Université de Montréal

Planification de la récolte et allocation des produits aux usines

par

Géraldine GEMIEUX
Département d'informatique et de recherche opérationnelle
Faculté des arts et des sciences

Mémoire présenté à la Faculté des arts et des sciences
en vue de l'obtention du grade de Maître ès Sciences
en informatique,
option recherche opérationnelle

Août 2009

Université de Montréal

Faculté des arts et des sciences

Ce mémoire intitulé :

Planification de la récolte et allocation des produits aux usines

présenté par :

Géraldine GEMIEUX

a été évalué par un jury composé des personnes suivantes :

Jean-Yves Potvin
président-rapporteur

Bernard Gendron
directeur de recherche

Jacques Ferland
codirecteur

Michel Gendreau
membre du jury

Résumé

L'industrie forestière est un secteur qui, même s'il est en déclin, se trouve au cœur du débat sur la mondialisation et le développement durable. Pour de nombreux pays tels que le Canada, la Suède et le Chili, les objectifs sont de maintenir un secteur florissant sans nuire à l'environnement et en réalisant le caractère fini des ressources. Il devient important d'être compétitif et d'exploiter de manière efficace les territoires forestiers, de la récolte jusqu'à la fabrication des produits aux usines, en passant par le transport, dont les coûts augmentent rapidement.

L'objectif de ce mémoire est de développer un modèle de planification tactique/opérationnelle qui permet d'ordonnancer les activités pour une année de récolte de façon à satisfaire les demandes des usines, sans perdre de vue le transport des quantités récoltées et la gestion des inventaires en usine. L'année se divise en 26 périodes de deux semaines. Nous cherchons à obtenir les horaires et l'affectation des équipes de récolte aux blocs de coupe pour une année.

Le modèle mathématique développé est un problème linéaire mixte en nombres entiers dont la structure est basée sur chaque étape de la chaine d'approvisionnement forestière. Nous choisissons de le résoudre par une méthode exacte, le branch-and-bound. Nous avons pu évaluer combien la résolution directe de notre problème de planification était difficile pour les instances avec un grand nombre de périodes. Cependant l'approche des horizons roulants s'est avérée fructueuse. Grâce à elle en une journée, il est possible de planifier les activités de récolte des blocs pour l'année entière (26 périodes).

Mots Clés : foresterie, recherche opérationnelle, planification de la récolte, chaîne d'approvisionnement, branch-and-bound, horizons roulants

Summary

Forest industry is a sector located at the heart of the debate on globalisation and sustainable development, even if it is in decline. For many countries like Canada, Sweden and Chile, the objectives are to maintain a flourishing sector without damaging the environment and to realize the finite nature of resources. It is important to be competitive and to operate effectively on forest territories, from harvesting to manufacturing products, through transport, in a context where costs increase rapidly. This master's thesis is developing a tactical operational planning model to organize activities for a year to meet requests for factories, without losing sight of the transport of harvested quantities and inventory management factory. The year is divided into 26 periods of two weeks.

We seek harvest teams schedules and assignment to harvest areas (units) for a year. The problem is formulated as a mixed integer programming model, whose structure is based on each stage of the forest supply chain. We choose to solve it by an exact method, branch-and-bound.

We were able to assess how the direct resolution of our planning problem was difficult for instances with a large number of periods. However the rolling horizon approach has proved successful. In a day, we obtained the harvest activities planning for 26 periods.

Keywords: forestry, operational research, harvest planning, supply chain, branch-and-bound, rolling horizons

Table des matières

Liste des figures

Liste des tableaux

Remerciements

Je tiens à remercier mes directeurs Bernard Gendron et Jacques Ferland, qui se sont toujours montrés disponibles et encourageants. L'expérience de chacun et la pertinence de leurs réflexions ont permis à ce projet, après de nombreuses retouches, de prendre une forme définitive plus complète. Grâce à eux, cette première expérience de la recherche a été plus que positive.

Je remercie ma mère et ma sœur, qui malgré la distance, se sont toujours montrées présentes, m'ont soutenue et encouragée à toujours suivre ma voie. Une pensée à mon père, même s'il n'est plus.

Je remercie MITACS et FPInnovations pour leur soutien financier apporté au cours de la réalisation du projet.

Une attention toute spéciale pour Serge Bisaillon, qui, avec son « œil », a permis la correction de plusieurs failles, et a toujours tendu une main secourable lorsque l'implantation devenait une odyssée.

Je n'oublie pas la collaboration de Jean Favreau et Sébastien Lacroix, de FPInnovations, dont les connaissances et l'expérience en foresterie ont été indispensables à la construction du projet, et à la compréhension des enjeux.

Introduction

L'industrie forestière est un secteur qui, même s'il est en déclin, se trouve au cœur du débat sur la mondialisation et le développement durable. Pour de nombreux pays tels que le Canada, la Suède et le Chili, les objectifs sont de maintenir un secteur florissant sans nuire à l'environnement et en réalisant le caractère fini des ressources. Il devient important d'être compétitif et d'exploiter de manière efficace les territoires forestiers, de la récolte jusqu'à la fabrication des produits aux usines, en passant par le transport, dont les coûts augmentent rapidement. De plus, dans l'ensemble des activités de la chaîne d'approvisionnement forestière, il est important d'assurer la préservation de la qualité du bois transporté. La planification forestière couvre de nombreuses activités liées à l'exploitation des forêts : de la sylviculture à la production en usine de pâte et papier, en passant par la récolte et le transport. Des préoccupations écologiques sont apparues au cours des dernières années, et influencent la gestion des activités forestières.

Au cours des dernières décennies, la recherche opérationnelle s'est imposée comme discipline favorisant le développement d'outils de planification. Avec la complexité croissante des problèmes, la recherche opérationnelle s'avère de plus en plus utile, voire indispensable pour assurer une compétitivité et une efficacité accrues. Le grand nombre de variables de décision et de contraintes, ainsi que la nature incertaine des données, sont autant d'éléments qui mettent à l'épreuve la robustesse et la précision des outils de planification.

Dans ce mémoire, nous nous intéressons à la planification de la récolte et de l'allocation des produits aux usines. Ce projet s'inscrit dans un cadre à mi-chemin entre la planification tactique et de la planification opérationnelle. Jusqu'alors la planification de la récolte par le partenaire, est issue d'un travail manuel, sans aucun souci d'optimisation. Le temps nécessaire pour établir le calendrier de récolte pour environ quatre mois prend presqu'un mois. Obtenir un

outil capable de leur fournir un calendrier de récolte optimal en un temps moindre qu'un mois, serait un véritable gain en temps et en argent.

La recherche d'efficacité passe par une organisation de la récolte visant à satisfaire la demande des usines, en passant par une affectation judicieuse des équipes de récolte, tout en tenant compte du transport des produits entre la forêt et les usines, et des niveaux d'inventaires aux usines. La planification doit aussi respecter les accès restreints aux sites de récolte, le caractère limité de l'activité des équipes et les souhaits logistiques de chaque industriel. Nous proposons alors de déterminer les calendriers voulus, en traduisant le problème en un modèle de programmation linéaire mixte en nombres entiers, puis en résolvant ce dernier.

Nous commençons d'abord par une mise en contexte dans le premier chapitre avec une description de la chaîne d'approvisionnement forestière, qui est le fil conducteur du modèle. Les liens entre chaque activité de récolte seront ainsi mis en avant. Par la suite, nous présenterons en détails le modèle correspondant. Dans le chapitre suivant, en route vers la résolution, nous y exposerons les outils mathématiques et les composantes informatiques nécessaires à la résolution. Dans le troisième et dernier chapitre, les résultats obtenus seront résumés.

Chapitre 1 : Description et modélisation du problème

Au cours de ce chapitre, nous précisons le contexte forestier dans lequel s'inscrit notre modèle. Nous commencerons par la définition et la description de la chaîne d'approvisionnement forestière. Avant de décrire le modèle, nous présenterons le problème de façon détaillée. Puis, nous compléterons ce chapitre avec une revue de la littérature pertinente au projet.

1.1 Description de la chaîne d'approvisionnement entre la forêt et les usines

Au cours d'une année, le domaine de la forêt exploitable se divise en deux niveaux d'unités territoriales. Le plus petit est le bloc de coupe. Il s'agit d'une « *superficie déterminée sur le terrain et destinée à être récoltée* » (Côté, 2003). Chaque bloc de coupe est unique, possède ses propriétés en matière d'offre de produits, de volumes et de surface exploitable. Ainsi, la capacité de production d'une équipe de récolte dépendra non seulement du type de matériel utilisé par l'équipe, mais aussi de la taille moyenne des arbres présents sur le bloc. Cela déterminera le nombre de périodes (une période représentant deux semaines) que devra consacrer une équipe à la récolte complète d'un bloc. On notera que la superficie des blocs de coupe est telle qu'une seule équipe de récolte peut effectuer le travail en un petit nombre de périodes (en moyenne deux périodes). En début de planification, les blocs qui sont autorisés à la récolte sont connus, ainsi que leurs propriétés. Le niveau territorial supérieur est le secteur. Il est composé d'un ensemble d'au moins deux blocs de coupe voisins sur le territoire. Pour l'application testée dans ce mémoire, le domaine forestier compte un peu moins de 200 blocs regroupés en une vingtaine de secteurs.

A noter que les déplacements entre blocs d'un même secteur peuvent être considérés comme instantanés pour une planification annuelle puisqu'une période

représente deux semaines et que le déplacement entre blocs est de l'ordre de l'heure. Ainsi le coût associé au déplacement entre blocs est considéré nul. Pour les déplacements entre secteurs, l'utilisation de machinerie lourde et coûteuse est nécessaire, et c'est pour cela qu'il est important de s'assurer qu'une équipe de récolte ne fasse d'aller-retour entre les mêmes secteurs que si elle n'a pas d'autre choix. Ainsi, même si les déplacements entre deux secteurs sont aussi considérés instantanés, ils seront pénalisés dans la fonction objectif.

Chaque bloc de coupe est connecté à un unique chemin. Cependant, un chemin peut desservir plusieurs blocs de coupe. A chaque planification, les chemins existants sont entretenus ou étendus, créant un accès à d'autres blocs. En effet, la récolte d'un bloc de coupe n'est envisageable que si la construction de la route qui le dessert a été réalisée auparavant. Le déboisement précède la construction de chemin. Le plus souvent, déboisement et construction de chemins sont ramenés à une seule activité.

Les extrémités de la chaîne d'approvisionnement se dessinent alors : d'un côté, la source, soit la forêt, représentée par les blocs composant chaque secteur, de l'autre côté, la destination, soit les usines de production.

La chaîne d'approvisionnement entre les blocs et les usines est constituée de cinq activités principales :

- Le déboisement et la construction de chemins;
- La récolte des blocs;
- L'inventaire en forêt, étape transitoire précédant le transport;
- Le transport entre la forêt et l'usine;
- L'inventaire en usine, seconde étape transitoire, avant la livraison des produits finis.

1.2 Description détaillée du problème

L'objectif est de développer un modèle de planification tactique/opérationnelle qui permet d'ordonnancer les activités pour une année de récolte de façon à satisfaire les demandes des usines, sans perdre de vue le transport des quantités récoltées et la gestion des inventaires en usine. L'année se divise en 26 périodes de deux semaines. Nous cherchons à obtenir les horaires et l'affectation des équipes de récolte aux blocs de coupe pour une année.

La détermination des blocs candidats à la récolte au cours de l'année, ainsi que les chemins d'accès aux blocs résulte d'un travail de planification que nous considérons préalable à notre problème. De plus, toute référence à l'activité de transport est à associer à un flot de volumes de bois. Nous ne traitons pas des détails liés au transport de produits comme les horaires, la gestion des camions, les circuits à suivre. De même, le tri des produits à la cour de bois de l'usine principale, lieu de livraison des produits toutes destinations confondues, n'est pas inclus dans le sujet. On s'assure seulement que les volumes livrés dans la cour de bois satisfont la demande des usines.

Les détails opérationnels en forêt comme le déplacement des machines ne sont pas l'objet du mémoire. Tel que mentionné précédemment, le temps de déplacement d'une équipe entre blocs ou entre secteurs est considéré comme instantané. Les déplacements entre secteurs n'interviendront que pour pénaliser les sorties de secteurs et ainsi influencer le modèle de façon à prévenir les allers-retours coûteux.

Autre remarque concernant le transport de bois : on notera l'absence de destination explicite pour chaque volume transporté. En effet, chaque produit est défini de façon à inclure l'usine destinataire. Un produit n'est donc demandé que par une seule usine, ce qui n'empêche pas qu'une usine commande plusieurs produits.

Les contraintes formant le modèle peuvent être regroupées par activité de la chaîne d'approvisionnement. Nous utiliserons cette structure pour introduire et présenter le modèle.

1.2.1 La construction de chemin et la récolte

La partie la plus importante du modèle est associée à la construction de chemins et à la récolte des blocs. Dans notre modèle, ces deux activités sont confondues. Les chemins sont considérés comme des blocs de récolte dont la récolte est prioritaire à celle des blocs réels qu'ils desservent. Les chemins seront aussi appelés « blocs-chemins ». Ainsi, par la suite, nous ne parlerons que d'une seule activité : la récolte.

Par Γ, on désigne l'ensemble des blocs incluant les blocs-chemins; par *AC*, on regroupe les blocs pour lesquels le bloc-chemin qui y mène n'est pas encore construit en début de planification. L'ensemble des secteurs est désigné par *S* et la liste des blocs qui compose chaque secteur *s* est notée *D(s)*.

Au cours d'une période $t \in T$, toute équipe $q \in Q$ affectée à un bloc de récolte $b \in \Gamma$, peut commencer son activité à tout moment au cours de la période. Une fois la récolte commencée, l'activité de q est continue jusqu'à ce que l'offre du bloc soit entièrement récoltée (son offre est épuisée).

Les principales décisions sont prises au niveau du bloc. On constituera le calendrier de récolte grâce aux variables z_{btq}, x_{btq} et y_{btq} qui désignent respectivement l'affectation d'une équipe q à un bloc b à une période t donnée, le début d'activité de l'équipe sur le bloc, et la fin de cette récolte. L'autre niveau de décision est à l'échelle des secteurs avec I_{stq}(indique si une équipe q est présente ou non sur un secteur s à la période t) et $I_{s_1 s_2 q}$ qui indique si une équipe q quitte un secteur s_1 pour le secteur s_2.

1.2.1.1 Restrictions temporelles, spatiales et logistiques

L'affectation d'une équipe dépend de restrictions temporelles spatiales et logistiques. Parmi les contraintes spatiales et temporelles, il y a l'accessibilité des blocs. Les conditions d'accès sont soumises à des autorisations ou à des périodes d'exclusion, dues à la chasse ou à l'impraticabilité des sols à cause du dégel. Ces conditions d'accès varient à chaque période, d'où l'introduction du sous-ensemble Γ_t, qui représente l'ensemble des blocs accessibles à la période *t*. D'autres contraintes d'accès concernent la construction des blocs-chemins préalables à la récolte des blocs dont ils dépendent. Ainsi, au moins deux périodes (un mois) doivent s'écouler entre la fin de la construction d'un chemin et la récolte des blocs qui en dépendent. Au cours de ces deux périodes d'exclusion, le chemin aura le temps de s'assécher et d'être plus apte à la circulation. Autre remarque concernant les blocs-chemins, nous choisissons de forcer leur ouverture dès le début de la planification, afin d'éviter que la limitation d'accès à certains blocs n'entrave la récolte future.

En ce qui concerne les préoccupations logistiques, il est important que le modèle assure que l'équipe affectée à un bloc soit en mesure de récolter les produits présents. Ainsi, pour un bloc b, seulement les équipes $q \in Q_b$ peuvent y être affectées. Notons que dans notre problème, on distingue deux types d'équipes, celles affectées aux bois courts et celles affectées aux bois longs. De plus, pour assurer une certaine fluidité des déplacements et éviter des embouteillages sur le terrain, on limite la présence d'équipes pour chaque unité spatiale (bloc de récolte et secteur) et pour chaque période. Ainsi il y aura au plus une équipe par bloc, par période, et au plus 5 par secteur et par période.

1.2.1.2 Continuité de l'activité de récolte

La récolte d'un bloc est réalisée par une seule et même équipe. L'activité de l'équipe affectée doit être sans interruption jusqu'à ce que l'offre du bloc soit réduite à 0.

Soit O_b l'offre d'un bloc, tous produits confondus, et p_{bq} la quantité en m^3 de bois que peut récolter une équipe q sur un bloc b en une période.

Dénotons par $T_{bq} = \left\lceil \frac{O_b}{p_{bq}} \right\rceil$ le nombre de périodes nécessaires pour récolter complètement le bloc b d'offre globale O_b par l'équipe q, en supposant que l'équipe travaille au maximum de sa capacité de production p_{bq} dès la première période de son affectation. Puisqu'une équipe peut avoir entrepris la récolte de b au cours d'une période où elle terminait la récolte d'un autre bloc, elle n'a peut-être pas consacré le maximum de sa capacité de production p_{bq}pour la récolte de b. Ainsi, en début et en fin de récolte, une équipe ne travaille pas nécessairement au maximum de sa capacité sur ce bloc. Par conséquent, le nombre de périodes au cours desquelles une équipe exploite un bloc b n'est connu qu'à une période près : il s'agit soit de T_{bq} soit de $(T_{bq} + 1)$.

Au cours d'une période t, par la variable α_{btq}, on désigne la portion du temps total de la période durant laquelle l'équipe q exploite le bloc b. Les variables ρ_{btq} et θ_{btq} prennent la même valeur que α_{btq}, respectivement, au cours des périodes où débute et finit la récolte. Elles sont nulles au cours des autres périodes. Ces dernières variables nous seront utiles pour fixer la date de la fin de la récolte du bloc b par l'équipe q et elles permettent de préciser la ou les périodes où se déroulera la partie fractionnaire $\Omega_{bq} = \frac{O_b}{p_{bq}} - T_{bq} + 1$ (en supposant que $T_{bq} \neq \frac{O_b}{p_{bq}}$) du « temps réel » $\frac{O_b}{p_{bq}}$ de l'exploitation du bloc b.

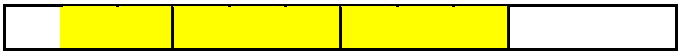
Si la portion de temps consacrée au début de récolte d'un bloc b par l'équipe q à la période t, est plus importante que Ω_{bq}, seules T_{bq} périodes sont nécessaires à la récolte complète du bloc. Ainsi, la date de fin de récolte est $t + T_{bq} - 1$. Sinon, une période supplémentaire sera nécessaire pour que q finisse la récolte de b. Le schéma de la figure 1 illustre les deux situations possibles.

Soient un bloc *b,* une équipe *q*

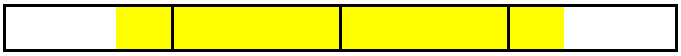
$T_{bq} = \left\lceil \frac{O_b}{p_{bq}} \right\rceil = \lceil 2{,}8 \rceil = 3$ et $\Omega_{bq} = 0{,}8$

On commence la récolte de *b* par *q* à la période *t.* Nous illustrons les deux situations selon lesquelles la durée de travail par période peut se répartir.

Si $\rho_{btq} - \Omega_{bq} > 0$ la date de fin est $t + T_{bq} - 1$, et la durée réelle de travail est $\frac{O_b}{p_{bq}} = 2{,}8$



Sinon la date de fin est à $t + T_{bq}$ et la durée de travail reste inchangée



Les zones colorées désignent des portions de période ou l'équipe q est en activité sur le bloc *b*.

Figure 1 Progression de la récolte et date de fin d'activité

Ainsi, en dehors des périodes de début ou de fin de récolte, et sous réserve qu'elles ne soient ni confondues ni successives, une équipe q consacre tout sont temps à la récolte du bloc b, pour toutes les périodes intermédiaires, entamé afin d'assurer la continuité de l'exploitation du bloc. La production au cours de ces périodes intermédiaires est égale à la capacité de production p_{bq}.

Considérons maintenant l'ensemble des blocs $\Gamma 1\_2_q \subseteq \Gamma$ qui sont récoltés par l'équipe q en au plus 2 périodes. La récolte d'un tel bloc b ne compte pas de périodes intermédiaires (où q consacrerait le maximum de sa capacité de production p_{bq}), et les dates de début et de fin de récolte sont soit confondues dans la même période, soit dans des périodes consécutives. Pour assurer la continuité de l'exploitation de ces blocs, il faut éviter le phénomène que nous qualifions de « parallélisme ». Considérons une équipe q qui travaillerait simultanément sur deux blocs de $\Gamma 1\_2_q$, de façon continue. Nous illustrons dans la figure 2, les différentes situations qui pourraient se présenter lorsque l'équipe q exploite deux blocs de $\Gamma 1\_2_q$ au cours d'une même période (phénomène de parallélisme). Les Cas1A,1B et 2B sont autorisés, mais le Cas2A illustrant le phénomène de parallélisme doit être rejeté, puisqu'au cours d'une période l'équipe q commence l'exploitation d'un bloc sans avoir fini la récolte de l'autre bloc.

Blocs b1,b2 ∈ $\Gamma 1\_2_q$

q une équipe

x	q est affectée au bloc courant

Cas 1A	b1	b2
t	x	x
t+1		x

Cas 1B	b1	b2
t	x	
t+1	x	x

Cas 2A	b1	b2
t	x	x
t+1	x	x

Cas 2B	b1	b2
t		x
t+1	x	x

Situation à écarter

Figure 2 Description du "parallélisme"

1.2.1.3 Les ensembles de périodes T et $\bar{T}$

L'ensemble *T* désigne les périodes de l'horizon de la planification. Les données reliées à la demande ou au niveau d'inventaire, de même que les variables continues associées au flot de bois qui dépendent du temps, sont définis sur l'ensemble *T*. Par contre, pour ce qui est en rapport avec la récolte, l'ensemble des périodes est $\bar{T}$ où $\bar{T} = T \cup \{périodes\ tampons\}$. Cet ensemble $\bar{T}$ a été introduit afin de toujours pouvoir déterminer une date de fin de récolte pour tout bloc dont la récolte a été amorcée à une période quelconque $t \in T$. En effet il peut exister des périodes *t* telles que la date de fin, $t + T_{bq} - 1 \notin T$, mais appartienne à $\bar{T}$. Il est important de souligner qu'aucune activité ne commence au cours des périodes $\bar{T} \backslash T$ c'est-à-dire que $x_{btq} = 0\ \forall t \in \bar{T} \backslash T, \forall b \in \Gamma, \forall q \in Q$. Les seules variables pouvant prendre des valeurs positives désignent une activité qui avait commencé en *T*, et qui se poursuit en dehors de l'horizon de la planification : z_{btq}, y_{btq}, α_{btq}. Le nombre de périodes tampons est égal à la plus grande durée de récolte d'un bloc par l'ensemble des équipes : $|\bar{T} \backslash T| = \max_{b \in \Gamma, q \in Q} T_{bq}$.

1.2.2 L'inventaire en forêt

Les volumes récoltés sont laissés en bordure de chemin. Une partie restera en inventaire forêt et l'autre sera transportée vers l'usine. v_{bkt}^{F} désigne le volume de produit *k* issu du bloc *b* laissé en inventaire forêt à la période *t*, et v_{bkt}^{T}, le volume de produit *k* issu du bloc *b* transporté à la période *t*. τ_{kb} est une fraction de l'offre totale du bloc *b* consacrée au produit *k* : $\tau_{kb} = O_b^k / O_b$. Peu importe l'endroit du bloc où l'équipe récolte au cours d'une période *t,* on estime que la quantité de produit *k* récoltée est égale à τ_{kb} fois le volume récolté. Un pourcentage exact demanderait un travail de repérage, et d'énumération de chaque type de produit sur le terrain, sur chaque surface d'opération, dépendamment de la production de chaque équipe. Cette évaluation est tout simplement trop coûteuse, et superflue, car somme toute, l'ensemble du bloc est récolté, et donc le volume de produit *k* issu du bloc *b* qui sera en bordure de chemin, correspondra à une moyenne.

1.2.3 Le transport

Via le transport, nous visons à satisfaire les demandes à l'usine d_{kt} le plus justement possible. Pour donner une certaine souplesse au problème, nous considérons des variables de pénuries l_{kt}, qui peuvent aussi être considérées comme une quantité de bois qui serait commandée à une source externe. Cette option n'étant pas souhaitable, la variable l_{kt} sera fortement pénalisée dans l'objectif à l'aide de la quantité L_{kt}.

Les produits transportés ne sont pas acheminés directement aux usines. L'usine mère représente une étape transitoire avant la livraison définitive aux usines. À partir de celle-ci, d'autres opérations de tri et d'acheminement auront lieu afin de livrer les produits demandés par les usines. Cependant, ces opérations ne font pas l'objet de ce mémoire. Le modèle assure que la quantité de bois transportée

satisfera les besoins des usines, aussi bien au niveau de la demande, que pour les niveaux d'inventaire minimum à satisfaire.

La quantité de produits transportés en une période entre les blocs et les usines est bornée supérieurement par ξ_b, qui désigne le volume de bois, tous produits confondus, qu'il est possible d'acheminer du bloc *b* vers l'usine mère. Cette borne est calculée en fonction de la capacité des camions et de la distance qui sépare le bloc *b* courant et l'usine mère. ξ_b est d'autant plus grande que le bloc *b* est proche de l'usine mère.

1.2.4 L'inventaire en usine

C'est la dernière activité de la chaîne d'approvisionnement. Une partie du volume transporté est destinée à la consommation des usines, puis le reste constitue l'inventaire en usine. Pour chaque période *t* et pour chaque produit *k*, un niveau d'inventaire minimum est donné: $IMin_{kt}$. Ce seuil correspond à une sécurité pour l'industriel. Il s'agit d'avoir en stock une quantité disponible de produits pour un envoi à la consommation de l'usine, en cas de difficultés diverses au cours du transport ou de la récolte qui entraveraient la satisfaction de la demande des usines.

Si la quantité de produit *k* en inventaire usine excède ce minimum, la quantité en excès est déterminée par la variable continue E_{kt}. Une légère pénalité M_{kt} est associée à cette quantité, qui s'apparente à un coût d'inventaire.

Si le niveau minimum d'inventaire n'est pas respecté, la variable R_{kt} évalue la quantité en défaut. Parce que la situation où une usine manquerait de bois n'est pas envisageable, ronger dans l'inventaire usine sera pénalisé dans l'objectif par $\bar{L}_{kt} > M_{kt}$. Cette pénalité est cependant nettement moins élevée que L_{kt} qui est associée aux volumes de produits commandés, l_{kt}, d'une source extérieure.

1.3 Présentation du modèle

1.3.1 Ensembles, paramètres et variables du modèle

Nous listons ici les ensembles, les paramètres et les variables qui seront utiles pour décrire le modèle de façon détaillée.

Ensembles

$T =$ ensemble des périodes

$\bar{T} = T \cup \{périodes\ tampons\}$

$S =$ ensemble des secteurs

$\Gamma =$ ensemble des blocs de coupe

$D(s) \subset \Gamma =$ ensemble des blocs qui composent le secteur *s*

$C \subset \Gamma =$ ensemble des blocs-chemins

$\Gamma_t =$ ensemble de blocs de coupe auxquels on a accès à la période *t*

$AC =$ ensemble des blocs nécessitant au préalable la récolte d'un bloc-chemin

$Q =$ ensemble des équipes

$Q_b =$ ensemble des équipes pouvant exploiter le bloc *b*

$\Gamma 1\_2_q \subseteq \Gamma$ = ensemble des blocs qui sont récoltés par l'équipe q en au plus 2 périodes

$K =$ ensemble des produits

Paramètres

$O_b =$ offre en m^3 au bloc

$O_b^k =$ offre en m^3 du produit *k* au bloc *b*

$\tau_{kb} =$ fraction de l'offre totale du bloc *b* consacrée au produit *k* (O_b^k / O_b)

$C(b) =$ unique bloc-chemin menant au bloc *b*

$p_{bq} =$ production d'une équipe *q* au bloc *b* par période (m^3/période)

$T_{bq} = \left\lceil \frac{O_b}{p_{bq}} \right\rceil =$ nombre de périodes nécessaires à q pour exploiter entièrement le bloc *b*

$\Omega_{bq} = \frac{O_b}{p_{bq}} - T_{bq} + 1$ partie fractionnaire de la durée de récolte (pour $T_{bq} \neq \frac{O_b}{p_{bq}}$)

$d_{kt} =$ demande en produit *k* à la période $t \in T$ (m^3)

$\xi_b =$ capacité de transport en m^3 du bloc *b* vers l'usine mère (m^3)

$IMin_{kt} =$ inventaire minimum pour le produit *k* à la période *t* (m^3)

$c_b =$ coût de récolte de *b* (\$/m^3)

$c_{bkt}^T =$ coût de transport du produit *k* issu du bloc *b* à la période $t \in T$ (\$/m^3)

$c_{kt}^F =$ coût d'inventaire forêt du produit *k* à la période $t \in T$ (\$/m^3)

$c_{kt}^U =$ coût d'inventaire usine du produit *k* à la période $t \in T$ (\$/m^3)

$c_{s_1 s_2}^D =$ coût de déplacement du secteur s_1 à s_2 (\$)

$\pi_s =$ pénalité associée à la sortie du secteur *s*

$M_{kt} =$ pénalisation de l'excès de livraison du produit *k* à la période $t \in T$(\$/m^3)

$L_{kt} =$ pénalisation de la pénurie de livraison du produit *k* à la période $t \in T$(\$/m^3)

$\bar{L}_{kt} =$ pénalité en cas de volume manquant de k pour satisfaire $IMin_{kt}$ $t \in T(\$/\mathrm{m}^3)$

Variables

$z_{btq} = \begin{cases} 1 \text{ si l'équipe } q \text{ récolte } b \text{ au cours de la période } t \in \bar{T} \\ 0 \text{ sinon} \end{cases}$

x_{btq}

$= \begin{cases} 1 \text{ si l'équipe } q \text{ commence la récolte du bloc } b \text{ au cours de la période } t \in T \\ 0 \text{ sinon} \end{cases}$

y_{btq}

$= \begin{cases} 1 \text{ si l'équipe } q \text{ achève de récolter un bloc } b \text{ au cours de la période } t \in \bar{T} \\ 0 \text{ sinon} \end{cases}$

$I_{stq} = \begin{cases} 1 \text{ si l'équipe } q \text{ est dans le secteur } s \text{ au cours de la période } t \in \bar{T} \\ 0 \text{ sinon} \end{cases}$

$I_{sq} = \begin{cases} 1 \text{ si l'équipe } q \text{ est présente sur le secteur } s \text{ au cours de l'horizon} \\ 0 \text{ sinon} \end{cases}$

$\alpha_{btq} =$ pourcentage de temps à appliquer à la production d'une équipe q au bloc b à la période $t \in \bar{T}$

$\rho_{btq} = \begin{cases} \alpha_{btq} \ si \ s_{btq} = 1 \\ 0 \ sinon \end{cases}$

$\theta_{btq} = \begin{cases} \alpha_{btq} \ si \ f_{btq} = 1 \\ 0 \ sinon \end{cases}$

$v_{bkt}^{F} =$ Volume du produit k issu du bloc b en inventaire forêt à la période $t \in T$

$v_{bkt}^{T} =$ Volume du produit k issu du bloc b transporté à la période $t \in T$

$E_{kt} =$ Volume de produit k en excès d'inventaire à l'usine associée à la période $t \in T$

$l_{kt} =$ Volume de produit k commandé à la période $t \in T$

$R_{kt} =$ Volume de produit k en défaut d'inventaire à la période $t \in T$

v_{kt}^{U} = Volume de produit k en inventaire à son usine à la période $t \in T$

1.3.1.1 Contraintes associées à la récolte

On introduit la contrainte qui spécifie que la récolte des blocs dont le bloc-chemin associé n'a pas encore été construit en début de planification, commence au plus tôt 2 périodes après la fin de la récolte du bloc-chemin associé :

$$x_{btq} \leq \sum_{\bar{q} \in Q} \sum_{\tau=0}^{t-2} f_{c\tau\bar{q}} \quad \forall b \in AC, c = C(b), \forall t \in \bar{T}, \forall q \in Q \quad (1)$$

Pour forcer l'ouverture de chemins, on introduit la contrainte suivante, qui assure que tant qu'il y aura encore un chemin à construire, au moins une équipe sera affectée à l'ouverture d'un des chemins.

$$|C| \sum_{q \in Q} \sum_{c \in C} z_{ctq} \geq |C| - \sum_{q \in Q} \sum_{c \in C} \sum_{\tau=0}^{t} x_{b\tau q} \quad \forall t \in \bar{T} \quad (2)$$

Pour éviter une concentration d'activité dans une même unité spatiale, on limite le nombre d'équipes présentes. Une seule équipe peut travailler sur un bloc de récolte par période :

$$\sum_{q \in Q} z_{btq} \leq 1 \qquad \forall b \in \Gamma, \forall t \in \bar{T} \qquad (3)$$

De même, on limite à 5 le nombre d'équipes travaillant au cours d'une période sur un même secteur :

$$\sum_{b \in s} \sum_{q \in Q} z_{btq} \leq 5 \qquad \forall s \in S, \forall t \in \bar{T} \quad (4)$$

1.3.1.2 Contraintes de récolte et détermination d'une date de fin

Tout d'abord, un bloc ne peut être récolté qu'une seule fois, c'est-à-dire qu'il n'y a qu'une seule date de début et une seule date de fin :

$$\sum_{t\in\bar{T}}\sum_{q\in Q} x_{btq} \leq 1 \qquad \forall b \in \Gamma \qquad (5)$$

$$\sum_{t\in\bar{T}}\sum_{q\in Q} y_{btq} \leq 1 \qquad \forall b \in \Gamma \qquad (6)$$

Une fois la décision prise de commencer la récolte d'un bloc, la date de fin d'activité est connue à une période près (voir la section 1.2.1.2)

$$x_{btq} \leq y_{b(t+T_{bq}-1)q} + y_{b(t+T_{bq})q} \qquad \forall b \in \Gamma_t, \forall t \in \bar{T}, \forall q \in Q \qquad (7)$$

Une remarque est à faire. Dans la contrainte (7), les variables x_{btq} et y_{btq} étant binaires, dans un autre contexte, l'égalité pourrait être acceptée. Ici elle n'est pas envisageable à cause des contraintes (5). Si on avait l'égalité :

$$\sum_{t\in\bar{T}}\sum_{q\in Q} x_{btq} = \sum_{t\in\bar{T}}\sum_{q\in Q} y_{b(t+T_{bq}-1)q} + y_{b(t+T_{bq})q} \leq 1$$

Or une fois la date de fin fixée, à par exemple $t + T_{bq} - 1$, pour $t \geq 1$ il y a nécessairement une apparition de doublons : $y_{b(t+T_{bq}-1)q}$ et $y_{b((t-1)+T_{bq})q}$. Notre modèle deviendrait alors infaisable pour tout $t \geq 1$.

La date de fin de récolte étant liée au temps accordé en début d'activité nous introduisons les contraintes traduisant la définition des variables ρ_{btq} et θ_{btq}.

Par ρ_{btq} on évalue la portion de temps accordée par q lors de sa première affectation au bloc b :

$$\rho_{btq} \leq x_{btq} \qquad \forall t \in \bar{T}, \forall b \in \Gamma_t, \forall q \in Q_b \qquad (8)$$

$$\rho_{btq} \leq \alpha_{btq} \qquad \forall t \in \bar{T}, \forall b \in \Gamma_t, \forall q \in Q_b \qquad (9)$$

Pour forcer l'égalité à tenir entre ρ_{btq} et α_{btq} lors de la première période de la récolte de *b* nous introduisons :

$$\rho_{btq} \geq \alpha_{btq} + x_{btq} - 1 \qquad \forall t \in \bar{T}, \forall b \in \Gamma_t, \forall q \in Q_b \quad (10)$$

θ_{btq} représente la portion de temps accordée par q à sa dernière affectation au bloc *b* :

$$\theta_{btq} \leq y_{btq} \qquad \forall t \in \bar{T}, \forall b \in \Gamma_t, \forall q \in Q_b \qquad (11)$$

$$\theta_{btq} \leq \alpha_{btq} \qquad \forall t \in \bar{T}, \forall b \in \Gamma_t, \forall q \in Q_b \qquad (12)$$

Pour forcer l'égalité à tenir entre θ_{btq} et α_{btq} lors de la dernière période de la récolte de *b* nous introduisons :

$$\theta_{btq} \geq \alpha_{btq} + y_{btq} - 1 \qquad \forall t \in \bar{T}, \forall b \in \Gamma_t, \forall q \in Q_b \quad (13)$$

Les contraintes suivantes visent à déterminer la date de fin de récolte. D'abord les contraintes (14) et (15) influencent le choix de la période de fin selon le signe de la différence entre le temps ρ_{btq} consacré lors de la première période de la récolte d'un bloc par une équipe, et Ω_{bq} la partie fractionnaire du temps réel d'exploitation d'un bloc au complet (voir la figure1 de la section 1.2.1.2).

$$y_{b(t+T_{bq}-1)q} \geq \rho_{btq} - \Omega_{bq} + \theta_{b(t+T_{bq}-1)q} \quad \forall b \in \Gamma_t, \forall t \in \bar{T}, \forall q \in Q \quad (14)$$

$$y_{b(t+T_{bq})q} \geq \theta_{b(t+T_{bq})q}\Omega_{bq} - \rho_{btq} \qquad \forall b \in \Gamma_t, \forall t \in \bar{T}, \ \forall q \in Q \quad (15)$$

Si $\rho_{btq} > \Omega_{bq}$, l'équipe travaille suffisamment longtemps lors de la première période de la récolte, alors celle-ci s'achève en T_{bq} périodes. Seule la contrainte (14) est alors active. Autrement si $\rho_{btq} < \Omega_{bq}$, seule (15) est active et la récolte s'achèvera en $T_{bq} + 1$ périodes.

Ces contraintes ne sont cependant pas suffisantes pour déterminer la période de fin de récolte lorsque $\rho_{btq} = \Omega_{bq}$. Dans ce cas, (14) indique que $f_{b(t+T_{bq}-1)q} \geq 0$, et (15) que $f_{b(t+T_{bq})q}$ est plus grand qu'une quantité négative.

Déterminer la date de fin de récolte revient à connaître la date où l'offre résiduelle du bloc courant est réduite à 0. Par le produit $\alpha_{btq}p_{bp}$ on obtient le volume que l'équipe *q* a récolté sur le bloc *b* au cours de la période *t*. Les contraintes (16) et (17) fonctionnent en tandem pour fixer la date de fin de récolte d'un bloc.

$$1 - \sum_{\tau=1}^{t}\sum_{q\in Q} y_{b\tau q} \leq O_b - \sum_{q\in Q}\sum_{\tau=1}^{t} \alpha_{b\tau q}p_{bp} \quad \forall t \in \bar{T}\,, \forall b \in \Gamma_t \quad (16)$$

$$O_b - \sum_{q\in Q}\sum_{\tau=1}^{t} \alpha_{b\tau q}p_{bp} \leq \left(1 - \sum_{\tau=1}^{t}\sum_{q\in Q} y_{b\tau q}\right) O_b \quad \forall t \in \bar{T}, \forall b \in \Gamma_t \quad (17)$$

Si $f_{btq} = 1$, (16) traduit que la quantité récoltée par l'équipe *q* sur le bloc *b*, jusqu'à la période *t*, n'excède pas l'offre du bloc. (17) précise alors que cette quantité récoltée est exactement l'offre du bloc.

Si aucune date de fin n'a été fixée, (16) indique qu'il reste au moins 1m^3 de bois non récolté sur le bloc *b,* et (17) , que la quantité récoltée sur le bloc jusqu'à la période courante est positive ou nulle.

1.3.1.3 Continuité de l'activité de la récolte d'un bloc

Pour chaque période $t \in \bar{T}$ et pour chaque équipe *q*, la somme des portions de temps allouées à l'ensemble des blocs ne peut excéder 1 :

$$\sum_{b\in\Gamma} \alpha_{btq} \leq 1 \qquad \forall t \in \bar{T}\,, \forall q \in Q \qquad (18)$$

La contrainte (19) ci-dessous force l'équipe qui récolte le bloc *b* à lui consacrer tout son temps, pour peu que la période courante ne soit ni celle du début, ni celle de la fin de la récolte:

$$\alpha_{btq} \geq z_{btq} - x_{btq} - y_{btq} \quad \forall t \in \bar{T}, \forall b \in \Gamma_t, \forall q \in Q_b \qquad (20)$$

(19) assure qu'une équipe en cours de récolte d'un bloc, en dehors des périodes de début et de fin d'activité, ne peut travailler sur un autre bloc au cours de la même période.

Notons que la contrainte (19) combinée avec (5) et (6), assurent que la récolte d'un bloc a lieu sans interruption si $T_{bq} \geq 3$. Par contre, ce n'est pas le cas pour les blocs $b \in \Gamma 1\_2_q$, qui sont récoltés en moins de deux périodes par q. En effet, en se référant à l'analyse de la figure 2 de la section 1.2.1.2, il faut éliminer les situations de « parallélisme » illustrées au cas 2A. Nous avons besoin de la contrainte (20) pour éliminer ces situations.

$$z_{b_1tq} + z_{b_1(t+1)q} + z_{b_2tq} + z_{b_2(t+1)q} \leq 3$$

$$\forall\, b_1, b_2 \neq b_1 \in \Gamma 1_{2_q}, \forall t \in \bar{T}, \forall q \in Q \quad (20)$$

1.3.1.4 Liens entre les variables associées à la récolte

Les autres contraintes liées à la récolte définissent les liens entre les variables. (21) à (23) traduisent qu'une équipe ne peut avoir d'activité sur un bloc que si elle y est affectée. Nous rappelons qu'aucune récolte ne commence au cours des périodes $\bar{T} \backslash T$, les variables x_{btq} sont nulles pour $t \in \bar{T} \backslash T$. Cependant pour alléger l'écriture des prochaines contraintes nous ne ferons référence qu'à l'ensemble $\bar{T}$.

$$\alpha_{btq} \leq z_{btq} \qquad \forall t \in \bar{T}, \forall b \in \Gamma_t, \forall q \in Q_b \quad (21)$$

$$x_{btq} \leq z_{btq} \qquad \forall t \in \bar{T}, \forall b \in \Gamma_t, \forall q \in Q_b \quad (22)$$

$$y_{btq} \leq z_{btq} \qquad \forall t \in \bar{T}, \forall b \in \Gamma_t, \forall q \in Q_b \quad (23)$$

Le modèle assure aussi qu'une affectation entraîne nécessairement une activité à la même période :

$$\alpha_{btq} p_{bp} \geq z_{btq} \qquad \forall t \in \bar{T}, \forall b \in \Gamma_{t,} \forall q \in Q_b \quad (24)$$

Les contraintes (25) et (26) lient respectivement les variables z_{btq} et x_{btq}, ainsi que z_{btq} et y_{btq}. Ainsi, $x_{btq} = 1$ si *t* est la première fois où *q* est affectée au bloc *b* :

$$x_{btq} \geq z_{btq} - z_{b(t-1)q} \quad \forall t \geq 2\ t \in \bar{T}\ , \forall b \in \Gamma_t\ , \forall q \in Q_b \qquad (25)$$

De même $y_{btq} = 1$ si *t* est la dernière période où *q* est affectée au bloc *b* :

$$y_{btq} \geq z_{btq} - z_{b(t+1)q} \qquad \forall t \leq |\bar{T}| - 1, \forall b \in \Gamma_t\ , \forall q \in Q_b \quad (26)$$

Pour accélérer la résolution on introduit la contrainte (27) qui fixe à 0 la valeur des variables z_{btq} inappropriées, une fois que la période de début de récolte de *b* est fixée :

$$z_{b\tau q} \leq 1 - x_{btq} \qquad \forall\ \tau \geq t + T_{bq} + 1 \quad \forall b \in \Gamma_t\ , \forall q \in Q_b \qquad (27)$$

1.3.1.5 Au niveau du secteur

Revenons maintenant aux décisions prises à l'échelle du secteur. Il n'y a pas de contraintes explicites concernant les activités au niveau du secteur, sinon que les déplacements entre secteurs soient réduits afin d'éviter des allers-retours coûteux. En effet, à chaque déplacement d'un secteur à un autre, il faut déplacer des machineries lourdes et coûteuses. Nous allons donc introduire un terme dans la fonction économique pour réduire le nombre de secteurs où l'équipe peut récolter.

La relation (28) indique dans quel secteur se trouve une équipe en activité à la période t:

$$x_{btq} \leq I_{stq} \qquad \forall s \in S, \forall b\ \in D(s), \forall t \in T, \forall q \in Q \qquad (28)$$

Pour définir le terme de la fonction économique, introduisons la variable d'indication I_{sq}qui prend la valeur 1 si l'équipe *q* récolte sur le secteur *s* au cours de l'horizon de la planification.

$$I_{stq} \leq I_{sq}\ \forall s \in S, \forall t \in T, \forall q \in Q \quad (29)$$

À I_{sq} est associée la pénalité π_s, qui est une moyenne des coûts de déplacement d'un secteur à un autre. Ainsi le terme $\sum_{s\in S}\pi_s \sum_{q\in q} I_{sq}$est utilisé pour simuler l'éparpillement des équipes puisque ce terme augmente quand les équipes récoltent sur un grand nombre de secteurs au cours de l'horizon.

1.3.2 Contraintes associées à l'inventaire en forêt

La quantité qui reste en inventaire en forêt est la différence entre l'inventaire passé, la production issue de la récolte, et les volumes transportés vers les usines.

$$v^F_{bkt} = v^F_{b,k,t-1} + \left(\sum_{q\in Q} \alpha_{btq}\, p_{bq}\right)\tau_{kb} - v^T_{bkt} \quad \forall b \in \Gamma, \forall t \in T, \forall k \in K \quad (30)$$

1.3.3 Contraintes associées au transport

Même si le modèle ne traite pas des détails opérationnels du transport, il faut tout de même s'assurer d'une circulation réaliste du flot de bois entre la forêt et l'usine. Pour cette raison nous introduisons une capacité de transport ξ_b calculée de façon à refléter aussi bien la distance qui sépare la forêt de l'usine mère, que la capacité moyenne d'une flotte de camions effectuant les livraisons au cours d'une période.

$$\sum_{k\in K} v^T_{bkt} \leq \xi_b \qquad \forall b \in \Gamma, \forall t \in T \quad (31)$$

1.3.4 Contraintes associées à l'inventaire en usine

La relation (32) traduit la conservation de flot en inventaire usine. Le niveau actuel est la somme des quantités en inventaire de la période précédente et de celles issues de la livraison des produits, à laquelle on soustrait la demande de

l'usine. Au besoin, l'éventuelle commande de produits à une source extérieure peut être requise.

$$v_{kt}^{U} = v_{k,t-1}^{U} + \sum_{b \in \Gamma} v_{bkt}^{T} - d_{kt} + l_{kt} \quad \forall t \in T \ \forall k \in K \qquad (32)$$

Les relations (33) et (34) quantifient les variations des quantités en inventaire par rapport au seuil minimum $IMin_{kt}$.

$$E_{kt} \geq v_{kt}^{U} - IMin_{kt} \quad \forall t \in T, \forall k \in K \ (33)$$

$$R_{kt} \geq IMin_{kt} - v_{kt}^{U} \quad \forall t \in T, \forall k \in K \ (34)$$

1.3.5 Fonction objectif

A chaque activité de la chaîne d'approvisionnement est associé un coût : un coût de récolte dépendant du bloc et de la période c_{bt}, un coût de transport c_{bkt}^{T} associé au bloc d'origine, au produit et à la période, des coûts d'inventaire forêt c_{kt}^{F} et usine c_{kt}^{U}, qui varient en fonction du produit et des saisons. Des pénalisations sont aussi présentes dans l'objectif, une première pour limiter les déplacements entre secteurs, une seconde pour pénaliser l'excès de livraison M_{kt}, qui s'apparente au coût d'inventaire c_{kt}^{U}. Selon les produits et les saisons, la pénalisation des volumes en excès sera d'autant plus forte, que la dégradation des produits sera importante. Enfin, l'avant-dernier coût est la pénalisation en cas de pénurie L_{kt}: les volumes amenés en usine sont inférieurs à la demande, il faut donc commander la quantité manquante. Cette situation est en réalité peu souhaitable, les coûts de pénalisations sont donc d'un ordre beaucoup plus grand que les valeurs des autres coûts. La dernière pénalité est associée aux volumes R_{kt} qui manquent pour satisfaire le seuil minimum en inventaire usine.

$$\text{Min} \sum_{b \in \Gamma} c_b \sum_{q \in Q} p_{bq} \sum_{t \in T} \alpha_{btq} + \sum_{t \in T} \sum_{b \in} \sum_{k \in K} c_{bkt}^{T} v_{bkt}^{T} + \sum_{t \in T} \sum_{k \in K} c_{kt}^{F} \sum_{b \in \Gamma} v_{bkt}^{F}$$

$$+ \sum_{t \in T} \sum_{k \in K} c_{kt}^{U} v_{kt}^{U} + \sum_{s \in S} \pi_s \sum_{q \in Q} I_{sq} + \sum_{k \in K} \sum_{t \in T} M_{kt} E_{kt}$$

$$+\sum_{k\in K}\sum_{t\in T} L_{kt} l_{kt} + \sum_{k\in K}\sum_{t\in T} \bar{L}_{kt} R_{kt}$$

Le modèle peut donc se résumer comme suit.

1.3.6 Récapitulatif du modèle

$$\text{Min} \sum_{b\in\Gamma} c_b \sum_{q\in Q} p_{bq} \sum_{t\in\bar{T}} \alpha_{btq} + \sum_{t\in T}\sum_{b\in}\sum_{k\in K} c^T_{bkt} v^T_{bkt} + \sum_{t\in T}\sum_{k\in K} c^F_{kt} \sum_{b\in\Gamma} v^F_{bkt}$$

$$+\sum_{t\in T}\sum_{k\in K} c^U_{kt} v^U_{kt} + \sum_{s\in S} \pi_s \sum_{q\in Q} I_{sq} + \sum_{k\in K}\sum_{t\in T} M_{kt} E_{kt}$$

$$+\sum_{k\in K}\sum_{t\in T} L_{kt} l_{kt} + \sum_{k\in K}\sum_{t\in T} \bar{L}_{kt} R_{kt}$$

Sujet à

Contraintes :

Récolte

$$x_{btq} \le \sum_{\bar{q}\in Q}\sum_{\tau=0}^{t-2} y_{c\tau\bar{q}} \quad \forall b \in AC\,, c = C(b)\,, \forall t \in \bar{T}\,, \forall q \in Q \quad (1)$$

$$|C| \sum_{q\in Q}\sum_{c\in C} z_{ctq} \ge |C| - \sum_{q\in Q}\sum_{c\in C}\sum_{\tau=0}^{t} x_{b\tau q} \quad \forall t \in \bar{T} \quad (2)$$

$$\sum_{q\in Q} z_{btq} \le 1 \qquad \forall b \in \Gamma, \forall t \in \bar{T} \qquad (3)$$

$$\sum_{b\in s}\sum_{q\in Q} z_{btq} \le 5 \qquad \forall s \in S\,, \forall t \in \bar{T} \qquad (4)$$

$$\sum_{t\in\bar{T}} \sum_{q\in Q} x_{btq} \le 1 \qquad \forall b \in \Gamma \qquad (5)$$

$$\sum_{t\in\bar{T}} \sum_{q\in Q} y_{btq} \le 1 \qquad \forall b \in \Gamma \qquad (6)$$

$$x_{btq} \le y_{b(t+T_{bq}-1)q} + y_{b(t+T_{bq})q} \quad \forall b \in \Gamma_t, \forall t \in \bar{T}\,, \forall q \in Q \quad (7)$$

$$\rho_{btq} \leq x_{btq} \qquad \forall t \in \bar{T},\ \forall b \in \Gamma_t\ , \forall q \in Q_b \qquad (8)$$

$$\rho_{btq} \leq \alpha_{btq} \qquad \forall t \in \bar{T}, \forall b \in \Gamma_t, \forall q \in Q_b \qquad (9)$$

$$\rho_{btq} \geq \alpha_{btq} + x_{btq} - 1 \qquad \forall t \in \bar{T}\ , \forall b \in \Gamma_t\ , \forall q \in Q_b \quad (10)$$

$$\theta_{btq} \leq y_{btq} \qquad \forall t \in \bar{T},\ \forall b \in \Gamma_t\ , \forall q \in Q_b \qquad (11)$$

$$\theta_{btq} \leq \alpha_{btq} \qquad \forall t \in \bar{T}, \forall b \in \Gamma_t\ , \forall q \in Q_b \qquad (12)$$

$$\theta_{btq} \geq \alpha_{btq} + y_{btq} - 1 \qquad \forall t \in \bar{T}, \forall b \in \Gamma_t\ , \forall q \in Q_b \quad (13)$$

$$y_{b(t+T_{bq}-1)q} \geq \rho_{btq} - \Omega_{bq} + \theta_{b(t+T_{bq}-1)q} \qquad \forall b \in \Gamma_t, \forall t \in \bar{T}\ , \forall q \in Q \qquad (14)$$

$$y_{b(t+T_{bq})q} \geq \theta_{b(t+T_{bq})q}\Omega_{bq} - \rho_{btq} \qquad \forall b \in \Gamma_t\ , \forall t \in \bar{T},\ \forall q \in Q \qquad (15)$$

$$1 - \sum_{\tau=1}^{t}\sum_{q \in Q} y_{b\tau q} \leq O_b - \sum_{q \in Q}\sum_{\tau=1}^{t} \alpha_{b\tau q} p_{bp} \quad \forall t \in \bar{T}\ \ \forall b \in \Gamma_t \quad (16)$$

$$O_b - \sum_{q \in Q}\sum_{\tau=1}^{t} \alpha_{b\tau q} p_{bp} \leq \left(1 - \sum_{\tau=1}^{t}\sum_{q \in Q} y_{b\tau q}\right) O_b \quad \forall t \in \bar{T}\ , \forall b \in \Gamma_t \quad (17)$$

$$\sum_{b \in \Gamma} \alpha_{btq} \leq 1 \qquad \forall t \in \bar{T}\ , \forall q \in Q \qquad (18)$$

$$\alpha_{btq} \geq z_{btq} - x_{btq} - y_{btq} \qquad \forall t \in \bar{T},\ \forall b \in \Gamma_t\ , \forall q \in Q_b \qquad (19)$$

$$z_{b_1 tq} + z_{b_1(t+1)q} + z_{b_2 tq} + z_{b_2(t+1)q} \leq 3 \quad \forall\ b_1, b_2 \neq b_1 \in\ \Gamma 1\_2_q,$$
$$\forall t \in \bar{T}\ \forall q \in Q \quad (20)$$

$$\alpha_{btq} \leq z_{btq} \qquad \forall t \in \bar{T},\ \forall b \in \Gamma_t\ , \forall q \in Q_b \quad (21)$$

$$x_{btq} \leq z_{btq} \qquad \forall t \in \bar{T}\ , \forall b \in \Gamma_t\ , \forall q \in Q_b \quad (22)$$

$$y_{btq} \leq z_{btq} \qquad \forall t \in \bar{T}\ , \forall b \in \Gamma_t\ , \forall q \in Q_b \quad (23)$$

$$\alpha_{btq} p_{bp} \geq z_{btq} \qquad \forall t \in \bar{T}\ , \forall b \in \Gamma_t\ , \forall q \in Q_b \quad (24)$$

$$x_{btq} \geq z_{btq} - z_{b(t-1)q} \quad \forall t \geq 2\ t \in \bar{T}\ , \forall b \in \Gamma_t\ , \forall q \in Q_b \quad (25)$$

$$y_{btq} \geq z_{btq} - z_{b(t+1)q} \quad \forall t \leq |\bar{T}| - 1, \forall b \in \Gamma_t\ , \forall q \in Q_b \quad (26)$$

$$z_{b\tau q} \leq 1 - x_{btq} \quad \forall\ \tau \geq t + T_{bq} + 1 \quad \forall b \in \Gamma_t\ \forall q \in Q_b \quad (27)$$

$$x_{btq} \leq I_{stq} \quad \forall s \in S\ \forall b\ \in D(s) \forall t \in \bar{T}\ \forall q \in Q \quad (28)$$

$$I_{stq} \leq I_{sq}\ \forall s \in S, \forall t \in T, \forall q \in Q \quad (29)$$

Inventaire en forêt

$$v^F_{bkt} = v^F_{b,k,t-1} + \left(\sum_{q \in Q} \alpha_{btq}\, p_{bq}\right) \tau_{kb} - v^T_{bkt} \quad \forall b \in \Gamma, \forall t \in T, \forall k \in K \quad (30)$$

Transport

$$\sum_{k \in K} v^T_{bkt} \leq \xi_b \quad \forall b \in \Gamma\ , \forall t \in T \quad (31)$$

Inventaire en usine

$$v^U_{kt} = v^U_{k,t-1} + \sum_{b \in \Gamma} v^T_{bkt} - d_{kt} + l_{kt} \quad \forall t \in T\ , \forall k \in K \quad (32)$$

$$E_{kt} \geq v^U_{kt} - IMin_{kt} \quad \forall t \in T, \forall k \in K \quad (33)$$

$$R_{kt} \geq IMin_{kt} - v^U_{kt} \quad \forall t \in T, \forall k \in K \quad (34)$$

1.3.7 Variante du modèle

Les contraintes (20) qui visent à écarter les cas où une équipe travaillerait sur deux blocs en même temps de façon continue augmentent considérablement la taille du problème.

$$z_{b_1tq} + z_{b_1(t+1)q} + z_{b_2tq} + z_{b_2(t+1)q} \leq 3$$

$$\forall\, b_1, b_2 \neq b_1 \in \Gamma 1\_2_q\ \forall t \in \bar{T}\ \forall q \in Q \quad (20)$$

En effet, cette contrainte s'applique à toutes les paires de blocs dans $\Gamma 1\_2_q$, par période et par équipe. Avec une autre formulation nous cherchons à ne plus utiliser des couples de blocs et ainsi réduire la taille du problème. Au lieu d'introduire (20), on a :

$$1 - z_{btq} + f_{btq} \geq \sum_{\bar{b} \neq b} x_{\bar{b}tq} \quad \forall b \in \Gamma 1\_2_q, \forall t \in \bar{T}\,, \forall q \in Q \quad (20B)$$

(20B) traduit qu'une équipe en activité sur un bloc *b* ne peut commencer la récolte sur un autre bloc que si elle est à la période de fin d'activité sur *b*.

1.4 Revue de la littérature

L'objet de cette section est de présenter les travaux qui ont été les plus utiles à ce mémoire. Les articles suivants nous ont permis une familiarisation avec la planification forestière, la place de l'optimisation dans le domaine de la foresterie, et ont permis d'appréhender certaines difficultés liées à la nature des modèles utilisés. Cette revue de la littérature ne se veut pas exhaustive, mais suffisante pour situer notre projet parmi certaines recherches qui ont été faites au Canada, au Chili et en Suède.

1.4.1 L'optimisation en foresterie

Pour un domaine aussi vaste que la foresterie, il est intéressant de connaître les contextes et les degrés d'implication de la recherche opérationnelle. De nombreuses activités qui composent l'exploitation forestière comme la sylviculture, la construction des routes, la coupe du bois en forêt ou en usine, la récolte, les tournées de véhicules etc., ont fait l'objet de recherches scientifiques. Avec ces articles, il est possible de parcourir les grandes lignes de ce qui a été développé pour les principales activités forestières.

Rönnqvist (2003) décrit le flot de bois au cours de la planification forestière et une variété de problèmes associés à la planification à des niveaux différents. Chaque élément de la chaîne d'approvisionnement bénéficie d'une description détaillée. L'horizon de planification se distingue selon les niveaux : stratégique (entre 5 et 100 ans), tactique (entre 1 et 5 ans) et opérationnelle (d'un jour à un an). L'auteur présente des exemples de problèmes concrets pour chaque niveau de planification, et le type de modèle utilisé. Par exemple, dans le cas d'un problème de développement régional, pour un horizon de 100 ans, le modèle est un problème linéaire. Pour la décision de coupes sur le terrain, qui fait partie de la planification opérationnelle dite à très court terme, l'outil d'optimisation à bord des machines de récolte relève de la programmation dynamique. Pour des problèmes de planification tactique ou opérationnelle à court terme (6 mois à un an) comme celui de notre mémoire, les modèles sont souvent linéaires mixtes.

Un handicap commun à ces problèmes est leur grande dimension. Une résolution explicite est sinon difficile, trop coûteuse en temps pour être pratique et, par conséquent, encourage l'utilisation d'heuristiques et de métaheuristiques.

Epstein (1999) parcourt les différents systèmes d'aide à la décision implantés dans le secteur forestier chilien. Il insiste sur les gains en efficacité, les économies réalisées et la génération d'emplois dans le domaine. Il situe en effet les besoins qui ont motivé le développement de chacun des outils, et il en décrit le fonctionnement dans les grandes lignes. Nous retiendrons l'outil OPTICORT, qui

est un système d'aide à la décision pour la récolte à court terme (3 mois), parce qu'il se rapproche de notre problème. La planification y est cependant plus détaillée et s'appuie sur les décisions de récolte et de transport. En effet, les décisions principales du modèle déterminent:

- Parmi les blocs dont les arbres sont à maturité pour la récolte, quels sont ceux qui seront récoltés?
- Pour chaque opération de coupe, quel type de machine doit être utilisé?
- Quelle quantité faut-il récolter par semaine, selon quel plan de coupe?
- Quel produit doit être livré pour satisfaire la demande des usines?

Nous précisons que la gestion des inventaires n'est pas incluse. OPTICORT est basée sur un modèle linéaire qui est résolu par génération de colonnes.

1.4.2 La planification des activités de récolte pour un horizon d'au plus un an

Les problèmes de cette catégorie appartiennent tous au même niveau de planification : la planification opérationnelle. On distinguera ceux à court terme (6 mois à 1 an) de ceux à très court terme (moins de 6 mois).

Karlsonn, Rönnqvist et Bergstrom (2004) abordent un problème de planification annuelle de la récolte du point de vue des compagnies suédoises. Les objectifs de cet article sont très proches de ceux que l'on cherche à atteindre dans ce mémoire. Il s'agit de déterminer quels blocs de coupe seront récoltés durant l'année pour satisfaire la demande des usines. De plus, il faut affecter les équipes à chaque bloc, et gérer les volumes en inventaire. Contrairement à notre projet, la gestion des chemins, de l'ouverture à la fermeture en passant par la maintenance, fait partie des décisions qui doivent être prises. Si les objectifs de ce problème et celui de ce mémoire se ressemblent, les modélisations sont clairement distinctes.

Une différence majeure vient du fait que dans cet article, la durée de récolte des blocs dure au plus une période. Cependant, une souplesse est permise grâce à une variable de pourcentage, qui évalue à quel point un bloc est récolté; rendant alors possible que la récolte d'un bloc commence à la période *t*, et se

termine au début de la suivante. Dans notre modèle, la durée de récolte varie non seulement avec la surface des blocs, mais aussi avec la productivité des équipes et peut durer plus de deux périodes.

Même si les modèles sont distincts, des ressemblances apparaissent au niveau des variables pour l'affectation des blocs, ou pour suivre l'évolution de la récolte. Cependant, la progression de la récolte procède différemment car elle se mesure par rapport au temps d'activité dans notre mémoire, tandis que dans cet article, c'est la surface exploitée qui est l'indicateur. Dans notre modèle avec la variable α_{btq}, on quantifie quelle portion d'unité de temps t l'équipe q consacre à la récolte de *b,* dans cet article la variable de pourcentage est associée à la portion de surface du bloc qui a été exploitée.

En ce qui concerne la résolution, une première étape était d'utiliser CPLEX, qui n'a le plus souvent pu donner qu'une solution non optimale après plus d'un jour d'exécution. Ils proposent alors une heuristique dont le comportement rappelle le Branch-and-Bound, où les règles de branchement sont basées sur la récolte.

Dans le mémoire de Lacroix (2005), nous trouvons en détails la description de la chaîne d'approvisionnement et les difficultés rencontrées sur le terrain. En particulier, on retrouve certains éléments reliés au type de machines utilisées, ainsi qu'aux risques de dégâts au sol. Même si ces éléments ne se retrouvent pas dans notre mémoire, il n'en reste pas moins que ce mémoire a été très utile pour établir le squelette de notre modèle. Il a permis aussi de compléter la mise en contexte. En ce qui concerne les modèles, ils sont très distincts, notamment parce qu'ici, nous travaillons avec des variables entières et continues, tandis que celui de Lacroix ne fait appel qu'à des variables continues. L'échelle de travail n'est pas non plus la même car nous travaillons au niveau des blocs, et non des secteurs, ce qui augmente la dimension et la difficulté de notre modèle. De plus, pour faciliter la résolution du problème, il fait appel à une seconde étape de simulation exploitant les résultats de la première résolution linéaire, afin de permettre un suivi des activités (transport des produits, débardage et consommation des usines). Dans notre projet, il s'agit de faire de la première étape (résolution d'un problème

linéaire en nombre entiers) la seule étape d'aide à la décision. Nous finirons en notant que l'outil d'optimisation utilisé par Sébastien Lacroix pour résoudre son modèle linéaire est *What's best*, outil complémentaire à Excel.

Finalement, Karlsonn, Rönnqvist et Bergstrom (2003) fournissent un projet de planification opérationnelle qui lie ce niveau de planification à celui qui lui est supérieur (planification tactique). Pour une période de 3 à 6 semaines, l'objectif est de choisir parmi un ensemble de calendriers, celui à affecter à chaque équipe. Un coût est associé à chaque calendrier, incluant la récolte, les déplacements entre secteurs, le transport et les inventaires. Aux contraintes usuelles de transport et d'inventaire, s'ajoutent celles traduisant l'entretien et la gestion des routes.

Chapitre 2 : Méthode de résolution

Notre modèle étant un programme linéaire mixte en nombre entiers, dans ce chapitre, nous nous intéresserons d'abord aux grandes lignes de ce type de modèle de programmation mathématique. Puis, nous décrirons l'algorithme de branch-and-bound qui est un outil courant pour la résolution de problèmes linéaires en nombres entiers.

2.1 Survol de la programmation linéaire en nombres entiers

Nous nous inspirons du chapitre 11.5 d'Hillier et Lieberman (2004, pp. 501-503) donnant quelques pistes sur la résolution de problème linéaire mixte en nombre entiers pour construire cette section.

Généralement, les problèmes linéaires sont plus faciles à résoudre que les problèmes contenant des variables binaires ou entières. Cependant, ces deux domaines de la programmation mathématique ne sont pas disjoints; en effet la plupart des algorithmes pour la programmation en nombres entiers exploitent la méthode du simplexe ou la méthode duale du simplexe pour résoudre les problèmes linéaires associés où seules les contraintes d'intégralité ne sont pas incluses. Un tel problème linéaire associé est connu sous le nom de « problème relaxé ». L'ensemble réalisable du problème relaxé contient celui du problème original. Ainsi, une solution optimale du problème relaxé, n'est généralement pas réalisable pour le problème en nombres entiers, car certaines variables entières prennent des valeurs fractionnaires dans la solution optimale du problème relaxé.

Il existe des cas où résoudre le problème en nombres entiers est aussi facile que de résoudre le problème relaxé. C'est le cas lorsque la solution du problème relaxé est aussi celle du problème original. Par exemple, lorsque la matrice des contraintes d'un problème en nombres entiers est unimodulaire (le déterminant de toute sous-matrice carrée est 1, 0 ou -1), une solution du problème relaxé est entière. Cette propriété est liée au type du problème. Par exemple, la structure des problèmes de flots à coût minimum avec des paramètres entiers, de même que les

problèmes « dérivés » (problèmes de transport, d'affectation, du plus court chemin, et flot maximum) permet d'assurer que la solution du problème relaxé est une solution optimale du problème en nombres entiers dont il est issu.

La seule présence de variables entières n'est pas suffisante pour évaluer la difficulté d'un problème. En effet, trois autres éléments doivent être pris en compte:

a) le nombre de variables entières;

b) la distinction entre variables binaires et entières;

c) la structure du problème.

Contrairement à la programmation linéaire, le nombre de contraintes n'est que d'une relative importance par rapport au nombre de variables entières. Il arrive même qu'augmenter le nombre de contrainte diminue le temps de calcul, le domaine réalisable diminuant en taille.

L'algorithme du branch-and-bound est la façon la plus commune de résoudre des problèmes en nombres entiers. Cette méthode de résolution est utilisée dans la plupart des logiciels de résolution. Certaines variations sont apportées à l'algorithme de façon à améliorer le temps d'exécution. Nous en verrons quelques aspects dans les deux sections suivantes.

2.2 Algorithme de branch-and-bound

Pour cette section, nous nous inspirons des articles de Atamtürk (2005), et Linderoth (2005) ainsi que du chapitre 7 du livre de Wolsey (1998, pp. 100-107) sur le branch-and-bound.

Considérons le problème de programmation linéaire mixte en nombres entiers suivant noté (PLE) :

$$Min\ c^T x + d^t y$$

Sujet à : $Ax + Gy \leq b$

$$x \in \mathbb{Z}^n, y \in \mathbb{R}^m$$

où *c, d*, *A* et *G* sont des matrices rationnelles avec des dimensions appropriées.

On note :

- *F(P)* le domaine réalisable d'un problème *P* quelconque;
- $v(P)$ la valeur optimale d'un problème *P* quelconque;
- $\bar{P}$ le problème relaxé associé à un problème *P*

On suppose que *F(PLE)* est borné et non vide.

2.2.1 Principe général

L'algorithme du branch-and-bound suit le principe de « Diviser pour régner » en décomposant *F(PLE)* en plusieurs sous-régions, qui seront chacune explorées récursivement. La figure 3 résume l'algorithme de branch-and-bound. Il est courant et pratique de concevoir l'algorithme du branch-and-bound comme un arbre de recherche. A chacune des sous-régions du domaine réalisable, est associé un sous-problème *PLE(k)* qui correspond à un sommet de l'arbre des solutions. La racine de l'arbre fait référence au problème original *PLE* ou encore *PLE(0)*. On résout alors chaque problème relaxé $\overline{PLE(k)}$.

Notons que $F(PLE(k)) \subseteq F(\overline{PLE(k)})$. Par conséquent, on a la propriété suivante : $v(\overline{PLE(k)}) \leq v\big(PLE(k)\big)$, ce qui signifie que résoudre le problème relaxé d'un problème en nombres entiers, fournit une borne inférieure à la valeur optimale du problème original.

On peut borner supérieurement $v(PLE)$ par $v(PLE(k))$. En effet, toute solution réalisable entière du problème original fournit une borne supérieure à la solution optimale.

Au cours du branch-and-bound, parmi les solutions réalisables entières trouvées, celle qui donne la plus petite valeur d'objectif est nommée « solution candidate ». Elle est notée (x^*, y^*) et sa valeur est z^*.

On définit l'ensemble L de sous-problèmes dits non élagués ou actifs. Un problème est dit élagué s'il vérifie une des trois conditions suivantes :

1. $v\big(\overline{PLE(k)}\big) \geq z^*$, car alors la solution entière optimale correspondante ne peut améliorer la solution candidate;
2. Le problème relaxé est non réalisable : $F\big(\overline{PLE(k)}\big) = \emptyset$;
3. La solution $(\bar{x}, \bar{y})$du problème relaxé $\overline{PLE(k)}$ est entière. Dans ce cas, on vérifie si la solution candidate peut être améliorée : Si $v\big(\overline{PLE(k)}\big) \leq z^*$ alors $(x^*, y^*) \leftarrow (\bar{x}, \bar{y})$ et $z^* \leftarrow v\big(\overline{PLE(k)}\big)$.

Pour les problèmes non élagués $(PLE(k))$, nous procédons au branchement pour le subdiviser en plusieurs sous-problèmes $PLE(k(i))$ qui sont ajoutés à L. Notons que $PLE(k) = \bigcup_{i=1}^{q} PLE(k(i))$ et $\bigcap_{i=1}^{q} PLE\big(k(i)\big) = \emptyset$.

L'élagage assure une amélioration dans la résolution, en évitant de perdre du temps dans le parcours de l'arbre, là où aucune amélioration de l'objectif ne peut être attendue.

Pour augmenter l'efficacité de l'algorithme de branch-and-bound, les efforts se concentrent donc sur la vitesse à laquelle l'écart entre les bornes inférieure et supérieure se réduit. Décroître la borne supérieure ne peut se réaliser qu'en trouvant une meilleure solution réalisable entière durant la recherche.

L'augmentation de la borne inférieure est liée au choix du nœud. Nous verrons quelques méthodes qui permettent une augmentation de l'efficacité.

Pseudo-code de l'algorithme de Branch-and-Bound (Atamtürk & Savelsbergh, 2005)

Étape 1 : **Initialisation**
$L = \{PLE\}$ $z^* = \infty$ Pas de solution candidate.

Étape 2 : **Critère d'arrêt**
Si $L = \emptyset$, (x^*, y^*) est solution optimale

Étape 3 : **Choix du nœud**
On choisit $PLE(k)$ de L
L ← $L \backslash \{ PLE(k) \}$

Étape 4 : **Évaluation**
Résoudre $\overline{PLE(k)}$

Étape 5 : **Élagage**
Si $\overline{PLE(k)}$ est non réalisable alors : aller à l'étape 2.
Sinon, soit$(\bar{x}, \bar{y})$ solution de $v\left(\overline{PLE(k)}\right)$.

- Si $v\left(\overline{PLE(k)}\right) \geq z^{best}$: aller à l'étape 2.
- Si $\bar{x}$ ne vérifie pas les contraintes d'intégralité : aller à l'étape 6.
- Sinon z^*← $v\left(\overline{PLE(k)}\right)$ et (x^*, y^*) ← $(\bar{x}, \bar{y})$

Étape 6 : **Branchement**
On divise $F(\overline{PLE(k)})$ en i $F(\overline{PLE(k(\imath))})$avec $i = 1,2, \dots q$; tel que $F\left(PLE(k)\right) = \cup_{i=1}^{q} F(PLE(k(i)))$.
L← $L \cup_i PLE(k(i))$
Aller à l'étape 2.

Figure 3 Pseudo-code de l'algorithme du branch-and-bound

2.2.2 Sélection du nœud

L'un des premiers mécanismes pour améliorer les bornes est la stratégie de sélection du nœud ou sous-problème. Deux objectifs sont à considérer lors du choix de la stratégie :

- Réduire le temps de résolution;
- Trouver une « bonne » solution réalisable rapidement.

Nous ne présenterons pas toutes les stratégies, mais seulement celles qui nous intéressent dans notre projet.

2.2.2.1 Recherche meilleur d'abord

La première est la recherche meilleur d'abord, ou *best-first search*. Elle choisit le nœud qui a la plus petite valeur optimale $v(\overline{PLE(k)})$. Cette stratégie vise à augmenter la borne inférieure en cherchant à améliorer la relaxation du nœud de plus petite valeur optimale $v\left(\overline{PLE(k)}\right)$. Malheureusement au cours de l'exécution de l'algorithme, elle ne trouve pas nécessairement dans de brefs délais des solutions réalisables. Par conséquent, elle peut entraîner un parcours de nombreux nœuds pouvant engendrer un grand nombre de problèmes non élagués et peut donc être gourmande en mémoire.

2.2.2.2 Recherche en profondeur

La seconde est la recherche en profondeur ou *depth-first search*. Cette stratégie choisit toujours le nœud le plus en profondeur jusqu'à son élagage, puis remonte plus haut dans l'arbre. Cette approche favorise alors la décroissance de la borne supérieure, parce que les solutions réalisables sont typiquement trouvées dans les zones les plus profondes de l'arbre. Ainsi, une solution réalisable est trouvée plus rapidement. Cette stratégie facilite aussi la réoptimisation du calcul

de borne. Cependant, si la borne inférieure initiale est trop faible, la taille de l'arbre peut exploser, contrairement à la recherche meilleur d'abord.

La plupart des logiciels de programmation en nombres entiers choisissent de créer une stratégie hybride entre ces deux premières, afin de profiter de leurs avantages respectifs. Au début, la recherche en profondeur est favorisée, pour trouver rapidement des solutions de bonne qualité, tandis que subséquemment, l'emphase est généralement mise sur la recherche meilleur d'abord, afin de réduire le nombre total de nœuds explorés.

2.2.3 Quelques règles de branchement

Nous présenterons quelques règles de branchement, notamment celles que l'on retrouve parmi les options de branchement disponibles dans le logiciel de résolution utilisé.

2.2.3.1 Branchement sur une seule variable

Un principe de branchement « naturel » consiste à choisir une variable x_i qui est fractionnaire dans la solution du problème courant, que l'on note x_i^k, et d'ajouter dans un premier sous-problème (par branchement inférieur) la contrainte $x_i \leq \lfloor x_i^k \rfloor$, et dans le second par branchement supérieur $x_i \geq \lceil x_i^k \rceil$. Cette règle porte le nom de branchement simple sur une variable, ou dichotomie de variable. Cependant, souvent, le nombre de variables fractionnaires dans la solution courante peut être élevé, et un choix plus pointu peut s'avérer gagnant en efficacité en trouvant une variable qui améliorera considérablement les bornes. Il n'y a pas de règles générales pour le choix du branchement, mais il y a des moyens de prévoir quelles variables fractionnaires augmenteront la borne inférieure, une fois entières.

Une façon de procéder au choix de la variable à brancher, appelée branchement fort (« *strong branching* »), est d'évaluer parmi un ensemble de variables de base

qui ne sont pas entières dans le problème relaxé, celle dont le branchement améliorerait le plus l'objectif. Il s'agit de créer pour chacune de ces variables, un branchement supérieur et inférieur, et on optimise le sous-problème relaxé sur chaque branche en résolvant le programme linéaire correspondant. La variable entraînant la plus grande diminution de l'objectif est celle choisie pour le branchement. Le défaut principal de cette règle est que pour chaque variable candidate, deux programmes linéaires sont résolus. Pour restreindre le nombre de variables candidates, on peut les choisir selon leur influence sur l'objectif, et dans le cas de variables binaires, choisir les variables qui sont proches de ½.

2.2.3.2 Branchement de type GUB

La nature de certaines contraintes engendre des règles de branchement favorisant l'équilibre de l'arbre. Quand le problème a des contraintes de type dit *generalized upper bound* (GUB), dont la forme est la suivante :

$$\sum_{j \in B} x_j = 1 \; ou \sum_{j \in B} x_j \leq 1$$

où $B = \{ j \mid x_j \; est \; binaire \}$, une autre possibilité de branchement existe et porte le nom de *GUB Branching*.

Soit x^k la solution du problème relaxé $\overline{PLE(k)}$. On définit B' un sous-ensemble de B , tel que $0 < \sum_{j \in B'} x_j^k < 1$.

La règle de branchement de type GUB, ne concerne pas le branchement sur une variable, mais sur plusieurs. En effet, on ajoute dans un premier sous-problème la contrainte suivante :

$$\sum_{j \in B'} x_j = 0 \qquad \text{(Cas 1)}$$

et dans l'autre :

$$\sum_{j \in B \setminus B'} x_j = 0 \qquad \text{(Cas 2)}$$

Ces contraintes peuvent aussi être interprétées comme une fixation de certaines variables à leurs bornes : dans le cas 1, $\forall j \in B' \ x_j = 0$; dans le cas 2, $\forall j \in B \setminus B' \ x_j = 0$.

Quand il existe une relation d'ordre entre les variables de *B*, cet ensemble est dit SOS pour *Special Ordered Set*; la méthode de branchement associée est parfois appelée *SOS branching*.

Ce type de branchement sur un ensemble de variables, plutôt que sur une seule variable à la fois, favorise le balancement de l'arbre (Linderoth & Savelsbergh, 1999).

2.2.4 Aides à la résolution

2.2.4.1 Le prétraitement

Le prétraitement fait référence à un ensemble de transformations du programme mixte en nombres entiers (PLE), dans le but de réduire le temps de résolution. Les techniques de prétraitement permettent d'éliminer les contraintes redondantes, ou de resserrer les bornes sur les variables, ou de fixer des variables à leur borne par la dualité (écarts complémentaires). Le modèle peut ainsi se simplifier et le temps de résolution être réduit. Notons que le travail réalisé par le prétraitement peut demander beaucoup de temps et d'espace en mémoire.

Il est important de savoir que le prétraitement peut non seulement se faire à la racine de l'arbre, mais aussi à n'importe lequel de ses nœuds. Cependant, l'influence du prétraitement à un nœud, se limite au sous-arbre dont il est la racine. Il faut toujours évaluer si le temps consacré au prétraitement est compensé par la réduction du temps de calcul consacrés aux problèmes relaxés.

2.2.4.2 Plans de coupe

Définir un plan de coupe consiste à identifier une inégalité valide satisfaite par les solutions réalisables du *(PLE)*, mais qui ne l'est pas nécessairement par tous les points du domaine réalisable de la relaxation $\overline{PLE}$. Une telle coupe, ajoutée aux contraintes, peut permettre d'augmenter la qualité des résultats issus de la relaxation. En effet, seul le domaine réalisable du problème relaxé est réduit, le domaine du problème original *F(PLE)* demeurant inchangé. Après la résolution du nouveau problème relaxé, la génération de coupes se poursuit, si nécessaire.

La méthode du branch-and-cut est une généralisation de l'algorithme du branch-and-bound, dans laquelle on ajoute des coupes à la formulation associée à un nœud de l'arbre de recherche. Même si la résolution des relaxations à chaque nœud est plus longue, une réduction significative de la taille de l'arbre peut résulter de l'amélioration des bornes sur l'objectif de *(PLE)*. Une bonne implantation de la méthode du branch-and-cut peut s'avérer plus rapide que l'algorithme du branch-and-bound.

2.2.4.3 Utilisation des heuristiques

Les heuristiques primales sont des algorithmes qui tentent de trouver des solutions réalisables à *(PLE)* rapidement. Elles sont utilisées pour réduire la borne supérieure de la valeur de l'objectif. Au lieu d'attendre une solution entière réalisable issue de la relaxation à un nœud, il s'agit de trouver une solution réalisable à *(PLE)* par des heuristiques simples et rapides. Ainsi, très tôt dans la résolution, les heuristiques produisent une bonne borne supérieure permettant d'élaguer l'arbre de recherche plus rapidement.

Le rôle des heuristiques est complémentaire à celui joué par les plans de coupe. En effet, l'utilisation des plans de coupe réduit le domaine réalisable du problème relaxé $\overline{PLE}$, ce qui augmente les chances de l'heuristique de trouver une première solution réalisable proche de l'optimum. Le jeu des plans de coupe et de l'heuristique peuvent resserrer les bornes de la valeur optimale de *(PLE)* et ainsi réduire la taille de l'arbre.

Chapitre 3 : Implantation et résultats

Initialement le partenaire industriel aurait souhaité ne pas avoir à investir dans une licence de logiciel de résolution comme CPLEX. Nous avons alors cherché à utiliser le logiciel gratuit CBC (version 2.3) distribué par COIN-OR (www.coin-or.org). C'est pourquoi l'outil de modélisation utilisé est FlopC++ (Formulation of Linear Optimization Problems in C++), un langage algébrique aussi distribué par COIN-OR, où l'écriture du modèle se rapproche énormément de la formulation mathématique. L'autre avantage de FlopC++, est qu'il est possible à partir de la même modélisation, de lancer la résolution avec différents logiciels. Nous utiliserons CBC dans un premier temps, puis CPLEX(version 10) par la suite.

Au cours de ce chapitre, nous résumerons les efforts fournis pour résoudre le problème par une méthode exacte (algorithme du branch-and-bound). Nous présenterons d'abord les résultats obtenus pour la résolution du problème et de sa variante pour de courts horizons de temps. Puis nous présenterons une seconde approche de résolution dite par horizons roulants.

Les données utilisées pour la résolution du problème ont été fournis par FPInnovations. Ces données sont réelles et viennent d'une entreprise forestière.

3.1 Première approche

Voici une description du problème généré à partir des données fournies :

On cherche à obtenir un calendrier annuel de récolte soit $|T| = 26$ où une période représente deux semaines.

Le nombre de blocs noté $|\Gamma| = 184$ dont $|C| = 15$. Il y a $|K| = 11$ produits et on distingue 8 équipes dont 6 équipes de bois court et 2 de bois long.

Nombres de périodes	$\|T\|$	26
Nombre de secteurs	$\|S\|$	25
Nombre de blocs	$\|\Gamma\|$	184
Nombre de chemins	$\|C\|$	15
Nombre d'équipes	$\|Q\|$	8

Modèle original (avec la contrainte (20))

Nombre de contraintes	5 043 402
Nombre de variables binaires	143 296

Variante du modèle (avec la contrainte (20B))

Nombre de contraintes	1 358 086
Nombre de variables binaires	143 296

Tableau 1 Dimensions des problèmes

La taille du problème original ou sa variante étant considérable, nous avons d'abord souhaité analyser les premiers résultats sur des horizons de taille plus petite.

3.1.1 L'expérience avec CBC

Nous avions auparavant évalué les performances du logiciel CBC sur un autre problème avec une structure plus légère (détermination d'une règle de coupe, et un problème de transport associé). Les résultats étaient plutôt encourageants même pour des instances de grande taille; des solutions optimales étaient obtenues en des temps du même ordre que celles données par CPLEX. Cependant, dans ce mémoire, notre problème a une structure combinatoire difficile. Pour un problème avec deux périodes, il n'y avait toujours pas de solution au bout d'une heure de résolution. La seule solution réalisable trouvée avait un écart relatif qui dépassait les 90%. Nous avons donc abandonné cette approche. Dès lors, l'ensemble des tests effectués l'ont été en utilisant CPLEX, par l'intermédiaire de FlopC++.

3.1.2 Résolution avec CPLEX

Dans notre expérimentation avec CPLEX, nous cherchions d'abord à déterminer le nombre maximum de périodes pour lequel nous obtiendrions une solution optimale en un temps raisonnable.

Quelques choix de paramètres sont à préciser. Tout d'abord, les critères d'arrêt sont :

- une durée limite d'exécution TL = 18 000 secondes
- si une solution réalisable avec un écart relatif inférieur ou égal à 1% est trouvée.

Pour influencer les branchements, et ainsi accélérer la résolution du programme en nombres entiers, nous avons introduit des priorités de branchement sur les variables binaires. La priorité des variables d'affectation des équipes aux blocs de récolte est supérieure à celles des variables associées aux secteurs. En effet, ce sont les décisions liées aux blocs qui auront un effet plus direct sur l'amélioration ou non de la solution candidate au cours du branch-and-bound.

Finalement notons qu'avant de résoudre le problème en nombres entiers, CPLEX fait appel à une heuristique, dont nous ne connaissons pas les détails, qui fournit une première solution réalisable.

Les tests ont été effectués sur une plateforme du type Dual Core AMD Opteron(tm) Processor 285.

3.1.2.1 Résolution du problème original

Dans le tableau 2, nous constatons que jusqu'à $|T| = 4$ périodes, les solutions obtenues le sont grâce à l'heuristique de CPLEX à la racine. Cette heuristique détermine rapidement une première solution réalisable, et la solution quasi optimale (écart relatif < 1%) ne tarde pas à être trouvée. On notera aussi que la valeur de la relaxation et la valeur quasi optimale sont très proches. Malgré la taille importante des problèmes, on constate que la durée de la relaxation et celle

de la résolution du problème en nombres entiers sont raisonnables pour des problèmes comptant jusqu'à 4 périodes. Au-delà, le temps de résolution devient prohibitif. En effet, la relaxation du problème est déjà longue à résoudre pour $|T| = 5$, ou la résolution dépasse le temps limite d'exécution imposé TL = 18 000 secondes.

|T|	Nombre de variables binaires	Nombre total de variables	Nombre de contraintes	Durée de la relaxation (s)	Durée de la résolution (s)	Écart relatif de la solution fournie par l'heuristique
2	32 512	71 652	955 404	1	98	0,01%
3	37 128	84 798	1 115 633	16	209	0,93%
4	41 744	97 944	1 277 540	122	481	1%
5	46 360	111 090	1 450 285	404	>18 000	2,49%

Tableau 2 Résultats pour le problème original 1ere approche

L'ordre du nombre des contraintes étant du million, il est intéressant de comparer la résolution directe du problème original avec sa variante qui génère nettement moins de contraintes.

3.1.2.2 Résolution de la variante

Les tests concernant la résolution de la variante ont aussi été exécutés sur une plateforme du type Dual Core AMD Opteron(tm) Processor 285.

|T|	Nombre de variables binaires	Nombre total de variables	Nombre de contraintes	Durée de la relaxation (s)	Durée de la résolution (s)	Écart relatif de la solution fournie par l'heuristique
2	32 512	71 652	182 034	1	43	0,18%
3	37 128	84 798	214 973	4	74	0,55%
4	41 744	97 944	249 626	15	305	0,93%
5	46 360	111 090	285 721	40	> 18 000	25,31%

Tableau 3 Résultats pour la variante 1ere approche

Le problème formulé selon la variante ne semble pas plus facile à résoudre. Comme pour la résolution du problème original, le nombre de périodes maximum pour lequel on obtient une solution optimale est $|T| = 4$.

Même si les contraintes [20] augmentent la taille du problème, elles sont simples à écrire, et l'inférence de redondances une fois qu'une autre variable est fixée, doit faciliter le prétraitement et la résolution dans le MIP.

Après analyse, nous constatons que si la durée du prétraitement du problème original est deux fois plus longue que celle de la variante, les simplifications, éliminations et améliorations sont aussi plus importantes pour le problème original. En effet, le prétraitement réduit en moyenne le nombre de contraintes de 86% pour le problème original, contre 66% pour la variante. Pour les variables, en moyenne, les simplifications sont sensiblement les mêmes dans les deux cas.

3.2 Résolution par horizons roulants

La résolution directe du problème étant limitée à des instances ayant 5 périodes, on utilise une autre approche afin de pouvoir planifier sur un horizon de temps plus long en travaillant par horizons roulants. Il s'agit de résoudre une première fois le problème sur un court horizon. Une fois la solution trouvée, on relance la résolution sur un horizon plus long, en fixant certaines variables à leur valeur dans la solution de l'horizon précédent.

Voici le pseudo-code de cette approche :

nFix = nombre de périodes fixées

nLg = nombre de périodes de chaque horizon

nBTours = nombre d'horizons considérés

iT =0

s(iT) = solution optimale issue de la résolution du iT^{eme} horizon

Étape 1 : Création du problème pour NT = nLg + iT.nFix

Fixer les variables pour les périodes $t \leq$ iT. nFix

Résoudre la planification pour NT = nLg + nFix.iT périodes où les variables associées aux iT.nFix premières périodes sont fixées

Enregistrer la solution dans s(iT)

iT←iT+1

Étape 2 : Si iT< nBTours

Aller à Étape1

Étape 3 : Fin de la résolution

On obtient la planification pour un horizon global de longueur nBTours.nFix+nLg.

Figure 4 Pseudo-code de la résolution par horizons roulants

Il est difficile d'évaluer *a priori* la longueur de chaque horizon nLg qui permettra d'obtenir une bonne qualité de solution, même s'il semble raisonnable de penser qu'une planification sur un horizon trop court devrait être moins efficace à long

terme. Le modèle utilisé pour l'approche par horizons roulants est le problème original. Ce choix est motivé par les temps de résolution pour 3 et 4 périodes, qui, étant très raisonnables, nous offrent la possibilité de comparer les résolutions quand la taille de chaque horizon varie entre nLg = 3 et nLg = 4 périodes. Pour tous les tests, nous choisissons nFix = 1.

3.2.1 Introduction d'un estimé de la demande future

Dans le graphe ci-dessous (Figure 5), nous rassemblons la demande annuelle des usines. Certains produits comme $k = 1$, ont une demande généralement croissante et importante. D'autres, comme $k = 6$, ont une demande nulle pour les premières périodes, mais qui augmente par la suite. La résolution sur des horizons roulants de longueur trop courte manque de perspective, et les décisions prises pour un horizon, peuvent dégrader la qualité de la solution pour l'horizon suivant.

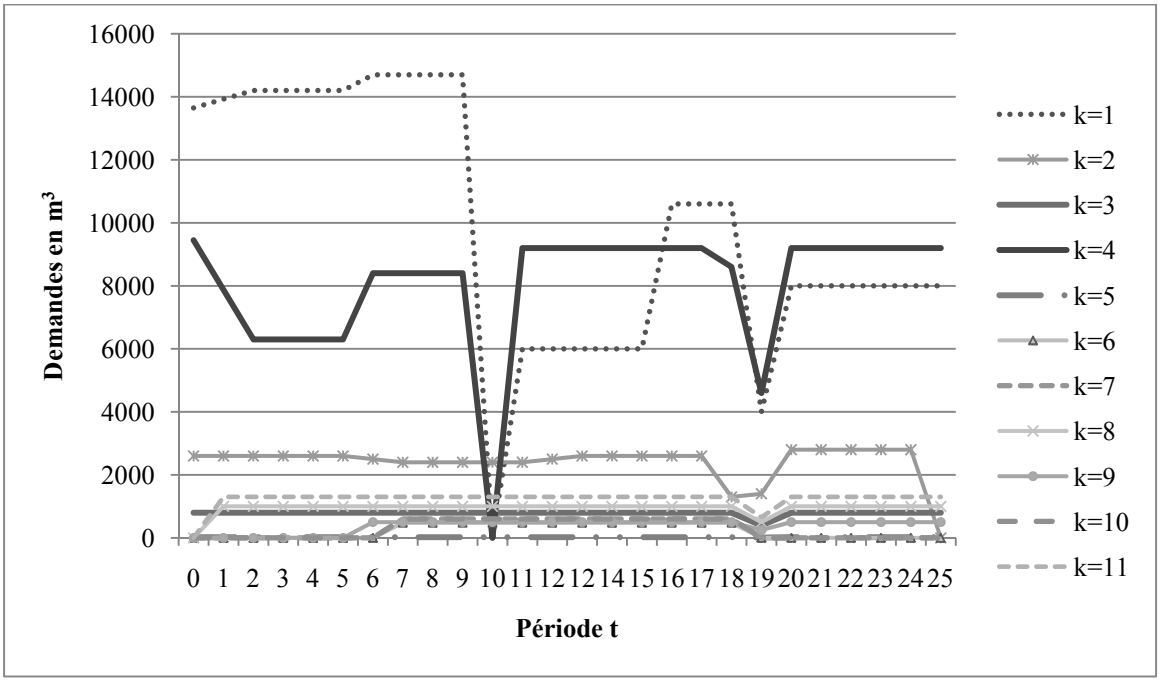


Figure 5 Demande des usines par période pour chaque produit

L'inconvénient d'une planification sur un court horizon est que les décisions prises ne tiennent pas compte des demandes futures. La planification a lieu pour un horizon de taille nLg, comme si toute activité cessait au-delà. Pour

éviter que la résolution à chaque horizon ne soit trop « myope », on introduit une période fictive en fin de chaque horizon. L'ensemble des périodes NT comprend une période fictive additionnelle à laquelle est associée un estimé de la demande future. Les données dépendant de cette période fictive, comme la demande des usines et les niveaux d'inventaire requis, résultent d'une moyenne pondérée des données d'un certain nombre de périodes futures, où la pondération est plus importante pour les périodes les plus proches. Nous avons choisi de faire varier le nombre de périodes considérées pour calculer les estimés, entre 3 et 4 périodes. Nous n'avons pas compté plus de périodes pour calculer l'estimé, car plus grand est le nombre de périodes considérées, plus faible est la pondération associée à la demande de chaque période future. La motivation principale de l'introduction de l'estimé étant de mieux appréhender les demandes futures, la pondération qui leur est associée ne doit donc pas être trop faible. Après quelques tests, nous avons constaté qu'une pondération minimum de 10% est nécessaire pour influencer de façon sensible le travail de récolte.

Voici un exemple du calcul de l'estimé de la demande :

Soient t_1, t_2, t_3 et t_4 les quatre périodes suivant les NT = 3 premières périodes.

$\mathrm{NT} = |\{0,1,2\}|$

$t_1 = 3, t_2 = 4, t_3 = 5, t_4 = 6$

Soit t^* la période fictive.

La demande en produit k à la période t^* d_{kt^*} est calculée de la façon suivante:

- $d_{kt^*} = \frac{1}{3}(0{,}75d_{kt_1} + 0{,}15d_{kt_2} + 0{,}10d_{k3})$ si l'estimé est calculé à partir des demandes des trois périodes futures.
- $d_{kt^*} = \frac{1}{4}(0{,}50d_{kt_1} + 0{,}20d_{kt_2} + 0{,}15d_{kt_3} + 0{,}15d_{kt_4})$ si l'estimé est calculé à partir des demandes des quatre périodes futures.

Les pourcentages sont choisis de façon à donner priorité aux demandes immédiates, sans pour autant trop négliger les autres demandes suivantes.

3.2.2 Précisions concernant les paramètres utilisés lors de la résolution par horizons roulants

Nous introduisons d'autres paramètres afin de mieux contrôler la résolution du problème par horizons roulants.

Pour un horizon, le processus de résolution peut se décomposer en 2 étapes :

- Étape 1 : génération du modèle, relaxation et résolution du problème (heuristique initiale de CPLEX suivie du branch-and-bound si la solution fournie par l'heuristique est supérieure à la tolérance choisie (1%))
- Étape 2 : le polishing.

Le « *polishing* » est une option de CPLEX. Il s'agit d'une heuristique (dont nous ne connaissons pas les détails) qui est disponible pour la résolution de problèmes linéaires mixtes en nombres entiers complexes c'est-à-dire « pour lesquels l'optimalité est peu probable ». La priorité est alors d'améliorer la solution réalisable la plus acceptable, ici celle fournie par l'étape 1.[1]

Chacune de ces deux étapes a une limite de temps d'exécution. Notons T_BB la limite de temps pour la première étape, et T_P la seconde.

- T_BB = 3 600s
- T_P = 1 800s

Quand la limite de temps T_BB est atteinte, cela signifie que la solution réalisable obtenue n'a pas un écart relatif à l'optimum inférieur à la tolérance (< 1%).

Au cours du processus de résolution, le polishing s'exécute pendant 1 800s, quelque soit l'écart de la solution fournie par la première étape. Il se peut alors que le polishing ne soit pas nécessaire, notamment quand la solution issue de la

[1] http://www.ilog.com/corporate/connection/nov08jan09fr/product.cfm

première étape est déjà optimale ou quasi-optimale (gap <1%). Cependant, le gain de temps réalisé par le polishing quand le branch-and-bound n'arrive pas à effectuer des branchements qui diminuent la borne supérieure de façon significative, est considérable, comme nous le verrons dans le Tableau 4.

Les priorités de branchement introduites sur les variables binaires sont maintenues.

3.2.3 Résultats de la résolution du problème original par horizons roulants avec introduction d'un estimé

Dans le tableau 4, nous comparons les résolutions avec des horizons roulants de nLg = 3 ou nLg = 4 périodes incluant la période fictive. Le nombre de périodes fixées à chaque horizon est nFix = 1 et les données pour la période fictive sont obtenues en utilisant les trois prochaines périodes. Nous indiquons la durée de résolution pour chaque NT. Cette durée comprend toutes les résolutions, depuis le premier horizon NT =3 ou 4 à l'horizon courant $NT = nLg + iT * nFix$ .Les tests ont été exécutés sur une plateforme du type Dual Core AMD Opteron(tm) Processor 285.

NT	nLg	Limite du B&B atteinte ?	Réduction du gap par polishing	Durée de la résolution (s)	Objectif	Ecart relatif
4	3	non	0,52%->0,27%	4 207	9,33E+06	0,13
	4	non	0,55%->0,41%	2 218	9,32E+06	
5	3	non	0,83%->0,64%	9 818	1,05E+07	0,40
	4	non	1,73%->0,61%	7 958	1,04E+07	
6	3	non	1,07%->0,80%	15 054	1,24E+07	1,93
	4	non	0,61%->0,00%	11 110	1,21E+07	
7	3	oui	1,69%->0,89%	20 770	1,63E+07	14,89
	4	non	2,06%->0,52%	16 787	1,38E+07	
8	3	oui	70,85%->5,08%	26 498	2,31E+08	93,28
	4	oui	8,24%->4,62%	22 534	1,55E+07	
9	3	non	0,77%->0,42%	30 240	5,00E+09	99,54
	4	oui	93,83%->3,62%	28 328	2,28E+07	

10	3	non	0,71%->0,13%	35 300	1,02E+10	54,52
	4	oui	10,25%->2,08%	34 160	4,63E+09	
11	3	non	0,80%->0,13%	40 922	1,59E+10	38,26
	4	oui	4,43%->1,11%	39 971	9,84E+09	
12	3	non	0,52%->0,01%	43 141	1,59E+10	38,26
	4	non	0,80%->0,01%	45 507	9,84E+09	
13	3	non	0,00%	45 499	1,59E+10	38,26
	4	non	0,00%	47 845	9,84E+09	
14	3	non	0,04%->0,00%	47 803	1,59E+10	38,26
	4	non	0,68%->0,00%	50 187	9,84E+09	
15	3	non	0,01%->0,00%	50 154	1,59E+10	38,26
	4	non	0,56%->0,00%	52 557	9,84E+09	
16	3	non	0,63%->0,00%	52 524	1,59E+10	38,26
	4	non	0,00%	54 961	9,84E+09	
17	3	non	0,53%->0,00%	54 901	1,59E+10	38,25
	4	non	0,03%->0,00%	57 439	9,84E+09	
18	3	non	0,22%->0,00%	57 299	1,59E+10	38,25
	4	non	0,29%->0,00%	59 957	9,84E+09	
19	3	non	0,02%->0,00%	59 777	1,59E+10	38,26
	4	non	0,17%->0,00%	62 457	9,84E+09	
20	3	non	0,04%->0,00%	62 250	1,59E+10	38,25
	4	non	0,13%->0,00%	64 997	9,84E+09	
21	3	non	0,26%->0,00%	64 800	1,59E+10	38,25
	4	non	0,44%->0,00%	67 545	9,84E+09	
22	3	non	0,10%->0,00%	67 320	1,59E+10	38,25
	4	non	0,29%->0,00%	70 176	9,84E+09	
23	3	non	0,05%->0,00%	69 836	1,59E+10	38,25
	4	non	0,06%->0,00%	72 725	9,85E+09	
24	3	non	0,05%->0,00%	72 373	1,59E+10	38,25
	4	non	0,03%->0,00%	75 360	9,85E+09	
25	3	non	0,02%->0,00%	73 345	1,59E+10	38,25
	4	non	0,11%->0,00%	77 921	9,85E+09	
26	3	non	0,06%->0,00%	75 893	1,60E+10	38,27
	4	non	0,09%->0,00%	80 430	9,85E+09	

Tableau 4 Résolutions par horizons roulants du problème original nLg=3, nLg=4

Pour mesurer l'écart relatif entre les objectifs selon que nLg=3 et nLg=4 nous utilisons la formule suivante :

$$\frac{objectif(nLg = 3) - objectif\ (nLg = 4))}{objectif(nLg = 3)} \times 100$$

Commençons par quelques remarques sur les durées de résolution. De NT = 3 à 11, la résolution par horizons roulants avec nLg = 3, prend en moyenne 17% plus de temps que la résolution avec nLg = 4. Ceci découle en particulier du fait que pour NT = 4, nous devons résoudre deux horizons roulants pour nLg = 3 et un seul pour nLg = 4. Cependant, au final, sur 26 périodes, la résolution par horizons roulants avec nLg = 4, demande 5% plus de temps que la résolution avec nLg = 3.

La rapidité de la résolution avec nLg = 3 n'est pas sans contrepartie. En effet la valeur optimale obtenue avec nLg = 3 est en moyenne 38% plus élevée que celle obtenue avec nLg = 4. La plus grande différence entre ces deux valeurs est obtenue lorsque NT = 9 où l'écart est un peu moins de 100%.

Les troisième et quatrième colonnes du Tableau 4 précisent respectivement pour chaque horizon NT, si la limite T_BB est atteinte (colonne 3) et détaille l'amélioration du polishing (colonne 4). En étudiant ces deux colonnes, nous pouvons identifier les horizons pour lesquels la résolution est compliquée car l'étape 1 atteint sa limite d'exécution pour NT = 8, 9 10 et 11. L'heuristique fournit une moins bonne solution réalisable et le branch-and-bound poursuit donc la résolution. Quand nLg = 3, la limite T_BB est atteinte seulement pour NT = 8. La solution alors obtenue à la limite d'exécution T_BB du branch-and-bound a un écart de plus de 70%, mais l'application du polishing engendre une solution réalisable avec un écart d'environ 5%. Pour nLg = 4, une situation similaire se présente pour NT = 8, 9, 10 et 11. La plus nette amélioration par polishing est observée pour NT = 9, car la solution ayant un écart de 93%, est améliorée pour obtenir une solution ayant un écart d'environ 3%.

Pourquoi les problèmes où NT = 8, 9, 10 et 11 sont-ils plus « difficiles » à résoudre? Notons que lorsque t=8, 9, 10 et 11, tous les produits ont une demande non nulle, la saison correspondante est l'été où les coûts d'inventaire sont les plus élevés, puisque la chaleur peut altérer la qualité du bois en inventaire. De plus les équipes étant déjà au travail, il faut s'assurer que les affectations respectent une

continuité avec les périodes précédentes. Par conséquent, l'heuristique initiale a moins de chances de fournir une solution réalisable d'écart très faible, comme c'était le cas pour les horizons précédents. Ainsi le branch-and-bound doit être appliquée plus longtemps et la limite d'exécution T_BB est plus souvent atteinte. De plus, le polishing trouve une plus grande utilité pour ces mêmes périodes. En effet l'amélioration qu'il est possible d'atteindre en utilisant seulement le branch-and-bound pendant une journée, peut être réalisée en quelques minutes avec le polishing.

Également, pour les périodes où la limite T_BB n'est pas atteinte, l'heuristique fournit sinon la solution optimale, du moins une solution réalisable proche de l'optimale. Le polishing le plus souvent la rend optimale. Pour NT = 13 et 16, la solution qu'exploite le polishing est déjà optimale.

Dans le graphe de la Figure 6, nous observons l'évolution de l'objectif lors de la résolution par horizons roulants, pour nLg =3 et nLg = 4. Les courbes ont le même comportement, mais le saut de valeurs que nous constatons entre nT = 9 et nT = 11, est beaucoup plus élevé quand nLg =3. Même si la résolution avec nLg = 4 est moins rapide qu'avec nLg = 3, la qualité de la solution est nettement meilleure. Ainsi, le reste des tests avec la résolution par horizons roulants, sera complété avec nLg = 4 incluant une période fictive.

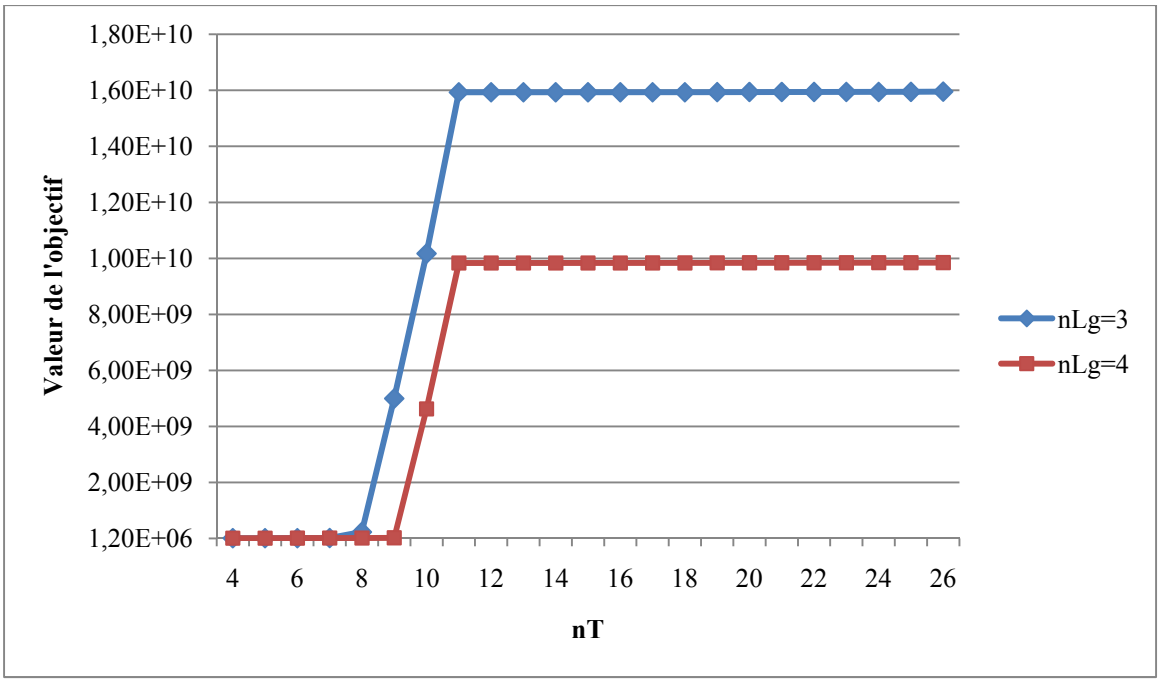


Figure 6 Évolution de l'objectif lors de la résolution par horizons roulants

Nous souhaitions savoir quels changements pouvaient être observés en augmentant le nombre de périodes prises en compte pour le calcul de l'estimé. Dans la Figure 7, on compare d'abord l'évolution de l'objectif en utilisant un estimé basé sur 3 et 4 périodes, avec nLg = 4. Le comportement des courbes est similaire même si l'objectif pour l'estimé de 4 périodes est légèrement plus élevé. Ceci s'applique par le fait que plus on considère de périodes pour calculer l'estimé, plus faible devient la pondération associée à chaque demande par période (voir page 49). Finalement, les courbes de la Figure 7 indiquent que le saut dans l'objectif se maintient, peu importe l'estimé de la demande future.

Nous avons également résolu le problème sans introduire la limite de temps T_BB sur l'exécution du branch-and-bound. Nous n'avons pas pu aller au-delà de la résolution pour 22 périodes car l'espace mémoire disponible était alors trop réduit pour que la résolution se poursuive. La durée de la résolution pour 22 périodes est d'environ une semaine. En comparant avec la résolution avec T_BB, notons que la solution fournie par cette dernière n'est pas altérée de façon majeure. En effet, l'écart entre les deux solutions est alors inférieur à 5%.

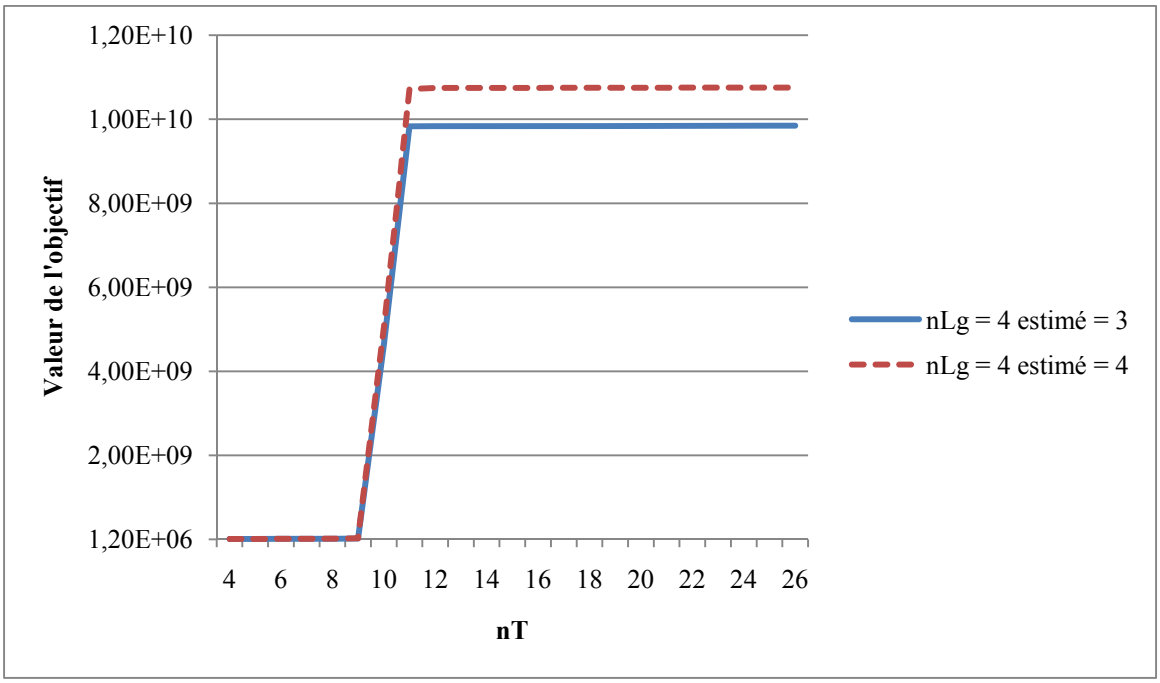


Figure 7 Évolution de l'objectif selon l'estimé lors de la résolution par horizons roulants nLg=4

3.2.4 Résultats de la résolution de la variante par horizons roulants avec introduction d'un estimé

Nous résumons ici les résultats de la résolution par horizons roulants de la variante. Les paramètres choisis sont les mêmes que ceux utilisés pour la résolution par horizons roulants du problème original.

NT	nLg	Limite du B&B atteinte ?	Réduction du gap par polishing	Durée de la résolution (s)	Objectif	Ecart relatif
4	3	non	0,79%->0,33%	3963	9,34E+06	0,11
	4	non	0,50%->0,42%	2040	9,32E+06	
5	3	non	0,43%->0,30%	6737	1,05E+07	0,26
	4	non	0,63%->0,58%	4147	1,04E+07	
6	3	non	1,02%->0,42%	10325	1,23E+07	1,60
	4	oui	1,00%->0,38%	9704	1,21E+07	
7	3	oui	2,71%->1,18%	15924	1,66E+07	16,10
	4	oui	1,77%->1,22%	15267	1,39E+07	
8	3	oui	47,21%->28,30%	21537	5,28E+08	97,07
	4	oui	9,69%->3,18%	20902	1,55E+07	
9	3	oui	2,52%->1,43%	27175	5,46E+09	99,55
	4	oui	98,37%->13,51%	26649	2,44E+07	
10	3	oui	1,27%->0,47%	32867	1,07E+10	55,90
	4	oui	39,40%->8,50%	32321	4,73E+09	
11	3	oui	1,43%->0,80%	38564	1,65E+10	34,22
	4	oui	19,80%->8,01%	38260	1,09E+10	
12	3	non	0,07%->0,00%	40622	1,65E+10	34,23
	4	non	0,60%->0,01%	40889	1,09E+10	
13	3	non	0,04%->0,00%	42744	1,65E+10	34,23
	4	non	0,00%	43061	1,09E+10	
14	3	non	0,49%->0,00%	44865	1,65E+10	34,23
	4	non	0,12%->0,00%	45114	1,09E+10	
15	3	non	0,06%->0,00%	47011	1,65E+10	34,23
	4	non	0,02%->0,00%	47228	1,09E+10	
16	3	non	0,02%->0,00%	49127	1,65E+10	34,23
	4	non	0,04%->0,00%	49770	1,09E+10	
17	3	non	0,05%->0,00%	51275	1,65E+10	34,23
	4	non	0,10%->0,00%	51973	1,09E+10	
18	3	non	0,41%->0,00%	53376	1,65E+10	34,23
	4	non	0,12%->0,00%	54136	1,09E+10	
19	3	non	0,84%->0,01%	55575	1,65E+10	34,25
	4	non	0,05%->0,00%	56354	1,09E+10	
20	3	oui	2,51%->0,32%	61296	1,66E+10	34,46
	4	non	0,61%->0,03%	58584	1,09E+10	
21	3	non	0,30%->0,01%	63446	1,66E+10	34,47
	4	non	0,20%->0,01%	60803	1,09E+10	
22	3	non	0,03%->0,00%	65655	1,66E+10	34,47
	4	non	0,02%->0,00%	63624	1,09E+10	

23	3	non	0,07%->0,00%	67862	1,66E+10	34,48
	4	non	0,14%->0,00%	65790	1,09E+10	
24	3	non	0,10%->0,00%	70080	1,66E+10	34,49
	4	non	0,10%->0,00%	67984	1,09E+10	
25	3	non	0,02%->0,00%	72299	1,66E+10	34,51
	4	non	0,11%->0,00%	70173	1,09E+10	
26	3	non	0,05%->0,00%	74469	1,66E+10	34,53
	4	non	0,14%->0,01%	72832	1,09E+10	

Tableau 5 Résolution par horizons roulants de la variante nLg=3, nLg=4

Comme pour le problème original, la Figure 8 illustre que la résolution est meilleure quand la taille de chaque horizon est de nLg = 4. L'écart entre les valeurs optimales est aussi plus élevé quand NT = 8 et 9, que nLg =3 ou 4. Cependant la qualité des solutions trouvées après avoir utilisé le polishing, est moins bonne que pour le problème original. Par exemple, quand NT = 8 avec nLg = 3, la solution à la fin du polishing a un gap de 28,30%. De même pour NT = 9, 10 et 11 avec nLg = 4, la solution à la fin du polishing a respectivement un écart de 13,51%, 8,50% et 8,51%. Par contre pour l'ensemble des autres valeurs de NT, les solutions sont optimales ou quasi-optimales.

Comme pour le problème original, nous avons souhaité évaluer l'influence du nombre de périodes considérées lors du calcul de l'estimé (Figure 9). Le même constat réalisé pour le problème original (page 55) est aussi valable pour la variante. La résolution issue de l'introduction d'un estimé de 3 périodes fournit une meilleure solution que l'estimé de 4 périodes. L'écart moyen entre les solutions est du même ordre que celui pour le problème original. En effet l'écart moyen entre la solution de la résolution avec un estimée de 4 périodes et celle de 3 périodes est d'environ 6,86% pour le problème original et de 7,89% pour la variante.

Finalement dans la Figure 10 nous retrouvons les courbes montrant l'évolution de la fonction économique pour les deux formulations. Ainsi il semble que la formulation du problème original engendre de meilleurs résultats.

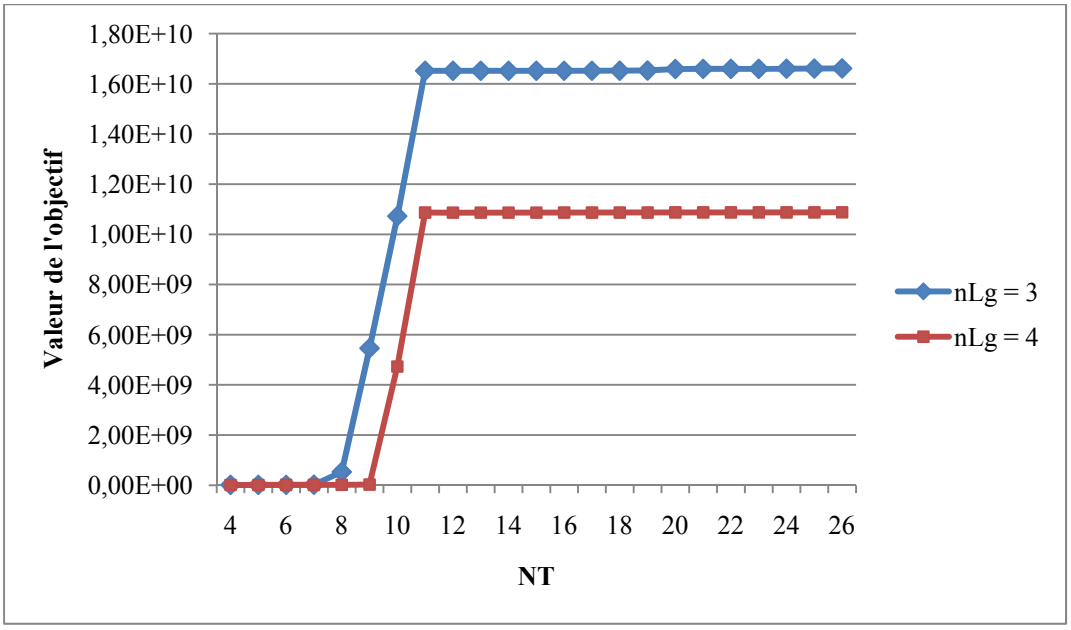


Figure 8 Évolution de l'objectif lors de la résolution de la variante par horizons roulants (estimée de 3 périodes)

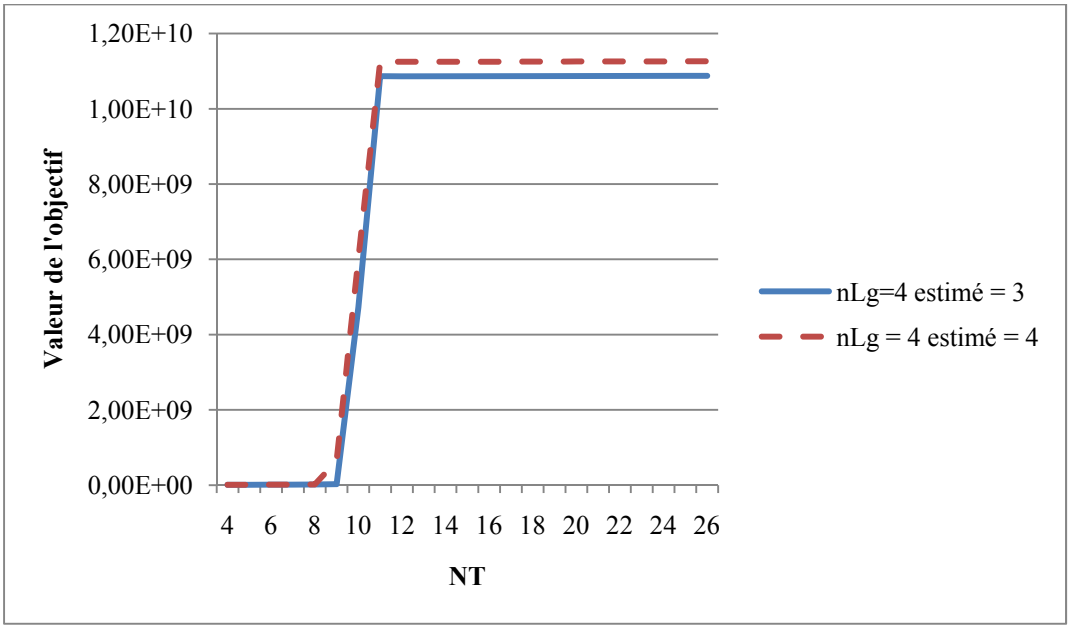


Figure 9 Évolution de l'objectif selon l'estimé lors de la résolution de la variante par horizons roulants nLg=4

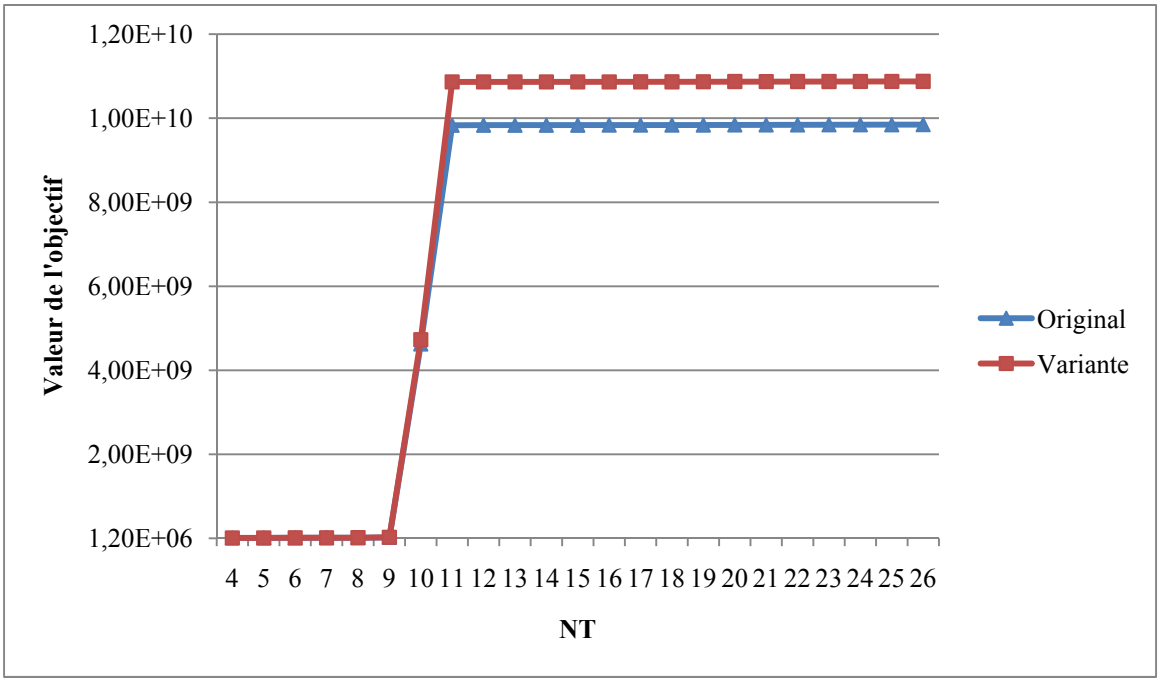


Figure 10 Problème original vs variante : résolution par horizons roulants, nLg=4 estimée de 3 périodes

3.3 Analyse des résultats

Nous résumons ici, l'ensemble des expériences menées pour résoudre notre problème de planification, qui ont été décrites dans ce chapitre.

D'abord, la résolution par le logiciel gratuit a été un échec, ce qui a favorisé le choix d'une utilisation principale de CPLEX.

Nous avons commencé par résoudre de petites instances du problème et de sa variante, les horizons variant entre 2 et 6 périodes. Pour les 4 premières périodes, nous avons obtenu en un temps raisonnable des solutions quasi optimales (écart relatif inférieur à 1%). Pour 5 et 6 périodes le temps d'exécution limite est atteint sans qu'une solution quasi optimale ne soit obtenue.

Nous choisissons de résoudre d'abord le problème original par horizons roulants, en introduisant des limites de temps d'exécution afin de garder un certain contrôle sur la résolution. Pour que les décisions prises au cours de la planification tiennent compte des demandes futures, nous avons inséré une période fictive à laquelle est associée un estimé de la demande future. En effet en perdant de vue les demandes futures (particulièrement celles qui sont croissantes), ou en ne les prévoyant pas suffisamment tôt, on pourrait se retrouver dans une situation où même si toutes les équipes travaillaient à capacité, la récolte et les stocks en inventaire ne suffiraient pas pour satisfaire la demande des usines. La quantité manquante doit être alors commandée, ces volumes sont extrêmement pénalisés dans l'objectif, provoquant ainsi un saut dans la valeur de l'objectif. Cette situation est illustrée dans les Figure 6 et Figure 7. En analysant les résultats de la planification, la demande du produit 1 est satisfaite jusqu'à NT = 8 périodes, mais à partir de NT = 9 pour nLg=3, ou de NT = 10 pour nLg = 4, la récolte et les inventaires ne suffisent plus. Des commandes de produit 1 sont alors effectuées, provoquant ainsi un saut dans la valeur de l'objectif. Ce saut est cependant moins important quand nLg = 4, confirmant ainsi que la résolution sur un horizon trop court, dégrade la qualité de la résolution au long terme. Cependant, quelque soit la valeur de nLg (nLg=3 ou nLg = 4), il y a toujours un saut de la valeur de l'objectif.

A priori, en augmentant le nombre de périodes futures pour calculer l'estimé, le phénomène de saut de la valeur de l'objectif risque de se maintenir (voir figure 7). En effet, plus le nombre de périodes considéré est grand, plus faible est le poids attribué à chaque demande future. Une forte demande ne serait donc pas forcément appréhendée avec cette stratégie.

Puis nous avons souhaité comparer la résolution de la variante et celle du problème original sous les mêmes conditions. Même si la variante génère nettement moins de contraintes que le problème original, sa structure ne semble pas favoriser la vitesse de résolution. Pour les horizons critiques de NT = 8 à 11, le branch-and-bound et le polishing améliorent moins bien la borne supérieure. La

solution fournie sous les mêmes conditions que le problème original par horizons roulants est donc de moins bonne qualité.

Le nombre de périodes prises en compte pour le calcul de l'estimé n'ayant qu'une influence limitée sur la résolution, il semble important de penser à une autre alternative afin de réduire au moins la quantité de produits commandés, et ainsi réduire le saut de l'objectif. Une future approche pour la résolution par horizons roulants consistera à influencer le travail de récolte, de façon à satisfaire les demandes immédiates des usines, mais aussi de façon à satisfaire une « demande moyenne » calculée en prenant en compte toutes les demandes de l'année. Ainsi les décisions prises au cours d'une période devraient être favorables à la réalisation de la planification pour les périodes futures qui en dépendront. En fait, ceci sera réalisé en ajoutant des contraintes supplémentaires pour s'assurer qu'à chaque période la production des équipes puisse satisfaire un niveau suffisant pour tenir compte des périodes futures. Ces contraintes s'ajouteront à celles qui assurent la satisfaction de la demande et le respect des niveaux d'inventaire pour la période actuelle.

Conclusion

Dans le premier chapitre nous avons donc décrit le contexte forestier dans lequel s'inscrit notre projet, le problème de planification que nous cherchons à résoudre, et sa modélisation. Il s'agit d'un problème linéaire mixte en nombres entiers que nous choisissons de résoudre par une méthode exacte, le branch-and-bound. Après une revue de la littérature proche de notre projet, nous rappelons les grandes lignes de la programmation en nombres entiers, et détaillons le fonctionnement de l'algorithme du branch-and-bound. Dans le troisième et dernier chapitre, nous avons pu évaluer combien la résolution directe de notre problème de planification était difficile pour les instances avec un grand nombre de périodes. Cependant l'approche des horizons roulants s'est avérée fructueuse. Grâce à elle, en moins d'une journée, il est possible de planifier les activités de récolte des blocs pour l'année (26 périodes), en utilisant une méthode exacte, le branch-and-bound.

Il est en effet important de rappeler que la plupart des études sur la planification de la récolte à un niveau opérationnel (i.e. pour un horizon de moins d'un an) ont utilisé des heuristiques pour résoudre leur modèle, car très vite la résolution du problème en nombre entiers par CPLEX atteignait ses limites, notamment à cause de la taille imposante des modèles.

Bien sûr nous n'entendons pas fournir une solution miracle à tout problème de planification forestière, mais il est intéressant de constater la qualité des résultats obtenus avec une méthode du branch-and-bound par horizons roulants. En ce qui concerne le partenaire industriel, cet outil, même dans son état actuel, pourrait lui sauver du temps, puisque la planification des opérations pour 3 à 4 mois nécessite un travail de près d'un mois, sans oublier le fait qu'il n'y avait aucun souci d'optimisation dans la réalisation des calendriers de récolte.

Pour des travaux futurs, il faudrait approfondir la piste présentée à la section 3.3, qui vise à faire mieux que la résolution avec introduction d'un estimé de la demande future. Cette approche pourrait réduire l'importance du saut dans la

valeur de l'objectif qui est apparu au cours de la résolution par horizons roulants avec estimé. On pourrait aussi chercher à résoudre le problème dans sa globalité en définissant une heuristique pour ce problème. Nous rappelons que l'heuristique de CPLEX a su fournir de très bonnes solutions pour les petites instances. Enfin, une autre approche envisageable pour la résolution de ce problème est celle de la décomposition. En effet, la structure du modèle s'y prête puisque la partie combinatoire du modèle est essentiellement liée à la récolte, le reste du modèle étant un problème linéaire.

La gestion de l'espace mémoire est un facteur important pour la résolution avec un horizon long, c'est pour cela qu'améliorer l'implantation actuelle pourrait être une bonne initiative. Il serait aussi pratique d'utiliser la modélisation directement par Ilog Concert, au lieu de FlopC++.

Une autre expérience à faire est de tester ce modèle avec un plus grand nombre de blocs disponibles, $|\Gamma| > 184$. En effet, nous ne disposions pas de telles données pour faire ces tests. La planification pour une année pourrait être différente, peut être même plus rapide même si le nombre de contraintes générées explose. En effet plus grand est le nombre de blocs, plus grande sera l'offre par produit. Ainsi les disponibilités et les conditions d'accès pourraient jouer en faveur d'une résolution plus rapide.

Bibliographie

Atamtürk, A., Savelsbergh, M. (2005). Integer Programming Software Systems. *Annals of Operations Research* 140, pp. 67-124.

Côté, M. (2003). *Dictionnaire de la foresterie.* Sainte-Foy, Québec: Ordre des ingénieurs forestiers du Québec.

Epstein, R., Morales, R., Seron, J., Weintraub, A. (1999). Use of OR Systems in the Chilean Forest Industries. *Interfaces* 29, pp. 7-29.

Hillier, F., Lieberman, G. (2004). *Introduction to Operations Research* McGrawHill.

Karlsonn, J., Rönnqvist, M., Bergstrom, J. (2004). An optimization model for annual harvest planning. *Canadian Journal of Forest Research* 34 , pp. 1747-1754.

Karlsonn, J., Rönnqvist, M., Bergstrom, J. (2003). Short-term harvest planning including scheduling of harvest crews. *International Transactions in Operational Research* 10, pp. 413-431.

Lacroix, S. (2005). *Modèle intégré de sélection du moment et du lieu de récolte en fonction des coûts d'opérations et des impacts au niveau du sol.* Mémoire de maitrise, Faculté de foresterie de gérographie et de géomatique, Département des sciences du bois et de la forêt, Université Laval.

Linderoth, J., Ralphs, T. (2005). Noncommercial Software for Mixed-Integer Linear Programming. *CRC Press Operations Research Series*, pp. 253-303.

Linderoth, J., Savelsbergh, M. (1999). A computational study of search strategies for mixed integer programming. *Informs Journal on Computing* 11, pp. 173-187.

Mitchell, S. (2004). *Operational Forest Harvest Scheduling A mathematical model and solution strategy,* PhD Thesis, Departement of Engineering Science, Scool of Engeenering,University of Auckland.

Mitra, G. (1973). Investigation of some branch-and-bound strategies for the solution of mixed integer linear programs. *Mathematical Programming* 4, pp. 155-173.

Rönnqvist, R. (2003). Optimization in forestry. *Mathematical Programming* 97 , pp. 267-284.

Wolsey, L. (1998). *Integer Programming.* Wiley and Sons.

COIN-OR, www.coin-or.org site consulté le 09 Septembre 2009.