UNIVERSITÉ DE MONTRÉAL

PLANIFICATION ROBUSTE POUR LE STATIONNEMENT DES AUTOBUS DANS UN DÉPÔT

MOHAMED HAMDOUNI
DÉPARTEMENT DE MATHÉMATIQUES ET DE GÉNIE INDUSTRIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

THÈSE PRÉSENTÉE EN VUE DE L'OBTENTION
DU DIPLÔME DE PHILOSOPHIÆ DOCTOR (Ph.D.)
(MATHÉMATIQUES DE L'INGÉNIEUR)
MAI 2006

Your file   Votre référence
ISBN: 978-0-494-17977-2
Our file   Notre référence
ISBN: 978-0-494-17977-2

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Cette thèse intitulée :

PLANIFICATION ROBUSTE POUR LE STATIONNEMENT DES AUTOBUS DANS UN DÉPÔT

présentée par : HAMDOUNI Mohamed
en vue de l'obtention du diplôme de : Philosophiæ Doctor
a été dûment acceptée par le jury d'examen constitué de :

M. TURGEON André, Ph.D., président
M. SOUMIS François, Ph.D., membre et directeur de recherche
M. DESAULNIERS Guy, Ph.D., membre et codirecteur de recherche
M. FLEURENT Charles, Ph.D., membre
M. MAHEY Philippe, Doctorat, membre externe

À mon frère Ali

À mes parents

REMERCIEMENTS

Je tiens à remercier en premier lieu mon directeur de thèse François Soumis pour son aide financière, scientifique et morale qu'il m'a apportée durant ce long travail. Également, je tiens à remercier mon codirecteur Guy Desaulniers avec qui j'ai eu la chance de travailler en collaboration constante et étroite tout au long de cette thèse. Ensuite, je remercie le ministre de l'enseignement supérieur du Maroc pour m'avoir accordé la bourse d'exemption des frais de scolarité majorés. Cela m'a grandement aidé à faire mes études dans de bonnes conditions. J'adresse aussi un merci tout particulier aux membres du jury qui ont si aimablement accepté de siéger sur le jury de cette thèse.

Je remercie mes frères Ali et Abdoulouahad pour leur support indéfectibles au cours de ce long processus doctoral. Je remercie également mes parents, ainsi que mes sœurs qui ont fait tout ce qui était en leur pouvoir pour m'aider, comme d'accepter mon éloignement, mes longs silences et mon absence.

J'exprime ma reconnaissance envers le personnel du GERAD, plus particulièrement à Pierre Girrard pour son aide technique avec le système informatique au début de ce projet de recherche. Je ne peux passer sous silence mes amis Hassan Elhoussian et Hachimi Khlifi pour leur amitié. Monsieur Robert Guénette a aussi sa place dans mes souvenirs. Il m'a fait découvrir un beau pays et une belle ville (que je ne nommerai pas ici).

RÉSUMÉ

Dans la présente thèse, nous étudions des variantes du problème de stationnement des autobus dans un dépôt qui surviennent dans le contexte suivant. Les autobus sont de plusieurs types : à plancher bas, publicitaire, moteur puissant,... Ils arrivent en soirée au dépôt pour en ressortir le lendemain matin afin de desservir les routes du matin. Chacune de ces routes requiert un type spécifique d'autobus. Par exemple, les autobus à plancher bas sont requis dans les quartiers avec un taux élevé de personnes âgées ou handicapées et les autobus publicitaires doivent circuler sur les artères spécifiées dans le contrat de publicité.

En général, la majorité des autobus reviennent au dépôt le soir pour y passer la nuit, donc le dépôt est complètement rempli durant la nuit. Il est alors difficile de repositionner tous les autobus la nuit pour assurer une sortie ordonnée des autobus le lendemain matin. Il faut donc les garer dès leur arrivée d'une manière compatible avec l'ordre des sorties de la prochaine matinée. Cela n'est pas toujours possible et les manoeuvres de repositionnement deviennent nécessaires. Puisque les autobus ne peuvent reculer dans le dépôt, le repositionnement de deux autobus d'une même voie implique des manoeuvres qui consistent à faire sortir tous les autobus de la voie avant de les rentrer dans le bon ordre.

Les manoeuvres nécessitent un travail additionnel la nuit et causent des dérangements pour les voisins. Pour éviter les manoeuvres, les compagnies de transport permettent de servir certaines routes par des autobus de types différents de ceux requis. Lorsqu'une route se voit affecter un autobus d'un type différent de celui demandé, on parle alors d'affectation erronée.

Dans une première étape, nous examinons le problème de stationnement dans un contexte déterministe (les heures d'arrivée et de départ sont connues) où les autobus arrivent et partent aux heures planifiées. Cependant, en pratique, les autobus arrivent avant ou après les heures prévues. Afin d'augmenter la robustesse de la solution, nous introduisons un nouveau mode de stationnement des autobus qui est appelé *le mode de stationnement par bloc* où chaque voie contient au plus deux types d'autobus et les autobus de chaque type forment un bloc. Le problème de stationnement des autobus en utilisant le mode de stationnement par bloc est nommé *le problème de stationnement par bloc.* Nous démontrons d'abord que ce problème est NP complet. Nous étudions ensuite deux de ses variantes. La première minimise le nombre de voies à manoeuvrer. La deuxième minimise le nombre d'affectations erronées.

Dans certains cas, l'arrivée des autobus subit des variations importantes par rapport à l'ordre d'arrivée prévue, et cela peut rendre la solution prédéterminée inutilisable parce qu'elle nécessiterait un trop grand nombre de manoeuvres ou produirait trop d'affectations erronées. Pour obtenir des solutions plus robustes qui résistent mieux aux changements dans l'ordre d'arrivée, nous considérons le problème de stationnement par bloc en considérant que l'arrivée des autobus est stochastique.

Une première contribution de la thèse est de proposer une formulation mathématique pour la variante qui minimise le nombre de voies à manoeuvrer et de démontrer que cette formulation produit des solutions plus robustes que celles qui peuvent être engendrées par les méthodes existantes dans la littérature. Les expérimentations effectuées sur des données fournies par la STM (Société de Transport de Montréal) montrent l'efficacité des modèles proposés. Nous avons résolu avec CPLEX des problèmes de grande taille en des temps de calcul raisonnables.

Une deuxième contribution consiste à formuler la variante qui minimise le nombre d'affectations erronées comme un programme linéaire en nombres entiers de très grande taille, et à développer une approche de résolution qui utilise la décomposition de Benders dans un contexte où les variables du sous-problème doivent prendre des valeurs entières. Nous avons testé cette approche sur des données provenant de la STM et avons constaté qu'elle permettait de résoudre en des temps de 20 à 30 minutes des instances moyennes qui ne pouvaient l'être par CPLEX en moins de 20 heures.

Comme troisième contribution, nous introduisons deux approches pour traiter les perturbations dans l'ordre d'arrivée : une approche déterministe nommé *k-position* (k est un entier positif) et une autre stochastique qui considère des heures d'arrivée stochastiques. L'approche k-position produit des solutions planifiées qui restent réalisables pendant les opérations même si les autobus s'écartent d'au plus k positions de l'ordre planifié. Un modèle linéaire en nombres entiers qui minimise le nombre de voies avec deux blocs est présenté pour cette variante du problème. Dans l'approche stochastique, nous proposons un modèle linéaire en nombres entiers qui vise à minimiser l'espérance d'une fonction corrélée avec le nombre d'autobus qui rendent la solution planifiée non réalisable parce qu'ils arrivent trop tôt ou trop tard par rapport à l'heure d'arrivée prévue. Les coefficients de la fonction objectif sont calculés par simulation.

Les deux approches proposées ont été expérimentées sur des instances réelles de grande et moyenne tailles provenant de la STM. Les résultats numériques montrent que les deux approches produisent des solutions plus robustes par rapport aux solutions obtenues par une approche qui ne prend pas en compte la stochasticité du problème. De plus, l'approche stochastique performe mieux que l'approche $k-$position pour les instances pour lesquelles il est difficile de trouver une solution robuste.

ABSTRACT

In this thesis, we study variants of the problem of dispatching buses in a depot that arise in the following context. Buses are of different types: low-floor, advertising, extra power,... They arrive at the depot in the evening to be dispatched in the next morning to morning routes. Each of these routes requires a specific bus type. For example, low-floor buses are required in districts with a high proportion of elderly or handicapped people and advertising buses are required on lines specified in by the advertising contracts.

In general, most buses spend the night in the depot, so the depot is completely full during that period. It is difficult to move buses during the night and to park them in the right order for the next morning departures. Thus, buses have to be parked upon their arrival in an order compatible with the next morning departures. This is not always possible and manoeuvers are necessary to achieve this goal. Since buses are not allowed to go backward in the depot, maneuvring two buses of the same lane implies moving out of the depot all buses parked in this lane before parking them in the right order.

Maneuvers require additional night-time personnel to reorder the buses and might be disturbing close to a residential neighbourhood. To avoid maneuvers, certain transit authorities allow to assign bus types that are not required on bus routes. Such an assignement is called a type mismatch.

In the first part of this thesis, we study the bus dispatching problem in a deterministic context where buses arrive and leave as scheduled (the time and type of each evening arrival and each morning departure are known). However, in practice, buses arrive sooner or later. In order to increase the robustness of the solution, we introduce a new parking mode, called parking by block, where each lane contains at most two bus types and the buses of each type form a block. The problem of dispatching buses in a depot using blocks is called the *bus dispatching problem by block*. We show that this problem is NP-complete. We study two variants of this problem. In the first, the number of lanes with manoeuvers is minimized. In the second, the number of mismatches is minimized.

In same cases, buses rarely respect their arrival schedule and the planned solution often becomes impractical because it yields additionnal maneuvers or mismatches. To generate robust solutions that resist more to changes in the planned arrivals, we study in the second part of this thesis the bus dispatching problem by block considering stochastic arrival times.

The first contribution of the thesis is to propose two mathematical formulations for the variant of the bus dispatching problem by block that minimizes the number of lanes with maneuvers and show that parking buses by block yields much more robust solutions than parking them without respecting a block structure. Experiments performed on data provided by the STM (Société de Transport de Montréal) show the efficiency of the proposed models. Using CPLEX, large size instances can be solved in short computational times.

A second contribution is to propose a mathematical formulation for the second variant that minimizes the number of mismatches, and a solution approach based on Benders decomposition that yields a discrete subproblem. To handle the subproblem integrality constraints, we propose strong pruning criteria and a specialized branching scheme that imposes decisions on the master problem variables. We tested the proposed model and approach on data provided by the STM and observed that instances which could not be solved directly using CPLEX within 20 hours, were solved in less than 30 minutes by the proposed approach.

Finally, as a third contribution, we introduce two approaches for handling perturbations in the arrival order: one deterministic approach named the *$k-$position* approach, (k is a given nonnegative integer) and another stochastic approach that considers stochastic arrival times. The $k-$position approach produces a planned solution that remains feasible during the operations when any bus is shifted by at most k positions from the planned arrival order. An integer linear model that minimizes the number of two-block lanes is presented for this variant of the problem. In the stochastic approach, we propose a linear integer program that minimizes, under stochastic arrival times, the expectation of a function positively correlated with the number of buses that make the planned solution infeasible because they arrive too late or too early. The coefficients of the objective function are computed by a simulation procedure.

The two proposed approaches were tested on real-life instances of large and medium sizes from the STM. The numerical results show that the two approaches yield more robust solutions than an approach which does not take into account the stochasticity of the problem. Also, they show the stochastic approach performs better than the $k-$position approach on instances for which very robust solutions do not exist.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES ALGORITHMES

CHAPITRE 1 : INTRODUCTION

Les sociétés de transport utilisent le dépôt (garage) pour garer les autobus lorsqu'ils sont inactifs, subissent des opérations d'entretien ou font le plein d'essence. En général, le dépôt est constitué d'un ensemble de voies de tailles variées dans lesquelles les autobus doivent être stationnés un derrière l'autre ; chaque voie regroupe un nombre de positions de même taille. Une position est une place réservée pour le stationnement d'un autobus. Chaque voie a deux extrémités : une extrémité est utilisée pour l'entrée des autobus et l'autre pour la sortie. Notons que, pour des raisons de sécurité, les autobus ne peuvent reculer dans le dépôt. La configuration des voies varie d'un dépôt à l'autre. Il existe des dépôts avec une configuration simple où toutes les voies sont parallèles. Il en existent d'autres avec une configuration complexe où certaines voies sont perpendiculaires à d'autres et bloquent l'entrée de celles-ci. Ce type de configuration entraîne des contraintes de préséance pour le remplissage des voies.

Une société de transport possède plusieurs autobus pour desservir les parcours d'autobus. Afin d'assurer une bonne qualité de service, il est préférable d'utiliser un type particulier d'autobus pour certains parcours. En effet, les autobus sont de plusieurs types. On trouve par exemple des autobus à plancher bas qui doivent circuler dans les quartiers avec un taux élevé de population âgée ou handicapée ; des autobus avec un moteur puissant qui sont requis dans les pentes abruptes. Il y a aussi des autobus de type publicitaire qui devraient circuler sur les artères spécifiés dans les contrats. Le nombre de types d'autobus rend le problème de gestion des entrées et sorties des autobus dans un dépôt plus compliqué.

Les autobus quittent le dépôt en matinée selon un ordre appelé *ordre de départ* où le type et l'heure de chaque départ sont connus. Ils y retournent le soir selon un ordre prévu nommé *ordre d'arrivée.* Le type et l'heure d'arrivée de chaque autobus sont connus. Étant donné une suite d'arrivées le soir et une suite de départs en matinée, le problème de stationnement des autobus consiste à stationner les arrivées dans un ordre compatible avec les départs de la prochaine matinée. Par exemple, si un départ requiert un type particulier d'autobus, il faut que ce type soit accessible au moment de ce départ. Un autobus devient accessible quand tous les autobus stationnés devant lui dans sa voie sont partis.

Ce problème peut être devisé en un problème de planification et un problème d'opération. Une solution planifiée du problème de stationnement des autobus, consiste à identifier le type d'autobus à affecter à chaque position du dépôt. Les décisions d'opération : chaque arrivée (resp. départ) selon l'ordre planifié est affectée à une position de bonne type. Cette thèse présente des modèles pour le problème de planification produisant des solutions réalisables pour l'opération. On s'intéresse à une classe des solutions *robustes* pour le problème de stationnement des autobus. Une solution de planification est plus robuste si l'opération reste réalisable pour plusieurs scénarios de perturbations dans l'ordre d'arrivée planifié.

Comme indiqué dans le paragraphe précédent, en principe, l'heure d'arrivée de chaque autobus est connue à l'avance. Mais, en pratique, les autobus arrivent en avance ou en retard de l'heure prévue et cela cause des changements dans l'ordre planifié. De plus, après l'heure de pointe en après-midi, le taux d'arrivées par minute est assez élevé (quatre à cinq arrivées par minute) ce qui laisse peu de temps au gareur pour décider dans quelle voie doit être stationné l'autobus arrivé. Afin d'augmenter la robustesse des solutions du problème de stationnement des autobus pour réduire les effets négatifs des perturbations dans l'ordre d'arrivée, nous introduisons un nouveau mode de stationnement des autobus qu'on appelle le *mode de stationnement par bloc.* Ce mode partitionne chaque voie du dépôt en au plus deux blocs, chaque bloc contenant un seul type d'autobus. Le problème de stationnement utilisant le mode de stationnement par bloc est nommé *le problème de stationnement par bloc.* Une solution planifiée par bloc, donne au gareur une certaine flexibilité au moment de l'opération. Il faut placer dans un bloc les autobus du même type dans l'ordre où ils arrivent plutôt que dans l'ordre prévu sans problème pour les départs.

Cette thèse porte sur le problème de stationnement des autobus par bloc. Elle s'attaque aux différentes variantes de ce problème qui se différencient par la nature des arrivées (déterministes ou stochastiques) et la fonction objectif retenue. Au début, nous examinons le problème de stationnement par bloc dans un contexte déterministe où l'heure et le type de chaque arrivée sont connus. Plus tard, ce problème est étudié dans un contexte stochastique où les autobus peuvent s'écarter de l'ordre d'arrivée prévu.

Comme la majorité des autobus retournent au dépôt le soir pour y passer la nuit, la congestion du dépôt est maximal durant la nuit. Cela restreint l'espace disponible dans le dépôt et il devient difficile d'effectuer des manoeuvres de repositionnement pour assurer une sortie ordonnée des autobus pour la prochaine matinée. Par conséquent, il est fort souhaitable de bien stationner les autobus dès leur arrivée dans un ordre compatible avec les prochains départs. Mais, ce n'est pas toujours possible. Au besoin, il est permis d'effectuer des manoeuvres dans certaines voies. Puisque les autobus ne peuvent reculer dans le dépôt, le repositionnement de deux autobus d'une même voie implique des manoeuvres qui consistent à faire sortir tous les autobus de la voie avant de les rentrer dans le bon ordre. Le coût de repositionnement des autobus d'une même voie est donc le même, indépendamment du nombre de respositionnements à réaliser dans cette voie.

Ensuite, nous proposons des modèles linéaires en nombres entiers pour le problème de stationnement par bloc qui minimise le nombre de voies à manoeuvrer, appelées voies mixtes. Comme les solutions avec au plus deux blocs sont fortement désirées, nous étudions la variante du modèle dont la fonction objectif minimise le nombre de voies mixtes et ces voies peuvent être transformés en voies avec deux blocs après manoeuvres. Puisque les instances de cette variante ne sont pas toujours réalisables, nous considérons aussi la variante qui n'oblige pas les voies mixtes à avoir deux blocs après manoeuvres. Pour montrer l'efficacité des modèles proposés, nous les avons testés sur plusieurs jeux de données découlant d'une instance réelle.

Afin d'éviter les manoeuvres, certaines sociétés de transport ont recours à une autre alternative qui consiste à minimiser le nombre d'affectations erronées. Une affectation erronée se produit lorsqu'un autobus d'un type donné est affecté à un départ qui requiert un type différent. Avant d'aborder le problème avec affectations erronées, nous nous penchons sur un autre objectif qui vise à maximiser la robustesse. Cet objectif minimise les incompatibilités avec les blocs (à l'arrivée et au départ). Une incompatibilité avec un bloc est une arrivée ou un départ affecté à un bloc de type différent. Nous formulons le problème de stationnement par bloc en minimisant les incompatibilités aux blocs comme un programme linéaire en nombres entiers. Ensuite, nous présentons un modèle linéaire de grande taille qui minimise les affectations erronées et proposons une approche basée sur la décomposition de Benders pour le résoudre où les variables du sous-problème doivent prendre des valeurs entières. Afin de pallier cette dificulté, nous développons une stratégie de branchement qui impose des décisions sur les variables du problème maître et fait à critères d'élagage efficaces. Ici encore, nous rapportons des résultats numériques pour montrer la performance de la méthode proposée.

La prise en compte des perturbations dans l'ordre d'arrivée est un facteur important à considérer dans le processus de planification pour le stationnement des autobus dans un dépôt. Nous abordons donc à la fin de cette thèse le problème de stationnement par bloc dans un contexte stochastique où les autobus peuvent dévier de leur ordre d'arrivée prévu, soit qu'ils arrivent plus tôt ou plus tard par rapport à l'ordre prévu. Dans ce contexte, il devient nécessaire de produire des solutions encore plus robustes qui résistent aux variations dans l'ordre d'arrivée. Nous proposons deux approches pour prendre en compte la possibilité de changements dans l'ordre planifié. La première est une approche déterministe, nommé *k-position* (k est un entier positif donné), qui considère que chaque arrivée peut dévier avec au plus k positions par rapport à son ordre planifié. Pour la variante k-position du problème de stationnement par bloc, nous présentons un modèle linéaire en nombres entiers qui minimise le nombre de voies avec deux blocs. Comme nous verrons au chapitre 6, l'approche k-position présente certains inconvénients, qui réduisent son efficacité en pratique. Une deuxième approche stochastique considère que les heures d'arrivée des autobus sont stochastiques. Nous formulons cette variante stochastique du problème de stationnement par bloc comme un problème linéaire en nombres entiers qui minimise l'espérance d'une fonction corrélée avec le nombre d'autobus qui rendent la solution planifiée non réalisable parce qu'ils arrivent trop tôt ou trop tard. Les coefficients de la fonction objectif sont calculés à l'aide d'une procédure de simulation. Les deux variantes ($k-$position et stochastique) sont expérimentées sur des jeux de données de la STM.

Cette thèse est structurée comme suit. Le chapitre 2 présente une revue de littérature sur les problèmes de stationnement de véhicules. Au chapitre 3, nous décrivons l'organisation du coeur de cette thèse. Le chapitre 4 s'attaque au problème de stationnement des autobus par bloc avec manoeuvres tandis que le chapitre 5 s'interesse à la variante avec affectations erronées. Le chapitre 6 étudie le problème de stationnement par bloc avec des heures d'arrivée stochastiques. Finalement, le dernier chapitre présente nos conclusions.

CHAPITRE 2 : REVUE LITTÉRATURE

La littérature relative au problème de stationnement des autobus est très restreinte. On trouve deux travaux principaux dans ce domaine. Le premier a été effectué par Winter et Zimmermann (2000). Ils ont présenté un modèle d'affectation quadratique pour ce problème. En 2001, Gallo et Di Miele ont proposé un autre modèle linéaire à deux niveaux pour le problème de stationnement. Les deux travaux sont présentés en détails dans les prochaines sections. Une dernière section est dédiée aux travaux complémentaires qui sont en lien avec le problème en question.

2.1 Les modèles de Winter et Zimmermann

Winter dans sa thèse (1999) et Winter et Zimmermann (2000) ont étudié le problème de stationnement des trains dans un dépôt. Un dépôt pour le stationnement des trains est différent de celui utilisé pour le stationnement des autobus. Pour les trains, le dépôt est constitué d'un ensemble de voies où chaque voie a une seule extrémité utilisée pour l'entrée et la sortie des trains. Winter et Zimmermann considèrent que le dépôt est partitionné en R piles (voies) et chaque pile $r \in \{1, 2, ..., R\}$ est partitionnée en P_r positions de taille égale. Une position correspond à une place pour stationner un train. Toutes les positions sont numérotées du bas au sommet de la pile. L'ensemble $\mathcal{P}$ représente toutes les positions du dépôt et $\mathcal{P}_r$ représente l'ensemble des positions de la pile r.

Soient $\mathcal{A} = \{1, 2, 3, ..., N\}$ la suite d'arrivées, $\mathcal{D} = \{1, 2, 3, ..., N\}$ la suite de départs et $\mathcal{T} = \{1, 2, 3, ..., T\}$ l'ensemble des types de trains. Appelons événement une arrivée ou un départ. Considérons les paramètres suivants

$$\theta_{ij} = \begin{cases} 1 & \text{si l'arrivée } i \text{ et le départ } j \text{ sont de même type} \\ 0 & \text{sinon} \end{cases}$$

$$\alpha_{ij} = \begin{cases} 1 & \text{si l'événement } i \text{ survient avant l'événement} j \ (i < j) \\ 0 & \text{sinon} \end{cases}$$

$$\beta_{qp} = \begin{cases} 1 & \text{si } q > p \text{ pour } p, q \in P_r \text{ et } r \in \{1, ..., R\} \\ 0 & \text{sinon} \end{cases}$$

et définissons les variables

$$x_{iq} = \begin{cases} 1 & \text{si l'arrivée } i \text{ est affectée à la position } q, \\ 0 & \text{sinon} \end{cases}$$

$$y_{jq} = \begin{cases} 1 & \text{si le départ } j \text{ est affecté à la position } q, \\ 0 & \text{sinon.} \end{cases}$$

Définissons l'ensemble $\mathcal{N}$ des paires d'arrivée et départ non-compatibles, i.e., $(i, j) \in \mathcal{N}$ si et seulement si $\theta_{ij} = 0$. Deux arrivées i et j affectées aux positions l et q dans une même pile sont manœuvrées à l'arrivée si et seulement si $\alpha_{ij}\beta_{lq} = 1$. Définissons ainsi l'ensemble

$$\mathcal{L}_a = \{(i, j, l, q) \in (\mathcal{A}, \mathcal{A}, \mathcal{P}, \mathcal{P}) \mid \alpha_{ij}\beta_{lq} = 1\}.$$

On définit également pour les départs l'ensemble

$$\mathcal{L}_d = \{(i, j, l, q) \in (\mathcal{D}, \mathcal{D}, \mathcal{P}, \mathcal{P}) \mid \alpha_{ij}\beta_{lq} = 1\}$$

Notons que l'affectation d'un départ j à une position q signifie l'affectation du départ j au train stationné à la position q. Winter et Zimmermann ont étudié deux variantes du problème de stationnement des trains. La première minimise le nombre de manoeuvres et la deuxième minimise le nombre d'affectations erronées. Ils ont démontré que le problème de stationnement est un problème NP complet. Dans les prochaines sections, nous présentons les modèles proposés pour chaque variante.

2.1.1 Variante avec manoeuvres

Winter et Zimmermann ont modélisé le problème de stationnement des trains en minimisant les manoeuvres comme un problème d'affectation quadratique. Le modèle proposé, noté MSP pour Minimizing Shunting Problem, est donné ci-dessous.

$$
\begin{array}{llll}
(MSP)\min & \displaystyle\sum_{q\in\mathcal{P}}\sum_{p\in\mathcal{P}}\left(\sum_{i\in\mathcal{A}}\sum_{j\in\mathcal{A}}\alpha_{ij}\beta_{qp}x_{iq}x_{jp}+\sum_{j\in\mathcal{D}}\sum_{k\in\mathcal{D}}\alpha_{jk}\beta_{pq}y_{jq}y_{kq}\right) & & (2.1)\\
\text{s.à} & \displaystyle\sum_{i\in\mathcal{A}}x_{iq}=1 & \forall q\in\mathcal{P}, & (2.2)\\
& \displaystyle\sum_{q\in\mathcal{P}}x_{iq}=1 & \forall i\in\mathcal{A}, & (2.3)\\
& \displaystyle\sum_{j\in\mathcal{D}}y_{jq}=1 & \forall q\in\mathcal{P}, & (2.4)\\
& \displaystyle\sum_{q\in\mathcal{P}}y_{jq}=1 & \forall j\in\mathcal{D}, & (2.5)\\
& x_{iq}+y_{jq}\leq 1 & \forall q\in\mathcal{P},(i,j)\in\mathcal{N}, & (2.6)\\
& x_{iq},y_{jq}\in\{0,1\} & \forall\, i\in\mathcal{A},j\in\mathcal{D},q\in\mathcal{P}. & (2.7)
\end{array}
$$

L'objectif (2.1) minimise le nombre de manoeuvres à l'arrivée et au départ. Les contraintes (2.2) et (2.3) assurent que chaque arrivée est affectée à une seule position et chaque position reçoit une seule arrivée. Les contraintes (2.4) et (2.5) sont symétriques à (2.2) et (2.3) en remplaçant arrivée par départ. Les contraintes (2.6) assurent que chaque départ est affecté à un train de bon type. Les contraintes (2.7) sont des contraintes d'intégrité. Les trains i et j sont manoeuvrés à l'arrivée si et seulement si $\alpha_{ij}\beta_{qp}=1$.

Pour résoudre efficacement le modèle (MSP), les auteurs ont utilisé la méthode de linéarisation de Kaufman et Broeckx. Avant de donner le modèle linéaire, définissons les variables et les paramètres suivants :

$$s_{iq}=\sum_{j\in\mathcal{D}}\sum_{q\in\mathcal{P}}\alpha_{ij}\beta_{qp}x_{jq}\quad\text{et}\quad d_{iq}=\sum_{j\in\mathcal{D}}\sum_{p\in\mathcal{P}}\alpha_{ij}\beta_{qp}.$$

Le modèle linéaire MSPLIN obtenu est

$$
\begin{aligned}
(MSPLIN)\ \min \sum_{i\in A}\sum_{q\in P} z_{iq} & & \quad (2.8)\\
\text{s.à} \ \sum_{i\in\mathcal{A}} x_{iq} = 1 & \quad \forall q\in\mathcal{P}, & \quad (2.9)\\
\sum_{q\in\mathcal{P}} x_{iq} = 1 & \quad \forall i\in\mathcal{A}, & \quad (2.10)\\
\sum_{j\in\mathcal{D}} y_{jq} = 1 & \quad \forall q\in\mathcal{P}, & \quad (2.11)\\
\sum_{q\in\mathcal{P}} y_{jq} = 1 & \quad \forall j\in\mathcal{D}, & \quad (2.12)\\
s_{iq} + d_{iq}x_{iq} - z_{iq} \leq d_{iq} & \quad \forall i\in\mathcal{A},\ q\in\mathcal{P}, & \quad (2.13)\\
x_{iq} + y_{jq} \leq 1 & \quad \forall (i,j)\in\mathcal{N}, q\in\mathcal{P}, & \quad (2.14)\\
y_{jq} + y_{kl} \leq 1 & \quad \forall (i,j,l,p)\in\mathcal{L}_d, & \quad (2.15)\\
x_{iq}, y_{jq} \in \{0,1\} & \quad \forall i\in\mathcal{A}, j\in\mathcal{D}, q\in\mathcal{P}, & \quad (2.16)\\
z_{iq} \geq 0 & \quad \forall i\in\mathcal{A}, j\in\mathcal{D}, q\in\mathcal{P}. & \quad (2.17)
\end{aligned}
$$

Le modèle linéaire MSPLIN minimise le nombre de manoeuvres à l'arrivée. Les contraintes (2.15) les empêchent au départ parce que l'écart entre deux départs consécutifs est très petit (une à deux minutes) ce qui ne laisse pas du temps pour effectuer des manoeuvres. Winter a démontré dans sa thèse que le modèle MSPLIN est équivalent au modèle MSP.

2.1.2 Variante avec affectations erronées

Winter et Zimmermann ont aussi étudié le problème de stationnement des trains en minimisant les affectations erronées. Ils ont proposé le modèle linéaire binaire présenté ci-dessous, noté MTM pour Minimizing Type Mismatches.

$$
\begin{aligned}
(MTM)\ \min \sum_{q\in\mathcal{P}}\sum_{i\in\mathcal{A}}\sum_{j\in\mathcal{D}} \theta_{ij}|x_{iq}-y_{jq}| & & & \quad (2.18)\\
\text{s.à} \qquad \sum_{i\in\mathcal{A}} x_{iq} &= 1 & \forall q \in \mathcal{P}, & \quad (2.19)\\
\sum_{q\in\mathcal{P}} x_{iq} &= 1 & \forall i \in \mathcal{A}, & \quad (2.20)\\
x_{iq} + x_{jl} &\leq 1 & \forall (i,j,q,l) \in \mathcal{L}_a, & \quad (2.21)\\
\sum_{q\in\mathcal{P}} y_{jq} &= 1 & \forall j \in \mathcal{D}, & \quad (2.22)\\
\sum_{j\in\mathcal{D}} y_{jq} &= 1 & \forall q \in \mathcal{P}, & \quad (2.23)\\
y_{jq} + y_{kl} &\leq 1 & \forall (j,k,q,l) \in \mathcal{L}_d, & \quad (2.24)\\
x_{iq}, y_{jq} &\in \{0,1\} & \forall i \in \mathcal{A}, j \in \mathcal{D}, q \in \mathcal{P}. & \quad (2.25)
\end{aligned}
$$

L'objectif (2.18) minimise le nombre d'affectations erronées. Les contraintes (2.19), (2.20), (2.22), (2.23) et (2.25) peuvent être expliquées de la même façon que les contraintes (2.2), (2.3), (2.4), (2.5) et (2.6), respectivement. Les contraintes (2.21) et (2.24) empêchent complètement les manoeuvres à l'arrivée et au départ.

2.1.3 Résultats

Les modèles MSPLIN et MTM ont été résolus par CPLEX. Les résultats numériques rapportés dans l'article montrent que Winter et Zimmermann ont résolu des instances de 27 trains, 9 types et 5 voies en 50 minutes pour le problème MSPLIN. Pour améliorer les temps de résolution, ils ont développé des heuristiques qui produisent des solutions dans un temps raisonnable (moins de deux minutes). L'heuristique LIFO a été considérée comme l'heuristique la plus efficace. LIFO a affecté une arrivée de type τ au dernier départ non affecté de même type, puis l'affecte à une position dans une voie.

Pour le modèle MTM, que des instances de petite taille (14 trains, 5 types et 5 voies) ont été résolues avec CPLEX. De nouveau, l'heuristique LIFO été utilisée pour résoudre des instances de moyenne taille du problème MTM.

2.1.4 Le problème de stationnement des trains en temps réel

En pratique, l'arrivée des trains au dépôt subit des variations importantes dans l'ordre d'arrivée. En présence de ces variations, l'utilisation de la solution planifiée augmente considérablement le nombre de manoeuvres et d'affectations erronées. Une mise à jour de la solution planifiée devient indispensable pour minimiser le nombre de manoeuvres et d'affectations erronées. On appelle ce problème, le problème de stationnement en temps réel.

Winter et Zimmermann (2000) ont étudié le problème de stationnement des trains en temps réel. Ils ont proposé deux algorithmes pour la mise à jour de la solution planifiée lors des opérations suite à l'observation d'une perturbation. La première consiste à planifier une affectation partielle des arrivées et des départs aux positions. Une affectation partielle concerne que les arrivées et les départs impliqués dans le changement d'ordre d'arrivée. Le second algorithme consiste à planifier une affectation partielle pour les arrivées et une affectation complète des départs aux positions. Ces deux algorithmes ont été appliqués sur des instances réelles et aléatoires.

2.2 Modèle à deux niveaux

Un autre travail dans le domaine du stationnement des autobus est celui de Gallo et Di Miele (2001). Les auteurs ont présenté un modèle linéaire à deux niveaux. Dans un premier niveau, ils affectent les autobus et les départs aux voies. Dans le deuxième niveau, ils réarrangent les autobus dans les voies pour qu'ils soient compatibles avec les départs de la prochaine matinée. Dans la suite, on présente le modèle ainsi que la méthode de résolution proposée. Avant de donner le modèle, on introduit les paramètres et les variables suivants.

Soit $V = \{1, 2, 3, ..., n\}$ la suite d'arrivées, et $D = \{1, 2, 3, ..., n\}$ la suite de départs. Notons par t'_i le type de l'arrivée i, et l'_i sa taille, t''_j le type d'autobus requis par le départ j et l''_j sa taille. Soit aussi $C = \{1, 2, 3, ..., m\}$ l'ensemble de toutes les voies du dépôt. Le problème consiste à déterminer m suites ordonnées $V_1, V_2, ..., V_m$ et $D_1, D_2, ...,$ D_m de telle manière que, pour chaque $k \in C$, on a :

$$
\begin{aligned}
V_k &= \{i_1, ..., i_{n_k}\} && && (2.26)\\
D_k &= \{j_1, ..., j_{n_k}\} && && (2.27)\\
i_h &\geq i_{h+1} \text{ et } j_h \geq j_{h+1} && h = 1, ..., n_k - 1, && (2.28)\\
t'_{i_h} &= t''_{j_h} && h = 1, ..., n_k, && (2.29)\\
\sum_{h=1}^{n_k} l'_{i_h} &= \sum_{h=1}^{n_k} l''_{j_h} \leq L_k. && && (2.30)
\end{aligned}
$$

L'ordre de l'ensemble V_k correspond à l'ordre dans lequel les arrivées sont positionnées dans les voies. L'ordre des éléments de l'ensemble D_k correspond à l'ordre des départs associés à la voie k. Les conditions (2.28) et (2.29) assurent que, pour chaque départ d'un certain type, il existe une voie contenant un autobus de même type qui peut quitter le dépôt sans manœuvre. La contrainte de capacité (2.30) assure que la taille totale des arrivées affectées à une voie ne dépasse pas la capacité de la voie.

Les variables du modèle suivantes :

$$x_{ik} = \begin{cases} 1 & \text{si l'arrivée } i \text{ est positionnée dans la voie } k, \\ 0 & \text{sinon} \end{cases}$$

$$y_{jk} = \begin{cases} 1 & \text{si le départ } j \text{ est servi par une arrivée positionnée dans la voie } k, \\ 0 & \text{sinon.} \end{cases}$$

L'affectation des arrivées à des voies est donnée par :

$$\sum_{k \in C} x_{ik} = 1 \qquad \forall i \in V \tag{2.31}$$

$$\sum_{i \in V} l'_i x_{ik} \leq L_k \qquad \forall k \in C. \tag{2.32}$$

L'affectation des départs à des voies est donnée par :

$$\sum_{k \in C} y_{jk} = 1 \qquad \forall j \in D \tag{2.33}$$

$$\sum_{j \in D} l''_j y_{jk} \leq L_k \qquad \forall k \in C. \tag{2.34}$$

Les variables x_{ij} et y_{jk} sont indépendantes. Pour les relier, les auteurs ont introduit les variables z_{ijk} définies par :

$$z_{ijk} = \begin{cases} 1 & \text{si le départ } j \text{ est servi par l'arrivée } i \text{ positionnée dans la voie } k, \\ 0 & \text{sinon.} \end{cases}$$

Ces variables ne sont définies que pour les paires (i, j) de types compatibles, i.e., l'arrivée i est de même type que le départ j. Notons par A l'ensemble de toutes les paires compatibles. Les variables z_{ijk} satisfont les contraintes suivantes :

$$\sum_{j \in D(i)} z_{ijk} \leq x_{ik} \qquad \forall i \in A,\ k \in C \tag{2.35}$$

$$\sum_{i \in V(j)} z_{ijk} \leq y_{jk} \qquad \forall j \in D,\ k \in C \tag{2.36}$$

$$z_{ijk} + z_{uvk} \leq 1 \qquad \forall (i,j) \in A, (u,v) \in A, i > u, j < v, k \in C, \tag{2.37}$$

où $V(j)$ est l'ensemble des autobus compatibles avec le départ j

$$V(j) = \{i \in V : (i,j) \in A\},$$

et $D(i)$ est l'ensemble des départs compatibles avec l'arrivée i

$$D(i) = \{j \in D : (i,j) \in A\}.$$

Les contraintes (1.37) empêchent les manoeuvres. Le modèle à deux niveaux est donné par :

$$\begin{array}{rll} BD \text{ max} & \sum_{k \in C} \sum_{(i,j) \in A} z_{ijk} & \\ \text{s.à} & (1.31) - (1.37) & \\ & x_{ik},\ y_{jk},\ z_{ijk} \in \{0,1\} & \text{pour tous les indices valides.} \end{array}$$

Les contraintes du modèle BD sont formées de deux blocs : un bloc correspond aux contraintes d'affectation généralisée (2.31)-(2.34) et un autre bloc correspond à des contraintes d'élimination de manœuvres (2.35)-(2.37).

2.2.1 Approche de résolution

Le modèle BD peut être relaxé en utilisant l'approche de décomposition lagrangienne de la manière suivante :

1. Remplacer les variables x_{ik} et y_{jk} dans (2.35) et (2.36) par des nouvelles variables $x'_{i,k}$ et $y'_{j,k}$;
2. Introduire des nouvelles contraintes $x'_{ik} = x_{ik}\ \ \forall(i,k)$, et $y'_{jk} = y_{jk}\ \ \forall(j,k)$;
3. Relaxer les contraintes introduites avec des multiplicateurs μ et λ.

Le problème relaxé est :

$$\Phi(\mu,\lambda) = \max_{z,x,x',y,y'} \{\sum_{k,i,j} z_{ijk} + \sum_{i,k} \lambda_{ik}(x'_{ik} - x_{ik}) + \sum_{j,k} \mu_{j,k}(y'_{jk} - y_{jk})\} \quad (2.38)$$

s.à

$$\sum_{k\in C} x_{ik} = 1, \qquad \forall i \in V, \tag{2.39}$$

$$\sum_{i\in V} l'_i x_{ik} \leq L_k, \qquad \forall k \in C, \tag{2.40}$$

$$\sum_{k\in C} y_{jk} = 1, \qquad \forall j \in D, \tag{2.41}$$

$$\sum_{j\in \mathcal{D}} l''_j y_{jk} \leq L_k \qquad \forall k \in C, \tag{2.42}$$

$$\sum_{j\in \mathcal{D}(i)} z_{ijk} \leq x'_{ik} \qquad \forall i \in V,\ k \in C, \tag{2.43}$$

$$\sum_{i\in V(j)} z_{ijk} \leq y'_{jk} \qquad \forall j \in D,\ k \in C, \tag{2.44}$$

$$z_{ijk} + z_{uvk} \leq 1 \qquad \forall (i,j) \in A,\ i > u, j < v,\ (u,v) \in A,\ k \in C, \tag{2.45}$$

$$x_{ik},\ y_{jk},\ z_{ijk} \in \{0,1\} \qquad \text{pour tous les indices valides.} \tag{2.46}$$

Le dual lagrangien correspondant au modèle BD est

$$\text{(DL)} \ \min_{\mu,\lambda}\{\Phi(\mu,\lambda)\}.$$

Pour évaluer $\Phi(\mu,\lambda)$, on doit résoudre $2m$ problèmes d'affectation généralisée et m problèmes d'affectation sans croisement. Une fois que les variables x_{ik} et y_{jk} sont fixées, on obtient un problème nommé Design Noncrossing Matching (DNCM). Le problème d'affectation généralisée est un problème NP-difficile. Pour le résoudre, on utilise un code de programmation linéaire mixte comme CPLEX. Le problème DNCM est un problème facile à résoudre. Pour chaque voie k, on a le problème DNCM suivant (il est à noter que l'indice k a été laissé tombé afin de simplifier la notation) :

$$(DNCM)\ \max \quad \sum_{i,j\in \mathcal{A}} z_{ij} + \sum_{i\in V} \lambda_i x'_i + \sum_{j\in \mathcal{D}} \mu_j y'_j \tag{2.47}$$

s.à

$$\sum_{j\in\mathcal{D}(i)} z_{ij} \leq x'_i \qquad \forall i \in V, \tag{2.48}$$

$$\sum_{i\in V(j)} z_{ij} \leq y'_j \qquad \forall j \in D, \tag{2.49}$$

$$z_{ij} + z_{uv} \leq 1 \qquad \forall (i,j) \in A, (u,v) \in A,\ i > u, j < v, \tag{2.50}$$

$$x'_i,\ y'_j,\ z_{i,j} \in \{0,1\} \qquad \text{pour tous les indices valides.} \tag{2.51}$$

On peut démonter que DNCM est équivalent au problème d'affectation sans croisement noté WNCM :

$$(WNCM)\ \ \max \quad \sum_{(i,j)\in A} (1+\lambda_i+\mu_j) z_{i,j} \tag{2.52}$$

$$\text{s.à}$$

$$\sum_{j\in\mathcal{D}(i)} z_{i,j} \leq 1 \qquad \forall i \in V, \tag{2.53}$$

$$\sum_{i\in V(j)} z_{i,j} \leq 1 \qquad \forall j \in D, \tag{2.54}$$

$$z_{i,j} + z_{u,v} \leq 1 \qquad \forall (i,j) \in A, (u,v) \in A, i > u, j < v, \tag{2.55}$$

$$z_{i,j} \in \{0,1\} \qquad \text{pour tous les indices valides.} \tag{2.56}$$

L'équivalence est dans le sens suivant : si $\overline{z}$ est une solution optimale pour WNCM, alors $(\overline{z}, \overline{x}, \overline{y})$ est optimale pour DNCM où

$$\overline{x}_i = \begin{cases} 0 & \text{si } \overline{z}_{i,j} = 0,\ \forall j \\ & \\ 1 & \text{sinon} \end{cases} \qquad i = 1, ..., n$$

et

$$\overline{y}_j = \begin{cases} 0 & \text{si } \overline{z}_{i,j} = 0,\ \forall i, \\ & \\ 1 & \text{sinon} \end{cases} \qquad j = 1, ..., n.$$

L'algorithme heuristique proposé par Gallo et Di Miele (2001) se décrit comme suit. **Dual lagrangien :** Résoudre DL le dual lagrangien du problème BD par un algorithme d'ascension qui maximise la fonction linéaire par morceaux $\Phi(\lambda, \mu)$ en exploitant par une mise-à-jour d'une itération à l'autre l'information fournie par l'ensemble des sous-gradients. Soit $(\overline{z}, \overline{x}, \overline{x}', \overline{y}, \overline{y}')$ la solution obtenue sans la condition $\overline{x} = \overline{x}'$ et $\overline{y} = \overline{y}'$. Dans la plupart des cas, la solution obtenue est non réalisable.

Faisabilité : Pour obtenir une solution réalisable, on fixe les variables x et y dans le modèle du problème BD aux valeurs de $\overline{x}$ et $\overline{y}$ et on résout le problème d'affectation sans croisement obtenu. En général, cette nouvelle solution laisse certains véhicules sans affectation.

Postoptimisation : Dans cette étape, on utilise une procédure heuristique pour affecter les véhicules et les départs qui ne sont pas affectés dans l'étape précédente.

Avec l'approche de résolution proposée, Gallo et Di Miele ont résolu des instances de 50 autobus, 9 voies et 4 types dans un temps inférieur à une heure.

2.3 Revue de littérature complémentaire

En plus des travaux présentés dans les sections précédentes, on trouve quelques travaux complémentaires. Blasum et al. (1999) ont étudié le problème d'affectation des trains à des sorties de la prochaine matinée tout en supposant que les trains sont déjà positionnés dans les voies. Ils ont démontré que ce problème est NP-complet.

Récemment, Freling et al. (2005) ont étudié un problème de stationnement des trains dans un contexte différent de celui de cette thèse. Dans leur problème, les trains peuvent être séparés en unités. Ils ont proposé une approche de résolution en deux étapes. Dans la première phase, les unités qui arrivent sont affectées à des unités de sortie en utilisant un programme linéaire binaire. Dans la deuxième phase, ils utilisent la génération de colonnes pour résoudre le problème d'affectation des groupes d'unités à des positions.

Dans le reste de ce chapitre, nous recensons les travaux sur le problème d'empilage des conteneurs dans les ports ou dans les bateaux qui est un problème NP-difficile.

Le problème d'empilage des conteneurs (CSP) a été étudié par Avriel et Penn en 1993 et 1996 et par Avriel, Penn, et Naomi en 1999. Dans ce problème, un vaisseau dédié au transport de conteneurs visite plusieurs ports. Dans chaque port, les conteneurs doivent être chargés et déchargés. Les conteneurs sont stockés par pile dans le vaisseau. Quand un conteneur doit être déchargé dans un port, tous les conteneurs situés dans la pile au-dessus de lui doivent aussi être déchargés. Et quand ces conteneurs sont destinés à d'autres ports, ils doivent être rechargés. Ceci entraîne des manipulations longues et coûteuses. L'objectif est de minimiser le nombre de manipulations.

L'ensemble des ports S est défini par {1,2,3,...,N} avec 1 représentant le premier port à visiter et N le dernier. Le nombre de conteneurs qui doivent être transportés du port i vers le port j est noté par $T_{i,j}$ et T dénote la matrice $(T_{i,j})$.

La méthode de résolution qui a été developpée par Avriel et Penn (1993, 1996) pour résoudre CSP n'est valable que pour les instances de petite taille. Pour les grands problèmes, des méthodes heuristiques doivent être développées pour déterminer une bonne solution. À cet effet, ils ont introduit une heuristique qui peut être décrite comme suit.

Dans une pile ne contenant que des conteneurs qui ont la même origine et la même destination, aucune manipulation n'est nécessaire. Des manipulations sont toutefois nécessaires si, pour certains ports i et j, $T_{i,j} \neq kR$ où k est un entier positif et R

le nombre de conteneurs par pile. L'idée de la méthode heuristique développée pour CSP est basée sur la décomposition de la matrice de transport $(T_{i,j})$ en une somme de deux sous-matrices. La première Φ a toutes des colonnes dont chaque élément est un multiple de R et l'autre Θ est $T - \Phi$. Les conteneurs de la matrice Φ seront placés dans des piles, qui ne seront jamais manipulées jusqu'à leur destination finale. Les conteneurs de la matrice Θ seront placés dans des piles où les manipulations sont nécessaires. Ce petit problème peut être résolu en utilisant un modèle de programmation linéaire.

Winter (1999) a aussi étudié dans sa thèse la relation entre le problème d'empilage des conteneurs et le problème de stationnement des trains. Il a montré qu'on peut résoudre TDP en utilisant le modèle CSP. Cependant, la résolution de ce modèle n'améliore pas le temps de résolution pour déterminer une solution optimale de TDP.

2.4 Contributions

La revue littérature que nous venons de présenter montre que le problème de stationnement a été peu étudié. En effet, quelques méthodes exactes et des heuristiques ont été développées pour résoudre des problèmes de petite et moyenne tailles (25 à 50 véhicules). De plus, les solutions obtenues en utilisant ces méthodes ne sont pas robustes : elles deviennent inutilisables si l'ordre planifié est perturbé.

La présente thèse fait avancer les connaissances sur le problème de stationnement des autobus en apportant les contributions suivantes.

1. Nous introduisons un nouveau mode de stationnement par bloc qui produit des solutions robustes.
2. Nous proposons des modèles pour résoudre des problèmes de grande taille (jusqu'à 144 autobus et 6 types).

3. Le cas avec affectations erronées est résolu par une méthode de Benders dans laquelle le sous-problème est sujet à des contraintes d'intégrité. Nous proposons une stratégie de branchement qui permet d'obtenir une solution optimale dans ce cas.

4. Pour faire face aux variations dans l'ordre d'arrivée, nous introduisons deux modèles qui permettent d'augmenter la robustesse des solutions. Ces modèles utilisent des contraintes ou des coûts définis en considérant les variations dans l'ordre d'arrivée résultant de certains scenarios extrêmes ou d'une simulation des arrivées stochastiques.

CHAPITRE 3 : ORGANISATION DU TRAVAIL

Cette thèse étudie le problème de stationnement des autobus dans un dépôt. Dans le chapitre précédent, nous avons présenté une revue de littérature sur les travaux effectués dans ce domaine. Elle démontre que les chercheurs ne se sont pas encore intéressés à définir des modèles qui permettent de produire des solutions robustes. Comme la suite d'arrivées planifiée n'est pas souvent respectée en pratique, ce type de solutions qui résistent à des changements dans l'ordre d'arrivée est fortement désirée. Pour obtenir de telles solutions, nous introduisons dans cette thèse des modèles d'optimisation reposant sur un mode de stationnement par bloc. Ce mode, qui est utilisé à la STM de façon manuelle par certains gareurs, consiste à partitionner les voies du dépôt en au plus deux blocs par voie, chaque bloc contenant un seul type d'autobus. Nous examinons le problème de stationnement par bloc dans deux contextes. Un contexte déterministe où l'heure et le type de chaque arrivée et chaque départ sont connus, et un contexte stochastique où l'heure d'arrivée de chaque autobus est connue avec certain niveau d'incertitude.

Les chapitres 4 et 5 traitent le problème de stationnement par bloc dans le contexte déterministe. Deux variantes sont étudiées. La première minimise le nombre de voies mixtes et fait l'objet de l'article présenté au chapitre 4. Pour cette variante, nous développons des modèles linéaires en nombres entiers qui peuvent être résolus par CPLEX. Afin d'éviter les manoeuvres, les compagnies de transport permettent certaines affectations erronées. Dans l'article du chapitre 5, nous examinons la variante du problème de stationnement par bloc qui minimise le nombre d'affectations erronées. Nous modélisons cette deuxième variante comme un programme linéaire en nombres entiers de grande taille qui ne peut être résolu directement par CPLEX. Nous proposons alors comme approche de résolution une méthode de décomposition de Benders avec un sous-problème en nombres entiers et nous développons une stratégie de branchement efficace pour prendre en compte ces contraintes d'intégrité.

La dernière partie de la thèse est dédiée à l'aspect stochastique du problème de stationnement par bloc. Au chapitre 6, nous introduisons dans un troisième article deux approches pour traiter les variations dans l'ordre planifié. Ces deux approches font appel à la programmation en nombres entiers et produisent des solutions plus robustes. Les modèles linéaires en nombres entiers développés pour ces deux approches peuvent être résolus par CPLEX.

CHAPITRE 4 : DISPATCHING BUSES IN A DEPOT USING BLOCK PATTERNS

Ce chapitre fait l'objet d'un article intitulé "Dispatching Buses in a Depot Using Block Patterns" qui est accepté pour publication dans *Transportation Science*. Dans la littérature, différents modèles ont été présentés pour le problème de stationnement, mais ils produisent des solutions qui manquent de robustesse. Comme les autobus arrivent rarement tels que planifiés (ils arrivent en avance ou en retard par rapport à l'heure prévue), des solutions robustes qui résistent aux changements dans l'ordre d'arrivée sont fortement désirées. Dans cet article, nous introduisons le mode de stationnement par bloc où chaque voie est partitionnée en au plus deux blocs. Les voies avec un seul bloc contiennent un seul type d'autobus et les voies avec deux blocs contiennent deux types d'autobus, les autobus de même type étant regroupés consécutivement dans la voie. Nous démontrons que ce problème de stationnement par bloc est NP-complet.

Dans ce chapitre, nous nous concentrons sur la variante du problème de stationnement des autobus par bloc qui minimise le nombre de voies mixtes (voies avec manoeuvres). Pour cette variante, nous proposons une formulation mathématique qui est basée sur l'énumération des patrons admissibles. Un patron admissible est un partitionnement de voie en au plus deux blocs. Nous examinons deux versions de la fonction objectif pour la formulation présentée. Dans la première, nous supposons que les voies mixtes contiennent deux types d'autobus et qu'une fois manoeuvrées, s'il y a lieu, ces voies n'ont pas nécessairement une configuration en deux blocs. Dans la deuxième, nous supposons que les voies mixtes contiennent aussi deux types d'autobus et peuvent être transformées en des voies avec deux blocs après les manoeuvres. Les résultats numériques montrent que des instances réelles provenant de données de la STM

(Société de Transport de Montréal) peuvent être résolues facilement par CPLEX. De plus, nous montrons aussi que les solutions obtenues par nos modèles sont plus robustes que celles qui peuvent être produites par des modèles qui ne tiennent pas compte de l'aspect robustesse, et que la complexité du problème augmente avec le nombre de types d'autobus et la taille des voies.

Dispatching buses in a depot using block patterns

Mohamed Hamdouni

École Polytechnique and GERAD, Montréal, Québec, Canada

Mohamed.Hamdouni@gerad.ca

Guy Desaulniers

École Polytechnique and GERAD, Montréal, Québec, Canada

Guy.Desaulniers@gerad.ca

Odile Marcotte

Université du Québec à Montréal and GERAD, Montréal, Québec, Canada

Odile.Marcotte@gerad.ca

François Soumis

École Polytechnique and GERAD, Montréal, Québec, Canada

Francois.Soumis@gerad.ca

Marianne van Putten

Eindhoven University of Technology, Eindhoven, The Netherlands

M.P.v.Putten@student.tue.nl

Abstract

In this article we consider the problem of assigning parking slots to buses of different types so that the required buses can be dispatched easily in the morning. More precisely, if a bus of a certain type is needed at a given time, the buses that precede it in the lane must have departed already. Thus care must be taken to ensure that the buses arriving in the evening are parked in an order compatible with the types required for the morning departures. Maneuvers (i.e., rearrangements of buses within lanes) might be necessary to achieve this goal. Since the transit authorities need robust solutions to this problem (known as the dispatching problem in the literature), we formulate a model in which the depot lanes are filled according to specific patterns, called one-block or two-block patterns. We present two versions of this model, study their properties and show that some real-life instances can be solved within reasonable times by a commercial MIP solver. We also demonstrate that the solutions of the model are very robust, and can thus be used by transit authorities.

4.1 Introduction

In this article we consider one of the problems facing urban transit authorities, namely the problem of dispatching the buses arriving at a depot (or garage) so that the right buses are available for the morning trips in the right order. Buses of different types may be assigned to a depot, because some trips require low-floor buses (used in districts with a high proportion of elderly or handicapped people), buses with extra power (used on steep inclines), advertising buses (used on lines specified in the advertising contracts), and so on. Standard buses are used for the other trips. As the buses of different types arrive at the depot each night, it becomes almost full and there is little room inside the depot for maneuvering the buses. Hence they are not necessarily positioned in such a way that the buses available for the first morning trips are of the correct type.

The depot layout varies a lot from depot to depot, but most layouts are subdivided into lanes of variable length. A bus enters a given lane in the evening and will leave the depot in the morning at the other endpoint of the same lane. Thus a lane can be considered as a *queue* or FIFO list (see Gallo and Di Miele 2001). For safety reasons, a bus is not allowed to back up in the depot. If the buses within a lane are parked in an order that is not compatible with the morning departures, one can maneuver them outside the depot during the night. Since the buses are not allowed to back up, a "maneuver" consists of letting all the buses in the lane leave the lane and reenter it in the right order. Note that a maneuver will be necessary even if a single bus is misplaced, and hence the number of buses to be repositioned is either 0 or the number of buses in the lane. This is not always the case in the dispatching problem for trams or trains (see for instance Winter and Zimmermann (2000) and the other references below). On the other hand, maneuvers involving two or more lanes at the same time are impractical because there is little space in the vicinity of the depot. Thus we shall not consider them in this article.

The problem of *dispatching buses in a depot* can be stated as follows. Given an evening arrival sequence and a morning departure sequence (including the bus type and the arrival or departure time), this problem consists of assigning slots to the incoming buses in such a way that buses of the correct type are available in the morning (without maneuvers). The basic dispatching problem is often not feasible, and one must minimize either the number of maneuvers or the number of "mismatches" (i.e., assignments of buses with the wrong type). In some depots, a lane forms a right angle with another lane and thus blocks its entry point. In that case, the second lane must be filled completely before the first one is used for storing buses. This constraint is called a *precedence constraint.* If there are any such constraints, the assignment of slots to buses must satisfy them.

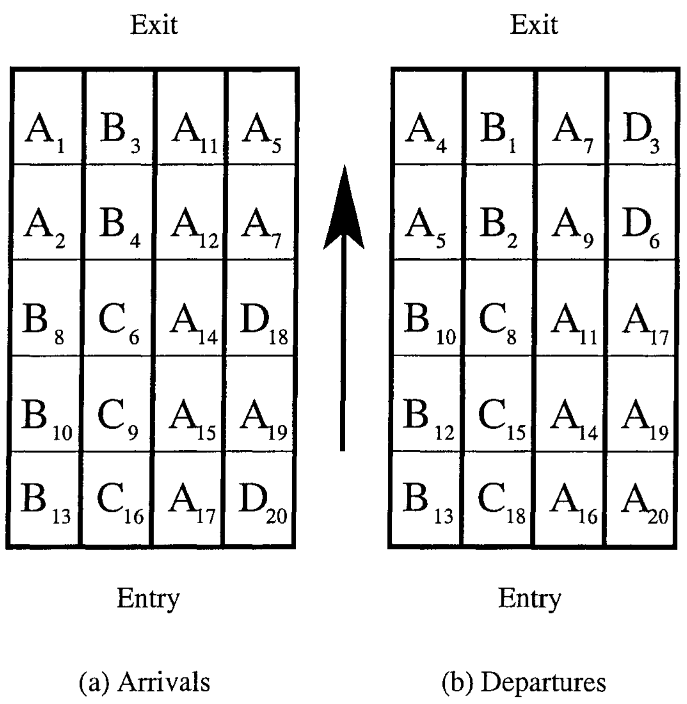


Figure 4.1 – An instance of bus depot layout

Figure 1 illustrates these definitions. It contains a snapshot of the depot layout just after the arrival of the last bus and another snapshot just before the first bus departs. Note that in this example, all the slots have the same size and all the lanes have the same length. The letters (A, B, C and D) represent bus types, and the number in the bottom right corner of a slot represents the position of the bus in the arrival sequence (resp. departure sequence). For instance, the sixth bus in the arrival sequence is of type C while the tenth bus in the departure sequence is of type B. In order for the correct bus types to be available in the morning, it is necessary to maneuver the fourth lane.

Blasum *et al.* (1999) and Winter and Zimmermann (2000) study the dispatching problem in the context of trams, where it is called the *shunting* problem. Blasum *et al.* (1999) show that the problem is NP-complete when all the buses have already arrived and must be assigned new slots because of the order of departures. Winter and Zimmermann (2000) prove that two versions of the dispatching problem are NP-hard : in the first (resp. second) version, the number of type mismatches (resp. shunting movements) is minimized. They formulate the first version as a binary linear program and the second as a quadratic integer program that can be linearized and solved by a commercial MIP solver. They also propose several heuristic approaches for the second version.

Freling *et al.* (2002) consider the shunting problem for trains, but it is different from the problem considered in the present article and in Winter and Zimmermann (2000) because trains can be broken up into "units" before shunting. Their solution approach consists of two steps. In the first, each incoming train unit is assigned to a leaving train unit by solving a binary linear program. In the second step, groups of train units are assigned to shunt tracks by using a column generation method. Gallo and Di Miele (2001) consider the dispatching problem for buses and their objective is to maximize the number of matches between incoming vehicles and the duties assigned to them in the morning. They model this problem as a binary linear program and use a heuristic Lagrangean decomposition approach to solve it.

In this article we adopt the point of view that any solution to the dispatching problem must be robust (in a sense to be defined shortly), and we try to minimize the number of lanes that must undergo maneuvers (instead of the number of buses that are wrongly positioned). Our contribution is threefold. First, we introduce a class of robust solutions for the bus dispatching problem. Second, we propose two integer linear programming formulations corresponding to slightly different types of robust solutions. These formulations are based on the enumeration of all admissible lane patterns (partitions). Finally, we report computational results showing that real-world instances can be solved using a commercial MIP solver. We describe the problem precisely in Section 4.2, prove that it is $\mathcal{NP}$-complete in Section 4.3, present the formulations in Section 4.4, give some theoretical results in Section 4.5 and discuss the computational results in Section 4.6.

4.2 Dispatching using block patterns

Although the arrival and departure times are known in principle, the buses do not necessarily arrive on schedule, and because there are many arrivals within a given period (sometimes two or three per minute), there is little time to assign a lane to an incoming bus. Therefore a solution to the dispatching problem that includes complex lane patterns will not be robust and is undesirable. In the good solutions, there should only be one or two bus types per lane, and if possible, the buses of a given type should form a block. This is so because the pattern of such a lane is less likely to change if a bus arrives later or sooner than the scheduled time, and because a typical morning departure does not require a specific bus but any bus of a given type.

To illustrate these points, let us consider the simple example in which four buses of type A and four buses of type B arrive at the depot in the evening. We assume that the depot has two lanes of length four and that the arrival and departure sequences are the sequence $ABABABAB$. For this example, parking the buses so that the pattern of each lane is $ABAB$ is "feasible", in the sense that incoming buses can be parked in the depot without maneuvers and buses of the right type may leave without maneuvers. This solution, however, is not robust, because any swap of two consecutive arrivals will make it infeasible. On the other hand, parking the buses so that the lane patterns are $AAAA$ and $BBBB$, respectively, is also feasible but much more robust. Indeed, the latter solution is feasible for any arrival sequence and any departure sequence.

To define what we mean by "robustness", we first introduce the notion of breakpoint. If the pattern of a lane is of the form $S_1S_2\ldots S_m$, where S_i is a sequence of arrivals of the same type and S_i and S_{i+1} are of different types for $i = 1, 2, \ldots, m-1$, we say that the lane has $m-1$ *breakpoints.* We claim that a pattern is robust if it has few breakpoints. To support this claim, we consider a depot of a single lane of length 12 and assume that there are 6 buses of type A and 6 buses of type B. We also assume that the arrival time of the i^{th} bus is i (for $i = 1, 2, \ldots, 12$) and that the buses leave the depot in the order in which they arrived. Clearly, in a lane of length 12, there are at most 11 breakpoints. So we selected 11 patterns, one for each number of breakpoints (for instance the pattern $AAABBBBBBAAA$ for 2 breakpoints and the pattern $AABBAABBAABB$ for 5 breakpoints). The chosen patterns have the property that all the blocks inside a pattern have roughly the same size, in so far as the number of blocks permits.

For each of the 11 patterns, we generated a pseudo-random number representing the true arrival time of the i^{th} bus; this variable followed the normal distribution of mean i and variance 1. In this way we obtained a "realistic" arrival sequence instead of the ideal one (i.e., the pattern). For instance, if the ideal sequence is $AABBAABBAABB$, the "real" sequence could be $ABABAABBABAB$, in which case two buses would be wrongly positioned. We repeated the generation of "real" sequences 10000 times for each pattern in order to compute the average number of wrongly positioned buses. We also computed the average number of times that a maneuver must be carried out (recall that a maneuver is necessary if and only if there is at least one bus to reposition). The results are summarized in the following table. NB (resp. $ANWB$, $ANTM$) represents the number of breakpoints (resp. the average number of wrongly positioned buses, the average number of times a maneuver is necessary).

Tableau 4.1 – Robustness characteristics of a one-lane dispatching problem

NB	$ANWB$	$ANTM$
1	0.6282	0.3122
2	1.2526	0.5268
3	1.8844	0.6717
4	2.4726	0.7743
5	3.0364	0.8576
6	3.4858	0.8857
7	3.673	0.9185
8	3.9318	0.9373
9	4.1576	0.95
10	4.426	0.9628
11	4.6256	0.9728

The results of Table 4.1 show that the number of wrongly positioned buses and the number of times a maneuver is necessary grow rapidly as the number of breakpoints increases. In particular, if the number of breakpoints is at least 2, the probability that a maneuver is necessary will be greater than 50%. Therefore if the depot has a single lane, one must minimize ν (the number of breakpoints) in order to ensure the robustness of the solution. If the depot has more than one lane, it is still desirable to find a solution in which each lane has at most one breakpoint, for two reasons. The first reason is again that the probability of a maneuver occurring will be small. The second reason is that a solution containing only lanes with no breakpoint or exactly one breakpoint is conceptually simpler ; indeed, the Société de Transport de Montréal (STM), whose data were used in our computational study, implements solutions of that kind.

Let us call *mixed lanes* those lanes that must undergo a maneuver in order for the solution to be feasible. Thus in Figure 1, the fourth lane is a mixed lane. A robust solution should contain as few mixed lanes as possible. Our main objective is thus to minimize the number of mixed lanes (or eliminate them altogether), and our secondary objective is to minimize the number of two-block lanes. We considered the possibility of eliminating mixed lanes altogether, but this policy may not be feasible for some problem instances. Note that the models found in the literature do not take the robustness into account, and that their solutions could contain many breakpoints and complicated patterns. We will come back to this point when we discuss the solutions of our models (see Subsection 4.6.3).

The problem thus defined will be called the problem of *dispatching buses in a depot by block patterns*, henceforth abbreviated as the *dispatching problem by blocks*. We shall first consider a simplified version of the problem, by restricting the feasible solutions to those in which a lane contains at most two blocks. This simplified version does not always have a feasible solution. In a more general version of the problem (not addressed in this article), we allow the solution to contain some mixed lanes of arbitrary patterns whose rearrangement will also lead to lanes of arbitrary patterns. For instance, a mixed lane could contain the four buses $ABCB$ (where the letters represent different types) and undergo a maneuver leading to the sequence $BABC$.

In the context of train or tram shunting, one could distinguish between the cost of rearranging one lane into a two-block lane and the cost of rearranging it into a more complex pattern (see Winter and Zimmermann (2000)), but in the context of bus dispatching, there is no difference between these two costs.

On the other hand, we could be more restrictive and allow only mixed lanes with two bus types. For instance, a mixed lane containing the buses $ABAB$ could be rearranged so as to contain $BAAB$. Finally, we could be even more restrictive and allow only mixed lanes with two bus types whose final pattern consists of two blocks. Thus if the lane $ABAB$ is rearranged, its final shape should be $AABB$ or $BBAA$. We will focus on the last two versions of the problem as they correspond to the most robust solutions. These will be called respectively the *version with arbitrary rearrangements* and the *version with two-block rearrangements.*

The following assumptions will also hold throughout the article.

- All the buses have the same length.
- All the lanes have the same length and are subdivided into parking slots, each of which can hold a bus.
- The number of buses is equal to the total number of parking slots.
- The depot is empty when the first bus arrives.
- Each lane can be accessed regardless of the status of the other lanes (i.e., there are no precedence constraints).

The first four assumptions were made to simplify the model. It is an easy matter to adapt our model when they do not hold (as we shall see later). On the other hand, removing the last assumption leads to more complex models, to be studied later. In the case where each mixed lane contains two bus types only, it is also easy to construct instances of the dispatching problem by blocks that require maneuvers or are infeasible (even if maneuvers are allowed). For instance, if there are two lanes of length three and the arrival and departure sequences are $BBBBCA$ and $ABBBBC$, respectively, one maneuver is required for parking ; indeed, the assignment of BBC to the first lane and ABB to the second (after the maneuver) will ensure that the departures can proceed as planned in the morning. On the other hand, if the arrival and departure sequences are $ABBCCA$ and $CCBAAB$, respectively, it is impossible to devise a parking solution in which each lane contains two blocks after rearrangement.

4.3 Complexity of the dispatching problem by blocks

We now state precisely the simplest version of the problem and prove that it is $\mathcal{NP}$-complete. Let v denote the number of lanes in the depot, ℓ the lane length and n the number of arrivals (resp. departures). Note that $n = v\ell$ by assumption. Also let $\mathcal{A}$, $\mathcal{D}$, $\mathcal{L}$ and $\mathcal{S}$ be the set of arrivals (with their types), the set of departures (with their types), the set of lanes and the set of slots within a lane, respectively. Finally let us assume that the elements of $\mathcal{A}$ (resp. $\mathcal{D}$) are indexed by $1, 2, 3, \ldots, n$ and that the i^{th} arrival (resp. departure) occurs before the j^{th} arrival (resp. departure) for $i < j$. The dispatching information will be represented by two bijective functions, a function f from $\mathcal{A}$ to $\mathcal{L} \times \mathcal{S}$ and a function g from $\mathcal{D}$ to $\mathcal{L} \times \mathcal{S}$. $f_1(i)$ (resp. $f_2(i)$) denotes the first (resp. second) component of $f(i)$. The functions g_1 and g_2 are defined similarly.

The problem Dispatching by blocks can now be stated as follows : given v, ℓ, a sequence of n arrivals (with their types) and a sequence of n departures (with their types), do there exist bijective functions f and g such that

- for each couple $(k, r) \in \mathcal{L} \times \mathcal{S}$, the type of the arrival $f^{-1}(k, r)$ and the type of the departure $g^{-1}(k, r)$ are identical,
- for each k, the arrivals assigned to lane k (i.e., the set $\{i \,|\, f(i) = (k, r) \text{ for some } r\}$) form at most two blocks,

- $i < j$ implies that $f_1(i) \neq f_1(j)$ or $f_2(i) < f_2(j)$ and
- $i < j$ implies that $g_1(i) \neq g_1(j)$ or $g_2(i) < g_2(j)$?

Clearly the dispatching problem by blocks belongs to $\mathcal{NP}$, and it suffices to show that some $\mathcal{NP}$-complete problem can be reduced to it. Note that the reduction described in Winter and Zimmermann (2000) cannot be used because it produces instances in which there are four types per lane. Actually, the reduction given below shows that the dispatching problem by blocks is quite different from the dispatching problem considered by Winter and Zimmermann (2000). Recall that PARTITION is the following problem : given n positive integers $c_1, c_2, \ldots, c_n$, does there exist a partition $\{I, J\}$ of the set $\{1, 2, 3, \ldots, n\}$ such that $\sum\limits_{i \in I} c_i = \sum\limits_{i \in J} c_i$? Without loss of generality, we will assume that $\{c_i\}_{i=1}^n$ is a nondecreasing sequence (i.e., $c_1 \leq c_2 \leq \cdots \leq c_n$) and let C denote $\sum\limits_{i=1}^{n} c_i$.

It is well known that PARTITION is $\mathcal{NP}$-complete (see Garey and Johnson (1979)). We first need to show that a specialized version of PARTITION is also $\mathcal{NP}$-complete. PARTITION-RESTRICTED is defined as follows : given n positive integers $c_1, c_2, \ldots, c_n$ satisfying $c_n \leq 2c_1 - 2$, does there exist a partition $\{I, J\}$ of the set $\{1, 2, 3, \ldots, n\}$ such that $\sum\limits_{i \in I} c_i = \sum\limits_{i \in J} c_i$?

Proposition 1. PARTITION-RESTRICTED *is $\mathcal{NP}$-complete.*

Proof. Let $\{c_i\}_{i=1}^n$ be an instance of PARTITION and K a number greater than or equal to $\max(c_n + 2, C + 1)$. We define a sequence of $2n$ numbers d_i as follows : $d_i = K$ for $1 \leq i \leq n$ and $d_i = c_i + K$ for $n + 1 \leq i \leq 2n$. Clearly $\{d_i\}_{i=1}^{2n}$ is a nondecreasing sequence such that $d_{2n} \leq 2d_1 - 2$, that is, an instance of PARTITION-RESTRICTED. Note that the sum of the d_i is equal to $2nK + C$. Let us assume that there exists a subset I of $\{1, 2, 3, \ldots, n\}$ such that $\sum\limits_{i \in I} c_i = \frac{C}{2}$ and let I' denotes the set $\{i+n \mid i \in I\}$. Then $\sum_{i=1}^{n-|I|} d_i + \sum_{i \in I'} d_i = nK + \frac{C}{2}$, which proves that the answer to the instance of PARTITION-RESTRICTED is "yes". Conversely, assume that there exists a subset I' of $\{1,2,3,\ldots,2n\}$ such that $\sum_{i \in I'} d_i = nK + \frac{C}{2}$. Then the cardinality of $|I'|$ must be equal to n because K is greater than C. Therefore if one defines I as $\{i-n \mid i \in I', i > n\}$, we conclude that $\sum_{i \in I} c_i = \frac{C}{2}$ and the answer to the instance of PARTITION is "yes". □

We now consider a version of the dispatching problem by blocks, called LANE-COVERING, in which the arrival and departure times do not play any role. LANE-COVERING can be stated as follows : given positive numbers v, ℓ and $b^1, b^2, b^3, \ldots, b^m$ such that $\sum_{t=1}^{m} b^t = v\ell$, is it possible to find nonnegative integers x_{tj} (for $t \in \{1,2,3,\ldots,m\}$ and $j \in \{1,2,3,\ldots,v\}$) such that $\sum_{t=1}^{m} x_{tj} = \ell$ for each $j \in \{1,2,3,\ldots,v\}$, at most two of the x_{tj} are nonzero for a given j and $\sum_{j=1}^{v} x_{tj} = b^t$ for each $t \in \{1,2,3,\ldots,m\}$? Clearly, LANE-COVERING consists of assigning buses to lanes so that each lane contains at most two blocks, and as such, is a simplified version of the dispatching problem by blocks.

Proposition 2. LANE-COVERING *is $\mathcal{NP}$-complete.*

Proof. Let $\{c_i\}_{i=1}^{n}$ be an instance of PARTITION-RESTRICTED and define an instance of LANE-COVERING as follows : $v = n$, $\ell = c_n + 1$, $m = n+2$, $b^t = \ell - c_t$ for $t \in \{1,2,3,\ldots,n\}$ and $b^{n+1} = b^{n+2} = \frac{C}{2}$. Note that for $t \in \{1,2,3,\ldots,n\}$, we have

$$b^t \leq c_n + 1 - c_1 \leq c_n + 1 - \frac{c_n + 2}{2} = \frac{c_n}{2} < \frac{\ell}{2},$$

where the second inequality follows from the definition of PARTITION-RESTRICTED. We conclude that in any lane covering, each lane contains one and only one block of type t for some $t \in \{1,2,3,\ldots,n\}$. We note that the space to be "filled" in the i^{th}

lane is exactly c_i and that it can only be filled by buses of type $n+1$ or $n+2$. Hence there is a partition $\{I, J\}$ of the set $\{1, 2, 3, \dots, n\}$ verifying $\sum_{i \in I} c_i = \sum_{i \in J} c_i$ if and only if there exists a lane covering in which at most two blocks appear in each lane. We have shown that the answer to PARTITION-RESTRICTED is "yes" if and only if the answer to LANE-COVERING is "yes". □

Proposition 3. DISPATCHING BY BLOCKS *is $\mathcal{NP}$-complete.*

Proof. Let again $\{c_i\}_{i=1}^{n}$ be an instance of PARTITION-RESTRICTED and define the b^t as in the preceding proof. Assume that the buses arrive at the depot in the order indicated by the types, i.e., all the buses of type t arrive before any bus of type $t+1$ (for $t \in \{1, 2, 3, \dots, m-1\}$). Assume also that the departure order is the same as the arrival order. Then each lane will contain one and only one block of type $t \leq n$ in any solution of the dispatching problem by blocks. This implies that the answer to the instance of PARTITION-RESTRICTED is "yes" if and only if the answer to the instance of DISPATCHING BY BLOCKS is "yes". □

4.4 Mathematical programming formulations

We now present formulations of the problem based on the enumeration of the admissible lane patterns. An *admissible lane pattern* is a partition of a lane into at most two blocks of fixed types, an *exit block* of type u (say) and an *entry block* of a type $t \neq u$. If the pattern consists of a single block, it will be deemed the exit block. The buses in the exit block will leave the depot before the buses in the entry block. Thus the exit block is filled (resp. emptied) before the entry block at arrival time (resp. departure time).

4.4.1 The version with no mixed lane

We now formulate as an integer program the version of the dispatching problem by blocks in which no mixed lane is allowed. There is no objective function in this version since our only concern is the existence of feasible solutions. We first introduce the notation (recall that v denotes the number of lanes and ℓ the lane length).

T : The set of bus types.

b^t : The number of buses of type $t \in T$.

n : The total number of buses. Note that by assumption, $n = \sum\limits_{t \in T} b^t = v\ell$.

P^{ut} : The set of admissible patterns with an exit block of type u and a nonempty entry block of type t.

P_2 : The set of admissible patterns with two nonempty blocks.

P : The set of admissible patterns ($P_2 \subset P$).

s_p^t : The number of buses of type $t \in T$ in the exit block of pattern $p \in P$.

e_p^t : The number of buses of type $t \in T$ in the entry block of pattern $p \in P$.

I : The set $\{1, 2, \ldots, n\}$ (used to label the arrivals).

I^t : The subset $\{i \in I \mid \text{the arrival } i \text{ is of type } t\}$ for $t \in T$.

ha_i : The time of the arrival $i \in I$.

a_i^t : The number of type t buses whose arrival time is at most ha_i (for $t \in T$ and $i \in I$).

J : The set $\{1, 2, \ldots, n\}$ (used to label the departures).

J^t : The subset $\{j \in J \mid \text{the departure } j \text{ is of type } t\}$ for $t \in T$.

hd_j : The time of the departure $j \in J$.

d_j^t : The number of type t buses whose departure time is at most hd_j (for $t \in T$ and $j \in J$).

X_p : The number of lanes partitioned according to the pattern $p \in P$.

Y_{pi} : The number of lanes partitioned according to the pattern $p \in P_2$ whose exit block is full at the time ha_i for $i \in I$ (before the i^{th} arrival).

Z_{pj} : The number of lanes partitioned according to the pattern $p \in P_2$ whose exit block is empty at the time hd_j for $j \in J$ (before the j^{th} departure).

Note that for two fixed types u and t, the cardinality of P^{ut} is equal to $\ell - 1$. Since P_2 is the disjoint union of the P^{ut} and P is the disjoint union of P_2 and the set of one-block patterns, their cardinalities are given by the formulas

$$|P_2| = |T|\,(|T|-1)\,(\ell-1),$$
$$|P| = |T|\,(|T|-1)\,(\ell-1) + |T|.$$

The constraints of the integer program are the following.

$$\sum_{p \in P} X_p = v \tag{4.1}$$

$$\sum_{p \in P} (s_p^t + e_p^t) X_p = b^t, \qquad \forall\, t \in T \tag{4.2}$$

$$\sum_{p \in P} s_p^t X_p + \sum_{p \in P_2} e_p^t Y_{pi} \geq a_i^t, \qquad \forall\, t \in T, i \in I^t \tag{4.3}$$

$$\sum_{p \in P_2} s_p^u Y_{pi} \leq a_i^u, \qquad \forall\, u \in T, i \in I \setminus I^u \tag{4.4}$$

$$Y_{pi} \leq Y_{p,i+1}, \qquad \forall\, p \in P_2, i \in I \setminus \{n\} \tag{4.5}$$

$$Y_{pn} \leq X_p, \qquad \forall\, p \in P_2 \tag{4.6}$$

$$\sum_{p \in P} s_p^t X_p + \sum_{p \in P_2} e_p^t Z_{pj} \geq d_j^t, \qquad \forall\, t \in T, j \in J^t \tag{4.7}$$

$$\sum_{p \in P_2} s_p^u Z_{pj} \leq d_j^u, \qquad \forall\, u \in T, j \in J \setminus J^u \tag{4.8}$$

$$Z_{pj} \leq Z_{p,j+1}, \qquad \forall\, p \in P_2, j \in J \setminus \{n\} \tag{4.9}$$

$$Z_{pn} \leq X_p, \qquad \forall\, p \in P_2 \tag{4.10}$$

$$X_p, Y_{pi}, Z_{pj} \text{ are nonnegative integers, } \qquad \text{for all valid indices} \qquad (4.11)$$

The first constraint (4.1) means that each lane must be partitioned in a certain way. The constraints (4.2) mean that the numbers of type t parking slots and type t buses must be equal. The constraints (4.3) mean that there must be sufficiently many type t slots that can be accessed by type t buses whose arrival time is at most ha_i (for $i \in I^t$). The accessible slots belong to the exit blocks of type t and the entry blocks of type t whose matching exit blocks have been filled before the arrival i. The constraints (4.4) mean that at the time ha_i (corresponding to an arrival of type $t \neq u$), the number of slots in the type u exit blocks that have already been filled is at most equal to the number of type u arrivals.

The constraints (4.5) mean that if an entry block becomes accessible before the arrival $i \in I \backslash \{n\}$, it remains accessible until the last arrival. The constraints (4.6) mean that the number of accessible entry blocks for the pattern $p \in P_2$ after the last arrival is at most the number of lanes partitioned according to p. The constraints (4.7)–(4.10) are the mirror image of the constraints (4.3)–(4.6), where "arrival" is replaced by "departure" and "full" by "empty". Finally, the program contains nonnegativity and integrality constraints. It is clear that any feasible dispatching of the buses will satisfy the above constraints. It is somewhat more difficult to show that a solution of this system corresponds to a feasible dispatching. We postpone this proof until Section 4.5.

Note also that the constraints (4.5), (4.6) and (4.3) for $i = \max_{k \in I^t} k$ and $t \in T$ imply that

$$\sum_{p \in P} (s_p^t + e_p^t) X_p \geq b^t, \ \forall \, t \in T.$$

Adding all these inequalities and taking the constraints (4.1) into account, we conclude that they must be satisfied at equality in any feasible solution. Thus the constraints

(4.2) are redundant. We have included them into the program, however, because they accelerate its resolution (see Section 4.6.2). A similar remark applies to the other versions presented in this section.

We can relax some of the assumptions made previously. If the lane length is not a constant, we can subdivide the set of lanes into subgroups of uniform length and replace the constraint (4.1) by several constraints (one for each subgroup). Now it is possible to enumerate the admissible patterns for each group of lanes, whether all the buses have the same length or not. Thus the assumption of a uniform bus length can be relaxed as well. We can also relax the assumption that the sum of the bus lengths is equal to the sum of the lane lengths and assume instead that the first sum is at most equal to the second one. This follows because the empty slots can be filled with "dummy" buses of unit length (in the case where the two sums are not equal). Finally, we can relax the assumption that the depot is empty when the first bus arrives by taking into account the allowable types and the number of available positions for each of the partially filled lanes. For instance, if a two-block lane of length 5 contains two type A buses, the pattern for the rest of the lane must be of length 3 and contain either one block or two blocks (one of which is a type A exit block).

4.4.2 The version with arbitrary rearrangements

In this version of the problem we assume that the mixed lanes contain two bus types and that their patterns (once they have been rearranged) are arbitrary. For a mixed lane, a pattern will be defined by the number of buses of each type in the pattern; the slots occupied by buses in the lane don't matter in this case. For instance, if a lane has five slots, the sequences $ABBAB$ and $BABAB$ are deemed to have the same pattern. We have to introduce the notation below to formulate the new version.

Q_2 : The set of all patterns for mixed lanes with two bus types.

b_q^t : The number of type t buses in the pattern $q \in Q_2$ (for $t \in T$).

X_q : The number of lanes partitioned according to the pattern $q \in Q_2$.

There are $\ell - 1$ possible patterns for mixed lanes containing buses of types t and u (for a given pair $\{t, u\}$). Therefore the cardinality of Q_2 is given by the formula $|T|(|T|-1)(\ell-1)/2$. The formulation for the version with arbitrary rearrangements is the following, where M denotes a large number.

$$
\begin{array}{llll}
\text{Minimize} & M\sum_{q\in Q_2} X_q + \sum_{p\in P_2} X_p & & (4.12)\\
\text{s.t.} & \sum_{p\in P} X_p + \sum_{q\in Q_2} X_q = v & & (4.13)\\
 & \sum_{p\in P}(s_p^t + e_p^t)X_p + \sum_{q\in Q_2} b_q^t X_q = b^t, & \forall\, t\in T & (4.14)\\
 & \sum_{p\in P} s_p^t X_p + \sum_{p\in P_2} e_p^t Y_{pi} + \sum_{q\in Q_2} b_q^t X_q \ge a_i^t, & \forall\, t\in T, i\in I^t & (4.15)\\
 & \sum_{p\in P_2} s_p^u Y_{pi} \le a_i^u, & \forall\, u\in T, i\in I\setminus I^u & (4.16)\\
 & Y_{pi} \le Y_{p,i+1}, & \forall\, p\in P_2, i\in I\setminus\{n\} & (4.17)\\
 & Y_{pn} \le X_p, & \forall\, p\in P_2 & (4.18)\\
 & \sum_{p\in P} s_p^t X_p + \sum_{p\in P_2} e_p^t Z_{pj} + \sum_{q\in Q_2} b_q^t X_q \ge d_j^t, & \forall\, t\in T, j\in J^t & (4.19)\\
 & \sum_{p\in P_2} s_p^u Z_{pj} \le d_j^u, & \forall\, u\in T, j\in J\setminus J^u & (4.20)\\
 & Z_{pj} \le Z_{p,j+1}, & \forall\, p\in P_2, j\in J\setminus\{n\} & (4.21)\\
 & Z_{pn} \le X_p, & \forall\, p\in P_2 & (4.22)\\
 & X_p, X_q, Y_{pi}, Z_{pj} \text{ are nonnegative integers,} & \text{for all valid indices} & (4.23)
\end{array}
$$

We chose the objective function (4.12) because our main goal is to minimize the number of mixed lanes and our secondary goal to minimize the number of two-block lanes. As noted in Section 4.2, this objective will enhance the robustness of the solution. The constraints (4.13)-(4.23) are similar to the constraints (4.1)-(4.11), with the difference that incoming buses may be assigned to mixed lanes.

4.4.3 The version with two-block rearrangements

In the second version of the problem with mixed lanes, the requirements are as follows : there are two bus types in any mixed lane, and any mixed lane can be transformed into a two-block lane in such a way that the overall solution is compatible with the departure sequence. It is thus necessary to relate the mixed lane patterns to the two-block patterns obtained by rearranging them. In order to do this we need the following notation.

P_2^q : The set of two-block patterns that can be obtained by rearranging the pattern $q \in Q_2$ (note that $|P_2^q| = 2$ and $P_2 = \bigcup_{q \in Q_2} P_2^q$).

W_p : The number of mixed lanes partitioned according to the pattern $p \in P_2$ after rearrangement.

The formulation for the version with two-block rearrangements is obtained from the version (4.12)–(4.23) by adding the constraints

$$\sum_{p \in P_2^q} W_p = X_q, \qquad \forall\, q \in Q_2, \tag{4.24}$$

$$W_p \geq 0, \text{ integers}, \qquad \forall\, p \in P_2, \tag{4.25}$$

expressing the fact that all mixed lanes must be rearranged, and then by replacing the constraints (4.19) and (4.22) with the constraints

$$\sum_{p \in P} s_p^t X_p + \sum_{p \in P_2} s_p^t W_p + \sum_{p \in P_2} e_p^t Z_{pj} \geq d_j^t, \qquad \forall\, t \in T, j \in J^t, \tag{4.26}$$

$$Z_{pn} \leq X_p + W_p, \qquad \forall\, p \in P_2. \tag{4.27}$$

The latter constraints take into account the lanes that have been rearranged.

4.5 Model validity and improvements

4.5.1 Model validity

We have already observed that the constraints of any of the versions were satisfied by all solutions of the dispatching problem by blocks. We now prove that all feasible solutions of the third version actually represent solutions of this problem, i.e., assignments of incoming buses to slots with the prescribed number of maneuvers. The proofs for the first two versions are similar to the proof given below.

Theorem 4.5.1. *Let $\{X_p\} \cup \{Y_{pi}\} \cup \{X_q\} \cup \{W_p\} \cup \{Z_{pj}\}$ be any feasible solution of the version with two-block rearrangements. There is an algorithm for parking and dispatching the buses that is compatible with this solution, i.e., an algorithm such that*

1. *there is a parking slot for each bus,*
2. *a bus is parked in the entry block of a lane only if the exit block of that lane is already full, and*
3. *a bus leaves from the entry block of a lane only if the exit block of that lane is empty.*

Proof. The algorithm consists of two parts : a parking algorithm and a dispatching algorithm. The parking algorithm first tries to park an incoming bus in an exit block. Among the exit blocks, the priority is determined by the values of the Y_{pi}. This priority depends upon the bus type t. For a fixed t, let P^t denote the set of patterns whose exit block is of type t and entry block is not empty (i.e., $p \in P^t$ if and only if $0 < s_p^t < \ell$). Let also m_i^t denote $\sum\limits_{p \in P^t} Y_{pi}$. The sequence m_i^t is clearly a nondecreasing function of i. Finally let m_0^t be equal to 0 and $i_1, i_2, \ldots, i_k$ denote the increasing sequence of all indices such that $m_{i_1}^t > m_{i_1-1}^t, m_{i_2}^t > m_{i_2-1}^t, \cdots, m_{i_k}^t > m_{i_k-1}^t$.

We construct a *list of lanes*, denoted by L_t, as follows. Initially L_t contains $Y_{p,i_1} - Y_{p,i_1-1}$ lanes of pattern p, for each $p \in P^t$ such that $Y_{p,i_1} > Y_{p,i_1-1}$. We shall say that the *closing time* of these lanes is ha_{i_1} because it will not be possible to park buses in their exit blocks at the time ha_{i_1} or later. Then $Y_{p,i_2} - Y_{p,i_2-1}$ lanes of pattern p, for each $p \in P^t$, will be placed at the end of L_t. The closing time of these lanes is equal to ha_{i_2}. We repeat this procedure until $Y_{p,i_k} - Y_{p,i_k-1}$ lanes of pattern p and closing time ha_{i_k} are placed at the end of L^t, for each $p \in P^t$.

The *parking algorithm* for assigning a slot to an incoming bus of type t arriving at the time ha_i can now be described as follows. In this paragraph we say that a mixed lane is *full* if it contains the maximum number of type t buses for the pattern of that lane.

- Scan the list of lanes L_t to find the first lane whose exit block is not full. Park the bus in the exit block of that lane if such a lane exists.
- If such a lane does not exist, park the bus into a one-block lane of type t (if it exists).
- If all the exit blocks of lanes in L_t and all the one-block lanes of type t are full, park the bus into a mixed lane that is not full.
- If all the exit blocks of lanes in L_t, all the one-block lanes of type t and all the mixed lanes are full, park the bus into an entry block of type t.

Note that the respective priorities of one-block lanes and mixed lanes are not crucial. The important feature of the algorithm is that it gives the highest priority to the first lane in the list L_t whose exit block is not full. To show that the parking algorithm is correct, we must show that

- there is at least one parking slot for the bus i of type t and
- the bus i is parked in the entry block of a lane only if the exit block of that lane is already full.

Consider the arrival of the bus i, of type t, at the time ha_i. Let us assume that it cannot be parked in an exit block, a one-block lane or a mixed lane. The constraints

(4.15) imply that there is at least one parking slot for the bus i in a lane of pattern p whose closing time is at most ha_i. The pattern p is such that $e_p^t > 0$ and $s_p^u > 0$ for some $u \in T \backslash \{t\}$. Now the constraint (4.16) for the type u means that the number of arrivals of type u up to the time ha_i (that is, a_i^u) is at least the number of available slots in the exit blocks of lanes whose pattern is in P^u and closing time is at most ha_i. Let B_i^u denote the left-hand side of (4.16), i.e., $\sum\limits_{p \in P_2} s_p^u Y_{pi}$. The parking algorithm ensures that the first B_i^u buses of type u, which arrive before the time ha_i, are parked in the exit blocks of those lanes. Therefore when the bus i is parked in a lane of pattern p, the exit block of that lane is already full.

The *dispatching algorithm* is the mirror image of the parking algorithm (except for the mixed lanes, which have been rearranged). It first tries to dispatch a leaving bus parked in an exit block. Among the exit blocks, the priority is determined by the values of the Z_{pj}. For each type t, a list L'_t similar to L_t is constructed. When a bus of type t must leave to perform a duty, it is chosen as follows. Note that "closing time" now refers to the time at which the exit block of a lane becomes empty.

- Scan the list of lanes L'_t to find the first lane whose exit block is not empty. Dispatch the first bus in that lane if such a lane exists.
- If such a lane does not exist, dispatch a bus parked in a one-block lane of type t (if it exists).
- If all the exit blocks of lanes in L'_t and all the one-block lanes of type t are empty, dispatch a bus parked in an entry block of type t.

As before, the constraints (4.26) and (4.20) ensure that the dispatching algorithm is correct, i.e., at least one bus of the correct type will be available to perform a given duty. $\square$

Note that the interpretation of the Y_{pi} used in this proof is slightly different from that we gave when we introduced them. Y_{pi} is actually the number of lanes *made available* by the model at the time ha_i, in the sense that the entry blocks of those lanes may be used to park incoming buses. A similar remark applies to the Z_{pj}.

4.5.2 Reductions in the numbers of variables and constraints

In this section we show that we can assume, without loss of generality, that many of the variables Y_{pi} are equal. As a result, only some of these variables need be retained and several constraints may be removed from the model. Let us assume that there are several consecutive arrivals of the same type, as in the sequence $AABBBCBBCAAA$. We may consider each subsequence consisting of arrivals of the same type as a "group arrival". There are thus six group arrivals in our example. We will show that the number of arrivals (resp. departures) need be compared to the number of available slots only at the end of a subsequence. Define i_1 as the index of the last arrival in a subsequence of type u and i_3 as the index of the last arrival in the following subsequence of type u. Define also i_2 as the index of the last arrival of a type $t \neq u$ occurring before time ha_{i_3}. For instance, in the above example, there are two subsequences of type B and thus i_1 is equal to 5, i_2 to 6 and i_3 to 8.

We claim that for any feasible solution of the model (described by the X_p, the W_p, the Y_{pi} and the Z_{pj}), there exists a feasible solution (described by the X'_p, the W'_p, the Y'_{pi} and the Z'_{pj}) such that

- $X'_p = X_p$ and $W'_p = W_p$ for all p,
- $Z'_{pj} = Z_{pj}$ for all p and j,
- $Y'_{pi} = Y_{p,i_2}$ for all $i \in \{i_1, i_1+1, \dots, i_3-1\}$ and any p whose exit block is of type u, and
- $Y'_{pi} = Y_{pi}$ for every other couple (p, i).

The constraints (4.17) and (4.18) are clearly satisfied by the new solution.

The constraints (4.16) are satisfied by the new solution for any i such that $i_2 + 1 \leq i \leq i_3 - 1$, because they are satisfied by the original solution and $Y'_{pi} \leq Y_{pi}$ for any

p and any i comprised between $i_2 + 1$ and $i_3 - 1$. The constraints (4.16) are also satisfied by the new solution for any i such that $i_1 \leq i \leq i_2$, because a_i^u is equal to $a_{i_2}^u$ in that case and we have

$$\sum_{p \in P_2} s_p^u Y'_{pi} = \sum_{p \in P_2} s_p^u Y_{p,i_2} \leq a_{i_2}^u = a_i^u.$$

The constraints (4.15) are satisfied by the new solution for any i such that $i_1 \leq i \leq i_2$, because they are satisfied by the original solution and $Y'_{pi} \geq Y_{pi}$ for any p and any i comprised between i_1 and i_2. Finally, we note that for any i comprised between $i_2 + 1$ and $i_3 - 1$, i belongs to I^u; thus the constraint (4.15) for such an i does not include any Y'_{pi} for which the exit block is of type u. Hence any Y'_{pi} appearing in this constraint will be equal to Y_{pi}, and the constraint (4.15) will be satisfied because it was satisfied by the original solution.

The above argument proves that we may add to any of our models, without loss of generality, the constraints $Y_{pi} = Y_{p,i_2}$ for i comprised between i_1 and $i_3 - 1$ (note that the coefficients of the Y_{pi} in the objective function are equal to 0). Of course, the argument can be repeated for any type u and any triple of indices (i_1, i_2, i_3) defined as above for the type u. We now argue that some constraints (4.16) may be removed from the model. The constraint $\sum_{p \in P_2} s_p^u Y_{p,i_2} \leq a_{i_2}^u$ implies the constraint $\sum_{p \in P_2} s_p^u Y_{pi} \leq a_i^u$ for any i comprised between i_1 and $i_2 - 1$ (because $Y_{pi} = Y_{p,i_2}$ and $a_i^u = a_{i_2}^u$ in that case) and for any i comprised between $i_2 + 1$ and $i_3 - 1$ (because $Y_{pi} = Y_{p,i_2}$ and $a_i^u \geq a_{i_2}^u$). Hence we will discard all the constraints (4.16) for i comprised between i_1 and $i_3 - 1$, except the constraint $\sum_{p \in P_2} s_p^u Y_{p,i_2} \leq a_{i_2}^u$. Note that our reasoning remains valid even in boundary cases (when $i_1 = 0$ or $i_3 = n + 1$).

Naturally, similar observations can be made for the Z_{pj} and the constraints (4.20). They lead us to define $\underline{I}$, $\underline{J}$ and the $\underline{I}^t$ and $\underline{J}^t$ (for $t \in T$) as follows.

$\underline{I}^t$: The set of indices $i \in I$ such that the arrival i is of the type $t \in T$ and either i is the last arrival or the arrival $i+1$ is of a type $u \in T \setminus \{t\}$ (that is, i is the end of a subsequence).

$\underline{I}$: The set $\bigcup_{t \in T} \underline{I}^t$.

$\underline{J}^t$: The set of indices $j \in J$ such that the departure j is of the type $t \in T$ and either j is the last departure or the departure $j+1$ is of a type $u \in T \setminus \{t\}$.

$\underline{J}$: The set $\bigcup_{t \in T} \underline{J}^t$.

We have shown that we can reduce the number of constraints by keeping only the constraints (4.16) for which i belongs to $\underline{I}^t$ and the constraints (4.20) for which j belongs to $\underline{J}^t$ (this follows because i_2 belongs to $\underline{I}^t$ for some type t). After this change, we can discard some of the variables Y_{pi} by using the constraints $Y_{pi} = Y_{p,i_2}$ for i comprised between i_1 and $i_3 - 1$. Specifically, among the variables $Y_{p,i_1}, Y_{p,i_1+1}, \ldots, Y_{p,i_3-1}$, only the variable Y_{p,i_1} is required in the model. Since i_1 is the index of the last arrival in a subsequence of type u, we need only keep the variables Y_{pi} for which $i \in \underline{I}^u$. A similar remark applies to the Z_{pj}. To summarize, recall that P^{ut} is the set of admissible patterns with an exit block of type u and a nonempty entry block of type t. We include the variable Y_{pi} (resp. Z_{pj}) into the model only if the conditions $p \in P^{ut}$ and $i \in \underline{I}^u$ (resp. $j \in \underline{J}^u$) hold. We must now rewrite slightly the constraints to take this change into account.

Let g be the largest index in $\underline{I}^u$. Then the constraints (4.17) must be replaced by $Y_{pi} \leq Y_{pk}$ for each $i \in \underline{I}^u \backslash \{g\}$, where p is a pattern in P^{ut} and k is the smallest index in $\underline{I}^u$ that is greater than i. The constraints (4.18) must also be replaced by $Y_{pg} \leq X_p$ for any p in P^{ut}. A similar remark applies to the constraints (4.21), (4.22) and (4.27). To rewrite the constraints (4.15), we define $g(i,u)$ as the largest index in $\underline{I}^u$ less than or equal to i. The constraints (4.15) can be replaced by

$$\sum_{p \in P} s_p^t X_p + \sum_{u \in T \backslash \{t\}} \sum_{p \in P^{ut}} e_p^t Y_{p,g(i,u)} + \sum_{q \in Q_2} b_q^t X_q \geq a_i^t$$

for $i \in I^t$. If $g(i,u)$ does not exist (for some $u \in T \backslash \{t\}$), the term $\sum_{p \in P^{ut}} e_p^t Y_{p,g(i,u)}$ vanishes. Similarly, the constraints (4.16) can be rewritten as

$$\sum_{p \in P_2} s_p^u Y_{p,g(i,u)} \leq a_i^u.$$

We observe that a constraint of the form (4.15) for some $i \notin \underline{I}^t$ is implied by the same constraint with i replaced by i', where i' is the last index in the subsequence containing i. This follows because $a_i^t \leq a_{i'}^t$ and the left-hand sides of the two constraints are identical. Therefore we keep in the model the constraints (4.15) such that i belongs to $\underline{I}^t$ and discard the others. Of course, the constraints (4.19), (4.20) and (4.26) can be rewritten in the same manner as the constraints (4.15) and (4.16), and we can discard all such constraints with $i \notin \underline{I}^t$.

Clearly, any solution of the "reduced" model can be extended to a solution of the original model by reintroducing the variables $Y_{p,i_1+1}, \ldots, Y_{p,i_3-2}$ and Y_{p,i_3-1} and assigning the value of Y_{p,i_1} to each of them. The solution of the reduced model and the extended solution have the same objective function value. Thus Theorem 4.5.1 applies to the reduced model as well. We report in Section 4.6 the reductions in the numbers of variables and constraints for the chosen instances. Finally, in order to alleviate the notation, we shall use I (resp. I^t, J, J^t) instead of $\underline{I}$ (resp. $\underline{I}^t$, $\underline{J}$, $\underline{J}^t$).

4.5.3 Tightening of the integer programs

We can improve the integer programs described above by adding valid inequalities. Let R^t denote the set of all two-block, one-block or mixed patterns containing at least one slot of type t. Then the inequality

$$\sum_{p \in R^t} X_p \geq \left\lceil \frac{b^t}{\ell} \right\rceil \tag{4.28}$$

is valid for all three versions of the dispatching problem by blocks. Adding these inequalities for all t also accelerates the resolution of the integer programs.

4.5.4 Restrictions on the number of two-block lanes

Suppose that in the current solution the first lane is partitioned according to $p_1 \in P^{ut}$ and the second lane according to $p_2 \in P^{ut}$. Let β_k^u (resp. β_k^t) denote the number of buses of type u (resp. t) in the k^{th} lane (for $k \in \{1, 2\}$). Let also β^u denote $\beta_1^u + \beta_2^u$ and β^t denote $\beta_1^t + \beta_2^t$. The following proposition shows that we may assume, without loss of generality, that the first lane has a single block.

Proposition 4. *The current solution can be replaced by a solution in which*
- *the first lane has a single block,*
- *the second lane has at most two blocks, and*
- *all the lanes (with the possible exception of the first two) are identical to those of the current solution.*

Proof : We consider only the first two lanes of the current solution, since the other lanes will not be modified. If β^u and β^t are equal, we have $\beta^u = \beta^t = \ell$ and the proposition is trivially true. Let us assume first that $\beta^u > \beta^t$ and let r denote $\beta^u - \ell$. Consider the list of buses of type u parked in the first two lanes, ordered by arrival time. Since β_1^u is greater than $\beta^u - \ell = r$, the r^{th} bus of type u arrives at the depot no later than the last bus of type u to be parked in the first lane. For the same reason, since β_2^u is greater than $\beta^u - \ell = r$, the r^{th} bus of type u arrives at the depot no later than the last bus of type u to be parked in the second lane. Therefore the r^{th} bus of type u arrives at the depot no later than any bus of type t. Similarly, one can show that among the departures of the first two lanes, the departure time of the r^{th} duty of type u is at most the departure time of any duty of type t. We conclude that it is possible to park the first r buses of type u and all the buses of type t in the second lane. The other buses of type u will be parked in the first lane as they arrive at the depot.

In the case where $\beta^u < \beta^t$, we consider the $(\ell+1)^{\text{th}}$ bus of type t. This bus must arrive later than any bus of type u since $\beta_1^t > \beta^t - \ell$ and $\beta_2^t > \beta^t - \ell$. The conclusion follows by the same argument as above. □

We can deduce the following proposition from the previous one by induction.

Proposition 5. *The current solution can be replaced by a solution in which*
- *at most one lane is partitioned according to some $p \in P^{ut}$ (for any couple (u,t)), and*
- *all the mixed lanes are identical to those of the current solution.*

The last proposition implies that we can redefine the feasible sets of the formulations in Sections 4.4.1 to 4.4.3 by adding the constraints

$$\sum_{p \in P^{ut}} X_p \leq 1, \qquad \forall\, t \in T, \forall\, u \in T \setminus \{t\}. \tag{4.29}$$

In the two versions of the dispatching problem that include an objective function, only suboptimal solutions can violate the constraints (4.29). Thus adding the constraints (4.29) does not remove any optimal solution. Also, if we add the constraints (4.29) to any of the models, we can also add the constraints that for any two-block pattern p, X_p and the Y_{pi} (for all i) belong to $\{0,1\}$. Finally, the inequalities

$$\sum_{u \in T \setminus \{t\}} \sum_{p \in P^{ut}} X_p \leq 1, \qquad \forall\, t \in T$$

are not valid, as the following example demonstrates. Consider an instance of the dispatching problem by blocks in which $v = 2$, $\ell = 3$ and $T = \{A, B, C\}$. Assume that the arrival sequence is $BBACAA$ and the departure sequence $CAABBA$. The only feasible solution not requiring maneuvers is the solution in which a lane contains the sequence BBA (from exit block to entry block) and another lane the sequence CAA. Therefore $\sum\limits_{u \in T \setminus \{t\}} \sum\limits_{p \in P^{ut}} X_p$ is equal to 2 in this case.

4.6 Computational results

In this section we show that real-life instances of the dispatching problem by blocks can be solved within relatively short times by a commercial MIP solver, in particular CPLEX 8.1. CPLEX combines several strategies to solve mixed integer programs : cutting planes, the classical branch-and-bound paradigm and strategies for finding heuristic solutions (and thus computing upper bounds). The user may also select priorities for the branching variables.

In what follows we describe the instances used in our experiments and present the numerical results that helped us choose the best strategy for our instances. In devising this strategy we considered the possibility of using priorities for the branching variables and the possibility of including new constraints or cutting planes. The cutting planes (4.28) were always included into our models but we carried out some experiments with and without the constraints (4.29). In the case where α, the optimal value of $\sum\limits_{q \in Q_2} X_q$, is not an integer, we considered the possibility of including some constraints of the form

$$\sum_{q \in Q_2} X_q \geq \lceil \alpha \rceil. \tag{4.30}$$

We present the results obtained by using the best strategy (i.e., the best mix of priorities and constraints) and evaluate the performance of this strategy as a function of the number of bus types and the required maneuvers. Finally, we assess the solutions obtained by comparing their robustness to that of "ideal" solutions.

4.6.1 The instances

All instances used in our tests are based on a real-life instance provided by the Société de Transport de Montréal. This real-life instance consists of a sequence of 124 arrivals and a sequence of 124 departures. There are four bus types and a type is specified for each arrival (resp. departure). The values of v (the number of lanes) and ℓ (the number of slots per lane) vary from one test to the next. Because the integer 124 has few divisors, we have included in the instance 20 extra buses, while retaining the relative frequencies of the bus types. We have thus obtained an instance with 144 arrivals and 144 departures, and we have carried out tests for the following values of ℓ : 6, 8, 9, 12, 16, 18 and 24. The last three values (16, 18 and 24) are not likely to be found in real-life instances, but we used them to test our strategy.

We modified the instance with 144 buses and 4 types to obtain instances with the same number of buses but 6 or 8 types. To construct a new instance, we "split" a type into two types. For example, one can see in Table 4.2 that the type A of the 4-type instance has been split into the type A of the 6-type instance (with 40 buses) and the type E of the 6-type instance (with 41 buses).

Tableau 4.2 – Number of buses of each type for all instances

	Bus type							
	A	B	C	D	E	F	G	H
4 bus types	81	43	8	12	-	-	-	-
6 bus types	40	22	8	12	41	21	-	-
8 bus types	40	22	3	6	41	21	5	6

Assuming that there are four bus types, we have also computed the numbers of variables and constraints for the dispatching problem by blocks with two-block rearrangements. Recall that the objective of this model is (4.12) and its constraints are

(4.13)–(4.18), (4.24), (4.26), (4.20), (4.21), (4.27), (4.23) and (4.25). The numbers of variables and constraints are presented in Table 4.3, under the heading "Complete", for several values of v and ℓ. As we saw in Section 4.5, we may reduce the number of indices representing the arrivals (resp. the departures). The numbers of variables and constraints of the reduced model are presented in Table 4.3 under the heading "Reduced". Since CPLEX preprocesses the linear programs before attempting to solve them, we have included the numbers of variables and constraints after preprocessing under the heading "Preprocessed". Finally, the average number of nonzero entries per column of the linear program is presented under the heading "Nz/Col".

Tableau 4.3 – Numbers of variables and constraints in the instances with 4 types

		Complete		Reduced		Preprocessed		
v	ℓ	Var	Rows	Var	Rows	Var	Rows	Nz/Col
24	6	17434	18467	1410	1569	1040	1191	9.4
18	8	24406	25391	1991	2101	1385	1488	9.6
16	9	27892	28848	2293	2381	1552	1632	9.6
12	12	38350	39234	3397	3431	1850	1892	9.4
9	16	52294	53087	5048	5031	2127	2153	9.3
8	18	59266	60011	6027	5984	2226	2256	9.3
6	24	80182	80553	8739	8618	2551	2568	9.1

Observe that the reduced model is much smaller than the complete one for every instance. The CPLEX preprocessing enables one to reduce the model size further (by 24% to 70%). The size of the preprocessed model for the largest instance is reasonable, and we expect CPLEX to solve it within a relatively short time. Finally we note that the instance size is an increasing function of ℓ, no matter the model used. This follows because the number of patterns considered is itself an increasing function of ℓ. We expect the instances with large values of ℓ to be more difficult to solve than the others.

4.6.2 The results

To select the resolution strategy, we compared several approaches by testing them on the 6-type instances. We now present these approaches and the results that enabled us to choose the best strategy. For the *basic approach*, we used the CPLEX MIP solver with the following features : dual simplex algorithm, strong branching, generation of cutting planes by CPLEX at node 0 and heuristic search of a feasible integral solution (when selected by CPLEX). The computations were carried out on a Sun Ultra 10 with 440 MHz. The CPU times for all instances are displayed in Table 4.4, along with the number of cuts generated by CPLEX, the number of variables with fractional values at the root node and the number of nodes in the branch-and-bound tree. In the column labeled *Nb Nodes*, an asterisk indicates that the optimal solution was found by the CPLEX heuristic. We did not report the CPU time spent solving the LP relaxation at the root node because it represented at most 3% of the total CPU time. Note that as the lane length increases, the problem size (see Table 4.3) and the related CPU time also increase.

Tableau 4.4 – Results with the basic approach

v	ℓ	Nb Cuts	Nb Frac	Nb Nodes	CPU Time (s)
24	6	103	1	1*	2
18	8	37	37	1*	2
16	9	142	0	1	6
12	12	48	141	51*	113
9	16	76	258	179*	489
8	18	66	334	751	2803
6	24	74	548	7203	24628

In Table 4.5, we report the results obtained when the constraints (4.29) are added to the model. Adding these constraints increases the computing times, except in the case where v is equal to 8. Actually, if one uses the model without the constraints (4.29), the bound obtained after the generation of cutting planes by CPLEX at node 0 is usually better than the bound obtained for the model including the constraints (4.29).

Tableau 4.5 – Results with the constraints (4.29)

		Nb	Nb	Nb	CPU
v	ℓ	Cuts	Frac	Nodes	Time (s)
24	6	70	50	1*	2
18	8	24	47	1*	3
16	9	74	131	1*	18
12	12	64	103	140	220
9	16	121	240	647*	1803
8	18	106	373	421*	1636
6	24	824	436	7268	36469

We also considered priorities in choosing the branching variables. After several experiments, we assigned the highest priority to variables of the form X_p (where p is a one-block pattern), followed by expressions of the form $\sum_{p \in R^t} X_p$ (i.e., the left-hand sides of the constraints (4.28)), by expressions of the form $\sum_{p \in P^{tu}} X_p$ (i.e., the left-hand sides of the constraints (4.29)), by the variables X_p such that p is a two-block pattern and finally by all the other variables. The results obtained by using the basic approach with these branching priorities are reported in Table 4.6. Note that this table consists of two parts. In the second part, we present the results obtained when one adds the constraints of the form (4.30) as the need arises. In the first part of Table 4.6, these constraints are not used.

Table 4.7 is similar to Table 4.6, with the exception that the constraints (4.29) are now part of the model. The results reported in Tables 4.6 and 4.7 show that the

Tableau 4.6 – Results with branching priorities

				Without (4.30)		With (4.30)	
		Nb	Nb	Nb	CPU	Nb	CPU
v	ℓ	Cuts	Frac	Nodes	Time (s)	Nodes	Time (s)
24	6	103	1	1*	2	1*	2
18	8	37	37	1*	2	1*	3
16	9	142	0	1	6	1	6
12	12	55	141	50	72	2*	28
9	16	80	258	324	507	352*	644
8	18	61	334	341*	897	191*	379
6	24	71	548	5470	10697	3735	7175

use of branching priorities reduces computing times for almost all the instances that require branching (whether the constraints (4.29) are included or not). On the other hand, the use of the constraints (4.30) (as the need arises) seems to increase the performance of the basic approach but to decrease it when the constraints (4.29) are included into the model. We conclude that the best strategy is the one combining the branching priorities and the constraints (4.29), which seem to be effective in this context, contrary to what we observed in Tables 4.4 and 4.5.

Finally, in order to assess the usefulness of the redundant constraints (4.14) (see Section 4.4), we decided to compute the solutions of the instances without these constraints. Table 4.8 contains the results obtained by excluding the constraints (4.14) but including branching priorities and the constraints (4.29). The results, especially those for the last three instances, demonstrate clearly that the constraints (4.14) are useful, in the sense that their inclusion entails a significant decrease in computing times. Indeed, if the constraints (4.14) are not included into the model, the numbers of variables and constraints after preprocessing are greater than otherwise, the number of variables with fractional values is larger and the CPLEX heuristic for finding a good integral solution has a very poor performance.

Tableau 4.7 – Results with branching priorities and the constraints (4.29)

				Without (4.30)		With (4.30)	
		Nb	Nb	Nb	CPU	Nb	CPU
v	ℓ	Cuts	Frac	Nodes	Time (s)	Nodes	Time (s)
24	6	70	50	1*	2	1*	2
18	8	24	47	1*	3	1*	3
16	9	74	131	1*	18	1*	18
12	12	59	71	38	43	16*	35
9	16	118	240	229*	489	156	317
8	18	61	373	146	355	671	1265
6	24	273	436	1584	4612	8011*	18891

After finding the best strategy through experiments on the 6-type instances, we applied the best strategy (i.e., the strategy combining the branching priorities and the constraints (4.29)) to all the instances described at the beginning of this section, i.e., the 4-type, 6-type and 8-type instances. The results are reported in Table 4.9. Each of the three parts of this table contains the optimal value of the optimal relaxation, the optimal value of the integer program, the number of nodes of the branch-and-bound tree and the solution time (in seconds). Note that the "value" INF indicates an infeasible instance, and that the value of M (the weight of the main objective) has been set to 100. Thus only two instances have optimal solutions containing mixed lanes : the 6-type instance with 6 lanes and the 8-type instance with 9 lanes. For each instance, the program found an optimal solution in less than 8.15 hours and the 4-type instances were solved in less than one minute. The instances with $\ell \geq 16$ are the most difficult to solve, but in practice the lane length will not be greater than 12.

In Table 4.10, we present the results obtained by the best strategy for the version of the dispatching problem with arbitrary rearrangements. This strategy can solve all the instances and, when v is at least 9 and the number of types is at least 6, its

Tableau 4.8 – Results with branching priorities and the constraints (4.29) but without the constraints (4.14)

		Nb	Nb	Nb	CPU
v	ℓ	Cuts	Frac	Nodes	Time (s)
24	6	53	136	11	15
18	8	52	113	10	27
16	9	55	144	64	118
12	12	71	144	80	166
9	16	133	163	349	1448
8	18	73	182	602	2043
6	24	31	762	20699	82705

Tableau 4.9 – Results with branching priorities and the constraints (4.29)

		4 Types				6 Types				8 Types			
		Z	Z	Nb	CPU	Z	Z	Nb	CPU	Z	Z	Nb	CPU
v	ℓ	LP	IP	Nodes	Time (s)	LP	IP	Nodes	Time (s)	LP	IP	Nodes	Time (s)
24	6	2.0	2	1*	1	3.0	3	1*	2	4.0	4	1*	36
18	8	2.0	2	1*	1	2.0	3	1*	3	4.1	6	100	89
16	9	2.0	2	1*	3	3.0	4	1*	18	4.5	7	1*	23
12	12	2.0	2	1*	2	3.0	4	38	43	4.7	7	182	243
9	16	2.4	3	1*	6	3.3	6	229*	489	5.7	206	8958	29357
8	18	2.3	4	30	20	4.3	6	146	355	6.8	INF	4048	7616
6	24	2.5	4	61*	38	4.0	204	1584	4612	INF	INF	1	16

performance is much better than on the model with two-block rearrangements. The only instances whose optimal solutions contain mixed lanes are the 6-type instance with 6 lanes of length 24 and the three 8-type instances with $\ell \geq 16$. We have also noticed that the performance of the CPLEX heuristic for finding an integral solution is even worse for the model with arbitrary rearrangements than for the model with two-block rearrangements.

Tableau 4.10 – Arbitrary rearrangement results with branching priorities and the constraints (4.29)

		4 Types				6 Types				8 Types			
		Z	Z	Nb	CPU	Z	Z	Nb	CPU	Z	Z	Nb	CPU
v	ℓ	LP	IP	Nodes	Time (s)	LP	IP	Nodes	Time (s)	LP	IP	Nodes	Time (s)
24	6	2.0	2	1	2	3.0	3	13	11	4.0	4	1	7
18	8	2.0	2	1	1	2.0	3	6	19	4.1	6	85	105
16	9	2.0	2	4	6	3.0	4	11	27	4.5	7	106	158
12	12	2.0	2	1	3	3.0	4	70	90	4.7	7	186	332
9	16	2.4	3	51	38	3.3	6	193	579	5.7	108	1450	5436
8	18	2.3	4	29	29	4.3	6	373	1158	6.8	206	127	331
6	24	2.5	4	30	29	4.0	105	640	2050	166.7	402	57	182

4.6.3 The robustness of the solutions

To assess the robustness of the solutions of the model with two-block rearrangements, we decided to compare ν, the total number of breakpoints in the solution, to a lower bound. Note that ν is actually the sum of the number of two-block lanes and the number of mixed lanes, since by assumption, any mixed lane in a feasible solution can be transformed into a two-block lane. To compute the lower bound on ν, we disregard the arrival and departure sequences. Therefore LB, the lower bound, is the optimal value of the following integer program, where X_k^t denotes the number of type t buses assigned to the k^{th} lane and Y_k^t is equal to 1 if X_k^t is greater than 0.

$$\begin{array}{rll}
\text{Minimize} & \sum_{k=1}^{v} \sum_{t \in T} Y_k^t - v & \\
\text{s.t.} & & \\
& \sum_{k=1}^{v} X_k^t = b^t & \forall\, t \in T \\
& \sum_{t \in T} X_k^t = \ell & \forall\, k = 1, 2, \dots, v \\
& X_k^t \leq \min\{b^t, \ell\}\, Y_k^t & \forall\, t \in T, \forall\, k = 1, 2, \dots, v \\
& X_k^t \text{ are nonnegative integers} & \text{for all valid indices} \\
& Y_k^t \in \{0, 1\} & \text{for all valid indices}
\end{array}$$

The values ν for the solutions of Table 4.9 may be found in Table 4.11, along with the lower bound LB. We note that ν is equal or close to LB, a fact that supports the usefulness of our models. Indeed, modeling the dispatching problem as we did produces solutions that respect the arrival and departure sequences and are almost as robust as the "ideal" solutions (those that do not take these sequences into account).

Tableau 4.11 – Assessment of the robustness of the solutions

		4 Types		6 Types		8 Types	
v	ℓ	LB	ν	LB	ν	LB	ν
24	6	2	2	3	3	4	4
18	8	2	2	3	3	5	6
16	9	2	2	4	4	4	7
12	12	2	2	3	4	5	7
9	16	3	3	4	6	6	8
8	18	3	4	5	6	6	INF
6	24	3	4	4	6	6	INF

4.7 Conclusion

We have presented a new version of the dispatching problem, in which we address the need for robust solutions. Since such solutions contain mostly one-block or two-block lanes, we have formulated the dispatching problem by blocks as an integer linear program based on the enumeration of admissible patterns. Our results show that real-life instances can be solved within reasonable times by a commercial MIP solver ; thus robust solutions of the dispatching problem can be computed fairly easily. In future research, we will consider depots whose layout is more complex than the ones considered in this article. We will also take precedence constraints into account and consider dispatching problems in which some buses come back to the depot between rush hours. Finally, in the case when there is no feasible solution to our model, it might be desirable to allow some type mismatches for the morning departures. We plan to investigate the problem of minimizing the number of mismatches.

Acknowledgments

The authors wish to thank the personnel of the Société de Transport de Montréal for helping them define the problem studied in this article and for providing real-life instances. They gratefully acknowledge the help of the two anonymous referees and the financial support of NSERC (Natural Sciences and Engineering Research Council of Canada).

References

Blasum, U., Bussieck, M.R., Hochstättler, W., Moll, C., Scheel, H.-H. and Winter, T. (1999). Scheduling Trams in the Morning. *Mathematical Methods of Operations Research* **49**, 137–148.

Freling, R., Lentink, R.M., Kroon, L.G., and Huisman, D. (2005). Shunting of Passenger Train Units in a Railway Station. *Transportation Science* **39**, 261–272.

Gallo, G. and Di Miele, F. (2001). Dispatching Buses in Parking Depots. *Transportation Science* **35**, 322–330.

Garey, M.R.and Johnson, D.S. (1979). *Computers and Intractability : A Guide to the Theory of NP-Completeness.* Freeman, W.H. New York.

WINTER, T. AND ZIMMERMANN, U.T. (2000). Real-time dispatch of trams in storage yards. *Annals of Operations Research* **96**, 287–315.

CHAPITRE 5 : PARKING BUSES IN A DEPOT USING BLOCK PATTERNS : A BENDERS DECOMPOSITION APPROACH FOR MINIMIZING TYPE MISMATCHES

Ce chapitre présente l'article "Parking Buses in a Depot Using Block Patterns : a Benders Decomposition Approach for Minimizing Mismatches". Cet article est accepté pour publication dans *Computers and Operations Research.*

L'affectation d'un autobus à un départ qui requiert un type différent d'autobus produit une affectation erronée. Afin d'éviter les manoeuvres des autobus la nuit, les compagnies de transport permettent certains affectations erronées mais leur nombre doit être minimisé. Dans ce chapitre, nous examinons le problème de stationnement des autobus par bloc (défini dans le chapitre précédent) en minimisant le nombre d'affectations erronées. Ce problème sert à partitionner les voies du dépôt en au plus deux blocs par voie. Cela n'est pas toujours possible ce qui nécessite de permettre des incompatibilités aux blocs : chaque arrivée (resp. départ) affectée à un bloc de mauvais type produit une incompatibilité.

Dans un premier temps, nous considérons le problème de stationnement par bloc en minimisant les incompatibilités aux blocs et le nombre de voies avec deux blocs. Cet objectif vise principalement à maximiser la robustesse de la solution. Nous proposons un programme linéaire en nombres entiers qui peut être résolu facilement pour les instances réelles en utilisant CPLEX. Notons que le nombre d'incompatibilités aux blocs (à l'arrivée et au départ) constitue une borne supérieure sur le nombre d'affectations erronées.

Ensuite, nous étendons le modèle proposé pour minimiser les incompatibilités aux blocs à un modèle linéaire de grande taille qui minimise une combinaison linéaire du nombre d'incompatibilités aux blocs, du nombre de voies à deux blocs et du nombre d'affectations erronées. Pour résoudre ce dernier modèle, nous développons une approche de décomposition de Benders qui produit un sous-problème avec des variables entières. Nous introduisons pour la première fois dans la littérature une stratégie de branchement qui traite l'intégrité des variables du sous-problème dans une telle décomposition. La stratégie considérée impose des décisions sur les variables du problème maître qui prennent déjà des valeurs entières.

Pour évaluer l'efficacité de la décomposition de Benders combinée avec la stratégie de branchement proposée, nous avons réalisé des tests sur des instances expérimentales de taille moyenne obtenues d'instances réelles de la STM (Société de Transport de Montréal). L'expérimentation numérique montre qu'il est très difficile de résoudre ces instances directement avec CPLEX mais qu'elles peuvent être résolues dans un temps raisonnable avec l'approche de résolution proposée dans ce chapitre.

Parking buses in a depot using block patterns : a Benders decomposition approach for minimizing type mismatches

Mohamed Hamdouni, Guy Desaulniers and François Soumis

Department of mathematics and industrial engineering

École Polytechnique and GERAD

Montréal, Québec, Canada

{Mohamed.Hamdouni, Guy.Desaulniers, Francois.Soumis}@gerad.ca

Abstract

In a transit authority bus depot, buses of different types arrive in the evening to be parked in the depot for the night, and then dispatched in the morning to a set of routes, each of which requests a specific bus type. A type mismatch occurs when the requested type is not assigned to a morning route. We consider the problem of assigning the buses to the depot parking slots such that the number of mismatches is minimized, under the constraint that the buses cannot be repositioned overnight. As in Hamdouni *et al.*(2004), we seek robust solutions by assigning a block pattern to each depot. This pattern partitions the lane into at most two blocks, each block containing buses of a given type. Since it may not be possible to respect the selected block patterns, the problem also involves a second objective which is to minimize the discrepancy between the bus type assignment to the parking slots and the block patterns. In this paper, we first study the simplified case where only the second objective is taken into account. We model this simplified problem as an integer linear program and show that practical instances of it can easily be solved using a commercial MIP solver. Then we formulate the general case as an extension of the simplified model and propose to solve it with a Benders decomposition approach embedded in a branch-and-bound procedure. This procedure is required because the Benders decomposition yields a subproblem with integrality constraints. Of particular interests, we develop strong pruning criteria and an innovative branching strategy that imposes decisions on the master problem variables which already take integer values. Computational results for the general case are also reported.

5.1 Introduction

Large public transport authorities own a fleet of buses that serves various bus lines in a metropolitean area. This area is usually divided into sectors and a subfleet of buses is assigned to each sector. This subfleet often contains different types of buses : for instance, low-floor buses, articulated buses, advertising buses, as well as standard buses. Certain bus types might be preferable for certain bus lines. For example, one should try to assign low-floor buses on lines with a high proportion of elderly people. A depot, housing the buses when they are idle for a long period of time, is located in each sector. The depot layout varies a lot from depot to depot, but most layouts are subdivided into lanes of variable length. Each lane is subdivided into parking slots. For safety reasons, a bus is not allowed to go backwards in a depot, nor to change lane. The lanes are thus oriented, with an entry endpoint and an exit endpoint for each lane.

As surveyed in Desaulniers and Hickman (2003), one of the operational planning problems faced by the transit authorities consists of parking and dispatching buses in a depot. In large cities, depots are often crowded from late evening to early morning. Therefore, when a particular morning route requires a bus of a specific type, several buses might be moved to clear the way out, potentially yielding a delay. To avoid such delays, buses should properly be parked upon their arrivals during the preceding evening. However, this is not always possible. Hence, the transit authorities resort to two alternatives. In the first alternative (see Hamdouni *et al.*, 2004), the mispositioned buses are reordered during the night, one lane at a time. This choice requires additional night-time personnel to reorder the buses and might be disturbing close to a residential neighborhood. In the second alternative, a bus directly accessible is always assigned to every morning pull-out even if this bus is not of the type requested by the corresponding bus route. In this case, we say that a *type mismatch* or simply a *mismatch* occurs. This second option is often more advantageous than the first one because it does not suffer from the above two drawbacks. It may, however, yield a reduction in service quality depending on the proposed type substitutions. In this paper, we study this second option and the objective of minimizing the number of mistmatches.

Even though buses leave and arrive all day long at a depot, dispatching the buses in the middle of the day is not a difficult problem due to the small number of buses present in the depot at that time. Consequently, we consider the bus dispatching problem occuring at night which consists of assigning parking slots to the buses arriving in the evening in such a way that these buses can be dispatched to the next morning routes with a minimum number of mismatches. We assume that the buses cannot be repositioned overnight. We study this problem in a deterministic context where the time and type of each evening arrival and each morning departure are known. However, in practice, buses rarely respect their arrival schedule and sometimes slightly deviate from their departure schedule. Obviously, permuting the order of two arrivals or two departures of different types may increase the number of mismatches. Therefore, to increase solution robustness, Hamdouni *et al.*(2004) proposed to configure as much as possible the lanes according to one-block and two-block patterns. In a one-block lane, buses are all of the same type, while in a two-block lane, they are of two different types and grouped by type. An *admissible lane pattern* is thus a partition of a lane into at most two blocks, an *entry* block (the closest to the lane entry) and an *exit* block, which specifies the number of slots in each block and their assigned type. When a pattern contains a single block, we consider it as an exit block. These lane patterns have the advantages to be easy to implement in practice and to provide robust solutions since swapping the order of two arrivals assigned to the same one-block lane does not increase the number of mismatches while it can increase it with a small probability when these arrivals are assigned to a two-block

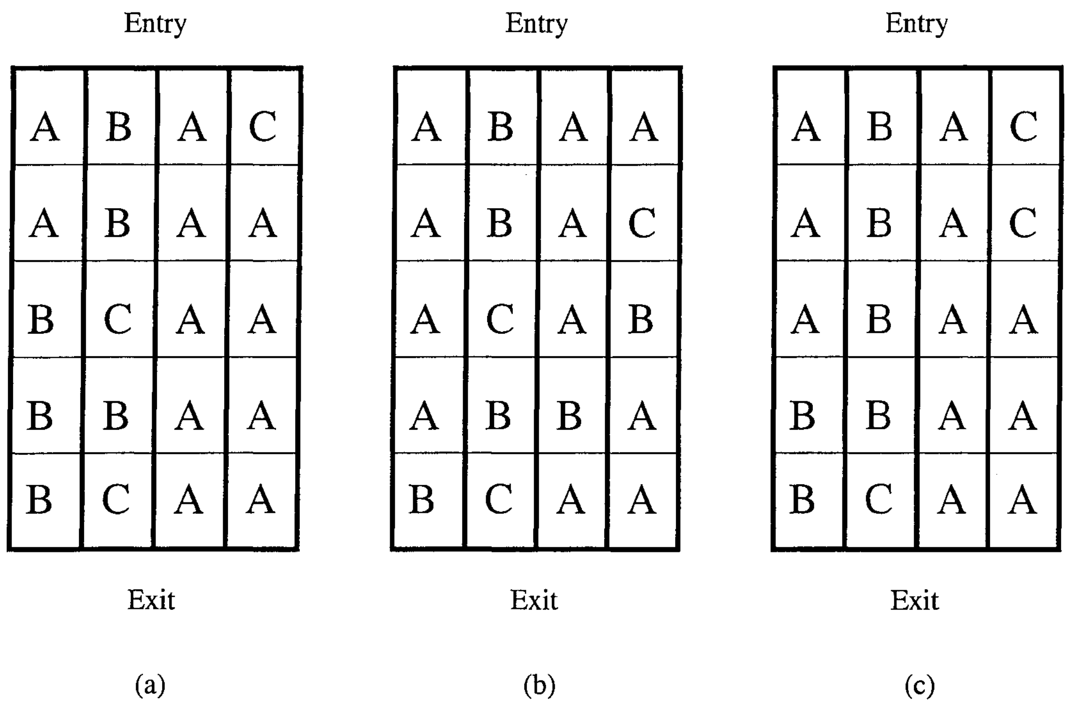


Figure 5.1 – Three views from above of the same depot, showing (a) an arrival assignment, (b) a departure assignment, and (c) lane block patterns.

lane. Obviously, the one-block lanes are more robust than the two-block lanes and should be favored.

The bus dispatching problem that we consider has two objectives, namely, minimizing the number of mismatches and maximizing solution robustness by using as much as possible one-block and two-block lanes. For clearly defining these objectives, let us look at the example illustrated in Figure 1. This figure presents three views from above of the same depot, which is full of buses and divided into four lanes of five slots (the small rectangles). We assume that the buses enter the lanes at the top and exit them at the bottom. The letters A, B and C correspond to three different bus types. There are 11, 6 and 3 buses of types A, B, and C, respectively. Figure 1(a) shows an assignment of the types to the slots that is compatible with the evening arrival sequence, while Figure 1(b) illustrates a similar assignment compatible this time with the morning departure sequence. One can see that, when both sequences are respected as planned, these two assignments yield six mismatches (two in lane 1, one in lane 3, and three in lane 4, when the lanes are numbered from the left). On the other hand, Figure 1(c) depicts block patterns that the above two assignments should try to respect in order to provide a robust solution. For instance, lane 1 has a two-block pattern with two type B buses in its exit block and three type A buses in its entry block, while lane 3 has a one-block pattern with five type A buses. One can notice that the arrival assignment does not respect the patterns for three slots : the third of lanes 1 and 2, and the fourth (from the bottom) of lane 4. We say that the buses assigned to these slots yield *incompatible arrivals.* Similarly, we see that the departure assignment exhibits five *incompatible departures*, namely one in lanes 1, 2 and 3, and two in lane 4. Notice that the choice of each lane pattern is a decision to make while solving the problem.

We can now clearly state the problem that we address in this paper and that we call the *problem of dispatching buses by blocks with mismatches* (PDBBM). Given a sequence of bus arrivals with known types and a sequence of morning departures with known requested types, determine a pattern for each lane and assign each arrival and each departure to a slot such that the arrival and departure sequences are respected and all lane patterns have one or two blocks of buses of the same type. The objective consists of minimizing a weighted sum of the number of mismatches, the number of incompatible arrivals, the number of incompatible departures, and the number of two-block lanes. Note that, in this paper, we consider that minimizing the last three terms of the sum is somewhat equivalent to maximizing solution robustness (see the empirical justification proposed in Hamdouni *et al.*, 2004). Also, we consider the minimization of the number of two-block lanes as a secondary objective.

We will first study a special case of this problem, called the *problem of dispatching buses by blocks with incompatibilities* (PDBBI), which is defined by setting to zero the weight associated with the number of mismatches, that is, the objective solely seeks at maximizing robustness. The PDBBI is interesting because it is less complex than the PDBBM and provides very robust solutions that do not contain a large number of mismatches. Indeed, it is easy to show that the number of incompatible arrivals and departures is an upper bound on the number of mismatches.

As in Hamdouni *et al.*(2004), we make the following simplifying assumptions :

1. The buses have all the same length.
2. The lanes have all the same length, that is, the same number of parking slots.
3. The number of buses is equal to the total number of parking slots.
4. The depot is empty when the first bus arrives.
5. Each lane has an independent entry and an independent exit, i.e., there are no precedence constraints between the lanes.

The models that we propose can easily be adapted when the assumptions 2–4 do not hold (as we shall see later). On the other hand, dropping the first or the last assumption is much more difficult. The resulting problem variant will not be addressed in this paper.

This paper presents two integer linear programming models, one for the PDBBI and the other for the PDBBM. The first model can be solved by a commercial MIP solver. For the second model which is much larger than the first one, we propose to use a Benders decomposition approach where the master problem corresponds to the PDBBI with additional Benders optimality and feasibility cuts. For both problems, we provide computational results obtained on a real-world instances.

5.1.1 Literature review

In the literature, very few papers have addressed the bus dispatching problem. Considering a variant of this problem in which the buses can be repositioned during the night, Gallo and Di Miele (2001) formulated this problem as an integer linear program where the objective minimizes the number of repositioning moves (maneuvers). They mentioned that this program can easily be adapted to the objective of minimizing the number of mismatches. To solve it, they developed a heuristic Lagrangian decomposition approach that produced good quality solutions in reasonable computational times for medium-sized real-world datasets (up to 70 buses).

Recently, Hamdouni *et al.*(2004) introduced another variant of the bus disptaching problem that highly favors the robustness of the solutions with respect to the variability of the arrival and departure sequences. In this variant, which is called the problem of dispatching buses by blocks, a maximum number of one-block and two-block lanes must be used. These lanes do not need any maneuver during the night. The other lanes, called the mixed lanes, contain mispositioned buses and need to be reordered during the night. The authors showed that this problem is also NP-complete and studied two cases that differ by the configurations allowed for the mixed lanes. For each case, they proposed an integer linear program based on the enumeration of all admissible lane patterns and showed that real-life instances involving 144 buses can be solved using a commercial MIP solver in reasonable solution times.

Prior to these works, some researchers have studied train (or tram) dispatching problems that exhibit many similarities with the bus dispatching problem. They mainly differ by the following characteristics : i) trains are typically parked in a depot using a last-in, first-out policy while buses use a first-in, first-out policy, ii) the train length is much more variable than the bus length, and iii) shunting trains is not as easy as maneuvering buses because trains move on rail tracks. Consequently, adapting a solution method designed for a train dispatching problem to a bus dispatching problem or vice-versa is usually not straightforward. In this regard, the first characteristic causes no difficulty, while the second and third may be harder to handle. In fact, variable lengths can often be taken into account in bus dispatching approaches.

However, when mismatches are allowed, substitution costs should be considered in those approaches when the lengths of the vehicles differ significantly. As for the third characteristic, it involves a different way of costing the maneuvers since, in a bus dispatching problem, performing one maneuver in a lane requires getting all the buses in this lane out of the depot and, therefore, costs the same as performing several maneuvers in this lane. Indeed, once the buses are out, they can be sorted in any order at the same cost. This is not the case for trains that must remain on tracks. Finally, it should be noted that, in practical instances, the number of trains to consider and the average number of trains per track are much smaller than their respective counterparts for buses. Thus, the train dispatching problems are, in general, more complex than the bus dipatching problems, but usually much smaller in practice.

The main works on train and tram dispatching problems are the following ones. In 1999, Blasum *et al.*studied the departure part of a tram dispatching problem where the trams of different types are stacked on the sidings of a depot and must be assigned to morning departures. They showed that this simplified dispatching problem is NP-complete. Winter and Zimmermann (2000) considered two versions of a complete tram dispatching problem that only differ by the objective function : minimize the number of shunting movements (maneuvers) needed to reposition the trams during the night or minimize the number of mismatches upon dispatching the trams in the morning. They showed that both versions are NP-hard and proposed a quadratic assignment model that can be linearized for the first version and an integer linear program for the second one. Since both models required large computational times for small-sized instances, they also developed heuristics for obtaining suboptimal solutions in reasonable solution times. Recently, Freling *et al.*(2005) considered a train shunting problem where trains can be broken up into train units before shunting them. The authors proposed a two-step approach for this problem. In the first step, they used a binary linear program for assigning each inbound train unit to an outbound train unit. In the second step, they applied a column generation method to a set partitioning type model for assigning groups of train units to shunt tracks.

5.1.2 Contributions

In this paper, we address the PDBBM and its special case, the PDBBI. The PDBBM is a problem that combines the objective of using as much as possible one-block and two-block lanes to generate robust solutions as in Hamdouni *et al.*(2004) and the objective of minimizing the number of mismatches as in Winter and Zimmermann (2000) and Gallo and Di Miele (2001). The PDBBI is a similar problem that replaces this second objective with one of minimizing an upper bound on the number of mismatches.

The main contributions of this paper are as follows. Firstly, we present an integer programming formulation for the PDBBI and show that real-world PDBBI instances can be easily solved using a commercial MIP solver. Secondly, we formulate the PDBBM as an extension of the PDBBI model and propose to solve the resulting large-scale integer linear program using a Benders decomposition method embedded in a branch-and-bound procedure. This procedure is needed because Benders decomposition yields, in this case, a master problem and a subproblem that both involve integrality requirements. Thirdly, we develop for this procedure a simple way for generating integer solutions, strong pruning criteria and an innovative branching scheme that imposes decisions only on the master problem variables (even if they take integer values). These decisions are designed to favor the satisfaction of pruning criteria. Finally, we show through computational experiments that this methodology can be used to solve medium-sized instances of the PDBBM.

The remainder of this paper is organized as follows. Section 5.2 provides a mathematical formulation for the PDBBI. Section 5.3 presents the computational results for the PDBBI. In section 5.4, we formulate the PDBBM as a large-scale integer program while, in Section 5.5, we develop the proposed Benders decomposition approach for the PDBBM. Section 5.6 gives the computational results for the PDBBM and Section 5.7 draws some conclusions.

5.2 A mathematical formulation for the PDBBI

In this section, we provide a mathematical formulation for the PDBBI, which is based on the enumeration of the admissible lane patterns as in Hamdouni *et al.*(2004). The following notation will be used throughout the text. Let v be the number of lanes in the depot, ℓ the number of parking slots in a lane, n the overall number of buses, T the set of bus types, and b^t the number of buses of type $t \in T$. Note that, by assumption, $n = \sum_{t \in T} b^t = v\ell$. Denote by P the set of all admissible patterns and by P_2 the set of all admissible two-block patterns ($P_2 \subset P$). For each pattern $p \in P$, let s_p^t be the number of buses of type $t \in T$ in its exit block and e_p^t the number of buses of type $t \in T$ in its entry block. With each pattern $p \in P$, we associate a nonnegative integer variable X_p that indicates the number of lanes to which pattern p is assigned.

Let I be the set $\{1, 2, ..., n\}$ (used to label the arrivals in their arrival order) and I^t the subset $\{i \in I |\, \text{the arrival } i \text{ is of type } t\}$ for $t \in T$. Denote by ha_i the time of the arrival $i \in I$ and by a_i^t the number of buses of type $t \in T$ whose arrival time is at most ha_i. For each pair of two-block pattern $p \in P_2$ and arrival $i \in I$, we define a nonnegative integer variable Y_{pi} that represents the number of lanes partitioned according to pattern p whose exit block is full at time ha_i. For each triplet of type $t \in T$, arrival $i \in I^t$ and type $u \in T \setminus \{t\}$, we also define a binary variable E_i^u that takes value 1 if the arrival i is assigned to a slot of type u according to the chosen lane patterns, and 0 otherwise. Therefore, when $E_i^u = 1$, arrival i (of type t) yields an incompatible arrival.

A similar notation is defined for the morning departures. Let $J = \{1, 2, ..., n\}$ and $J^t = \{j \in J |\, \text{the departure } j \text{ is of type } t\}$ for $t \in T$. Denote by hd_j the time of the departure $j \in J$ and by d_j^t the number of buses of type $t \in T$ whose departure time is at most hd_j. With each pair of pattern $p \in P_2$ and departure $j \in J$, we associate a nonnegative integer variable Z_{pj} that indicates the number of lanes partitioned according to pattern p whose exit block is empty at time hd_j. Furthermore, for each triplet of type $t \in T$, departure $j \in J^t$ and type $u \in T \setminus \{t\}$, we define a binary variable F_j^u that takes value 1 if the bus parked in a slot of type u according to the chosen lane patterns is assigned to the departure j, and 0 otherwise. Hence, when $F_j^u = 1$, an incompatible departure occurs with the departure j.

Using this notation, we propose to formulate the PDBBI using an implicit model similar to the one proposed in Hamdouni *et al.*(2004). The model is implicit in the sense that its solution does not explicitely assign each arrival and each departure to a parking slot. However, it guarantees that such an assignment can be found a posteriori using a very simple polynomial-time procedure. For the sake of conciseness, we will not present the proof and procedure for the PDBBI. These can easily be deduced from those presented in Hamdouni *et al.*(2004) for the bus disptaching problem by blocks with maneuvers.

The implicit model for the PDBBI is the following integer linear program :

$$\text{Minimize} \quad w_1 \sum_{p \in P_2} X_p + w_2 \sum_{t \in T} \Big(\sum_{k \in I \setminus I^t} E_k^t + \sum_{k \in J \setminus J^t} F_k^t \Big) \tag{5.1}$$

$$\text{s.t.} \quad \sum_{p \in P} X_p = v \tag{5.2}$$

$$\sum_{p \in P} (s_p^t + e_p^t) X_p = b^t, \qquad \forall t \in T \tag{5.3}$$

$$\sum_{p\in P} s_p^t X_p + \sum_{p\in P_2} e_p^t Y_{pi} \geq a_i^t + \sum_{u\in T\setminus\{t\}} \Big(\sum_{\substack{k\in I^u\\ k\leq i}} E_k^t - \sum_{\substack{k\in I^t\\ k\leq i}} E_k^u\Big), \quad \forall t\in T, i\in I \qquad (5.4)$$

$$\sum_{p\in P_2} s_p^t Y_{pi} \leq a_i^t + \sum_{u\in T\setminus\{t\}} \Big(\sum_{\substack{k\in I^u\\ k\leq i}} E_k^t - \sum_{\substack{k\in I^t\\ k\leq i}} E_k^u\Big), \quad \forall t\in T, i\in I \qquad (5.5)$$

$$Y_{pi} \leq Y_{p,i+1}, \quad \forall p\in P_2, i\in I\setminus\{n\} \qquad (5.6)$$

$$Y_{pn} \leq X_p, \quad \forall p\in P_2 \qquad (5.7)$$

$$\sum_{p\in P} s_p^t X_p + \sum_{p\in P_2} e_p^t Z_{pj} \geq d_j^t + \sum_{u\in T\setminus\{t\}} \Big(\sum_{\substack{k\in J^u\\ k\leq j}} F_k^t - \sum_{\substack{k\in J^t\\ k\leq j}} F_k^u\Big), \quad \forall t\in T, j\in J \qquad (5.8)$$

$$\sum_{p\in P_2} s_p^t Z_{pj} \leq d_j^t + \sum_{u\in T\setminus\{t\}} \Big(\sum_{\substack{k\in J^u\\ k\leq j}} F_k^t - \sum_{\substack{k\in J^t\\ k\leq j}} F_k^u\Big), \quad \forall t\in T, j\in J \qquad (5.9)$$

$$Z_{pj} \leq Z_{p,j+1}, \quad \forall p\in P_2, j\in J\setminus\{n\} \qquad (5.10)$$

$$Z_{pn} \leq X_p, \quad \forall p\in P_2 \qquad (5.11)$$

$$E_k^t, F_k^t \in \{0,1\}, \quad \text{for all valid indices} \qquad (5.12)$$

$$X_p, Y_{pi}, Z_{pj} \geq 0, \text{ integers}, \quad \text{for all valid indices.} \qquad (5.13)$$

The objective function (5.1) aims at minimizing a weighted sum of the number of two-block lanes and the number of incompatible arrivals and departures, where w_1 and w_2 are nonnegative weights associated with each part of this sum. Constraint (5.2) ensures that an admissible pattern is associated with each lane. Constraints (5.3) specify that, for each type $t \in T$, the overall number of type t slots according to the selected lane patterns must be equal to the number of buses of type t. Constraints (5.4) ensure that each arrival $i \in I$ can be parked in an accessible parking slot at the time ha_i. The right-hand side of the constraint for $t \in T$ and $i \in I$ indicates the number of buses that must be parked in a type t slot at the time ha_i. Its left-hand side specifies the number of type t slots accessible at time ha_i. If arrival i is of type t', then the constraint for i and $t = t'$ implies that this arrival is incompatible or there is enough space to park it in the type t' exit blocks and the type t' entry blocks whose corresponding exit blocks are filled. The constraints for $t \neq t'$ ensure that there is an accessible parking slot of type t if the arrival i is incompatible and parked in a type t slot. Constraints (5.5) determine the exit blocks that are filled at time ha_i according to the number of buses of each type arrived at that time and the misassignment of the incompatible arrivals. Constraints (5.6) ensure that a filled exit block at time ha_i remains filled at the subsequent arrival times, while constraints (5.7) ensure that the number of exit blocks filled for the pattern $p \in P_2$ after the last arrival does not exceed the number of lanes partitioned according to p. Constraints (5.8)–(5.11) are simply the departure counterparts of constraints (5.4)–(5.7). Finally, nonnegative and integrality requirements on the variables are imposed by constraints (5.12) and (5.13).

Note that, as in Hamdouni *et al.*(2004), the integrality requirements on the X_p variables for $p \in P_2$ and also on all Y_{pi} and Z_{pj} variables can be replaced by binary requirements without eliminating all optimal solutions from the feasible domain. The proof for this result is similar to the one presented in Hamdouni *et al.*(2004) which simply showed that any optimal solution containing two two-block lanes, both with the same types assigned to the exit and entry blocks, can always be transformed into another optimal solution in which all the other lanes are identical while those two lanes are rearranged into either two one-block lanes or one one-block lane and one two-block lane. Based on this result, the size of the feasible domain can be reduced by replacing the constraints (5.13) with the following ones :

$$X_p \geq 0, \text{ integers}, \qquad \forall p \in P \setminus P_2 \tag{5.14}$$

$$X_p, Y_{pi}, Z_{pj} \in \{0,1\}, \qquad \forall p \in P_2, i \in I, j \in J. \tag{5.15}$$

When the assumptions 2–4 made in the introduction do not hold, the model can be adapted in the following way. When lanes have different lengths, a set of admissible patterns can be defined for each group of lanes with the same length and the constraint (5.2) can be replaced by one such constraint for each group. When there are more parking slots than buses, the set of admissible patterns can be enlarged to include patterns that do not completely fill a lane. Finally, when there are buses already parked in the depot before the first bus arrival, constraints forcing the use of partially filled admissible patterns can be added to the model.

5.3 Computational experiments for the PDBBI

For practical instances of the PDBBI, model (5.1)–(5.15) can be solved to optimality in short computational times using a commercial MIP solver, namely CPLEX. To illustrate this, we report in this section computational results that were obtained on instances derived from a real-world dataset provided by Société de Transport de Montréal (STM).

5.3.1 Data description

Typical large bus dispatching problems involve between 100 and 175 buses per depot and 3 to 6 bus types. The STM dataset defines such a problem. It consists of a sequence of 124 arrivals for a given evening and a sequence of 124 departures for the following morning. There are four bus types and a type is specified for each arrival and for each departure. Since $n = 124$ does not have many integer divisors and, by assumption, $n = v\ell$, we randomly removed four arrivals and four departures (of the corresponding types) from the dataset to obtain a total number of 120 buses. This number allows us to study the impact of the depot dimensions on the solution times. Since a lane length (ℓ) rarely exceeds 10, we performed tests with $\ell \in \{6, 8, 10, 12\}$, where $\ell = 12$ is considered an extreme case. This dataset with 120 buses of 4 types is denoted I-120-4, where I stands for incompatibilities only (that is, without considering mismatches). The bus distribution with respect to the 4 types (A, B, C and D) is given on the first line of Table 5.1.

Table 5.1 also presents the distribution of the buses for 4 other datasets originating from the I-120-4 dataset. Dataset I-120-6 contains 120 buses of 6 bus types and was obtained by randomly converting some type A buses into type E buses and some type B buses into type F buses. Dataset I-40-5 (40 buses of 5 types) was constructed from dataset I-120-4 by randomly removing 80 buses and converting some type A buses into type E buses. Removing 8 more buses yielded dataset I-32-5 (32 buses of 5 types) and converting some type B buses into type F buses produced the last dataset I-32-6 (32 buses of 6 types). The I-120-6 dataset was tested for all values of $\ell \in \{6, 8, 10, 12\}$, while the datasets I-40-5, I-32-5 and I-32-6 were tested only for $\ell = 8$. Because the largest datasets did not yield any incompatibility, we created these last three medium-sized datasets which are tighter and yield incompatibilities. Note that such medium-sized problems appear as subproblems of larger problems that involve precedence relationships.

Table 5.2 reports the size of model (5.1)–(5.12), (5.14) and (5.15) for all the instances by displaying its numbers of variables and constraints before and after CPLEX preprocessing. As one can observe, the size of the model increases with the number of buses, the number of types and the value of ℓ.

Tableau 5.1 – Number of buses per type for each dataset

Dataset	Bus type					
	A	B	C	D	E	F
I-120-4	68	36	6	10	-	-
I-120-6	37	21	6	10	31	15
I-40-5	13	12	3	3	9	-
I-32-5	9	10	2	3	8	-
I-32-6	9	6	2	3	8	4

5.3.2 Results for the PDBBI

Computational experiments were conducted on the PDBBI instances described above, where the weights w_1 and w_2 were set to 1 and 5000,respectively, in order to prioritize minimizing incompatibilities. For those instances, the model (5.1)–(5.12), (5.14) and (5.15) was solved on a 440-MHz Sun Ultra 10 workstation using the commercial CPLEX MIP solver (version 8.1). The following CPLEX features were used because they yielded the best results in a series of preliminary tests : the strong branching method, the cutting plane generator used moderately, and the heuristic procedure for finding integer solutions when invoked by CPLEX. Furthermore, branching priorities on the variables were established as follows : highest priority for the one-block pattern X_p variables ($p \in P \setminus \{P_2\}$), second highest priority for the two-block pattern X_p variables ($p \in P_2$) and the lowest priority for all the other variables.

Two series of tests were performed. In the first series, the instances were solved with the complete model. The second series relies on the assumption that the constraints (5.4) for $i \in I \setminus I^t$ and (5.8) for $j \in J \setminus J^t$ have very little chance to be violated when the number of incompatible arrivals and departures is small as it is usually the case in most practical instances. Most of these constraints are, therefore, not necessary and removing them can only speed up the solution process. Since the unnecessary

Tableau 5.2 – Numbers of variables and constraints for each instance

			Before CPLEX preprocessing		After CPLEX preprocessing	
Dataset	ℓ	v	Nb var.	Nb const.	Nb var.	Nb const.
I-120-4	6	20	15199	16325	14944	16085
	8	15	20977	22085	19053	20177
	10	12	26758	27845	21693	22085
	12	10	32522	33605	22898	24005
I-120-6	6	20	37353	38887	36741	38287
	8	15	48556	53287	48190	49687
	10	12	66265	67687	57110	58567
	12	10	80722	82087	63559	64987
I-40-5	8	5	11663	12006	6173	6578
I-32-5	8	4	9359	9570	4507	4822
I-32-6	8	4	13972	14215	5396	5763

constraints cannot be identified a priori, we resorted to constraint generation, that is, all these constraints were relaxed a priori and generated only when they were violated by the linear relaxation solution associated with a node of the branch-and-bound search tree. In this context, we will refer to these constraints as the dynamic constraints (DC).

The results for both series of tests are reported in Table 5.3 which is divided horizontally into four parts. For each instance, the first part describes the instance (dataset, ℓ and v). The second part provides the optimal solution value in terms of the number of incompatible arrivals (IA), the number of incompatible departures (ID), and the number of two-block lanes (2BL). The third part presents, for the tests without DC, the number of branch-and-bound nodes (BB) required (a star means that the optimal solution was found by the CPLEX heuristic procedure) and the total CPU time in seconds. Finally, in addition to these two statistics, the last part also indicates, for the tests with DC, the total number of DC (Tot DC) in the model and the number of DC generated during the solution process (Gen DC).

Tableau 5.3 – Results for the PDBBI instances

						Without DC		With DC			
			Nb	Nb	Nb	Nb	Time	Tot	Gen	Nb	Time
Dataset	ℓ	v	IA	ID	2BL	BB	(s)	DC	DC	BB	(s)
I-120-4	6	20	0	0	1	1*	10	720	0	1	11
	8	15	0	0	3	1*	83	720	54	1*	77
	10	12	0	0	2	1*	79	720	22	1	53
	12	10	0	0	3	87	1172	720	212	67	343
I-120-6	6	20	0	0	3	1*	77	1185	0	1*	73
	8	15	0	0	4	158	3499	1185	68	21*	655
	10	12	0	0	4	71	7996	1185	68	31*	1710
	12	10	0	0	5	107	41818	1185	26	51*	4800
I-40-5	8	5	4	0	4	60	776	320	84	37	399
I-32-5	8	4	2	2	4	106	434	256	107	136	271
I-32-6	8	4	3	5	4	51	544	320	139	129	351

We observe from these results that no incompatible arrivals or departures occur for the larger instances. These results also show that the solution time generally increases with the size of the problem. Furthermore, the constraint generation strategy seems to be quite efficient, especially for the most difficult instances. For example, the total CPU time was reduced by more than 88% for the instance I-120-6 with $\ell = 12$. In fact, for this instance, a very small percentage of the DC was generated. We also observe that, for all I-120-6 instances, the CPLEX heuristic found the optimal solution, thus reducing the size of the search tree and the total solution time. Overall, these results show that all these instances, except one, can be solved in less than 30 minutes.

As we mentioned in the introduction, the number of mismatches is always bounded from above by the sum of the number of incompatible arrivals and the number of incompatible departures. Consequently, notice that no mismatches occur for the large instances (I-120-4 and I-120-6) even though the objective of the PDBBI does not aim at minimizing the number of mismatches. For the other instances, only a small number of mismatches is possible. The rest of this paper will focus on the PDBBM, which takes mismatches into account.

5.4 A mathematical formulation for the PDBBM

The second part of this paper studies the PDBBM which includes the PDBBI as a special case. As mentioned in the introduction, the PDBBM can be obtained from the PDBBI by considering the additional (weighted) objective of minimizing the number of mismatches. With this addition, it is not possible anymore to use an implicit model such as the one proposed in Section 5.2 because, for identifying a mismatch, the exact slot assigned to each arrival and each departure must be known.

In this section, we provide a model for the PDBBM which is an extension of the model proposed for the PDBBI. It requires, however, that all X_p variables be binary even for those associated with the one-block patterns. Therefore, for a one-block pattern $p \in P \setminus P_2$ containing only type t buses, creating $\lfloor b^t/\ell \rfloor$ copies of this pattern in P allows to consider that all X_p variables are binary. Furthermore, the slots in each lane are numbered from 1 to ℓ starting from the exit to the entry. Let $O = \{1, 2, \ldots, \ell\}$ be the set of these numbers.

The following notation is also needed. Let A^i_{op} (resp. D^j_{op}) be a binary variable that takes value 1 if the arrival $i \in I$ (resp. departure $j \in J$) is assigned to slot $o \in O$ of pattern $p \in P$, and 0 otherwise. Let M_{op} be a binary variable that takes value 1 if a mismatch occurs in slot $o \in O$ of pattern $p \in P$, and 0 otherwise. Finally, denote by P^t_{EX} (resp. P^t_{EN}) the set of all patterns whose exit (resp. entry) block is of type $t \in T$.

Using this notation, the PDBBM can formulated as the following integer linear program :

$$\text{Minimize } w_1 \sum_{p \in P_2} X_p + w_2\Big(\lambda \sum_{p \in P} \sum_{o \in O} M_{op} + (1-\lambda) \sum_{t \in T} \big(\sum_{k \in I \setminus I^t} E_k^t + \sum_{k \in J \setminus J^t} F_k^t\big)\Big) \quad (5.16)$$

$$\text{s.t.} \qquad (5.2) - (5.12)$$

$$X_p, Y_{pi}, Z_{pj} \in \{0,1\}, \qquad \text{for all valid indices} \quad (5.17)$$

$$\sum_{p \in P_{\text{EX}}^t} \sum_{o=1}^{s_p^t} A_{op}^i + \sum_{p \in P_{\text{EN}}^t} \sum_{o=s_p^t+1}^{\ell} A_{op}^i = 1 - \sum_{u \in T \setminus \{t\}} E_i^u, \qquad \forall t \in T, i \in I^t \quad (5.18)$$

$$\sum_{p \in P_{\text{EX}}^u} \sum_{o=1}^{s_p^u} A_{op}^i + \sum_{p \in P_{\text{EN}}^u} \sum_{o=s_p^u+1}^{\ell} A_{op}^i = E_i^u, \qquad \forall u \in T, i \in I \setminus I^u \quad (5.19)$$

$$\sum_{i\in I} A^i_{op} \leq X_p, \qquad \forall p \in P, o \in O \tag{5.20}$$

$$A^i_{o+1,p} \leq \sum_{k=1}^{i-1} A^k_{op}, \qquad \forall p \in P, o \in O \setminus \{\ell\}, i \in I \tag{5.21}$$

$$\sum_{p\in P^t_{\text{EX}}} \sum_{o=1}^{s^t_p} D^j_{op} + \sum_{p\in P^t_{\text{EN}}} \sum_{o=s^t_p+1}^{\ell} D^j_{op} = 1 - \sum_{u\in T\setminus\{t\}} F^u_j, \qquad \forall t \in T, j \in J^t \tag{5.22}$$

$$\sum_{p\in P^u_{\text{EX}}} \sum_{o=1}^{s^u_p} D^j_{op} + \sum_{p\in P^u_{\text{EN}}} \sum_{o=s^u_p+1}^{\ell} D^j_{op} = F^u_j, \qquad \forall u \in T, j \in J \setminus J^u \tag{5.23}$$

$$\sum_{j\in J} D^j_{op} \leq X_p, \qquad \forall p \in P, o \in O \tag{5.24}$$

$$D^j_{o+1,p} \leq \sum_{k=1}^{j-1} D^k_{op}, \qquad \forall p \in P, o \in O \setminus \{\ell\}, j \in J \tag{5.25}$$

$$M_{op} \geq \sum_{i\in I^t} A^i_{op} - \sum_{j\in J^t} D^j_{op}, \qquad \forall t \in T, p \in P, o \in O \tag{5.26}$$

$$M_{op} \geq \sum_{j\in J^t} D^j_{op} - \sum_{i\in I^t} A^i_{op}, \qquad \forall t \in T, p \in P, o \in O \tag{5.27}$$

$$A^i_{op}, D^j_{op}, M_{op} \in \{0,1\}, \qquad \text{for all valid indices.} \tag{5.28}$$

The objective function (5.16) minimizes a weighted sum of the number of two-block lanes, the number of mismatches and the number of incompatible arrivals and departures, where w_1 and w_2 are nonnegative weights associated with the first part and the rest of this sum, respectively. The constant λ belongs to the interval $[0,1]$ and defines a convex combination of the number of mismatches and the number of incompatible arrivals and departures. Therefore, setting λ close to 0 puts most of the w_2 weight on the number of incompatible arrivals and departures, while setting it close to 1 assigns a high proportion of w_2 to the number of mismatches.

The constraints (5.2)–(5.12) and (5.17) defines the block pattern structure and determines the incompatible arrivals and departures. Constraints (5.18)–(5.21) ensure a valid assignment of the arrivals to the parking slots. Indeed, constraints (5.18) assign a slot of type t to each arrival $i \in I^t$ that is not an incompatible arrival, while constraints (5.19) assign a slot of type u to each incompatible arrival $i \in I \setminus I^u$ that must be assigned to a type u slot. Constraints (5.20) indicate that no arrival i can be assigned to a slot o unless this slot belongs to a pattern p that is selected for a lane. When such a pattern is selected, they also ensure that a maximum of one arrival is parked in each slot of this pattern. Next, constraints (5.21) force the slots of a selected pattern to be filled in the right order, that is, from the exit to the entry of the lane. Constraints (5.22)–(5.25) are simply the departure variants of the constraints (5.18)–(5.21). The values of the M_{op} variables are computed through the constraints (5.26) and (5.27), coupled with the objective function that aims at minimizing their values. In these constraints, the sum of the A^i_{op} (resp. D^j_{op}) variables is equal to 1 if an arrival (resp. departure) of type t is assigned to the slot o in pattern p and 0 otherwise. Finally, additional binary requirements are imposed by (5.28).

Note that, when the assumptions 2-4 stated in the introduction do not hold, the adaptations for the PDBBI model discussed at the end of Section 5.2 can also be applied to the above PDBBM model.

5.5 Solution methodology for the PDBBM

The PDBBM model (5.16), (5.2)–(5.12) and (5.17)–(5.28) is obviously much larger than the corresponding PDBBI model since it includes it. For example, for the instance I-32-5 with $v = 4$ and $\ell = 8$, the PDBBM model contains 92576 variables and 96966 constraints while the PDBBI model contains 9361 variables and 9606 constraints. We tried solving this instance using CPLEX directly on the PDBBM model. We halted the solution process after 20 hours of computation without finding an optimal solution. Only solving the linear relaxation took more than 2500 seconds. Given this poor performance, we investigated a different solution approach, namely a Benders decomposition method (Benders, 1962; see also Geoffrion and Graves, 1974) where the master problem corresponds to the PDBBI model augmented by Benders cuts and the subproblem is obtained by relaxing the integrality requirements on its variables. To satisfy these requirements, this method is embedded in a branch-and-bound procedure that imposes branching decisions on the master problem variables. The first part of this section presents the proposed Benders decomposition method that produces lower bounds, its second part describes the branching strategy in a general context, while the third part specializes this strategy for the PDBBM.

5.5.1 Benders decomposition

The Benders decomposition principle separates a problem into a master problem and a subproblem by identifying a set of coupling variables. In our case, these variables are the X_p, E_k^t and F_k^t variables. Let $\boldsymbol{X} = (X_p \mid p \in P)$, $\boldsymbol{Y} = (Y_{pi} \mid p \in P, i \in I)$, $\boldsymbol{Z} = (Z_{pj} | p \in P, j \in J)$, $\boldsymbol{E} = (E_k^t | k \in I, t \in T)$, $\boldsymbol{F} = (F_k^t | k \in J, t \in T)$, $\boldsymbol{A} = (A_{op}^i | p \in P, o \in O, i \in I)$, $\boldsymbol{D} = (D_{op}^j \mid p \in P, o \in O, j \in J)$, and $\boldsymbol{M} = (M_{op} \mid p \in P, o \in O)$ be the variable vectors and, for the sake of conciseness, define the compound vectors $\boldsymbol{\chi} = (\boldsymbol{X}, \boldsymbol{E}, \boldsymbol{F})$, $\boldsymbol{\psi} = (\boldsymbol{Y}, \boldsymbol{Z})$ and $\boldsymbol{\pi} = (\boldsymbol{A}, \boldsymbol{D}, \boldsymbol{M})$. Denote by $\mathcal{F}$ the set of vectors $\bar{\boldsymbol{\chi}}$ such that there exists at least one feasible solution to the constraints (5.2)–(5.12) and (5.17) when $\boldsymbol{\chi} = \bar{\boldsymbol{\chi}}$. Setting the coupling variable vectors $\boldsymbol{\chi}$ to specific values $\bar{\boldsymbol{\chi}} \in \mathcal{F}$, we obtain the following discrete subproblem

$$\begin{aligned} z_{\mathrm{SP}}^{\mathrm{IP}}(\bar{\boldsymbol{\chi}}) = \quad & \text{Minimize} \quad \sum_{p \in P} \sum_{o \in O} M_{op} \\ & \text{s.t.} \quad (5.18) - (5.28) \end{aligned} \tag{5.29}$$

whose optimal value $z^{\mathrm{IP}}_{\mathrm{SP}}(\bar{\chi})$ corresponds to the optimal number of mismatches that can be obtained from a given feasible solution of the PDBBI model with $\chi = \bar{\chi}$. This subproblem (and its linear relaxation) is always feasible and bounded because a feasible solution to the PDBBI model guarantees that the arrivals and the departures can be assigned to accessible parking slots of the appropriate type. Therefore, it simply consists of assigning the arrivals and the departures to such slots while minimizing the number of mismatches.

The PDBBM model can thus be rewritten as

$$\text{Minimize} \quad w_1 \sum_{p\in P_2} X_p + w_2\big(\lambda\, z^{\mathrm{IP}}_{\mathrm{SP}}(\chi) + (1-\lambda)\sum_{t\in T}\big(\sum_{k\in I\setminus I^t} E^t_k + \sum_{k\in J\setminus J^u} F^t_k\big)\big) \tag{5.30}$$

$$\text{s.t.} \quad (5.2) - (5.12), (5.17).$$

However, the application of the Benders decomposition principle requires relaxing the integrality requirements (5.28) to obtain the continuous subproblem

$$z^{\mathrm{LP}}_{\mathrm{SP}}(\bar{\chi}) = \text{Minimize} \quad \sum_{p\in P}\sum_{o\in O} M_{op} \tag{5.31}$$

$$\text{s.t.} \quad (5.18) - (5.27)$$

$$0 \le A^i_{op}, D^j_{op}, M_{op} \le 1, \text{ for all valid indices} \tag{5.32}$$

and the relaxed PDBBM model

$$\text{Minimize} \quad w_1 \sum_{p\in P_2} X_p + w_2\big(\lambda\, z^{\mathrm{LP}}_{\mathrm{SP}}(\chi) + (1-\lambda)\sum_{t\in T}\big(\sum_{k\in I\setminus I^t} E^t_k + \sum_{k\in J\setminus J^u} F^t_k\big)\big) \tag{5.33}$$

$$\text{s.t.} \quad (5.2) - (5.12), (5.17).$$

Applying Benders decomposition allows to rewrite this relaxed problem as a master problem whose definition relies on the extreme points and the extreme rays of the dual of the continuous subproblem (5.31), (5.18)-(5.27), and (5.32) for $\chi \in \mathcal{F}$. In our case, there are no extreme rays because this subproblem is always feasible when $\chi \in \mathcal{F}$. Let Γ be the set of extreme points. Denote by $\boldsymbol{\alpha} = (\alpha_{ti} \,|\, t \in T, i \in I^t)$, $\boldsymbol{\beta} = (\beta_{ui} \,|\, u \in T, i \in I \setminus I^u)$, $\boldsymbol{\eta} = (\eta_{po} \,|\, p \in P, o \in O)$, $\boldsymbol{\sigma} = (\sigma_{tj} \,|\, t \in T, j \in J^t)$, $\boldsymbol{\tau} = (\tau_{uj} \,|\, u \in T, j \in J \setminus J^u)$ and $\boldsymbol{\phi} = (\phi_{po} \,|\, p \in P, o \in O)$ the dual variable vectors associated with the subproblem constraints (5.18), (5.19), (5.20), (5.22), (5.23), (5.24), respectively, and by $\boldsymbol{\alpha}^q$, $\boldsymbol{\beta}^q$, $\boldsymbol{\eta}^q$, $\boldsymbol{\sigma}^q$, $\boldsymbol{\tau}^q$, $\boldsymbol{\phi}^q$ the corresponding components of the extreme point $q \in \Gamma$. Finally, denote by U a nonnegative variable that provides the value of the subproblem for any solution $\chi \in \mathcal{F}$.

The Benders master problem is then given by

$$\text{Minimize} \qquad w_1 \sum_{p \in P_2} X_p + w_2\big(\lambda\, U + (1-\lambda) \sum_{t \in T} \big(\sum_{k \in I \setminus I^t} E_k^t + \sum_{k \in J \setminus J^t} F_k^t \big)\big) \tag{5.34}$$

$$\begin{aligned}
\text{s.t.} \qquad & (5.2) - (5.12), (5.17) \\
U \geq\ & \sum_{\substack{t \in T \\ i \in I^t}} \alpha_{ti}^q \big(1 - \sum_{u \in T \setminus \{t\}} E_i^u\big) + \sum_{\substack{u \in T \\ i \in I \setminus I^u}} \beta_{ui}^q E_i^u + \sum_{\substack{o \in O \\ p \in P}} \eta_{op}^q X_p \\
& + \sum_{\substack{t \in T \\ j \in J^t}} \sigma_{tj}^q \big(1 - \sum_{u \in T \setminus \{t\}} F_j^u\big) + \sum_{\substack{u \in T \\ j \in J \setminus J^u}} \tau_{uj}^q F_j^u + \sum_{\substack{o \in O \\ p \in P}} \phi_{op}^q X_p, \qquad \forall q \in \Gamma
\end{aligned} \tag{5.35}$$

$$U \geq\ 0. \tag{5.36}$$

In this formulation, the constraints (5.2)-(5.12) and (5.17) define the feasible domain of a PDBBI problem, while the constraints (5.35) are the Benders optimality cuts which are derived from the objective of the dual of the continuous subproblem. The optimal value of this master problem (possibly including branching decisions) provides a lower bound for the corresponding branch-and-bound node.

Note that this master problem contains an exponential number of constraints, that is, one for each extreme point $q \in \Gamma$. Consequently, it is usually solved by a constraint

Algorithme 5.1 Benders algorithm

```
1: Set $r = 1$.
2: Solve the relaxed master problem RMP$^r$ with $\Gamma^r$ and denote by $(U^r, \chi^r, \psi^r)$ the computed optimal solution.
3: Define the subproblem $SP^r$ by setting $\chi$ to $\chi^r$ and solve it to obtain a primal optimal solution $\pi^r$.
4: Denote by $q^r$ the corresponding dual extreme point and by $z^r$ its cost.
5: si $U^r < z^r$ alors
6:     Set $\Gamma_{r+1} = \Gamma^r \cup \{q^r\}$ and $r = r + 1$. Return to step 2.
7: sinon
8:     Stop. The solution $(U^r, \chi^r, \psi^r)$ is optimal for the master problem and the solution $(\chi^r, \psi^r, \pi^r)$ is optimal for the relaxed PDBBM model.
```

generation algorithm known as the Benders algorithm. Assuming that the master problem is feasible, this iterative algorithm (see Algorithm 1) solves at each iteration, indexed by r, a relaxed master problem (RMP^r) and a continuous subproblem (SP^r). At iteration r, RMP^r is the integer program (5.34), (5.2)-(5.12), (5.17), (5.35) and (5.36) where the set Γ is replaced by a subset of it denoted Γ^r. At the first iteration, Γ^1 is usually empty. The computed optimal solution for RMP^r is denoted (U^r, χ^r, ψ^r). Note that U^r may underestimate the optimal value of the subproblem since only a subset of the optimality cuts (5.35) are considered in RMP^r. Given this solution, SP^r is defined by (5.31), (5.18)-(5.27), and (5.32) for $\chi = \chi^r$. The computed optimal solution for SP^r is denoted π^r. If its value z^r exceeds U^r, there is at least one relaxed optimality cut that is violated by the RMP^r solution, namely, the one associated with the extreme point q^r corresponding to the computed optimal dual solution of SP^r. This extreme point is added to the current extreme point subset Γ^r and another iteration is performed. Otherwise, $z^r = U^r$ and (U^r, χ^r, ψ^r) is an optimal solution for the master problem. The algorithm terminates. The optimal solution to the relaxed PDBBM model is then given by (χ^r, ψ^r, π^r). In this solution, the A^i_{op}, D^j_{op} and M_{op} variables are the only variables that can take non-integer values.

In our experiments, the CPLEX MIP solver and the CPLEX LP solver were used to solve, at each iteration, the relaxed master problem and the subproblem, respectively. The CPLEX MIP settings were as discussed in Section 5.3.2, including the dynamic generation of the constraints (5.4) for $i \in I \setminus I^t$ and (5.8) for $j \in J \setminus J^t$.

5.5.2 Generic branching strategy

To our knowledge, there is not much literature on Benders decomposition when the subproblem should be discrete. Typical solution approaches such as those proposed by Cordeau *et al.*(2000) and Mercier *et al.*(2005) are heuristics that consist of solving the discrete subproblem only once after applying the Benders method with a relaxed subproblem. Recently, Sherali and Fraticelli (2002) developed an exact approach for the case where the subproblem variables are subject to binary requirements. They applied RLT (reformulation-linearization technique) or lift-and-project cuts to solve the discrete subproblem and showed that these cuts can be expressed in terms of the master problem variables. The Benders cuts are thus influenced by these subproblem cuts. The authors showed that they are globally valid, yielding a finitely convergent procedure. Sherali and Fraticelli did not report any computational results.

Branch-and-bound procedures can also be devised to obtain optimal integer solutions. Imposing branching decisions on the subproblem variables, which seems the most natural choice, may however require extensive computational efforts because such decisions have a local impact on the model and generally yield a very large search tree. Furthermore, they may require the generation of additional Benders cuts since the cuts generated in the ancestor nodes may not be tight anymore.

In this paper, we propose to use a branching strategy that imposes decisions on the master problem variables, even if these variables already take integer values. This new approach is particularly well suited for problems in which only a small number of costly binary variables can take a positive value in a master problem solution. In this case, branching decisions on the master problem variables usually have a great impact on the overall solution cost and can yield a very efficient pruning of the branch-and-bound tree. In this section, we introduce this new branching scheme on a generic problem. This scheme will be detailled for the PDBBM in the next section.

Borrowing notation from the previous section, let us consider three vectors of variables : χ (resp. ψ) is the vector of the master problem variables involved (resp. not involved) in the definition of the subproblem, and π is the subproblem variable vector. With these vectors, we associate the cost vectors c_1^{T}, c_2^{T} and c_3^{T}, respectively. Next, we define the following discrete subproblem

$$z_{\mathrm{SP}}^{\mathrm{IP}}(\chi) \quad = \quad \text{Minimize} \qquad c_3^{\mathrm{T}}\,\pi \tag{5.37}$$

$$\text{s.t.} \qquad \pi \in \mathcal{D}^{\mathrm{SP}}(\chi) \tag{5.38}$$

where $\mathcal{D}^{\mathrm{SP}}(\chi)$ denotes its feasible domain which is restricted by integrality requirements and depends on χ.

Given this notation, we formulate the generic model as follows :

$$\text{Minimize} \qquad c_1^{\mathrm{T}}\,\chi + c_2^{\mathrm{T}}\,\psi + \gamma\, z_{\mathrm{SP}}^{\mathrm{IP}}(\chi) \tag{5.39}$$

$$\text{s.t.} \qquad (\chi, \psi) \in \mathcal{D}, \tag{5.40}$$

where $\mathcal{D}$ is the feasible domain for the variables χ and ψ, and γ is a weight that proportionates the cost of the subproblem decisions with that of the master problem decisions. As mentioned in the previous section, Benders decomposition is not applied directly on this model but rather on a relaxed model which is obtained from the generic model by replacing its objective function with

$$\text{Minimize} \qquad c_1^{\mathrm{T}}\,\chi + c_2^{\mathrm{T}}\,\psi + \gamma\, z_{\mathrm{SP}}^{\mathrm{LP}}(\chi). \tag{5.41}$$

In this function, $z_{\text{SP}}^{\text{LP}}(\chi)$ corresponds to the optimal value of the subproblem (5.37)–(5.38) when the integrality requirements are omitted. Applying Benders decomposition on this relaxed problem yields a master problem that involves the variables χ and ψ, as well as an additional variable U for representing the relaxed subproblem optimal value. To simplify notation in the following, we omit this variable because it is not needed.

At each node of the branch-and-bound tree, the Benders algorithm is applied to compute a solution $S_{\text{LP}} = (\chi_{\text{IP}}, \psi_{\text{IP}}, \pi_{\text{LP}})$ (recall that the master problem is an integer program (IP), while the relaxed subproblem is a linear program (LP)) for the relaxed problem. The cost $V_{\text{LP}} = c_1^{\text{T}}\ \chi_{\text{IP}} + c_2^{\text{T}}\ \psi_{\text{IP}} + \gamma\ z_{\text{SP}}^{\text{LP}}(\chi_{\text{IP}})$ of this solution provides a lower bound for the current node. When this solution is integer (that is, all discrete subproblem variables take integer values), V_{LP} is also an upper bound and the search tree can be pruned at this node. Otherwise, we first solve the discrete subproblem (5.37)–(5.38) for $\chi = \chi_{\text{IP}}$ to obtain an integer solution π_{IP} and a corresponding upper bound $V_{\text{IP}} = c_1^{\text{T}}\ \chi_{\text{IP}} + c_2^{\text{T}}\ \psi_{\text{IP}} + \gamma\ z_{\text{SP}}^{\text{IP}}(\chi_{\text{IP}})$. If $z_{\text{SP}}^{\text{IP}}(\chi_{\text{IP}}) = z_{\text{SP}}^{\text{LP}}(\chi_{\text{IP}})$, then V_{IP} is also a lower bound and the tree can also be pruned at this branch-and-bound node. In the next section, we devise a stronger pruning criterion for the PDBBM.

When branching is needed at a node, we create three child nodes as follows. Let Q be a set of indices used to index the components of χ, that is, $\chi = \left(\chi_q\right)_{q \in Q}$. Denote by $\tilde{Q} \subseteq Q$ the subset of these indices q for which $\chi_q = 1$ in χ_{IP} and set $m = |\tilde{Q}|$. The three branch-and-bound nodes are obtained by imposing the following decisions :

$$\sum_{q \in \tilde{Q}} \chi_q = m \quad \text{and} \quad \sum_{q \in Q \setminus \tilde{Q}} \chi_q \geq 1 \tag{5.42}$$

$$\sum_{q \in \tilde{Q}} \chi_q = m \quad \text{and} \quad \sum_{q \in Q \setminus \tilde{Q}} \chi_q = 0 \tag{5.43}$$

$$\sum_{q \in \tilde{Q}} \chi_q \leq m - 1. \tag{5.44}$$

The first decision (5.42) excludes the current solution S_{LP} by forcing to 1 at least one of the variables that takes value 0 in χ_{IP}. Since the variables χ_q, for $q \in \tilde{Q}$, must also remain at 1, this decision produces a significant increase in the lower bound obtained at the child node when the χ variables are the most costly variables in the problem. Note that this condition is often met because it is also sought for designing an efficient Benders decomposition approach. Consequently, there are very high chances that the tree can be pruned at this child node.

The second decision (5.43) does not eliminate the current solution, but restricts the feasible domain in such a way that the corresponding child node can be pruned immediately according to Proposition 6 which is stated below. This proposition requires the following definition.

Definition 1. *Let $\mathcal{S}$ be the set of feasible solutions (χ, ψ) of a Benders master problem at a given branch-and-bound node. We say that this* master problem yields a unique subproblem at this node *if the subproblem is the same for all solutions $(\chi, \psi) \in \mathcal{S}$.*

In particular, this property arises when $\mathcal{S}$ is composed of solutions that all have the same χ value. Decision (5.43) purposely forces the satisfaction of this condition.

Proposition 6. *At a branch-and-bound node, denoted N, the upper bound V_{IP} obtained by solving the discrete subproblem after completing the Benders algorithm is also a lower bound at N when the Benders master problem yields a unique subproblem at this node. The search tree can thus be pruned at N.*

Proof : Denote by IP and LP the generic model (5.39)–(5.40) and the relaxed model (5.40)–(5.41), respectively, both augmented by the branching decisions applicable at node N. Let $\mathcal{S}$ be the set of feasible solutions (χ, ψ) of the Benders master problem associated with node N. If this master problem yields a unique subproblem, then $z_{\text{SP}}^{\text{LP}}(\chi) = \kappa_{\text{LP}}$ and $z_{\text{SP}}^{\text{IP}}(\chi) = \kappa_{\text{IP}}$ for all feasible solutions in $\mathcal{S}$. In this case, the objective functions (5.39) and (5.41) of IP and LP can be rewritten as

$$\text{Minimize} \qquad c_1^{\text{T}}\,\chi + c_2^{\text{T}}\,\psi + \gamma\,\kappa^{\text{IP}} \tag{5.45}$$

and

$$\text{Minimize} \qquad c_1^{\text{T}}\,\chi + c_2^{\text{T}}\,\psi + \gamma\,\kappa^{\text{LP}}, \tag{5.46}$$

respectively. Taking out the constant terms ($\gamma\ \kappa_{\text{IP}}$ and $\gamma\ \kappa_{\text{LP}}$) from these functions, one can observe that the resulting problems are identical, which means that an optimal solution for LP is also optimal for IP. Consequently, the master problem solution computed by the Benders decomposition algorithm, which is an optimal solution for LP, is also optimal for IP. The cost of this solution in IP is V_{IP} and provides a lower bound at node N since it is the optimal value of IP. □

The third decision (5.44) eliminates the current solution S_{LP} by fixing at 0 at least one of the variables that take a positive value in χ_{IP}. In this case, it is difficult to evaluate the impact of this decision on the value of the child node lower bound. However, in typical applications solved by Benders decomposition, the number of master problem solutions that have a cost similar to the cost of a (globally) optimal master problem solution is often relatively small. Hence, we can expect that, after a few decisions of this type, the lower bound will increase sufficiently for allowing the pruning of the corresponding branch.

To summarize our discussion, we venture to say that, when the χ variables are the most costly variables in the problem and there are not too many very good master problem solutions, the proposed branching strategy should yield a rather slim and shallow branch-and-bound tree. Indeed, under those assumptions, child nodes derived from the first decision type have a very high probability of being pruned immediately, those created by the second decision type are immediately pruned, and, finally, the probability that a node obtained from the third decision type can be pruned increases rapdily with the depth of the node. In Section 5.6, we present computational results that support this claim.

5.5.3 Specialization for the PDBBM

The branching decisions presented in the previous subsection can be expressed as follows for the PDBBM. Let IT (resp. JT) be the set of pairs of indices (i,t) (resp. (j,t)) such that a variable E_i^t (resp. F_j^t) is defined. Furthermore, let $\tilde{P} \subseteq P$, $\tilde{IT} \subseteq IT$ and $\tilde{JT} \subseteq JT$ be the subsets of indices p, pairs of indices (i,t) and pairs of indices (j,t), respectively, such that $X_p = 1$, $E_i^t = 1$ and $F_j^t = 1$ in χ_{IP}. Finally, let $m = |\tilde{P}| + |\tilde{IT}| + |\tilde{JT}|$ be the total number of such variables. The three branching decisions (5.42)–(5.44) correspond to

$$\sum_{p \in \tilde{P}} X_p + \sum_{(i,t)\in \tilde{IT}} E_i^t + \sum_{(j,t)\in \tilde{JT}} F_j^t = m \quad \text{and} \quad \sum_{(i,t)\in IT\setminus\tilde{IT}} E_i^t + \sum_{(j,t)\in JT\setminus\tilde{JT}} F_j^t \geq 1 \tag{5.47}$$

$$\sum_{p \in \tilde{P}} X_p + \sum_{(i,t)\in \tilde{IT}} E_i^t + \sum_{(j,t)\in \tilde{JT}} F_j^t = m \quad \text{and} \quad \sum_{(i,t)\in IT\setminus\tilde{IT}} E_i^t + \sum_{(j,t)\in JT\setminus\tilde{JT}} F_j^t = 0 \tag{5.48}$$

$$\sum_{p \in \tilde{P}} X_p + \sum_{(i,t)\in \tilde{IT}} E_i^t + \sum_{(j,t)\in \tilde{JT}} F_j^t \leq m - 1. \tag{5.49}$$

Note that, compared to (5.42)–(5.43), the second parts of decisions (5.47)–(5.48) are simplified because $\sum_{p\in\tilde{P}} X_p = v$ and, consequently, $\sum_{p\in P\setminus\tilde{P}} X_p = 0$.

As mentioned in Section 5.5.2, the search tree can be pruned at a node when $V_{\text{LP}} = V^{\text{IP}}$ at this node. Hereafter, we establish a stronger pruning criterion which is applicable for the PDBBM. At each branch-and-bound node, a lower bound V_{LP} and an upper bound V_{IP} are computed. This upper bound is derived from an integer solution $S_1 = (\boldsymbol{\chi}_{\text{IP}}, \boldsymbol{\psi}_{\text{IP}}, \boldsymbol{\pi}_{\text{IP}})$. Denote by n^{B}_{IP}, n^{I}_{IP} and n^{M}_{LP} the number of two-block lanes, the number of incompatible arrivals and departures, and the number of mismatches that it yields, respectively. Assume that there exists another feasible integer solution S_2 yielding an upper bound smaller than V_{IP}. Denote by $n^{\text{B}}_{\text{IP}} + \Delta B$, $n^{\text{I}}_{\text{IP}} + \Delta I$ and $n^{\text{M}}_{\text{IP}} + \Delta M$ its number of two-block lanes, its number of incompatible arrivals and departures, and its number of mismatches, respectively. The values ΔB, ΔI and ΔM simply express the differences between these numbers and those obtained for S_1. It is easy to see that these values must satisfy the following conditions :

$$0 \le n^{\text{B}}_{\text{IP}} + \Delta B \le v \;\Rightarrow\; -n^{\text{B}}_{\text{IP}} \le \Delta B \le v - n^{\text{B}}_{\text{IP}} \tag{5.50}$$

$$n^{\text{I}}_{\text{IP}} + \Delta I \ge 0 \;\Rightarrow\; -n^{\text{I}}_{\text{IP}} \le \Delta I \tag{5.51}$$

$$n^{\text{M}}_{\text{IP}} + \Delta M \ge 0 \;\Rightarrow\; -n^{\text{M}}_{\text{IP}} \le \Delta M. \tag{5.52}$$

In addition, the following condition must hold for S_2 : its cost $w_1(n^{\text{B}}_{\text{IP}} + \Delta B) + w_2\big(\lambda(n^{\text{M}}_{\text{IP}} + \Delta M) + (1-\lambda)(n^{\text{I}}_{\text{IP}} + \Delta I)\big)$ must fall within the semi-open interval $[V_{\text{LP}}, V_{\text{IP}}[$. This condition translates to

$$-\frac{w_1}{w_2\lambda}\Delta B - \frac{(1-\lambda)}{\lambda}\Delta I - n^{\text{M}}_{\text{IP}} + n^{\text{M}}_{\text{LP}} \le \Delta M < -\frac{(1-\lambda)}{\lambda}\Delta I - \frac{w_1}{w_2\lambda}\Delta B,$$

which can be relaxed to

$$-\frac{w_1}{w_2\lambda}(v - n^{\text{B}}_{\text{IP}}) - \frac{(1-\lambda)}{\lambda}\Delta I - n^{\text{M}}_{\text{IP}} + n^{\text{M}}_{\text{LP}} \le \Delta M < -\frac{(1-\lambda)}{\lambda}\Delta I + \frac{w_1}{w_2\lambda}n^{\text{B}}_{\text{IP}} \tag{5.53}$$

using the above conditions (5.50) on ΔB. Moreover, as mentioned earlier, the number of mismatches in a solution is always bounded above by its total number of incompatible arrivals and departures, that is,

$$n^{\text{M}}_{\text{IP}} + \Delta M \le n^{\text{I}}_{\text{IP}} + \Delta I. \tag{5.54}$$

Combining conditions (5.51)–(5.54) with integrality requirements on ΔI and ΔM defines a feasible region $\mathcal{R}$ for ΔI and ΔM. The solution S_2 can only exist if $\mathcal{R}$ contains a point $(\Delta I, \Delta M) \neq (0,0)$. Otherwise, the cost of solution S_1 is also a lower bound at the current branch-and-bound node. The search tree can thus be pruned at this node.

Note that this pruning criterion ($\mathcal{R}$ does not contain a point $(\Delta I, \Delta M) \neq (0,0)$) can easily be verified. Indeed, the conditions (5.51)–(5.54) define a quadrilateral in the $(\Delta I, \Delta M)$ space from which we can deduce that ΔI must belong to the interval $\mathcal{I} = [max\{-n^{\mathrm{I}}_{\mathrm{IP}}, \lambda(n^{\mathrm{M}}_{\mathrm{LP}} - n^{\mathrm{I}}_{\mathrm{IP}}) - \frac{w_1}{w_2}(v - n^{\mathrm{B}}_{\mathrm{IP}})\}, \frac{\lambda}{(1-\lambda)} n^{\mathrm{M}}_{\mathrm{IP}} + \frac{w_1}{w_2(1-\lambda)} n^{\mathrm{B}}_{\mathrm{IP}}]$. Consequently, one can verify the pruning criterion by checking the conditions for each integer value of ΔI in $\mathcal{I}$, one value at a time. When robustness is a major concern (i.e., λ is relatively small) and the minimization of the number of two-block lanes is a secondary objective as stated in the introduction (i.e., w_1 is much smaller than w_2), the feasible region $\mathcal{R}$ and the interval $\mathcal{I}$ are rather small in practice because the numbers of two-block lanes, mismatches, and incompatibilities are also small. As reported in Section 5.6, the pruning criterion is often met in this case.

5.6 Computational experiments for the PDBBM

To evaluate the efficiency of the solution methodology proposed in the previous section, we performed a series of computational experiments using instances described in Section 5.3.1. The tests were carried out on the instances with 40 buses, 5 types and 5 lanes, 32 buses, 5 types and 4 lanes, and 32 buses, 6 types and 4 lanes. These instances will be denoted M-40-5, M-32-5 and M-32-6, respectively, where the letter M refers to the dispatching problem with mismatches. Note that no tests were conducted for the other instances because the results of Section 5.3.2 showed that these instances yielded no incompatible arrivals or departures and, consequently, no mismatches. All results presented in this section were also obtained on a 440-MHz Sun Ultra 10 workstation. Except where otherwise stated, the PDBBM model parameters were set to the following values : $w_1 = 1$, $w_2 = 5000$ and $\lambda = 0.1$. These values indicate that robustness has a very high priority in the multi-criteria objective.

For each instance, Table 5.4 presents on its first three lines the results obtained at the root node of the search tree : the number of incompatible arrivals (IA), the number of incompatible departures (ID), the number of two-block lanes (2BL), the number of mismatches for the continuous subproblem (M-LP), the number of mismatches for the discrete subproblem (M-IP), the number of Benders iterations performed (Iter), and the total CPU time in seconds. From these results, we observe that the number of iterations is very low for the instances M-32-6 and M-32-5, yielding solution times of less than one hour. The number of iterations is much higher for the instance M-40-5 as well as its solution time which exceeds 13 hours. For all instances, most of the time, that is, more than 90%, is spent solving the master problem (the solution time breakdowns are not reported). In all cases, we can notice that the number of mismatches derived from solving the discrete subproblem is equal to the total number of incompatible arrivals and departures. Therefore, each incompatibility produces one mismatch. On the other hand, we observe that there is a positive gap between the number of mismatches obtained with the continuous subproblem and that computed with the discrete subproblem. Therefore, we cannot directly conclude that these integer solutions are optimal.

Applying the pruning criterion proposed in Section 5.5.3, we can confirm that the integer solutions for the instances M-40-5 and M-32-6 are indeed optimal. For the M-32-5 instance, we obtain that $\mathcal{R} = \{(0,0),(0,-1),(0,-2)\}$ where the first and second components of these points are the ΔI- and ΔM-components. This indicates that there can only exist a better integer solution if the total number of incompatible

Tableau 5.4 – Results for the PDBBM instances

Dataset	ℓ	v	BB Node	Nb IA	Nb ID	Nb 2BL	Nb M-LP	Nb M-IP	Nb Iter	Time (s)
M-40-5	8	5	root	2	2	5	3.2	4	40	49969
M-32-6	8	4	root	3	5	4	7.33	8	7	3079
M-32-5	8	4	root	2	2	4	2	4	6	2133
M-32-5	8	4	first	2	2	4	3	3	4	1288

arrivals and departures remains equal to 4, that is, $\Delta I = 0$. The search for such a solution, if it exists, requires the exploration of a branch-and-bound tree using the branching decisions described in Section 5.5.3. For the instance M-32-5, three child nodes are appended to the root node. The node obtained by the addition of decision (5.48) does not need any evaluation since the corresponding master problem as a unique subproblem. The cost of its best solution is therefore the same as that of the root node, that is, $U_{\text{IP}} = 1*4+5000*(1-0.1)*(2+2)+5000*0.1*4 = 20004$. The node derived from decision (5.47) neither requires to be evaluated since this decision implies that $\Delta I > 0$. Consequently, there is a single node to evaluate, namely the one resulting from decision (5.49). Using again the Benders decomposition method to derive a lower bound at this node and solving afterwards the subproblem with integrality requirements, we obtain the results presented on the last line of Table 5.4. Since there is no integrality gap for the subproblem, no further branching is needed. The cost of the computed solution at this node is $U_{\text{IP}} = 1*4+5000*(1-0.1)*(2+2)+5000*0.1*3 = 19504$. This solution is thus an optimal solution for the M-32-5 instance and was computed in 3421 seconds overall.

To complete our computational experiments, we performed a sensitivity analysis on the value of λ for the instance M-32-5, only at the root node of the search tree. The results are reported in Table 5.5. Recall that increasing the value of λ puts more emphasis on the minimization of the number of mismatches compared to the minimization of the number of incompatible arrivals and departures. As one can observe, the solution exhibits the same characteristics (numbers of incompatibilities, two-block lanes and mismatches) for all values of λ smaller than or equal to 0.5. For higher values (that is, when the emphasis is put on minimizing the number of mismatches instead of minimizing the number of incompatibilities), the number of mismatches is smaller while the number of incompatibilities is greater. As for the solution time, it is more or less the same for values of λ smaller than 0.5. At that point, it increases drastically. This increase is not surprising since the proposed solution methodology is suited for problems where minimizing the number of incompatible arrivals and departures is the highest priority. Indeed, the objective of the Benders master problem is solely concerned with this objective until Benders optimality cuts provide additional information on the minimization of the number of mismatches. Consequently, when this second objective becomes as important or more important than the first one $\lambda \geq 0.5$, a large number of Benders iterations are required to transfer the appropriate information from the subproblem to the master problem before reaching the overall optimality.

5.7 Conclusions

In this paper, we studied the PDBBM and a special case of it, the PDBBI. We showed that medium- and large-sized PDBBI instances derived from a real-world dataset can easily be solved using the CPLEX MIP commercial solver. Then, we proposed to solve the PDBBM using a Benders decomposition approach embedded in a branch-and-bound procedure for handling integrality requirements on the subproblem variables. Of particular interests, we devised strong search tree pruning criteria and an innovative branching strategy that imposes decisions on the Benders master problem

Tableau 5.5 – Sensitivity analysis on the value of λ for the M-32-5 instance (root node only)

λ	Nb IA	Nb ID	Nb 2BL	NB M-LP	Nb M-IP	Nb Iter	Time (s)
0.1	2	2	4	2	4	6	2133
0.2	2	2	4	2	4	4	1367
0.3	2	2	4	2	4	5	1677
0.4	2	2	4	2	4	7	3157
0.5	2	2	4	2	4	48	30733
0.6	3	3	3	0.4	2	83	67145
0.7	3	3	3	0.4	2	76	68961

variables, even if these variables already take integer values. These decisions are designed to rapidly meet one of the pruning criteria at most of the search tree nodes. Finally, we reported computational results on medium-sized instances of the PDBBM to illustrate the efficiency of the proposed solution approach.

References

BENDERS, J.F.(1962). Partitioning Procedures for Solving Mixed-Variables Programming Problems. *Numerische Mathematik* **4**, 238–252.

BLASUM, U., BUSSIECK, M.R., HOCHSTÄTTLER, W., MOLL, C., SCHEEL, H.-H. AND WINTER, T. (1999). Scheduling Trams in the Morning. *Mathematical Methods of Operations Research* **49**, 137–148.

CORDEAU, J.-F., SOUMIS, F., AND DESROSIERS, J. (2000). A Benders Decomposition Approach for the Locomotive and Car Assignment Problem. *Transportation*

Science **34**, 133–149.

Desaulniers, G. and Hickman, M.D.(2003). Public Transit Technical Report, Les Cahiers du GERAD G-2003-77, HEC Montreal, Montreal, Canada . To appear in Handbooks in Operations Research and Management Science, Volume on Transportation, C. Barnhart and G. Laporte (eds.), Elsevier, Amsterdam, The Netherlands.

Freling, R., Lentink, R.M., Kroon, L.G., and Huisman, D. (2005). Shunting of Passenger Train Units in a Railway Station. *Transportation Science* **39**, 261–272.

Gallo, G. and Di Miele, F. (2001). Dispatching Buses in Parking Depots. *Transportation Science* **35**, 322–330.

Geoffrion, A.M. and Graves, G.W. (1974). Multicommodity Distribution System Design by Benders Decomposition. *Management Science* **20**, 822–844.

Hamdouni, M., Desaulniers, G., Marcotte, O., Soumis, F. and Van Putten, M. (2004). Dispatching Buses in a Depot using Block Patterns. Technical Report, Les Cahiers du GERAD G-2004-51, HEC Montreal, Montreal, Canada (2004). To appear in *Transportation Science.*

Mercier, A., Cordeau, J.-F. and Soumis, F (2005). A Computational Study of Benders Decomposition for the Integrated Aircraft Routing and Crew Scheduling Problem. *Computers & Operations Research* **32**, 1451–1476.

Sherali, H.D. and Fraticelli, B.M.P (2002). A Modification of Benders' Decomposition Algorithm for Discrete Subproblems : An Approach for Stochastic Programs with Integer Recourse. *Journal of Global Optimization* **22**, 319–342.

Winter, T. and Zimmermann, U.T. (2000). Real-time dispatch of trams in storage yards. *Annals of Operations Research* **96**, 287–315.

CHAPITRE 6 : PARKING BUSES IN A DEPOT WITH STOCHASTIC ARRIVAL TIMES

Ce chapitre décrit le problème de stationnement par bloc en présence de variations dans l'ordre d'arrivée, l'objectif est de produire des solutions planifiées encore plus robustes. Nous introduisons deux approches pour traiter les changements dans l'ordre d'arrivée prévu. Une première approche nommée *k-position* (k est un entier positif) considère que chaque arrivée peut s'écarter d'au plus k positions (en avance ou en retard) de l'ordre planifié. Une deuxième approche stochastique considère que les heures d'arrivées des autobus s'écartent aléatoirement de heures prévues (aléatoires avec une distribution connue). Ces deux approches permettent d'identifier des solutions plus robustes pour le problème de stationnement par bloc.

Dans la première partie de ce chapitre, nous étudions l'approche *k-position* qui permet d'identifier des solutions planifiées qui restent réalisables même si des arrivées dévient d'au plus k positions par rapport à l'ordre prévu. Pour modéliser la variante k-position du problème de stationnement par bloc, nous modifions légèrement la formulation proposée au chapitre 2 pour la variante déterministe du problème. Nous avons constaté que l'approche k-position a certains inconvénients. D'une part, il est difficile de déterminer pour chaque instance la valeur maximale de k pour laquelle le problème est réalisable. L'utilisation de cette valeur est souhaitable parce que l'approche k-position produit des solutions beaucoup plus robustes pour une telle valeur. D'autre part, le paramètre k ne dépend pas du type d'autobus. Il peut donc être plus contraignant pour certains types d'autobus que pour d'autres. Finalement, le paramètre k ne dépend pas des heures d'arrivée alors qu'il devrait en dépendre car une arrivée ne peut s'écarter de k positions de son ordre prévu que si son heure d'arrivée est proche des heures d'arrivée des autobus qui la précèdent et de celles qui la suivent.

Afin de remédier à ces inconvénients, nous introduisons une deuxième approche, dite stochastique, qui considère l'arrivée stochastique des autobus. Cette approche ne dépend d'aucun paramètre k. L'approche stochastique considère les probabilités d'avoir m arrivées de plus ou de moins que prévues après chaque arrivée. Ces probabilités peuvent être estimées par simulation. Nous présentons d'abord la façon de calculer ces probabilités. Ensuite, nous proposons un modèle stochastique linéaire en nombres entiers qui minimise l'espérance d'une fonction corrélée avec le nombre d'autobus qui rendent la solution planifiée non réalisable soit parce qu'ils arrivent trop tard ou trop tôt par rapport à l'heure prévue. Les modèles présentés pour les deux approches ont été résolus avec CPLEX. Les deux approches sont testées sur des jeux de données provenant de la STM. Les résultats numériques montrent que l'approche stochastique performe mieux que l'approche $k-$position lorsque les problèmes sont serrés et requièrent donc des solutions robustes.

Parking buses in a depot with stochastic arrival times

Mohamed Hamdouni, François Soumis, and Guy Desaulniers

Department of mathematics and industrial engineering

École Polytechnique and GERAD

Montréal, Québec, Canada

{Mohamed.Hamdouni, Francois.Soumis, Guy.Desaulniers}@gerad.ca

Abstract

Given buses of different types arriving at a depot during the evening, the bus parking problem consists of assigning these buses to parking slots in such a way that they can be dispatched adequately to the next morning routes without moving them between their arrivals and departures. In practice, the bus arrival times deviate stochastically from the planned schedule. In this paper, we introduce for this problem two solution approaches that produce solutions which are robust to variations in the arrival times. The first approach considers that each arrival can deviate from its planned arrival order (sooner or later) by at most k positions, where k is a predefined parameter. In the second approach, the objective aims at minimizing the expectation of a function positively correlated with the number of buses that make the planned solution infeasible because they arrive too late or too early. In both approaches, the problem is modeled as an integer linear program that can be solved by a commercial MIP solver. Computational results obtained on instances derived from a real-world dataset are reported.

6.1 Introduction

In this paper, we consider a stochastic version of the problem of parking and dispatching buses in a transit authority depot. In this problem which is called in the literature the *bus dispatching problem* (BDP), buses of different types arrive at a depot in the evening and must be parked immediately upon their arrival in such a way that, during the next morning, they can be dispatched directly (that is, without moving other buses to clear the way out for a bus) to departure routes, each of them requesting a specific bus type. In the deterministic version of this problem, the bus arrival times and the morning route departure times are known precisely, while in the stochastic version, the bus arrival times have a certain level of uncertainty.

A typical bus depot is composed of lanes that operate as queues because backing up is forbidden for security reasons. Therefore, each lane has one entry point and one exit point. Throughout the paper, we will assume that i) the lanes have all the same length, ii) the buses have all the same length, iii) the depot is empty before the first arrival, iv) the number of arrivals and the number of morning routes are both equal to the overall number of parking slots in the depot, and v) all lanes can be accessed or left independently of the other lanes (that is, no lanes can block the entry or exit point of another lane). The lanes can thus be divided into the same number of individual parking slots. Note that the models presented in this paper can easily be adapted when one or several of the first four assumptions do not hold (see Hamdouni *et al.*, 2005). On the other hand, the fifth assumption cannot be lifted easily.

In a deterministic context, the BDP can be formally defined as follows. Let T be the set of bus types, b^t the number of buses of type $t \in T$, v the number of lanes, $K =$

$\{1, 2, \ldots, v\}$ the set of lanes, ℓ the number of slots per lane, $L = \{1, 2, \ldots, \ell\}$ the set of slots in each lane, and $I = \{1, 2, \ldots, n\}$ (resp. $J = \{1, 2, \ldots, n\}$) the set of indices indicating the order of the arrivals (resp. morning routes), where $n = v\ell = \sum_{t \in T} b^t$ denotes the total number of buses. With each arrival index $i \in I$ (resp. route index $j \in J$), associate a bus type $t_i^A \in T$ (resp. $t_j^D \in T$) and a planned arrival time h_i^A (resp. departure time h_j^D) such that $h_i^A < h_{i+1}^A$ (resp. $h_j^D < h_{j+1}^D$) for each index $i \in I \setminus \{n\}$ (resp. $j \in J \setminus \{n\}$). The ordered sequences of pairs $s^A = \{(t_i^A, h_i^A)\}_{i \in I}$ and $s^D = \{(t_j^D, h_j^D)\}_{j \in J}$ are called the *planned arrival and departure scenarios*, respectively.

The BDP consists of assigning to each parking slot $l \in L$ of each lane $k \in K$ an arrival index $i_{kl} \in I$ and a morning route index $j_{kl} \in J$ such that

$$\begin{aligned} h^A_{i_{kl}} &< h^A_{i_{kl'}}, && \forall\, k \in K,\ (l, l') \in L \times L \text{ such that } l < l' \\ h^D_{j_{kl}} &< h^D_{j_{kl'}}, && \forall\, k \in K,\ (l, l') \in L \times L \text{ such that } l < l' \\ t^A_{i_{kl}} &= t^D_{j_{kl}}, && \forall\, k \in K,\ l \in L. \end{aligned}$$

During the operations, several factors can modify the planned bus schedules, including heavy traffic congestion that can delay certain arrivals and a smaller than expected number of passengers on the last trip of a bus route that can yield an early arrival. Therefore, buses often arrive at the depot in an order that differs from the order of the planned arrival scenario s^A, yielding a different *arrival scenario*. Such variations in the arrival sequence may result in an impossibility to operate the planned solution. For example, let us consider an instance of the BDPB that involves six buses of three different types A, B and C, and a depot containing two lanes with three slots each. Assuming that the planned arrival scenario is $ABBCAB$, a feasible planned solution assigns the first, fifth and sixth arrivals (AAB) to the first lane and the other arrivals (BBC) to the second lane. Now, assuming that all buses respect their schedule except the third arrival which arrives after all the others, then the arrival scenario is $ABCABB$. In this case, the planned solution is not feasible for this arrival scenario and the solution must be modified in real time.

To reduce the impact of such variations in the arrival sequence on the operability of a planned solution, Hamdouni *et al.*(2004) proposed to restrict the set of admissible solutions to those that partition each lane into at most two blocks of slots, each block containing buses of the same type. More formally, this restriction can be stated as : for each lane $k \in K$, there exists at most one slot $l \in L \setminus \{\ell\}$ such that $t^A_{i_{kl}} \neq t^A_{i_{k,l+1}}$. The lanes for which no such slot exists are called *one-block* lanes, while those for which one such slot exists are called *two-block* lanes. This restricted problem, which is called the *bus dispatching problem by blocks* (BDPB), yields solutions that are more robust to changes in the arrival sequence and also easier to implement in practice (see Hamdouni *et al.*, 2004).

In this paper, we pursue this direction of computing robust solutions by proposing two variants of the BDPB. In the first variant, the set of admissible solutions is further restricted to solutions that remain feasible even if, during the operations, any bus is shifted forward or backward in an arrival scenario by at most k positions with respect to its position in the planned arrival scenario. In the second variant, the arrival times are considered as random variables with known probability distributions and the objective consists of minimizing the expectation of a function positively correlated with the number of buses that make the planned solution infeasible because they arrive too late or too early. In both variants, we consider that the (planned) departure scenario is deterministic, that is, it cannot change. Both variants are modeled as integer programs and solved using a commercial mixed integer programming software, namely CPLEX. These robust approaches are called the *k-position approach* and the *stochastic approach*, respectively. They have different advantages and disadvantages that will be exposed later on.

All previous works on the BDP have considered a deterministic context and proposed approaches that allow the recourse to one of two alternatives when an instance of the BDP as stated above is infeasible for the planned arrival scenario. The first alternative allows to maneuver (reposition) the buses during the night, while the second simply accepts that a morning route be serviced by a type different from the one requested (such assignment is called a *type mismatch*). These alternatives are current practice in certain depots where feasibility is harder to reach. They have attracted the researchers' attention because they increase the difficulty of solving the problem. However, it seems that a large number of the real-world BDP instances are feasible, especially when the number of lanes is relatively high and the number of slots per lane relatively low. In this paper, we focus on these instances and aim at increasing the chances that the computed solution remains feasible for the most probable arrival scenarios.

This paper is organized as follows. First, we review in Section 6.2 the literature on the BDP and related problems. Then, in Section 6.3, we describe the k-position approach while, in Section 6.4, we introduce the stochastic approach. Computational results for both approaches obtained on instances generated from a real-world dataset are reported in Section 6.5. Concluding remarks are presented in Section 6.6.

6.2 Literature review

The BDP has been addressed only in a few recent papers, while a closely related problem, the tram dispatching problem (TDP), has been studied in a few papers. The TDP differs from the BDP on the following aspects. In the TDP, the lanes are seen as stacks instead of queues. Moreover, when maneuvers (shunting moves) are allowed, each individual tram maneuver in the TDP yields a cost while, in the BDP, one or several maneuvers in the same lane costs the same because all the buses in that lane must be moved out of the depot in all cases. Finally, typical BDP instances usually involved much more vehicles than typical TDP instances.

Blasum *et al.*(1999) introduced the TDP. In fact, they studied a subproblem of the TDP where the trams are already parked in the depot and must be dispatched to morning departures. They showed that this subproblem is NP-complete and proposed one exact and two heuristic dynamic programming algorithms to compute a feasible solution when one exists. Winter and Zimmermann (2000) considered four variants of the TDP, including one where the objective is to minimize the number of individual maneuvers and another one with the objective of minimizing the number of type mismatches. They showed that these two problems are NP-hard and proposed an exact solution approach for each of them. The approach for minimizing maneuvers uses a quadratic assignment model that can be linearized to yield a large integer program, while the approach for minimizing mismatches relies directly on a integer linear programming model. Since both models cannot be solved easily by a MIP solver, they also developed efficient heuristic procedures. In the same paper, the authors studied the TDP in real time, that is, when the trams are arriving at the depot in an order that might differ from the planned one and must be dispatched upon their arrivals. For this problem, they proposed different approaches based on the planning models that consist of fixing tram assignments from the planned solution. They also developed a last-in, first-out heuristic. Very recently, Freling *et al.*(2005) tackled a train dispatching problem where the trains can be broken up into units before dispatching them. For this difficult problem, they devised a heuristic two-step approach based on a binary linear program and a column generation method.

In 2001, Gallo *et al.*presented an integer linear program for the BDP with maneuvers and mentioned that it can easily be modified for the case with type mismatches. They used a heuristic Lagrangean decomposition approach for solving it. More recently, Hamdouni *et al.*(2004) introduced a certain level of solution robustness in this family of problems by defining the BDPB with maneuvers. They showed that the BDPB is also NP-complete and proposed two integer linear programming formulations for two variants of the problem that can be solved in reasonable times using a commercial MIP solver. The formulations presented in this paper are based on Hamdouni *et al.*(2004) formulations.

Very recently, Hamdouni *et al.*(2005) addressed the BDPB with type mismatches. They formulated this problem as a large-scale integer linear program and developed for solving it an exact Benders decomposition approach in which the subproblem is subject to integrality requirements. For handling these requirements, they designed a specialized branch-and-bound scheme.

In this paper, we seek additional solution robustness in the case where no maneuvers or type mismatches are needed. For this purpose, we introduce two different approaches which are described in the next two sections.

6.3 The k-position approach

This section presents a robust approach for the BDPB, called the k-position approach, which produces a planned solution that will remain feasible during the operations when any bus in an arrival scenario is shifted by at most k positions with respect to its position in the planned arrival scenario. Let S be the set of all possible arrival scenarios, that is, all possible permutations of the arrivals. For a scenario $s \in S$, denote by $\sigma_s(i)$, $i \in I$, the position in s of the i^{th} arrival in the planned arrival scenario s^A. Now, for a nonnegative integer k, let us define S^k the subset of scenarios ($S^k \subseteq S$) in which any bus is shifted by at most k positions with respect to the planned arrival scenario, that is, $s \in S^k$ if and only if

$$i - k \leq \sigma_s(i) \leq i + k, \qquad \forall\, i \in I.$$

Consequently, the k-position variant of the BDPB consists of finding a solution to the BDPB such that this solution is feasible for all arrival scenarios in S^k. Obviously, for large values of k, such a solution might not exist, yielding an infeasible problem. To model this problem variant, we propose to modify the formulation developed by Hamdouni *et al.*(2004) for the BDPB with maneuvers. This formulation is implicit because it does not assign each arrival and each departure to a specific slot in a specific lane, but it ensures that such an assignment is feasible. As discussed below, the detailed assignment solution can be computed a posteriori in polynomial time.

The proposed model is based on the enumeration of all admissible lane patterns. An *admissible lane pattern* is a partition of a lane into at most two blocks, an *entry* block (the closest to the lane entry) and an *exit* block, which specifies the number of slots in each block and their assigned type. During the arrivals (resp. departures), the exit block of each lane must be filled (resp. emptied) before its entry block. When a pattern contains a single block, it is considered an exit block. For example, if A and B are two bus types and a lane contains five slots, then $AAAAA$ constitutes a one-block pattern while $AABBB$ is a two-block pattern with two buses of type A in the exit block and three buses of type B in the entry block.

Let P be the set of all admissible patterns and P_2 the set of two-block patterns ($P_2 \subset P$). For each pattern $p \in P$, denote by o_p^t (resp. e_p^t) the number of buses of type $t \in T$ in its exit (resp. entry) block and define a nonnegative integer variable X_p that indicates the number of lanes partitioned according to this pattern. Furthermore, let Y_{pi} (resp. Z_{pj}) be a nonnegative integer variable that stipulates, for the planned arrival (resp. departure) scenario, the number of lanes partitioned according to pattern $p \in P_2$ whose exit block is full (resp. empty) when the arrival in position $i \in I$ (resp. departure $j \in J$) has just been parked (resp. left).

To ensure that a feasible assignment exists for the computed solution, the model requires the following notation. Let a_{is}^t be the number of type t buses arrived when the arrival in position $i \in I$ in scenario $s \in S^k$ occurs and define the following minimal and maximal values :

$$\overline{a}_i^{tk} = \max_{s \in S^k} a_{is}^t \qquad \forall\, t \in T,\ i \in I \tag{6.1}$$

$$\underline{a}_i^{tk} = \min_{s \in S^k} a_{is}^t \qquad \forall\, t \in T,\ i \in I. \tag{6.2}$$

It is easy to verify that, for a given $t \in T$, the sequence $\{\overline{a}_i^{tk}\}_{i \in I}$ is non-decreasing. Denote by I_k^t the set of indices i where this sequence increases, that is, $I_k^t = \{i \in I \mid \overline{a}_i^{tk} > \overline{a}_{i-1}^{tk}\}$, where $\overline{a}_0^{tk} = 0$.

Note that, instead of considering all scenarios in S^k, the values (6.1) and (6.2) can easily be computed by only considering extreme scenarios. For instance, the maximal values $\overline{a}_i^{tk}$, for all $i \in I$ and a given type $t \in T$, correspond to the values a_{is}^t of the extreme forward scenario $s \in S^k$ in which all arrivals of type t are shifted forward by k positions with respect to their positions in the planned arrival scenario, while the other arrivals are shifted backward only to compensate the forward shifts of the type t arrivals. The minimal values $\underline{a}_i^{tk}$ can similarly be computed by considering extreme backward scenarios.

For the departures, simpler data are required because no changes to the planned departure scenario can occur. In this case, there is only one departure scenario to consider and the minimal and maximal required values are thus equal. Let d_j^t be the number of buses of type $t \in T$ out of the depot when the bus assigned to the morning route in position $j \in J$ is leaving the depot, and denote by J^t the set of indices where the sequence $\{d_j^t\}_{j \in J}$ is increasing, that is, $J^t = \{j \in J \mid d_j^t > d_{j-1}^t\}$, where $d_0^t = 0$.

Even though the k-position variant of the BDPB is only a feasibility problem as defined above, we propose to consider the objective of minimizing the number of two-block lanes (or equivalently, maximizing the number of one-block lanes) because one-block lanes facilitate the operations. In this case, the proposed formulation is as follows :

$$\text{Minimize} \qquad \sum_{p \in P_2} X_p \tag{6.3}$$

$$\text{subject to :} \qquad \sum_{p \in P} X_p = v \tag{6.4}$$

$$\sum_{p \in P} (o_p^t + e_p^t) X_p = b^t, \qquad \forall\, t \in T \tag{6.5}$$

$$\sum_{p\in P} o_p^t X_p + \sum_{p\in P_2} e_p^t Y_{pi} \geq \overline{a}_i^{tk}, \qquad \forall\, t \in T, i \in I_k^t \tag{6.6}$$

$$\sum_{p\in P_2} o_p^t Y_{pi} \leq \underline{a}_i^{tk}, \qquad \forall\, t \in T, i \in \bigcup_{\substack{u\in T\\ u\neq t}} I_k^u \tag{6.7}$$

$$Y_{pi} \leq Y_{p,i+1}, \qquad \forall\, p \in P_2, i \in I \setminus \{n\} \tag{6.8}$$

$$Y_{pn} \leq X_p, \qquad \forall\, p \in P_2 \tag{6.9}$$

$$\sum_{p\in P} o_p^t X_p + \sum_{p\in P_2} e_p^t Z_{pj} \geq d_j^t, \qquad \forall\, t \in T, j \in J^t \tag{6.10}$$

$$\sum_{p\in P_2} o_p^t Z_{pj} \leq d_j^t, \qquad \forall\, t \in T, j \in \bigcup_{\substack{u\in T\\ u\neq t}} J^u \tag{6.11}$$

$$Z_{pj} \leq Z_{p,j+1}, \qquad \forall\, p \in P_2, j \in J \setminus \{n\} \tag{6.12}$$

$$Z_{pn} \leq X_p, \qquad \forall\, p \in P_2 \tag{6.13}$$

$$X_p, Y_{pi}, Z_{pj} \text{ are nonnegative integers}, \qquad \text{for all valid indices.} \tag{6.14}$$

The objective function (6.3) minimizes the number of two-block lanes. Constraint (6.4) ensures that each lane is partitioned according to a specific one-block or two-block pattern. Constraints (6.5) forces the number of type t slots in the depot to be equal to the number of type t buses. Constraint sets (6.6)–(6.9) guarantee that a feasible assignment of the arrivals to the slots exists for all arrival scenarios in S^k. Indeed, constraints (6.6) impose that the number of slots of type t that are or have been accessible when the arrival in position $i \in I_k^t$ is being parked is at least $\overline{a}_i^{tk}$, the maximum number of type t arrivals at that time according to all possible scenarios in S^k. A slot is said to be or have been *accessible* at a given time if it belongs to an exit block or to an entry block for which the corresponding exit block is full at that time. Constraints (6.7) ensure that this last condition is satisfied when the corresponding entry block becomes accessible. To do so, they verify that the number of slots in the type t exit blocks that are filled when the arrival in position $i \in I$ has just been parked does not exceed $\underline{a}_i^{tk}$, the minimum number of type t arrivals at that time according to all possible scenarios in S^k. Constraints (6.8) indicate that an entry block which becomes accessible before the arrival in position $i \in I \setminus \{n\}$ remains accessible until the last arrival, while constraints (6.9) restrict the number of accessible entry blocks for the pattern $p \in P_2$ after the last arrival to be at most the number of lanes partitioned according to p. Constraints (6.10)–(6.13) are the departure counterparts of constraints (6.6)–(6.9). Finally, nonnegativity and integrality requirements are expressed by (6.14).

It is easy to see that a feasible planned solution for model (6.3)–(6.14) remains feasible during the operations for all possible arrival scenarios in S^k. Indeed, constraints (6.6) ensure that, no matter what scenario $s \in S^k$ is realized, enough slots of the proper type are accessible upon each arrival because $\overline{a}_i^{tk}$ is greater than or equal to a_{is}^t. Similarly, constraints (6.7) guarantee that enough arrivals of the proper type have occurred for filling the exit blocks of the accessible entry blocks because $\underline{a}_i^{tk}$ is smaller than or equal to a_{is}^t.

To solve the k-position model (6.3)–(6.14), we propose to use a commercial MIP solver such as CPLEX. Once an optimal solution has been computed, an optimal assignment of the planned arrivals and planned departures to the individual slots can be determined in polynomial time using the assignment procedure proposed in Hamdouni *et al.*(2004). This procedure proceeds in two independent steps, one for the arrivals and the other for the departures. Let us consider the arrival case (the departure case is similar). For each bus type $t \in T$, the procedure creates an ordered list of lanes L^t which contains in order all the two-block lanes with a type t exit block (in an order specified next), all the one-block lanes of type t (in any order), and all the two-block lanes with a type t entry block (in any order). To determine the order of the two-block lanes with a type t exit block, an arrival index i is associated with each of these lanes as follows. When a lane is the only one with a pattern $p' \in P_2$, then this lane is associated with the smallest index i such that $Y_{p'i} = 1$. When more than one lane (say G lanes) have the same pattern $p' \in P_2$, then they are associated with the indices $i_1, i_2, \ldots, i_G$ where i_g, $g = 1, 2, \ldots, G$, is the smallest index i such that $Y_{pi} \geq g$. In L^t, the two-block lanes with a type t exit block are ordered in increasing order of their associated index. After building the list L^t, the procedure assigns the type t buses to the slots in such a way that the type t blocks are filled according to the order of the lanes in L^t. Obviously, a bus assigned to a block is always assigned to the slot that is the closest to the lane exit among the slots that are still empty.

The k-position approach presented in this section is easy to implement and yields robust solutions, especially when the problem is not too tight (see Section 6.5). On the other hand, it has the following disadvantages. First, it is difficult to find, for each instance, the largest value of k for which the instance is feasible. This value is desirable as it would yield the most robust solution. Second, the k value is independent of the bus type. If type-dependent values were used, additional robustness would be gained. However, it would be much more difficult to determine these values and, moreover, the computation of the maximal and minimal values (6.1) and (6.2) would not be possible using only extreme scenarios as discussed above. Finally, the k value does not depend on the planned arrival times for the same reasons. In fact, it should depend on them because the probability of shifting forward (resp. backward) an arrival by k positions is surely higher when the k preceding (resp. succeding) arrivals arrive in a short period of time rather than in a long period of time. In the next section, we present a stochastic approach that does not rely on such a k value and, therefore, does not have the above disadvantages.

6.4 The stochastic approach

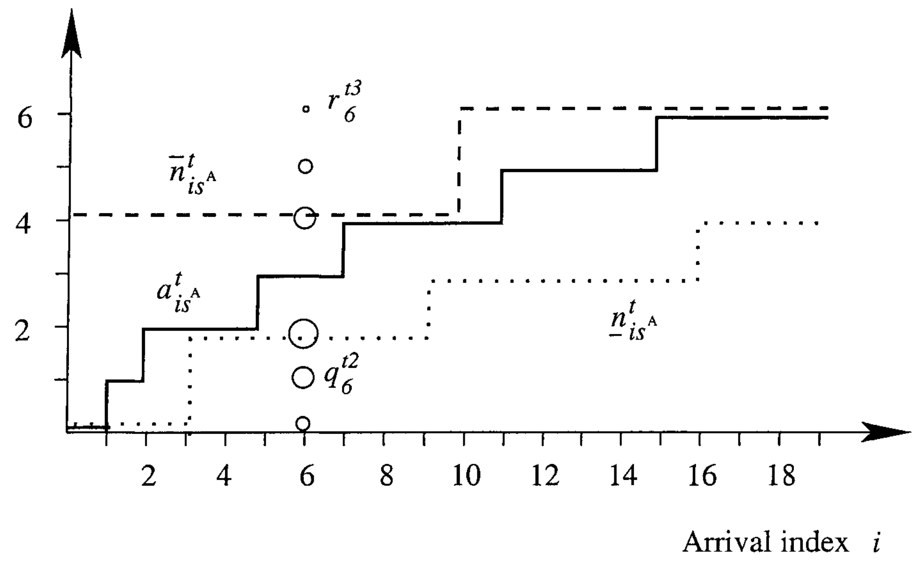
This section describes a robust approach for the BDPB, called the stochastic approach, which produces a planned solution that minimizes, under stochastic arrival times, the expectation of a function positively correlated with the number of buses that make the planned solution infeasible because they arrive too late or too early. To gain intuition on the proposed approach, let us consider an example involving six buses of type $t \in T$ which arrive in positions 1, 2, 5, 7, 11, and 15 in a planned arrival scenario s^A. A proposed solution for this scenario assigns these buses in three exit blocks and one entry block of four two-block lanes. These exit blocks contains two, one, and one slot, respectively, and must be filled before the 4th, 10th, and 17th arrival, respectively. The entry block contains two slots that must become accessible when the 10th arrival occurs. Given this arrival scenario and this planned solution, one can compute the number $a^t_{is^A}$ of buses of type t arrived when the ith arrival occurs, the number $\overline{n}^t_{is^A}$ of type t slots that are or have been accessible at this time, and the number $\underline{n}^t_{is^A}$ of type t slots that must necessarily be filled in the exit blocks of a two-block lane at this time. Using the notation of Section 6.3, the numbers $\overline{n}^t_{is^A}$ and $\underline{n}^t_{is^A}$ are computed from the planned solution as follows :

$$\overline{n}^t_{is^A} = \sum_{p \in P} o^t_p X_p + \sum_{p \in P_2} e^t_p Y_{pi} \tag{6.15}$$

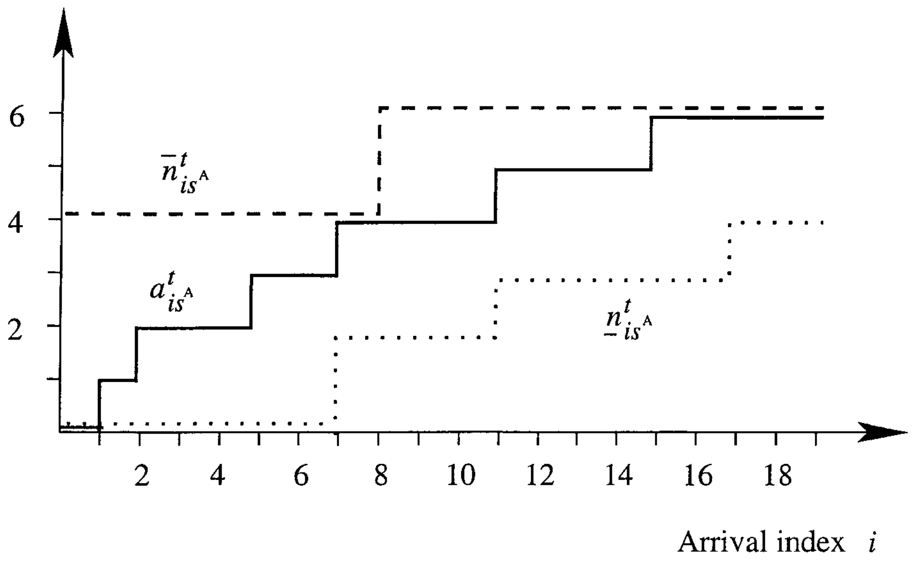
$$\underline{n}^t_{is^A} = \sum_{p \in P_2} o^t_p Y_{pi}. \tag{6.16}$$

In part (a), Figure 6.1 illustrates these numbers as stepwise curves (solid line for a^t_{isA}, dashed line for $\overline{n}^t_{isA}$, and dotted line for $\underline{n}^t_{isA}$). The planned solution is feasible for this scenario with respect to the type t buses because the solid line lays between the other two lines, that is, $\underline{n}^t_{isA} \leq a^t_{isA} \leq \overline{n}^t_{isA}$ for all $i \in I$. When this condition is violated for at least one bus type, the solution is infeasible.

Now, notice that the lines in part (a) of this figure indicate that the planned solution does not provide much leeway around positions 3 and 9 to account for a change in the arrival scenario. Indeed, as soon as one of the first two arrivals of type t is delayed after the 3th arrival, then the first type t exit block cannot be filled on time (assuming that the first bus assigned to the corresponding entry block is not late) for this planned solution which becomes infeasible. On the other hand, if the 5th arrival of type t, which is assigned in the planned solution to the entry block of a two-block lane, arrives in position 9 instead of 11, then this arrival cannot be parked properly (assuming that the buses assigned to the corresponding exit block are not ahead of time) and the planned solution becomes again infeasible in this case. Therefore, a planned solution should be more robust when the dashed and dotted lines are as far as possible from the solid line for every position $i \in I$. Part (b) of Figure 6.1 shows the stepwise curves for a different planned solution which increases the robustness of the solution for the type t buses. However, it should be noted that this solution may diminish the robustness of the solution for another bus type. For instance, with this new solution, the fifth and sixth type t slots, which belong to an entry block, must become accessible after the 8th arrival instead of the 10th. Therefore, the corresponding exit block must be filled more rapidly, yielding a loss in robustness for the type assigned to this block (assuming that the same type is used in both solutions). Hence, a tradeoff in the solution robustness for every type must



(a)



(b)

Figure 6.1 – Numbers of arrived buses (solid line), accessible slots (dashed line), and required buses (dotted line) for a planned arrival scenario s^A and a bus type t. Parts (a) and (b) present different planned solutions.

be reached to obtain the most robust solution from a global perspective.

The proposed stochastic approach determines this tradeoff by considering the following probabilities. First, we consider the probability over all possible scenarios $s \in S$ that a^t_{is} exceeds by m units the number of arrived buses $a^t_{is^A}$ in the planned scenario s^A for all types $t \in T$, positions $i \in \{1, 2, \ldots, \bar{\imath}^t - 1\}$, and values $m \in \{1, 2, \ldots, b^t - a^t_{is^A}\}$, where $\bar{\imath}^t$ is the position of the last arrival of type t in scenario s^A. This probability is denoted r^{tm}_i. We also consider the probability q^{tm}_i that a^t_{is} is smaller by m units than $a^t_{is^A}$ for all types $t \in T$, positions $i \in \{\underline{i}^t, \ldots, n\}$, and values $m \in \{1, 2, \ldots, a^t_{is^A}\}$, where $\underline{i}^t$ is the position of the first type t arrival in scenario s^A. The probabilities r^{tm}_i and q^{tm}_i for $i = 6$ are illustrated in part (a) of Figure 6.1 using circles. The area of a circle is proportional to the value of the corresponding probability. For a given position i and a type t, the sum of the probabilities associated with the circles that lay above the dashed line or under the dotted line is a measure of the lack of robustness of a solution with respect to this position and type. A low value (close to 0) for this sum represents a high level of robustness, while a high value (close to 1) indicates the opposite.

Our stochastic approach is based on an integer linear program that uses the r^{tm}_i and q^{tm}_i probabilities as coefficients in the objective function. In this section, we first describe how to compute these probabilities using simulation and, then, present this integer program.

6.4.1 Estimating probabilities r_i^{tm} and q_i^{tm} by simulation

The value of each probability r_i^{tm} and q_i^{tm} depends on complex interactions between many arrival time probability distributions. Furthermore, there are no reasons to believe that these distributions are among the classic distributions that can easily be handled in the computation of a convolution. Therefore, it appears nearly impossible to develop analytical formulas for these probabilities. To overcome this difficulty, we propose an approach which is able to deal with very complex interactions, namely, the simulation. Hence, we estimate the probabilities r_i^{tm} and q_i^{tm} by simulating a large number of arrival scenarios and collecting data from these scenarios. To simulate a scenario, we first generate for each index $i \in I$ a random arrival time $\tilde{h}_i^A$ derived from the original time h_i^A as follows. Let H_i^A be a random variable representing the arrival time of the bus in position $i \in I$ in the planned scenario s^A. In our experiments, we used for H_i a triangular probability density function $f_i(h)$ that takes a positive value only in the interval $[\alpha_i, \beta_i]$ and has a mode of h_i^A (note that a different density function could have been used easily) :

$$f_i(h) = \begin{cases} \frac{2(h-\alpha_i)}{(h_i^A-\alpha_i)(\beta_i-\alpha_i)} & \text{if } \alpha_i \leq h \leq h_i^A \\ \frac{2(\beta_i-h)}{(\beta_i-h_i^A)(\beta_i-\alpha_i)} & \text{if } h_i^A < h \leq \beta_i \\ 0 & \text{otherwise.} \end{cases} \tag{6.17}$$

To reflect the tendency of the drivers, especially in the evening, to arrive earlier than their planned arrival times, we used $[\alpha_i, \beta_i] = [h_i^A - 20, h_i^A + 10]$ in our tests. The corresponding cumulative distribution function $F_i(h)$ is

$$F_i(h) = \begin{cases} 0 & \text{if } h < \alpha_i \\ \frac{(h-\alpha_i)^2}{(h_i^A-\alpha_i)(\beta_i-\alpha_i)} & \text{if } \alpha_i \leq h \leq h_i^A \\ 1 - \frac{(\beta_i-h)^2}{(\beta_i-h_i^A)(\beta_i-\alpha_i)} & \text{if } h_i^A < h \leq \beta_i \\ 1 & \text{otherwise.} \end{cases} \tag{6.18}$$

The inverse $F_i^{-1}(u)$ of this function, which is defined on the open interval $(0,1)$, is then given by

$$F_i^{-1}(u) = \begin{cases} \alpha_i + \sqrt{u(h_i^A - \alpha_i)(\beta_i - \alpha_i)} & \text{if } 0 < u \leq \frac{h_i^A-\alpha_i}{\beta_i-\alpha_i} \\ \beta_i - \sqrt{(1-u)(\beta_i - h_i^A)(\beta_i - \alpha_i)} & \text{if } \frac{h_i^A-\alpha_i}{\beta_i-\alpha_i} < u < 1. \end{cases} \tag{6.19}$$

To generate a random arrival time $\tilde{h}_i^A$ for index i, we first generate a random variate u_i from a uniform distribution in $[0,1]$ and then compute $\tilde{h}_i = F_i^{-1}(u_i)$. Once a random arrival time $\tilde{h}_i^A$ has been generated for each arrival index $i \in I$, a random arrival scenario is obtained by sorting the arrivals in increasing order of their generated arrival times.

To estimate the probabilities r_i^{tm} and q_i^{tm}, we generate as described above a set $\tilde{S}$ of N random arrival scenarios. For each scenario $s \in \tilde{S}$, we compute the value of a_{is}^t for each type $t \in T$ and position $i \in I$. Then, we estimate the probabilities r_i^{tm} and q_i^{tm} using the formulas

$$r_i^{tm} \simeq \overline{N}_i^{tm}/N, \qquad \forall\, t \in T,\ i \in I,\ m \in \{1, 2, \ldots, b^t - a_{is^A}^t\} \tag{6.20}$$

$$q_i^{tm} \simeq \underline{N}_i^{tm}/N, \qquad \forall\, t \in T,\ i \in I,\ m \in \{1, 2, \ldots, a_{is^A}^t\}, \tag{6.21}$$

where $\overline{N}_i^{tm}$ (resp. $\underline{N}_i^{tm}$) is the number of scenarios $s \in \tilde{S}$ for which $a_{is}^t = a_{is^A}^t + m$ (resp. $a_{is}^t = a_{is^A}^t - m$). In our experiments, the value of N was set to 5000. With this value, we observed that the probabilities were accurate to 0.01 when repeating the simulation process with different random seed values.

6.4.2 An integer programming model

As mentioned above, the stochastic approach also formulates the BDPB as an integer program, which is very similar to the program (6.3)–(6.14) used for the k-position approach. The differences between these programs only appear in the objective function and in some of the constraints ensuring that the arrivals can be parked correctly. In the stochastic model, the objective aims at minimizing the sum over all types $t \in T$ and positions $i \in I$ of the probabilities that, when the arrival in position i occurs, either there are too many type t buses arrived for the number of type t accessible slots at that time or there are not enough type t buses arrived to fill the type t exit blocks that should be filled at that time. As it will be shown by the computational results reported in Section 6.5, this objective is positively correlated with the main objective of maximizing solution robustness.

Beside the notation previously described, the following notation is required to present the proposed integer programming model. For each $t \in T$ and $i \in \{1, 2, \ldots, \bar{i}^t - 1\}$, let $\overline{M}_i^t = \max \{m \mid r_i^{tm} > 0\}$ be the maximum difference between a_{is}^t and $a_{is^A}^t$ among all arrival scenarios $s \in \tilde{S}$. For each $t \in T$, $i \in \{1, 2, \ldots, \bar{i}^t - 1\}$, and $m \in \{0, 1, \ldots, \overline{M}_i^t\}$, define a binary variable R_i^{tm} that takes value 1 if the number $\overline{n}_{is^A}^t$ of type t slots that are or have been accessible when the i^{th} arrival occurs in scenario s^A exceeds by m units the number $a_{is^A}^t$ of type t buses arrived at this time, and 0 otherwise. With this variable, associate a cumulative probability given by $\rho_i^{tm} = \sum_{w=m+1}^{\overline{M}_i^t} r_i^{tw}$ if $m < \overline{M}_i^t$ and by $\rho_i^{tm} = 0$ if $m = \overline{M}_i^t$.

Similarly, for each $t \in T$ and $i \in \{\underline{i}^t, \ldots, n\}$, let $\underline{M}_i^t = \max\{m \mid q_i^{tm} > 0\}$ be an approximation of the maximum difference between a_{is}^t and $a_{is^A}^t$ among all arrival scenarios $s \in S$. For each $t \in T$, $i \in \{\underline{i}^t, \ldots, n\}$, and $m \in \{0, 1, \ldots, \underline{M}_i^t\}$, define a binary variable Q_i^{tm} that takes value 1 if the number $\underline{n}_{is^A}^t$ of type t slots that must necessarily be filled in the exit blocks of a two-block lane when the i^{th} arrival occurs in scenario s^A is smaller than $a_{is^A}^t$ by m units, and 0 otherwise. With this variable, associate a cumulative probability given by $\pi_i^{tm} = \sum_{w=m+1}^{\underline{M}_i^t} q_i^{tw}$ if $m < \underline{M}_i^t$ and by $\pi_i^{tm} = 0$ if $m = \underline{M}_i^t$.

Using this notation, the integer program for the stochastic approach is :

$$\text{Minimize} \quad \sum_{t \in T} \left(\sum_{i=1}^{\bar{i}^t - 1} \sum_{m=0}^{\overline{M}_i^t} \rho_i^{tm} R_i^{tm} + \sum_{i=\underline{i}^t}^{n} \sum_{m=0}^{\underline{M}_i^t} \pi_i^{tm} Q_i^{tm} \right) \tag{6.22}$$

$$\text{subject to :} \quad \sum_{p \in P} X_p = v \tag{6.23}$$

$$\sum_{p \in P} (o_p^t + e_p^t) X_p = b^t, \qquad \forall t \in T \tag{6.24}$$

$$\sum_{p \in P} o_p^t X_p + \sum_{p \in P_2} e_p^t Y_{pi} \geq a_{is^A}^t + \sum_{m=0}^{\overline{M}_i^t} m R_i^{tm}, \qquad \forall t \in T, i \in I \tag{6.25}$$

$$\sum_{p \in P_2} o_p^t Y_{pi} \leq a_{is^A}^t - \sum_{m=0}^{\underline{M}_i^{tm}} m Q_i^{tm}, \qquad \forall t \in T, i \in I \tag{6.26}$$

$$Y_{pi} \leq Y_{p,i+1}, \qquad \forall p \in P_2, i \in I \setminus \{n\}$$

$$Y_{pn} \leq X_p, \qquad \forall p \in P_2 \tag{6.27}$$

$$\sum_{p \in P} o_p^t X_p + \sum_{p \in P_2} e_p^t Z_{pj} \geq d_j^t, \qquad \forall\, t \in T, j \in J^t \tag{6.28}$$

$$\sum_{p \in P_2} o_p^t Z_{pj} \leq d_j^t, \qquad \forall\, t \in T, j \in J \tag{6.29}$$

$$Z_{pj} \leq Z_{p,j+1}, \qquad \forall\, p \in P_2, j \in J \setminus \{n\} \tag{6.30}$$

$$Z_{pn} \leq X_p, \qquad \forall\, p \in P_2 \tag{6.31}$$

$$\sum_{m=0}^{\overline{M}_i^t} R_i^{tm} = 1, \qquad \forall\, t \in T, i \in I \tag{6.32}$$

$$\sum_{m=0}^{\underline{M}_i^t} Q_i^{tm} = 1, \qquad \forall\, t \in T, i \in I \tag{6.33}$$

$$R_i^{tm}, Q_i^{tm} \in \{0,1\} \qquad \text{for all valid indices} \tag{6.34}$$

$$X_p, Y_{pi}, Z_{pj} \text{ are nonnegative integers}, \qquad \text{for all valid indices.} \tag{6.35}$$

The objective function (6.22) aims at minimizing the sum over all types $t \in T$ and positions $i \in I$ of the probabilities that the numbers a_i^t of type t buses arrived in position i or before do not fall in the interval $[\underline{n}_{isA}^t, \overline{n}_{isA}^t]$. Constraints (6.23), (6.24), (6.27)–(6.31), and (6.35) are identical to constraints (6.4), (6.5), (6.8)–(6.13), and (6.14), respectively. Constraints (6.25) and (6.26) are very similar to constraints (6.6) and (6.7). They only differ by their right-hand sides which can be adjusted in the stochastic model instead of being fixed in the k-position model. Constraints (6.32) and (6.33) ensure that exactly one level of flexibility in the right-hand sides of constraints (6.25) and (6.26) is chosen for each type $t \in T$ and each position $i \in I$. Finally, binary requirements (6.34) are imposed on the R_i^{tm} and Q_i^{tm} variables.

Notice that any feasible solution to this stochastic model is feasible for the planned arrival scenario because the right-hand side of a constraint (6.25) (resp. constraint (6.26)) cannot be less (resp. greater) than a^t_{isA}. Notice also that the main difference between the stochastic model (6.22)–(6.35) and the k-position model (6.3)–(6.14) is the following. In the k-position model, the right-hand sides of constraints (6.6) and (6.7) define strict lower and upper bounds on the values that can be taken by $\underline{n}^t_{isA}$ and $\overline{n}^t_{isA}$ in a planned solution, while, in the stochastic model, these right-hand sides are flexible and the values taken by $\underline{n}^t_{isA}$ and $\overline{n}^t_{isA}$ in a planned solution are rather influenced by the cumulative probabilities ρ^{tm}_i and π^{tm}_i.

As for the k-position model (6.3)–(6.14), the stochastic model (6.22)–(6.35) can be solved using a commercial MIP solver, namely CPLEX. This model is also an implicit model that do not provide a detailed assignment of the buses to the slots. The polynomial-time assignment procedure described at the end of Section 6.3 is therefore needed to obtain such a detailed assignment.

We would like to point out that using simulation in a preprocessing step to compute cost or constraint coefficients of an optimization model is a powerful idea to model complex factors. This idea was used in the stochastic model (6.22)–(6.35) and can certainly be used also in several other contexts.

6.5 Computational results

To evaluate the solution approaches exposed in the previous sections, we performed a series of computational experiments. We report the results of these experiments in this section. Beforehand, we describe the datasets used for these tests.

6.5.1 Data description

All instances used in our experiments are derived from a real-world dataset provided by the Montreal transit authority, which is called *Société de Transport de Montréal* (STM). This dataset involves 124 buses of 4 different types. It consists of a sequence of arrivals and a sequence of departures. For each arrival and each departure, it specifies a bus type and an event time.

From this dataset, we created instances with different characteristics, namely, different depot sizes, numbers of buses, and numbers of types. Since in most depots, the number ℓ of slots per lane rarely exceeds 10, we used depots with $\ell \in \{6, 8, 10\}$. Given the assumption that $n = v\ell$ (recall that v is the number of lanes), we removed randomly four arrivals and four departures of the corresponding types to obtain a dataset with 120 buses. From this dataset with 4 types (denoted A, B, C, and D), we created another dataset involving 6 types by converting randomly some of the A (resp. B) arrivals into arrivals of a new type E (resp. F), and an equal number of A (resp. B) departures into E (resp. F) departures. To obtain a dataset with 40 buses and 4 types, we selected the first 40 arrivals and departures of the dataset with 120 buses and 4 types. The types of these arrivals and departures were randomly readjusted to reflect the type frequencies in the STM real-world dataset. Two other datasets, one with 64 buses and 5 types and the other with 64 buses and 6 types, were generated in a similar fashion.

The medium-sized problems (those with 40 and 64 buses) were created to obtain problems that are tighter than the large-sized problems. They yield a smaller average number of lanes per bus type, leaving very little flexibility to park the buses in a

Tableau 6.1 – Number of buses of each type in each dataset

Dataset	Bus types					
	A	B	C	D	E	F
D-120-4	68	36	6	10	-	-
D-120-6	37	21	6	10	31	15
D-40-4	22	12	3	3	-	-
D-64-5	19	19	4	5	17	-
D-64-6	13	19	4	5	17	6

different lane when it cannot be parked in its assigned lane because it arrives too early or too late. Such tight problems may arise as subproblems of larger real-life problems in which the depot layout is more complex than a set of lanes that do not interfere with themselves. Indeed, often in this case, some lanes block the entry of other lanes, yielding precedence constraints in the filling order of these lanes. The depot is then partitioned into groups of lanes, where a group is composed of lanes that do not interfere with themselves. Each group defines a medium-sized BDPB. As highlighted by the results on the solution robustness presented in Section 6.5.3, planned solutions for these medium-sized instances are, in general, less robust than those computed for the large-sized instances.

Table 6.1 reports the number of buses of each type in each dataset used for our experiments. Each dataset is associated with an identifier $D - n - |T|$, where D stands for dataset, n is the number of buses and $|T|$ the number of types. For instance, $D - 120 - 4$ refers to the dataset with 120 buses and 4 types.

6.5.2 Computational times

This section reports the computational times obtained by the two proposed approaches on various BDPB instances. All integer programs were solved on a 440-MHz Sun Ultra 10 workstation using the version 8.1 of the CPLEX MIP solver. The following CPLEX features were applied : the strong branching method, the cutting plane generator used moderately, and the heuristic procedure for finding integer solutions when invoked by CPLEX. Branching priorities on the variables were also used as follows : highest priority for the one-block pattern X_p variables ($p \in P \setminus \{P_2\}$), second highest priority for the two-block pattern X_p variables ($p \in P_2$) and the lowest priority for all the other variables.

Tables 6.2 and 6.3 present, for the tested instances, the sizes of the k-position and stochastic models, respectively, before and after the preprocessing performed by CPLEX. For the k-position model, these sizes are provided for a single value of k. They are similar for other values of k. From these statistics, one can observe that the model size increases with the number of buses and the number of types. Also, the stochastic model involves approximately the same number of constraints, but much more variables than the k-position model. These additional variables are the R_i^{tm} and Q_i^{tm} variables.

Computational results for the k-position approach are reported in Table 6.4 for a wide variety of instances. For the large datasets (involving 120 buses), instances with 6, 8, and 10 slots per lane were solved with three different values of k, namely, 0, 10, and the largest value k_{max} for which the problem remains feasible. For the smaller ones, instances with 8 slots per lane were tackled with two different values of k, namely, Tableau 6.2 – Numbers of variables and constraints in the k-position model

				Before CPLEX preprocessing		After CPLEX preprocessing	
Dataset	v	ℓ	k	Nb var	Nb const	Nb var	Nb const
D-120-4	20	6	10	7024	14063	3577	8114
	15	8	10	10730	19019	4970	11012
	12	10	10	14395	23975	5575	12172
D-120-6	20	6	10	18143	33195	10664	23731
	15	8	10	27423	45035	13120	28538
	12	10	10	37244	56875	14578	31305
D-40-4	5	8	2	2881	5955	970	2194
D-64-5	8	8	7	8601	18148	3900	8665
D-64-6	8	8	5	10996	22995	4764	10430

0 and k_{max}. For each test, Table 6.4 specifies the number of two-block lanes in the computed solution (2BL), the total number of branch-and-bound nodes explored in the search tree, and the total CPU time in seconds. From these results, we notice that all these instances except one ($D - 120 - 6$ with 15 lanes and $k = 23$) are relatively easy to solve : they require less than 40 minutes of computational time. We also observe that the computational time usually increases with the value of k, that is, as the instances become close to being infeasible. Finally, let us recall that to obtain the most robust solutions, k should be set to k_{max}. However, the determination of k_{max} requires a dichotomic search that substantially increases the overall computational time.

Table 6.5 shows the results obtained by the stochastic approach for the same instances. In this table, the computational times include the times required to compute the probabilities r_i^{tm} and q_i^{tm} by simulation. It should also be noted that, to obtain the reported computational times for the instances involving datasets $D - 64 - 5$ and $D - 64 - 6$, all cumulative probabilities ρ_i^{tm} and π_i^{tm} whose values were less

Tableau 6.3 – Numbers of variables and constraints in the stochastic model

			Before CPLEX preprocessing		After CPLEX preprocessing	
Dataset	v	ℓ	Nb var.	Nb const.	Nb var.	Nb const.
D-120-4	20	6	18939	13090	16208	10082
	15	8	23006	17386	17934	12236
	12	10	27531	21682	18699	13254
D-120-6	20	6	34747	27356	29830	22484
	15	8	44609	36896	33442	26637
	12	10	54926	46436	35577	29168
D-40-4	5	8	4925	5616	1933	2118
D-64-5	8	8	11591	13667	5935	7028
D-64-6	8	8	16678	19210	7543	8687

than 0.1 were fixed to 0. Then, because ρ_i^{tm} (resp. π_i^{tm}) is necessarily 0 when $\rho_i^{t,m-1}$ (resp. $\pi_i^{t,m-1}$) is equal to 0, all R_i^{tm} (resp. Q_i^{tm}) variables such that $\rho_i^{t,m-1} = 0$ (resp. $\pi_i^{t,m-1} = 0$) were also removed to reduce problem size. With these modifications, all computational times, which vary between 0.5 and 36 minutes, are reasonable. In fact, they are comparable to the times obtained by the k-position approach for a single value of k.

6.5.3 Solution robustness

To evaluate the robustness of the solutions computed by the two proposed approaches on all tested instances, we performed the following experiment. For each solution, we first established the lane filling order using the procedure described at the end of Section 6.3. Then, we generated 5000 arrival scenarios as described in Section 6.4.1. For each of these scenarios, we verified if the planned solution remains feasible when the lane filling order is respected. When this is not the case, we computed the number

Tableau 6.4 – Computational results for the k-position approach

Dataset	v	ℓ	k	2BL	Nb nodes	CPU time (s)
D-120-4	20	6	0	1	1	4
			10	1	1	8
			44	1	1	3
D-120-4	15	8	0	3	1	3
			10	3	8	165
			19	5	152	1109
D-120-4	12	10	0	2	1	6
			10	2	1	24
			17	2	1	10
D-120-6	20	6	0	3	1	7
			10	3	1	23
			48	4	1	211
D-120-6	15	8	0	4	51	274
			10	4	1	1231
			23	5	211	14395
D-120-6	12	10	0	4	11	98
			10	4	19	605
			22	4	1	1412
D-40-4	5	8	0	3	1	1
			2	4	17	11
D-64-5	8	8	0	4	1	15
			7	6	133	1183
D-64-6	8	8	0	5	61	125
			5	6	278	2373

of arrivals that could not be parked upon their arrival and, for each of them, the number of additional buses that arrived subsequently before it can be parked. In practice, these last two statistics are also important when the probability that a solution be infeasible for an arrival scenario is positive.

For all the instances involving 120 buses except in one case, no planned solutions were infeasible. The computed solutions are thus very robust, even those obtained by the

Tableau 6.5 – Computational results for the stochastic approach

Dataset	v	ℓ	2BL	Nb nodes	CPU time (s)
D-120-4	20	6	1	1	40
	15	8	3	1	64
	12	10	2	1	33
D-120-6	20	6	5	1	77
	15	8	5	11	588
	12	10	4	11	438
D-40-4	5	8	4	67	76
D-64-5	8	8	7	221	1104
D-64-6	8	8	7	237	2158

k-position approach with $k = 0$. The only planned solution that became infeasible for 13% of the generated arrival scenarios is the one computed by the 0-position approach for the instance $D - 120 - 4$ with 15 lanes. The situation is different for the medium-sized instances which offer less flexibility than the 120-bus instances. Indeed, the computed solutions are less robust as shown by the results reported in Table 6.6. This table specifies, for each instance and each solution approach, the probability that a solution be infeasible for an arrival scenario (Prob inf), the average number of arrivals that cannot be parked upon their arrival (Avg nb inf arr), and the average number of additional buses that must arrive before parking an arrival that could not be parked right away (Avg wait). As expected, we see that the k-position approach solutions are more robust when computed with $k = k_{max}$ than with $k = 0$. We also observe that the stochastic approach produces solutions that are more robust for the instances involving 64 buses. For the $D - 40 - 4$ instance, the solutions obtained by the k_{max}-position and the stochastic approaches are relatively similar in terms of robustness with a slight advantage for the k_{max}-position solution that yields a smaller average number of buses to wait for when an arrival cannot be parked upon

Tableau 6.6 – Simulation results for evaluating solution robustness

			0-position			k_{max}-position			stochastic		
			Prob	Avg nb	Avg	Prob	Avg nb	Avg	Prob	Avg nb	Avg
Dataset	v	ℓ	inf	inf arr	wait	inf	inf arr	wait	inf	inf arr	wait
D-40-4	5	8	0.51	1.25	5.22	0.40	0.61	2.17	0.36	0.69	3.18
D-64-5	8	8	0.24	0.32	1.34	0.07	0.12	0.28	0.05	0.05	0.20
D-64-6	8	8	0.35	0.49	2.11	0.19	0.22	1.11	0.08	0.09	0.32

its arrival.

6.6 Conclusion

This paper has addressed the bus dispatching problem by blocks in which the bus arrival times can deviate stochastically from their planned arrival times. Two solution approaches have been proposed. In the k-position approach, the set of feasible solutions is restricted to the solutions that remain feasible even if the order of any bus in the arrival sequence is changed by at most k positions in a stochastic arrival scenario. The parameter k is either predetermined or adjusted through dichotomic search to obtain maximal robustness. In the stochastic approach, the objective consists of minimizing the expectation of a function positively correlated with the number of buses that yield an infeasible solution because they arrive too late or too early in a stochastic arrival scenario. Simulation is used to compute the objective function coefficients. In both approaches, the problem is modeled as an integer program solvable by a commercial MIP solver.

The experimentation conducted on instances derived from a real-world dataset showed that the stochastic approach performed better than the k-position approach on the tested medium-sized instances. Indeed, compared to the k-position approach with a maximal value for k, it produced solutions that are better or relatively similar in terms of robustness, while requiring smaller computational times when the dichotomic search time is included in the computational time of the k-position approach. On the other hand, for large-sized instances, both approaches are comparable and yields very robust solutions.

The idea of using simulation to compute cost or constraint coefficients for modeling complex factors in an optimization model has the potential to be applied in other contexts. This will be explored in a future research.

References

BLASUM, U., BUSSIECK, M.R., HOCHSTÄTTLER, W., MOLL, C., SCHEEL, H.-H. AND WINTER, T. (1999). Scheduling Trams in the Morning. *Mathematical Methods of Operations Research* **49**, 137–148.

FRELING, R., LENTINK, R.M., KROON, L.G., AND HUISMAN, D. (2005). Shunting of Passenger Train Units in a Railway Station. *Transportation Science* **39**, 261–272.

GALLO, G. AND DI MIELE, F. (2001). Dispatching Buses in Parking Depots. *Transportation Science* **35**, 322–330.

HAMDOUNI, M., DESAULNIERS, G., MARCOTTE, O., SOUMIS, F. AND VAN PUTTEN, M. (2004). Dispatching Buses in a Depot using Block Patterns. Technical Report, Les Cahiers du GERAD G-2004-51, HEC Montreal, Montreal, Canada (2004). To appear in *Transportation Science.*

HAMDOUNI, M., DESAULNIERS, G., SOUMIS, F. (2005). Parking buses in a depot using block patterns : a Benders decomposition approach for minimizing type mismatches Technical Report, Les Cahiers du GERAD G-2005-70, HEC Montreal, Montreal, Canada (2005). To appear in *Computers & Operations Research.*

WINTER, T. AND ZIMMERMANN, U.T. (2000). Real-time dispatch of trams in storage yards. *Annals of Operations Research* **96**, 287–315.

CHAPITRE 7 : DISCUSSION GÉNÉRALE ET CONCLUSION

Le problème de stationnement des véhicules dans un dépôt a peu été étudié dans la littérature. Les approches proposées pour le résoudre ne produisent pas de solutions robustes. En pratique, les gareurs utilisent des solutions robustes et simples à implanter qui sont basées sur le partitionnement de chaque voie en au plus deux blocs par voie et chaque bloc est réservé pour un seul type d'autobus. Dans cette thèse, nous nous sommes attaqués à cette problématique avec comme but de développer des méthodes pour différents contextes produisant des solutions robustes et faciles à implanter. Cet objectif a été atteint. Dans une première étape, nous avons introduit aux chapitre 4 et 5 des modèles basé sur le mode de stationnement par bloc dans un contexte déterministe et ce pour deux variantes du problème, soit avec manoeuvres et avec affectations erronées. Dans un deuxième temps, au chapitre 6, nous avons développé deux modèles avec stationnement par bloc dans un contexte stochastique.

Dans le chapitre 4, nous avons prouvé que le problème de stationnement par bloc est un problème NP-complet. Nous avons aussi démontré que le mode de stationnement des autobus par bloc permet d'obtenir des solutions plus robustes aux changements dans l'ordre d'arrivée que dans le cas où les autobus peuvent être stationnés de façon quelconque dans le dépôt. Finalement, nous avons développé des modèles pour différentes variantes du problème de stationnement par bloc avec manoeuvres et montré qu'ils pourraient être résolus efficacement par CPLEX.

Le chapitre 5 a été dédié au problème de stationnement par bloc en minimisant le nombre d'affectations erronées. Nous avons formulé ce problème comme un problème linéaire en nombres entiers de grande taille et proposé de le résoudre par une méthode de décomposition de Benders qui contient un sous-problème avec contraintes d'intégrité. Pour traiter l'intégrité des variables du sous-problème, nous avons développé une stratégie de branchement qui impose des décisions au niveau des variables du problème maître. L'efficacité de cette approche a été testée sur plusieurs instances de taille moyenne.

Nous avons ensuite examiné le problème de stationnement par bloc dans un contexte stochastique où les autobus peuvent arriver en avance ou en retard par rapport à l'heure prévue pour produire des solutions encore plus robustes. Nous avons introduit deux approches pour prendre en compte les perturbations dans l'ordre d'arrivée : Une approche déterministe nommé k-position et une autre stochastique. Nous avons formulé ces deux variantes du problème de stationnement par bloc comme des programmes linéaires en nombres entiers. Nous avons eu recours à une procédure de simulation pour mesurer la performance des deux approches proposées. Les résultats numériques montrent que l'approche stochastique performe mieux que l'approche k-position pour les problèmes serrés qui requièrent des solutions robustes.

Pour conclure cette thèse, voyons des extensions au travail accompli qui pourraient faire l'objet de recherches futures. Tel qu'il est souligné dans l'introduction, une configuration complexe du dépôt rend le problème de stationnement plus compliqué. En particulier, certaines voies sont perpendiculaires à d'autres et bloquent l'entrée de celles-ci. Ces voies doivent donc être remplies après les autres. Cela nécessite de considérer dans les modèles proposés des contraintes de préséance au moment du remplissage des voies. L'étude du problème de stationnement en tenant compte de la configuration du dépôt constitue une extension intéressante du problème de stationnement.

L'approche de décomposition de Benders combinée avec la stratégie de branchement pour traiter l'intégrité des variables du sous-problème proposée au chapitre 5 est une nouvelle approche qui résout le problème d'une façon optimale. Une telle approche pourrait être appliquée avec succès à d'autres problèmes quand l'intégrité des variables du sous-problème est exigée. Jusqu'à date seulement des heuristiques ont été utilisées pour déterminer une approximation de la solution entière dans une décomposition de Benders (voir Mercier et al. 2005).

Au chapitre 6, nous avons introduit deux nouvelles approches pour traiter les perturbations dans l'ordre d'arrivée des autobus. Les perturbations dans l'ordre d'arrivée rendent le nombre d'arrivées de chaque type après chaque arrivée connu avec incertitude. Les deux approches proposées peuvent être utilisées pour traiter d'autres problèmes où la demande est sujette à des variations ou est incertaine. Il est bien connu que les problèmes de design de services appartiennent à la catégorie des problèmes dont la demande est stochastique. Citons, par exemple, le problème de design d'un réseau de télécommunications pour un marché donné afin de fournir des services aux utilisateurs. Le design consiste à determiner la localisation et la capacité des liens. Le problème d'opération consiste à faire circuler la demande dans le réseau et cette demande peut être stochastique (voir Sanso et Soumis, 1992). Un deuxième exemple est le design des horaires de travail dans un centre d'appels. La demande est stochastique quand on considère l'arrivée des appels. Le problème d'opération consiste à affecter les appels à des agents en tenant compte de leurs qualifications et de contraintes sur la durée de la réponse (voir Cezik et Lecuyer, 2004). Un autre exemple est le design d'un réseau de transport (voir Soumis et Nagurney, 1993, Crainic, 1997) pour servir la demande prévue. L'opération consiste alors à transporter des passagers dans un réseau aérien ou des wagons dans un réseau ferroviaire. Des modèles d'optimisation de réseaux tenant compte des performances durant l'opération pourraient être développés pour ces domaines en reprenant l'idée d'utiliser une simulation pour estimer une fonction de coût représentant les aléas de l'opération.

CHAPITRE 8 : BIBLIOGRAPHIE

AVRIEL, M., PENN, M. et NAOMI, S. (2000). Container Ship Stowage Problem : complexity and connection to the coloring of circle graphs. *Discrete Applied Mathematics* **103**, 271–279.

AVRIEL, M. et PENN, M. (1993). Exact And Approximate Solution of the Conteneur Ship Stowage Problem. *Computer and Industrial Engineering* **25**, 271–274.

BENDERS, J.F. (1962). Partitioning Procedures for Solving Mixed-Variables Programming Problems. *Numerische Mathematik* **4**, 238–252.

BLASUM, U., BUSSIECK, M.R., HOCHSTÄTTLER, W., MOLL, C., SCHEEL, H.-H. et WINTER, T. (1999). Scheduling trams in the morning. *Mathematical Methods of Operations Research.* **49**, 137–148.

BIRGE, J R., et LOUVEAUX, F. (1997). Introduction to Stochastic Programming. Springer Series in Operations Research. New York.

CORDEAU, J.-F., SOUMIS, F. et DESROSIERS, J. (2000). A Benders Decomposition Approach for the Locomotive and Car Assignment Problem. *Transportation Science* **34**, 133–149.

DESAULNIERS, G., et HICKMAN, M.D. (2003). Public Transit. Technical Report, Les Cahiers du GERAD G-2003-77, HEC Montreal, Montreal, Canada . To appear in Handbooks in Operations Research and Management Science, Volume on Transportation, C. Barnhart and G. Laporte (eds.), Elsevier, Amsterdam, The Netherlands.

Freling, R., Lentink, R.M., Kroon, L.G., et Huisman, D. (2005). Shunting of Passenger Train Units in a Railway Station. *Transportation Science* **39**, 261–272.

Gallo, G. et Di Miele, F. (2001). Dispatching Buses in Parking Depots. *Transportation Science* **35**, 322–330.

Garey, M.R.et Johnson, D.S. (1979). *Computers and Intractability : A Guide to the Theory of NP-Completeness.* Freeman, W.H. New York.

Geoffrion, A.M. et Graves, G.W. (1974). Multicommodity Distribution System Design by Benders Decomposition. *Management Science* **20**, 822–844.

Hamdouni, M., Desaulniers, G., Marcotte, O., Soumis, F. et Van Putten, M. (2004). Dispatching Buses in a Depot using Block Patterns. Technical Report, Les Cahiers du GERAD G-2004-51, HEC Montreal, Montreal, Canada (2004). To appear in *Transportation Science.*

Hamdouni, M., Desaulniers, G., et Soumis, F. (2005). Parking buses in a depot using block patterns : a Benders decomposition approach for minimizing type mismatches. Technical Report, Les Cahiers du GERAD G-2005-70, HEC Montreal, Montreal, Canada (2005). To appear in *Computers & Operations Research.*

Hamdouni, M., Soumis, F., et Desaulniers, G. (2006). Parking buses in a depot with stochastic arrival times. Technical Report, Les Cahiers du GERAD G-2006-11, HEC Montréal, Canada (2005). To appear in *Transportation Science.*

KAUFMANN, L. et BROECKX, F. (1978). An algorithm for the quadratic assignment problem. *European Journal of Operational Research* **2**, 204–211.

MERCIER, A., CORDEAU, J.-F. et SOUMIS, F (2005). A Computational Study of Benders Decomposition for the Integrated Aircraft Routing and Crew Scheduling Problem. *Computers & Operations Research* **32**, 1451–1476.

SHERALI, H.D. et FRATICELLI, B.M.P (2002). A Modification of Benders' Decomposition Algorithm for Discrete Subproblems : An Approach for Stochastic Programs with Integer Recourse. *Journal of Global Optimization* **22**, 319–342.

WINTER, T. (1999). *Online and real-time dispatching problems.* Thèse de Doctorat., TU Braunschweig. Allemagne.

WINTER, T. et ZIMMERMANN, U.T. (2000). Real-time dispatch of trams in storage yards. *Annals of Operations Research* **96**, 287–315.