

Problème de tournées de collectes et livraisons multi-périodique : résolution grâce au GRASP

E. Grellier¹, P.Dejax¹, et N.Jussien²

¹ IRCCyN, Ecole des Mines de Nantes, 4 rue Alfred Kastler, 44300 Nantes
{emilie.grellier,pierre.dejax}@emn.fr

² LINA, Ecole des Mines de Nantes, 4 rue Alfred Kastler, 44300 Nantes
narendra.jussien@emn.fr

Mots-clés : GRASP, collectes et livraisons, multi-périodique.

1 Description du problème

La logistique inverse concerne la prise en compte des flux de retour de produits, d’emballages ou de produits de manutention des clients vers le système productif à des fins économiques, environnementales ou de service. C’est dans ce contexte que nous nous intéressons à la construction des tournées prenant en compte les flux directs (livraisons) et indirects (collectes) sur un horizon de temps fini (5 jours). Nous nous plaçons donc dans un réseau où un dépôt livre plusieurs sites dont les demandes sont connues pour chaque journée. Par ailleurs, les sites expriment une demande de collectes de produits en retour et de produits de manutention à destination du dépôt. Pour satisfaire les demandes de livraisons et de collectes nous disposons d’une flotte de véhicules homogènes qui partent et reviennent tous au dépôt. Plusieurs véhicules peuvent livrer une même demande. Nous allons tester deux stratégies différentes : une visant à répondre à la demande le jour où elle doit être honorée, l’autre visant à prendre de l’avance sur les livraisons si cela diminue le coût global.

2 Résolution

Ce problème est résolu grâce à la méta-heuristique GRASP (Greedy Randomized Adaptive Search Procedure) proposée par Feo et Resende [1]. Cette méthode de résolution est un processus itératif. Chaque itération comprend deux étapes : la construction et l’exploration d’un voisinage. L’étape de construction crée une solution réalisable grâce à une fonction d’évaluation gloutonne, son voisinage est alors exploré dans la seconde étape de la méthode. Pour explorer le voisinage de la solution nous allons utiliser deux méthodes : une recherche locale (basée sur des échanges et des déplacements), une autre basée sur la technique LNS (Large Neighbourhood Search) [2]. Cette technique explore un grand voisinage de la solution actuelle en choisissant un certain nombre de visites de site que l’on enlève de la solution, la ré-insertion de ces visites est effectuée grâce à une recherche arborescente basée sur la programmation par contraintes. Les paramètres de la méthode GRASP, qui sont la taille de la liste de candidats et le nombre d’itérations du processus, sont variables.

2.1 Résolution du problème sans avance

Nous chercherons à minimiser les coûts de routage et à satisfaire toutes les demandes. Chaque demande de livraison ou de collecte doit être honorée le jour même. Nous débutons la phase de construction de solution par une tournée vide (dépôt \rightarrow dépôt) par véhicule pour chaque journée. Pour chaque site nous devons décider par quelle tournée sa demande est satisfaite, quelle quantité de demande est satisfaite et quel est son instant de visite dans la tournée (de façon à respecter les fenêtres de visite). Pour cela, nous calculons le coût de planification et la pénalité de chaque nœud, dans chaque tournée selon chacune des quantités possibles. Le coût de planification du site k entre les sites i et j dans la tournée t_1 peut se diviser en 2 coûts et la pénalité d’insertion d’un site dans une tournée se divise en 3 pénalités. La pénalité d’insertion prend en compte :

- Le remplissage du véhicule (on remplit le plus possible les véhicules) : pénalité 1,
- L’augmentation de la distance parcourue due à l’insertion du site : pénalité 2,

- L'augmentation du coût due à la création éventuelle d'une nouvelle tournée pour l'insertion du site (on cherche ici à minimiser le nombre de tournées) : pénalité 3.

Le coût d'insertion d'un site correspond à la somme des pénalités 2 et 3. La meilleure insertion pour chaque nœud est retenue. On mémorise alors dans la liste de candidats les 3 nœuds qui ont les plus grandes pénalités d'insertion et on choisit aléatoirement parmi cette liste le site qui va être inséré dans la solution courante. On répète ceci tant qu'il existe des sites dont les demandes ne sont pas satisfaites entièrement.

Pour effectuer la deuxième étape de la méta-heuristique nous avons tout d'abord effectué une recherche locale "classique" : au sein de chaque tournée nous essayons d'échanger les sites planifiés entre eux, dans toutes les tournées planifiées un même jour nous essayons d'échanger tous les sites et au sein des tournées réalisées le même jour nous réalisons également le déplacement des sites au sein des tournées du même jour. Puis, à la place de cette recherche locale nous avons utilisé LNS.

2.2 Résolution du problème avec avance

La phase de construction d'une solution reste la même que celle pour la résolution sans avance. La pénalité et le coût d'insertion sont enrichis d'une autre pénalité (pénalité 4) qui correspond au coût de stockage engendré par l'insertion du site. Dans un premier temps nous effectuons la même recherche locale "classique" que dans la méthode sans avance, à laquelle nous ajoutons une recherche locale visant à regrouper les sites sur les journées afin de livrer avec anticipation (si le coût de stockage est moins fort que le coût de routage sur une journée ultérieure). Le surcoût engendré par un déplacement de livraison du jour j au jour $j - 1$ est le coût de stockage et le gain est le coût de routage et éventuellement le coût de création de la tournée. Dans un second temps nous appliquons LNS à la solution construite grâce à la première phase de la méta-heuristique GRASP.

2.3 Les instances

Nous utilisons comme base les données des instances de Solomon concernant des VRPTW à 25 clients, auxquelles nous ajoutons les valeurs dont nous avons besoin sur les états des différents stocks. Chaque instance possède ces caractéristiques : un dépôt, 5 jours de planification et 4 produits par palette. Nous créons ensuite, trois catégories de sites : les *petits*, les *grands* et les *très grands*. Les caractéristiques des différentes catégories de sites sont :

- Les *petits sites* : ils ont une capacité de stockage de 50 produits, une demande journalière de 20 produits, un taux de retour de 2 produits par jour. Leur stock de palettes vides au début de la simulation est de 5 et le stock de produits en retour est de 2.
- Les *grands sites* : ils ont une capacité de stockage de 100 produits, une demande journalière de 40 produits, un taux de retour de 4 produits par jour. Leur stock de palettes vides au début de la simulation est de 10 et le stock de produits en retour est de 4.
- Les *très grands sites* : ils ont une capacité de stockage de 150 produits, une demande journalière de 80 produits, un taux de retour de 8 produits par jour. Leur stock de palettes vides au début de la simulation est de 20 et le stock de produits en retour est de 8.

3 Conclusions

Nous résolvons ici un problème de collectes et livraisons avec gestion du stock sur plusieurs jours avec une méta-heuristique. Cette méta-heuristique est hybridée grâce à LNS et à la technique de programmation par contraintes. Dans le cadre de cet article, nous présenterons plus en détails les différents résultats obtenus pour les deux stratégies avec les deux méthodes de résolution et comparerons la méthode classique du GRASP avec celle hybridée.

Références

1. Feo T.A., and Resende M. : A Probabilistic heuristic for a Computationally Difficult Set Covering Problem. *Opérations Research Letters* 8,67-71, (1988)
2. Shaw P. : Using constraint programming and local search methods to solve vehicle routing problems. Technical report, ILOG S.A., Gentilly, FRANCE (1998)