	primaire	secondaire	$N_{\mathcal{C}}$
F_{263}	248	38	661	220	248	193	2280
F_{388}	370	34	984	252	361	335	3475
F_{516}	497	53	1333	386	482	465	4550
F_{1036}	984	96	2499	604	968	927	9100

TAB. 6.1 – Caractéristiques des jeux de données fictives.

6.3.2. Instances considérées

Pour chacun des deux problèmes de planification de tournées de véhicules scolaires considérés (*i.e.* l'un avec la fonction objectif $\mathcal{F}_1$ et l'autre avec la fonction objectif $\mathcal{F}_2$), quelques instances construites à partir des jeux de données décrits ci-dessus ont été optimisées après avoir été

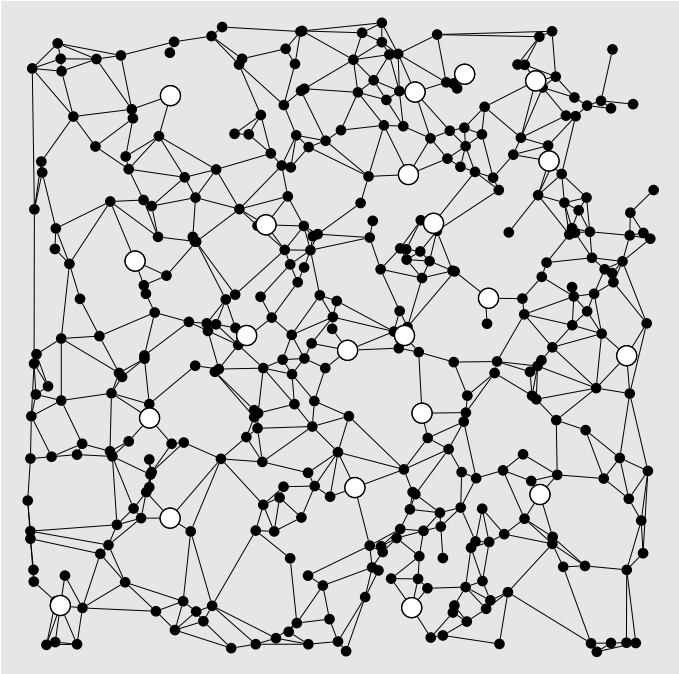


FIG. 6.2 – Réseau fictif de 388 sommets où les écoles sont représentées en blanc.

décomposées en un certain nombre de sous-problèmes. Dans les résultats présentés ci-dessous, on note $F_V(B, D)$ l'instance obtenue à partir du jeu de données F_V pour un nombre de bus B et une décomposition en D sous-problèmes. Les instances considérées sont énumérées dans la table 6.2. Notons que le nombre de bus disponibles a été choisi de manière à obtenir un rapport d'environ 45 enfants par bus.

	F_{263}						F_{388}		F_{516}		F_{1036}
B	50			55			78		100		205
D	3	4	5	6	10	15	6	7	8	10	20

TAB. 6.2 – Instances sur lesquelles les heuristiques ont été testées.

6.4. Décomposition initiale

La première étape de la résolution par décomposition d'un problème de grande taille consiste à définir les D sous-problèmes en utilisant les algorithmes 6.1, 6.2 et 6.3 correspondant aux étapes I, II et III décrites dans la section 6.1.3. Afin de suivre la construction des sous-problèmes, nous présentons, pour les instances $F_{105}(25, 3)$ et $F_{263}(50, 4)$, diverses caractéristiques des sous-problèmes, après avoir appliqué les étapes I (création de zones géographiques connexes) et II (répartition des bus), de même qu'à la fin de l'étape III (équilibrage des sous-problèmes). Ces valeurs sont indiquées dans les tables 6.3 et 6.4. Les figures 6.3 et 6.4 illustrent l'évolution de la construction des régions correspondantes.

On constate que les zones géographiques connexes obtenues à la fin de l'étape II ne contiennent généralement pas un nombre similaire de sommets. Ce phénomène est clair sur la figure 6.3(a), où la zone noire s'avère très grande (58.1% des sommets), ainsi que sur la figure 6.4(a), où elle est très petite (12.9% des sommets).

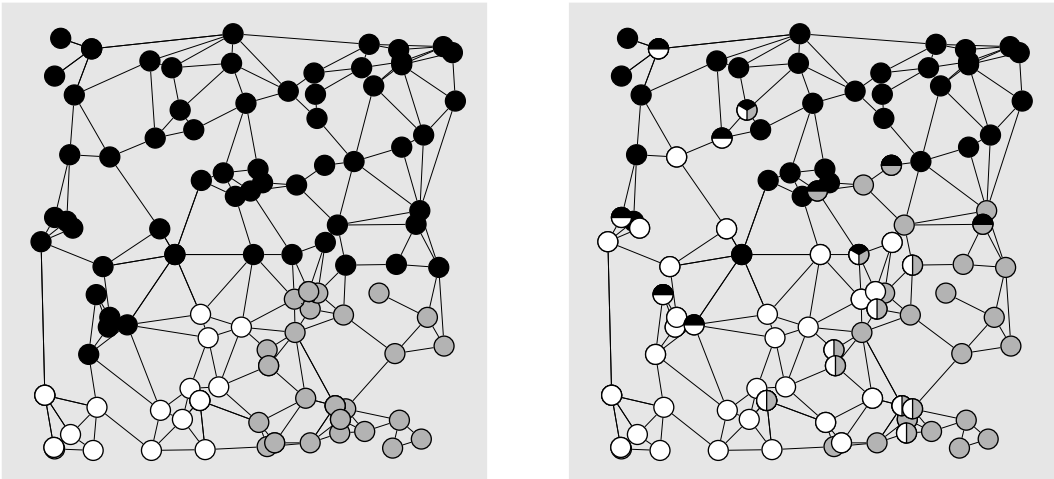
i^{e} sous-problème	1	2	3
$\lvert\mathcal{B}_i\rvert$	9	8	8
$\lvert\mathcal{G}_i\rvert$	31	108	51
$E(\mathcal{G}_i)$	-132.8	171.9	-39.1
$\lvert\mathcal{V}_{\mathcal{G}_i}\rvert$	18	61	26
Diamètre de $\mathcal{V}_{\mathcal{G}_i}$	40	74	39

Étapes I et II

i^{e} sous-problème	1	2	3
$\lvert\mathcal{B}_i\rvert$	9	8	8
$\lvert\mathcal{G}_i\rvert$	70	69	51
$E(\mathcal{G}_i)$	-0.8	0.9	-0.1
$\lvert\mathcal{V}_{\mathcal{G}_i}\rvert$	49	43	33
Diamètre de $\mathcal{V}_{\mathcal{G}_i}$	56	70	51

Étape III

TAB. 6.3 – Caractéristiques de la décomposition à la fin de l'application des étapes II et III pour l'instance $F_{105}(25, 3)$, où le diamètre des régions est indiqué en minutes.



(a) Étapes I et II (b) Étape III

FIG. 6.3 – Sous-problèmes obtenus à la fin de l'application des étapes II et III pour l'instance $F_{105}(25, 3)$, où la première région (9 bus) est représentée en blanc, la seconde (8 bus) en noir et la troisième (8 bus) en gris. Notons qu'après l'étape III, les sommets appartenant à plusieurs régions correspondent à des écoles.

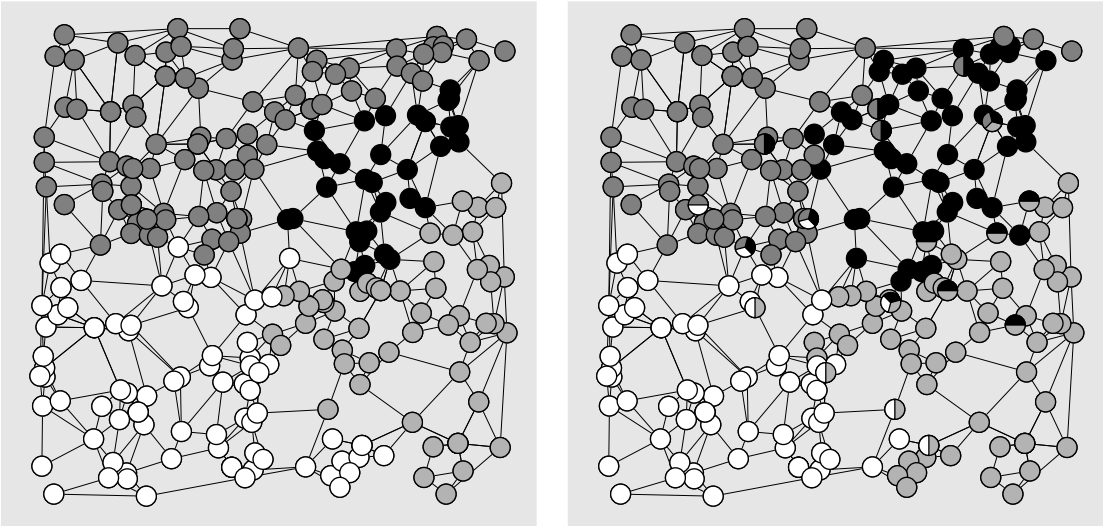
i^{e} sous-problème	1	2	3	4
$\lvert\mathcal{B}_i\rvert$	13	13	12	12
$\lvert\mathcal{G}_i\rvert$	233	144	90	194
$E(\mathcal{G}_i)$	193.2	-65.8	-228.2	100.8
$\lvert\mathcal{V}_{\mathcal{G}_i}\rvert$	92	56	34	81
Diamètre de $\mathcal{V}_{\mathcal{G}_i}$	40	32	22	44

Étapes I et II

i^{e} sous-problème	1	2	3	4
$\lvert\mathcal{B}_i\rvert$	13	13	12	12
$\lvert\mathcal{G}_i\rvert$	173	160	157	171
$E(\mathcal{G}_i)$	0.2	0.2	-2.2	-1.8
$\lvert\mathcal{V}_{\mathcal{G}_i}\rvert$	76	72	69	75
Diamètre de $\mathcal{V}_{\mathcal{G}_i}$	40	35	27	41

Étape III

TAB. 6.4 – Caractéristiques de la décomposition à la fin de l'application des étapes II et III pour l'instance $F_{263}(50, 4)$, où le diamètre des régions est indiqué en minutes.



(a) Étapes I et II (b) Étape III

FIG. 6.4 – Sous-problèmes obtenus à la fin de l'application des étapes II et III pour l'instance $F_{263}(50, 4)$, où la première région (13 bus) est représentée en gris foncé, la deuxième (13 bus) en gris clair, la troisième (12 bus) en noir et la quatrième (12 bus) en blanc. Notons, qu'après l'étape III, les sommets appartenant à plusieurs régions correspondent à des écoles.

À ce stade de la décomposition, hormis le fait que les zones sont bien connexes, les autres indicateurs d'une «bonne décomposition» (voir section 6.1.1), c'est-à-dire l'équilibrage du nombre d'enfants entre sous-problèmes et des excès de charge proches de zéro, ne sont guère convaincants (voir tables 6.3 et 6.4). L'application de l'étape III permet d'améliorer considérablement cette situation initiale : les sous-problèmes finalement obtenus contiennent environ le même nombre d'enfants à transporter et les valeurs absolues des excès de charge sont extrêmement petites.

Les tables 6.3 et 6.4 indiquent également quel est le diamètre des régions obtenues. Même si cette notion n'intervient pas dans la génération des sous-problèmes, on ne peut que se réjouir de constater que l'application de l'étape III tend aussi à équilibrer cette valeur.

6.5. Paramètres utilisés pour l'heuristique Dec$_{\text{Tab}\,\mathcal{N}_i}$

Après cette phase de décomposition, l'heuristique Dec$_{\text{Tab}\,\mathcal{N}_i}$ est appliquée aux planifications obtenues. Cette méthode a été testée avec trois températures initiales, deux valeurs pour le nombre d'enfants à changer de groupe, deux longueurs de liste tabou, ainsi qu'un germe du générateur de nombres pseudo-aléatoires. Chaque heuristique a donc été exécutée avec douze jeux de paramètres, sur un Xeon cadencé à 2.4GHz. Après quelques tests, les valeurs suivantes des paramètres d'optimisation ont été retenues :

- ▷ température initiale : $T_0 \in \begin{cases} \{250,\ 500,\ 750\} & \text{si } \mathcal{F} = \mathcal{F}_1 \\ \{5, 6, 7\} & \text{si } \mathcal{F} = \mathcal{F}_2\,; \end{cases}$
- ▷ coefficient de diminution de la température : $\omega = 0.93$;
- ▷ nombre d'itérations consécutives avec la même température : $\mu = 5$;
- ▷ nombre d'enfants à changer de groupe lors de la modification de la décomposition : $\varsigma \in \{3, 5\}$;

- ▷ paramètre de la loi de probabilité (5.1) pour le choix de l'enfant à déplacer : $\lambda = 1$;
- ▷ constante ajoutée à la perte de temps de chaque enfant dans la loi de probabilité (5.1) pour le choix de l'enfant à déplacer : $m = 0$;
- ▷ nombre maximum d'itérations consécutives pendant lesquelles il est autorisé que la planification courante ne soit pas changée : $\gamma_{max} = 15$;
- ▷ taille des sous-voisinages considérés : $\nu = 18$;
- ▷ longueur de la liste tabou : $\zeta \in \{7, 10\}$;
- ▷ coefficient de pénalité associé à la fonction objectif $\mathcal{F}$: $\kappa_0 = 1.02$;
- ▷ probabilité de choisir un voisin dans $\mathcal{N}_i$: $\varphi_0 = 0.7$;
- ▷ coefficient de diminution de la probabilité φ_0 de choisir le voisinage non admissible : $\psi = 0.95$;
- ▷ nombre maximum d'évaluations de planifications consécutives sans amélioration de la meilleure solution rencontrée avec la recherche tabou non admissible : $\eta^{loc}_{max} = 7\,500$;
- ▷ nombre maximum d'évaluations de planifications consécutives sans amélioration de la meilleure solution rencontrée : $\eta_{max} = 50$.

6.6. Analyse des planifications obtenues

6.6.1. Première optimisation des sous-problèmes

Afin d'évaluer la pertinence de l'approche développée pour la décomposition initiale d'un problème, nous allons observer la qualité des planifications obtenues à la fin de la première itération ($n = 1$), *i.e.* après avoir optimisé une seule fois chaque sous-problème sans modifier la décomposition initiale. Les tables 6.5 et 6.6 indiquent, pour quelques instances générées à partir du jeu de données F_{263}, les grandeurs suivantes.

- ▷ La valeur de la planification initiale obtenue en réunissant les solutions initiales générées pour les divers sous-problèmes selon la méthode décrite dans la section 6.1.3.
- ▷ La valeur de la planification obtenue après l'application d'une itération de l'heuristique $\mathrm{Dec}_{\mathrm{Tab}\ \mathcal{N}_i}$, c'est-à-dire après avoir optimisé une seule fois chaque sous-problème. Pour chaque instance, sur les douze exécutions effectuées, seule la planification présentant la meilleure fonction objectif a été retenue.

	$F_{263}(50,3)$	$F_{263}(50,4)$	$F_{263}(50,5)$	$F_{263}(50,6)$	$F_{263}(50,10)$	$F_{263}(50,15)$
Planifications initiales	19.4	22.8	18.8	19.9	22.7	24.4
Planifications optimisées $n = 1$	8.2	7.8	7.5	7.8	8.8	10.5

TAB. 6.5 – Pertes de temps moyennes (en minutes) obtenues pour les planifications initiales et après une itération de $\mathrm{Dec}_{\mathrm{Tab}\ \mathcal{N}_i}$ pour le problème $F_{263}(50)$.

Ces résultats sont encourageants. En effet, la décomposition initiale ne semble pas mener à des sous-problèmes dégénérés (difficiles à optimiser). Elle laisse suffisamment de marge de manœuvre pour l'optimisation des planifications initiales. L'amélioration est particulièrement nette dans le cas de la fonction objectif $\mathcal{F}_1$, mais plus modeste dans le cas de la fonction objectif

	$F_{263}(50,3)$	$F_{263}(50,4)$	$F_{263}(50,5)$	$F_{263}(50,6)$	$F_{263}(50,10)$	$F_{263}(50,15)$
Planifications initiales	88	130	101	97	126	127
Planifications optimisées $n=1$	71	104	83	84	92	116

TAB. 6.6 – Pertes de temps maximales (en minutes) obtenues pour les planifications initiales et après une itération de $\text{Dec}_{\text{Tab}\ \mathcal{N}_i}$ pour le problème $F_{263}(50)$.

$\mathcal{F}_2$. Notons que ce phénomène a déjà été constaté dans les tables 4.8 et 4.9 avec les méthodes développées précédemment.

6.6.2. Efficacité de l'approche par décomposition

L'approche par décomposition a été développée dans le but d'accélérer la construction de planifications de bonne qualité pour des problèmes de grande taille. Afin de vérifier si cet objectif est atteint, nous allons comparer les planifications obtenues par l'application de $\text{Dec}_{\text{Tab}\ \mathcal{N}_i}$ avec celles construites à l'aide de Tab $\mathcal{N}_i$. Pour que la comparaison soit équitable, nous avons alloué le même budget de temps aux deux méthodes. Pour la même raison, l'heuristique Tab $\mathcal{N}_i$ n'a été évaluée qu'en ne considérant que les douze premiers jeux de paramètres parmi les vingt décrits en section 4.2. Les tables 6.7 à 6.13 indiquent les valeurs des meilleures planifications obtenues (sur les douze exécutions considérées) pour un même temps limite donné. Les instances de la forme $F_V(B,1)$ ont été optimisées avec l'heuristique Tab $\mathcal{N}_i$ et toutes les autres avec l'heuristique $\text{Dec}_{\text{Tab}\ \mathcal{N}_i}$. Les différents tests ont été effectués sur un Xeon cadencé à 2.4Ghz.

	Tab $\mathcal{N}_i$	$\text{Dec}_{\text{Tab}\ \mathcal{N}_i}$					
	$F_{263}(50,1)$	$F_{263}(50,3)$	$F_{263}(50,4)$	$F_{263}(50,5)$	$F_{263}(50,6)$	$F_{263}(50,10)$	$F_{263}(50,15)$
Planifications initiales	20.7	19.4	22.8	18.8	19.9	22.7	24.4
Planifications après 1.5 jours	10.7	7.7	7.4	7.2	7.6	8.0	10.1
Temps moyen par itération	—	86.1	101.0	132.9	103.4	80.4	61.4

TAB. 6.7 – Pertes de temps moyennes (en minutes) des planifications initiales et optimisées après un jour et demi de temps de calcul pour le problème $F_{263}(50)$. Le temps moyen nécessaire (en minutes) pour effectuer une itération de l'heuristique $\text{Dec}_{\text{Tab}\ \mathcal{N}_i}$ est également indiqué.

Une estimation du temps moyen nécessaire pour effectuer une itération de l'heuristique $\text{Dec}_{\text{Tab}\ \mathcal{N}_i}$ pour le problème $F_{263}(50)$ est également donnée dans les tables 6.7 et 6.8 (notons que la première itération a été exclue du calcul, car elle nécessite l'optimisation de tous les sous-problèmes, ce qui n'est généralement plus le cas par la suite). On en déduit qu'une dizaine d'itérations ont pu être effectuées dans les limites du budget de temps alloué.

Les résultats obtenus (voir tables 6.7 à 6.13) montrent qu'une approche par décomposition permet de réduire les temps de calcul nécessaires à l'obtention d'une planification de bonne

qualité. Pour le problème consistant à minimiser la moyenne des pertes de temps, quel que soit le choix de D, les planifications obtenues sont entre 6% et 61% meilleures que celles construites sans décomposition. Pour la fonction objectif $\mathcal{F}_2$, le choix de D est plus délicat et peut mener à une amélioration comme à une détérioration des performances.

	Tab $\mathcal{N}_i$	$\text{Dec}_{\text{Tab } \mathcal{N}_i}$					
	$F_{263}(50,1)$	$F_{263}(50,3)$	$F_{263}(50,4)$	$F_{263}(50,5)$	$F_{263}(50,6)$	$F_{263}(50,10)$	$F_{263}(50,15)$
Planifications initiales	104	88	130	101	97	126	127
Planifications après 1.5 jours	77	66	101	83	76	85	99
Temps moyen par itération	—	138.4	99.0	101.1	89.6	104.0	81.2

TAB. 6.8 – Pertes de temps maximales (en minutes) des planifications initiales et optimisées après un jour et demi de temps de calcul pour le problème $F_{263}(50)$. Le temps moyen nécessaire (en minutes) pour effectuer une itération de l'heuristique $\text{Dec}_{\text{Tab } \mathcal{N}_i}$ est également indiqué.

	Tab $\mathcal{N}_i$	$\text{Dec}_{\text{Tab } \mathcal{N}_i}$					
	$F_{263}(55,1)$	$F_{263}(55,3)$	$F_{263}(55,4)$	$F_{263}(55,5)$	$F_{263}(55,6)$	$F_{263}(55,10)$	$F_{263}(55,15)$
Planifications initiales	18.1	16.4	16.3	16.5	15.3	17.6	16.9
Planifications après 1 jour	9.6	6.8	5.8	6.1	6.4	6.8	7.7

TAB. 6.9 – Pertes de temps moyennes (en minutes) des planifications initiales et optimisées après un jour de temps de calcul pour le problème $F_{263}(55)$.

	Tab $\mathcal{N}_i$	$\text{Dec}_{\text{Tab } \mathcal{N}_i}$	
	$F_{388}(78,1)$	$F_{388}(78,6)$	$F_{388}(78,7)$
Planifications initiales	24.5	25.8	27.5
Planifications après 1 jour	20.0	11.3	11.6
Planifications après 2 jours	18.8	10.9	10.2

TAB. 6.10 – Pertes de temps moyennes (en minutes) des planifications initiales et optimisées après un et deux jours de temps de calcul pour le problème $F_{388}(78)$.

	Tab $\mathcal{N}_i$	$\text{Dec}_{\text{Tab } \mathcal{N}_i}$	
	$F_{388}(78,1)$	$F_{388}(78,6)$	$F_{388}(78,7)$
Planifications initiales	165	136	172
Planifications après 1 jour	133	110	146
Planifications après 2 jours	131	110	144

TAB. 6.11 – Pertes de temps maximales (en minutes) des planifications initiales et optimisées après un et deux jours de temps de calcul pour le problème $F_{388}(78)$.

	Tab $\mathcal{N}_i$	$\text{Dec}_{\text{Tab } \mathcal{N}_i}$	
	$F_{516}(100,1)$	$F_{516}(100,8)$	$F_{516}(100,10)$
Planifications initiales	25.1	24.8	20.8
Planifications après 1.5 jours	21.6	9.1	8.8
Planifications après 3.5 jours	21.0	8.9	8.1

TAB. 6.12 – Pertes de temps moyennes (en minutes) des planifications initiales et optimisées après un jour et demi et trois jours et demi de temps de calcul pour le problème $F_{516}(100)$.

	Tab $\mathcal{N}_i$	$\text{Dec}_{\text{Tab } \mathcal{N}_i}$
	$F_{1036}(205,1)$	$F_{1036}(205,20)$
Planifications initiales	18.5	16.5
Planifications après 3.5 jours	16.3	10.1

TAB. 6.13 – Pertes de temps moyennes (en minutes) des planifications initiales et optimisées après trois jours et demi de temps de calcul pour le problème $F_{1036}(205)$.

6.6.3. Évolution de la qualité des planifications en fonction du nombre de régions

Le nombre de sous-problèmes influence la qualité des planifications produites. Comme nous l'avons déjà mentionné en section 6.1.1, ce nombre doit être compris entre 2 et $\lfloor B/2 \rfloor$ afin de ne pas retomber dans le cas d'un problème sans décomposition. Les tables 6.7 à 6.9 montrent l'évolution typique de la qualité des planifications en fonction du nombre de sous-problèmes. On constate que pour le problème correspondant à la minimisation de la somme des pertes de temps, la fonction associant la valeur de la planification obtenue au nombre de sous-problèmes

est convexe. Partant d'un autre point de vue, la table 6.14 indique quel a été le temps nécessaire aux différentes heuristiques pour atteindre le même niveau de qualité que celui de Tab $\mathcal{N}_i$ pour le problème $F_{263}(50)$ (voir table 6.7). On note que la fonction associant ce temps de calcul au nombre de sous-problèmes considérés est à nouveau convexe. On en conclut qu'un trop petit nombre de sous-problèmes a pour effet de ralentir le processus d'optimisation, alors qu'un nombre trop important semble mener au cloisonnement du problème et finit par réduire la flexibilité nécessaire à l'obtention d'une bonne planification.

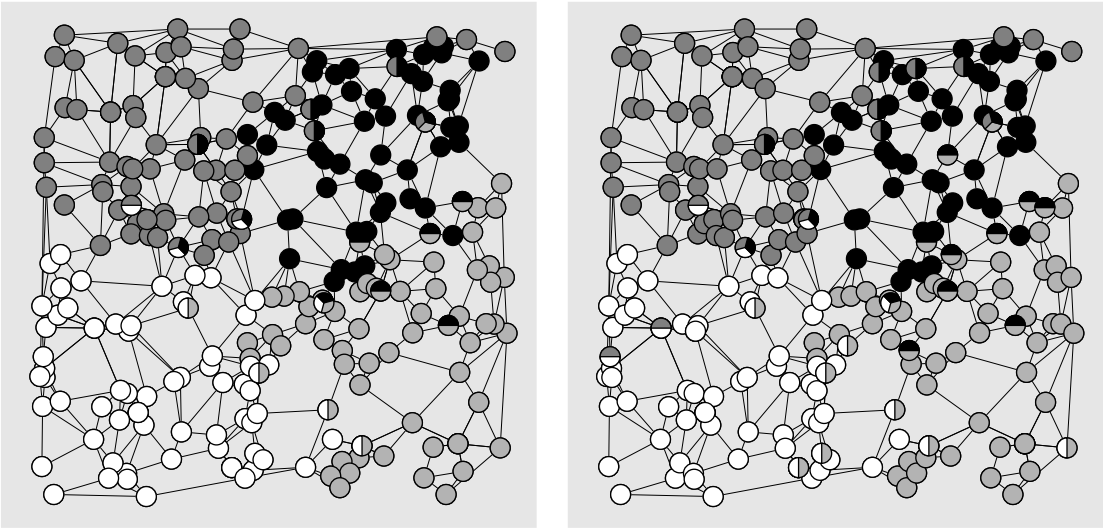
	Tab $\mathcal{N}_i$	$\text{Dec}_{\text{Tab } \mathcal{N}_i}$					
	$F_{263}(50,1)$	$F_{263}(50,3)$	$F_{263}(50,4)$	$F_{263}(50,5)$	$F_{263}(50,6)$	$F_{263}(50,10)$	$F_{263}(50,15)$
Temps	27	7	6.3	8.1	8.6	11.2	13.2

TAB. 6.14 – Temps nécessaire (en heures) pour l'obtention d'une planification de perte de temps moyenne égale à 10.7 minutes pour le problème $F_{263}(50)$.

La table 6.14 montre surtout que, dans ce cas particulier, l'utilisation d'une approche par décomposition permet de réduire d'un facteur 3 le temps de calcul nécessaire à l'obtention d'une planification de même qualité.

6.6.4. Évolution de la décomposition au cours de l'optimisation

Pour l'instance $F_{263}(50,4)$, la figure 6.5 permet de comparer les régions correspondant à la décomposition initiale (figure de gauche) avec celles obtenues après dix itérations de l'heuristique $\text{Dec}_{\text{Tab } \mathcal{N}_i}$ (figure de droite).



(a) $n = 0$ (b) $n = 10$

FIG. 6.5 – Régions obtenues au début de l'optimisation et après dix itérations (1 jour et demi) de l'heuristique $\text{Dec}_{\text{Tab } \mathcal{N}_i}$, pour l'instance $F_{263}(50,4)$.

La décomposition de la figure 6.5(b) correspond à la meilleure planification rencontrée (parmi les douze exécutions considérées) après dix itérations de l'heuristique $\mathrm{Dec}_{\mathrm{Tab}\ \mathcal{N}_i}$. Dans le jeu de paramètres permettant d'obtenir cette planification, la valeur de ς est égale à trois, ce qui signifie qu'à chaque remise en question de la décomposition trois enfants sont déplacés. Au cours de cette optimisation, sachant que cinq planifications voisines (parmi les dix considérées) ont été acceptées, on en conclut que quinze enfants ont changé de région entre les deux figures ci-dessus.

On note également l'apparition de nouveaux sommets appartenant à plusieurs régions à la fois. Ces derniers correspondent à des enfants habitant en une même maison, mais qui ont été séparés en cours d'optimisation. De tels déplacements d'enfants permettent de faire évoluer efficacement la décomposition en fonction de la structure du problème.

CHAPITRE 7

Interface utilisateur

Dans ce chapitre, les principales caractéristiques de l'interface utilisateur que nous avons développée sont présentées. Cet outil permet de visualiser et de modifier une planification. Un exemple d'utilisation est également proposé.

7.1. Introduction

Les planifications obtenues à l'aide de nos heuristiques peuvent ne pas être entièrement satisfaisantes d'un point de vue pratique. En effet, il est fréquemment nécessaire de tenir compte de contraintes supplémentaires non modélisées, comme par exemple le fait de vouloir privilégier un enfant handicapé. Il est donc important de disposer d'un outil permettant de visualiser et, le cas échéant, de modifier une solution quelconque. Par ailleurs, un tel outil devrait également permettre de mettre à jour une solution lorsque les données du problème ont été légèrement modifiées, par exemple lorsqu'un enfant déménage. Dans de telles situations, une modification manuelle est d'une part souvent plus rapide que la résolution complète du problème et, d'autre part, la planification obtenue a l'avantage d'être proche de l'originale (il est plus facile pour les chauffeurs d'utiliser une planification légèrement modifiée, que d'exploiter un nouvel itinéraire).

7.2. Fonctionnalités et caractéristiques de l'interface utilisateur

L'interface utilisateur OPLABUS (Outil de PLAnification de tournées de BUS Scolaires [Emo00]) est implémentée en Python (voir [LA99]) et communique avec nos heuristiques, développées en *C++*, à l'aide du module de liaison Swig (voir [Bea98]). Elle présente deux fonctionnalités majeures : la visualisation et l'édition d'une planification.

7.2.1. Visualisation d'une planification

Une première fenêtre, nommée «Caractéristiques de la planification» (voir figure 7.1), permet d'évaluer la qualité d'une planification. Les deux premières informations sont :

- ▷ l'admissibilité de la planification ;
- ▷ le nombre de bus à disposition.

L'utilisateur peut ensuite observer les caractéristiques suivantes, soit pour un bus particulier, soit pour la planification complète :

- ▷ la moyenne des pertes de temps $\mathcal{F}_1/N_{\mathcal{C}}$;
- ▷ le maximum des pertes de temps $\mathcal{F}_2$;
- ▷ le nombre de sièges disponibles ;
- ▷ le nombre minimum de sièges occupés (hors trajets à vide) ;
- ▷ le nombre moyen d'enfants par bus, c'est-à-dire le nombre moyen d'enfants présents sur le temps total de trajet, en comptant également les tronçons à vide (*moyenne A*) ;

- ▷ le nombre moyen d'enfants par bus, c'est-à-dire le nombre moyen d'enfants présents sur le temps total de trajet, sans les tronçons à vide (*moyenne B*) ;
- ▷ le nombre maximum de sièges occupés ;
- ▷ les temps de trajet et d'attente, ainsi que l'éventuel retard[1] (par rapport à l'heure de début des cours) des enfants transportés (valeurs minimales, moyennes et maximales).

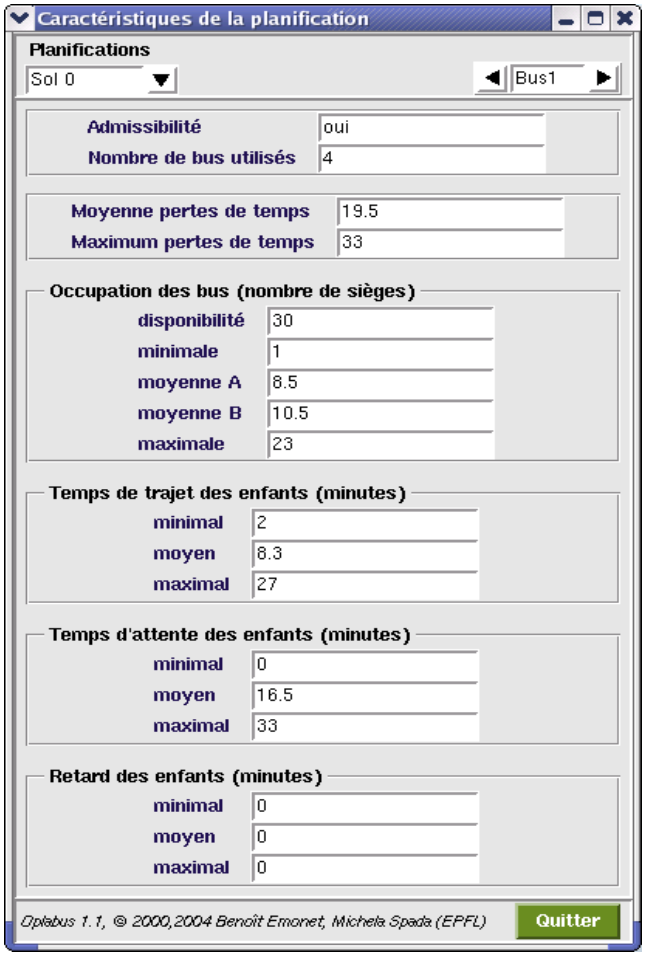


FIG. 7.1 – Caractéristiques d'une planification.

Une seconde fenêtre, nommée «Visualisation» (voir figure 7.2) permet de visualiser le trajet d'un bus et de tout enfant.

- ▷ La partie supérieure représente le réseau $G = (\mathcal{V}, \mathcal{E}, \ell)$ du problème considéré. Une animation permet de découvrir le parcours d'un bus sur un intervalle de temps choisi par l'utilisateur (voir figure 7.3). Il est également possible d'observer le trajet d'un enfant quelconque (voir figure 7.4).
- ▷ La partie inférieure comprend un diagramme représentant le nombre de sièges occupés en fonction du temps. En cliquant sur l'un des bâtonnets de l'histogramme, on obtient la liste des enfants s'y trouvant à l'instant en question, avec mention des temps de trajet et d'attente des enfants correspondants (voir figure 7.5).

[1]Bien que nos heuristiques ne produisent pas de planifications présentant des retards, l'interface utilisateur permet néanmoins de visualiser des solutions créées manuellement comprenant des arrivées tardives aux écoles.

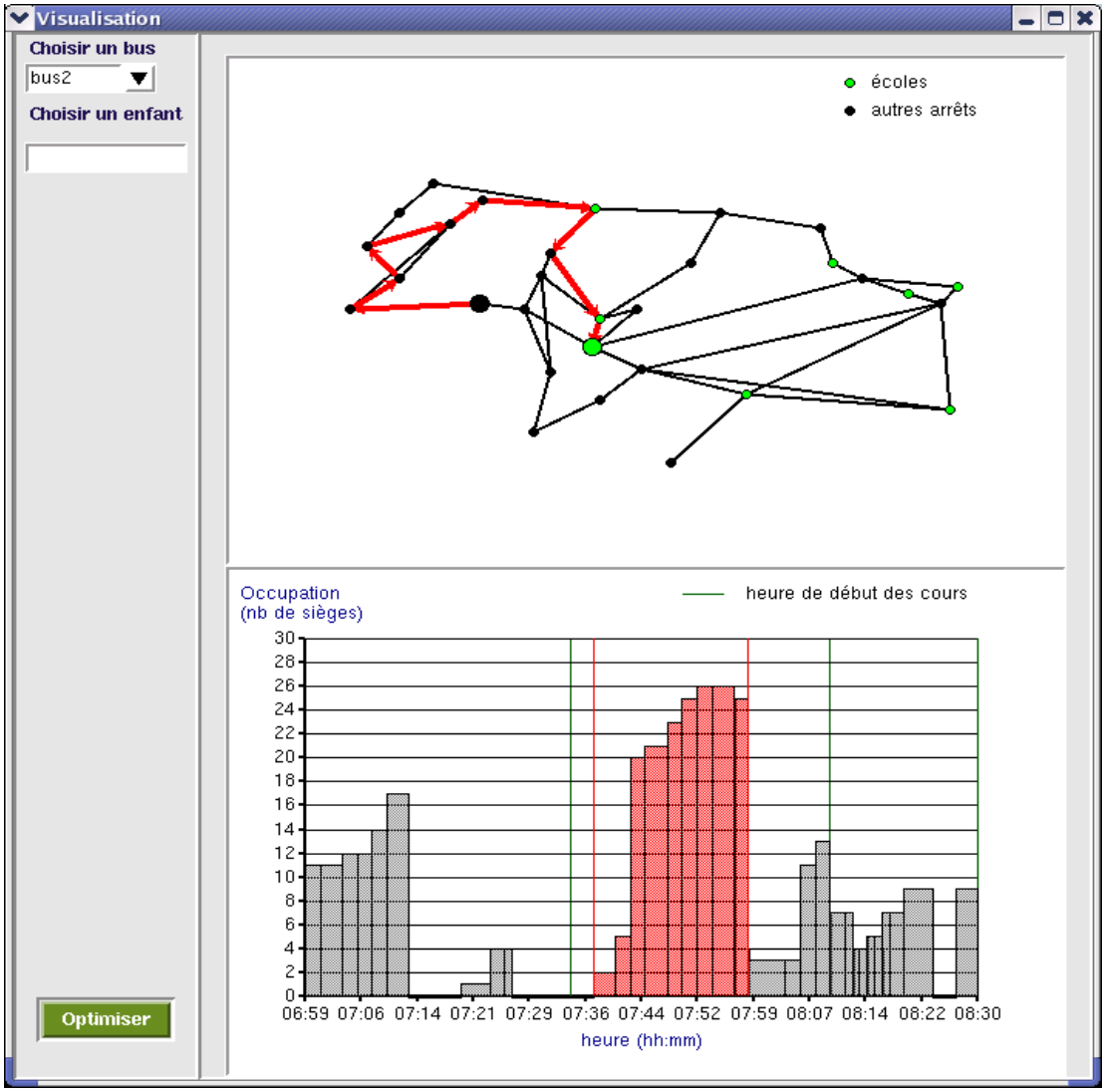


FIG. 7.2 – Tournée d'un bus et évolution de son occupation.

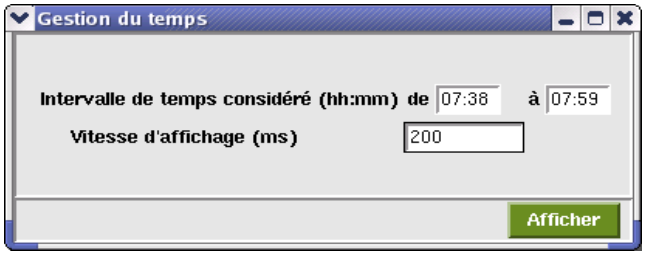


FIG. 7.3 – Spécification de l'intervalle de temps pour la visualisation animée des tournées.

7.2.2. Édition d'une planification

Notre outil permet à l'utilisateur de modifier le trajet d'un enfant et d'évaluer l'impact de cette modification sur la solution. Pour déplacer un enfant dans la planification, il suffit de cliquer sur la case correspondante de la fenêtre représentant l'occupation du bus à tout instant (voir

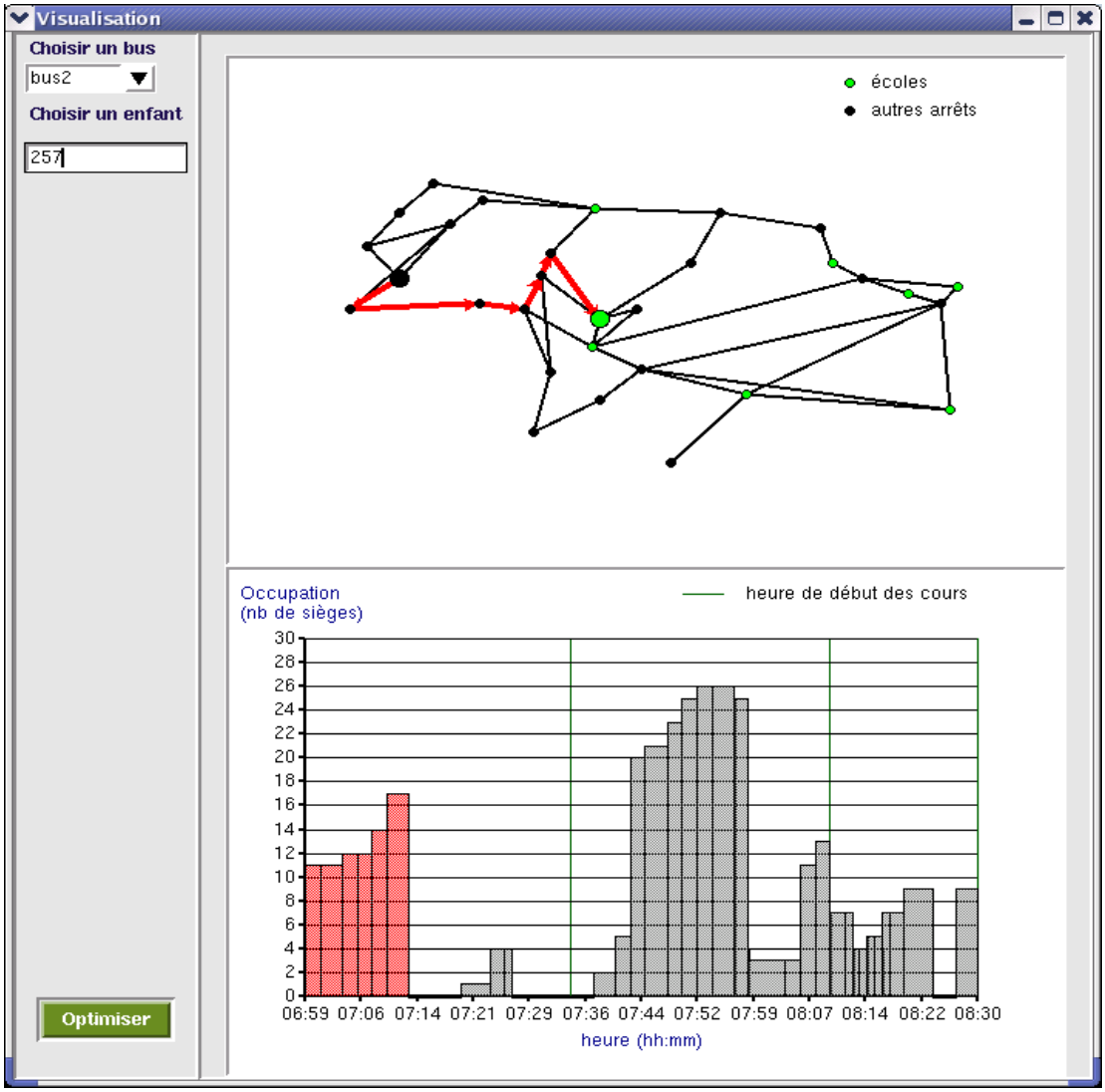


FIG. 7.4 – Trajet de l'enfant 257 dans le bus 2.

FIG. 7.5 – Occupation du bus 2 à 7h43.

figure 7.5). L'outil propose alors, pour chaque bus y compris celui qui transporte l'enfant, la planification voisine admissible (voir figure 7.6) qui minimise son temps total de trajet au sens de l'expression (3.3) (c'est-à-dire selon le voisinage admissible $\mathcal{N}_f$ décrit dans la section 3.2.2). L'impact de chacun de ces mouvements sur les fonctions objectif $\mathcal{F}_1/N_\mathcal{C}$ et $\mathcal{F}_2$ (moyenne et maximum des pertes de temps) est indiqué dans la figure 7.6. L'utilisateur est ensuite libre d'opter pour l'une des planifications voisines proposées (voir figure 7.7) ou de conserver la solution courante. Notons que, contrairement à ce qui est effectué à l'aide des méthodes décrites dans le chapitre 3, ces mouvements consistent à déplacer des enfants «réels». En effet, pour des raisons de non-discrimination, nous avons supposé dans notre modèle que tous les enfants présentant les mêmes caractéristiques, *i.e.* la même maison et la même école, effectuent le même trajet en même temps et dans le même bus. Notre interface utilisateur permet de dissocier ces groupes et donc de récupérer la flexibilité perdue.

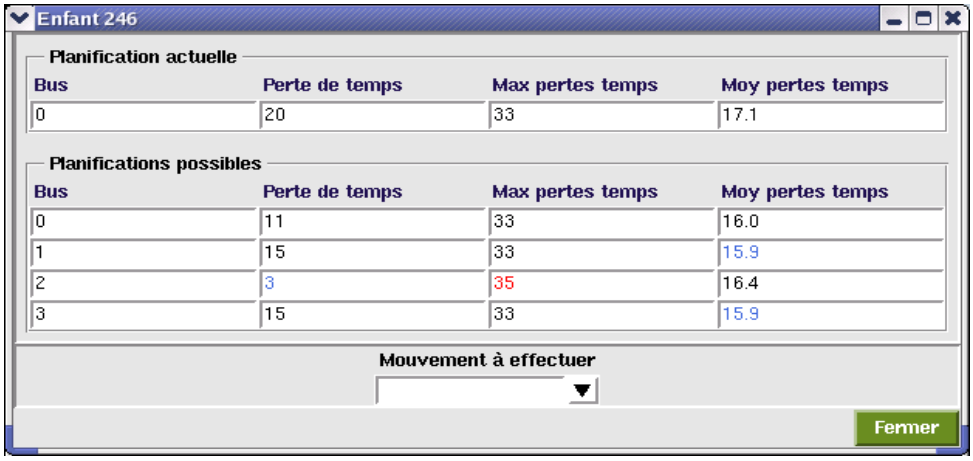


FIG. 7.6 – Mouvements possibles pour l'enfant 246.

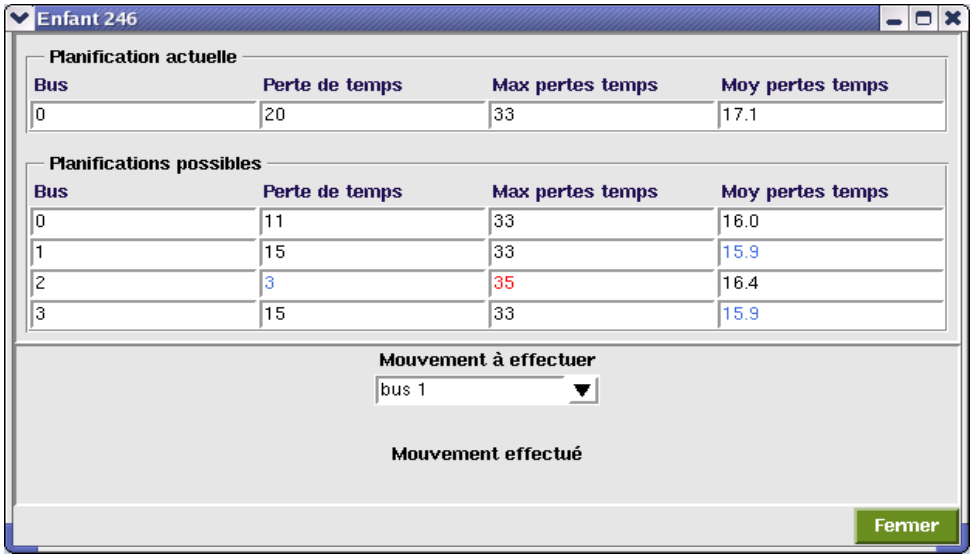


FIG. 7.7 – Déplacement de l'enfant 246 du bus 0 vers le bus 1.

7.3. Exemple d'utilisation

La planification représentée dans les figures de ce chapitre est une solution de l'instance $R_{34}(4)$ pour la fonction objectif $\mathcal{F}_2$. Elle a les caractéristiques suivantes :

- ▷ $\mathcal{F}_2 = 33$ minutes et $\mathcal{F}_1/N_{\mathcal{C}} = 17.1$ minutes ;
- ▷ l'enfant 246 monte dans le bus 0 à 7h13 et a une perte de temps de 20 minutes.

Si l'on décide de déplacer l'enfant 246, l'interface utilisateur propose les planifications reportées dans la figure 7.6. Si l'on choisit de déplacer l'enfant dans le bus 1 (voir figure 7.8), la planification obtenue présente les caractéristiques suivantes :

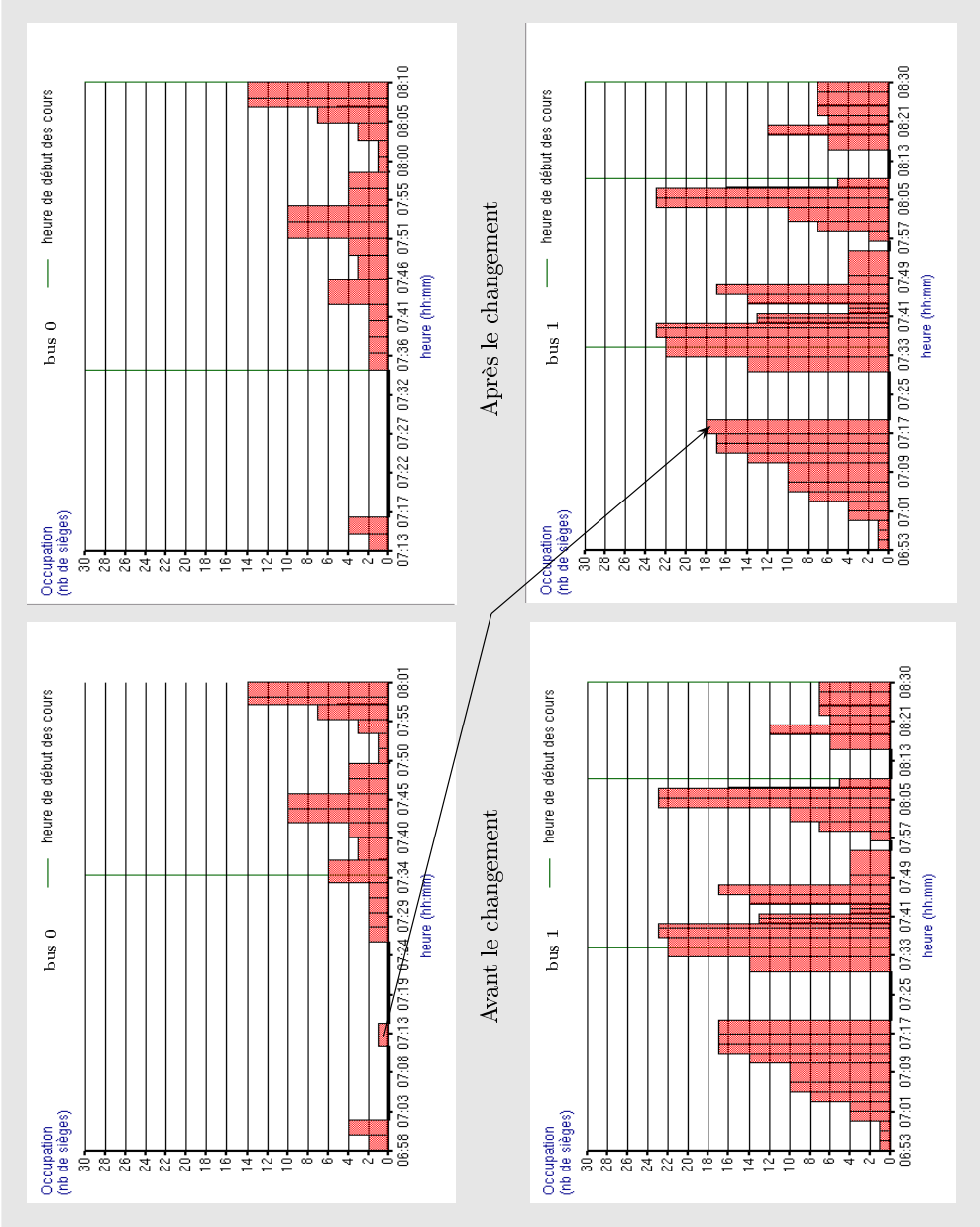


FIG. 7.8 – Déplacement de l'enfant 246 du bus 0 dans le bus 1.

- ▷ $\mathcal{F}_2 = 33$ minutes et $\mathcal{F}_1/N_{\mathcal{C}} = 15.9$ minutes ;
- ▷ l'enfant 246, déplacé dans le bus 1, part de chez lui à 7h18 et sa perte de temps passe de 20 à 15 minutes.

On constate, que cette modification n'a aucun impact sur le maximum des pertes de temps (ce maximum étant atteint par un autre enfant), alors que la moyenne des pertes de temps passe de 17.1 minutes à 15.9 minutes. Le gain de 1.2 minutes est dû au fait que le bus 0 peut maintenant commencer sa tournée à 7h13 au lieu de 6h58, délai dont bénéficient les enfants qui précédaient l'enfant 246 dans la tournée du bus 0 avant le changement.

Conclusion

Une des spécificités de ce travail se situe au niveau de la modélisation retenue. En effet, nous avons délibérément placé les enfants au centre du problème en nous focalisant sur la minimisation de leur perte de temps, alors que les précédents travaux dans le domaine cherchaient à optimiser une fonction du nombre de bus utilisés ou de la longueur des trajets effectués. La modélisation considérée se distingue également par le fait que nos planifications permettent à un véhicule de transporter simultanément des enfants devant se rendre en des écoles différentes. Notons qu'il serait même envisageable d'aller encore plus loin et d'autoriser des changements de bus en cours de trajet.

Deux problèmes ont été étudiés, l'un correspondant à la minimisation de la somme des pertes de temps (fonction objectif $\mathcal{F}_1$) et l'autre à la minimisation du maximum des pertes de temps (fonction objectif $\mathcal{F}_2$). Afin d'obtenir un point de départ en vue de l'exploration de l'espace des planifications, une procédure constructive menant à une planification admissible a été proposée. Cette solution étant généralement d'assez mauvaise qualité, deux méthodes de recherche locale, le recuit simulé et la recherche tabou, ont été adaptées afin de l'améliorer. Chacune de ces heuristiques a été proposée en deux variantes, l'une ne considérant que des planifications admissibles et l'autre autorisant la visite de planifications violant la contrainte sur la capacité des véhicules. Sans surprise, ce sont les versions non admissibles des heuristiques qui permettent d'obtenir les meilleurs résultats. Pour la fonction objectif $\mathcal{F}_1$, nos expériences numériques montrent qu'il vaut mieux utiliser le recuit simulé non admissible si le temps imparti est limité et la recherche tabou non admissible dans le cas contraire.

Pour la fonction objectif $\mathcal{F}_2$, la recherche tabou non admissible s'avère systématiquement meilleure que le recuit simulé non admissible. Ce phénomène provient du fait que la probabilité est plus grande de cycler avec le recuit simulé qu'avec la recherche tabou, car le voisinage d'une solution contient généralement de nombreuses planifications présentant le même maximum des pertes de temps. Ce problème peut être corrigé par l'ajout de critères discriminants permettant de distinguer deux planifications présentant la même valeur du maximum des pertes de temps. Une recherche à voisinage variable a également été greffée sur nos premières heuristiques. Les résultats obtenus sont intéressants, en particulier pour la fonction objectif $\mathcal{F}_2$, car l'utilisation de divers voisinages permet parfois d'échapper plus facilement à un minimum local.

Du point de vue de la personne chargée de la planification de tournées réelles, il est sans doute difficile de choisir entre la minimisation de $\mathcal{F}_1$ et celle de $\mathcal{F}_2$. En effet, l'idéal serait d'obtenir une planification présentant à la fois une petite valeur de la moyenne des pertes de temps, tout en minimisant la perte de temps maximale. Nous avons donc choisi de procéder en deux étapes : on commence par déterminer une planification minimisant $\mathcal{F}_2$, puis on la modifie en cherchant à minimiser $\mathcal{F}_1$ sans augmenter la valeur maximale précédemment trouvée. Notons que cette méthode permet parfois de sortir d'un minimum local pour la fonction objectif $\mathcal{F}_2$.

Ces heuristiques nous ont permis de construire, pour des instances réelles, des planifications de bien meilleure qualité que celles effectivement utilisées par les écoles en question. Pour l'instance

des communes de Savigny et Forel, la valeur de la fonction objectif $\mathcal{F}_1$ a pu être améliorée de 39% et celle de $\mathcal{F}_2$ de 31% par rapport à la planification actuellement utilisée. Pour l'instance de l'École Nouvelle de la Suisse Romande, la diminution est de 56% pour la fonction objectif $\mathcal{F}_1$ et de 30% pour $\mathcal{F}_2$. Notons que cette institution utilise actuellement nos méthodes pour planifier ses tournées de véhicules scolaires.

Le temps de calcul nécessaire à la construction d'une planification de bonne qualité devenant prohibitif lorsque le nombre d'enfants à transporter augmente, une approche par décomposition a été développée. Chaque sous-problème a été construit de manière à satisfaire certains critères prometteurs. Dans un premier temps, tous les enfants habitant en un même lieu sont associés au même sous-problème, mais cette répartition initiale est ensuite remise en cause au cours de l'optimisation proprement dite. Expérimentalement, cette méthode a par exemple permis de réduire d'un facteur 3 le temps de calcul nécessaire à l'obtention d'une solution d'aussi bonne qualité que la meilleure planification construite à l'aide de la recherche tabou non admissible pour une instance correspondant au transport de 2 280 enfants réels. Diverses extensions de cette méthode sont envisageables, comme par exemple la considération d'échanges consistant à déplacer un bus d'un sous problème à un autre, ou encore la prise en compte des heures de début des cours (afin d'obtenir une meilleure répartition du travail des bus sur la plage horaire considérée) lors de la confection des régions initiales.

Pour terminer, signalons qu'une interface utilisateur permettant de visualiser, d'analyser et de modifier une planification a été implémentée. Cette dernière permet non seulement de tenir compte de contraintes non modélisées, mais également d'adapter une solution en cas de modification légère des données du problème.

Bibliographie

[ACNW72] R. D. Angel, W. L. Caudle, R. Noonan, and A. Whinston, *Computer-assisted school bus scheduling*, Management Science B **18** (1972), 279–288.

[AK89] E. Aarts and J. Korst, *Simulated annealing and Boltzmann machines : A stochastic approach to combinatorial optimization and neural computing*, John Wiley & Sons, New-York, 1989.

[BB79] L. Bodin and L. Berman, *Routing and scheduling of school buses by computer*, Transportation Science **13** (1979), 113–129.

[BBPSL97] J. Braca, J. Bramel, B. Poser, and D. Simchi-Levi, *A computerized approach to the New York city school bus routing problem*, IIE Transactions **29** (1997), 693–702.

[Bea98] D. M. Beazley, *Interfacing c/c++ and python with swig*, The Seventh International Python Conference, November 10-13, 1998 - South Shore Harbour Resort, Houston, Texas, 1998.

[Ber01] R. Berger, *Planification de tournées de véhicules scolaires*, Travail pratique de diplôme, EPFL, Lausanne, Suisse, 2001.

[BG72] B. Bennett and D. Gazis, *School bus routing by computer*, Transportation Research **6** (1972), 317–326.

[BGAB83] L. Bodin, B. Golden, A. Assad, and M. Ball, *Routing and scheduling of vehicles and crews. the state of the art*, Computers and Operations Research **10** (1983), 63–211.

[BGKK99] R. Borndörfer, M. Grötschel, F. Klostermeier, and Ch. Küttner, *Telebus Berlin : Vehicle Scheduling in a Dial-a-Ride System*, Computer-aided transit scheduling. Proceedings, Cambridge, MA, USA, August 1997 (Nigel H. M. Wilson, ed.), Springer, 1999.

[BSL95] J. Bramel and D. Simchi-Levi, *A location based heuristic for general routing problems*, Operations Research **43** (1995), 649–660.

[Č85] V. Černý, *Thermodynamical approach to the traveling salesman problem : an efficient simulation algorithm*, Journal of Optimization Theory and Applications **45** (1985), 41–51.

[CFN85] G. Cornuéjols, J. Fonlupt, and D. Naddef, *The traveling salesman problem on a graph and some related integer polyhedra*, Mathematical Programming **33** (1985), 1–27.

[CH93] G. Cornuéjols and F. Harche, *Polyhedral study of the capacitated vehicle routing problem*, Mathematical Programming **60** (1993), no. 1, Ser. A, 21–52.

[CL03] J.-F. Cordeau and G. Laporte, *A tabu search heuristic for the static multi-vehicle dial-a-ride problem*, Transportation Research Part B (2003), no. 37, 579–594.

[CPR94] J. Current, H. Pirkul, and E. Rolland, *Efficient algorithms for solving the shortest covering path problem*, Transportation Science **28** (1994), 317–327.

[CW64] G. Clarke and J. W. Wright, *Scheduling of vehicle from a central depot to a number of delivery points*, Operations Research **12** (1964), 568–581.

[DDI+98] G. Desaulniers, J. Desrosiers, I. Ioachim, M. M. Solomon, F. Soumis, and D. Villeneuve, *A unified framework for deterministic time constrained vehicle routing and crew scheduling problems*, Fleet Management and Logistics (Boston, MA) (T. G. Crainic and G. Laporte, eds.), Kluwer, 1998, pp. 57–93.

[DFR+81] J. Desrosiers, J. A. Ferland, J.-M. Rousseau, G. Lapalme, and L. Chapleau, *An overview of a school busing system*, Scientific Management of Transport Systems (N. K. Jaiswal, ed.), vol. IX, International Conference on Transportation, New Delhi, November 26-28, 1980, 1981, pp. 235–243.

[DM02] E. D. Dolan and J. J. Moré, *Benchmarking optimization software with performance profiles*, Mathematical Programming, Serie A **91** (2002), 201–213.

[DR59] G. B. Dantzig and J. H. Ramser, *The truck dispatching problem*, Management Science **6** (1959), 81–91.

[EMK94] P. Eibl, R. Mackenzie, and D. Kidner, *Vehicle routing and scheduling in the brewing industry : A case study*, International Journal of Physical Distribution & Logistics Management **24** (1994), 27–37.

[Emo00] B. Emonet, *Interface graphique pour la confection d'horaires et de tournées de véhicules scolaires*, Projet de semestre, EPFL, Lausanne, Suisse, 2000.

[GDL92] C. Glardon, V. Delaloye, and Th. M. Liebling, *Referee assignment for volleyball championships : a competition between three local search heuristics*, Investigación Operativa **2** (1992), no. 3, 199–219.

[GJ79] M. R. Garey and D. S. Johnson, *Computers and intractability. A guide to the theory of NP-Completeness*, W. H. Freeman and Company, New York, 1979.

[Glo86] F. Glover, *Future paths for integer programming and links to artificial intelligence*, Computers and Operations Research **13** (1986), 533–549.

[GS79] B. Gavish and E. Shlifer, *An approach for solving a class of transportation scheduling problems*, EJOR **3** (1979), no. 2, 122–134.

[GW87] B. L. Golden and E. A. Wasil, *Computerized vehicle routing in the soft drink industry*, Operations Research **35** (1987), 6–17.

[GWKC98] B. L. Golden, E. A. Wasil, J. P. Kelly, and I. M. Chao, *Metaheuristic in vehicle routing*, Fleet Management and Logistics (Boston,MA) (T. G. Cranic and G. Laporte, eds.), Kluwer, 1998, pp. 33–56.

[Han86] P. Hansen, *The steepest ascent mildest descent heuristic for combinatorial programming*, Tech. report, Congress on Numerical Methods in Combinatorial Optimization, Capri,Italy, 1986.

[HJ87] P. Hansen and B. Jaumard, *Algorithms for the maximum satisfiability problem*, Tech. report, Rutgers University, 1987.

[HM03] P. Hansen and N. Mladenović, *Variable neighborhood search*, Handbook of Metaheuristics (F. Glover and G. A. Kochenberger, eds.), Kluwer, 2003, pp. 145–184.

[HP98] J.-K. Hao and J. Pannier, *Simulated annealing and tabu search for constraint solving*, Proceedings of the fifth international symposium on artificial intelligence and mathematics, 1998.

[JOPW86] J.-J. Jaw, A. R. Odoni, H. N. Psaraftis, and N. H. M. Wilson, *A heuristic algorithm for the multi-vehicle advance request dial-a-ride problem with time windows*, Transportation Research B **20B** (1986), no. 3, 243–257.

[KGV83] S. Kirkpatrick, C. D. Jr. Gelatt, and M. P. Vecchi, *Optimization by simulated annealing*, Science **220** (1983), 671–680.

[KH68] K. Knight and J. Hofer, *Vehicle scheduling with timed and connected calls : A case study*, Operational Research Quaterly **19** (1968), 299–310.

[Kur98] K. Kurbel, *The trade-off between solution quality and computing times of intelligent algorithms - a computational study on the role of parameters and time budgets*, Progress in Connectionist-Based Information Systems Proceedings of the 1997 International Conference on Neural Information Processing and Intelligent Systems (Kasabov, N. et al., ed.), vol. 1, 1998, pp. 604–607.

[L'É99] P. L'Écuyer, *Good parameter sets for combined multiple recursive random number generators*, Operations Research **47** (1999), no. 1, 159–164.

[LA99] M. Lutz and D. Ascher, *Learning python*, O'Reilly & Associates, 1999.

[Lap92] G. Laporte, *The vehicle routing problem : An overview of exact and approximate algorithms*, European Journal of Operational Research **59** (1992), 345–358.

[Lin65] S. Lin, *Computer solutions of the traveling salesman problem*, Bell System Tech. Journal **44** (1965), 2245–2269.

[LLKS85] E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys, *The traveling salesman problem*, Wiley, Chichester, UK, 1985.

[Men32] K. Menger, *Ergebnisse eines Mathematischen Kolloquiums*, vol. 2, ch. Das Botenproblem, pp. 11–12, Teubner, Leipzig, 1932.

[MH97] N. Mladenović and P. Hansen, *Variable neighborhood search*, Computers Operations Research **24** (1997), no. 11, 1097–1100.

[MRR+53] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, *Equation of state calculation by fast computing machines*, Journal of Chemical Physics (1953), no. 21, 1087–1091.

[NP98] D. Naddef and Y. Pochet, *The traveling salesman polytope revisited*, Discussion paper, CORE, University of Louvain-la-Neuve, Belgium, 1998.

[NSW82] K. E. Nygard, D. Summers, and R. Wagner, *Computer-aided school bus routing for rural districts*, Association for Educational Data Systems Journal **15** (1982), no. 4, 200–208.

[NT69] R. M. Newton and W. H. Thomas, *Design of school bus routes by computer*, Socio-Economic Planning Science **3** (1969), 75–85.

[PS85] F.P. Preparata and M.I. Shamos, *Computational geometry : an introduction*, Springer, New York, 1985.

[PW67] H. Pullen and M. Webb, *A computer application to a transport scheduling problem*, Computer Journal **10** (1967), 10–13.

[RSL74] D. Rosenkrantz, R. Stearns, and P. Lewis, *Approximate algorithms for the traveling-salesperson problem*, Proceedings of the 15th Annual IEEE Symposium on Switching and Automata Theory, 1974, pp. 33–42.

[RTL86] Y. Rossier, M. Troyon, and Th. M. Liebling, *Probabilistic exchange algorithms and Euclidean traveling salesman problems*, OR Spektrum **8** (1986), no. 3, 151–164.

[SB84] A. J. Swersey and W. Ballard, *Scheduling school buses*, Management Science **30** (1984), no. 7, 844–853.

[SBL03] M. Spada, M. Bierlaire, and Th. M. Liebling, *School bus routing and scheduling problem*, Operations Research Proceedings 2002 (Ulrike Leopold-Wildburger, F. Rendl, and G. Wäscher, eds.), 2003, pp. 180–186.

[SBL04] M. Spada, M. Bierlaire, and Th. M. Liebling, *Decision-aiding methodology for the school bus routing and scheduling problem*, to be published in Transportation Science, 2004.

[TN92] S. R. Thangiah and K. E. Nygard, *School bus routing using genetic algorithms*, Applications of Artificial Intelligence X : Knowledge-Based Systems, Proceeding / SPIE- the international Society for Optical Engineering, vol. 1707, 1992, pp. 387–398.

[TV02] P. Toth and D. Vigo (eds.), *The vehicle routing problem*, Siam, 2002.

[Wal04] L. Walter, *Planification de tournées de véhicules scolaires par voisinages variables*, Projet de semestre, EPFL, Lausanne, Suisse, 2004.

[Wat67] L. J. Watters, *Reduction of integer polynomial programming problems to zero-one linear programming problems*, Operations Research **15** (1967), 1171–1174.

Curriculum vitæ

Spada Michela Anna
Née le 25 mars 1974 à Genève
Nationalités suisse (GE) et italienne (BL)

Formation

1999	Diplôme du cours postgrade de recherche opérationnelle Lausanne-Grenoble : « Aide à la décision en management et technologie »
1998	Ingénieure mathématicienne diplômée EPFL Travail de diplôme à l'Université de Montréal
1993	Maturité de type A au collège de Saussure à Genève Prix de physique-chimie

Activités professionnelles

1998-2004	Assistante à l'institut de Mathématiques de l'EPFL au sein de la chaire de Recherche Opérationnelle du Professeur Th. M. Liebling
2004	Mandatée par l'École Nouvelle de la Suisse Romande (Chailly) pour l'adaptation d'un logiciel permettant de planifier des tournées de véhicules scolaires
2003	Organisation et coordination des «Journées Portes Ouvertes de l'EPFL» pour la section de mathématiques.