Multiflots dynamiques avec contraintes de synchronisation de ressources

Xavier Libeaut

l'université nantes angers le mans
PÔLE DE RECHERCHE ET D'ENSEIGNEMENT SUPÉRIEUR

université angers

Thèse de Doctorat

Xavier LIBEAUT

Mémoire présenté en vue de l'obtention du
grade de Docteur de l'Université d'Angers
sous le label de l'Université de Nantes Angers Le Mans

Discipline : Informatique (CNU 27)
Spécialité : Recherche Opérationnelle
Laboratoire : LISA (EA 4094)

Soutenue le 17/12/2013

École doctorale : STIM (503)
Thèse n° : 2013 UA 1384

Multiflots dynamiques avec contraintes de synchronisation de ressources

JURY

Rapporteurs :	**M. Louis-Martin ROUSSEAU**, Professeur, Ecole Polytechnique de Montréal, Québec **M. Dritan NACE**, Professeur, Heudiasyc UMR CNRS 6599, Compiègne
Examinateurs :	**M^{me} Christelle JUSSIEN-GUERET**, Professeur, LISA EA 4094, Angers **M. Boris DETIENNE**, Maître de Conférences, RealOpt INRIA, Bordeaux
Directeur de thèse :	**M. Eric PINSON**, Professeur, LISA EA 4094, Angers
Co-encadrants :	**M. Jorge E. MENDOZA**, Maître de Conférences, LISA EA 4094, Angers **M^{me} Sophie DEMASSEY**, Maître de Conférences, LINA CNRS UMR 6241, Nantes

Remerciements

Je tiens à adresser en premier lieu mes plus chaleureux remerciements à mon directeur de thèse Eric Pinson, Professeur à l'Institut de Mathématiques Appliquées. Il n'a pas simplement accepté de diriger ma thèse ; il n'a eu de cesse de m'encourager et de me soutenir durant ces trois années. Ses conseils avisés me permirent de découvrir la méthodologie de la recherche et de confirmer mon intérêt pour la recherche opérationnelle. J'en profite pour lui exprimer ici ma plus profonde gratitude.

Je tiens aussi à remercier mon co-encadrant Jorge E. Mendoza, Maître de Conférence à l'Institut de Mathématiques Appliquées, pour sa générosité et ses nombreuses idées distillées pendant ma thèse. Je le remercie aussi pour le soutien qu'il a pu m'apporter lors de moments de doutes.

Je remercie Monsieur Louis-Martin Rousseau, Professeur à l'Ecole Polytechnique de Montréal (Québec) et Monsieur Dritan Nace, Professeur à l'Université de Technologie de Compiègne de m'avoir fait l'honneur d'être les rapporteurs de cette thèse. J'adresse également toute ma gratitude à Madame Christelle Guéret-Jussien, Professeur à l'Université d'Angers et à Monsieur Boris Detienne, Maître de Conférence à l'Université de Bordeaux 1 pour leur participation à ce jury.

Mes remerciements vont également au projet régional LigéRO pour avoir financé cette thèse.

Je souhaite sincèrement dire merci aux personnes de l'Institut de Mathématiques Appliquées pour leur gentillesse, leur disponibilité et leur bonne humeur. Merci à Christine, David, Pierre, Dominique ainsi qu'à tous les autres.

Enfin, je remercie mes parents, ainsi que l'ensemble de ma famille d'avoir toujours été présents et de m'avoir encouragé dans ce projet personnel. Mes remerciements les plus sincères vont aussi à mes amis.

Mes derniers remerciements vont à Benoît, doctorant et ami avec qui j'ai partagé mon bureau. Ces années en sa compagnie furent le théâtre d'échanges stimulants sur nos problématiques respectives, mais aussi de moments de détente plus légers.

Table des matières

Nota Bene : Une partie des développements techniques inhérents au modèle multiflot et à l'heuristique sérielle ainsi qu'aux bornes inférieures s'y rapportant ont fait l'objet de recherches conjointes avec Benoît Tricoire.

1

Introduction

Sommaire

1.1 Contexte de l'étude

Dans le secteur concurrentiel et mondialisé de la messagerie express, la compétitivité d'une entreprise s'avère indissociable de processus optimisés. En France, selon une étude de l'ASLOG (association française pour la logistique), le coût logistique imputable au transport s'élève en moyenne à 5% du chiffre d'affaire. Plus particulièrement, dans le domaine de la messagerie, 60% des coûts incombant au transport [1]. Une logistique du transport performante est donc garante de la compétitivité des acteurs de ce secteur.

Dans l'industrie de la messagerie rapide, l'accent est mis sur la qualité du service, dont l'élément prépondérant est la rapidité d'acheminement des produits, avec généralement des livraisons au maximum deux jours après les collectes. Dans le cadre de ces contraintes de temps très limité l'un des leviers d'action majeurs réside dans l'optimisation de l'ordre d'acheminement des produits, en respect des différentes restrictions inhérentes au réseau (moyens logistiques disponibles, stratégie de massification impliquant des hubs, multi-modalité, etc.).

De nos jours, un contexte mondialisé, ainsi que l'apparition de l'informatique à tous les niveaux de la chaîne logistique, complexifie la gestion de la logistique du transport dans la messagerie. Dans ce contexte, la Recherche Opérationnelle devient souvent un outil d'aide à la décision indispensable pour améliorer la processus mis en jeu.

La problématique abordée traite du routage optimal d'un ensemble de flots élémentaires sur un horizon de temps fixé dans un réseau logistique sur les sites duquel ces flots

ont à subir certains traitements. Une flotte illimitée mais typée de véhicules permet le routage de ces flux contraints par des contraintes temporelles. L'objectif est de déterminer une stratégie de moindre coût permettant l'acheminement de ces flux.

Ces recherches s'articulent dans le prolongement d'une étude ayant été réalisée par l'Institut de Mathématiques Appliquées (IMA) entre 2005 et 2009 pour le compte du groupe La Poste, et portant sur l'optimisation du réseau national d'acheminement du courrier. L'objectif de ces travaux est de proposer des approches de résolution pour un modèle général et de les valider sur des jeux de données représentatifs d'organisations réelles, tant en termes de structure du réseau logistique, qu'en termes de volumétrie. Ce doctorat a été financé par la Région des Pays de la Loire dans le cadre du projet LigéRO dont nous explicitons le rôle ci-après.

1.2 Le projet LigéRO

Dans les Pays de Loire, une cinquantaine de chercheurs, enseignants-chercheurs et ingénieurs R&D relèvent spécifiquement du domaine de la Recherche Opérationnelle, ce qui représente l'un des plus forts potentiels en France. Disséminés dans différents laboratoires et entreprises de la région (IRCCyN, LERIA, LINA, LISA/IMA, Optilogistic, EuriSys, etc.), ils développent des expériences variées, en investiguant divers domaines d'application et approches de résolution.

L'objectif du projet régional LigéRO (figure 1.1), décroché dans le contexte de l'AAP 2009, est la création d'un groupe de recherche régional fédérant les différents acteurs de la discipline permettant de positionner les Pays de la Loire comme un des pôles incontournables en Recherche Opérationnelle. Il vise notamment à :

- favoriser et systématiser de manière mesurable et pérenne les collaborations entre équipes,
- favoriser la diffusion des connaissances et identifier les Pays de la Loire comme un pôle d'excellence en Recherche Opérationnelle à un niveau national comme international,
- préparer les équipes impliquées pour le dépôt de projets ANR et européens majeurs accessibles en raison de leur excellence,
- renforcer les enseignements en Recherche Opérationnelle et promouvoir la visibilité internationale des formations en Recherche Opérationnelle proposées dans la région.

1.3 Formulation de la problématique

Notre étude s'inscrit sur un horizon temporel court, typique de la messagerie express (24 ou 48 heures). Au sein de celui-ci, un ensemble de produits doivent être acheminés au moindre coût dans le réseau logistique décrit ci-après.

FIGURE 1.1 – Le projet LigéRO : un groupe de recherche régional fédérant de nombreux laboratoire

1.3.1 Le réseau logistique

Le réseau logistique ciblé dans cette étude est composé de centres de traitement (CT) et de hubs. A partir de ces CT, des flux de produits sont émis vers d'autres CT destination où ils subissent un traitement adapté afin d'être ensuite acheminés jusqu'à des destinataires finaux. Le transfert direct de produits entre CT n'est pas intéressant financièrement. Ainsi, à des fins de diminution du coût logistique, les flux à destinations communes sont massifier en recourant à un ensemble de hubs, positionnés dans le réseau dans le cadre d'une étude stratégique supposée sortir du périmètre de ces travaux. Plus précisément, les flux produits transitent par un ou plusieurs hubs de natures différentes : hubs routiers, ferroviaires ou aériens, mais aux caractéristiques de production identiques. La structure du réseau considéré permet alors de distinguer des liaisons directes, dont chacune implique nécessairement au moins un hub. Une telle liaison physique décline une distance, ainsi qu'un durée de parcours. La figure 1.2 illustre le réseau logistique considéré.

1.3.2 Les flux produits

Chaque produit doit être acheminé depuis un CT origine vers un CT destination, en respect d'une date de disponibilité sur le CT origine ainsi que d'une date échue sur le CT destination.

L'acheminement d'un produit nécessite de visiter au moins un hub de massification. La séquence de sites visités par un produit est figée est sera nommée routage primaire du produit. Elle résulte d'une étude stratégique sortant du périmètre de cette étude.

Une quantité, exprimée en Unité Equivalent Transport (UET) est associée au produit et sur chaque liaison caractérisant son routage primaire, le produit peut être acheminé en plusieurs départs répartis sur l'horizon temporel (contexte de fractionnement des quantités).

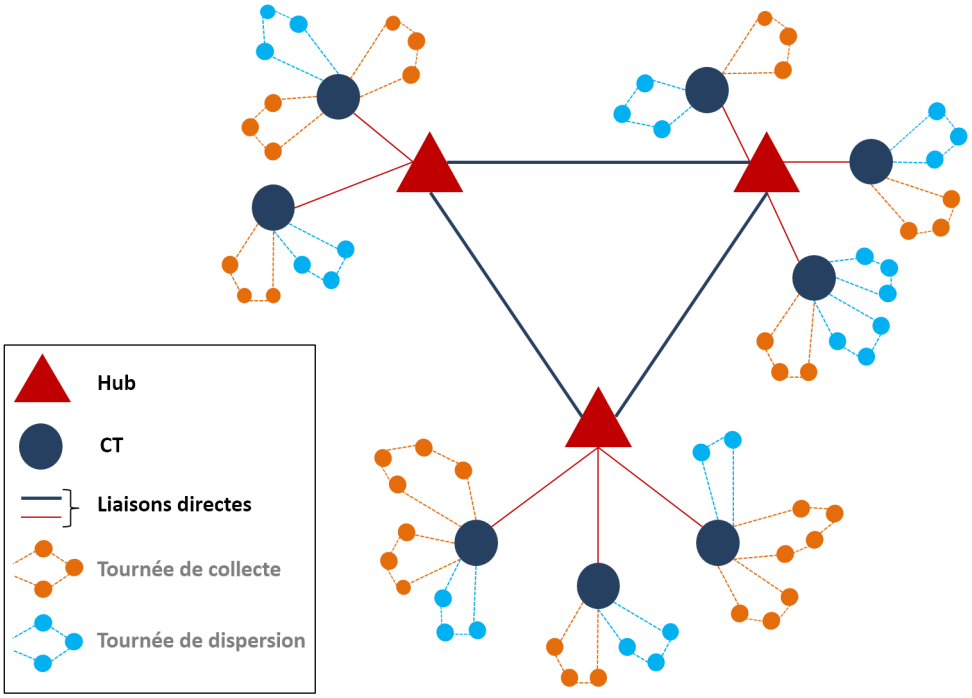


FIGURE 1.2 – Le réseau logistique étudié

1.3.3 Les moyens logistiques

L'acheminement des flux est réalisé par le biais de trois modes de transport : route, fer, air. Une flotte de véhicules hétérogène illimitée permet la desserte des plates-formes et hubs routiers. Différents types de moyens sont considérés, chacun d'eux étant caractérisé par une capacité, un coût fixe kilométrique, ainsi qu'un profil de vitesse.

Les gares et escales sont, quant à elles, approvisionnées par le biais d'une flotte connue et limitée de trains et d'avions dont les horaires sont planifiés. A l'instar des moyens routiers, les coûts kilométriques, capacités de transport et profils de vitesse sont connus pour les trains et avions. Pour ces derniers, un coût variable d'utilisation de l'appareil proportionnel à la charge transportée est également décliné.

1.3.4 Le traitement des flux

Outre les aspects transport, nous considérons les traitements des flux sur les sites logistiques du réseau : un traitement de tri concernant tous les flux arrivant sur un CT destination et un traitement de « cross-dock » pour les produits transitant par hubs. Chaque CT destination possède ainsi une plage horaire de traitement dédiée ainsi qu'une capacité de traitement (exprimée dans l'UET retenue) spécifiée pour chaque période l'horizon temporel. De même, chaque hub possède une plage horaire de traitement ainsi qu'une capacité de traitement.

1.3.5 Objectif

L'objectif est de déterminer un acheminement au moindre coût des produits par synchronisation des moyens logistiques utilisés, en respect des capacités de ces derniers, des capacités de traitement limitées sur chaque site du réseau et des dates de disponibilité et des dates échues associées aux flux

Notre problématique s'inscrit dans la famille des problèmes de multiflots et s'inscrit plus spécifiquement dans les problèmes dits de conception de plan de chargement (Load Plan Design), pour lesquels nous proposons un état de l'art dans la section 1.4 qui suit.

1.4 Etat de l'art

Les problèmes de flots furent initiés par Ford et Fulkerson en 1956 sur la base du problème de flot maximal [8]. Dans ce problème, un graphe orienté dont chaque arc possède une capacité est considéré. L'objectif est alors de déterminer le flot maximal pouvant transiter depuis un nœud source vers un nœud destination en respect des capacités des arcs. Ce problème s'applique à des domaines aussi variés que le trafic routier, les télécommunications, les circuits électriques, le traitement d'images, etc.

Peu de temps après, en 1961, Gomory et Hu [13] posent le problème du réseau avec multi-terminal en vue de déterminer le flot maximal entre chaque paire de nœuds du réseau. Hu [15] introduit en 1963 le premier problème de multiflots, dont l'objectif est de maximiser deux flots simultanés avec deux sources et deux destinations distinctes. En 1966, Tomlin [24] généralise ce problème à plus de deux flots simultanés et cherche à maximiser l'ensemble des flots. De nos jours, la majorité des problèmes de multiflots considère des flots possédant, en plus d'une source et d'une destination, une quantité donnée. Notre étude s'inscrit dans ce type de problématiques, dont l'objectif est de satisfaire le routage d'un ensemble des flots au moindre coût.

Dans le contexte de l'optimisation du transport, le coût minimisé décline plusieurs composantes, fixes ou variables. La partie fixe est généralement induite par la mobilisation de moyens logistiques. Ceci peut correspondre à une sous-traitance de flottes de véhicules, un coût de vérification de l'appareil pour le mode aérien, etc. La partie variable du coût dépend de la distance parcourue et, pour certains modes de transport (notamment l'aérien et le fer), de la quantité acheminée. Les travaux de Grünert et al. [14], Barnhart et Shen [3], Crainic et Kim [5], Jarrah et al. [16] et Erera et al. [7] s'inscrivent dans ce type de valorisation. Gabrel et Minoux [11] , Gabrel et al. [9, 10] et Minoux [20] se focalisent plus spécifiquement sur les fonctions de coûts à pas discret et exposent des méthodes de résolution adaptées à ce cas particulier, souvent inscrit dans des problématiques appliquées au domaine des télécommunications.

Les fonctions de coûts à pas discret interviennent également dans les problèmes d'optimisation des flux dans le domaine du transport. En effet, le coût d'acheminement d'un produit varie de manière discrète en fonction du nombre de véhicules mobilisés. En 1999, Gabrel et al. [9] proposent une méthode de résolution exacte pour ce type de problématiques. Il s'agit d'une adaptation de la technique de décomposition de Benders consistant à résoudre itérativement des programmes linéaires en 0-1 dans un contexte de génération de contraintes. Les réseaux considérés dans cette étude demeurent néanmoins de taille modeste puisque qu'ils déclinent au plus 20 nœuds et 37 liaisons.

L'optimisation des flots au sein de réseaux fait intervenir essentiellement deux types de problématiques :

- la conception du réseau,
- la planification des flots au sein du réseau.

1.4.1 Conception des réseaux

La conception d'un réseau, modélisé sous forme de graphe, vise principalement à déterminer la position des nœuds dans ce réseau ainsi que l'existence et la nature des liaisons les joignant. Appliquées à l'optimisation du transport, ces problématiques relèvent d'un niveau stratégique et concernent le placement d'entrepôts, de hubs de massification, etc. Les problèmes de conception de réseaux s'appliquent notamment au domaine du transport ou des télécommunications. A ce titre, Gendron et al. [12], Crainic [4] et Crainic et Kim [5] exposent un état de l'art des modélisations et des algorithmes relatifs à la conception de ces types de réseaux.

Dans le domaine de la messagerie, la conception des réseaux est abordée en 1999 par Grünert et al. [14]. La phase de conception d'un tel réseau implique souvent des problèmes de localisation, étudiés notamment en 2003 par Daskin et Owen [6]. Pour une vision globale de la conception de réseaux, nous en référons aux travaux de Magnanti et Wong [18] ainsi que de Minoux [19].

1.4.2 Planification des flux au sein des réseaux

L'étape suivant la conception d'un réseau relève de la planification des services au sein de celui-ci.

Grünert et al. [14] étudient en 1999 la planification des flux dans les réseaux de messagerie. Ils segmentent leur problème en deux parties et distinguent la planification des flux au sein d'un sous-réseau inter-hubs de celle plus locale portant sur des sous-réseaux de collecte et de dispersion. Leurs développements abordent non seulement la planification des moyens logistiques mais aussi l'affectation des chauffeurs. Des dates de disponibilité et des dates échues sont associées aux flux à acheminer. Au même titre que notre étude, des capacités de traitement limitées sur les hubs ainsi que des temps de traitement liés aux opérations de tri et de cross-docking sont considérés. Dans un contexte multi-modal, le réseau décline des liaisons routières sur les étages de collecte et dispersion ainsi que des liaisons aériennes sur l'étage inter-hubs. L'objectif est de déterminer une stratégie de routage des flux au moindre coût. A ce titre, la fonction objectif intègre des coûts de transport, déclinés en coûts d'embarquement pour les liaisons aériennes et en coûts kilométriques pour les liaisons routières, ainsi que des coûts liés à l'utilisation des hubs. Enfin, les auteurs exploitent une méthode taboue pour résoudre le problème.

En 2005, Barnhart et Shen [3] exposent également une problématique dans le domaine de la messagerie express s'inscrivant sur un horizon temporel de 2 jours. Les auteurs proposent un modèle suffisamment générique pour être transposé à d'autres problématiques (multiflots, affectation de chauffeurs, etc.). L'étude cible des réseaux de grande taille et plusieurs techniques de génération de colonnes sont proposées pour résoudre le problème.

Plus récemment, en 2013, Erera et al. [7] abordent un problème d'acheminement de marchandises dans le domaine de la messagerie. Le réseau décline là encore des sites

clients et des hubs de massification. L'horizon temporel considéré est discrétisé en seulement 8 périodes. Le problème implique non seulement de router les flux au coût minimal mais aussi de gérer le repositionnement des véhicules vides. Contrairement à notre étude, les auteurs ne considèrent pas de capacités de traitement sur les sites du réseau mais seulement une latence forfaitaire associée aux opérations de cross-docking. Notons que le choix de la séquence de hubs visités par un produit à acheminer depuis son nœud origine jusqu'à son nœud destination est une composante du problème. Pour formaliser ce dernier, les auteurs introduisent un modèle basé sur un graphe indexé dans le temps. Pour le résoudre, ils proposent une recherche locale basée sur un programme linéaire en nombres entiers.

En 2009, Jarrah et al. [16] étudient eux aussi un problème d'acheminement de marchandises au sein de réseaux composés de terminaux origine et destination ainsi que des hubs de massification. Les réseaux considérés sont de grande taille (plus de 700 nœuds). Au même titre que Erera et al. [7], le repositionnement des véhicules vides est géré et valorisé dans la fonction objectif. Cette dernière intègre également des coûts associés aux traitements de tri et de cross-docking sur les hubs, ainsi que des coûts kilométriques. Dans ce problème, la séquence d'arcs empruntée pour router un produit n'est pas déterminée à l'avance. Un produit peut être acheminé depuis sa source vers sa destination de manière directe ou peut visiter jusqu'à 2 hubs de massification. Aucune capacité de traitement n'est considérée mais chaque produit subit un traitement d'une durée déterminée lorsqu'il transite par un hub. Une décomposition spatiale est introduite et génère des sous-problèmes associés aux sites destination des flux. Une heuristique de slope-scaling est mise en œuvre pour résoudre le problème. Elle implique de résoudre des programmes linéaires continus pour lesquels les auteurs proposent une technique de génération de colonnes.

En conclusion, la littérature comprend des problèmes très similaires au notre ([3, 7, 14, 16]), mais ils considèrent un découpage de l'horizon de temps avec peu de périodes. Or, certaines applications pratiques, telles que la messagerie express, nécessitent une plus petite granularité du temps. De plus, à notre connaissance, seul Grünert et al. [14] considèrent des capacités de traitement sur les sites logistiques, ces contraintes pouvant devenir importantes lorsque le niveau de produits atteint un certain seuil.

Dans cette thèse, nous proposons diverses techniques heuristiques basées pour certaines sur des MIP ou leur relaxation linéaire afin de résoudre notre problématique.

Formalisation et modélisation

2

Formalisation de la problématique

Sommaire

2.1 Introduction

Afin de simplifier quelques peu cette problématique, un certain nombre d'hypothèses sont prises en compte :

- chaque produit admet un routage primaire unique,
- seul le cas monomodal est considéré et le mode de transport retenu est la route,
- ce qui induit que seul les coûts fixes sont considérés,
- les dates au plus tard de fin de traitement des produits à destination d'un même CT sont identiques,
- le splitting des produits durant leurs différentes phases de traitement (hubs ou CT destination) ainsi que de transfert inter-sites est autorisé, ce qui englobe aussi bien la préemption que le chevauchement de fractions de produits,

La suite de ce chapitre formalise le problème en tenant compte des hypothèses énoncées ci-dessus.

2.2 Réseau logistique

Rappelons que le réseau logistique ciblé dans cette étude est composé de centres de traitement (CT) et de hubs. A partir de ces CT, des flux produit sont émis vers d'autres CT destination où ils subissent un traitement adapté afin d'être ensuite acheminés jusqu'à des destinataires finaux. L'ensemble des sites logistiques impliqués dans ce réseau sera noté S et partitionné en centres de traitements (C) et hubs (H) : $S = C \cup H$.

2.3 Flux

L'horizon temporel de planification Θ sous-jacent à cette étude est discrétisé en τ périodes selon un pas de temps (mn, $\frac{1}{4}$ h, $\frac{1}{2}$ h, h, etc.). Les flux sont banalisés au travers de la notion de produit. Un produit $p \in P$ est caractérisé par un 6-uplet [$s(p)$, $d(p)$, $q(p)$, $es(p)$, $lc(p)$, $RP(p)$] où :

- $s(p)$ désigne son CT origine,
- $d(p)$ son CT destination,
- $q(p)$ la quantité de produit associée exprimée en UET (Unité Equivalent Transport),
- $es(p)$ sa date au plus tôt de disponibilité sur le CT origine,
- $lc(p)$ sa date au plus tard de fin de traitement sur le CT destination,
- $RP(p)$ son routage primaire (notion définie ultérieurement).

2.4 Moyens de transport

Une flotte de véhicules hétérogène illimitée V permet le routage des produits au sein du réseau logistique. Différents types de moyens sont considérés, chacun d'eux étant caractérisé par les attributs [$cp(v)$, $cf(v)$], où :

- $cp(v)$ désigne la capacité du véhicule $v \in V$ (exprimée dans l'UET (Unité Equivalent Transport) retenue),
- $cf(v)$ son coût kilométrique.

Nous considérons deux types de véhicules. Une décomposition spatiale du réseau introduite par la suite permet de définir des étages C-H, H-H et H-C. Par hypothèse, la capacité du type de véhicule associé à chaque liaison des étages C-H ou H-C (resp. H-H) est constante et sera notée Q_C (resp. Q_H).

2.5 Traitement des flux

Outre les aspects transport, nous considérons les traitements des flux sur les sites logistiques du réseau. Un traitement de tri est opéré sur les produits arrivant sur un CT destination tandis que ceux transitant par des hubs subissent un traitement de cross-docking. Chaque CT destination $c \in C$ possède ainsi une plage horaire de traitement dédiée notée $\Theta_c = [\theta_c^-, \theta_c^+]$ ainsi qu'une capacité de traitement η_c constante sur l'horizon temporel, exprimée par période dans l'UET retenue. De même, chaque hub $h \in H$ possède une plage horaire de traitement $\Theta_h = [\theta_h^-, \theta_h^+]$ et une capacité de traitement η_h.

2.6 Routages primaires

Nous définissons un routage primaire $RP(p)$ associé à un produit $p \in P$ comme la séquence ordonnée de sites logistiques visités par ce produit lors de son acheminement de son CT origine vers son CT destination, via un transit éventuel par un ou deux hubs de transfert baptisés hubs origine et destination. L'ensemble des produits est partitionné en deux types. Pour un produit p de type 1, le routage primaire associé $RP(p)$ est $s(p) \rightarrow h(p) \rightarrow d(p)$. Pour un produit de type 2, il est de la forme $s(p) \rightarrow h_o(p) \rightarrow h_d(p) \rightarrow d(p)$, $h_o(p)$ (resp. $h_d(p)$) désignant le hub origine (resp. destination) du routage primaire du produit p. Par soucis de simplification, pour un routage primaire 1-hub associé à un produit p, $h_o(p)$ et $h_d(p)$ désignerons par la suite $h(p)$. La figure 2.1 synthétise les deux types de routages primaires considérés. Par la suite, l'ensemble des produits du premier type nommé 1-hub sera notée P_{t1} et l'ensemble des produits du second type appelé 2-hubs sera notée P_{t2}.

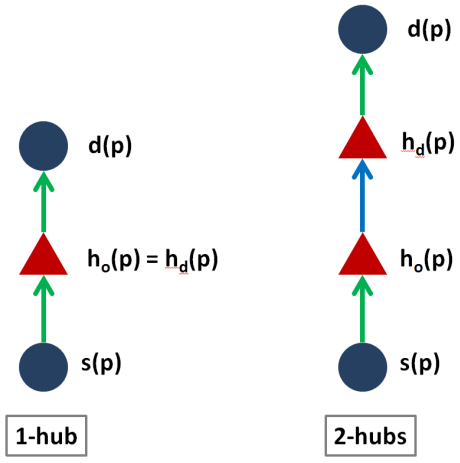


FIGURE 2.1 – Routages primaires 1-hub et 2 -hubs

2.7 Segments de transport

Chaque paire de sites consécutifs participant à un tel routage est nommé segment de transport (ST). On notera par la suite Γ l'ensemble des segments de transport associés au réseau logistique considéré. Pour tout ST $\nu \in \Gamma$, on notera $o(\nu)$ (resp. $d(\nu)$) les sites origine et destination associés. Ces segments de transport sont partitionnés en trois étages : C-H, H-H et H-C, correspondant respectivement aux liaisons CT origine vers hub origine, inter-hubs et hub destination vers CT destination. On notera $l(s, s')$ (ou $l_{s,s'}$) la durée de roulage pour rallier le site j depuis le site i ($s, s' \in S$). La figure 2.2 illustre un exemple de distribution de segments de transport dans un réseau logistique.

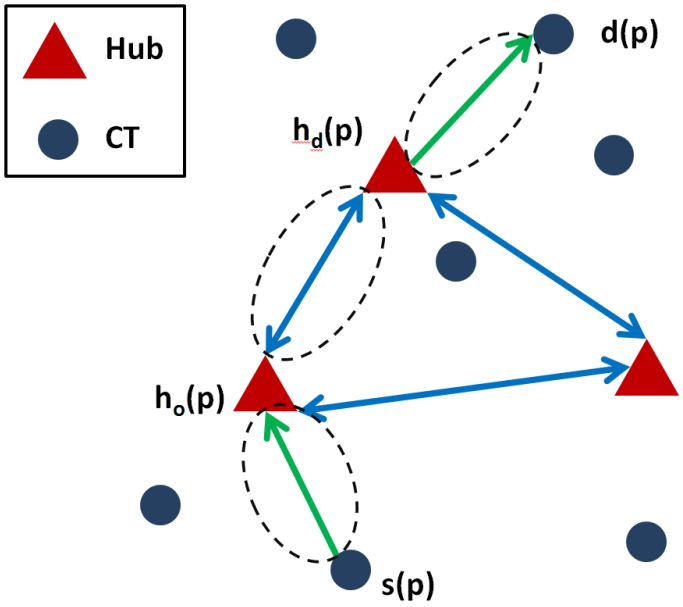


FIGURE 2.2 – Exemple de réseau avec segments de transport

2.8 Ordres de Transport

Un Ordre de Transport (OT) correspond à la sélection et au "calage" temporel d'un type de véhicule sur un segment de transport. Chaque OT ω est ainsi caractérisé par les attributs suivants :

- $\nu(\omega) \in \Gamma$: segment de transport sur lequel porte ω,
- $st(\omega)$ (resp. $ct(\omega)$) : date de départ du site origine du ST associé (resp. d'arrivée au site destination),
- $w(\omega)$: coût de l'OT, égal à la distance entre les sites origine et destination du segment de transport référant, multiplié par le coût kilométrique du type de véhicule associé à ce segment de transport,
- $cp(\omega)$: capacité du type de véhicule associé au segment de transport référant.

A noter que les attributs de coût et de capacité sont directement induits par le type de la liaison, la distance et le type de véhicules étant alors connus. De même, le segment transport $\nu(\omega)$ référant un OT ω conditionne les dates de visite des sites logistiques impliqués. Nous remarquons aussi que les OT portant sur un même segment de transport ont un coût identique. Dans la suite, on notera Ω l'ensemble de tous les ordres de transport induits par les caractéristiques physiques et temporelles du réseau.

2.9 Routages

Définissons enfin un routage $\sigma = [\omega_1, ..., \omega_q]$ comme une séquence ordonnée admissible de $q \leq 3$ ordres de transport. L'admissibilité d'un routage est relative à la composante temporelle :

$$\forall j \in [2, q], st(\omega_j) \geq ct(\omega_{j-1}) + \epsilon_T \tag{2.1}$$

où ϵ_T désigne une latence forfaitaire considérée comme égale à une période par la suite. En fait, un routage correspond simplement à la concaténation de segments de transport par intégration de la composante temporelle relative à la synchronisation de moyens de transport. Ainsi, un routage σ est admissible pour un produit p si :

- La séquence d'OT le définissant est compatible avec le routage primaire imposé pour p. Par exemple, pour un produit de type 2 (2-hubs), on aura $\sigma = [\omega_1, \omega_2, \omega_3]$. Pour que σ soit admissible pour p, le ST de collecte $\nu(\omega_1)$ doit partir de $s(p)$ (i.e., $o(\nu(\omega_1)) = s(p)$) et le ST $\nu(\omega_3)$ de dispersion doit aboutir sur le CT destination $d(p)$ (i.e., $d(\nu(\omega_3)) = d(p)$). En outre, nous avons nécessairement $d(\nu(\omega_1)) = o(\nu(\omega_2)) = h_o(p)$ et $d(\nu(\omega_2)) = o(\nu(\omega_3)) = h_d(p)$.
- Le calage temporel des OT constitutifs du routage est compatible avec les dates de disponibilité $es(p)$ et échue $lc(p)$ du produit p, autrement dit : $st(\omega_1) \geq es(p)$ et $ct(\omega_3) \leq lc(p)$.

L'ensemble des routages admissibles induit par Ω est noté R. Un routage $\sigma \in R$ est représenté par son indicatrice sur l'ensemble Ω des ordres de transport : $\forall \omega \in \Omega, a^\sigma_\omega = 1$ si l'OT ω participe au routage σ, 0 sinon. Notons R^p le sous-ensemble des routages admissibles pour le produit $p \in P$, par induction $R = \bigcup_{p \in P} R^p$. De plus, pour tout site $s \in S$, R_s désigne le sous-ensemble de routages transitant par s.

2.10 Notations additionnelles

Dans la suite de nos développements, nous exploitons les notations additionnelles suivantes :

- $\forall c \in C, P^+_c = \{p \in P / s(p) = c\}$: produits au départ du CT c,
- $\forall c \in C, P^-_c = \{p \in P / d(p) = c\}$: produits à destination du CT c,
- $\forall c \in C, \forall h \in H, P_{c,h} = \{p \in P / s(p) = c \wedge h_o(p) = h\}$: produits au départ du CT c et transitant par le hub h en tant que hub origine,
- $\forall h \in H, \forall c \in C, P_{h,c} = \{p \in P / h_d(p) = h \wedge d(p) = c\}$: produits à destination du CT c en provenance du hub destination h,
- $\forall (h, h') \in H^2 / h \neq h', P_{h,h'} = \{p \in P / h_o(p) = h \wedge h_d(p) = h'\}$: produits de type 2 transitant par la liaison inter-hubs $h - h'$,
- $\forall h \in H, C^-_h = \{c \in C / \exists p \in P / s(p) = c \wedge h_o(p) = h\}$: ensemble des CT ayant h pour hub origine,
- $\forall h \in H, C^+_h = \{c \in C / \exists p \in P / h_d(p) = h \wedge d(p) = c\}$: ensemble des CT ayant h pour hub destination,
- $\forall c \in C, H^-_c = \{h \in H / \exists p \in P / h_d(p) = h \wedge d(p) = c\}$: hubs connectés à c sur l'étage H-C,

- $\forall h \in H, H_h^- = \{h' \in H / h' \neq h \wedge (\exists p \in P / (h_o(p) = h' \wedge h_d(p) = h)\}$: hubs origine connectés au hub destination h via une liaison inter-hubs,
- $\forall h \in H, H_h^+ = \{h' \in H / h' \neq h \wedge (\exists p \in P / h_o(p) = h \wedge h_d(p) = h')\}$: hubs destination connectés au hub origine h via une liaison inter-hubs,
- Ω_{CH} (resp. Ω_{HH}, Ω_{HC}) : OT de l'étage C-H (resp. H-H et H-C),
- $\forall s \in S, \Omega_s$: ordres de transport portant sur un segment de transport de site origine ou destination s,
- $\forall (s, s') \in S^2, \Omega_{s,s'}$: ordres de transport portant sur la liaison inter-sites (s,s'),
- $\forall s \in S, \Omega_s^-$: ordres de transport de site destination s,
- $\forall s \in S, \Omega_s^+$: ordres de transport de site origine s,
- $\forall \theta \in \Theta, \forall s \in S, R_s^\theta$: routages arrivant sur le site s durant la période θ,
- $\forall s \in S, P_s$: sous-ensemble de produits transitant par le site s,
- Γ_{CH} (resp. Γ_{HH}, Γ_{HC}) : ensemble de segments de transport portant sur l'étages C-H (resp. H-H et H-C).

3

Une formalisation mathématique multiflots du problème

Sommaire

3.1 Introduction

Nous proposons, dans ce chapitre, une première formalisation de la problématique étudiée exploitant les notions d'ordres de transport et de routages. Nous l'appellerons modélisation multiflots (MMF).

3.2 Modélisation mathématique

Introduisons les variables de décision :
– $\forall \omega \in \Omega, y_\omega$: nombre d'ordres de transport ω sélectionnés, chaque OT sélectionné donnant lieu à un départ de véhicule sur son segment de transport référant,
– $\forall p \in P, \forall \sigma \in R^p, x_p^\sigma$: quantité de produit p transitant par le routage σ.
Nous aboutissons alors à la formalisation "compacte" $[P_{MMF}]$.

$$[P_{MMF}] \qquad \min \sum_{\omega \in \Omega} w(\omega) y_\omega$$

SC

$$\forall p \in P, \qquad \sum_{\sigma \in R^p} x_p^\sigma = q(p) \tag{3.1}$$

$$\forall \omega \in \Omega, \qquad \sum_{p \in P} \sum_{\sigma \in R^p} a_\omega^\sigma x_p^\sigma \leq cp(\omega) y_\omega \tag{3.2}$$

$$\forall s \in S, \qquad [CCT_s] \tag{3.3}$$

$$\forall \omega \in \Omega, \qquad y_\omega \in \mathbb{N} \tag{3.4}$$

$$\forall p \in P, \forall \sigma \in R^p, \qquad x_p^\sigma \geq 0 \tag{3.5}$$

Dans cette formalisation, le groupe de contraintes 3.1 stipule que l'intégralité de chaque produit doit être routée dans le réseau. Les contraintes 3.2 contraignent le flux total de produits empruntant la liaison associée à chaque OT ω à ne pas excéder la capacité cumulée des véhicules activés sur cette liaison. Les contraintes 3.3 sont, quant à elles, relatives au Contrôle de Capacité de Traitement (CCT) que nous explicitons ci-après.

3.3 Contrôle de capacité de traitement sur les hubs

Considérons un hub particulier $h \in H$. Rappelons que Ω_h^- (resp. Ω_h^-) désigne le sous-ensemble d'OT d'origine (resp. de destination) le hub h. Définissons alors $R_h(\theta, \theta')$ le sous-ensemble de routages transitant par h durant l'intervalle de temps $[\theta, \theta']$:

$$\begin{aligned} R_h(\theta, \theta') = \{\sigma \in R_h / (\exists \omega \in \Omega_h^- / a_\omega^\sigma = 1 \wedge ct(\omega) \geq \theta) \\ \wedge (\exists \omega \in \Omega_h^+ / a_\omega^\sigma = 1 \wedge st(\omega) \leq \theta')\} \end{aligned} \tag{3.6}$$

Proposition 1 *Le respect des contraintes de capacité de traitement des produits sur le hub h est équivalent à :*

$$[CCT_h] : \ \forall (\theta, \theta') \in \Theta_h^2 / \theta \leq \theta', \quad \sum_{p \in P_h} \sum_{\sigma \in R_h(\theta, \theta')} x_p^\sigma \leq (\theta' - \theta + 1) \eta_h \tag{3.7}$$

Preuve 1 *Toute quantité x_p^σ de produit p impliqué dans un routage de $\sigma \in R_h(\theta, \theta')$ doit clairement être traitée dans l'intervalle de temps $[\theta, \theta']$. Sur cette période, la capacité cumulée de traitement est $(\theta' - \theta + 1)\eta_h$. La condition $[CCT_h]$ impose donc que, pour tout intervalle $[\theta, \theta']$, le flux cumulé de produits devant être traité sur le hub h n'éxcède pas la capacité cumulée de traitement sur cette même période.* ■

Remarque 1 *Dans le cas où les ordres de transport activés pour desservir le hub h sont connus, ce contrôle peut être réduit en termes de nombre de contraintes impliquées. Introduisons à cet effet les notations additionnelles : $\Theta_h^- = \{st(\omega) / y_\omega > 0 \wedge \omega \in \Omega_h^+\}$ et $\Theta_h^+ = \{ct(\omega) / y_\omega > 0 \wedge \omega \in \Omega_h^-\}$, correspondant respectivement aux différentes périodes de départ et d'arrivée des OT activés pour desservir le hub h. Nous avons alors :*

$$[CCT_h] : \ \forall (\theta, \theta') \in \Theta_h^- \times \Theta_h^+ / \theta \leq \theta', \quad \sum_{p \in P_h} \sum_{\sigma \in R_h(\theta, \theta')} x_p^\sigma \leq (\theta' - \theta + 1) \eta_h \tag{3.8}$$

3.4 Contrôle de capacité de traitement sur les CT destination

Par hypothèse, les flux à destination d'un même centre de traitement c ont la même date échue θ_c^+, cette dernière coïncidant avec la fin de la plage de traitement sur le CT considéré (au besoin par simple réajustement de celle-ci). Ainsi, le contrôle de capacité de traitement sur un CT destination peut être sensiblement simplifié. Pour un CT c quelconque, rappelons que P_c^- désigne le sous-ensemble de produits à destination du CT c. Notons $\Delta_c = \sum_{p \in P_c} q(p)$ la quantité cumulée de produits à destination de c et $\phi_{c,\theta}$ le flux cumulé de produits arrivé en c en fin de période θ. Rappelons que R_c^θ désigne le sous-ensemble de routages transitant par c durant la période θ. Nous obtenons la relation de récurrence :

$$\begin{cases} \phi_{c,0} = 0 \\ \phi_{c,\theta} = \phi_{c,\theta-1} + \sum_{p \in P_c^-} \sum_{\sigma \in R_c^\theta} x_p^\sigma \end{cases} \tag{3.9}$$

Corrélativement à Θ_h^- et Θ_h^+ définis pour énoncer les contraintes de traitement associées aux hubs, soit $\Theta_c = \{ct(\omega)/y_\omega > 0 \wedge \omega \in \Omega_c^-\}$.

Proposition 2 *Le respect des contraintes de traitement des produits sur le CT c est équivalent à :*

$$[CCT_c] : \ \forall \theta \in \Theta_c, \quad \phi_{c,\theta} \geq \Delta_c - (\theta_c^+ - \theta + 1)\eta_c \tag{3.10}$$

Preuve 2 *En effet, dans le pire cas, η_c unités de produits peuvent être traitées sur le CT c sur chaque période θ de la plage de traitement Θ_c. Ainsi, pour respecter les contraintes de dates échues inhérentes aux produits en sortie de c, les Δ_c unités de produits doivent être arrivées à la fin de la période θ_c^+, au moins $\Delta_c - \eta_c$ unités en fin de $\theta_c^+ - 1$, et ainsi de suite.* ■

Remarque 2 *Comme pour les hubs, les contrôles de traitements relatifs aux CT destination s'expriment en fonction des variables de flux x_p^σ. A ce titre, notons $R_c(\theta)$ le sous ensemble de routages arrivant au plus tard lors de la période θ sur le CT c :*

$$R_c(\theta) = \{\sigma \in R_c / \exists \omega \in \Omega_c^-, a_\omega^\sigma = 1 \wedge ct(\omega) \leq \theta\} \tag{3.11}$$

Sur la base de ce sous-ensemble de routage $R_c(\theta)$, l'expression des contraintes de capacité de traitement sur le CT destination c devient :

$$[CCT_c] : \ \forall \theta \in \Theta_c, \sum_{p \in P_c^-} \sum_{\sigma \in R_c(\theta)} x_p^\sigma \geq \Delta_c - (\theta_c^+ - \theta + 1)\eta_c \tag{3.12}$$

La figure 3.1 illustre l'évolution de la quantité de produit devant arriver sur le CT c avant sa date de fin de mise à disposition.

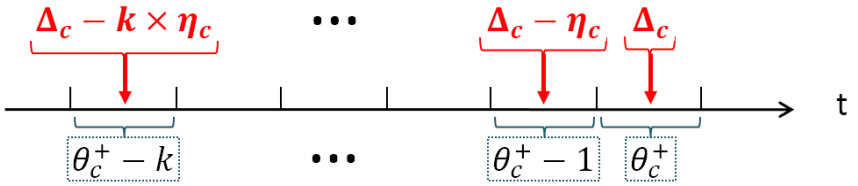


FIGURE 3.1 – Evolution des quantités de produits devant arriver sur un CT destination c

3.5 Reformulation de $[P_{MMF}]$

Le modèle général, intégrant explicitement les contraintes de traitement est alors le MIP $[P_{MMF}]$.

$$[P_{MMF}] \qquad \min \sum_{\omega \in \Omega} w(\omega) y_\omega$$

SC

$$\forall p \in P, \qquad \sum_{\sigma \in R^p} x_p^\sigma = q(p) \qquad (3.1)$$

$$\forall \omega \in \Omega, \qquad \sum_{p \in P} \sum_{\sigma \in R^p} a_\omega^\sigma x_p^\sigma \leq cp(\omega) y_\omega \qquad (3.2)$$

$$\forall h \in H, \forall (\theta, \theta') \in \Theta_h^2 / \theta \leq \theta', \qquad \sum_{p \in P_h} \sum_{\sigma \in R_h(\theta, \theta')} x_p^\sigma \leq (\theta' - \theta + 1)\eta_h \qquad (3.7)$$

$$\forall c \in C, \forall \theta \in \Theta_c, \qquad \sum_{p \in P_c^-} \sum_{\sigma \in R_c(\theta)} x_p^\sigma \geq \Delta_c - (\theta_c^+ - \theta + 1)\eta_c \qquad (3.12)$$

$$\forall \omega \in \Omega, \qquad y_\omega \in \mathbb{N} \qquad (3.4)$$

$$\forall p \in P, \forall \sigma \in R^p, \qquad x_p^\sigma \geq 0 \qquad (3.5)$$

3.6 Jalonnement temporel des produits

La notion d'admissibilité d'un routage explicitée au 2.9 sous-tend la possibilité de détermination, par propagation avant et arrière, de fenêtres temporelles de transfert admissibles pour chaque produit. L'objectif est de déterminer les plages de départ TW_p^{CH}, TW_p^{HH} et TW_p^{HC} autorisées pour le produit p sur les étages C-H (collecte), H-H (inter-hubs) et H-C (dispersion). Bien entendu, le calcul du jalonnement diffère selon que le produit considéré soit de routage primaire 1-hub ou 2-hubs. Pour rappel, le temps de latence ϵ_T est considéré égal à une période.

3.6.1 Jalonnement des produits 2-hubs

Considérons un produit $p \in P$ de routage primaire 2-hubs $s(p) \rightarrow h_o(p) \rightarrow h_d(p) \rightarrow d(p)$. Sa date de départ au plus tôt $est_o(p)$ de son CT origine $s(p)$ est par définition :

$$est_o(p) = es(p)$$

Nous en déduisons sa date de mise à disposition au plus tôt en entrée du hub origine $h_o(p)$:

$$ect_{h_o}(p) = est_o(p) + l(s(p), h_o(p))$$

Par induction, celle de mise à disposition au plus tôt en sortie du hub origine $h_o(p)$ est :

$$est_{h_o}(p) = ect_{h_o}(p) + \epsilon_T$$

L'arrivée au plus tôt sur le hub destination $h_d(p)$ est alors donnée par :

$$ect_{h_d}(p) = est_{h_o}(p) + l(h_o(p), h_d(p))$$

Cette dernière date permet d'obtenir la mise à disposition la plus anticipée en sortie de cross-docking sur $h_d(p)$:

$$est_{h_d}(p) = ect_{h_d}(p) + \epsilon_T$$

Enfin l'arrivée au plus tôt sur le CT destination est donnée par :

$$ect_d(p) = est_{h_d}(p) + l(h_d(p), d(p))$$

Symétriquement, un produit $p \in P$ arrive au plus tard sur son CT destination $d(p)$ à la date :

$$lct_d(p) = \theta_c^+ - \epsilon_T$$

Il part donc au plus tard de son hub destination $h_d(p)$ à :

$$lst_{h_d}(p) = lct_d(p) - l(h_d(p), d(p))$$

Par conséquent, il arrive au plus tard sur ce même hub à la date :

$$lct_{h_d}(p) = lst_{h_d}(p) - \epsilon_T$$

Par induction, la date de départ au plus tard du hub origine $h_o(p)$ est donnée par :

$$lst_{h_o}(p) = lct_{h_d}(p) - l(h_o(p), h_d(p))$$

Celle d'arrivée au plus tard sur ce même hub est alors :

$$lct_{h_o}(p) = lst_{h_o}(p) - \epsilon_T$$

Finalement, p part au plus tard de son CT origine $s(p)$ à la date :

$$lst_o(p) = lct_{h_o}(p) - l(s(p), h_o(p))$$

La figure 3.2 illustre le principe de jalonnement appliqué à un produit de type 2 (routage primaire 2-hubs).

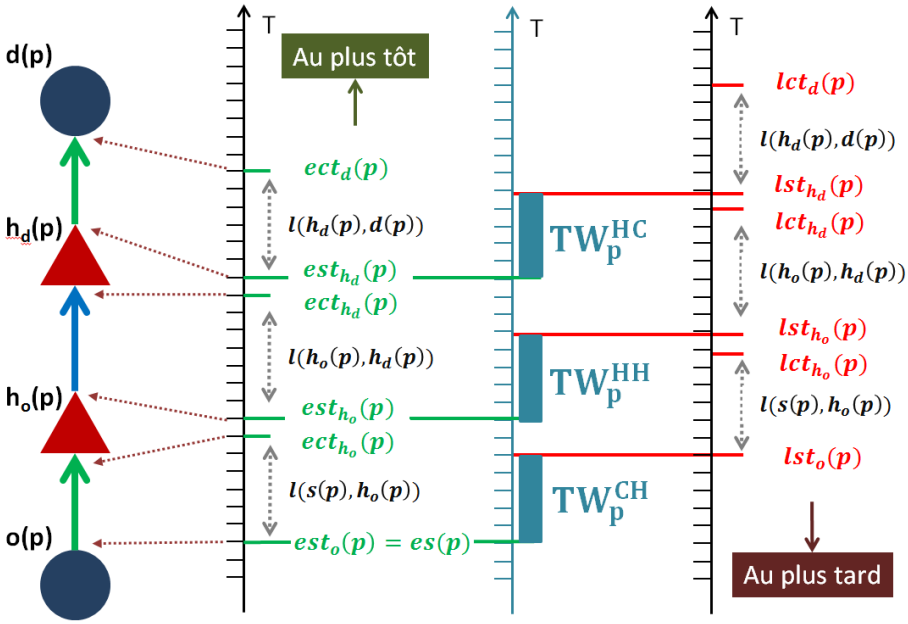


FIGURE 3.2 – Illustration du jalonnement d'un produit de type 2 (2-hubs)

3.6.2 Jalonnement des produits 1-hub

Considérons maintenant un produit $p \in P$ de routage primaire 1-hub $s(p) \rightarrow h(p) \rightarrow d(p)$. Ce produit arrive au plus tôt en entrée du hub $h(p)$ à la date :

$$ect_h(p) = es(p) + l(s(p), h(p))$$

Il est alors disponible au plus tôt en sortie de ce hub à :

$$est_h(p) = ect_h(p) + \epsilon_T$$

Par conséquent, il arrive au plus tôt sur son CT destination $d(p)$ à :

$$ect_d(p) = est_h(p) + l(h(p), d(p))$$

Symétriquement, la date d'arrivée au plus tard sur le CT destination est donnée par :

$$lct_d(p) = \theta_c^+ - \epsilon_T$$

Elle permet d'en déduire celle de départ au plus tard du hub $h(p)$:

$$lst_h(p) = lct_d(p) - l(h(p), d(p))$$

Ainsi, p arrive au plus tard sur ce même hub à la date :

$$lct_h(p) = lst_h(p) - \epsilon_T$$

Et enfin, il part au plus tard du CT d'origine $s(p)$ à :

$$lst_o(p) = lct_h(p) - l(s(p), h(p))$$

La figure 3.3 illustre le principe de jalonnement appliqué à un produit de type 1 (de routage primaire 1-hub).

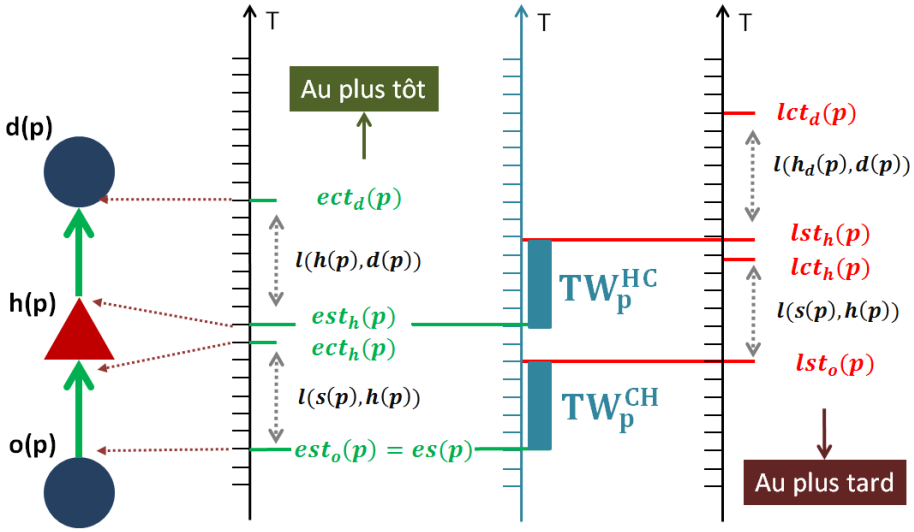


FIGURE 3.3 – Illustration du jalonnement d'un produit de type 1 (1-hub)

4

Une formalisation indexée dans le temps

Sommaire

4.1 Introduction

Les Graphes Indexés dans le Temps (GIT) permettent une modélisation assez simple et compacte de problématiques de nature dynamique dans lesquelles la composante temporelle intervient parallèlement à la gestion de flux mono ou multi-produits. Par contre, ils induisent des programmes mathématiques impliquant un nombre souvent rédhibitoire de variables de décision. Dans la problématique étudiée, chaque produit p peut être vu comme un flot élémentaire devant être routé d'une source $s(p)$ sur un puits $d(p)$ en quantité $q(p)$. Cette composante flot sous-jacente étant de nature "continue", une formalisation de type multiflots indexée dans le temps semble assez naturelle.

4.2 Graphe indexé dans le temps

Le GIT cible relatif à la problématique étudiée est noté $G = [X, U, l, u, w]$:
– X est un ensemble de sommets partitionnés en : $X = X_C \cup X_H \cup P$, où :

$$\begin{cases} X_C = \{(c,\theta)/c \in C, \theta \in]\tau]\} \cup \{(\bar{c},\theta)/c \in C, \theta \in]\tau]\} \\ X_H = \{(h,\theta)/h \in H, \theta \in]\tau]\} \cup \{(\bar{h},\theta)/h \in H, \theta \in]\tau]\} \end{cases}$$

$\bar{c}$ (resp. $\bar{h}$) désignant une copie du centre de traitement c (resp. du hub h).

– U est un ensemble d'arcs partitionnés en :
$U = U_{PC} \cup U_C^1 \cup U_C^2 \cup U_C^3 \cup U_H^1 \cup U_H^2 \cup U_H^3 \cup U_{CH} \cup U_{HH} \cup U_{HC} \cup U_{CP}$ où :

$$\begin{cases} U_{PC} = \{[p, (s(p), es(p))]/p \in P\} \\ U_C^1 = \{[(c,\theta),(c,\theta+1)]/c \in C, \theta \in]\tau-1] \cap \Theta_C\} \\ U_C^2 = \{[(c,\theta),(\bar{c},\theta+1)]/c \in C, \theta \in]\tau-1] \cap \Theta_C\} \\ U_C^3 = \{[(\bar{c},\theta),(\bar{c},\theta+1)]/c \in C, \theta \in]\tau-1] \cap \Theta_C\} \\ U_H^1 = \{[(h,\theta),(h,\theta+1)]/h \in H, \theta \in]\tau-1] \cap \Theta_H\} \\ U_H^2 = \{[(h,\theta),(\bar{h},\theta+1)]/h \in H, \theta \in]\tau-1] \cap \Theta_H\} \\ U_H^3 = \{[(\bar{h},\theta),(\bar{h},\theta+1)]/h \in H, \theta \in]\tau-1] \cap \Theta_H\} \\ U_{CH} = \{[(\bar{c},\theta),(h,\theta+l_{c,h})]/c \in C, h \in H, \theta \in]\tau-l_{c,h}] \cap \Theta_c\} \\ U_{HH} = \{[(\bar{h}_o,\theta),(h_d,\theta+l_{h_1,h_2})]/h_o, h_d \in H^2, \theta \in]\tau-l_{h_o,h_d} \cap \Theta_{h_o}\} \\ U_{CH} = \{[(\bar{h},\theta),(c,\theta+l_{h,c})]/c \in C, h \in H, \theta \in]\tau-l_{h,c}] \cap \Theta_h\} \\ U_{CP} = \{[(\bar{d}(p), lc(p)), p]/p \in P\} \end{cases}$$

– Les bornes inférieures de capacité de l'intégralité des arcs du réseau sont nulles, à l'exception des arcs de U_{PC} pour lesquels : $\forall p \in P, LB_{s(\bar{p}),es(p)} = q(p)$.
– Les bornes supérieures de capacité de l'intégralité des arcs du réseau sont infinies, à l'exception des arcs de :

$$\begin{cases} U_{PC} = \{[p, (\bar{s}(p), es(p))]/p \in P\} : UB_{s(\bar{p}),es(p)} = q(p) \\ U_C^2 = \{[(c,\theta),(\bar{c},\theta+1)]/c \in C, \theta \in]\tau-1] \cap \Theta_C\} : UB_{c,\theta} = \eta_c \\ U_H^2 = \{[(h,\theta),(\bar{h},\theta+1)]/h \in H, \theta \in]\tau-1] \cap \Theta_H\} : UB_{h,\theta} = \eta_h \\ u \in U_{CH} \cup U_{HC} : UB_u = Q_C \\ u \in U_{HH} : UB_u = Q_H \\ U_{CP} = \{[(\bar{d}(p), lc(p)), p]/p \in P\} : UB_{d(\bar{p}),lc(p)} = q(p) \end{cases}$$

– Les coûts associés aux arcs sont tous nuls à l'exception des arcs $u \in U_{CH} \cup U_{HH} \cup U_{HC}$ auxquels on associe un coût fixe de traversée w_u correspondant à l'exploitation d'un moyen de capacité Q_C sur les étages C-H et H-C, Q_H sur les inter-hubs, permettant d'acheminer les flux (produits) dans le réseau.
La figure 4.1 présente le graphe indexée dans le temps pour un produit p de type 1-hub.
Chaque produit $p \in P$ correspond à un flot élémentaire de valeur $q(p)$ devant être routé de $s(p)$ à $d(p)$ dans ce réseau. Pour ce faire, on associe à chaque arc u d'un sous-réseau G_p un flux (≥ 0). Le sous-réseau G_p filtre certains arcs du réseau G en interdisant tout flux associé à p de les traverser :
  – Les routages empruntés par le produit p traversent nécessairement l'arc $[p, (s(\bar{p}), es(p))]$
  – Ils ont ensuite la possibilité de suivre une séquence d'arcs de $U^3_{s(p)}$, correspondant à une attente de départ de flux associé à ce produit : $U^3_{s(p)} = \{[(\bar{s}(p),\theta),(\bar{s}(p),\theta+1)]/\theta \in]\tau-1] \cap \Theta_{s(p)}\}$
  – Un routage admissible emprunte alors des arcs de la restriction $U_{\bar{s}(p),h_o(p)} \cup U_{\bar{h}_o(p),h_d(p)} \cup U_{\bar{h}_d(p),d(p)}$ et en conformité avec la loi de conservation aux noeuds

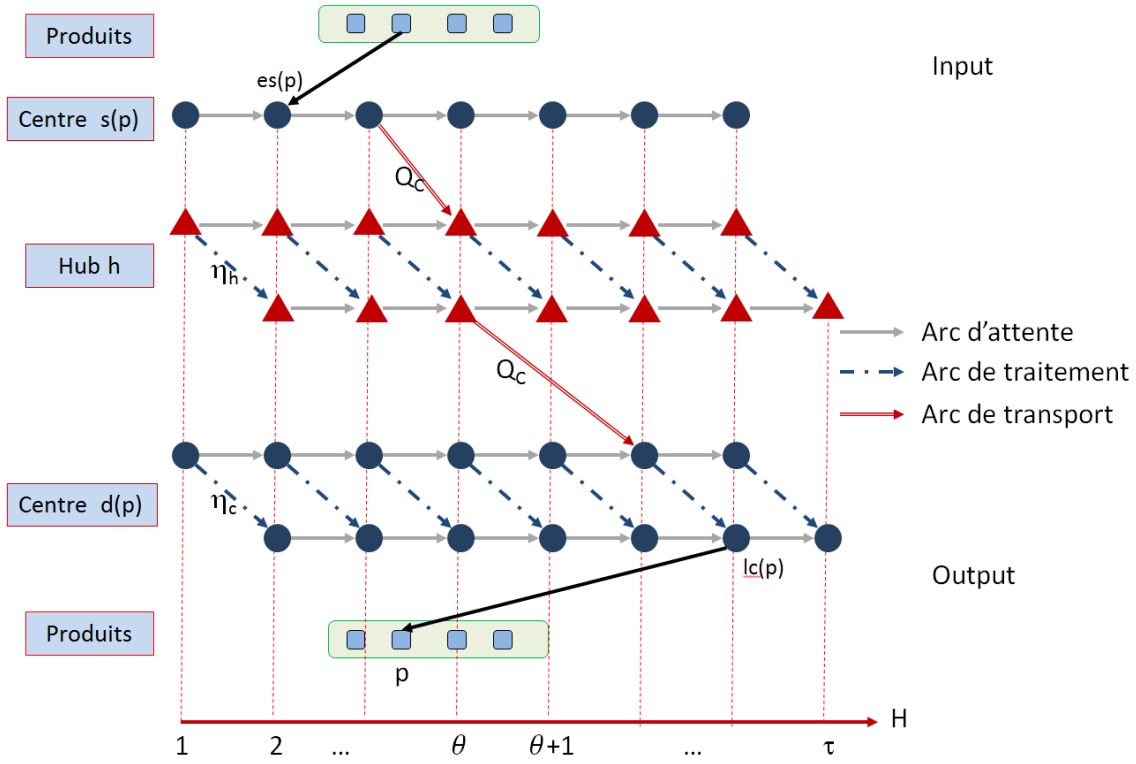


FIGURE 4.1 – Graphe indexé dans le temps

– Les routages empruntés par le produit p aboutissent enfin nécessairement en p via l'arc $[(\bar{d}(p), lc(p)), p]$.

Comme toute problématique de multiflot, les flots élémentaires sont non miscibles, mais la somme algébrique des flux associés traversant un arc quelconque du réseau ne peut excéder la borne supérieure de capacité de ce dernier. La valorisation économique du routage des flots élémentaires associés aux produits est prise en compte par le biais de variables de décision entières z_u en correspondance avec les arcs de $U_{CH} \cup U_{HH} \cup U_{HC}$. z_u quantifie le nombre de véhicules associés à l'arc u et monopolisés sur la liaison correspondant.

La problématique étudiée peut ainsi être vu comme un problème de dimensionnement de réseau sur routage d'un multiflot.

4.3 Une formalisation indexée dans le temps

4.3.1 Les variables de décision

Définissons les variables de décision :

– $x^A_{s,\theta}(p)$ = flux produit p sur la liaison $[(s,\theta),(s,\theta+1)]$ en attente de traitement (A) $\rightarrow s \in \{h_o(p), h_d(p), d(p)\}$

– $x^T_{s,\theta}(p)$ = flux produit p sur la liaison $[(s,\theta),(\bar{s},\theta+1)]$ en traitement $(T) \rightarrow s \in \{h_o(p), h_d(p), d(p)\}$

– $x^D_{s,\theta}(p)$ = flux produit p sur la liaison $[(\bar{s},\theta),(\bar{s},\theta+1)]$ en attente de départ (D) $\rightarrow s \in \{s(p), h_o(p), h_d(p)\}$

– $y^{\theta}_{s,s'}(p)$ = flux de produit p en transfert du site s vers le site s' entre les dates θ et $\theta + l_{s,s'}$.

4.3.2 Les lois de conservation aux noeuds

Considérons dans un premier temps un produit p de routage primaire $s(p) \to h_o(p) \to h_d(p) \to d(p)$. On a alors :

1. $c = s(p)$ et $h = h_o(p)$
   – $x^D_{c,es(p)-1}(p) = q(p)$
   – $\forall \theta \in]\tau] \cap \Theta_c, x^D_{c,\theta-1}(p) = x^D_{c,\theta}(p) + y^{\theta}_{c,h}(p)$
2. $h = h_o(p)$, $c = s(p)$ et $h' = h_d(p)$
   – $\forall \theta \in]\tau] \cap \Theta_h, x^A_{h,\theta-1}(p) + y^{\theta-l_{c,h}}_{c,h}(p) = x^A_{h,\theta}(p) + x^T_{h,\theta}(p)$
   – $\forall \theta \in]\tau] \cap \Theta_h, x^D_{h,\theta-1}(p) + x^T_{h,\theta-1}(p) = x^D_{h,\theta}(p) + y^{\theta}_{h,h'}(p)$
3. $h' = h_d(p)$, $h = h_o(p)$ et $c' = d(p)$
   – $\forall \theta \in]\tau] \cap \Theta_{h'}, x^A_{h',\theta-1}(p) + y^{\theta-l_{h,h'}}_{h,h'}(p) = x^A_{h',\theta}(p) + x^T_{h',\theta}(p)$
   – $\forall \theta \in]\tau] \cap \Theta_{h'}, x^D_{h',\theta-1}(p) + x^T_{h',\theta-1}(p) = x^D_{h',\theta}(p) + y^{\theta}_{h',c'}(p)$
4. $c' = d(p)$ et $h' = h_d(p)$
   – $\forall \theta \in]\tau] \cap \Theta_{c'}, x^A_{c',\theta-1}(p) + y^{\theta-l_{h',c'}}_{h',c'}(p) = x^A_{c',\theta}(p) + x^T_{c',\theta}(p)$
   – $\forall \theta \in]\tau] \cap \Theta_{c'}, x^D_{c',\theta-1}(p) + x^T_{c',\theta-1}(p) = x^D_{c',\theta}(p)$
   – $x^D_{c',lc(p)}(p) = q(p)$

Dans le cas d'un produit p de routage primaire $s(p) \to h(p) \to d(p)$, la simplification est immédiate :

5. $c = s(p)$ et $h = h(p)$
   – $x^D_{c,es(p)-1}(p) = q(p)$
   – $\forall \theta \in]\tau] \cap \Theta_c, x^D_{c,\theta-1}(p) = x^D_{c,\theta}(p) + y^{\theta}_{c,h}(p)$
6. $h = h(p)$, $c = s(p)$ et $c' = d(p)$
   – $\forall \theta \in]\tau] \cap \Theta_h, x^A_{h,\theta-1}(p) + y^{\theta-l_{c,h}}_{c,h}(p) = x^A_{h,\theta}(p) + x^T_{h,\theta}(p)$
   – $\forall \theta \in]\tau] \cap \Theta_h, x^D_{h,\theta-1}(p) + x^T_{h,\theta-1}(p) = x^D_{h,\theta}(p) + y^{\theta}_{h,c'}(p)$
7. $c' = d(p)$ et $h = h(p)$
   – $\forall \theta \in]\tau] \cap \Theta_{c'}, x^A_{c',\theta-1}(p) + y^{\theta-l_{h,c'}}_{h,c'}(p) = x^A_{c',\theta}(p) + x^T_{c',\theta}(p)$
   – $\forall \theta \in]\tau] \cap \Theta_{c'}, x^D_{c',\theta-1}(p) + x^T_{c',\theta-1}(p) = x^D_{c',\theta}(p)$
   – $x^D_{c',lc(p)}(p) = q(p)$

Les différents cas relatifs à un site s sont synthétisés dans le schéma 4.2.

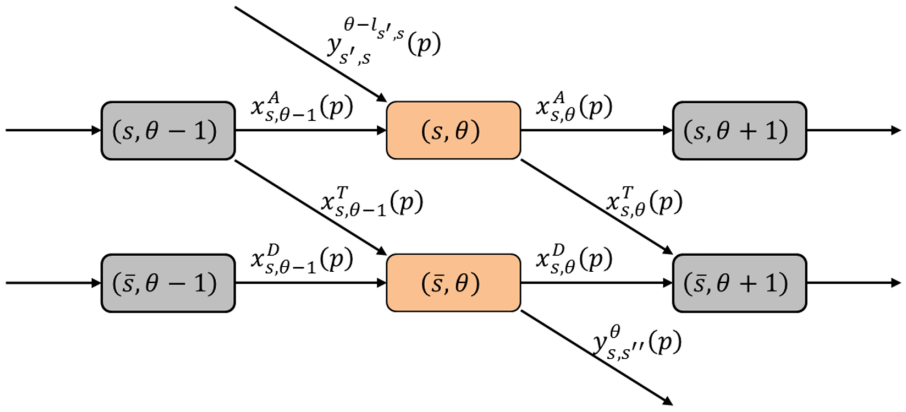


FIGURE 4.2 – Zoom sur un site s

4.3.3 Bornes supérieures de capacité

Définissons les variables de décision additionnelles suivantes :

- $z^{\theta}_{s,s'}$ = nombre de véhicules de capacité Q ($= Q_C$ ou Q_H) au départ du site s à θ et à destination du site s' (la date d'arrivée en s' étant alors : $\theta + l_{s,s'}$). Par construction, on a nécessairement $(s, s') \in \{(c, h), (h, h'), (h, c)\}$. Notons de plus :
  - $P_c = \{p \in P / d(p) = c\}$
  - $P_h = \{p \in P / h_o(p) = h \vee h_d(p) = h\}$
  - $P_{c,h} = \{p \in P / s(p) = c \wedge h_o(p) = h\}$
  - $P_{h,h'} = \{p \in P / h_o(p) \wedge h_d(p) = h'\}$
  - $P_{h,c} = \{p \in P / h_d(p) = h \wedge d(p) = c\}$

Les contraintes couplantes portant sur le respect des bornes supérieures de capacité sont alors :

- $u \in U^2_C = \{[(c, \theta), (\bar{c}, \theta + 1)] / c \in C, \theta \in]\tau - 1] \cap \Theta_C\}$:
  $\sum_{p \in P_c} x^T_{c,\theta}(p) \leq \eta_c$
- $u \in U^2_H = \{[(h, \theta), (\bar{h}, \theta + 1)] / h \in H, \theta \in]\tau - 1] \cap \Theta_H\}$:
  $\sum_{p \in P_h} x^T_{h,\theta}(p) \leq \eta_h$
- $u \in U_{CH} = \{[(\bar{c}, \theta), (h, \theta + l_{c,h})] / c \in C, h \in H, \theta \in]\tau - l_{c,h}] \cap \Theta_c\}$:
  $\sum_{p \in P_{c,h}} y^{\theta}_{c,h}(p) \leq Q_C z^{\theta}_{c,h}$
- $u \in U_{HH} = \{[(\bar{h_1}, \theta), (h_2, \theta + l_{h_1,h_2})] / h_1, h_2 \in H^2, \theta \in]\tau - l_{h_1,h_2} \cap \Theta_{h_1}\}$:
  $\sum_{p \in P_{h,h'}} y^{\theta}_{h,h'}(p) \leq Q_H z^{\theta}_{h,h'}$
- $u \in U_{CH} = \{[(\bar{h}, \theta), (c, \theta + l_{h,c})] / c \in C, h \in H, \theta \in]\tau - l_{h,c}] \cap \Theta_h\}$:
  $\sum_{p \in P_{h,c}} y^{\theta}_{h,c}(p) \leq Q_C z^{\theta}_{h,c}$

4.3.4 Modèle

Pour récapituler, le modèle est le suivant :

$$[P_{MIT}] \qquad \min \sum_{s\in S}\sum_{s'\in S}\sum_{\theta\in\Theta} w_{s,s'} z^{\theta}_{s,s'}$$

SC

$$\forall (s,s') \in A, \forall \theta \in \Theta \qquad \sum_{p\in P_{s,s'}} y^{\theta}_{s,s'}(p) \leq cp_{s,s'} z^{\theta}_{s,s'} \quad (4.1)$$

$$\forall s \in S, \forall \theta \in \Theta \qquad \sum_{p\in P_s} x^{T}_{s,\theta}(p) \leq \eta_s \quad (4.2)$$

$$\forall p \in P_{t2}, \forall \theta \in \Theta \qquad x^{A}_{h_o(p),\theta-1}(p) + y^{\theta-l_{s(p),h_o(p)}}_{s(p),h_o(p)}(p) = x^{A}_{h_o(p),t}(p) + x^{T}_{h_o(p),\theta}(p) \quad (4.3)$$

$$\forall p \in P_{t2}, \forall \theta \in \Theta \qquad x^{A}_{h_d(p),\theta-1}(p) + y^{\theta-l_{h_o(p),h_d(p)}}_{h_o(p),h_d(p)}(p) = x^{A}_{h_d(p),\theta}(p) + x^{T}_{h_d(p),\theta}(p) \quad (4.4)$$

$$\forall p \in P_{t1}, \forall \theta \in \Theta \qquad x^{A}_{h_d(p),\theta-1}(p) + y^{\theta-l_{s(p),h_d(p)}}_{s(p),h_d(p)}(p) = x^{A}_{h_d(p),\theta}(p) + x^{T}_{h_d(p),\theta}(p) \quad (4.5)$$

$$\forall p \in P, \forall \theta \in \Theta \qquad x^{A}_{d(p),\theta-1}(p) + y^{\theta-l_{h_d(p),d(p)}}_{h_d(p),d(p)}(p) = x^{A}_{d(p),\theta}(p) + x^{T}_{d(p),\theta}(p) \quad (4.6)$$

$$\forall p \in P \qquad x^{D}_{s(p),es(p)-1}(p) = q(p) \quad (4.7)$$

$$\forall p \in P, \forall \theta \in \Theta \qquad x^{D}_{s(p),\theta-1}(p) = x^{D}_{s(p),\theta}(p) + y^{\theta}_{s(p),h_o(p)}(p) \quad (4.8)$$

$$\forall p \in P_{t2}, \forall \theta \in \Theta \qquad x^{D}_{h_o(p),\theta-1}(p) + x^{T}_{h_o(p),\theta-1}(p) = x^{D}_{h_o(p),\theta}(p) + y^{\theta}_{h_o(p),h_d(p)}(p) \quad (4.9)$$

$$\forall p \in P_{t2}, \forall \theta \in \Theta \qquad x^{D}_{h_d(p),\theta-1}(p) + x^{T}_{h_d(p),\theta-1}(p) = x^{D}_{h_d(p),\theta}(p) + y^{\theta}_{h_d(p),d(p)}(p) \quad (4.10)$$

$$\forall p \in P_{t1}, \forall \theta \in \Theta \qquad x^{D}_{h_d(p),\theta-1}(p) + x^{T}_{h_d(p),\theta-1}(p) = x^{D}_{h_d(p),t}(p) + y^{\theta}_{h_d(p),d(p)}(p) \quad (4.11)$$

$$\forall p \in P, \forall \theta \in \Theta \qquad x^{D}_{d(p),\theta-1}(p) + x^{T}_{d(p),\theta-1}(p) = x^{D}_{d(p),\theta}(p) \quad (4.12)$$

$$\forall p \in P \qquad x^{D}_{d(p),lc(p)} = q(p) \quad (4.13)$$

$$\forall (s,s') \in A, \forall \theta \in \Theta \qquad z^{\theta}_{s,s'} \in \mathbb{N} \quad (4.14)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{A}_{s,\theta}(p) \geq 0 \quad (4.15)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{D}_{s,\theta}(p) \geq 0 \quad (4.16)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{T}_{s,\theta}(p) \geq 0 \quad (4.17)$$

$$\forall (s,s') \in A, \forall p \in P_{s,s'}, \forall \theta \in \Theta \qquad y^{\theta}_{s,s'}(p) \geq 0 \quad (4.18)$$

5

Inégalités valides et Surrogates

Sommaire

5.1 Introduction

Nos tests préliminaires ont montré que pour certaines instances de grande taille, le temps de résolution inhérent notamment à l'exploitation de la première formalisation par solveur peut devenir rédhibitoire pour des tailles d'instances pourtant modérée. Fort de ce constat, nous introduisons dans cette section des coupes valides ajoutées à ces modèles et permettant de rétablir les temps de résolution à des valeurs acceptables. Dans ce prolongement, nous mettons également en exergue l'exploitation de contraintes redondantes (simple désagrégation des contraintes 3.2 ou 4.1) ou encore une propriété de dominance sur les routages permettant de sur-contraindre les variables de flux dans la première formalisation.

5.2 Coupes valides

Soit une liaison $s - s'$ portant indifféremment sur C-H, H-C ou H-H. Rappelons que $P_{s,s'}$ désigne l'ensemble des produits transitant sur ce segment de transport. Sur la base du jalonnement des produits exposé en section 3.6, tout produit $p \in P_{s,s'}$ décline une date de départ au plus tôt $est_s(p)$ (resp. au plus tard $lst_s(p)$) sur s. De manière générale, les fenêtres temporelles d'admissibilité des produits portant sur la liaison s-s' peuvent se chevaucher, tel qu'illustré par la figure 5.1.

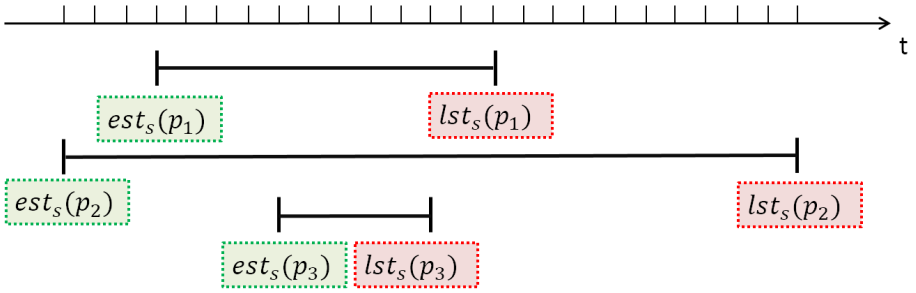


FIGURE 5.1 – Exemple de jalonnement de produits sur un segment de transport

Notons $Est_{s,s'} = \{est_s(p), p \in P_{s,s'}\}$, l'ensemble des dates de départ au plus tôt sur s des produits transitant sur la liaison $s - s'$ et corrélativement, définissons $Lst_{s,s'} = \{lst_s(p), p \in P_{s,s'}\}$ l'ensemble des dates de départ au plus tard sur s. Soit :

$$\Theta_{s,s'} = \{[\theta^-, \theta^+] / \theta^- \in Est_{s,s'} \wedge \theta^+ \in Lst_{s,s'} \wedge \theta^- \leq \theta^+ \wedge (\exists p \in P_{s,s'} / \quad (5.1)$$

$$est_s(p) \geq \theta^- \wedge lst_s(p) \leq \theta^+\} \quad (5.2)$$

$\Theta_{s,s'}$ filtre l'ensemble des intervalles de temps constitués sur la base des combinaisons de dates de départ au plus tôt et au plus tard des produits routés sur la liaison $s - s'$ et dans lesquels au moins un produit doit nécessairement arriver et repartir. La figure 5.2 illustre l'exemple de la figure 5.1.

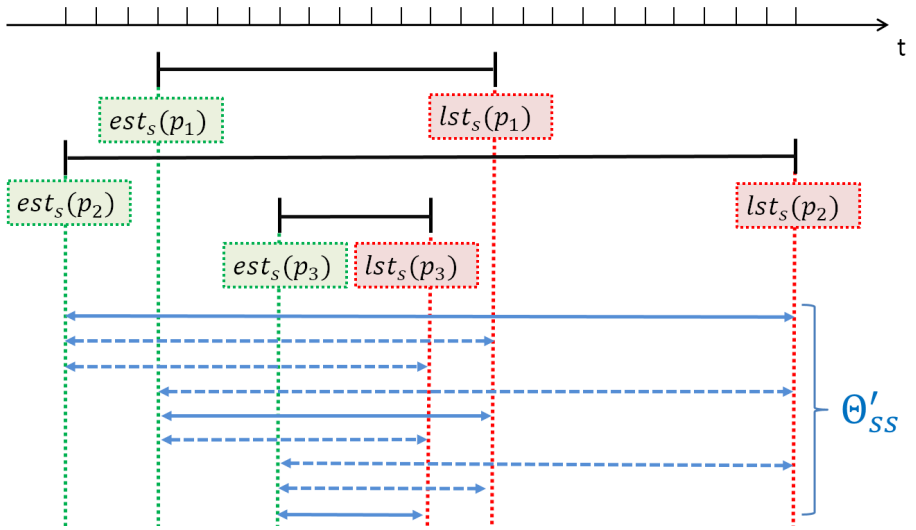


FIGURE 5.2 – Jalonnement des produits sur une liaison $s - s'$: Génération de $\Theta_{s,s'}$

Pour tout ensemble de produits dont les périodes d'admissibilité sont comprises dans un ensemble d'intervalles de $\Theta_{s,s'}$, supposons qu'un traitement ad-hoc ne conserve dans $\Theta_{s,s'}$ que l'intervalle d'amplitude minimale. Dans l'exemple de la figure 5.2, les intervalles ainsi supprimés sont figurés par des lignes pointillées. A l'issue de ce filtrage, chaque intervalle $[\theta^-, \theta^+] \in \Theta_{s,s'}$ couvre un sous-ensemble unique des produits de $P_{s,s'}$. Notons $P_{s,s'}^{\theta^-,\theta^+} = \{p \in P_{s,s'} / est_s(p) \geq \theta^- \wedge lst_s(p) \leq \theta^+\}$ ce sous ensemble.

Proposition 3 *Pour toute liaison $s-s'$, les contraintes suivantes constituent un ensemble de coupes valides :*

$$\forall s \in S, \forall s' \in S, \forall [\theta^-, \theta^+], \left\lceil \frac{\sum_{p \in P_{s,s'}}^{\theta^-,\theta^+} q(p)}{Q_{s,s'}} \right\rceil \leq \sum_{\theta=\theta^-}^{\theta^+} z_{s,s'}^{\theta} \tag{5.3}$$

où $Q_{s,s'}$ correspond à la capacité du moyen logistique sous-jacent à la liaison $s-s'$.

Preuve 3 *La somme $\sum_{p \in P_{s,s'}}^{\theta^-\theta^+} q(p)$ donne la quantité totale de produit qui doit nécessairement partir dans l'intervalle $[\theta^-, \theta^+]$. Diviser cette valeur par la capacité totale du moyen logistique sous-jacent à la liaison donne dès lors le nombre minimal de véhicules requis pour router cette quantité. Trivialement, cette fraction non entière doit être arrondie à l'entier supérieur. Cet entier est ensuite posé en borne inférieure du nombre de départs déclenchés entre θ^- et θ^+ dans la solution ($\sum_{\theta=\theta^-}^{\theta^+} z_{s,s'}^{\theta}$).* ■

Les mêmes coupes peuvent s'écrire pour la modélisation multiflots :

$$\forall [\theta^-, \theta^+] \in \Theta_{ss'}, \left\lceil \frac{\sum_{p \in P_{\theta^-\theta^+}} q(p)}{cp(ss')} \right\rceil \leq \sum_{\omega \in \Omega_{ss'}^{\theta^-\theta^+}} y_{\omega} \tag{5.4}$$

où $cp(ss')$ correspond à la capacité du moyen logistique sous-jacent à la liaison s-s'.

5.3 Surrogates

De plus, afin d'améliorer la qualité de la relaxation linéaire et donc le temps de résolution des modèles, nous ajoutons les surrogates suivantes :

$$\forall s \in S, \forall s' \in S, (s,s') \in U, \forall \theta \in \Theta_{s,s'}, \forall p \in P_{s,s'}^{\theta}, y_{s,s'}^{\theta}(p) \leq q(p) z_{s,s'}^{\theta} \tag{5.5}$$

Avec $P_{s,s'}^{\theta}$: ensemble des produits pouvant transiter sur (s,s') à l'instant θ.

Ces surrogates ont pour but d'interdire au flux de transiter à un instant donné si aucun véhicule n'est actif pour cet instant et inversement, d'activer un véhicule pour un instant donné si un flux transite lors de cet instant.

5.4 Dominances sur les routages

Considérons, pour un produit p de type 2, les deux routages admissibles $\sigma = [\omega_1, \omega_2^1, \omega_3]$ et $\sigma' = [\omega_1, \omega_2^2, \omega_3]$ avec :

$$st(\omega_2^2) > st(\omega_2^1) \text{ et } x_p^{\omega_1} \geq x_p^{\omega_2^1} + x_p^{\omega_2^2}$$

Alors il existe une solution optimale dans laquelle on a :

$$x_p^{\omega_2^2} > 0 \Rightarrow \sum_{p \in P} x_p^{\omega_2^1} = k.cp(\omega_2^1) \text{ avec } k \in \mathbb{N}$$

En effet, si le produit p emprunte l'OT ω_2^2 via le routage $\sigma' = [\omega_1, \omega_2^2, \omega_3]$ ($x_p^{\omega_2^2} > 0$) alors que $\sum_{p \in P} x_p^{\omega_2^1} = k.cp(\omega_2^1) - s$ avec $s > 0$, alors on peut transférer un flux $\phi = min(x_p^{\omega_2^2}, s)$ sur l'OT ω_2^1 via le routage $\sigma = [\omega_1, \omega_2^1, \omega_3]$ sans affecter la réalisabilité.

Bornes et Méthodes de résolution

6

Calcul de bornes inférieures : Approche par segmentation

Sommaire

6.1 Introduction

Nous proposons dans cette section un calcul de bornes inférieures couplant une décomposition spatiale du problème selon les étages C-H (collecte), H-C (dispersion) et H-H (inter-hubs) à une agrégation des produits au travers de la notion de flux cumulés.

6.2 Routage des flux sur l'étage collecte (C-H)

L'étage C-H se décompose naturellement suivant les liaisons c-h (segments de transport) pour lesquelles la liste de produits $P_{c,h}$ est non vide. Cette décomposition est exploitée ici pour évaluer l'étage C-H. Plus précisément, il s'agit de déterminer une borne inférieure de la composante du coût du modèle global induite par l'activation des OT de l'étage collecte.

6.2.1 Une modélisation MIP locale de l'étage collecte

Notons $W_h^p = [ect_{h_o}(p), lct_{h_o}(p)]$ la fenêtre temporelle au sein de laquelle le produit p doit arriver sur son hub origine associé $h_o(p)$, en respect du jalonnement exposé en section 3.6. L'intégration de celui-ci a pour objectif d'éviter d'emblée un problème de non réalisabilité liée à l'impossibilité de router un produit en raison d'un départ anormalement différé de son CT d'origine pour des raisons de massification. Soit $z_{c,h}^{\theta}$ la variable de décision entière correspondant au nombre de véhicules arrivant sur le hub h dans la période θ en provenance de c (et donc, partant à $\theta - l(c,h)$ du CT origine c). Par ailleurs, définissons les variables de flux x_p^{θ} correspondant à la quantité de produit p arrivant sur le hub origine $h = h_o(p)$ en fin de période θ en provenance de son CT origine $s(p)$. Le MIP $[FluxCH(c,h)]$ associé au sous-problème sous-jacent à chaque segment de transport c-h de l'étage collecte est ainsi :

$$[FluxCH(c,h)] \qquad \min \sum_{\theta \in \Theta} z_{c,h}^{\theta}$$

$$SC$$

$$\forall p \in P_{c,h} \qquad \sum_{\theta \in W_h^p} x_p^{\theta} = q(p) \tag{6.1}$$

$$\forall \theta \in \Theta \qquad \sum_{p \in P_{c,h}} x_p^{\theta} \leq Q_C z_{c,h}^{\theta} \tag{6.2}$$

$$\forall \theta \in \Theta \qquad z_{c,h}^{\theta} \in \mathbb{N} \tag{6.3}$$

$$\forall \theta \in \Theta, \forall p \in P_{c,h} \qquad x_p^{\theta} \geq 0 \tag{6.4}$$

Proposition 4 *$\sum_{c \in C} \sum_{h \in H} Opt[FluxCH(c,h)]$ est une borne inférieure du nombre optimal de moyens nécessaires sur les liaisons C-H dans toute solution de $[P]$.*

Preuve 4 *Aucune capacité de traitement n'étant intégrée à ce stade, il n'y a aucun couplage entre les flux transitant par des liaisons C-H différentes. Par conséquent, le routage optimal des flux sur l'étage collecte peut s'opérer à un niveau local au segment de transport.* ■

Les OT portant sur un même segment de transport ont un coût identique. Par conséquent, pour toute liaison $c-h$, la borne inférieure du nombre de départs $Opt[FluxCH(c, h)]$ nous permet d'obtenir la borne inférieure du coût induit par ces départs, en multipliant cette valeur par la distance $l(c,h)$ ainsi que par le coût du type de véhicules associé au liaisons C-H.

6.2.2 Un algorithme polynomial de résolution de [FluxCH(c,h)]

Nous introduisons dans cette section un algorithme dont nous supposons qu'il produit une solution optimale de $[FluxCH(c,h)]$ en temps polynomial. Cette version n'intègre pas explicitement l'instanciation des variables de flux. Cet algorithme route progressivement les produits dans une stratégie de massification maximale. Notons $L_{c,h}$ la liste des produits restant à router à une étape de ce processus et B celles des produits candidats

pour un départ sur une même période. La procédure **GetPrem**(B) renvoie simplement le premier élément de la liste B. Introduisons maintenant la procédure **GetMinProdInterCH**($L_{c,h}, B$). Parmi les produits restant à router (de $L_{c,h}$), cette procédure détermine tout d'abord ceux dont le jalonnement autorise un départ sur une période admissible pour l'ensemble des produits de B. Parmi les produits ainsi filtrés, la procédure retourne celui dont le nombre de produits de $L_{c,h}$ avec lequel il peut partir sur une même période est minimal. Si aucun produit ne respecte ces conditions, la procédure retourne "$Null$". Définissons en outre la procédure **DeclencherDeparts**(B) (algorithme 1), permettant de déclencher un ou plusieurs départs de véhicules sur une même période pour le groupe de produit B.

Algorithme 1 Procédure **DeclencherDeparts**(B)

Entrées:

- B : ensemble de produits pour lequel déclencher des départs

$v \leftarrow \lfloor C_B / Q_C \rfloor$
si $v = 0$ **alors**
  $\bar{C} \leftarrow C_B$
  $\gamma^+_{c,h} \leftarrow \gamma^+_{c,h} + 1$
sinon
  $\bar{C} \leftarrow v \times Q_C$
  $\gamma^+_{c,h} \leftarrow \gamma^+_{c,h} + v$
fin si
tant que $\bar{C} > 0$ **faire**
  $C^R \leftarrow \min(\bar{C}, Q_C)$
  $\bar{C} \leftarrow \bar{C} - C^R$
  $\tilde{C} \leftarrow C^R$; $\theta \leftarrow -1$
  tant que $C^R > 0$ **faire**
    $p^* \leftarrow$**GetPrem**(B)
    $\theta \leftarrow \max(\theta, es(p^*))$
    si $\bar{q}_{p^*} > C^R$ **alors**
      $\bar{q}_{p^*} \leftarrow \bar{q}_{p^*} - C^R$; $C^R \leftarrow 0$
      $z^\theta_{c,h} \leftarrow z^\theta_{c,h} + \frac{Q_C - \tilde{C}}{Q_C}$
    sinon
      $C^R \leftarrow C^R - \bar{q}_{p^*}$; $\bar{q}_{p^*} \leftarrow 0$
      $B \leftarrow B \ominus \{p^*\}$
    fin si
  fin tant que
fin tant que

Bien que nous ne soyons pas en mesure d'apporter une preuve formelle de la validité de notre algorithme, nos expérimentations montrent qu'il donne une solution de même valorisation que le modèle $[FluxCH(c,h)]$ pour l'ensemble de nos instances.

La solution obtenue par résolution directe du modèle $[FluxCH(c,h)]$ ou par l'algorithme 2 est localement optimale car elle favorise la massification des produits sur chaque segment de transport c-h.

Algorithme 2 Calcul polynomial d'une solution optimale de $[FluxCH(c,h)]$

Entrées:

- segment de transport $c-h$

Sorties:

- borne inférieure du nombre de départs $\gamma^+_{c,h}$ sur la liaison $c-h$ considérée
- $\forall \theta \in \Theta, z^\theta_{c,h}$: nombre de départs déclenchés sur la période θ

Notations:

- $\forall p \in P_{c,h}, \bar{q}(p)$: quantité de produit p restant à acheminer
- C_B : quantité à router pour les produits de B

$L_{c,h} \leftarrow P_{c,h}$
$\gamma^+_{c,h} \leftarrow 0$
$\forall \theta \in \Theta, z^\theta_{c,h} \leftarrow 0$
$\forall p \in L_{c,h}, \bar{q}(p) \leftarrow q(p)$
$B \leftarrow \varnothing$
$C_B \leftarrow 0$
tant que $L_{c,h} \neq \varnothing$ **faire**
    $p^* \leftarrow$**GetMinProdInterCH**$(L_{c,h}, B)$
    si $p^* \neq Null$ **alors**
        $B \leftarrow B \oplus \{p^*\}$
        $C_B \leftarrow C_B + \bar{q}(p)$
        si $C_B > Q_C$ **alors**
            DeclencherDeparts(B)
            $L_{c,h} \leftarrow L_{c,h} \ominus \{p^*\}$
            $L_{c,h} \leftarrow L_{c,h} \cup B$
            $B \leftarrow \varnothing$
        sinon
            si $C_B = Q_C$ **alors**
                DeclencherDeparts(B)
            fin si
            $L_{c,h} \leftarrow L_{c,h} \ominus \{p^*\}$
        fin si
    sinon
        DeclencherDeparts(B)
    fin si
fin tant que
si $B \neq \varnothing$ **alors**
    DeclencherDeparts(B)
fin si

Dans un contexte général, cette stratégie peut sur-contraindre les niveaux supérieurs (H-H et H-C) et pénaliser la massification sur ces étages, voire même la réalisabilité. En effet, le départ d'un produit sur C-H peut être fortement retardé pour des raisons de massification et induire une configuration ou le traitement sur les hubs de transit ou sur le CT destination de ce produit n'est plus réalisable car la plage d'admissibilité révisée pour le produit sur H-H ou H-C devient trop restreinte.

6.3 Routage des flux sur l'étage dispersion (H-C)

Contrairement à l'étage collecte (C-H), l'étage dispersion ne se décompose plus naturellement suivant chaque segment de transport à cause des contraintes de traitement sur les CT destination, induisant un couplage des flux en provenance des différents hubs.

6.3.1 Une modélisation MIP locale de l'étage dispersion

Considérons un CT destination $c \in C$ et rappelons que P_c^- désigne l'ensemble des produits à destination de ce CT. Pour tout $p \in P_c^-$, $W_c^p = [ect_d(p), lct_d(p)]$ correspond au domaine temporel sur lequel p doit nécessairement arriver sur son CT destination $d(p) = c$. Notons $z_{h,c}^{\theta}$ la variable de décision correspondant au nombre de véhicules quittant le hub h à la période $\theta - l(h,c)$ pour arriver sur le CT destination c lors de la période θ. Soit x_p^{θ} la variable de décision correspondant à la quantité de produit p arrivant sur le CT destination $c = d(p)$ en provenance du hub destination $h_d(p)$ à la période θ. Enfin, rappelons que $\Delta_c = \sum_{p \in P_c^-} q(p)$ désigne la quantité cumulée de produits à destination de c.

Le programme linéaire mixte $[FluxHC(c)]$ détermine le nombre minimal de véhicules requis pour router l'intégralité des produits à destination du CT destination c sur l'étage H-C :

$$[FluxHC(c)] \qquad \min \sum_{\theta \in \Theta} \sum_{h \in H_c^-} z_{h,c}^{\theta}$$

$$SC$$

$$\forall p \in P_c^- \qquad \sum_{\theta \in W_c^p} x_p^{\theta} = q(p) \tag{6.5}$$

$$\forall \theta \in \Theta_c, \forall h \in H_c^- \qquad \sum_{p \in P_{h,c}} x_p^{\theta} \leq Q_C z_{h,c}^{\theta} \tag{6.6}$$

$$\forall \theta \in \Theta_c \qquad \sum_{p \in P_c^-} \sum_{\theta' \leq \theta} x_p^{\theta'} \geq \Delta_c - (\theta_c^+ - \theta + 1)\eta_c \tag{6.7}$$

$$\forall \theta \in \Theta_c, \forall h \in H_c^- \qquad z_{h,c}^{\theta} \in \mathbb{N} \tag{6.8}$$

$$\forall \theta \in \Theta_c, \forall p \in P_c^- \qquad x_p^{\theta} \geq 0 \tag{6.9}$$

Proposition 5 *$\sum_{c \in C} Opt[FluxHC(c)]$ est une borne inférieure du nombre de véhicules requis sur les liaisons H-C dans toute solution optimale de $[P]$.*

Preuve 5 *Les capacités de traitement sur les hubs étant occultées lors de ce calcul de borne inférieure sur H-C, localement, les produits à router sur cet étage ne sont donc couplés que par les contraintes de traitements sur les CT destination. Par conséquent, le routage optimal des flux sur l'étage dispersion peut s'opérer à un niveau local au CT destination.* ■

Au même titre que pour le calcul de borne inférieure sur C-H, nous obtenons naturellement le coût induit par ces départs, sur la base des distances $l(h,c)$ ($h \in H_c^-$) et du coût du type de véhicules associé aux liaisons de l'étage dispersion.

6.3.2 Une stratégie heuristique de résolution de [FluxHC(c)]

Dans cette section, nous introduisons une heuristique supposée produire une solution optimale du problème $[FluxHC(c)]$. La stratégie mise-en-œuvre réside dans la prise en compte des dates d'arrivée au plus tôt des produits sur c permettant d'éviter les cas directs d'absence de réalisabilité, parallèlement à une intégration du contrôle de capacité traitement sur c.

Pour ce CT $c \in C$ et son sous-ensemble de produits associé P_c^-, l'idée est alors de caler ces derniers au plus tard, en priorisant les produits p de dates d'arrivée au plus tôt $ect_d(p)$ les plus tardives, en respect de la capacité de traitement η_c associée à c.

Itérativement, les produits de P_c^- restant à traiter, et stockés dans une liste L_c, sont routés dans la solution courante dans l'ordre des dates d'arrivée $ect_c(p)$ décroissantes. La procédure **FluxHC**(c) (algorithme 4) donne la mise en oeuvre de la stratégie d'affectation des véhicules sur H-C, en respect des capacités de traitement. $\gamma_{h,c}^-$ désigne le nombre de moyens affrétés au départ du hub h et à destination de c. La routine **MajTemps**($\chi, \delta q, \eta_c$) (algorithme 3) est appelée en vue de mettre à jour la période de temps χ courante sur l'horizon, en fonction d'une charge traitée δq, et à concurrence de la capacité de traitement η_c du CT destination considéré.

Algorithme 3 Procédure **MajTemps**($\chi, U, \delta q, \eta_c$)

Entrées:

- χ : période courante, à mettre à jour
- U : capacité de traitement résiduelle sur la période courante
- δq : quantité sur la base de laquelle mettre à jour la période courante, à concurrence de la capacité de traitement
- η_c : capacité de traitement considérée

1: **tant que** $\delta q > 0$ **faire**
2: **si** $U = 0$ **alors**
3: $\chi \leftarrow \chi - 1$
4: $U \leftarrow \eta_c$
5: **fin si**
6: **si** $\delta q < U$ **alors**
7: $U \leftarrow U - \delta q$
8: $\delta q \leftarrow 0$
9: **sinon**
10: $\delta q \leftarrow \delta q - U$
11: $U \leftarrow 0$
12: **fin si**
13: **fin tant que**
14: **renvoyer** χ

Malgré des ajustements, nos tests montrent que la stratégie heuristique proposée pour résoudre le problème $[FluxHC(c)]$ ne donne pas toujours la même solution que celle obtenue par résolution directe du modèle $[FluxHC(c)]$ par solveur. Plus précisément, l'algorithme proposé produit parfois des solutions dont le nombre de départs est inférieur à celui de l'optimum du modèle $[FluxHC(c)]$.

A l'instar de l'étage collecte, la stratégie de massification maximale opérée sur l'étage de dispersion peut avoir un impact négatif sur les autres étages (C-H et H-H), à cause de produits de date de départ trop anticipée sur H-C pour des raisons de massification.

Algorithme 4 Procédure **FluxHC**(c)

Entrées:

- c : CT destination pour lequel calculer la borne inférieure du nombre de départs

Sorties:

- $\forall h \in H_c^-, \gamma_{h,c}^-$: nombre de départs sur chaque liaison h-c
- $\forall h \in H_c^-, \forall \theta \in \Theta, z_{h,c}^{\theta}$: nombre de véhicules arrivant sur c à la période θ en provenance de h

Notations:

- L_c : produits restant à router
- $\forall h \in H_c^-, C_h^R$: capacité résiduelle du véhicule en court de remplissage sur la liaison h-c
- χ : période courante considérée sur l'horizon temporel
- U : capacité de traitement résiduelle du CT c sur la période courante

$L_c \leftarrow P_c^-$
$\forall h \in H_c^-, \gamma_{h,c} \leftarrow 1$
$\forall h \in H_c^-, \forall \theta \in \Theta_c, z_{h,c}^{\theta} \leftarrow 0$
$\forall h \in H_c^-, C_h^R \leftarrow Q_C$
$U \leftarrow \eta_c$
$\chi \leftarrow \theta_c^+$
tant que $L_c \neq \varnothing$ **faire**
    $p \leftarrow \text{argmax}_{p \in L_c} ect_c(p)$
    $qr \leftarrow q(p)$
    $\delta q \leftarrow 0$
    $h \leftarrow h_d(p)$
    tant que $qr > 0$ **faire**
        si $qr < C_h^R$ **alors**
            $C_h^R \leftarrow C_h^R - qr$
            $\delta q \leftarrow \delta q + qr$
            $qr \leftarrow 0$
            $\chi \leftarrow$**MajTemps**$(\chi, U, \delta q, \eta_c)$
        sinon
            $qr \leftarrow qr - C_h^R$
            $\delta q \leftarrow \delta q + C_h^R$
            $\chi \leftarrow$**MajTemps**$(\chi, U, \delta q, \eta_c)$
            $\theta \leftarrow \chi$
            $z_{h,c}^{\theta} \leftarrow z_{h,c}^{\theta} + \frac{Q_C - C_h^R}{Q_C}$
            $\gamma_{h,c}^- \leftarrow \gamma_{h,c}^- + 1$
            $C_h^R \leftarrow Q_C$
        fin si
    fin tant que
fin tant que

6.4 Routage des flux sur l'étage inter-hubs (H-H)

Nous introduisons dans cette section un modèle destiné à déterminer localement une borne inférieure du nombre de départs sur l'étage inter-hubs. A ce stade, nous choisissons d'occulter les contraintes de traitement sur les hubs. Ainsi, nous ne considérons pas les produits 1-hubs dans ce calcul puisque sur l'étage H-H, ils n'entrent en interaction avec les produits 2-hubs que par l'intermédiaire des contraintes de traitement sur les hubs.

De manière similaire à la stratégie mise en œuvre pour calculer une borne inférieure du nombre de départs sur l'étage collecte, nous déclinons ce calcul pour chaque liaison h-h' ($(h,h') \in H^2, h \neq h'$) sur laquelle transite au moins un produit (i.e. $P_{h,h'} \neq \varnothing$).

6.4.1 Une modélisation MIP locale de l'étage inter-hub

Notons $W^p_{h_o} = [est_{h_o}(p), lst_{h_o}(p)]$ la fenêtre temporelle sur laquelle tout produit p doit nécessairement partir de son hub origine $h_o(p)$. Soit $z^\theta_{h,h'}$ la variable de décision correspondant au nombre de véhicules quittant le hub h à destination du hub h' sur la période θ (et donc arrivant en h' à la période $\theta + l(h,h')$). Définissons en outre les variables de flux x^θ_p correspondant à la quantité de produit $p \in P_{h,h'}$ quittant le hub origine $h = h_o(p)$ à destination du hub $h' = h_d(p)$ durant la période θ. Le programme linéaire mixte $[FluxHH(h,h')]$ détermine une borne inférieure du nombre de départs requis pour router les produits transitant par la liaison h-h'.

$$[FluxHH(h,h')] \qquad \min \sum_{\theta \in \Theta} z^\theta_{h,h'}$$

$$SC$$

$$\forall p \in P_{h,h'}, \qquad \sum_{\theta \in W^p_h} x^\theta_p = q(p) \tag{6.10}$$

$$\forall \theta \in \Theta_h, \qquad \sum_{p \in P_{h,h'}} x^\theta_p \leq Q_H z^\theta_{h,h'} \tag{6.11}$$

$$\forall \theta \in \Theta_h, \qquad z^\theta_{h,h'} \in \mathbb{N} \tag{6.12}$$

$$\forall \theta \in \Theta_h, \forall p \in P_{h,h'}, \qquad x^\theta_p \geq 0 \tag{6.13}$$

Proposition 6 *$\sum_{h \in H} \sum_{h' \in H} Opt[FluxHH(h,h')]$ est une borne inférieure du nombre optimal de moyens nécessaires sur les liaisons H-H pour toute solution optimale de $[P]$.*

Preuve 6 *Comme énoncé précédemment, ce calcul de borne inférieure local à l'étage inter-hubs n'intègre pas les contraintes de traitement sur les hubs. Par conséquent, les produits 2-hubs considérés dans cet étage ne sont couplés qu'à un niveau local aux liaisons inter-hubs. Le routage localement optimal des flux inter-hubs se décompose donc par segment de transport H-H.* ■

Le modèle introduit pour le calcul d'une borne inférieure du nombre de départs sur l'étage H-H est en tous points similaire à celui proposé pour évaluer l'étage C-H. La stratégie heuristique proposée pour l'étage collecte se transpose donc aisément à l'étage H-H. Les seules adaptations a opérer concernent :

- la prise en compte de la capacité Q_H du type de véhicule associé aux liaisons inter-hubs (et non de Q_c),
- la prise en compte des dates de départ au plus tôt sur le hub origine $h_o(p)$ (et non de la date de départ au plus tôt sur le CT origine $es(p)$).

Comme pour l'étage collecte, nos tests montrent que la résolution algorithmique du problème $[FluxHH(h, h')]$ donne toujours une solution avec un nombre de départs identique à celui obtenu en résolvant le modèle par solveur.

Au même titre que sur les étages C-H et H-C, la massification maximale opérée sur l'étage HH peut détériorer les possibilités de massification sur C-H et H-C voir même induire des cas de non réalisabilité.

Pour l'évaluation de la qualité des bornes inférieures introduites dans ce chapitre, nous en référons aux résultats numériques exposés en section 13.12.

7

Heuristique sérielle

Sommaire

7.1 Introduction

Cette section expose une heuristique constructive permettant d'obtenir une première solution à notre problème, mais pouvant induire des déficits en quantité routée pour certains produits (solution partielle).

L'approche développée dans ce chapitre consiste à appliquer une méthode sérielle (algorithme de liste) routant progressivement et chronologiquement les produits dans le réseau logistique. A cet effet, les routages partiels de produits sont traduits au travers de jobs sur lesquels s'applique un algorithme de liste basé sur la règle de priorité de Schrage (date échue minimale). Chaque job j impliqué dans ce processus est caractérisé par un quintuplet $[p_j, s_j, r_j, f_j, d_j]$ où p_j désigne un produit, s_j un site du routage primaire associé à ce produit (hub $h_o(p_j)$ ou $h_d(p_j)$ ou CT destination $d(p_j)$), r_j une date de démarrage au plus tôt, f_j une quantité, et d_j une date d'exécution au plus tard.

Ce job doit être ordonnancé entre les période r_j et d_j sur une ressource associée au site s_j (traitement hub ou CT destination), de capacité η_{s_j} sur chaque période.

Si la ressource associée est un CT destination, alors une fois que le job est traité, la quantité de produit f_j est routée. S'il s'agit d'un hub, ce job représente tout ou partie de la quantité du produit p_j candidat au transfert de $h_o(p_j)$ vers $h_d(p_j)$ ou de $h_d(p_j)$ vers $d(p_j)$, au plus tard à d_j, une fois réalisé son traitement hub et à concurrence de la quantité f_j.

7.2 Structures de données

Considérons les structures de données suivantes, requise pour la mise en œuvre de cette heuristique constructive :

- un compteur de temps θ permettant de contrôler le caractère chronologique de l'heuristique,
- la liste J des jobs impliqués dans le contrôle de capacité de traitement et évoluant dynamiquement en fonction de la période courante,
- la liste FIFO $B_{s,s'}$, déclinée par segment de transport s-s' et stockant chaque job traité et éventuellement candidat au transfert inter-sites s-s'. Chaque élément contenu dans $B_{s,s'}$ est caractérisé par le triplet $[p_j, d_j, f_j]$, f_j correspondant à une fraction de la quantité $q(p_j)$ disponible pour le transfert concerné (éventuellement $f_j = q(p_j)$),
- des capacités de traitement résiduelles indexées dans le temps ($\bar{\eta}_s^\theta$, $\theta \in \Theta$) sur chaque site s et initialisées à η_s.
- Les capacités de traitement de chaque site ne pouvant être dépassées, certains produits peuvent se retrouver en rejet, c'est-à-dire incomplètement routés dans leur domaine temporel imparti. Dans la procédure décrite ci-après, le contrôle de réalisabilité temporelle n'est effectué que sur la date de mise à disposition d'un produit en sortie de traitement sur son CT destination. Il s'agit alors de s'assurer que la date échue du produit est respectée. Dans le cas contraire, le produit est stocké dans une liste de rejets E. Parallèlement, le remplissage des moyens de transport disponibles sur les étages H-H (resp. C-H et H-C) est conditionnée par un paramètre $\mathcal{E}_H$ (resp $\mathcal{E}_C$) permettant un transfert inter-sites sans que le véhicule impliqué ne soit à charge pleine. Plus précisément, un départ de véhicule v est déclenché sur une liaison H-H (resp. C-H ou H-C) si son remplissage est supérieur à $\mathcal{E}_H \times cp(v)$ (resp. $\mathcal{E}_C \times cp(v)$). Plus les paramètres $\mathcal{E}_H$ et $\mathcal{E}_C$ prennent une valeur élevée, plus le degré de massification est grand mais aussi plus la quantité de produit mise en rejet est importante car des départs retardés de produits en raison de la massification impliquent qu'ils ne puissent parfois plus être traités par la suite.
- Dès qu'un produit est intégralement routé (ou mis en rejet), il est extrait d'une liste $\overline{P}$ initialisée sur la base de l'ensemble P des produits du problème. Les ordres de transport utilisés pour le routage de chaque produit p sont stockés chronologiquement et par étages (C-H, H-H et H-C) dans des listes $O_p^k(k \in]3]$, avec par convention 1 = C-H, 2 = H-H, 3 = H-C). Le sous-ensemble d'OT monopolisé est stocké dans l'ensemble $\bar{\Omega}$.
- L'heuristique débute par le routage des flux sur l'étage C-H. Différentes stratégies étant envisageables à ce stade, nous proposons ici la procédure **InitJob(**J**)**, décrite par l'algorithme 5. Dans cette procédure, le degré de massification retenu pour l'acheminement des produits jusqu'à leur hub origine est conditionné par le paramètre $\mathcal{E}_C$.

Algorithme 5 Procédure **InitJob()**

$J \leftarrow \varnothing$
$Q \leftarrow Q_c \times \mathcal{E}_C$
pour tout $c \in C$ **faire**
    pour tout $h \in H$ **faire**
        $B_{c,h} \leftarrow P_{c,h}$; Continuer $\leftarrow$ Vrai
        tant que Continuer **faire**
            $D_{c,h} \leftarrow \sum_{p \in B_{c,h}} q(p)$
            si $D_{c,h} \geq Q$ **alors**
                SelectionnerOT(c, h, θ)
            sinon
                Continuer $\leftarrow$ Faux
            fin si
        fin tant que
    fin pour
fin pour

7.3 Algorithme

L'heuristique sérielle est décrite par l'algorithme 6. La procédure **Ordonnancer**(s, θ) (algorithme 9) ordonnance les jobs disponibles sur le site s durant la période θ en respect de la capacité de traitement associée à ce site. Si un job j est traité intégralement, ce dernier est extrait de la liste J par appel à la procédure **ExtraireJob**(s, j) (algorithme 10). Dans cette dernière, si le produit p_j arrive sur son CT destination $d(p_j)$ trop tard pour être traité, il est mis en rejet (dans la liste E).

Si un produit ne peut pas être intégralement traité durant la période pendant laquelle il est ordonnancé (au sein de la procédure **Ordonnancer**(s, θ)) alors la procédure **MajJob**(j, f_j) est appelée en vue de met à jour la quantité f_j du job j.

La routine **AjouterProduit**$(B_{s,s'}, p_j, q)$ stocke quant à elle le produit p_j associé à une quantité q dans la liste $B_{s,s'}$ de telle sorte que cette quantité soit candidate au transfert inter-sites de s vers s'.

La procédure **Transferer**(s, s', θ) (algorithme 7) procède au transfert de produits sur la liaison s-s' à partir de la période θ. Ici, les paramètres $\mathcal{E}_H$ et $\mathcal{E}_C$ sont exploités pour déterminer le seuil de remplissage à partir duquel déclencher un départ. Si un départ doit être déclenché, alors la procédure **SelectionnerOT**(s, s', θ) (algorithme 8) est appelée pour sélectionner un OT ω associé à la liaison inter-sites s-s' de date de départ sur le site s égale à $st(\omega) = \theta + \epsilon_T$ $(= \theta + 1)$.

Au sein de la procédure **SelectionnerOT**(s, s', θ), la procédure **Niveau**(s, s') est appelée pour obtenir le niveau (C-H, H-C ou H-H) de l'OT ω sélectionné. Les jobs de $B_{s,s'}$ de priorité maximale (les premiers dans la file car de date d_j minimale), à concurrence de la capacité du moyen $cp(\omega)$, sont alors routés sur cet OT ω. Le(s) job(s) correspondant(s) au traitement de cette fraction de produit sur le site suivant en conformité avec le routage primaire du produit concerné est (sont) alors ajouté(s) à J de manière à émuler le contrôle de traitement les concernant. La procédure **AjouterJob**(p, q, s, s') réalise cet ajout. La procédure **DeterminerOTPeriode**(θ, s, s'), renvoie l'OT ω portant sur la liaison

s-s' ($\omega \in \Omega_{s,s'}$) et de date de départ $st(\omega) = \theta + \epsilon_T$ ($= \theta + 1$).

L'obtention de la solution de l'heuristique implique l'instanciation de variables de flux x^p_ω correspondant à la quantité du produit $p \in P$ routée par l'OT $\omega \in \Omega$. Le nombre de départs déclenchés sur chaque OT $\omega \in \Omega$ est quant à lui précisé par les variables y_ω. Ces instanciations sont effectuées dans la procédure **SelectionnerOT**(s, s', θ).

Algorithme 6 Heuristique sérielle

Entrées:

- l'ensemble des entités du problème (produits, types de véhicules, etc.)

Sorties:

- une instanciation des variables de flux x^p_ω correspondant à la quantité du produit $p \in P$ routée par l'OT $\omega \in \Omega$
- une instanciation des variables y_ω relatives au nombre de départs associé à chaque OT $\omega \in \Omega$
- une liste de produits incomplètement routés E (rejets)
- $\forall p \in P$, $\bar{q}(p)$: quantité de produit p non routée

Notations:

- $\forall s \in S, S^+_s$: ensemble des sites sur lesquels transitent des produits en provenance directe de s.

$\bar{P} \leftarrow P$
$E \leftarrow \varnothing$
$\bar{\Omega} \leftarrow \varnothing$
$\forall p \in P, \bar{q}(p) \leftarrow q(p)$
$\forall p \in P, \forall k \in]3], O^k_p \leftarrow \varnothing$
$\forall s \in S, \forall \theta \in \Theta, \bar{\eta}^\theta_s \leftarrow \eta_s$
$\forall \omega \in \Omega, y_\omega \leftarrow 0$
$\forall \omega \in \Omega, \forall p \in P, x^\omega_p \leftarrow 0$
InitJob(J)
$\theta \leftarrow \min_{j \in J} r_j$
tant que $\bar{P} \neq \varnothing \wedge \theta \leq \Theta$ **faire**
    pour tout $s \in S$ **faire**
        Ordonnancer(s, θ)
        pour tout $s' \in S^+_s$ **faire**
            Transferer(s, s', θ)
        fin pour
    fin pour
    $\theta \leftarrow \theta + 1$
fin tant que

Algorithme 7 Procédure **Transferer**(s, s', θ)

Entrées:

- s, s' : sites constitutifs de la liaison considérée
- θ : période sur laquelle opérer le transfert de produits

si $s' \in C$ **alors**
  $Q \leftarrow \mathcal{E}_C \times Q_C$
sinon
  $Q \leftarrow \mathcal{E}_H \times Q_H$
fin si
$Ok \leftarrow$Vrai
tant que Ok **faire**
  $D_s \leftarrow \sum_{j \in B_{s,s'}} f_j$
  si $D_s \geq Q$ **alors**
    SelectionnerOT(s, s', θ)
  sinon
    $Ok \leftarrow$Faux
  fin si
fin tant que

Algorithme 8 Procédure **SelectionnerOT**(s, s', θ)

Entrées:

- (s, s') : sites formant la liaison considérée
- θ : période de temps sur laquelle sélectionner un OT

$\omega \leftarrow$ **DeterminerOTPeriode**(θ, s, s')
$\bar{\Omega} \leftarrow \bar{\Omega} \oplus \{\omega\}$
$y_\omega \leftarrow y_\omega + 1$
$\bar{C} \leftarrow cp(\omega)$
tant que $\bar{C} > 0 \wedge B_{ss'} \neq \varnothing$ **faire**
  $j^* \leftarrow argmin_{j \in B_{ss'}} d_j$
  $p \leftarrow p_{j^*}$
  $q \leftarrow \min(\bar{C}, f_{j^*})$
  $k \leftarrow$**Niveau**(s, s')
  $x^p_\omega \leftarrow x^p_\omega + q$
  $O^k_p \leftarrow O^k_p \oplus \{\omega\}$
  $\bar{C} \leftarrow \bar{C} - q$
  $f_{j^*} \leftarrow f_{j^*} - q$
  si $f_{j^*} = 0$ **alors**
    $B_{s,s'} \leftarrow B_{s,s'} \ominus \{j^*\}$
  fin si
  AjouterJob(p, q, s, s')
fin tant que

Algorithme 9 Procédure **Ordonnancer**(s, θ)

Entrées:

- s : site considéré
- θ : période de temps sur laquelle ordonnancer les jobs sur le site considéré

Notations:

- q : quantité associée au job candidat à un transfert inter-sites

1: $J_s^\theta \leftarrow \{j \in J / s_j = s \wedge r_j \leq \theta\}$
2: **tant que** $J_s^\theta \neq \varnothing \wedge \bar{\eta}_s^\theta \neq 0$ **faire**
3: $j^* \leftarrow argmin_{j \in J_s^\theta} d_j$
4: **si** $f_{j^*} \leq \bar{\eta}_s^\theta$ **alors**
5: $\bar{\eta}_s^\theta \leftarrow \bar{\eta}_s^\theta - f_{j^*}$
6: $q \leftarrow f_{j^*}$
7: **si** $s = d(p_j)$ **alors**
8: $\bar{q}(p_{j^*}) \leftarrow \bar{q}(p_{j^*}) - f_{j^*}$
9: **fin si**
10: **ExtraireJob**(s, j^*)
11: **sinon**
12: $f_{j^*} \leftarrow f_{j^*} - \bar{\eta}_s^\theta$
13: $q \leftarrow \bar{\eta}_s^\theta$
14: **si** $s = d(p_j)$ **alors**
15: $\bar{q}(p_{j^*}) \leftarrow \bar{q}(p_{j^*}) - \bar{\eta}_s^\theta$
16: **fin si**
17: $\bar{\eta}_s^\theta \leftarrow 0$
18: **MajJob**(j^*, f_{j^*})
19: **fin si**
20: **si** $s = h_o(p_j)$ **alors**
21: $s' \leftarrow h_d(p_j)$
22: **fin si**
23: **si** $s = h_d(p_j)$ **alors**
24: $s' \leftarrow d(p_j)$
25: **fin si**
26: **si** $s \neq d(p_j)$ **alors**
27: **AjouterProduit**$(B_{s,s'}, p_{j^*}, q)$
28: **fin si**
29: **fin tant que**

Algorithme 10 Procédure **ExtraireJob**(s, j)

1: $J \leftarrow J \ominus \{j\}$
2: **si** $s = d(p_j) \wedge \bar{q}(p_j) = 0$ **alors**
3: $\bar{P} \leftarrow \bar{P} \ominus \{p_j\}$
4: **si** $\theta > lc(p_j)$ **alors**
5: $E \leftarrow E \cup \{p_j\}$
6: **fin si**
7: **fin si**

8

Heuristique d'amélioration

Sommaire

8.1 Introduction

La solution fournie par la méthode sérielle décrite dans la section précédente peut être sensiblement améliorée par une phase de post-processing résolvant un MIP restreint de structure similaire à celui explicité dans le chapitre 3. La restriction porte simplement sur l'ensemble d'ordres de transport support de ce programme linéaire.

Cette heuristique est utilisée entre l'heuristique sérielle et la LNS. Elle améliore la solution trouvée par l'heuristique sérielle en plaçant le résultat de celle-ci dans un des modèles mathématiques. Dans un premier temps, elle est utilisée au sein du modèle basé sur la formalisation avec des Ordres de Transport et des routages pour ensuite passer à la première version de la LNS basée sur cette modélisation. Dans la deuxième partie, nous l'avons utilisée avec la formalisation indexée dans le temps.

8.2 Exploitation de la modélisation multiflots

Au lieu de considérer l'ensemble Ω de tous les OT possiblement impliqués pour le routage des différents produits dans le réseau logistique, nous nous limitons ici aux seuls OT mis en exergue par la méthode sérielle et stockés dans $\bar{\Omega}$. Rappelons que dans ce modèle, on a :

- $\forall \omega \in \Omega, y_\omega$: nombre d'ordres de transport ω sélectionnés,
- $\forall p \in P, \forall \sigma \in R^p, x_p^\sigma$: quantité de produit p transitant par le routage σ.

On aboutit alors à la formalisation $[P_{MMF}(\bar{\Omega})]$.

$[P_{MMF}(\bar{\Omega})]$ $\min \sum_{\omega \in \bar{\Omega}} w(\omega) y_\omega$

SC

$\forall p \in P,$ $\sum_{\sigma \in R^p} x_p^\sigma = q(p)$ (3.1)

$\forall \omega \in \bar{\Omega},$ $\sum_{p \in P} \sum_{\sigma \in R^p} a_\omega^\sigma x_p^\sigma \leq cp(\omega) y_\omega$ (8.1)

$\forall h \in H, \forall (\theta, \theta') \in \Theta_h^2 / \theta \leq \theta',$ $\sum_{p \in P_h} \sum_{\sigma \in R_h(\theta,\theta')} x_p^\sigma \leq (\theta' - \theta + 1)\eta_h$ (3.7)

$\forall c \in C, \forall \theta \in \Theta_c,$ $\sum_{p \in P_c^-} \sum_{\sigma \in R_c(\theta)} x_p^\sigma \geq \Delta_c - (\theta_c^+ - \theta + 1)\eta_c$ (3.12)

$\forall \omega \in \bar{\Omega},$ $y_\omega \in \mathbb{N}$ (8.2)

$\forall p \in P, \forall \sigma \in R^p,$ $x_p^\sigma \geq 0$ (3.5)

Les routages admissibles associés à chacun des produits p, R^p, sont instanciés au moyen de la procédure **Routage**$(\cdot,\cdot)$. Soit pour $k \in [\![3]\!]$, O_p^k le tableau multidimensionnelle comprenant à la $k^{\text{ème}}$ colonne pour le produit p les OT générés par l'heuristique sérielle compatibles avec le produit p pour être en $k^{\text{ème}}$ position d'un de ces routages. Et ϵ_T désigne une latence forfaitaire de traitement.

Algorithme 11 Procédure **Routage**(p, O_p^k)

$R^p \leftarrow \varnothing$
si p produit de type 1 **alors**
  pour tout $i_1 \in [\![1, |O_p^1|]\!]$ **faire**
    pour tout $i_3 \in [\![1, |O_p^3|]\!]$ **faire**
      $\omega_1 \leftarrow O_p^1[i_1]$
      $\omega_3 \leftarrow O_p^3[i_3]$
      si $st(\omega_3) \leq st(\omega_1) + \epsilon_T$ **alors**
        $R^p \leftarrow R^p \oplus \{[\omega_1, \omega_3]\}$
      fin si
    fin pour
  fin pour
sinon
  pour tout $i_1 \in [\![1, |O_p^1|]\!]$ **faire**
    pour tout $i_2 \in [\![1, |O_p^2|]\!]$ **faire**
      pour tout $i_3 \in [\![1, |O_p^3|]\!]$ **faire**
        $\omega_1 \leftarrow O_p^1[i_1]$
        $\omega_2 \leftarrow O_p^2[i_2]$
        $\omega_3 \leftarrow O_p^3[i_3]$
        si $st(\omega_2) \leq st(\omega_1) + \epsilon_T \wedge st(\omega_3) \leq st(\omega_2) + \epsilon_T$ **alors**
          $R^p \leftarrow R^p \oplus \{[\omega_1, \omega_2, \omega_3]\}$
        fin si
      fin pour
    fin pour
  fin pour
fin si

9

Heuristique d'arrondis successifs

Sommaire

9.1 Introduction

L'heuristique d'arrondis successifs se base sur la modélisation indexée dans le temps. Nous relaxons les contraintes d'intégrité des variables $z^{\theta}_{s,s'}$. Ensuite, de manière itérative, nous résolvons la relaxation linéaire et nous fixons les valeurs des variables $z^{\theta}_{s,s'}$, obtenant ainsi une solution entière en fin de processus.

9.2 Le principe

Tout d'abord la relaxation du modèle indexée dans le temps $[RL(P_{MIT})]$ est présentée ci-dessous. Ce sont les contraintes 9.2 qui sont les contraintes relaxées.

Cette heuristique part de la solution du modèle relaxé, nous l'appelons $MIP - R$. Puis itérativement, elle fixe la valeur d'un certain nombre de variables $z^{\theta}_{s,s'}$ à leur entier supérieur. Elle s'arrête au bout d'un temps imparti ou lorsque toutes les variables $z^{\theta}_{s,s'}$ sont entières. L'algorithme 12 donne la structure de la méthode.

$[RL(P_{MIT})]$ $$\min \sum_{s\in S}\sum_{s'\in S}\sum_{\theta\in\Theta} w_{s,s'} z^{\theta}_{s,s'}$$

SC

$$\forall (s,s') \in A, \forall \theta \in \Theta \qquad \sum_{p\in P_{s,s'}} y^{\theta}_{s,s'}(p) \leq cp_{s,s'} z^{\theta}_{s,s'} \quad (4.1)$$

$$\forall s \in S, \forall \theta \in \Theta \qquad \sum_{p\in P_{s}} x^{T}_{s,\theta}(p) \leq \eta_{s} \quad (4.2)$$

$$\forall p \in P_{t2}, \forall \theta \in \Theta \quad x^{A}_{h_o(p),\theta-1}(p) + y^{\theta-l_{s(p),h_o(p)}}_{s(p),h_o(p)}(p) = x^{A}_{h_o(p),t}(p) + x^{T}_{h_o(p),\theta}(p) \quad (4.3)$$

$$\forall p \in P_{t2}, \forall \theta \in \Theta \quad x^{A}_{h_d(p),\theta-1}(p) + y^{\theta-l_{h_o(p),h_d(p)}}_{h_o(p),h_d(p)}(p) = x^{A}_{h_d(p),\theta}(p) + x^{T}_{h_d(p),\theta}(p) \quad (4.4)$$

$$\forall p \in P_{t1}, \forall \theta \in \Theta \quad x^{A}_{h_d(p),\theta-1}(p) + y^{\theta-l_{s(p),h_d(p)}}_{s(p),h_d(p)}(p) = x^{A}_{h_d(p),\theta}(p) + x^{T}_{h_d(p),\theta}(p) \quad (4.5)$$

$$\forall p \in P, \forall \theta \in \Theta \quad x^{A}_{d(p),\theta-1}(p) + y^{\theta-l_{h_d(p),d(p)}}_{h_d(p),d(p)}(p) = x^{A}_{d(p),\theta}(p) + x^{T}_{d(p),\theta}(p) \quad (4.6)$$

$$\forall p \in P \qquad x^{D}_{s(p),es(p)-1}(p) = q(p) \quad (4.7)$$

$$\forall p \in P, \forall \theta \in \Theta \qquad x^{D}_{s(p),\theta-1}(p) = x^{D}_{s(p),\theta}(p) + y^{\theta}_{s(p),h_o(p)}(p) \quad (4.8)$$

$$\forall p \in P_{t2}, \forall \theta \in \Theta \quad x^{D}_{h_o(p),\theta-1}(p) + x^{T}_{h_o(p),\theta-1}(p) = x^{D}_{h_o(p),\theta}(p) + y^{\theta}_{h_o(p),h_d(p)}(p) \quad (4.9)$$

$$\forall p \in P_{t2}, \forall \theta \in \Theta \quad x^{D}_{h_d(p),\theta-1}(p) + x^{T}_{h_d(p),\theta-1}(p) = x^{D}_{h_d(p),\theta}(p) + y^{\theta}_{h_d(p),d(p)}(p) \quad (4.10)$$

$$\forall p \in P_{t1}, \forall \theta \in \Theta \quad x^{D}_{h_d(p),\theta-1}(p) + x^{T}_{h_d(p),\theta-1}(p) = x^{D}_{h_d(p),t}(p) + y^{\theta}_{h_d(p),d(p)}(p) \quad (4.11)$$

$$\forall p \in P, \forall \theta \in \Theta \qquad x^{D}_{d(p),\theta-1}(p) + x^{T}_{d(p),\theta-1}(p) = x^{D}_{d(p),\theta}(p) \quad (4.12)$$

$$\forall p \in P \qquad x^{D}_{d(p),lc(p)} = q(p) \quad (9.1)$$

$$\forall (s,s') \in A, \forall \theta \in \Theta \qquad z^{\theta}_{s,s'} \geq 0 \quad (9.2)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{A}_{s,\theta}(p) \geq 0 \quad (4.14)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{D}_{s,\theta}(p) \geq 0 \quad (4.15)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{T}_{s,\theta}(p) \geq 0 \quad (4.16)$$

$$\forall (s,s') \in A, \forall p \in P_{s,s'}, \forall \theta \in \Theta \qquad y^{\theta}_{s,s'}(p) \geq 0 \quad (4.17)$$

Algorithme 12 Heuristique d'arrondis successifs

Entrées:

- T'_{max} : temps maximal donné pour l'exécution de l'heuristique
- Z : ensemble des variables $z^{\theta}_{s,s'}$

1: $C \leftarrow \varnothing$
2: $F \leftarrow \varnothing$
3: $stop \leftarrow faux$
4: **tant que** $temps \leq T'_{max} \wedge \neg stop$ **faire**
5: $s \leftarrow$ **solve**$(MIP - R)$
6: $Z \leftarrow$ **retrieve**(s)
7: $stop \leftarrow vrai$
8: **pour tout** $z \in Z$ **faire**
9: **si valeur**$(z) \geq \lceil$**valeur**$(z)\rceil - \epsilon \wedge z \notin F$ **alors**
10: $C \leftarrow C \oplus \{$**fix**$(z, \lceil$**valeur**$(z)\rceil)\}$
11: $F \leftarrow F \oplus \{z\}$
12: **fin si**
13: **si valeur**$(z) \notin \mathbb{N}$ **alors**
14: $stop \leftarrow faux$
15: **fin si**
16: **fin pour**
17: $\epsilon \leftarrow \epsilon + \alpha$
18: **miseajour**$(MIP - R, C)$
19: **fin tant que**

A chaque itération, l'heuristique résout le modèle relaxé $MIP - R$ avec la procédure **solve**$(MIP - R)$. Puis, elle récupère l'ensemble des variables $z^{\theta}_{s,s'}$ (**retrieve**(s)). Pour chacune de ces variables, elle contrôle si la distance entre sa valeur et l'entier immédiatement supérieur est inférieure ou égale à ϵ (un paramètre ajusté itérativement). Si c'est le cas et que cette variable n'a pas déjà été fixée ($z \notin F$), alors nous créons une contrainte fixant la valeur de la variable à celle de son entier supérieur (**fix**$(z, \lceil$valeur$(z)\rceil)$). De plus, nous vérifions que la solution trouvée n'est pas entière, c'est à dire que toutes les variables $z^{\theta}_{s,s'}$ prennent un entier pour valeur. Enfin, le $MIP - R$ est mis à jour avec les contraintes fixant les variables pour être de nouveau résolu (**miseajour**$(MIP - R, C)$).

10

Heuristique d'échelonnement de capacité

Sommaire

10.1 Introduction

Cette section présente une heuristique d'échelonnement de capacité (Capacity Scaling) pour résoudre notre problème. Cette approche a été proposée en 2009 par Katayama et al. [17] dans le cadre de la conception de réseaux sous contraintes de multiflots avec capacités sur les arcs. Cette méthode tente d'améliorer itérativement une solution de référence en faisant évoluer la capacité des arcs en fonction des flux qui y transitent. Pour sa mise en œuvre, nous utilisons la modélisation mathématique indexée dans le temps proposée dans le chapitre 4. Néanmoins, celle-ci doit être adaptée en raison d'une spécification fonctionnelle de notre problématique autorisant l'utilisation d'un nombre $z^{\theta}_{s,s'}$ (illimité) de véhicules de capacité $cp_{s,s'}$ sur un segment transport (s, s') et sur une période θ, alors que la décision basique relative au développement proposé par Katayama et al est binaire, spécifiant la disponibilité ou non d'une capacité C.

$[P_{MITAC}]$ $$\min \sum_{s\in S}\sum_{s'\in S}\sum_{\theta\in\Theta} w_{s,s'} z^{\theta}_{s,s'}$$

SC

$$\forall (s,s') \in A, \forall \theta \in \Theta \qquad \sum_{p\in P_{s,s'}} y^{\theta}_{s,s'}(p) \leq cp_{s,s'} z^{\theta}_{s,s'} \quad (4.1)$$

$$\forall s \in S, \forall \theta \in \Theta \qquad \sum_{p\in P_s} x^{T}_{s,\theta}(p) \leq \eta_s \quad (4.2)$$

$$\forall p \in P_{t2}, \forall \theta \in \Theta \quad x^{A}_{h_o(p),\theta-1}(p) + y^{\theta-l_{s(p),h_o(p)}}_{s(p),h_o(p)}(p) = x^{A}_{h_o(p),t}(p) + x^{T}_{h_o(p),\theta}(p) \quad (4.3)$$

$$\forall p \in P_{t2}, \forall \theta \in \Theta \quad x^{A}_{h_d(p),\theta-1}(p) + y^{\theta-l_{h_o(p),h_d(p)}}_{h_o(p),h_d(p)}(p) = x^{A}_{h_d(p),\theta}(p) + x^{T}_{h_d(p),\theta}(p) \quad (4.4)$$

$$\forall p \in P_{t1}, \forall \theta \in \Theta \quad x^{A}_{h_d(p),\theta-1}(p) + y^{\theta-l_{s(p),h_d(p)}}_{s(p),h_d(p)}(p) = x^{A}_{h_d(p),\theta}(p) + x^{T}_{h_d(p),\theta}(p) \quad (4.5)$$

$$\forall p \in P, \forall \theta \in \Theta \quad x^{A}_{d(p),\theta-1}(p) + y^{\theta-l_{h_d(p),d(p)}}_{h_d(p),d(p)}(p) = x^{A}_{d(p),\theta}(p) + x^{T}_{d(p),\theta}(p) \quad (4.6)$$

$$\forall p \in P \qquad x^{D}_{s(p),es(p)-1}(p) = q(p) \quad (4.7)$$

$$\forall p \in P, \forall \theta \in \Theta \qquad x^{D}_{s(p),\theta-1}(p) = x^{D}_{s(p),\theta}(p) + y^{\theta}_{s(p),h_o(p)}(p) \quad (4.8)$$

$$\forall p \in P_{t2}, \forall \theta \in \Theta \quad x^{D}_{h_o(p),\theta-1}(p) + x^{T}_{h_o(p),\theta-1}(p) = x^{D}_{h_o(p),\theta}(p) + y^{\theta}_{h_o(p),h_d(p)}(p) \quad (4.9)$$

$$\forall p \in P_{t2}, \forall \theta \in \Theta \quad x^{D}_{h_d(p),\theta-1}(p) + x^{T}_{h_d(p),\theta-1}(p) = x^{D}_{h_d(p),\theta}(p) + y^{\theta}_{h_d(p),d(p)}(p) \quad (4.10)$$

$$\forall p \in P_{t1}, \forall \theta \in \Theta \quad x^{D}_{h_d(p),\theta-1}(p) + x^{T}_{h_d(p),\theta-1}(p) = x^{D}_{h_d(p),t}(p) + y^{\theta}_{h_d(p),d(p)}(p) \quad (4.11)$$

$$\forall p \in P, \forall \theta \in \Theta \qquad x^{D}_{d(p),\theta-1}(p) + x^{T}_{d(p),\theta-1}(p) = x^{D}_{d(p),\theta}(p) \quad (4.12)$$

$$\forall p \in P \qquad x^{D}_{d(p),lc(p)} = q(p) \quad (4.13)$$

$$\forall (s,s') \in A, \forall [\theta^-,\theta^+] \in \Theta_{s,s'} \qquad \left\lceil \frac{\sum_{p\in P_{\theta^-,\theta^+}} q(p)}{cp_{s,s'}} \right\rceil \leq \sum_{\theta=\theta^-}^{\theta^+} z^{\theta}_{s,s'} \quad (5.3)$$

$$\forall (s,s') \in A, \forall \theta \in \Theta, \forall p \in P^{\theta}_{s,s'} \qquad y^{\theta}_{s,s'}(p) \leq q(p) z^{\theta}_{s,s'} \quad (5.5)$$

$$\forall (s,s') \in A, \forall \theta \in \Theta \qquad z^{\theta}_{s,s'} \in \mathbb{N} \quad (4.14)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{A}_{s,\theta}(p) \geq 0 \quad (4.15)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{D}_{s,\theta}(p) \geq 0 \quad (4.16)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{T}_{s,\theta}(p) \geq 0 \quad (4.17)$$

$$\forall (s,s') \in A, \forall p \in P_{s,s'}, \forall \theta \in \Theta \qquad y^{\theta}_{s,s'}(p) \geq 0 \quad (4.18)$$

Dans la suite de ce chapitre, le bloc du modèle $[P_{MITAC}]$ (modèle décrit dans le chapitre 4 auquel nous avons rajoutés les coupes et les surrogates décrites dans le chapitre 5) contenant les équations 4.3, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 4.11, 4.12 et 4.13 est appelé $Loi\ de\ conservation\ des\ flux$.

Dans l'objectif de transposer cette démarche à notre problématique, nous définissons une borne supérieure B du nombre de départ sur toute liaison inter sites (s,s') et pour toute période, induisant ainsi une capacité maximale $B.cp_{s,s'}$. Deux versions sont alors déclinées dans les développements qui suivent :

– La première considère une unique variable de décision de nature bivalente $zB^{\theta}_{s,s'}$ par liaison (s,s') et par période. La capacité agrégée initiale de l'arc associé dans le graphe indexé dans le temps est égale à $C^{\theta}_{s,s'} = B.cp_{s,s'}$.

– La seconde version considère quant à elle B variables binaires $zB^{\theta,b}_{s,s'}$ $(b = 1, .., B)$ associées aux B véhicules disponibles pour la liaison inter sites (s, s') sur toute période. La capacité élémentaire initiale résultante associée à chaque arc est alors égale à : $C^{\theta,b}_{s,s'} = cp_{s,s'}$.

Dans les deux versions, la méthode procède à une diminution itérative des capacités des arcs associés aux variables binaires, en cherchant à faire converger ces dernières vers 0 ou 1. Pour cela, à chaque itération, elle résout la relaxation linéaire d'un modèle en $0-1$ que nous déclinons pour chacune de ces stratégies. Lorsque dans la solution du modèle relaxé, un certain nombre de ces variables sont instanciées à 0 ou 1, nous récupérons pour chaque segment de transport et toute période, les valeurs des variables ainsi que les capacités des arcs associés. Nous calculons alors un système de bornes inférieures $LB^{\theta}_{s,s'}$ et supérieures $UB^{\theta}_{s,s'}$ portant sur ces variables, ces dernières étant alors injectées dans le modèle. Nous obtenons ainsi le MIP $[P_{Borne}]$.

$[P_{Borne}]$ $$\min \sum_{s\in S}\sum_{s'\in S}\sum_{\theta\in\Theta} w_{s,s'} z^{\theta}_{s,s'}$$

SC

$$\forall (s,s') \in A, \forall \theta \in \Theta \qquad \sum_{p\in P_{s,s'}} y^{\theta}_{s,s'}(p) \leq cp_{s,s'} z^{\theta}_{s,s'} \qquad (4.1)$$

$$\forall s \in S, \forall \theta \in \Theta \qquad \sum_{p\in P_s} x^{T}_{s,\theta}(p) \leq \eta_s \qquad (4.2)$$

$Loi\ de\ conservation\ des\ flux$

$$\forall (s,s') \in A, \forall [\theta^-,\theta^+] \in \Theta_{s,s'} \qquad \left\lceil \frac{\sum_{p\in P_{\theta^-,\theta^+}} q(p)}{cp_{s,s'}} \right\rceil \leq \sum_{\theta=\theta^-}^{\theta^+} z^{\theta}_{s,s'} \qquad (5.3)$$

$$\forall (s,s') \in A, \forall \theta \in \Theta, \forall p \in P^{\theta}_{s,s'} \qquad y^{\theta}_{s,s'}(p) \leq q(p) z^{\theta}_{s,s'} \qquad (5.5)$$

$$\forall (s,s') \in A, \forall \theta \in \Theta \qquad LB^{\theta}_{s,s'} \leq z^{\theta}_{s,s'} \leq UB^{\theta}_{s,s'} \qquad (10.1)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{A}_{s,\theta}(p) \geq 0 \qquad (4.15)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{D}_{s,\theta}(p) \geq 0 \qquad (4.16)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{T}_{s,\theta}(p) \geq 0 \qquad (4.17)$$

$$\forall (s,s') \in A, \forall p \in P_{s,s'}, \forall \theta \in \Theta \qquad y^{\theta}_{s,s'}(p) \geq 0 \qquad (4.18)$$

10.2 Version 1 : Modélisation agrégée

Dans cette version, le MIP initial est $[P_{EC1}]$.

$$[P_{EC1}] \qquad \min \sum_{s\in S}\sum_{s'\in S}\sum_{\theta\in\Theta} w_{s,s'} zB^{\theta}_{s,s'}$$

SC

$$\forall (s,s') \in A, \forall \theta \in \Theta \qquad \sum_{p\in P_{s,s'}} y^{\theta}_{s,s'}(p) \leq C^{\theta}_{s,s'} zB^{\theta}_{s,s'} \qquad (10.2)$$

$$\forall s \in S, \forall \theta \in \Theta \qquad \sum_{p\in P_s} x^{T}_{s,\theta}(p) \leq \eta_s \qquad (4.1)$$

$$Loi\ de\ conservation\ des\ flux$$

$$\forall (s,s') \in A, \forall \theta \in \Theta \qquad zB^{\theta}_{s,s'} \in \{0,1\} \qquad (10.3)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{A}_{s,\theta}(p) \geq 0 \qquad (4.15)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{D}_{s,\theta}(p) \geq 0 \qquad (4.16)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{T}_{s,\theta}(p) \geq 0 \qquad (4.17)$$

$$\forall (s,s') \in A, \forall p \in P_{s,s'}, \forall \theta \in \Theta \qquad y^{\theta}_{s,s'}(p) \geq 0 \qquad (4.18)$$

Comme nous l'avons mentionné précédemment, l'heuristique d'échelonnement de capacité procède à une révision de la capacité $C^{\theta}_{s,s'}$ des arcs associés aux variables de décision $zB^{\theta}_{s,s'}$ en fonction des flux produits qui y transitent. Dans cet objectif, la relaxation linéaire du MIP $[P_{EC1}]$ est résolue en exploitant une capacité C' au lieu de C avec $C'(1) = C$. A l'itération k, la relaxation linéaire résolue (notant $C'(k)$ le vecteur des capacités révisées) est $[LR_{PEC1}(C'(k))]$.

$$[LR_{PEC1}(C'(k))] \qquad \min \sum_{s\in S}\sum_{s'\in S}\sum_{\theta\in\Theta} w_{s,s'} zB^{\theta}_{s,s'}$$

SC

$$\forall (s,s') \in A, \forall \theta \in \Theta \qquad \sum_{p\in P_{s,s'}} y^{\theta}_{s,s'}(p) \leq C'^{\theta}_{s,s'}(k) zB^{\theta}_{s,s'} \qquad (10.4)$$

$$\forall s \in S, \forall \theta \in \Theta \qquad \sum_{p\in P_s} x^{T}_{s,\theta}(p) \leq \eta_s \qquad (4.1)$$

$$Loi\ de\ conservation\ des\ flux$$

$$\forall (s,s') \in A, \forall \theta \in \Theta \qquad 0 \leq zB^{\theta}_{s,s'} \leq \frac{C^{\theta}_{s,s'}}{C'^{\theta}_{s,s'}(k)} \qquad (10.5)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{A}_{s,\theta}(p) \geq 0 \qquad (4.15)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{D}_{s,\theta}(p) \geq 0 \qquad (4.16)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{T}_{s,\theta}(p) \geq 0 \qquad (4.17)$$

$$\forall (s,s') \in A, \forall p \in P_{s,s'}, \forall \theta \in \Theta \qquad y^{\theta}_{s,s'}(p) \geq 0 \qquad (4.18)$$

La partie de droite des contraintes 10.4 est modifiée en $C'^{\theta}_{s,s'}(k)$. De plus, la borne supérieure des variables $zB^{\theta}_{s,s'}$ (contraintes 10.5) est changée en $\frac{C^{\theta}_{s,s'}}{C'^{\theta}_{s,s'}(k)}$ afin de permettre aux flux de pouvoir, si besoin, remonter à la capacité initiale $C^{\theta}_{s,s'}$.

Soit $(\widetilde{x}, \widetilde{y}, \widetilde{z})$ une solution optimale de $[LR_{PEC1}(C'(k))]$, avec $\widetilde{y} = (\widetilde{y}^{\theta}_{s,s'})^{\theta\in\Theta}_{(s,s')\in A}$ et $\widetilde{z} = (\widetilde{zB}^{\theta}_{s,s'})^{\theta\in\Theta}_{(s,s')\in A}$. A l'itération suivante, le vecteur des capacités $C'(k+1)$ est révisée en posant : $C'(k+1) = \lambda\widetilde{y} + (1-\lambda)C'(k)$, $0 \leq \lambda \leq 1$ désignant un paramètre ajusté dynamiquement et permettant d'éviter une convergence prématurée du processus (vers une solution probablement de faible qualité). Si, toujours à une itération k, le nombre des variables de décision $zB^{\theta}_{s,s'}$ convergeant vers 0 ou 1 dépasse un seuil préfixé γ, alors nous calculons les bornes inférieures $LB^{\theta}_{s,s'}$ et supérieures $UB^{\theta}_{s,s'}$ portant sur les variables $z^{\theta}_{s,s'}$ lesquelles sont injectées dans le MIP $[P_{Borne}]$. Ces bornes sont fixées en réalisant, pour chaque segment transport (s, s') et chaque période θ, un bilan des flux y transitant. Notant $val^{\theta}_{s,s'}(k)$ cette quantité, on a :

$$\forall (s,s') \in A, \forall \theta \in \Theta, val^{\theta}_{s,s'}(k) = \widetilde{zB}^{\theta}_{s,s'} \times C'^{\theta}_{s,s'}(k) \tag{10.6}$$

La fraction $\frac{Val^{\theta}_{s,s'}(k)}{cp_{s,s'}}$ donne alors une estimation du nombre "théorique" de véhicule devant transiter entre s et s' sur la période θ. Une technique d'arrondi permet enfin d'obtenir le jeu de bornes cherché (algorithme 13).

Algorithme 13 Procédure **borne1**($\widetilde{Z}$,$C'(k)$)

Entrées:

$\epsilon \in [0,1]$

pour tout $(s,s') \in A$ **et** $\theta \in \Theta$ **faire**
  $val^{\theta}_{s,s'}(k) = \widetilde{zB}^{\theta}_{s,s'} \times C'^{\theta}_{s,s'}(k)$
  si $\frac{val^{\theta}_{s,s'}(k)}{cp_{i,j}} + \epsilon \geq \left\lceil \frac{val^{\theta}_{s,s'}(k)}{cp_{i,j}} \right\rceil$ **alors**
    $LB^{\theta}_{s,s'} \leftarrow \left\lceil \frac{val^{\theta}_{s,s'}(k)}{cp_{i,j}} \right\rceil$
    $UB^{\theta}_{s,s'} \leftarrow \left\lceil \frac{val^{\theta}_{s,s'}(k)}{cp_{i,j}} \right\rceil$
  sinon
    $LB^{\theta}_{s,s'} \leftarrow \left\lfloor \frac{val^{\theta}_{s,s'}(k)}{cp_{i,j}} \right\rfloor$
    $UB^{\theta}_{s,s'} \leftarrow \left\lceil \frac{val^{\theta}_{s,s'}(k)}{cp_{i,j}} \right\rceil$
  fin si
fin pour
resoudre($[P_{Borne}]$)
renvoyer $Opt([P_{Borne}])$

L'algorithme 14 synthétise le déroulement générale de notre heuristique.

Algorithme 14 Heuristique d'échelonnement de capacités V1

Entrées:

$\lambda \in [0,1]$; $\epsilon \in [0,1]$; $\gamma \in [0,1]$; $MaxIt \in \mathbb{N}$

1: $C'^{\theta}_{s,s'}(1) \leftarrow C^{\theta}_{s,s'}$; $UB \leftarrow \infty$
2: $k \leftarrow 1$; $nbVarZ \leftarrow 0$; $nbVarZFix \leftarrow 0$
3: **tant que** $k \leq MaxIt \wedge UB \neq \infty$ **faire**
4: $(\widetilde{Y}, \widetilde{Z}) \leftarrow$ **resoudre**$([LR_{PEC1}(C'(k))])$
5: **pour tout** $(s,s') \in A$ **et** $\theta \in \Theta$ **faire**
6: **si** $\widetilde{zB}^{\theta}_{s,s'} = 0$ **alors**
7: $\overline{zB}^{\theta}_{s,s'} \leftarrow 0$
8: $nbVarZFix \leftarrow nbVarZFix + 1$
9: **sinon**
10: **si** $\widetilde{zB}^{\theta}_{s,s'} \geq 1 - \epsilon$ **alors**
11: $\overline{zB}^{\theta}_{s,s'} \leftarrow 1$
12: $nbVarZFix \leftarrow nbVarZFix + 1$
13: **sinon**
14: $\overline{zB}^{\theta}_{s,s'} \leftarrow$ libre
15: **fin si**
16: **fin si**
17: $nbVarZ \leftarrow nbVarZ + 1$
18: **fin pour**
19: **si** $\frac{nbVarZFix}{nbVarZ} > \gamma$ **alors**
20: $Opt(k) \leftarrow$ **borne1**$(\widetilde{Z}, C'(k))$
21: **si** $Opt(k) < UB$ **alors**
22: $UB \leftarrow Opt(k)$
23: **fin si**
24: **fin si**
25: $k \leftarrow k + 1$
26: **pour tout** $(s,s') \in A$ **et** $\theta \in \Theta$ **faire**
27: $C'^{\theta}_{s,s'}(k) \leftarrow \lambda \sum_{p \in P^{\theta}_{s,s'}} \widetilde{y}^{\theta}_{s,s'}(p) + (1-\lambda) C'^{\theta}_{s,s'}(k-1)$
28: $UB(zB^{\theta}_{s,s'}) \leftarrow \frac{C^{\theta}_{s,s'}}{C'^{\theta}_{s,s'}(k)}$
29: **fin pour**
30: **fin tant que**

10.3 Version 2 : Modélisation désagrégée

Dans cette version, le modèle initial est $[P_{EC2}]$.

$$[P_{EC2}] \qquad \min \sum_{s\in S}\sum_{s'\in S}\sum_{\theta\in\Theta}\sum_{b\in]\!]B]\!]} w_{s,s'} zB^{\theta,b}_{s,s'}$$

SC

$$\forall (s,s') \in A, \forall \theta \in \Theta, \forall b \in]\!]B]\!] \qquad \sum_{p\in P_{s,s'}} y^{\theta}_{s,s'}(p) \leq C^{\theta,b}_{s,s'} zB^{\theta,b}_{s,s'} \qquad (10.7)$$

$$\forall s \in S, \forall \theta \in \Theta \qquad \sum_{p\in P_s} x^{T}_{s,\theta}(p) \leq \eta_s \qquad (4.1)$$

$$Loi\ de\ conservation\ des\ flux$$

$$\forall (s,s') \in A, \forall \theta \in \Theta, \forall b \in]\!]B]\!] \qquad zB^{\theta,b}_{s,s'} \in \{0,1\} \qquad (10.8)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{A}_{s,\theta}(p) \geq 0 \qquad (4.15)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{D}_{s,\theta}(p) \geq 0 \qquad (4.16)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{T}_{s,\theta}(p) \geq 0 \qquad (4.17)$$

$$\forall (s,s') \in A, \forall p \in P_{s,s'}, \forall \theta \in \Theta \qquad y^{\theta}_{s,s'}(p) \geq 0 \qquad (4.18)$$

A l'instar de la version 1, la version 2 de l'heuristique d'échelonnement de capacité procède à une révision de la capacité $C^{\theta,b}_{s,s'}$ des arcs associés aux variables de décision $zB^{\theta,b}_{s,s'}$ en fonction des flux produits qui y transitent. Dans cet objectif, la relaxation linéaire du MIP $[P_{EC2}]$ est résolue en exploitant une capacité C' au lieu de C avec $C'(1) = C$. A l'itération k, notons la relaxation linéaire résolue est ainsi (notant $C'(k)$ le vecteur des capacités révisées) est $[LR_{PEC2}(C'(k))]$:

$$[LR_{PEC2}(C'(k))] \qquad \min \sum_{s\in S}\sum_{s'\in S}\sum_{\theta\in\Theta}\sum_{b\in]\!]B]\!]} w_{s,s'} zB^{\theta,b}_{s,s'}$$

SC

$$\forall (s,s') \in A, \forall \theta \in \Theta, \forall b \in]\!]B]\!] \qquad \sum_{p\in P_{s,s'}} y^{\theta}_{s,s'}(p) \leq C'^{\theta,b}_{s,s'}(k) zB^{\theta,b}_{s,s'} \qquad (10.9)$$

$$\forall s \in S, \forall \theta \in \Theta \qquad \sum_{p\in P_s} x^{T}_{s,\theta}(p) \leq \eta_s \qquad (4.1)$$

$$Loi\ de\ conservation\ des\ flux$$

$$\forall (s,s') \in A, \forall \theta \in \Theta, \forall b \in]\!]B]\!] \qquad 0 \leq zB^{\theta,b}_{s,s'} \leq \frac{C^{\theta,b}_{s,s'}}{C'^{\theta,b}_{s,s'}(k)} \qquad (10.10)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{A}_{s,\theta}(p) \geq 0 \qquad (4.15)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{D}_{s,\theta}(p) \geq 0 \qquad (4.16)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{T}_{s,\theta}(p) \geq 0 \qquad (4.17)$$

$$\forall (s,s') \in A, \forall p \in P_{s,s'}, \forall \theta \in \Theta \qquad y^{\theta}_{s,s'}(p) \geq 0 \qquad (4.18)$$

Comme dans la première version, la partie de droite des contraintes 10.9 est modifiée en $C'^{\theta,b}_{s,s'}(k)$. De plus, la borne supérieure des variables $zB^{\theta,b}_{s,s'}$ (contraintes 10.10) est changée en $\frac{C^{\theta}_{s,s'}}{C'^{\theta,b}_{s,s'}(k)}$ afin de permettre aux flux de pouvoir, si besoin, remonter à la capacité initiale $C^{\theta,b}_{s,s'}$.

Soit $(\widetilde{x}, \widetilde{y}, \widetilde{z})$ une solution optimale de $[LR_{PEC2}(C'(k))]$, avec $\widetilde{y} = (\widetilde{y}^{\theta}_{s,s'})^{\theta\in\Theta}_{(s,s')\in A}$ et $\widetilde{z} = (\widetilde{zB}^{\theta,b}_{s,s'})^{\theta\in\Theta, b\in]\!]B]\!]}_{(s,s')\in A}$. A l'itération suivante, le vecteur des capacités $C'(k+1)$ est également révisée en posant : $C'(k+1) = \lambda\widetilde{y} + (1-\lambda)C'(k)$, $(0 \leq \lambda \leq 1)$. Si, toujours à une itération k, le nombre des variables de décision $zB^{\theta,b}_{s,s'}$ convergeant vers 0 ou 1 dépasse un seuil préfixé, alors nous calculons les bornes inférieures $LB^{\theta}_{s,s'}$ et supérieures $UB^{\theta}_{s,s'}$ portant sur les variables $z^{\theta}_{s,s'}$ lesquelles sont injectées dans le MIP $[P_{Borne}]$. Ces bornes sont fixées en réalisant, pour chaque segment transport (s, s') et chaque période θ, du nombre de véhicules monopolisés. Notant $som^{\theta}_{s,s'}(k)$ cette quantité, on a :

$$\forall (s,s') \in A, \forall \theta \in \Theta, som^{\theta}_{s,s'}(k) = \sum_{b\in]\!]B]\!]} \widetilde{zB}^{\theta,b}_{s,s'} \tag{10.11}$$

Cette somme est une estimation du nombre "théorique" de véhicule devant transiter entre s et s' sur la période θ. Une technique d'arrondi permet enfin d'obtenir le jeu de bornes cherchés (algorithme 15).

Algorithme 15 Procédure **borne2**($\widetilde{Z}$,$C'(k)$)

Entrées:

$\epsilon \in [0,1]$

pour tout $(s,s') \in A$ **et** $\theta \in \Theta$ **faire**
  $som^{\theta}_{s,s'}(k) = \sum_{b\in]\!]B]\!]} \widetilde{zB}^{\theta,b}_{s,s'}$
  si $som^{\theta}_{s,s'}(k) + \epsilon \geq \lceil som^{\theta}_{s,s'}(k) \rceil$ **alors**
    $LB^{\theta}_{s,s'} \leftarrow \lceil som^{\theta}_{s,s'}(k) \rceil$
    $UB^{\theta}_{s,s'} \leftarrow \lceil som^{\theta}_{s,s'}(k) \rceil$
  sinon
    $LB^{\theta}_{s,s'} \leftarrow \lfloor som^{\theta}_{s,s'}(k) \rfloor$
    $UB^{\theta}_{s,s'} \leftarrow \lceil som^{\theta}_{s,s'}(k) \rceil$
  fin si
fin pour
resoudre($[P_{Borne}]$)
renvoyer $Opt([P_{Borne}])$

L'algorithme 16 synthétise le déroulement générale de notre heuristique en version 2.

Algorithme 16 Heuristique d'échelonnement de capacités V2

Entrées:

$\lambda \in [0,1]$,$\epsilon \in [0,1]$, $\gamma \in [0,1]$, $MaxIt \in \mathbb{N}$

$C'^{\theta}_{s,s'}(1) \leftarrow C^{\theta}_{s,s'}$, $UB \leftarrow \infty$; $k \leftarrow 1$, $nbVarZ \leftarrow 0$, $nbVarZFix \leftarrow 0$
tant que $k \leq MaxIt \wedge UB \neq \infty$ **faire**
  $(\widetilde{Y}, \widetilde{Z}) \leftarrow$ **resoudre**($[LR_{PEC2}(C'(k))]$)
  pour tout $(s,s') \in A$ **et** $\theta \in \Theta$ **faire**
    $som^{\theta}_{s,s'}(k) = \sum_{b \in]\!]B]\!]} \widetilde{zB}^{\theta,b}_{s,s'}$
    pour tout $b \in]\!]B]\!]$ **faire**
      si $b \leq som^{\theta}_{s,s'}(k) + \epsilon$ **alors**
        $\overline{zB}^{\theta,b}_{s,s'} \leftarrow 1$
        $nbVarZFix \leftarrow nbVarZFix + 1$
      sinon
        si $b = \left\lceil som^{\theta}_{s,s'}(k) \right\rceil \wedge som^{\theta}_{s,s'}(k) + \epsilon < \left\lceil som^{\theta}_{s,s'}(k) \right\rceil$ **alors**
          $\overline{zB}^{\theta,b}_{s,s'} \leftarrow$ libre
        sinon
          $\overline{zB}^{\theta,b}_{s,s'} \leftarrow 0$
          $nbVarZFix \leftarrow nbVarZFix + 1$
        fin si
      fin si
    fin pour
  fin pour
  si $\frac{nbVarZFix}{nbVarZ} > \gamma$ **alors**
    $Opt(k) \leftarrow$ **borne2**($\widetilde{Z}$,$C'(k)$)
    si $Opt(k) < UB$ **alors**
      $UB \leftarrow Opt(k)$
    fin si
  fin si
  $k \leftarrow k + 1$
  pour tout $(s,s') \in A$ **et** $\theta \in \Theta$ **faire**
    pour tout $b \in]\!]B]\!]$ **faire**
      si $\overline{zB}^{\theta,b}_{s,s'} = 1$ **alors**
        $C'^{\theta,b}_{s,s'}(k) \leftarrow C'^{\theta,b}_{s,s'}(k-1)$
      sinon
        si $\overline{zB}^{\theta,b}_{s,s'} = 0$ **alors**
          $C'^{\theta,b}_{s,s'}(k) \leftarrow (1-\lambda) C'^{\theta,b}_{s,s'}(k-1)$
        sinon
          $C'^{\theta,b}_{s,s'}(k) \leftarrow \lambda(\sum_{p \in P^{\theta}_{s,s'}} \widetilde{y}^{\theta}_{s,s'}(p) - cp_{s,s'} \left\lceil som^{\theta}_{s,s'}(k) \right\rceil)$
            $+(1-\lambda) C'^{\theta,b}_{s,s'}(k-1)$
        fin si
      fin si
      $UB(zB^{\theta,b}_{s,s'}) \leftarrow \frac{C^{\theta,b}_{s,s'}}{C'^{\theta,b}_{s,s'}(k)}$
    fin pour
  fin pour
fin tant que

11

Large Neighborhood Search

Sommaire

11.1 Introduction

Dans cette section, nous présentons deux exploitations de la technique de LNS (Large Neighborhood Search) pour la résolution de notre problématique en lien avec chacune des modélisations que nous avons proposées. Cette stratégie fait partie de la famille des méthodes à très grands voisinages (Very Large Neighborhood Search : VLNS) initiée par Ahuja et al en 2002 [2]. Historiquement, la méthode a été proposée par Shaw [23] dans le cadre de la résolution d'un problème de tournées de véhicules. Son principe consiste à améliorer une solution initiale en alternant itérativement une phase de destruction de la solution courante et une phase de reconstruction d'une solution alternative supposée améliorer la solution de référence. Les voisinages de la LNS sont ainsi définis implicitement par deux types d'opérateurs :

- opérateurs de destruction,
- opérateurs de réparation.

Nous proposons une adaptation de cette technique à notre problème, nous permettant d'améliorer la planification du routage des produits au sein du réseau logistique à partir d'une solution heuristique initiale, dans des temps calcul acceptables pour un niveau de qualité prometteur.

11.2 La stratégie LNS

Le déroulement général de la méthode est synthétisé par l'algorithme 17. A partir d'une solution initiale, la méthode effectue itérativement une destruction - via la procédure **detruire**(·) - et une réparation de la solution par la procédure **reparer**(·), et ce, tant qu'un critère d'arrêt n'est pas vérifié. La procédure **accepter**(·,·) (ligne 4) vérifie si la nouvelle solution générée est acceptée en regard de la solution courante (le plus souvent sur un simple critère d'objectif). Dans le prolongement, le test réalisé ligne 7 vérifie si la valeur de la nouvelle solution générée est meilleure que la solution de référence (meilleure connue). Cette dernière est alors mise à jour (ligne 8).

Algorithme 17 Large Neighborhood Search (déroulement général)

Entrées:
s : solution initiale

1: $s^* \leftarrow s$
2: **tant que continuer() faire**
3: $s' \leftarrow$ **reparer**(**detruire**(s))
4: **si accepter**(s,s') **alors**
5: $s \leftarrow s'$
6: **fin si**
7: **si** $f(s) < f(s^*)$ **alors**
8: $s^* \leftarrow s$
9: **fin si**
10: $it \leftarrow it + 1$
11: **fin tant que**
12: **renvoyer** s^*

Dans la version initiale de la LNS proposée par Shaw [23], la stratégie d'acceptation d'une nouvelle solution est de type élitisme, ne conservant ainsi que les nouvelles solutions améliorantes. D'autres critères d'acceptation peuvent être utilisés. Par exemple Schrimpf et al. [22] et Ropke et Pisinger [21] utilisent un mécanisme d'acceptation emprunté à la technique de recuit simulé, donc basé sur des considérations stochastiques. Le critère d'arrêt de la LNS est, quant à lui, une limite de temps, un nombre d'itérations maximal ou une absence d'amélioration sur les k dernières itérations. La solution fournie pour initialiser la méthode peut être construite par des heuristiques constructives ou générée aléatoirement. La phase de destruction de la LNS est généralement faite par des heuristiques simples qui contiennent une part stochastique permettant à toute partie de la solution de pouvoir être détruite. Les méthodes utilisées pour la phase de réparation peuvent être de simples heuristiques pour des problèmes spécifiques. Des stratégies plus complexes, recourant par exemple à la Programmation Par Contraintes ou - et c'est notre cas – à la résolution de modèles mathématiques de type MIP sont également exploitées. Dans cette section, nous proposons deux approches LNS exploitant des voisinages basés sur chacun des modèles MIP que nous avons élaborés pour la problématique étudiée.

11.3 Approche LNS spécifiques à notre problématique

Les spécificités propres à notre problématique ont nécessité certains aménagements à la stratégie générique décrite précédemment. L'algorithme 18 décrit concrètement l'implémentation en découlant.

Algorithme 18 Large Neighborhood Search (notre version)

Entrées:

- T_{max} : temps maximal donné pour l'exécution de la LNS
- $f(s)$: valeur de la solution s

1: $s \leftarrow$ solution heuristique
2: $s^* \leftarrow s$
3: $it \leftarrow 0$
4: **tant que** $temps \leq T_{max}$ **faire**
5: **detruire**$(\cdot) \leftarrow$ **selection**(it)
6: $s \leftarrow$ **reparer**(**detruire**(s^*))
7: **si** $f(s) < f(s^*)$ **alors**
8: $s^* \leftarrow s$
9: **fin si**
10: $it \leftarrow it + 1$
11: **fin tant que**

Dans notre contexte, une solution est un ensemble de variables de décision, chacune de ces variables possédant un attribut la déclarant "fixée" ou "libre". La solution initiale est fournie par une heuristique différente pour chacune des deux versions de LNS proposée, ces heuristiques exploitant elles mêmes chacune des formalisations mathématiques présentées précédemment. Elles feront l'objet d'un développement ultérieur. Le critère d'arrêt que nous avons retenu est un temps d'exécution maximal T_{max}, en parallèle avec la gestion d'un compteur d'itérations (it). L'opérateur de destruction utilisé consiste à modifier l'attribut "fixée" ou "libre" de certaines variables de décision (ligne 6). Ces opérations sont effectuées par des algorithmes spécifiques sélectionnés en fonction du nombre d'itérations, (ligne 5 : procédure **selection**$(\cdot)$). La procédure **reparer**$(\cdot)$ (ligne 6) consiste à générer une solution alternative (réparer) en résolvant un MIP ayant pour solution initiale la solution "détruite" en instanciant les attributs "fixée" et "libre" des variables de décision impliquées dans la phase de destruction. En d'autres termes, la méthode résout un MIP dans lequel seulement les variables "libres" peuvent changer de valeur. Enfin, le critère d'acceptation que nous avons retenu (ligne 7) est l'élitisme. Comme nous l'avons mentionné précédemment, deux versions de la LNS ont été implémentées, la première basée sur le modèle multiflots LNS(MMF) présenté dans le chapitre 3, et la seconde exploitant la modélisation indexée dans le temps LNS(MIT) développée dans le chapitre 4. Dans ce qui suit, nous présentons en détail les composants spécifiques à chacune de ces versions.

11.3.1 Approche multiflots : LNS(MMF)

Dans cette version, une solution de la LNS est un ensemble d'OT et de routages actifs qui sont solution du MIP $[P_{MMF}]$. Un OT ou un routage est dit actif si la valeur de sa variable de décision lui correspondant est non nulle.

$$[P_{MMF}] \qquad \min \sum_{\omega \in \Omega} w(\omega) y_\omega$$

SC

$$\forall p \in P, \qquad \sum_{\sigma \in R^p} x_p^\sigma = q(p) \qquad (3.1)$$

$$\forall \omega \in \Omega, \qquad \sum_{p \in P} \sum_{\sigma \in R^p} a_\omega^\sigma x_p^\sigma \leq cp(\omega) y_\omega \qquad (3.2)$$

$$\forall h \in H, \forall (\theta, \theta') \in \Theta_h^2 / \theta \leq \theta', \qquad \sum_{p \in P_h} \sum_{\sigma \in R_h(\theta, \theta')} x_p^\sigma \leq (\theta' - \theta + 1)\eta_h \qquad (3.7)$$

$$\forall c \in C, \forall \theta \in \Theta_c, \qquad \sum_{p \in P_c^-} \sum_{\sigma \in R_c(\theta)} x_p^\sigma \geq \Delta_c - (\theta_c^+ - \theta + 1)\eta_c \qquad (3.12)$$

$$\forall \omega \in \Omega, \qquad y_\omega \in \mathbb{N} \qquad (3.4)$$

$$\forall p \in P, \forall \sigma \in R^p, \qquad x_p^\sigma \geq 0 \qquad (3.5)$$

Pour cette version, la solution fournie à la LNS lors de son initialisation est donnée par l'heuristique d'amélioration présentée dans le chapitre 8, qui est elle même initialisée par l'heuristique sérielle présentée dans le chapitre 7. En conséquence, la solution initiale n'est pas toujours réalisable. Si c'est la cas, elle est solution du modèle $[P_{NRMMF}]$. Notons que ce dernier modèle est similaire à $[P_{MMF}]$ par intégration des variables d'écart permettant de pallier aux cas d'irréalisabilité.

$$[P_{NRMMF}] \qquad \min \sum_{\omega \in \Omega} w(\omega) y_\omega + \sum_{p \in P} M s_p + \sum_{c \in C} \sum_{\theta \in \Theta_c} M s_c^\theta$$

SC

$$\forall p \in P \qquad \sum_{\sigma \in R^p} x_p^\sigma + s_p = q(p) \qquad (11.1)$$

$$\forall \omega \in \Omega, \qquad \sum_{p \in P} \sum_{\sigma \in R^p} a_\omega^\sigma x_p^\sigma \leq cp(\omega) y_\omega \qquad (3.2)$$

$$\forall h \in H, \forall (\theta, \theta') \in \Theta_h^2 / \theta \leq \theta', \qquad \sum_{p \in P_h} \sum_{\sigma \in R_h(\theta, \theta')} x_p^\sigma \leq (\theta' - \theta + 1)\eta_h \qquad (3.7)$$

$$\forall c \in C, \forall \theta \in \Theta_c, \qquad \sum_{p \in P_c^-} \sum_{\sigma \in R_c(\theta)} x_p^\sigma + s_c^\theta \geq \Delta_c - (\theta_c^+ - \theta + 1)\eta_c \qquad (11.2)$$

$$\forall \omega \in \Omega, \qquad y_\omega \in \mathbb{N} \qquad (3.4)$$

$$\forall p \in P, \forall \sigma \in R^p, \qquad x_p^\sigma \geq 0 \qquad (3.5)$$

$$\forall p \in P \qquad s_p \geq 0 \qquad (11.3)$$

$$\forall c \in C, \forall \theta \in \Theta_c \qquad s_c^\theta \geq 0 \qquad (11.4)$$

Si la solution initiale de la LNS n'est pas réalisable, la méthode utilise le MIP $[P_{NRMMF}]$. Dès lors qu'à une itération donnée, la somme des variables d'écart $s_p(\forall p \in P)$ et $s_c^\theta(\forall c \in C, \forall \theta \in \Theta_c)$ est nulle, alors la méthode bascule sur le modèle $[P_{MMF}]$ jusqu'à la fin de son exécution. En effet, une solution de $[P_{NRMMF}]$ pour laquelle la somme des variables d'écart est nulle est clairement une solution de $[P_{MMF}]$, et leur valorisation est la même.

Nous présentons dans les deux sous-sections suivantes les opérateurs de destruction et de réparation utilisés par cette version.

Phase de destruction partielle de la solution courante

L'objectif de cette phase est de détruire partiellement la solution courante de manière à pouvoir en modifier le routage des produits. Dans notre cas, détruire la solution courante revient simplement à changer l'attribut "fixée" ou "libre" des variables de décision (sélection d'OT et des routages des produits). Cette phase sélectionne donc à chaque itération les variables qui seront "libres" et celles qui seront "fixées". Ce choix se base sur la sélection d'un produit via la procédure **selection**$(\cdot)$ (rappelons que ce choix dépend de l'itération). Ensuite, un MIP est construit, dans lequel certaines variables sont "fixées" et conservent les valeurs qu'elles avaient dans la solution précédente, alors que d'autres sont "libres". La partie de la solution "détruite" est ainsi celle définie par les variables dont le statut est "libre".

Algorithme 19 Procédure **selection**(it)

Entrées:

- s : solution courante

1: $F \leftarrow \varnothing$
2: $p \leftarrow$ **recup**(s)
3: $F \leftarrow \{p\} \oplus$ **ensemble**(p)
4: **renvoyer construire**(F,s)

La procédure **recup**$(\cdot)$ sélectionne un produit en fonction de la solution courante (ligne 2). Une fois cette sélection opérée, la procédure **ensemble**$(\cdot)$ cherche un ensemble de produits interagissant avec le produit sélectionné (ligne 3). Enfin, la procédure **construire**$(\cdot,\cdot)$ construit un MIP intégrant un ensemble de contraintes dépendant des attributs "fixée" ou "libre" donnés aux variables de décision.

Phase de sélection d'un produit

Si la solution courante n'est pas réalisable, le produit sélectionné est un des produits incomplètement routé. Sinon, la procédure choisit aléatoirement un des trois critères de sélection d'un produit que nous avons implémentés, et ce en lien avec une fonction de scoring.

Le premier critère est basé sur la fréquence de sélection des segments de transport. Lorsqu'un produit est sélectionné, tous les produits ayant dans leur routage primaire un des segments transport du routage primaire du produit choisi "marquent" un point. Un produit appartenant au sous-ensemble de produits réalisant le plus petit score est alors tiré aléatoirement. La figure 11.1 illustre l'idée sous-jacente à cette fonction de scoring.

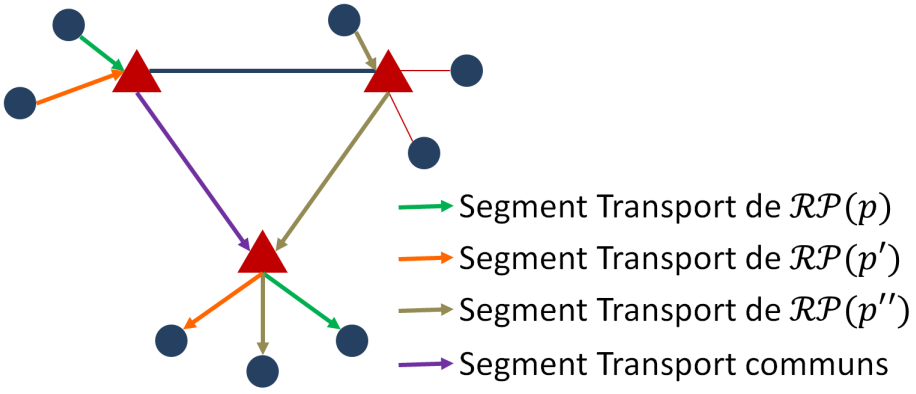


FIGURE 11.1 – Exemple pour la sélection d'un produit
Si le produit p est sélectionné, alors le produit p' marque un point car son routage primaire possède un segment transport commun au routage primaire du produit p, tandis que le produit p'' ne marque aucun point.

Le deuxième critère de sélection est basé sur le taux de remplissage des véhicules : pour chaque segment transport, nous calculons le taux de remplissage moyen des véhicules selon la formule 11.5.

$$\forall (i,j) \in U, \frac{\sum_{\theta \in \Theta} \frac{\sum_{p \in P^{\theta}_{i,j}} y^{\theta}_{i,j}(p)}{Q_{i,j} z^{\theta}_{i,j}}}{\tau} \tag{11.5}$$

Nous calculons aussi le taux de remplissage moyen par produit. Un produit parmi ceux ayant le plus faible taux de remplissage moyen est alors tiré aléatoirement.

Enfin, le troisième et dernier critère de sélection réside simplement dans le tirage aléatoire d'un produit.

Phase de récupération de l'ensemble de produits en interaction avec le produit p

Une fois la sélection d'un produit p opérée, nous recherchons, par le biais de la procédure **ensemble**$(\cdot)$, l'ensemble F des produits interagissant avec p dans le réseau via un segment transport commun à leurs routages primaires. Cette procédure stocke dans F tous les produits qui ont au moins un segment transport de leur routage primaire commun à ceux du routage primaire du produit p. Cet ensemble peut formellement être défini par relation 11.6.

$$F \leftarrow \{p\} \oplus \{p' \in P / \exists (i,j) \in RP(p') \wedge (i,j) \in RP(p)\} \tag{11.6}$$

La figure 11.2 illustre cet opérateur.

Phase de construction du MIP

La procédure **construire**$(\cdot,\cdot)$ de l'algorithme 19 construit un MIP intégrant un ensemble de contraintes dépendant des attributs "fixée" ou "libre" donnés aux variables de décision. Notons $(\bar{x}, \bar{y})$ la solution courante, $\bar{\Omega}$ le sous-ensemble d'OT actifs ($\bar{y}_{\omega} > 0$),

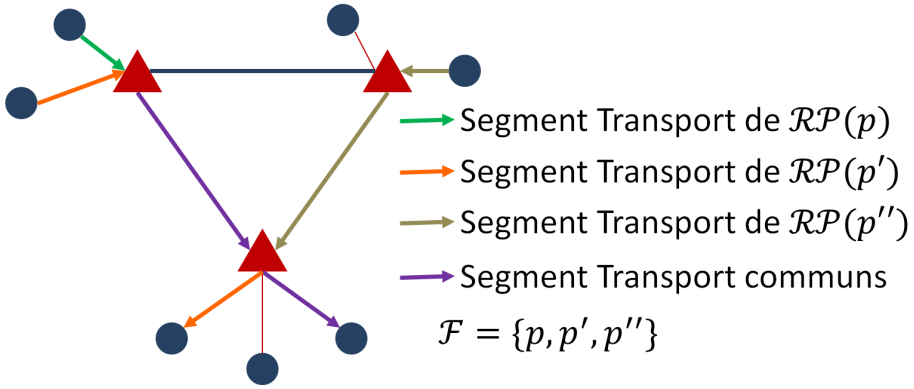


FIGURE 11.2 – Exemple pour la récupération de F
Pour le produit sélectionné p, l'ensemble F est constitué de p, p' et p''.

ainsi que $\bar{R}$ le sous-ensemble des routages actifs ($\exists p \in P, x_p^\sigma > 0$). Le MIP que nous utilisons à ce stade est une restriction du modèle $[P_{MMF}]$ à l'ensemble $\bar{\Omega}$, et noté $[P_{MMF}(\bar{\Omega})]$.

La procédure **construire(·,·)** de l'algorithme 19 construit un MIP auquel il est adjoint un ensemble de contraintes dépendant des attributs "fixée" ou "libre" donnés aux variables de décision. Le MIP que nous utilisons est une restriction du modèle $[P_{MMF}]$ à l'ensemble des OT $\bar{\Omega}$, il est noté $[P_{MMF}(\bar{\Omega})]$. Par "translation", nous avons aussi la restriction du modèle $[P^{NR}_{MMF}]$. L'ensemble $\bar{\Omega}$ est l'ensemble des OT actifs, c'est-à-dire $\forall \omega \in \bar{\Omega}, y_\omega > 0$. De même, l'ensemble des routages actifs $\bar{R}$ est constitué des routages σ tels que $\exists p \in P, x_p^\sigma > 0$.

$$[P_{MMF}(\bar{\Omega})] \qquad \min \sum_{\omega \in \bar{\Omega}} w(\omega) y_\omega$$

$$SC$$

$$\forall p \in P, \qquad \sum_{\sigma \in \bar{R}^p} x_p^\sigma = q(p) \qquad (11.7)$$

$$\forall \omega \in \bar{\Omega}, \qquad \sum_{p \in P} \sum_{\sigma \in \bar{R}^p} a_\omega^\sigma x_p^\sigma \leq cp(\omega) y_\omega \qquad (11.8)$$

$$\forall h \in H, \forall (\theta, \theta') \in \Theta_h^2 / \theta \leq \theta', \qquad \sum_{p \in P_h} \sum_{\sigma \in \bar{R}_h(\theta, \theta')} x_p^\sigma \leq (\theta' - \theta + 1)\eta_h \qquad (11.9)$$

$$\forall c \in C, \forall \theta \in \Theta_c, \qquad \sum_{p \in P_c^-} \sum_{\sigma \in \bar{R}_c(\theta)} x_p^\sigma \geq \Delta_c - (\theta_c^+ - \theta + 1)\eta_c \qquad (11.10)$$

$$\forall \omega \in \bar{\Omega}, \qquad y_\omega \in \mathbb{N} \qquad (11.11)$$

$$\forall p \in P, \forall \sigma \in \bar{R}^p, \qquad x_p^\sigma \geq 0 \qquad (11.12)$$

Si la solution courante est non réalisable, nous avons le modèle , nous avons aussi la restriction du modèle $[P_{NRMMF}]$ qui est appelée $[P_{NRMMF}(\bar{\Omega})]$, mais nous l'explicitons pas, car elle est très similaire à $[P_{MMF}(\bar{\Omega})]$

Concrètement, cette phase de construction du MIP récupère l'ensemble des OT actifs $\bar{\Omega}$ de la solution courante, auquel on adjoint l'ensemble Ω_p des OT compatibles avec le

produit p sélectionné (ensemble des OT ayant pour segment transport un des segment transport du routage primaire du produit p). Formellement, Ω_p est défini par la relation 11.13.

$$\Omega_p = \{\omega \in \Omega / \nu(\omega) \in RP(p)\} \tag{11.13}$$

On génère ainsi tous les nouveaux routages incluant un de ces nouveaux OT pour tous les produits de F. Ces routages sont ajoutés à l'ensemble des routages actifs de la solution courante. Les contraintes associées à ces nouveaux OT et nouveaux routages sont ajoutés au MIP.

Ensuite, la solution partielle relative aux produits non inclus dans F est figée. Pour cela, nous ajoutons des contraintes sur les variables y_ω correspondant aux OT non inclus dans un des routages des produits de F, en bloquant leur valeur à celle obtenue dans la solution courante $\bar{y}$.

$$\forall p \in F, \forall \omega \in \bar{\Omega} - \Omega_p, y_\omega = \bar{y}_\omega \tag{11.14}$$

L'attribut de ces variables de décision est donc "fixée".

Pour les routages, on distingue deux cas :

– Cas 1 : La solution courante est non réalisable
  Aucune nouvelle contrainte n'est générée, ce qui peut permettre de router quelques unités de produits supplémentaires. Toutes les variables de routage ont donc récupéré l'attribut "libre".
– Cas 2 : La solution courante est réalisable
  Les routages n'incluant pas un OT de p sont figés, les valeurs des variables associées étant bloquées à leur valeur dans la solution courante.

$$\forall p \in P, \forall \sigma \in R^p(\bar{\Omega} - \Omega_p), x_p^\sigma = \bar{x}_p^\sigma \tag{11.15}$$

Phase de réparation de la solution

A l'issue de ces différentes phases, il ne reste plus qu'à reconstruire une solution alternative. La procédure **reparer**(**detruire**$(\cdot)$) de l'algorithme 18 résout le MIP renvoyé par la procédure **detruire**$(\cdot)$ et retourne la solution. Dans le cas où la solution est réalisable, la méthode résout en fait une restriction du $[P_{MMF}(\bar{\Omega})]$, notée $[R(P_{MMF}(\bar{\Omega}))]$, obtenue par ajout des contraintes figeant certaines variables de décision (11.14 et 11.15).

$[R(P_{MMF}(\bar{\Omega}))]$ $\quad \min \sum_{\omega \in \bar{\Omega}} w(\omega) y_\omega$

SC

$\forall p \in P,$ $\quad \sum_{\sigma \in \bar{R}^p} x_p^\sigma = q(p)$ (11.7)

$\forall \omega \in \bar{\Omega},$ $\quad \sum_{p \in P} \sum_{\sigma \in \bar{R}^p} a_\omega^\sigma x_p^\sigma \leq cp(\omega) y_\omega$ (11.8)

$$\forall h \in H, \forall(\theta,\theta') \in \Theta_h^2/\theta \leq \theta', \qquad \sum_{p\in P_h}\sum_{\sigma\in\bar{R}_h(\theta,\theta')} x_p^\sigma \leq (\theta'-\theta+1)\eta_h \qquad (11.9)$$

$$\forall c \in C, \forall\theta \in \Theta_c, \qquad \sum_{p\in P_c^-}\sum_{\sigma\in\bar{R}_c(\theta)} x_p^\sigma \geq \Delta_c - (\theta_c^+-\theta+1)\eta_c \qquad (11.10)$$

$$\forall\omega \in \bar{\Omega}, \qquad y_\omega \in \mathbb{N} \qquad (11.11)$$

$$\forall p \in P, \forall\sigma \in \bar{R}^p, \qquad x_p^\sigma \geq 0 \qquad (11.12)$$

$$\forall p \in F, \forall\omega \in \bar{\Omega} - \Omega_p, \qquad y_\omega = y_\omega^* \qquad (11.14)$$

$$\forall p \in P, \forall\sigma \in R^p(\bar{\Omega} - \Omega_p), \qquad x_p^\sigma = x_p^{*\sigma} \qquad (11.15)$$

La solution de référence (meilleure solution trouvée) est mise à jour si la solution obtenue par réparation est de meilleure valorisation.

11.3.2 Approches indexée dans le temps : LNS(MIT)

Cette deuxième version est basée sur la modélisation indexée dans le temps présentée dans le chapitre 4. Une solution de la LNS est un ensemble de variables de flux (x et y) et de variables associées à la monopolisation de véhicules (z) qui sont solution du MIP $[P_{MITAC}]$.

$$[P_{MITAC}] \qquad \min \sum_{s\in S}\sum_{s'\in S}\sum_{\theta\in\Theta} w_{s,s'} z_{s,s'}^\theta$$

SC

$$\forall(s,s') \in A, \forall\theta \in \Theta \qquad \sum_{p\in P_{s,s'}} y_{s,s'}^\theta(p) \leq cp_{s,s'} z_{s,s'}^\theta \qquad (4.1)$$

$$\forall s \in S, \forall\theta \in \Theta \qquad \sum_{p\in P_s} x_{s,\theta}^T(p) \leq \eta_s \qquad (4.2)$$

$$\forall p \in P_{t2}, \forall\theta \in \Theta \quad x_{h_o(p),\theta-1}^A(p) + y_{s(p),h_o(p)}^{\theta-l_{s(p),h_o(p)}}(p) = x_{h_o(p),t}^A(p) + x_{h_o(p),\theta}^T(p) \qquad (4.3)$$

$$\forall p \in P_{t2}, \forall\theta \in \Theta \quad x_{h_d(p),\theta-1}^A(p) + y_{h_o(p),h_d(p)}^{\theta-l_{h_o(p),h_d(p)}}(p) = x_{h_d(p),\theta}^A(p) + x_{h_d(p),\theta}^T(p) \qquad (4.4)$$

$$\forall p \in P_{t1}, \forall\theta \in \Theta \quad x_{h_d(p),\theta-1}^A(p) + y_{s(p),h_d(p)}^{\theta-l_{s(p),h_d(p)}}(p) = x_{h_d(p),\theta}^A(p) + x_{h_d(p),\theta}^T(p) \qquad (4.5)$$

$$\forall p \in P, \forall\theta \in \Theta \quad x_{d(p),\theta-1}^A(p) + y_{h_d(p),d(p)}^{\theta-l_{h_d(p),d(p)}}(p) = x_{d(p),\theta}^A(p) + x_{d(p),\theta}^T(p) \qquad (4.6)$$

$$\forall p \in P \qquad x_{s(p),es(p)-1}^D(p) = q(p) \qquad (4.7)$$

$$\forall p \in P, \forall\theta \in \Theta \qquad x_{s(p),\theta-1}^D(p) = x_{s(p),\theta}^D(p) + y_{s(p),h_o(p)}^\theta(p) \qquad (4.8)$$

$$\forall p \in P_{t2}, \forall\theta \in \Theta \quad x_{h_o(p),\theta-1}^D(p) + x_{h_o(p),\theta-1}^T(p) = x_{h_o(p),\theta}^D(p) + y_{h_o(p),h_d(p)}^\theta(p) \qquad (4.9)$$

$$\forall p \in P_{t2}, \forall\theta \in \Theta \quad x_{h_d(p),\theta-1}^D(p) + x_{h_d(p),\theta-1}^T(p) = x_{h_d(p),\theta}^D(p) + y_{h_d(p),d(p)}^\theta(p) \qquad (4.10)$$

$$\forall p \in P_{t1}, \forall\theta \in \Theta \quad x_{h_d(p),\theta-1}^D(p) + x_{h_d(p),\theta-1}^T(p) = x_{h_d(p),t}^D(p) + y_{h_d(p),d(p)}^\theta(p) \qquad (4.11)$$

$$\forall p \in P, \forall\theta \in \Theta \qquad x_{d(p),\theta-1}^D(p) + x_{d(p),\theta-1}^T(p) = x_{d(p),\theta}^D(p) \qquad (4.12)$$

$$\forall p \in P \qquad x_{d(p),lc(p)}^D = q(p) \qquad (4.13)$$

$$\forall(s,s') \in A, \forall[\theta^-,\theta^+] \in \Theta_{s,s'} \qquad \left\lceil \frac{\sum_{p\in P_{\theta^-,\theta^+}} q(p)}{cp_{s,s'}} \right\rceil \leq \sum_{\theta=\theta^-}^{\theta^+} z_{s,s'}^\theta \qquad (5.3)$$

$$\forall (s,s') \in A, \forall \theta \in \Theta, \forall p \in P^{\theta}_{s,s'} \qquad y^{\theta}_{s,s'}(p) \leq q(p) z^{\theta}_{s,s'} \quad (5.5)$$

$$\forall (s,s') \in A, \forall \theta \in \Theta \qquad z^{\theta}_{s,s'} \in \mathbb{N} \quad (4.14)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{A}_{s,\theta}(p) \geq 0 \quad (4.15)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{D}_{s,\theta}(p) \geq 0 \quad (4.16)$$

$$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta \qquad x^{T}_{s,\theta}(p) \geq 0 \quad (4.17)$$

$$\forall (s,s') \in A, \forall p \in P_{s,s'}, \forall \theta \in \Theta \qquad y^{\theta}_{s,s'}(p) \geq 0 \quad (4.18)$$

La solution fournie à la LNS lors de son initialisation est donnée par l'heuristique d'arrondis successifs présentée dans le chapitre 9. Contrairement à la version précédente, une solution de la LNS est toujours réalisable. Nous présentons ci-après les opérateurs de destruction et de réparation utilisés dans cette version, certains étant par ailleurs identiques.

Phase de destruction partielle de la solution courante

A l'instar de la version précédente, l'objectif de cette phase est de détruire partiellement la solution courante de manière à pouvoir en modifier le routage des produits. Dans ce cas également, détruire la solution revient à changer l'attribut "fixée" ou "libre" associé à un sous-ensemble de variables de décision, en lien avec la sélection d'un produit. Une fois cette sélection opérée, un MIP est construit dans lequel certaines variables sont "fixées" et conservent les valeurs qu'elles avaient dans la solution courante, alors que d'autres sont "libres" et peuvent donc être ré-instanciées lors de la phase de réparation. La partie de la solution "détruite" est celle définie par les variables de statut "libres".

Phase de sélection d'un produit

Cette phase est quasiment similaire à celle de la version précédente (paragraphe 11.3.1). Néanmoins, seul le cas où la solution est réalisable est à considérer. Une fonction de scoring, en lien avec trois critères de sélection d'un produit p est ici aussi exploitée. Rappelons que le premier critère est basée sur la fréquence de sélection des segments de transport, le deuxième sur le taux de remplissage des véhicules, et le troisième et dernier réside dans le tirage aléatoire d'un produit.

Phase de récupération de l'ensemble de produits en interaction avec p

La première partie de cette phase est identique à la phase de récupération de l'ensemble de produits en interaction avec p de la version précédente de la LNS (paragraphe 11.3.1). Néanmoins, cette version exploite deux opérateurs différents. Le premier, identique à celui utilisé dans la version multi-flots, récupère l'ensemble F des produits interagissant avec p dans le réseau via un segment transport commun à leurs routages primaires. Le deuxième opérateur peut être apparenté à un opérateur de diversification du fait qu'il sélectionne plus de produits. Il stocke dans F tous les produits qui ont en commun, dans leur routage primaire avec le produit sélectionné p, soit leur CT origine, leur CT destination ou un de leur hubs. La relation 11.16 défini formellement cet ensemble F pour le produit p.

$$F \leftarrow \{p\} \oplus \{p' \in P / s(p') = s(p) \vee d(p') = d(p) \vee h \in RP(p') \wedge h \in RP(p), h \in H\} \quad (11.16)$$

La figure 11.3 illustre la construction de l'ensemble F via cet opérateur pour un produit sélectionné p.

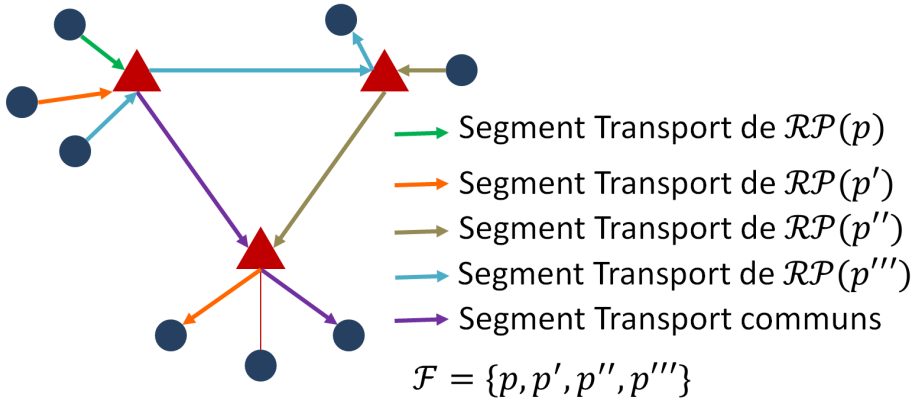


FIGURE 11.3 – Exemple pour l'opérateur de diversification
Les produits p', p'' et p''' sont inclus dans F, car ils sont au moins un des sites logistiques du routage primaire du produit p dans leur routage primaire.

Ce second opérateur a été rajouté à cette version de la LNS car la modélisation indexée dans le temps, plus compacte, permet l'exploitation de voisinages plus larges au travers de la résolution de MIP par solveur.

Phase de construction du MIP

A l'instar de la première version de la LNS, la procédure **construire**$(\cdot,\cdot)$ de l'algorithme 19 instancie un MIP intégrant un ensemble de contraintes dépendant des attributs "fixée" ou "libre" donnés aux variables de décision. La modélisation indexée dans le temps possédant une structure plus favorable en termes de résolution solveur (cf. chapitre 13 concernant les expérimentations numériques), cette procédure de la méthode fige dans ce cas moins de variables. Concrètement, elle sélectionne alors toutes les variables de flux des produits n'appartenant pas à F et elle bloque leur valeur à celle obtenue dans la solution courante (notée $(\bar{x}, \bar{y}\bar{z})$).

$$\forall s \in S, \forall p \in \{P_s \ominus F\}, \forall \theta \in \Theta_s, x^A_{s,\theta}(p) = \bar{x}^A_{s,\theta}(p) \tag{11.17}$$
$$\forall s \in S, \forall p \in \{P_s \ominus F\}, \forall \theta \in \Theta_s, x^D_{s,\theta}(p) = \bar{x}^D_{s,\theta}(p) \tag{11.18}$$
$$\forall s \in S, \forall p \in \{P_s \ominus F\}, \forall \theta \in \Theta_s, x^T_{s,\theta}(p) = \bar{x}^T_{s,\theta}(p) \tag{11.19}$$
$$\forall (s,s') \in A, \forall p \in \{P_{s,s'} \ominus F\}, \forall \theta \in \Theta_{s,s'}, y^\theta_{s,s'}(p) = \bar{y}^\theta_{s,s'}(p) \tag{11.20}$$

De même, la procédure ajoute des contraintes bloquant le nombre de véhicules sur les segments transport n'étant dans aucun des routages primaires des produits de F.

$$\forall (s,s') \notin RP(p/p \in F), \theta \in \Theta_{s,s'}, z^\theta_{s,s'} = \bar{z}^\theta_{s,s'} \tag{11.21}$$

L'ensemble des variables de décision contraintes à conserver la valeur qu'elles avaient dans la solution courante récupère l'attribut "fixée".

11.3.3 Réparation de la solution

A l'issue de la procédure **construire**(·,·), la procédure **reparer**(**detruire**(·)) de l'algorithme 18 résout le MIP renvoyé par la procédure **detruire**(·) et retourne la solution. Cette version, utilisant la modélisation indexée dans le temps, résout le MIP $[P_{MITAC}]$ auquel sont rajoutées les contraintes fixant un certain nombre de variables de décision (11.17, 11.18, 11.19, 11.20 et 11.21), cette restriction est notée $[R([P_{MITAC}])]$.

$[R([P_{MITAC}])]$ $\min \sum_{s\in S}\sum_{s'\in S}\sum_{\theta\in\Theta} w_{s,s'} z^{\theta}_{s,s'}$

SC

$\forall (s,s') \in A, \forall \theta \in \Theta$ $\sum_{p\in P_{s,s'}} y^{\theta}_{s,s'}(p) \leq cp_{s,s'} z^{\theta}_{s,s'}$ (4.1)

$\forall s \in S, \forall \theta \in \Theta$ $\sum_{p\in P_s} x^{T}_{s,\theta}(p) \leq \eta_s$ (4.2)

$\forall p \in P_{t2}, \forall \theta \in \Theta$ $x^{A}_{h_o(p),\theta-1}(p) + y^{\theta-l_{s(p),h_o(p)}}_{s(p),h_o(p)}(p) = x^{A}_{h_o(p),t}(p) + x^{T}_{h_o(p),\theta}(p)$ (4.3)

$\forall p \in P_{t2}, \forall \theta \in \Theta$ $x^{A}_{h_d(p),\theta-1}(p) + y^{\theta-l_{h_o(p),h_d(p)}}_{h_o(p),h_d(p)}(p) = x^{A}_{h_d(p),\theta}(p) + x^{T}_{h_d(p),\theta}(p)$ (4.4)

$\forall p \in P_{t1}, \forall \theta \in \Theta$ $x^{A}_{h_d(p),\theta-1}(p) + y^{\theta-l_{s(p),h_d(p)}}_{s(p),h_d(p)}(p) = x^{A}_{h_d(p),\theta}(p) + x^{T}_{h_d(p),\theta}(p)$ (4.5)

$\forall p \in P, \forall \theta \in \Theta$ $x^{A}_{d(p),\theta-1}(p) + y^{\theta-l_{h_d(p),d(p)}}_{h_d(p),d(p)}(p) = x^{A}_{d(p),\theta}(p) + x^{T}_{d(p),\theta}(p)$ (4.6)

$\forall p \in P$ $x^{D}_{s(p),es(p)-1}(p) = q(p)$ (4.7)

$\forall p \in P, \forall \theta \in \Theta$ $x^{D}_{s(p),\theta-1}(p) = x^{D}_{s(p),\theta}(p) + y^{\theta}_{s(p),h_o(p)}(p)$ (4.8)

$\forall p \in P_{t2}, \forall \theta \in \Theta$ $x^{D}_{h_o(p),\theta-1}(p) + x^{T}_{h_o(p),\theta-1}(p) = x^{D}_{h_o(p),\theta}(p) + y^{\theta}_{h_o(p),h_d(p)}(p)$ (4.9)

$\forall p \in P_{t2}, \forall \theta \in \Theta$ $x^{D}_{h_d(p),\theta-1}(p) + x^{T}_{h_d(p),\theta-1}(p) = x^{D}_{h_d(p),\theta}(p) + y^{\theta}_{h_d(p),d(p)}(p)$ (4.10)

$\forall p \in P_{t1}, \forall \theta \in \Theta$ $x^{D}_{h_d(p),\theta-1}(p) + x^{T}_{h_d(p),\theta-1}(p) = x^{D}_{h_d(p),t}(p) + y^{\theta}_{h_d(p),d(p)}(p)$ (4.11)

$\forall p \in P, \forall \theta \in \Theta$ $x^{D}_{d(p),\theta-1}(p) + x^{T}_{d(p),\theta-1}(p) = x^{D}_{d(p),\theta}(p)$ (4.12)

$\forall p \in P$ $x^{D}_{d(p),lc(p)} = q(p)$ (4.13)

$\forall (s,s') \in A, \forall [\theta^-,\theta^+] \in \Theta_{s,s'}$ $\left\lceil \frac{\sum_{p\in P_{\theta^-,\theta^+}} q(p)}{cp_{s,s'}} \right\rceil \leq \sum_{\theta=\theta^-}^{\theta^+} z^{\theta}_{s,s'}$ (5.3)

$\forall (s,s') \in A, \forall \theta \in \Theta, \forall p \in P^{\theta}_{s,s'}$ $y^{\theta}_{s,s'}(p) \leq q(p) z^{\theta}_{s,s'}$ (5.5)

$\forall (s,s') \in A, \forall \theta \in \Theta$ $z^{\theta}_{s,s'} \in \mathbb{N}$ (4.14)

$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta$ $x^{A}_{s,\theta}(p) \geq 0$ (4.15)

$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta$ $x^{D}_{s,\theta}(p) \geq 0$ (4.16)

$\forall s \in S, \forall p \in P_s, \forall \theta \in \Theta$ $x^{T}_{s,\theta}(p) \geq 0$ (4.17)

$\forall (s,s') \in A, \forall p \in P_{s,s'}, \forall \theta \in \Theta$ $y^{\theta}_{s,s'}(p) \geq 0$ (4.18)

$\forall s \in S, \forall p \in \{P_s \ominus F\}, \forall \theta \in \Theta_s,$ $x^{A}_{s,\theta}(p) = \bar{x}^{A}_{s,\theta}(p)$ (11.17)

$\forall s \in S, \forall p \in \{P_s \ominus F\}, \forall \theta \in \Theta_s,$ $x^{D}_{s,\theta}(p) = \bar{x}^{D}_{s,\theta}(p)$ (11.18)

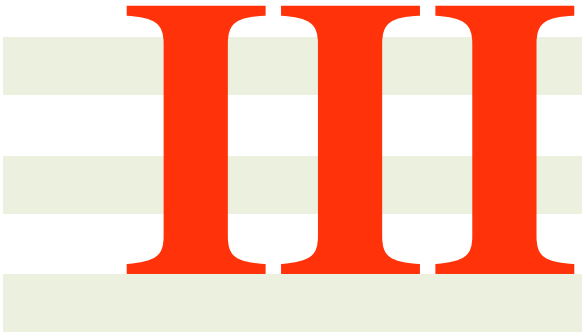
$\forall s \in S, \forall p \in \{P_s \ominus F\}, \forall \theta \in \Theta_s,$ $x^{T}_{s,\theta}(p) = \bar{x}^{T}_{s,\theta}(p)$ (11.19)

$$\forall (s,s') \in A, \forall p \in \{P_{s,s'} \ominus F\}, \forall \theta \in \Theta_{s,s'}, \qquad y^{\theta}_{s,s'}(p) = \bar{y}^{\theta}_{s,s'}(p) \quad (11.20)$$

$$\forall (s,s') \notin RP(p/p \in F), \theta \in \Theta_{s,s'}, \qquad z^{\theta}_{s,s'} = \bar{z}^{\theta}_{s,s'} \quad (11.21)$$

Dans cette version également, la solution de référence (meilleure solution trouvée) est mise à jour si la solution obtenue par réparation est de meilleure valorisation.

A noter que pour les deux version, et dans l'objectif d'accélérer la résolution, la solution courante est fournie en phase de presolve au solveur. De ce fait, la solution trouvée est au pire la solution courante.



Expérimentations et résultats

12

Générateur d'instances

Sommaire

12.1 Introduction

Comme nous l'avons mentionné lors de la formulation de notre problématique et de sa confrontation à l'état de l'art en termes de productions scientifiques connexes, une des spécifications fonctionnelles majeures de nos travaux réside dans la prise en compte de capacités de traitement sur les différents sites logistiques du réseau. La conséquence immédiate est nous ne disposions d'aucune instance nous permettant de tester nos approches. Nous nous sommes donc attelés au développement d'un générateur d'instances dédié à notre problématique.

12.2 Description du générateur

Une instance de notre problème est clairement caractérisée par un ensemble de sites logistiques (noeuds), centres de traitement (CT) et hubs, chacun d'eux possédant comme attributs une localisation dans le plan et une capacité de traitement, ainsi qu'un ensemble de liaisons (arcs) reliant certaines paires de ces sites. Le graphe résultant doit être logiquement connexe. Un flux non nul devant être routée entre une paire de CT de ce réseau spécifie un produit, chacun d'eux possédant une quantité, une date de disponibilité, une date échue et un routage primaire. Enfin, des types de véhicules sont créés pour assurer l'acheminement de ces produits.

La génération d'une instance se fait en cinq étapes :

– définition de la taille de l'instance,

– construction géographique du réseau,
– création des types de véhicules,
– création des produits,
– instanciation des capacités de traitement.

La première étape décide le nombre de CT et de hubs constituant réseau et la "taille" de la zone géographique sur laquelle ils sont positionnés. La deuxième définie la localisation géographique de ces noeuds et les arcs les reliant. La troisième étape génère les différents types de véhicules et leurs caractéristiques (vitesse et capacité). La quatrième définie l'ensemble des produits. Enfin, la cinquième étape affecte les capacités de traitement aux noeuds en fonction des flux qui y transitent.

12.2.1 Paramètres de l'instance

Un ensemble de paramètres est nécessaire pour la création d'une instance :
– $taille$: taille de la "carte" : par convention, il s'agit d'un carré 100×100,
– $nbCT$: nombre de CT (pouvant prendre les valeurs 30, 50, 100 dans nos expérimentations),
– nbH : nombre de hubs (pouvant prendre les valeurs 4, 6, 12 dans nos expérimentations),
– nbV : nombre de types de véhicules (fixé à 2 dans nos expérimentations),
– $horizon$: nombre de pas de temps de l'horizon temporel (fixé à 48 dans nos expérimentations),
– $tauxP$: taux de couples de CT entre lesquels transite un produit (25, 50, 75 dans nos expérimentations).

12.2.2 Localisation géographique des sites logistiques

Placement des CT

La procédure **placementCT**() place les $nbCT$ CT aléatoirement sur la carte en respectant un critère d'éloignement proposé pas Gabrel et Minoux (1997) [11]. Un candidat pour être le k^{me} noeud du graphe est accepté si la distance entre celui-ci et le point le plus proche des $k-1$ points déjà générés est supérieure au seuil :

$$\frac{taille}{5\sqrt{nbCT + nbH}} \tag{12.1}$$

La procédure **eloignement**(cx,cy) est dédiée à ce contrôle. De plus, la variable $typeCT$, valant 1, 2 ou 3, spécifie la taille d'un CT (respectivement, petit, moyen ou grand). Cette taille conditionne les quantités de produits émises et reçus par ce site. Les paramètres α_1, α_2 et α_3 donnent les pourcentages de petits, moyens et grands CT que l'instance regroupe ($\alpha_1 = 30\%$, $\alpha_2 = 40\%$ et $\alpha_3 = 30\%$ dans nos expérimentations). La procédure **ajoutCT**(num,cx,cy,$typeCT$) ajoute le CT numéroté num de taille $typeCT$ sur le point de coordonnées (cx,cy). Les distances utilisées étant euclidiennes, la procédure **distEuclidienne**(cx,cy,cx_0,cy_0) renvoie la distance entre les deux points passés en paramètre.

Algorithme 20 Procédure **placementCT**()

```
num ← 0
tant que num < nbCT faire
    cx ← U(0, taille)        ▷ U(a, b) représente la loi uniforme sur l'intervalle [a, b]
    cy ← U(0, taille)
    si eloignement(cx,cy) alors
        alea ← U(0, 1)
        si alea < α1 alors
            typeCT ← 1
        sinon
            si alea < α1 + α2 alors
                typeCT ← 2
            sinon
                typeCT ← 3
            fin si
        fin si
        ajoutCT(num,cx,cy,typeCT)
        num ← num + 1
    fin si
fin tant que
```

Algorithme 21 Procédure **eloignement**(cx,cy)

```
pour tout s ∈ S faire
    dist ← distEuclidienne(cx,cy,s.cx,s.cy)
    si dist < taille / (5√(nbCT+nbH)) alors renvoyer Faux
    fin si
fin pourrenvoyer Vrai
```

Placement des hubs

Pour placer les hubs de manière homogène, nous les apparentons à des zones géographiques de la "carte". Ainsi chaque va appartenir à une zone précise. La carte est découpée en nbH zones de taille égale. La figure 12.1 montre deux exemple de découpages.

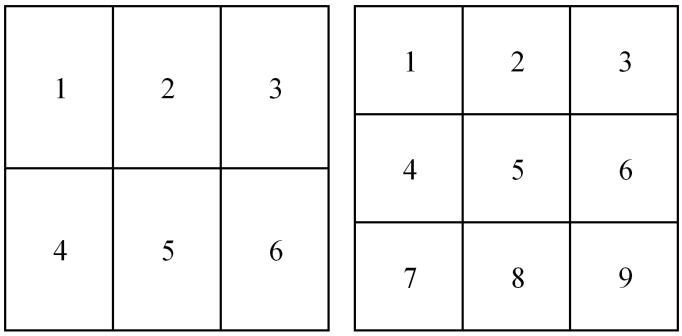


FIGURE 12.1 – Exemple de découpage de la carte en 6 et 9 zones

La procédure **placementHubs**() place un hub par zone. Pour chaque zone, la procédure **barycentre**($zone$) récupère le barycentre de la zone. Ensuite, un point du cercle de centre le barycentre et de rayon $\frac{\sqrt{nbH}}{4}$ est choisi aléatoirement. Ce point est validé, s'il vérifie la même condition d'éloignement que les CT. La figure 12.2 illustre un exemple de placement d'un hub.

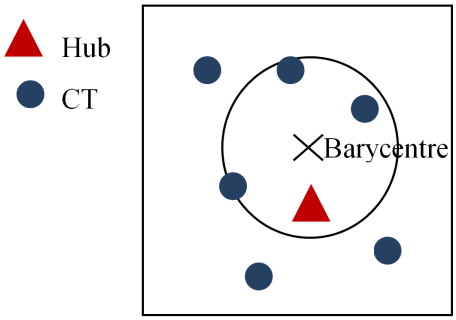


FIGURE 12.2 – Exemple de placement d'un hub

La procédure **ajoutHub**(num,cx,cy) ajoute le hub numéro num au point de coordonnées (cx,cy).

Algorithme 22 Procédure **placementHubs**()

$num \leftarrow nbCT$
$rayon \leftarrow \frac{\sqrt{nbH}}{4}$
pour tout $zone \in [\![1, nbH]\!]$ **faire**
    (cxB,cyB)$\leftarrow$ **barycentre**($zone$)
    répéter
        $a \leftarrow U(0, 360)$
        $r \leftarrow U(0, rayon)$
        $cx \leftarrow cxB + r.\cos a$
        $cy \leftarrow cyB + r.\sin a$
    jusqu'à eloignement(cx,cy)
    ajoutHub(num,cx,cy)
    $num \leftarrow num + 1$
fin pour

Création des arcs

Une de nos hypothèses est que nous n'avons pas de cabotage sur les liaisons C-H et H-C, donc aucun arc entre deux CT n'est utile. La création des arcs se fait en deux étapes :

- création des arcs pour les liaisons C-H et H-C,
- création des arcs inter-hubs.

Pour chaque hub, la procédure **arcsEtoile**() crée des arcs autour du hub pour former une étoile. La procédure **recupRayon**(h) renvoie la distance maximale entre un hub et les coins de sa zone. Ainsi un arc relie un hub et un CT, si la distance entre ces deux points est inférieure ou égale à cette distance maximale. Ceci permet d'assurer que tous les CT

d'une zone sont reliés au hub de celle-ci, mais cela permet aussi de relier un hub avec des CT d'une zone voisine. Tous les CT sont donc reliés à au moins un hub. La procédure **ajoutArc**(s,s') crée l'arc entre les sites s et s'.

Algorithme 23 Procédure **arcsEtoile**()

```
1: pour tout h ∈ H faire
2:     (r ← recupRayon(h)
3:     pour tout c ∈ C faire
4:         dist ← distEuclidienne(h.cx,h.cy,c.cx,c.cy)
5:         si dist ≤ r alors
6:             ajoutArc(h,c)
7:         fin si
8:     fin pour
9: fin pour
```

La figure 12.3 donne un exemple de création d'arcs intra-zone et inter-zone.

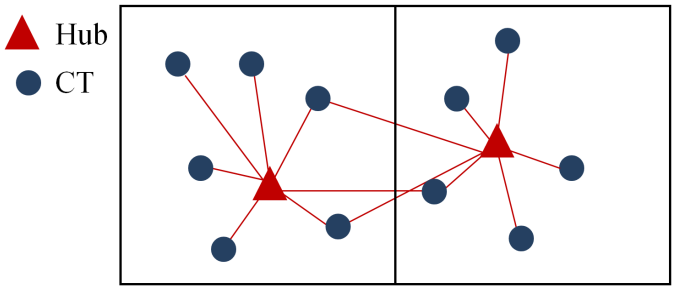


FIGURE 12.3 – Exemple de création d'arcs

Pour relier les hubs entre eux, la procédure **arcsHubs**() relie simplement tous les hubs par un arc. A noter que cette configuration est nécessaire, car une des hypothèses de travail que nous avons adoptée est que le routage primaire d'un produit contient au maximum deux hubs.

Algorithme 24 arcsHubs()

```
1: pour tout h ∈ H faire
2:     pour tout h' ∈ H faire
3:         si h.num < h'.num alors
4:             ajoutArc(h,h')
5:         fin si
6:     fin pour
7: fin pour
```

Génération des dates de fin de traitement

Une fois que tous les arcs sont créés, la procédure **calculDateFinCT**() génère les dates de fin de traitement des CT. Pour chaque C c, elle calcule la durée maximale $d_{max}(c)$ de roulage entre c et les autres CT dans le réseau. A cette durée, nous rajoutons trois périodes pour une prise en compte forfaitaire des traitements sur les sites logistiques. Puis, on y ajoute une durée correspondant à un pourcentage β de la différence entre taille de l'horizon (48) et cette date ($\beta = 25\%$ dans nos expérimentations). Enfin, la date de fin de traitement du CT c, d_c est tirée aléatoirement selon une loi uniforme sur l'intervalle $[d_{max}(c), hozizon]$.

Algorithme 25 Procédure **calculDateFinCT**()

```
1:  d_max ← 0
2:  pour tout c ∈ C faire
3:      pour tout c' ∈ C faire
4:          si c ≠ c' alors
5:              si d_max(c) < l_{c,c'} + 3 alors
6:                  d_max(c) ← l_{c,c'} + 3
7:              fin si
8:          fin si
9:      fin pour
10:     d_max(c) ← d_max(c) + β × (horizon − d_max(c))
11:     d_c ← U(d_max(c), hozizon)
12: fin pour
```

12.2.3 Création des types de véhicules

Dans nos expérimentations, nous ne considérons que deux types de véhicules :
– un pour les liaisons C-H et H-C,
– un pour les liaisons inter-hubs.

Ils disposent de la même vitesse de roulage, mais sont de capacité et de coût fixe kilométrique différents. Dans nos expérimentations, les véhicules utilisés sur les liaisons inter-hubs ont une capacité deux fois supérieure à celle des véhicules utilisés sur les autres liaisons. Ces capacités sont notées Q_C pour les véhicules des liaisons C-H et H-C et Q_H pour les liaisons inter-hubs. Cette règle de proportionnalité s'applique également aux coûts fixes kilométriques.

12.2.4 Génération des produits

La procédure **creeProduits**() génère les produits entre les CT. Pour rappel, chacun des produits possède une quantité, une date de disponibilité, une date échue et un routage primaire. Le nombre de produits à générer est donné par le paramètre $tauxP$. Cette génération se fait en trois phases :
– génération des quantités,
– génération des dates de disponibilité et des dates échues,
– génération du routage primaire.

Algorithme 26 Procédure **creeProduits**()

1: $nbP \leftarrow tauxP \times (nbCT * (nbCT - 1))$
2: $P \leftarrow \varnothing$
3: **tant que** $nbP > 0$ **faire**
4: $c \leftarrow U(0, nbCT - 1)$
5: $c' \leftarrow U(0, nbCT - 1)$
6: **si** $c \neq c' \wedge p_{c,c'} \notin P$ **alors**
7: $q \leftarrow$ **quantiteProduit**(c,c')
8: $p \leftarrow$ **nouveauProduit**(c,c',q)
9: $P \leftarrow P \oplus p$
10: $nbP \leftarrow nbP - 1$
11: **fin si**
12: **fin tant que**
13: **pour tout** $p \in P$ **faire**
14: **genDates**(p)
15: **genRP**(p)
16: **fin pour**

Tout d'abord, la quantité de chaque produit est tirée aléatoirement entre deux bornes dépendant de la tailles des CT origine et destination (petit, moyen ou grand). Si un des CT est petit (resp. moyen), alors la quantité est tirée dans l'intervalle $[\gamma_1 \times Q_C, \gamma_2 \times Q_C]$ (resp. $[\gamma_3 \times Q_C, \gamma_4 \times Q_C]$). Sinon, et dans ce cas les deux CT sont grands, la quantité de produit est tirée selon une loi uniforme sur l'intervalle $[\gamma_5 \times Q_C, \gamma_6 \times Q_C]$ ($\gamma_1 = 0.1$, $\gamma_1 = 0.5$, $\gamma_1 = 0.4$, $\gamma_1 = 0.9$, $\gamma_1 = 1$ et $\gamma_1 = 4$ dans nos expérimentations). Ceci est détaillé dans la procédure **quantiteProduit**(c,c').

Algorithme 27 Procédure **quantiteProduit**(c,c')

1: **si** $c.typeCT = 1 \vee c'.typeCT = 1$ **alors**
2: $q \leftarrow U(\gamma_1 \times Q_C, \gamma_2 \times Q_C)$
3: **sinon**
4: **si** $c.typeCT = 2 \vee c'.typeCT = 2$ **alors**
5: $q \leftarrow U(\gamma_3 \times Q_C, \gamma_4 \times Q_C)$
6: **sinon**
7: $q \leftarrow U(\gamma_5 \times Q_C, \gamma_6 \times Q_C)$
8: **fin si**
9: **fin si**
10: **renvoyer** q

Ensuite, la procédure **genDates**(p) récupère sa date échue et génère la date de disponibilité du produit p. Pour un CT c, la date échue de tous les produits y arrivant est la même, et est instancié à la date de fin de traitement du CT d_c. C'est en fonction de cette date qu'est calculée la date de disponibilité d'un produit. Pour un produit p transitant entre les CT c et c', nous calculons préalablement une date de départ au plus tard du CT origine c du produit, notée $e_{max}(c, c')$, en décomptant trois périodes pour une prise en compte forfaitaire du traitement sur les sites logistiques. La date de disponibilité du produit est alors

tirée aléatoirement selon une loi uniforme sur l'intervalle $[0, \delta \times e_{max}(c, c')]$ ($\delta = 75\%$ dans nos expérimentations).

Algorithme 28 Procédure **genDates**(p)

1: $c \leftarrow s(p)$
2: $c' \leftarrow d(p)$
3: $lc(p) \leftarrow d_{c'}$
4: $e_{max}(c, c') \leftarrow lc(p) - l_{c,c'} - 3$
5: $es(p) \leftarrow U(0, \delta \times e_{max}(c, c'))$

Enfin, pour déterminer le routage primaire d'un produit, nous calculons le plus court chemin séparant ses CT origine et destination à l'aide de l'algorithme de Dijkstra et nous récupérons le (ou les deux) hub(s) par le(s)quel(s) ce chemin transite. La procédure **genRP**(p) effectue cette opération.

Algorithme 29 Procédure **genRP**(p)

1: $(s_1, s_2, s_3, s_4) \leftarrow$ **dijkstra**($s(p)$,$d(p)$) ▷ $s_2 = s_3$ si le produit p est de type 1-hub
2: $RP(p) \leftarrow (s(p), s_2, s_3, d(p))$

12.2.5 Génération des capacités de traitement des sites logistiques

Les capacités de traitement des sites logistiques sont générées avec la procédure **creeCapa**(). Comme elles dépendent de la quantité de produits transitant par ces sites, la procédure **calculFlot**() calcule préalablement ces quantités. Pour chaque produit p, la quantité de produit $q(p)$ est ajoutée à la somme des flux SF_s associé à chaque site s du routage primaire de p qui est soit un hub, soit le CT destination. Les procédures **creeCapaCT**() et **creeCapaHubs**() génère les valeurs des capacités de traitement. Pour un CT c, la capacité de traitement η_c est calculée selon la formule 12.2 :

$$\eta_c = \left\lceil \frac{SF_c}{\frac{\delta_1 d_c}{4}} \right\rceil \tag{12.2}$$

Nous considérons donc que la quantité à traiter se fera en minimum $\frac{\delta_1 \times d_c}{4}$ périodes. Pour un hub h, le calcul de la capacité de traitement η_h est similaire et est calculé selon l'équation 12.3 :

$$\eta_h = \left\lceil \frac{SF_h}{\frac{\delta_2 horizon}{3}} \right\rceil \tag{12.3}$$

Dans nos expérimentations, le paramètre δ_2 vaut toujours 0.75, alors que le paramètre δ_1 peut prendre trois valeurs différentes (0.75, 1.00 et 1.25). Il sont le facteur de "dureté" d'une instance donné par les lettres L, M et H.

Algorithme 30 Procédure **creeCapa**()

1: **calculFlot**()
2: **creeCapaCT**()
3: **creeCapaHubs**()

Algorithme 31 Procédure **calculFlot**()

pour tout $s \in S$ **faire**
    $SF_s \leftarrow 0$
fin pour
pour tout $p \in P$ **faire**
    $SF_{d(p)} \leftarrow SF_{d(p)} + q(p)$
    $SF_{h_o(p)} \leftarrow SF_{h_o(p)} + q(p)$
    si $h_o(p) \neq h_d(p)$ **alors**
        $SF_{h_d(p)} \leftarrow SF_{h_d(p)} + q(p)$
    fin si
fin pour

Algorithme 32 Procédure **creeCapaCT**()

pour tout $c \in C$ **faire**
    $\eta_c \leftarrow \left\lceil \frac{SF_c}{\frac{\delta_1 d_c}{4}} \right\rceil$
fin pour

Algorithme 33 Procédure **creeCapaHubs**()

pour tout $h \in H$ **faire**
    $\eta_h = \left\lceil \frac{SF_h}{\frac{\delta_2 horizon}{3}} \right\rceil$
fin pour

13

Expérimentations

Sommaire

13.1 Jeux de données

Pour nos expérimentations, nous exploitons les instances produites par le générateur présenté dans le chapitre 12. Ces instances sont classées par familles suivant la convention syntaxique suivante :

I.XX.Y(F).T.Z

où :

– **XX** désigne le nombre de centres de traitement (origine + destination) de l'instance et prend pour valeur 30, 50 ou 100,

– **Y** correspond au nombre de hubs considérés, ils sont respectivement de 4, 6 et 12 hubs pour les instances avec 30, 50 et 100 CT,
– **F** spécifie le taux de couples de CT entre lesquels un produit est à router et prend pour valeur 0.25, 0.50 et 0.75,
– **T** désigne la complexité de l'instance : par ordre croissant de difficulté elle vaut L, M ou H,
– **Z** est le numéro associé à l'instance.

Nous avons généré 5 instances par jeu de paramètres, ce qui porte à 135 le nombre total d'instances sur lesquelles nous travaillons. De plus, pour un nombre de CT donné, et donc un nombre de hubs, nous avons généré des instances avec la même "géographie" et aussi les mêmes produits. Par exemple, les instances I.30.4(0.50).L.01 et I.30.4(0.50).M.01 ne diffèrent que par les capacités de traitement des hubs. Autre exemple, tous les produits de l'instance I.30.4(0.50).L.01 sont des produits de l'instance I.30.4(0.75).M.01.

Pour nos expérimentations, nous considérons 9 familles de 15 instances. Au sein d'une même famille, toutes les instances ont le même nombre de CT, le même nombre de hubs et le même taux de couples de CT entre lesquels un produit est à router. Le nom de ces familles est ainsi de la forme suivante :

I.XX.Y(F)

où les paramètres sont similaires à ceux utilisés pour l'identification des instances.

Les instances déclinant des produits pour 25 et 50% des paires de CT suivent une logique de répartition des produits propre au domaine de la messagerie (colis). Celles dont la densité atteint 75% modélisent quant à elles les configurations logistiques de la petite messagerie (courriers postaux).

Pour chaque instance, l'horizon temporel considéré couvre une journée discrétisée en 48 périodes d'une demi-heure. Nous rappelons que les capacités de traitement des hubs et des CT sont ajustées en fonction de la quantité de produits transitant par le site logistique. Par conséquent, les contraintes de capacités de traitements ne sont pas plus simple à respecter sur les instances ayant une plus faible densité.

13.2 Configuration matérielle

Les différentes stratégies de résolution présentées précédemment ont été implémentées en Java 7 et déployée sur un PC équipé d'un processeur Intel Core2 duo G860 3.0 GHz avec 8 Go de mémoire vive. Nous utilisons deux solveurs commerciaux qui sont Cplex 12.5 64 bits et Gurobi 5.5 64 bits. Notre code est exécuté au sein d'une machine virtuelle Java 1.7.0.15 en 64 bits, déployée sur Windows 7 Professionnel 64 bits.

13.3 Résolution du modèle de la modélisation multiflots

En premier lieu, nous avons tenté de résoudre le modèle global multiflots présenté dans le chapitre 3. Le tableau 13.1 présente, par famille, les résultats de la résolution directe de ce modèle par le solveur commercial Cplex12.5. Les expérimentations ont été effectuées sur les instances avec 30 et 50 CT, car le modèle ne pouvait être résolu pour les instances avec 100 CT, les modèles ne pouvant être chargés (i.e., construit dans la mémoire de l'ordinateur).

Le temps de résolution est borné et par ce fait, le solveur ne fournit pas obligatoirement une solution optimale lors de la résolution du modèle. Le temps d'exécution diffère selon le nombre de CT de l'instance :

– 1h30 pour les instances avec 30 CT,
– 2h pour les instances avec 50 CT.

La colonne $Réas$ donne le nombre d'instances pour lesquelles le solveur fournit une solution réalisable dans la limite de temps. Il s'agit en fait de la meilleure solution réalisable déterminée par la technique de Branch & Cut exécutée par le solveur.

La colonne $Gap(\%)\ Solveur$ rapporte l'écart moyen en pourcentage entre la borne inférieure du solveur (valeur de la relaxation du solveur à la fin de l'exécution) et la valeur de la fonction objectif pour la meilleure solution mixte trouvée (si une telle solution a été déterminée). Dans le document, tous les gaps sont calculés selon la même formule :

$$Gap = \frac{f(\cdot) - BI(\cdot)}{f(\cdot)} \tag{13.1}$$

Nous comparons aussi les résultats avec la meilleure borne inférieure calculée en prenant le maximum entre la borne inférieure du solveur et la borne inférieure introduites dans le chapitre 6. La colonne $Gap(\%)$ rapporte le gap moyen en pourcentage entre la meilleure borne inférieure et la valeur de la meilleure solution trouvée. Les colonnes $Min(\%)$, $Max(\%)$ et $\sigma(\%)$ donnent, respectivement, le minimum, maximum et l'écart type de la colonne $Gap(\%)$. Enfin, la ligne Synthèse donne, selon la colonne, la somme, la moyenne, le minimum, etc. des éléments de la colonne considérée

Tableau 13.1 – Résultats numériques du modèle multiflots

	$Réas$	$Gap(\%)$ $Solveur$	$Gap(\%)$	$Min(\%)$	$Max(\%)$	$\sigma(\%)$
I.30.4(0.25)	15/15	23.96	7.72	1.68	16.07	3.92
I.30.4(0.50)	11/15	15.85	6.78	1.98	15.74	4.55
I.30.4(0.75)	12/15	13.94	6.22	1.75	11.77	3.29
I.50.6(0.25)	4/15	99.55	99.49	99.42	99.57	0.06
I.50.6(0.50)	4/15	99.68	99.65	99.63	99.66	0.01
I.50.6(0.75)	0/15	X	X	X	X	X
Synthèse	46/90	32.57	23.08	1.68	99.66	35.66

Plus les instances sont volumineuses et plus le solveur a de difficultés à trouver des solutions réalisables. Pour toutes les instances de la famille I.30.4(0.25), le solveur trouve

une solution réalisable, tandis qu'aucune solution n'est trouvée pour les instances de la famille I.50.6(0.75). Nous remarquons aussi que le solveur se comporte mal pour toutes les instances avec 50 CT, car lorsqu'il trouve une solution réalisable, elle est de très mauvaise qualité, le gap moyen étant supérieur à 99%. La valorisation des solutions trouvée est en moyenne plus de 200 fois plus grande que la borne inférieure. Une analyse plus approfondie de ce phénomène montre que le solveur affecte des valeurs arbitrairement grandes aux variables associées au nombre de départs (variables $y_\omega, \omega \in \Omega$). Remarquons enfin que, pour le temps qui lui est imparti, le solveur n'est pas très stable, en effet pour les instances avec 30 CT, le gap oscille entre 2 et 16%.

13.4 Résolution du modèle indexé dans le temps

Dans cette section, nous présentons de résultats obtenus lors de la résolution du modèle global indexé dans le temps introduit dans le chapitre 4. En vue d'accélérer la résolution de ce modèle, nous y introduisons les inégalités valides et les surrogates présentées dans le chapitre 5. Les expérimentations ont été effectuées sur l'ensemble des instances.

Pour commencer, le tableau 13.2 nous donne la volumétrie des modèles pour quelques instances. Neufs instances ont servi de support pour cette synthèse, ces dernières se regroupent en trois familles. Rappelons en effet que, par exemple, l'instance I.30.4(0.25).H.01 possède la même topographie de réseau logistique que l'instance I.30.4(0.50).H.01. Les colonnes CT, $Hubs$, $Arcs$, $Produits$ et τ donnent la taille des instances en termes de nombre de CT, nombre de hubs, nombre d'arcs, nombre de produits et nombre de périodes de l'horizon temporel. Les deux colonnes $Contraintes$ et $Variables$ donnent respectivement le nombre de contraintes et le nombre de variables et donc la taille de la matrice à considérer. La dernière colonne $Non\text{-}nulles$ donne quant à elle le nombre de valeurs non-nulles de cette matrice.

Tableau 13.2 – Volumétrie du modèle indexé dans le temps avec les coupes

	CT	$Hubs$	$Arcs$	$Produits$	τ	$Contraintes$	$Variables$	$Non\text{-}nulles$
I.30.4(0.25).H.01	30	4	60	217 (25%)	48	25 680	36 680	98 540
I.30.4(0.50).H.01	30	4	60	435 (50%)	48	52 334	74 531	233 783
I.30.4(0.75).H.01	30	4	60	652 (75%)	48	80 496	111 233	403 682
I.50.6(0.25).M.04	50	6	105	612 (25%)	48	74 140	104 015	335 209
I.50.6(0.50).M.04	50	6	105	1 225 (50%)	48	157 178	208 133	871 828
I.50.6(0.75).M.04	50	6	105	1 837 (75%)	48	249 291	311 084	1 589 536
I.100.12(0.25).L.03	100	12	276	2 475 (25%)	48	333 407	457 632	1 772 576
I.100.12(0.50).L.03	100	12	276	4 950 (50%)	48	752 071	924 673	5 200 613
I.100.12(0.75).L.03	100	12	276	7 425 (75%)	48	1 258 474	1 391 689	10 306 061

Nous pouvons remarquer très rapidement que les matrices des contraintes sont très creuses. En effet, moins de 0.01% des valeurs sont non-nulles pour les instance avec 30 CT. Et ce taux diminue pour les instances avec plus de CT, avec par exemple un taux de valeur non-nulles inférieur à 0.001% pour les instances avec 100 CT.

Le tableau 13.3 synthétise par famille les résultats de la résolution directe du modèle

par le solveur commercial Gurobi 5.5. Comme pour le modèle multiflots, le temps de résolution est borné et il diffère selon le nombre de CT de l'instance :

- 1h30 pour les instances avec 30 CT,
- 2h pour les instances avec 50 CT,
- 3h pour les instances avec 100 CT.

Les colonnes sont les mêmes que pour les résultats des expérimentations sur le modèle multiflots. Nous y adjoignons cependant la colonne $Opts\ Solveur$ donnant le nombre d'instances résolues optimalement par le solveur pour chaque famille et la colonne $Opts$ précisant le nombre d'instances pour lesquelles la solution trouvée par le solveur est la solution optimale.

Tableau 13.3 – Résultats numériques du modèle indexé dans le temps avec les coupes

	$Réas$	$Opts$	$Opts$ $Solveur$	$Gap(\%)$ $Solveur$	$Gap(\%)$	$Min(\%)$	$Max(\%)$	$\sigma(\%)$
I.30.4(0.25)	15/15	5/15	5/15	0.88	0.88	0	4.79	1.29
I.30.4(0.50)	15/15	8/15	7/15	0.53	0.45	0.00	3.33	0.91
I.30.4(0.75)	15/15	8/15	8/15	0.29	0.27	0.00	1.20	0.40
I.50.6(0.25)	15/15	0/15	0/15	1.40	1.40	0.03	4.47	1.46
I.50.6(0.50)	15/15	0/15	0/15	0.65	0.65	0.04	2.18	0.71
I.50.6(0.75)	15/15	0/15	0/15	0.85	0.83	0.00	2.24	0.88
I.100.12(0.25)	15/15	0/15	0/15	2.35	2.28	0.50	4.32	1.31
I.100.12(0.50)	15/15	0/15	0/15	5.22	5.19	0.46	18.36	5.76
I.100.12(0.75)	14/15	0/15	0/15	7.48	7.44	0.63	22.13	6.67
Synthèse	134/135	21/135	20/135	2.14	2.12	0.00	22.13	3.75

Seule une seule instance de la famille I.100.12(0.75) reste sans solution réalisable trouvée. Aucune solution optimale n'est trouvée sur les instances avec 50 et 100 CT. En revanche, le solveur résout optimalement 20 des 45 instances avec 30 CT (il trouve même une 21$^{\text{ème}}$ solution optimale, mais n'a pas réussi à prouver son optimalité dans le temps imparti (Famille I.30.4(0.75))).

Les résultats obtenu sur les instances des familles avec 30 CT sont de bonne qualité en regard du temps CPU imparti. En effet le gap moyen est inférieur à 1%. Notons tout de même que le plus grand écart obtenu sur une instance avec 30 CT est proche de 5%, et qu'il correspond à une instance comprenant seulement 25% de produits. Pour les familles comportant 50 CT, les résultats obtenus sont aussi de bonne qualité. Néanmoins, remarquons que c'est encore sur une instance avec 25% de produits que le solveur est le plus éloigné de la borne inférieure en termes de gap. Ceci nous amène à conclure que ce n'est pas parce qu'une instance est petite en termes de nombre de produits à router qu'elle est systématiquement plus simple à résoudre. Rappelons à ce sujet que dans le processus de génération d'instances, nous avons adapté les capacités de traitement des sites logistiques à la quantité de flux qui devait y transiter. Remarquons enfin que le solveur a du mal à résoudre les instances avec 100 CT. La valeur des gaps moyens en atteste.

En conclusion de cette phase préliminaire d'expérimentations, le solveur n'a pas de grosses difficultés pour résoudre les instances de taille moyenne (30 ou 50 CT).

13.5 Evaluation de l'apport des coupes pour le modèle indexé dans le temps

Dans cette section, nous évaluons l'apport des inégalités valides et des surrogates sur le modèle indexé dans le temps. Pour cela, et à l'aide du solveur commercial Gurobi 5.5, nous avons résolu le modèle sans y intégrer ces coupes et surrogates. Puis nous avons comparé les résultats avec ceux obtenus précédemment sur le modèle complet. Dans un premier temps, le tableau 13.4 donne les résultats obtenus sur les différentes familles d'instances en les comparant à la meilleure borne inférieure trouvée, et nous rappelons le tableau 13.3 synthétisant les résultats obtenus pour le modèle avec coupes. Puis, dans un deuxième temps, les modèles sont comparés "seuls", c'est-à-dire en évaluant la valeur de leur fonction objectif par rapport à la valeur de leur relaxation linéaire en fin d'exécution du solveur.

Nous avons borné le temps de résolution au même temps que pour le modèle complet. Les colonnes données dans le tableau sont les même que celles pour le modèle complet et ont donc la même signification.

Contrairement au modèle avec coupes, le solveur ne parvient à prouver l'optimalité des solutions sur l'ensemble des instances dans le temps imparti. Il trouve malgré tout 4 solutions optimales en comparaison au 21 que trouve le modèle avec les coupes. Pour deux instances, le solveur exécuté sur le modèle sans les coupes ne parvient à ne trouver aucune solution réalisable. Une des deux est celle pour laquelle le modèle avec coupes n'a pas trouvé de solution réalisable non plus. Le gap moyen trouvé, en comparaison avec la meilleure borne inférieure (colonne $Gap(\%)$), est plus grand pour toutes les familles d'instances. De plus, nous remarquons que le gap du solveur en fin d'exécution (colonne $Gap(\%)\ Solveur$) est beaucoup moins bon que celui que nous calculons après l'exécution. Ce qui nous amène à dire que la valeur de la relaxation linéaire du modèle sans les coupes à la fin de l'exécution est mauvaise.

Tableau 13.4 – Résultats numériques du Modèle indexé dans le temps sans les coupes

	$Réas$	$Opts$	$Opts$ $Solveur$	$Gap(\%)$ $Solveur$	$Gap(\%)$	$Min(\%)$	$Max(\%)$	$\sigma(\%)$
I.30.4(0.25)	15/15	2/15	0/15	8.95	1.09	0.00	4.23	1.26
I.30.4(0.50)	15/15	2/15	0/15	8.18	1.32	0.00	6.54	1.64
I.30.4(0.75)	15/15	0/15	0/15	6.59	0.87	0.09	2.07	0.55
I.50.6(0.25)	15/15	0/15	0/15	11.92	3.92	1.11	7.32	1.92
I.50.6(0.50)	15/15	0/15	0/15	7.05	1.62	0.42	3.65	1.04
I.50.6(0.75)	15/15	0/15	0/15	4.91	1.05	0.23	1.58	0.44
I.100.12(0.25)	15/15	0/15	0/15	11.08	3.53	2.44	5.63	0.88
I.100.12(0.50)	15/15	0/15	0/15	13.59	9.33	1.10	17.03	6.54
I.100.12(0.75)	13/15	0/15	0/15	16.31	13.43	9.15	18.07	2.72
Synthèse	133/135	4/135	0/135	9.74	3.88	0.00	18.07	4.78

Tableau 13.3 – Résultats numériques du modèle indexé dans le temps avec les coupes

	$Réas$	$Opts$	$Opts$ $Solveur$	$Gap(\%)$ $Solveur$	$Gap(\%)$	$Min(\%)$	$Max(\%)$	$\sigma(\%)$
I.30.4(0.25)	15/15	5/15	5/15	0.88	0.88	0	4.79	1.29
I.30.4(0.50)	15/15	8/15	7/15	0.53	0.45	0.00	3.33	0.91
I.30.4(0.75)	15/15	8/15	8/15	0.29	0.27	0.00	1.20	0.40
I.50.6(0.25)	15/15	0/15	0/15	1.40	1.40	0.03	4.47	1.46
I.50.6(0.50)	15/15	0/15	0/15	0.65	0.65	0.04	2.18	0.71
I.50.6(0.75)	15/15	0/15	0/15	0.85	0.83	0.00	2.24	0.88
I.100.12(0.25)	15/15	0/15	0/15	2.35	2.28	0.50	4.32	1.31
I.100.12(0.50)	15/15	0/15	0/15	5.22	5.19	0.46	18.36	5.76
I.100.12(0.75)	14/15	0/15	0/15	7.48	7.44	0.63	22.13	6.67
Synthèse	134/135	21/135	20/135	2.14	2.12	0.00	22.13	3.75

Le tableau 13.5 synthétise les résultats des modèles avec coupes et sans coupes comparés sur la base de la valeur de la relaxation en fin d'exécution. Les colonnes $Gap(\%)$ représente l'écart moyen entre la valeur de la meilleure solution trouvée et la relaxation linéaire en fin d'exécution. Les colonnes $Min(\%)$, $Max(\%)$ et $\sigma(\%)$ donnent respectivement le minimum, le maximum et l'écart type associés à la colonne $Gap(\%)$.

Tableau 13.5 – Evaluation de l'apport des coupes pour le modèle indexé dans le temps

	MIT avec les coupes				MIT sans les coupes			
	$Gap(\%)$	$Min(\%)$	$Max(\%)$	$\sigma(\%)$	$Gap(\%)$	$Min(\%)$	$Max(\%)$	$\sigma(\%)$
I.30.4(0.25)	0.88	0.00	4.79	1.29	8.95	7.18	12.39	1.69
I.30.4(0.50)	0.53	0.00	3.33	0.94	8.18	5.48	14.89	2.29
I.30.4(0.75)	0.97	0.00	4.47	1.09	7.96	3.60	15.26	3.31
I.50.6(0.25)	1.40	0.03	4.47	1.46	11.92	8.12	15.26	2.23
I.50.6(0.50)	0.65	0.04	2.18	0.71	7.05	5.54	9.73	1.20
I.50.6(0.75)	4.96	0.46	22.21	5.41	13.53	5.14	21.48	4.48
I.100.12(0.25)	2.35	0.57	4.36	1.32	11.08	9.60	12.98	0.99
I.100.12(0.50)	5.22	0.46	18.46	5.77	13.59	5.14	21.48	6.26
I.100.12(0.75)	7.48	0.72	22.21	6.67	16.31	11.89	21.12	2.75
Synthèse	2.14	0.00	22.21	3.76	9.74	3.60	21.48	4.28

Remarquons tout d'abord que les gap moyens sur l'ensemble des familles est d'un peu plus de 2% pour le modèles avec les coupes, alors qu'il est presque de 10% pour le modèle sans les coupes. De plus, ce gap ne descend jamais en dessous des 3.6% pour le modèles sans les coupes. En revanche, sur les instances de grandes tailles (100 CT), les deux modèles peuvent mal se comporter.

13.6 Expérimentation de l'heuristique sérielle

L'heuristique sérielle introduite dans le chapitre 7 est évaluée selon deux axes :
– une analyse de la qualité des solutions, chiffrée par comparaison avec la borne inférieure,
– une analyse de la réalisabilité des solutions trouvées.

Rappelons en préambule que l'heuristique sérielle n'est pas toujours en mesure de router l'intégralité des produits. Notre analyse porte donc sur le taux de produits incomplètement routés (nombre de produits incomplètement routés divisé par le nombre total de produits) ainsi que sur le ratio de quantité d'unités de produits non routées (quantité d'unités de produits non routées divisée par la quantité totale de produits). Rappelons également que deux paramètres $\mathcal{E}_C$ (resp. $\mathcal{E}_H$) conditionnent le degré de massification des solutions de l'heuristique sur les étages C-H et H-C (resp. H-H). Dans un premier temps nous fixons ces deux paramètres à zéro dans l'optique de réduire autant que possible les déficits en quantité routée pouvant être constatés pour certains produits à l'issue de l'heuristique.

Le tableau 13.6 synthétise par famille les résultats associés à l'heuristique sérielle. La colonne $Réas$ donne le nombre d'instances pour lesquelles l'heuristique trouve une solution réalisable, donc lorsque l'intégralité des produits est routée. La colonne $Gap(\%)$ donne le gap moyen vis-à-vis de la meilleure borne inférieure connue. Il est calculé selon la formule 13.1. La colonne $\sigma(\%)$ donne l'écart type associé à ce gap, alors que la colonne $CPU(s)$ précise le temps moyen d'exécution de l'heuristique en secondes.

Tableau 13.6 – Résultats numériques de l'heuristique sérielle : qualité des solutions

	$Réas$	$CPU(s)$	$Gap(\%)$	$\sigma(\%)$
I.30.4(0.25)	9/15	0.147	45.29	4.43
I.30.4(0.50)	4/15	0.180	43.30	3.42
I.30.4(0.75)	6/15	0.205	39.02	3.35
I.50.6(0.25)	4/15	0.247	44.04	2.47
I.50.6(0.50)	0/15	0.536	38.65	2.48
I.50.6(0.75)	0/15	0.763	34.44	2.16
I.100.12(0.25)	0/15	1.815	44.56	1.92
I.100.12(0.50)	0/15	4.495	38.09	2.12
I.100.12(0.75)	0/15	8.411	32.54	2.36
Synthèse	23/135	1.866	39.99	5.16

Nous pouvons constater que le temps d'exécution de l'heuristique est très faible sur les instances avec 30 et 50 CT puisqu'il est au maximum de 0.965 s. En revanche sur les familles d'instances avec 100 CT, elle est un peu plus longue, mais ne dépasse jamais les dix secondes. Même pour les grosses instances, l'heuristique ne parvient jamais à trouver de solution réalisable. De plus, le gap vis-à-vis de la borne inférieure est très élevé et ce indépendamment de la taille de l'instance (il s'élève en moyenne à 40%). L'écart type semble par contre attester que l'heuristique est plutôt stable.

La colonne $Déficit(\%)$ pour les produits du tableau 13.7 précise pour chaque famille d'instance le pourcentage moyen de produits incomplètement routés et la colonne $\sigma(\%)$

donne l'écart type associé. La colonne $Déficit(\%)$ des quantités donne le pourcentage moyen d'unités de produits non routées par famille et la colonne $\sigma(\%)$ donne l'écart type associé à ce déficit.

Tableau 13.7 – Résultats numériques de l'heuristique sérielle : réalisabilité des solutions

	$Produits$		$Quantités$	
	$Déficit(\%)$	$\sigma(\%)$	$Déficit(\%)$	$\sigma(\%)$
I.30.4(0.25)	0.61	1.14	0.37	0.62
I.30.4(0.50)	0.54	0.69	0.39	0.47
I.30.4(0.75)	0.51	0.66	0.35	0.44
I.50.6(0.25)	0.39	0.36	0.26	0.26
I.50.6(0.50)	0.25	0.18	0.15	0.12
I.50.6(0.75)	0.24	0.17	0.18	0.12
I.100.12(0.25)	0.17	0.07	0.11	0.04
I.100.12(0.50)	0.18	0.08	0.12	0.06
I.100.12(0.75)	0.18	0.05	0.13	0.04
Synthèse	0.34	0.53	0.23	0.33

Nous pouvons constater que le taux de produits incomplètement routés est très faible, au même titre que le pourcentage de quantité non routée. En effet, en moyenne 0.34% des produits sont non routés pour une quantité moyenne de 0.23%. Enfin, plus la taille de l'instance est grande, plus les taux de déficits diminuent.

Nous mesurons maintenant l'impact des paramètres $\mathcal{E}_C$ et $\mathcal{E}_H$ conditionnant le degré de massification sur les véhicules. Pour cela, nous avons exécuté l'heuristique sérielle en faisant évoluer ces deux paramètres, en même temps, de 10% en 10%. Le tableau 13.8 donne ces résultats. La colonne $\mathcal{E}_C/\mathcal{E}_H(\%)$ donne la valeur des paramètres. La colonne $Gap(\%)$ donne le gap moyen de la solution trouvée par l'heuristique, ce dernier étant calculé à l'aide de la formule 13.1. Attention : cette solution ne prenant en compte que les quantités de produits routées, la valeur trouvée peut être inférieure à la valeur de la borne inférieure. La colonne $Déficit(\%)$ $Produits$ précise le déficit moyen, en pourcentage, du nombre de produits non routés entièrement observé dans les solutions trouvées. La colonne $Déficit(\%)$ $Quantité$ spécifie le déficit moyen, en pourcentage, des unités de produits non routées. L'ensemble des valeurs sont les moyennes calculées sur les 135 instances de notre test bed.

Nous remarquons clairement que plus la valeur des paramètres est élevée, plus le gap diminue, mais cela au détriment du nombre de produits et de la quantité routés. Le graphique 13.1 permet de voir l'évolution des écarts en pourcentage en fonction de la valeur des paramètres. La courbe bleu représente le gap moyen de la valeur des solutions, alors que les courbes rouge et verte représentent respectivement l'évolution du déficit en nombre de produits et quantité de produits non routés dans les solutions.

Tableau 13.8 – Résultats numériques de l'évolution des différents gaps de l'heuristique sérielle en fonction des paramètres $\mathcal{E}_C$ et $\mathcal{E}_H(\%)$

$\mathcal{E}_C/\mathcal{E}_H(\%)$	$Gap(\%)$	$Déficit(\%)$ $Produits$	$Déficit(\%)$ $Quantité$
0	39.99	0.34	0.23
10	35.81	2.77	0.67
20	27.75	7.31	2.22
30	17.28	13.17	5.16
40	7.09	19.81	8.61
50	-2.63	28.06	12.65
60	-12.36	33.40	17.14
70	-20.64	39.01	21.43
80	-30.04	44.19	25.80
90	-39.60	49.49	30.45
100	-48.37	54.33	34.86
Synthèse	-2.34	26.53	14.47

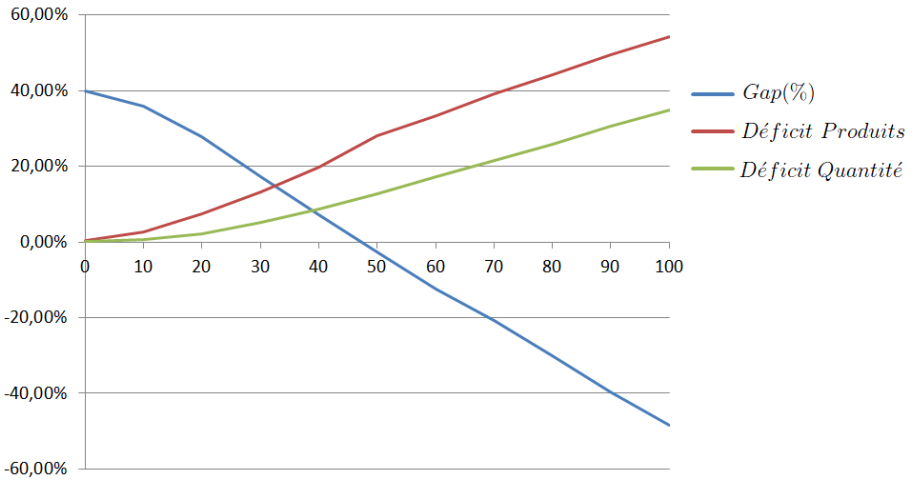


FIGURE 13.1 – Evolution des gaps en fonction de la valeur des paramètres $\mathcal{E}_C$ et $\mathcal{E}_H(\%)$

Ce graphique met clairement en évidence l'influence des paramètres sur la réalisabilité des solutions trouvées par l'heuristique sérielle, cette dernière étant meilleure plus la valeur des paramètres est faible.

13.7 Évaluation de l'heuristique d'amélioration

L'heuristique d'amélioration étant basée sur la modélisation multiflots, le solveur commercial Cplex 12.5 est utilisé pour l'évaluation de cette approche. Ce modèle ne tournant pas avec les instances avec plus de 50 CT (cf. section 13.3), seules les familles d'instances avec 30 et 50 CT sont considérées dans cette section.

Le tableau 13.9 présente les résultats associés à l'heuristique d'amélioration. Il se compose des mêmes colonnes que le tableau 13.6 relatif à l'heuristique sérielle, auquel nous adjoignons une nouvelle colonne $Gap(\%)$ $Amélioration$ qui représente l'amélioration en terme de fonction objectif qu'apporte cette phase suite à l'heuristique sérielle. Il est calculé avec la formule suivante :

$$Gap(Ame) = \frac{f(HS) - f(Ame)}{f(HS)} \tag{13.2}$$

où $f(HS)$ est la valeur de la fonction objectif à l'issue de l'heuristique sérielle et $f(Ame)$ est la valeur de la fonction objectif à la fin de l'heuristique d'amélioration.

Tableau 13.9 – Résultats numériques de l'heuristique d'amélioration

	$Réas$	$CPU(s)$	$Gap(\%)$	$\sigma(\%)$	$Gap(\%)$ $Amélioration$
I.30.4(0.25)	9/15	0.139	41.25	4.20	6.92
I.30.4(0.50)	4/15	0.530	41.56	3.39	2.99
I.30.4(0.75)	6/15	0.227	37.87	3.22	1.86
I.50.6(0.25)	4/15	0.119	42.17	2.59	3.23
I.50.6(0.50)	0/15	0.680	37.96	2.50	1.11
I.50.6(0.75)	0/15	0.510	34.02	2.14	0.63
Synthèse	23/90	0.189	39.14	4.14	2.79

Nous pouvons constater que l'heuristique d'amélioration n'obtient pas de nouvelles solutions réalisables par rapport à l'heuristique sérielle. Le gap moyen obtenu est un peu plus faible que celui obtenu par l'heuristique sérielle, mais il reste néanmoins très grand. Enfin, l'amélioration en termes de valeur de fonction objectif semble décroître plus la taille des instances est grande.

13.8 Evaluation de l'heuristique d'arrondi successif

L'heuristique d'arrondis successifs a été développée sur la base de la modélisation indexée dans le temps. Le temps d'exécution de cette heuristique a été borné à 10 minutes. Rappelons que pour fonctionner, l'heuristique a besoin de deux paramètres ϵ et α. ϵ représente le seuil pour lequel les variables sont arrondies et α est le pas d'évolution de ce seuil. Les valeurs initiales des paramètres utilisés pour l'exécution de cette méthode sont :

- $\epsilon = 0.000$
- $\alpha = 0.003$

Le tableau 13.10 présente les résultats de l'heuristique d'arrondis successifs sur l'ensemble des familles d'instances. La colonne $Gap(\%)$ donne l'écart entre la fonction objectif à l'issue de l'exécution de l'heuristique et la meilleure borne inférieure. Il est calculé au moyen de la formule 13.1. L'écart type lié à ce gap est représenté par la colonne $\sigma(\%)$. Enfin, la colonne $CPU(s)$ donne le temps moyen d'exécution en secondes.

Tableau 13.10 – Résultats numériques de l'heuristique d'arrondis successifs

	$Gap(\%)$	$\sigma(\%)$	$CPU(s)$
I.30.4(0.25)	18.05	3.78	13
I.30.4(0.50)	10.75	2.62	73
I.30.4(0.75)	11.20	9.70	364
I.50.6(0.25)	12.27	2.34	145
I.50.6(0.50)	22.72	13.14	549
I.50.6(0.75)	31.68	4.01	600
I.100.12(0.25)	45.92	2.81	600
I.100.12(0.50)	41.06	1.93	600
I.100.12(0.75)	35.74	2.34	600
Synthèse	25.49	14.07	394

Nous remarquons que le gap moyen est de qualité convenable sur les instances avec 30 CT. Notons de plus un résultats à la base assez contre intuitif, ce gap est moins bon pour la famille avec un taux de 25% des produits en comparaison aux familles avec un taux supérieur. Ceci peut s'expliquer par le fait que les modèles correspondant à ces instances contiennent moins de variables de décision et qu'ainsi l'effet itératif de la méthode a moins d'impact.

13.9 Evaluation de l'heuristique d'échelonnement de capacité

Cette section présente les résultats obtenus avec l'heuristique d'échelonnement de capacité présentée dans le chapitre 10. Cette heuristique comporte deux versions, mais suite à des expérimentations préliminaires, seule la deuxième version a été utilisée. Cette dernière consiste à démultiplier les variables portant sur les véhicules. En raison de la taille importante du modèle $[P_{EC2}]$ (variables véhicules dupliquées), cette approche ne "supporte" que des instances à 30 CT, et à 50 CT pour 25% et 50% de taux de produits. Le temps fixé pour l'exécution de cette heuristique est borné à la moitié du temps donné pour la LNS :

- 45 minutes pour les instances avec 30 CT,
- 1h pour les instances avec 50 CT.

Dans les résultats que nous présentons, les valeurs des paramètres utilisées sont les suivantes :

- $\lambda = 0.05$
- $\gamma = 0.85$
- $\epsilon = 0.01$

Le tableau 13.11 synthétise les résultats obtenus. La colonne $Gap(\%)$ précise le gap moyen entre la valeur de la fonction objectif de la meilleure solution trouvée et la valeur de la meilleure borne inférieure. Les colonnes $Min(\%)$, $Max(\%)$ et $\sigma(\%)$ donnent respectivement le minimum, le maximum et l'écart type associés à ce gap. La colonne $Itérations$ précise le nombre moyen d'itérations effectuées par la méthode, tandis que

les colonnes $Itérations\ Best$ et $CPU(s)\ Best$ donnent l'itération à laquelle la meilleure solution a été obtenue et le temps en secondes auquel elle a été obtenue.

Tableau 13.11 – Résultats numériques de l'heuristique d'échelonnement de capacité

	$Gap(\%)$	$Min(\%)$	$Max(\%)$	$\sigma(\%)$	$Itérations$	$Itérations\ Best$	$CPU(s)\ Best$
I.30.4(0.25)	7.46	3.16	12.4	2.76	344	31	188
I.30.4(0.50)	2.78	0.65	7.12	1.61	45	7	488
I.30.4(0.75)	0.81	0.04	2.09	0.60	14	4	719
I.50.6(0.25)	3.99	1.83	6.46	1.28	22	5	987
I.50.6(0.50)	1.15	0.45	2.12	0.53	4	3	3413
Synthèse	3.24	0.04	12.4	2.87	86	10	1611

En premier examen, les résultats semblent probants. Mais on peut cependant remarquer que sur n'importe quelle famille d'instance, la meilleure solution est trouvée lors des premières itérations. Ceci ne correspond pas au comportement escompté, la meilleure solution devant être trouvée en fin d'exécution de l'heuristique. Une explication possible de ce comportement pathologique réside dans le fait que plus on avance dans le processus de résolution, moins le nombre de variables libres est important, ce qui induit une baisse de qualité des solutions produites. Nos tentatives de tuning des paramètres de la méthode se sont avérées vaines.

13.10 Evaluation de la LNS LNS(MMF)

La première version de la LNS utilise la modélisation multiflots proposée dans le chapitre 3 de ce manuscrit. Comme nous l'avons déjà mentionné, elle ne peut être exécutée que sur les instances impliquant 50 CT maximum, le simple chargement du modèle sous Cplex étant déjà problématique. La LNS étant initialisée par l'heuristique d'amélioration, notons qu'elle peut partir de ce fait d'une solution non réalisable.

Le temps d'exécution octroyé à cette LNS est borné avec des temps CPU identiques à ceux utilisés pour l'exécution du solveur sur les modèles seuls. Il diffère selon le nombre de CT de l'instance :

- 1h30 pour les instances avec 30 CT,
- 2h pour les instances avec 50 CT.

Comme pour l'heuristique d'amélioration, le solveur utilisé est Cplex 12.5.

Le tableau 13.12 synthétise les résultats de la première version de la LNS. La colonne $Réas$ (resp. $Opts$) précise le nombre d'instances par famille pour lesquelles une solution réalisable (resp. optimale) a été trouvée en fin d'exécution de la LNS. La colonne $Gap(\%)$ donne le gap moyen entre la valeur de la fonction objectif de la meilleure solution trouvée et la valeur de la meilleure borne inférieure. Les colonnes $Min(\%)$, $Max(\%)$ et $\sigma(\%)$ donnent respectivement le minimum, le maximum et l'écart type associés à ce gap.

Tableau 13.12 – Résultats numériques de la LNS(MMF)

	$Réas$	$Opts$	$Gap(\%)$	$Min(\%)$	$Max(\%)$	$\sigma(\%)$
I.30.4(0.25)	15/15	0/15	4.85	0.29	19.67	5.78
I.30.4(0.50)	15/15	0/15	7.26	0.14	29.37	9.80
I.30.4(0.75)	12/15	0/15	0.81	0.04	2.77	0.84
I.50.6(0.25)	12/15	0/15	3.71	0.41	9.27	2.34
I.50.6(0.50)	8/15	0/15	2.08	1.31	3.33	0.65
I.50.6(0.75)	6/15	0/15	2.77	2.22	3.49	0.53
Synthèse	68/90	0/90	3.96	0.04	29.37	5.75

Tout d'abord, nous remarquons que cette version de la LNS ne parvient à trouver aucune solution optimale, ni de solutions réalisables pour certaines instances. c'est le cas pour trois instances de la famille I.30.4(0.75), et cette situation se produit plus souvent sur les familles d'instances avec 50 CT. En effet, l'heuristique sérielle trouvant moins de solutions réalisables sur ces instances, la LNS part avec un handicap. Le gap moyen obtenu sur les familles d'instances avec 30 CT est de 4.56%, performance médiocre étant donné le temps d'exécution laissé à la méthode. Ce gap atteint même 30% sur une instance. En contrepartie, nous pouvons constater un très bon comportement de la LNS sur la famille d'instances I.30.4(0.75). Ceci peut être expliqué par le fait que ces instances impliquent plus de produits offrant plus de possibilités pour réviser la solution, le voisinage étant plus grand. La LNS n'est d'autre part pas très stable sur les instances avec 30 CT et cela est confirmé par la valeur de l'écart type qui est élevée. En revanche, sur les familles d'instances avec 50 CT, et lorsque la méthode trouve une solution réalisable, celle-ci s'avère de meilleure qualité : le gap moyen est 3% et l'écart type a une valeur moyenne de 1.76%. Enfin, sur les 67 instances pour lesquelles la solution initiale n'était pas réalisable, la LNS réussit à trouver 45 solution réalisable à la fin de son exécution.

Le tableau 13.13 présente les moyennes du nombre d'itérations effectuées pas la LNS(MMF) (colonne $Itérations$). La colonne ($Itérations\ Irréalisables$) donne la moyenne du nombre d'itérations nécessaires à la LNS pour retomber sur une solution réalisable.

Tableau 13.13 – Nombre d'itérations de la LNS(MMF)

	$Itérations$	$Itérations\ Irréalisables$
I.30.4(0.25)	815	42
I.30.4(0.50)	838	32
I.30.4(0.75)	886	185
I.50.6(0.25)	1062	290
I.50.6(0.50)	860	294
I.50.6(0.75)	908	491
Synthèse	895	222

En observant le tableau 13.13, nous pouvons constater que le nombre moyen d'itérations est invariant avec la taille des instances. En contre partie, le nombre moyen d'ité-

rations nécessaires à la LNS pour trouver une solution réalisable augmente significativement avec la taille de l'instance. Notons cependant que pour les instances pour lesquelles aucune solution réalisable n'est trouvée, toutes les itérations sont comptabilisées.

13.11 Evaluation de la LNS LNS(MIT)

La deuxième version de la LNS utilise la modélisation indexée dans le temps présentée dans le chapitre 4 à laquelle sont ajoutées les coupes et les surrogates présentées dans le chapitre 5. La solution utilisée pour initialiser cette méthode est la solution trouvée en fin d'exécution de l'heuristique d'arrondis successifs avec une temps d'exécution maximum de 10 minutes pour ne pas empiéter de trop sur le temps CPU dévolu à la LNS. Contrairement à la version LNS(MMF), celle-ci peut s'exécuter sur des instances allant jusqu'à 100 CT et 12 hubs. Le solveur utilisé pour résoudre les MIP exploités au cours du déroulement de cette approche est ici Gurobi 5.5. Le temps maximal d'exécution de la méthode est borné et fonction de la taille de l'instance :

- 1h30 pour les instances avec 30 CT,
- 2h pour les instances avec 50 CT,
- 3h pour les instances avec 100 CT.

Le tableau 13.14 présente les résultats de la deuxième version de la LNS. La structure de ce tableau est la même que le tableau 13.12 pour la première LNS, à l'exception de la colonne $Itérations$ (tableau 13.13).

Tableau 13.14 – Résultats numériques de la LNS(MIT)

	$Réas$	$Opts$	$Gap(\%)$	$Min(\%)$	$Max(\%)$	$\sigma(\%)$	$Itérations$
I.30.4(0.25)	15/15	2/15	1.14	0.00	5.26	1.63	4096
I.30.4(0.50)	15/15	6/15	0.54	0.00	3.17	0.89	3658
I.30.4(0.75)	15/15	7/15	0.14	0.00	1.28	0.32	2935
I.50.6(0.25)	15/15	0/15	0.99	0.08	2.60	0.87	1579
I.50.6(0.50)	15/15	3/15	0.26	0.00	1.27	0.36	1410
I.50.6(0.75)	15/15	0/15	0.47	0.01	1.94	0.57	762
I.100.12(0.25)	15/15	0/15	1.40	0.24	3.02	0.81	780
I.100.12(0.50)	15/15	0/15	2.49	0.66	5.11	1.08	485
I.100.12(0.75)	15/15	0/15	3.78	0.34	7.54	2.25	347
Synthèse	135/135	18/135	1.24	0.00	7.54	1.58	1784

La solution initialisant cette LNS étant réalisable, les problèmes de réalisabilité constatés pour la version précédente n'ont plus lieu d'être ici. 18 optimums sont trouvés par cette méthode, la plupart sur des instances avec 30 CT. Les gaps moyens obtenus pour les différentes familles d'instances sont de bonne qualité. Pour les familles d'instances pour lesquelles des résultats ont été obtenus avec l'une et l'autre versions, la dominance de LNS(MIT) est indéniable. La modélisation indexée dans le temps, mieux conditionnée pour les solveurs de PLNE (Relaxation linéaire de meilleure qualité notamment) contribue à expliquer cet avantage. Une explication additionnelle réside dans le fait que ce modèle permet l'exploitation de voisinages plus grands en lien avec un nombre d'itérations accru

L'examen des écarts types permet enfin de statuer à une une meilleure stabilité de cette approche.

13.12 Evaluation de notre borne inférieure

Précisons avant toute chose que les bornes inférieures introduites dans le chapitre 6 sont calculées en résolvant les modèles associés à chaque étage, et non par l'intermédiaire des algorithmes polynomiaux s'y rapportant, aucune preuve formelle de la validité de ces heuristiques n'étant disponible. Afin d'évaluer la valeur de cette borne inférieure, nous la comparons à la meilleure que nous avons trouvée, soit en récupérant la relaxation linéaire à l'issue de la résolution par le solveur commercial Gurobi 5.5 du modèle indexé dans le temps avec les coupes et les surrogates, soit en récupérant celle calculée via la décomposition en étages.

Le tableau 13.15 synthétise les résultats obtenus. La colonne $MBIC$ spécifie le nombre d'instances par famille pour lesquelles notre borne inférieure est la meilleure borne inférieure connue. La colonne $Opts$ donne le nombre d'instances par famille pour lesquelles notre borne inférieure est égale à la solution optimale lorsque celle-ci est trouvée. La colonne $Gap(\%)$ précise l'écart moyen entre notre borne inférieure et la meilleure trouvée. Il est calculé grâce à la formule 13.1. Enfin, les colonnes $Min(\%)$, $Max(\%)$ et $\sigma(\%)$ donnent respectivement le minimum, le maximum et l'écart type associés à ce gap.

Tableau 13.15 – Evaluation de notre borne inférieure

	$MBIC$	$Opts$	$Gap(\%)$	$Min(\%)$	$Max(\%)$	$\sigma(\%)$
I.30.4(0.25)	0/15	0/5	1.27	0.13	2.40	0.85
I.30.4(0.50)	9/15	6/8	0.05	0.00	0.19	0.08
I.30.4(0.75)	15/15	10/10	0.00	0.00	0.00	0.00
I.50.6(0.25)	0/15	0/0	0.34	0.01	1.09	0.31
I.50.6(0.50)	10/15	3/3	0.01	0.00	0.11	0.03
I.50.6(0.75)	15/15	0/0	0.00	0.00	0.00	0.00
I.100.12(0.25)	13/15	0/0	0.00	0.00	0.00	0.00
I.100.12(0.50)	15/15	0/0	0.00	0.00	0.00	0.00
I.100.12(0.75)	15/15	0/0	0.00	0.00	0.00	0.00
Synthèse	92/135	19/26	0.19	0.00	2.40	0.49

Un premier constat est que notre borne inférieure est la meilleure pour 92 des 135 instances. Sur les 26 instances résolues optimalement par l'une de nos méthodes, sa valeur est égale à l'optimum dans 19 cas, et elle permet de prouver l'optimalité sur 3 instances avec 50 CT. Sinon, le gap moyen sur l'ensemble de instances est inférieur à 0.2%, et la qualité de ces bornes augmente avec la taille des instances, ce qui peut s'expliquer par le fait que le comportement des solveurs commerciaux est dégradé sur les instances de taille importante.

13.13 Comparaison des différentes méthodes

Dans cette section, nous comparons les résultats obtenus par l'ensemble des méthodes testées précédemment. Le tableau 13.16 présente les gaps moyens en pourcentage obtenus par ces différentes méthodes sur les différentes familles d'instances. La colonne $H.S.$ précise les résultats obtenus par l'heuristique sérielle. La colonne $H.Am.$ spécifie ceux obtenus par l'heuristique d'amélioration. Les résultats de l'heuristique d'arrondis successifs sont donnés dans la colonne $H.Ar.$, tandis que la colonne $H.E.C.$ contient les résultats de l'heuristique d'échelonnement de capacité. Les colonnes MMF et MIT donnent les résultats obtenus pour la résolution des modèles multiflots et indexé dans le temps, alors que les colonnes $LNS(MMF)$ et $LNS(MIT)$ spécifient les résultats obtenus par les deux versions de la LNS que nous avons implémentées.

Tableau 13.16 – Comparaison des gaps moyen en % de toutes les méthodes

	$H.S.$	$H.Am.$	$H.Ar.$	$H.E.C.$	MMF	MIT	LNS (MMF)	LNS (MIT)
I.30.4(0.25)	45.29	41.25	18.05	7.46	7.72	0.88	4.85	1.14
I.30.4(0.50)	43.30	41.56	10.75	2.78	6.78	0.45	7.26	0.54
I.30.4(0.75)	39.02	37.87	11.20	0.81	6.22	0.27	0.81	0.14
I.50.6(0.25)	44.04	42.17	12.27	3.99	99.49	1.40	3.71	0.99
I.50.6(0.50)	38.65	37.96	22.72	1.15	99.65	0.65	2.08	0.26
I.50.6(0.75)	34.44	34.02	31.68	X	X	0.83	2.77	0.47
I.100.12(0.25)	44.56	X	45.92	X	X	2.28	X	1.40
I.100.12(0.50)	38.09	X	41.06	X	X	5.19	X	2.49
I.100.12(0.75)	32.54	X	35.74	X	X	7.44	X	3.78
Synthèse	39.99	39.14	25.49	3.24	23.08	2.12	3.96	1.24

Ce tableau ne fait pas apparaitre les temps d'exécution, qui sont très faibles pour certaines méthodes comme l'heuristique sérielle en comparaison aux LNS et aux modèles complets, mais il semble clair que la LNS(MIT) et le modèle indexé dans le temps fournissent les meilleurs résultats. Le graphique 13.2 présente les différents atouts de nos méthodes.

Ce graphique met bien en évidence la dominance de la méthode LNS(MIT) en termes de qualité de résultats. En contre partie, son temps d'exécution important et sa complexité accrue en comparaison à d'autres méthodes comme l'heuristique sérielle ou l'heuristique d'arrondis successifs nuancent ces résultats.

Dans un second temps, nous comparons plus précisément la LNS(MIT) et le modèle complet MIT avec coupes et surrogates. Le tableau 13.17 synthétise les résultats de cette comparaison. Les colonnes $Réas$ donnent le nombre d'instances par famille pour lesquelles chacune des méthodes a trouvé une solution réalisable. Les colonnes $Opts$ précisent quant à elles le nombre de solutions optimales trouvées, alors que les colonnes MSC spécifient le nombre d'instances par famille pour lesquelles chacune des méthodes a trouvé la meilleure solution connue. Enfin, les colonnes $Gap(\%)$ donnent l'écart moyen entre la valeur de la meilleure solution trouvée par ces méthodes et la valeur de la meilleure borne inférieure calculé avec la formule 13.1.

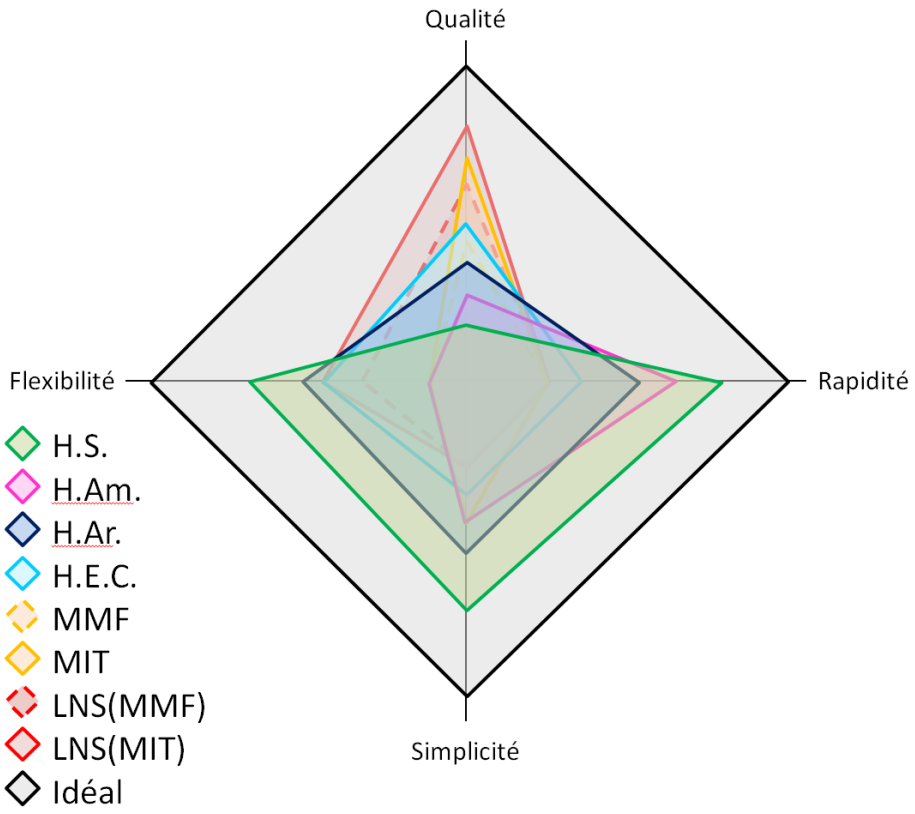


FIGURE 13.2 – Comparaison graphique des méthodes testées

Tableau 13.17 – Comparaison de deux meilleures méthodes : la LNS(MIT) et le modèle MIT

	$LNS(MIT)$				MIT			
	Réas	*Opts*	MSC	$Gap(\%)$	*Réas*	*Opts*	MSC	$Gap(\%)$
I.30.4(0.25)	15/15	2/15	6/15	1.14	15/15	5/15	12/15	0.88
I.30.4(0.50)	15/15	6/15	9/15	0.54	15/15	8/15	12/15	0.45
I.30.4(0.75)	15/15	7/15	11/15	0.14	15/15	8/15	8/15	0.27
I.50.6(0.25)	15/15	0/15	9/15	0.99	15/15	0/15	6/15	1.40
I.50.6(0.50)	15/15	3/15	15/15	0.26	15/15	0/15	0/15	0.65
I.50.6(0.75)	15/15	0/15	8/15	0.47	15/15	0/15	5/15	0.83
I.100.12(0.25)	15/15	0/15	14/15	1.40	15/15	0/15	1/15	2.28
I.100.12(0.50)	15/15	0/15	7/15	2.49	15/15	0/15	8/15	5.19
I.100.12(0.75)	15/15	0/15	14/15	3.78	14/15	0/15	1/15	7.44
Synthèse	135/135	18/135	93/135	1.24	134/135	21/135	53/135	2.12

Comme nous l'avions déjà constaté, le solveur ne trouve pas de solution sur une instance, alors que la LNS en trouve toujours une. Certes le solveur trouve plus de solutions optimales (21), mais celles-ci sont toutes des instances avec 30 CT. La LNS de son côté trouve trois solutions optimales sur des instances avec 50 CT. Pour les meilleures solutions connues, le solveur commercial domine la LNS sur les familles d'instances avec 30

CT. En revanche, dès que le nombre de CT est de 50 ou 100, c'est la LNS qui trouve le plus de meilleures solutions. Enfin, la comparaison des moyennes des gaps va dans le même sens, le solveur étant meilleur que la LNS sur les instances avec 30 CT. Il est juste derrière la LNS pour les instances avec 50 CT, et il est quasiment deux fois supérieur en moyenne sur les plus grandes instances. Malgré cela, il lui arrive quand même d'être meilleur que la LNS sur certaines instances. En conclusion, nous pouvons dire que la LNS fait jeu égal avec le solveur sur les instances de taille moyenne et dès que le nombre de CT augmente, elle devient la plupart du temps meilleure.

En conclusion, malgré toutes les méthodes mises en place, le problème s'est montré spécialement difficile. Pour de petites instances (30 CT), une bonne modélisation du problème permet à un solveur commercial de les résoudre, mais pas forcément de manière optimale. En revanche, si la taille des instances augmente, la modélisation atteint ses limites. Ensuite, l'heuristique sérielle et l'heuristique d'amélioration permettent de donner une solution initiale au problème très rapidement, mais celle-ci n'est pas de bonne qualité ni réalisable tout le temps. L'heuristique d'arrondis successifs donne de bons résultats en un temps acceptable sur les instances à 30 CT, mais celui-ci devient rapidement rédhibitoire sur des instances avec plus de CT. Quant à l'heuristique d'échelonnement de capacité, elle résout correctement le problème sur les petites instances, mais est rapidement bloquée par la taille des instances. Enfin, la LNS, dans ses deux versions, dispose d'un grand temps d'exécution, elle obtient de très bon résultats sur les familles d'instances avec 30 CT. Et c'est la méthode LNS(MIT) qui résout le mieux les instances avec 50 et 100 CT

14

Conclusion

Après avoir décrit notre problématique et établi un état de l'art autour de celle-ci, nous l'avons formalisée et modélisée de deux manières différentes. La première formalisation se base sur la notion de multiflots, alors que la seconde est une modélisation indexée dans le temps. Une ensemble de coupes et de surrogates ont été ajoutées pour améliorer la résolution de nos modèles. La résolution directe du modèle global issu de la première modélisation par solveur commercial est impraticable à partir d'instances de taille modérée. En revanche, la résolution du modèle issu de la seconde modélisation se comporte beaucoup mieux, mais elle atteint tout de même ses limites sur les instances de grande taille.

Nous avons aussi proposé une technique de décomposition spatiale du problème, couplée à une agrégation des produits au travers de la notion de flux cumulé dans l'optique de calculer une borne inférieure de l'optimum de notre problème. Nous proposons ensuite une heuristique sérielle en vue de construire une solution à notre problème. Bien que rapide, cette dernière est rarement en mesure de produire une solution routant l'intégralité des produits. Cette solution est améliorée par la suite avec une méthode nommée heuristique d'amélioration basée sur la modélisation multiflots. Elle permet de massifier un peu plus les flux de la solution fournie par l'heuristique sérielle.

Ensuite nous exposons une heuristique d'arrondis successifs utilisant la relaxation linéaire de la modélisation indexée dans le temps. Les expérimentations faites sur cette heuristique montrent qu'elle est performante sur les instances de petite taille, mais la résolution itérative de relaxations linéaires sur des modèles de grande taille ne lui permet pas d'atteindre de bonnes solutions en un temps de calcul raisonnable.

Nous proposons aussi deux versions d'une heuristique d'échelonnement de capacité pour résoudre notre problème. Suite à des tests préliminaires, la première, utilisant l'agrégation des moyens de transport, s'est avérée moins performante que la seconde version, utilisant quant à elle la désagrégation de ces véhicules. Malgré tout, nous n'avons pas obtenu les résultats escomptés pour cette deuxième version, celle-ci trouvant ses meilleures solutions lors de ses premières itérations.

Nous exposons alors une méta-heuristique LNS (Large Neighborhood Search). Après

avoir décrit le déroulement classique d'une LNS, nous proposons une version adaptée de la méthode à notre problématique basée sur des voisinages utilisant des MIP. De ce fait, nous déclinons cette méthode en deux versions : une première basée sur la modélisation multiflots appelée LNS(MMF) et la seconde version utilisant la formalisation indexée dans le temps appelée LNS(MIT).

La première version de la LNS, basée sur la modélisation multiflots, LNS(MMF), est initialisée avec la solution de l'heuristique d'amélioration. Ensuite, cette solution est itérativement détruite et réparée par la méthode. La destruction revient à libérer le routage de certains produits, en les autorisant à être routé via d'autres OT, tout en figeant l'autre partie de la solution. La réparation de cette solution n'est que la résolution du MIP associé contenant des contraintes bloquant les variables de décision associées à la partie de la solution figée. Les résultats obtenus par cette première version de LNS ne sont pas de bonne qualité au vu du temps qu'il lui est donné pour son exécution. Par ailleurs, celle-ci ne trouve pas toujours de solution réalisable pour toutes les instances et elle ne peut être utilisée que sur les instances ayant au plus 50 CT, car le modèle issu de la modélisation multiflots ne peut être chargé par le solveur pour des instances plus grandes (i.e., construit dans la mémoire de l'ordinateur).

Nous exposons ensuite la LNS LNS(MIT) basée sur la formalisation indexée dans le temps. Son déroulement est foncièrement le même que celui de la première LNS, à l'exception de quelques éléments. Tout d'abord, elle est initialisée à l'aide de l'heuristique d'arrondis successifs, ce qui lui procure un premier avantage : la solution initiale est toujours réalisable. Ensuite, la destruction itérative de la solution s'effectue avec deux opérateurs différents, mais tous les deux similaires à celui utilisé par LNS(MMF). La différence est que l'un d'eux libère une plus grande partie de la solution, ceci étant permis par le fait que la modélisation indexée dans le temps est plus compacte que la modélisation multiflots. La réparation de la solution est, comme pour le première version, la résolution d'un MIP contenant des contraintes bloquant une partie des variables de décision représentant la partie figée de la solution. Nos expérimentations montrent que cette méthode domine en tous points la première version de la LNS. En effet, elle permet de résoudre des instances de plus grande taille, tout en étant performante sur les instances de taille moyenne.

Pour finir, nous avons comparé notre meilleure méthode, LNS(MIT), avec la résolution par solveur commercial du modèle indexé dans le temps avec coupes en bornant le temps d'exécution. La LNS et le solveur trouvent quelques solutions optimales, mais la plupart sur des instances de petite taille (30CT). C'est d'ailleurs sur ces instances que le solveur commercial est plus compétitif que notre méthode. En revanche, elle est meilleure pour des instances avec plus de CT. Enfin, pour des instances de tailles représentatives d'organisations réelles (100 CT), le solveur, résolvant le modèle indexé dans le temps, atteint ses limites, tandis que notre LNS les résout en obtenant des solutions de bonne qualité.

Plusieurs pistes sont encore ouvertes pour notre problématique. A court terme, une perspective prometteuse consiste à revoir la conception de LNS(MIT), notre meilleure approche jusqu'à présent, pour tenter d'améliorer son efficacité. Pendant les premières itérations, l'exploration des voisinages en résolvant à l'optimalité des MIP restreints coûte peut-être inutilement du temps de calcul. A la place, nous pourrions ajouter un nouveau voisinage plus rapide. La réparation de la solution s'effectuerait à l'aide d'une heuristique

au lieu d'utiliser la résolution d'un MIP par un solveur commercial. Cet opérateur serait utilisé principalement lors des premières itérations afin d'accélérer la descente de la valeur de la fonction objectif au début de l'exécution de la méthode, mais concevoir une telle technique est loin d'être trivial. Nous avons aussi remarquer que LNS(MIT) pouvait rencontrer quelques soucis à améliorer la valeur de sa fonction objectif lors des dernières itérations. Pour améliorer la qualité de la méthode, un nouveau voisinage pourrait être ajouté, il sélectionnerait prioritairement des zones de la carte où la massification des produits est la plus faible, ainsi la LNS détruirait cette partie de la solution et la réparerait en tentant d'y améliorer la massification. Enfin, dans la même optique d'amélioration des dernières itérations de LNS(MIT), nous pourrions transformer notre méthode en ALNS (Adaptative Large Neighborhhod Search), qui est une méthode qui adapte le choix de son voisinage en fonction de leur réussite dans les itérations précédentes. Cela donnerait plus de liberté à la méthode pour utiliser les bons voisinages aux bons moments.

Une deuxième perspective serait de résoudre notre problème à l'aide de méthodes exactes. Durant ma thèse, nous avons réfléchi à certaines approches telles qu'une génération de colonnes ou un Branch & Bound et quelques réflexions en sont ressorties. La génération de colonnes pourrait poser, entre autres, trois difficultés majeures lors de sa mise en place. La première est de définir ce qu'est exactement une colonne : routage totale d'un produit de son origine vers sa destination, routage d'une unité d'un produit de son origine vers sa destination, routage d'un produit entre deux sites de son routage primaire, etc. La deuxième difficulté est le calcul de la valorisation d'une nouvelle colonne dans le programme secondaire. Enfin, la troisième difficulté est la gestion du grand nombre de colonnes ajoutées au programme maître.

L'utilisation d'une technique de Branch & Bound pour résoudre notre problème amènerait, elle aussi, à quelques difficultés. Une première difficulté est de trouver un schéma de séparation des solutions de bonne qualité. Une deuxième difficulté est de trouver la bonne fonction d'évaluation pour les noeuds.

Ensuite, dans un avenir proche, nous souhaitons étendre le problème étudié en ne considérant non plus un mais plusieurs routages primaires admissibles par produit. Cette extension permettra d'appliquer notre problème à un ensemble de secteurs d'activité plus étoffé que le domaine de la messagerie, comme par exemple le transport de petites marchandises nécessitant une massification entre les demandes. L'impact sur nos modélisations serait que pour la modélisation multiflots, cela ajouterait simplement des nouveaux routages par produit, tandis que pour la seconde formalisation, il suffirait de considérer un nouveau graphe annexe pour chaque nouveau routage. Quant à notre LNS, il faudrait revoir nos voisinages, car ils en seraient fortement agrandis.

Enfin, une perspective intéressante serait d'étendre la définition de notre problème pour y inclure la gestion du repositionnement des véhicules vides. En effet, certaines applications pratiques, comme le Less-Than-Truckload en transport de marchandises, les décisions de repositionnement sont essentielles pour améliorer la consolidation du fret (voir par exemple [7, 16]). Cette fonction peut être inclue dans nos modèles en valorisant les voyages à vides dans la fonction objectif. En ce qui concerne nos approches, notre LNS pourrait se voir ajouter un nouveau voisinage visant principalement à diminuer et/ou raccourcir les voyages à vides.

Bibliographie

[1] Cours transport : 2 - définition des coûts complets de transport. http://www.logistique-1.com/fr/2-formation-logistique-conseil-audit-transport-logistique/news/news-455-cours-transport-gratuit-:-2-definition-des-coûts-complets-de-transport.html - dernière visite le 17/10/2013.

[2] R. K. Ahuja, Ö. Ergun, J. B. Orlin, and A. P. Punnen. A survey of very large-scale neighborhood search techniques. *Discrete Applied Mathematics*, 123(1) :75–102, 2002.

[3] C. Barnhart and S. Shen. Logistics service network design for time-critical delivery. In *Proceedings of the 5th international conference on Practice and Theory of Automated Timetabling*, PATAT'04, pages 86–105, Berlin, Heidelberg, 2005. Springer-Verlag.

[4] T. G. Crainic. Service network design in freight transportation. *European Journal of Operational Research*, 122(2) :272–288, 2000.

[5] T. G. Crainic and K. H. Kim. Chapter 8 intermodal transportation. In C. Barnhart and G. Laporte, editors, *Transportation*, volume 14 of *Handbooks in Operations Research and Management Science*, pages 467 – 537. Elsevier, 2007.

[6] M. Daskin and S. Owen. Location models in transportation. In R. Hall, editor, *Handbook of Transportation Science*, volume 56 of *International Series in Operations Research & Management Science*, pages 321–370. Springer US, 2003.

[7] A. Erera, M. Hewitt, M. Savelsbergh, and Y. Zhang. Improved load plan design through integer programming based local search. *Transportation Science*, 47(3) :412–427, 2013.

[8] L. R. Ford and D. R. Fulkerson. Maximal flow through a network. *Canadian Journal of Mathematics*, 8 :399–404, 1956.

[9] V. Gabrel, A. Knippel, and M. Minoux. Exact solution of multicommodity network optimization problems with general step cost functions. *Operations Research Letters*, 25 :15–23, 1999.

[10] V. Gabrel, A. Knippel, and M. Minoux. A Comparison of Heuristics for the Discrete Cost Multicommodity Network Optimization Problem. *Journal of Heuristics*, 9(5) :429–445, 2003.

[11] V. Gabrel and M. Minoux. Lp relaxations better than convexification for multicommodity network optimization problems with step increasing cost functions. *Acta Mathematica Vietnamica*, 22(1) :123–145, 1997.

[12] B. Gendron, T. Crainic, and A. Frangioni. Multicommodity capacitated network design. In B. Sansò and P. Soriano, editors, *Telecommunications Network Planning*, pages 1–19. Springer US, 1999.

[13] R. E. Gomory and T. C. Hu. Multi-Terminal Network Flows. *Journal of the Society for Industrial and Applied Mathematics*, 9(4) :551–570, 1961.

[14] T. Grünert, H.-J. Sebastian, and M. Thärigen. The design of a letter-mail transportation network by intelligent techniques. In *Proceedings of the Hawaii International Conference on System Sciences*, 1999.

[15] T. Hu. Multi-commodity network flows. *Operations Research*, 11(3) :344–360, 1963.

[16] A. I. Jarrah, E. Johnson, and L. C. Neubert. Large-scale, less-than-truckload service network design. *Operations Research*, 57(3) :609–625, 2009.

[17] N. Katayama, M. Chen, and M. Kubo. A capacity scaling heuristic for the multicommodity capacitated network design problem. *Journal of computational and applied mathematics*, 232(1) :90–101, 2009.

[18] T. L. Magnanti and R. T. Wong. Network design and transportation planning : Models and algorithms. *Transportation Science*, 18(1) :1–55, 1984.

[19] M. Minoux. Networks synthesis and optimum network design problems : Models, solution methods and applications. *Networks*, 19(3) :313–360, 1989.

[20] M. Minoux. Discrete Cost Multicommodity Network Optimization Problems and Exact Solution Methods. *Annals of Operations Research*, 106(1) :19–46, 2001.

[21] S. Ropke and D. Pisinger. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation science*, 40(4) :455–472, 2006.

[22] G. Schrimpf, J. Schneider, H. Stamm-Wilbrandt, and G. Dueck. Record breaking optimization results using the ruin and recreate principle. *Journal of Computational Physics*, 159(2) :139–171, 2000.

[23] P. Shaw. Using constraint programming and local search methods to solve vehicle routing problems. In *Principles and Practice of Constraint Programming—CP98*, pages 417–431. Springer, 1998.

[24] J. Tomlin. Minimum-cost multicommodity network flows. *Operations Research*, 14(1) :44–51, 1966.

l'université nantes angers le mans
PÔLE DE RECHERCHE ET D'ENSEIGNEMENT SUPÉRIEUR

université angers

Thèse de Doctorat

Xavier LIBEAUT

Multiflots dynamiques avec contraintes de synchronisation de ressources

Résumé
Nous étudions un problème de multiflots caractéristique de l'organisation logistique de la messagerie. Ce problème est mis en situation dans la littérature puis formalisé de deux manières, donnant lieu à l'introduction d'un modèle basé sur la notion de multiflots et également d'un modèle indexé dans le temps. Dans un cadre générique applicable à ces deux modélisations, nous proposons des stratégies heuristiques pour résoudre notre problème : heuristique sérielle, technique d'échelonnement de capacité et enfin Large Neighborhood Search. La qualité des solutions obtenues est évaluée sur la base de bornes inférieures que nous calculons. Nous exploitons à cette fin des instances de grandes tailles, représentatives d'organisations réelles, et issues d'un générateur d'instance dédié.

Mots clés
Optimisation, Conception de plan de chargement (Load Plan Design), Multiflots, Modélisation mathématiques, LNS

Abstract
This research focuses on a multi-commodity flow problem arising in courier and small-package delivery services. In the first part, we present the context of the study and review relevant literature. In the second part, we introduce two different mix integer programming models for the problem and discuss valid inequalities and surrogate constraints helpful to straighten the formulations. In the third part, we propose a method to compute a lower bound and 5 heuristic approaches to solve the problem. The latter include tailored implementations of the capacity scaling heuristic and the large neighborhood search framework. In the last part of the manuscript, we first introduce a methodology to generate test instances for the problem. Then, we discuss and analyze extensive computational experiments carefully crafted to assess the performance of the proposed solution methods.

Key Words
Optimization, Load Plan Design, Multi-commodity, Mathematic modelisation, LNS