Optimiser la distribution de repas

Frantzen Christian, Küpper Marius, Baes Akira, Palmieri-Adant Emile

17 décembre 2015

Introduction

L'optimisation de la distribution des repas est un problème qui s'apparente fort au *Dial-A-Ride-Problem* (DARP). Il a fortement été étudié dans le but d'optimiser le transport des personnes âgées ou inaptes au déplacement. Dans le cadre de ce projet, ce sont des repas que l'on doit transporter des cuisiniers jusqu'aux clients. Ce problème implique un troisième personnage, le livreur. Le défi ici consiste à prendre en compte les contraintes de chacun et trouver une route de moindre coût qui satisfait les demandes de tous les clients.

Nous allons donc étudier le DARP plus en détail dans la prochaine section.

Description d'un DARP

Dans un DARP il y a n clients qui requièrent un transport depuis un point de départ jusqu'à une destination désirée et m véhicules qui réalisent les tournées. Un client spécifie soit l'heure de départ désirée (requête *outbound*), soit l'heure d'arrivée désirée (requête *inbound*) avec parfois la déviation maximale de cet instant. À l'aide de ces informations, des fenêtres de temps sont construits dans lesquelles le véhicule doit passer pour respecter ces contraintes. Chaque véhicule possède une capacité maximale et une durée maximale pour sa tournée. Chaque client a un temps de transport maximum (on ne peut pas faire trop attendre un client déjà embarqué).
Il existe deux types différents de DARP, *statique* et *dynamique*. Un DARP statique part de l'idée que toutes les requêtes sont connues avant qu'un véhicule ne commence sa tournée. Dans un DARP dynamique, les requêtes ont lieu graduellement au cours du temps. Le trajet de la tournée est donc déterminé en temps réel. Nous allons d'abord considérer un modèle statique dans le cadre de ce projet.

Notre sujet en tant que DARP

Dans notre projet il y a n repas qui seront transportés de leur cuisinier à leur destinataire et m véhicules qui réalisent les tournées. On peut voir le problème en tant que points de départ et points d'arrivée. Dans notre cas, que les heures d'arrivées sont spécifiées pour les repas, les heures de départ du transport sont calculées à l'aide de celles-ci. Ce ne sont donc que des requêtes *inbound*. Les véhicules (livreurs) possèdent une capacité maximale et la tournée d'un livreur a une durée limite tout comme le transport d'un repas.

Formulation du problème

Basé sur le doctorat de Jang-Jei Jaw [1] : "Solving large-scale dial-a-ride vehicle routing and scheduling problems".

À chaque repas est associé une heure d'arrivée désirée, une déviation maximale de cette heure d'arrivée, une durée de transport maximale et les sommets du cuisinier et du client dans le graphe. À chaque véhicule est associé l'heure de départ de sa tournée et la durée de celle-ci, ainsi que la nombre maximale de repas qui peuvent être transportés en même temps et le sommet de départ de sa tournée, qui est aussi le sommet d'arrivée auquel il doit retourner à la fin de sa tournée, nommée dépot. Dans notre cas, les distances sont équivalentes au temps de déplacements, c-à-d pour trouver les meilleures solutions on se base sur les distances entre les sommets du graphe et on essaye donc de minimiser celles-ci.

Les différentes approches

Tabu search

Les algorithmes tabu search commencent avec une solution initiale qu'ils essayent d'améliorer, itération par itération. On le nomme 'tabu' parce qu'on doit interdire (ou déclarer 'tabu') les solutions déjà visitées pour ne pas entrer dans un cycle. Les opérations qui sont effectuées pour améliorer une solution sont :

- enlever un couple (départ, destination) d'une route et l'insérer dans une autre route
- décaler un noeud soit de départ, soit de destination dans sa route
- permuter deux couples (départ, destination) de deux routes différentes

Pour la construction d'une solution initiale, les départs des requêtes sont répartis aléatoirement sur les différents véhicules et les destinations correspondantes sont ajoutées à la fin de la route de leur départ. Le voisinage d'une solution est l'ensemble de solutions pouvant être atteint par une simple opération : enlèvement d'un couple (départ, destination) d'une route et insertion dans une autre route. Les nouvelles solutions sont évaluées en fonction du coût et des contraintes (la capacité des véhicules, les fenêtres de temps, la durée limite d'un véhicule et la charge maximale d'un véhicule). Plus une solution viole de contraintes, plus elle est évaluée comme non favorable. Après un certain nombre d'itérations, il faut réarranger les nœuds pour chaque route tel que le coût de la route est réduit le plus possible. (<- A tabu search heurisitic for ...) Dans la recherche actuelle, un algorithme qui s'appelle 'Granular Tabu Search' donne des bons résultats. Un voisinage granulaire est utilisé. Il s'agit d'un voisinage réduit ; c'est à dire il ne contient pas des solutions qui sont suspectées d'amener à une aggravation des coûts et des violations des contraintes. Une solution appartenant au voisinage est décidé par un 'threshold' (en français 'barrière'). Le critère de cette barrière est le temps de voyage. Seulement les solutions dont le temps de voyage est en-dessous de la barrière restent dans le voisinage. Il peut arriver qu'on ne trouve aucune solution qui réponde aux exigences de la barrière, on peut donc augmenter la barrière.(<- Granular Tabu Search)

Branch-and-cut

Branch-and-Cut (BAC) est une approche générique d'optimisation combinatoire. Il a été montré qu'un algorithme de type BAC peut résoudre un DARP ainsi que d'autres problèmes de routage. La première étape consiste à réduire la taille du problème. La seconde utilise un procédé appelé « relaxation continue » pour identifier certaines inégalités encadrant le problème qui seraient violées. Le but étant de transformer ce problème NP-hard en un problème qui est solvable en un temps polynomial. Le rétrécissement du problème effectué lors de la première étape se fait en premier lieu par une contraction de la fenêtre de temps dépendamment d'une requête outbound ou inbound. Deuxièmement, il est possible de supprimer un certain nombre d'arcs du graphe qui à coup sûr ne font pas parti d'une solution correcte. Enfin, il est également possible que certains clients ne soient pas compatibles avec d'autres. Il est important de réduire le problème à sa taille minimale pour éviter des calculs inutiles, atténuant ainsi le phénomène d'explosion combinatoire. La deuxième étape est un procédé heuristique consistant à identifier les solutions qui violent les inégalités précédemment mentionnées suivant une méthode de BAC.

Cette technique algorithmique appliqué au DARP a l'avantage d'être très efficace pour résoudre des petites instances du problème. Elle a également l'avantage de tendre vers l'optimum. Pour des plus larges sets de données, elle devient beaucoup moins intéressante.

Branch-and-cut

Branch-and-Cut (BAC) est une approche générique d'optimisation combinatoire. Il a été montré qu'un algorithme de type BAC peut résoudre un DARP ainsi que d'autres problèmes de routage. La première étape consiste à réduire la taille du problème. La seconde utilise un procédé appelé « relaxation continue » pour identifier certaines inégalités encadrant le problème qui seraient violées. Le but étant de transformer ce problème NP-hard en un problème qui est solvable en un temps polynomial. Le rétrécissement du problème effectué lors de la première étape se fait en premier lieu par une contraction de la fenêtre de temps dépendamment d'une requête outbound ou inbound. Deuxièmement, il est possible de supprimer un certain nombre d'arcs du graphe qui à coup sûr ne font pas parti d'une solution correcte. Enfin, il est également possible que certains clients ne soient pas compatibles avec d'autres. Il est important de réduire le problème à sa taille minimale pour éviter des calculs inutiles, atténuant ainsi le phénomène d'explosion combinatoire. La deuxième étape est un procédé heuristique consistant à identifier les solutions qui violent les inégalités précédemment mentionnées suivant une méthode de BAC.

Variable Neighborhood Search

Un Variable Neighborhood Search part d'une solution initiale faisable et génère des solutions au voisinage. Chaque solution est comparée à la solution initiale et peut la remplacer si les pénalités sont moindres. Le problème est que l'on doit partir d'une solution initiale faisable pour avoir de bon résultats, et générer ce genre de solution peut être lourd pour de gros problèmes (avec beaucoup de requêtes, 1000). Le VNS comparé à une recherche Tabu (Cordeau

2003) trouve plus de solutions faisables, de meilleure qualité. Une variante, le Distributed VNS, regroupe les requêtes similaires par médoïdes (un partitionnement autour du représentant central d'une classe, plus robuste qu'une moyenne vis-à-vis des données aberrantes), puis lance des VNS en parallèle sur chacun des clusters, facilitant la recherche de solutions faisables. Les routes (groupes de requêtes) des solutions partielles sont de nouveau partitionnées par médoïdes et redistribuées à des VNS parallèles jusqu'à trouver une solution acceptable (où à une limite de temps de calcul). Le principe est donc de diviser l'espace des solutions en sous-espaces à explorer en parallèle tout en échangeant des informations entre les processus explorants pour optimiser la recherche. Cet algorithme a l'avantage d'être distribué et donc parallélisable. Le dVNS peut être lancé sur des problèmes plus gros où le VNS échoue (16000 requêtes) et donne des résultats de meilleure qualité. (Voir A variable neighborhood search algorithm for the optimization of a dial-a-ride problem in a large city (Santiago Muelas et al. 2013) et A distributed VNS algorithm for optimizing dial-a-ride problems in large-scale scenarios (Santiago Muelas et al. 2014))

Descriptions d'articles de référence

Solving large-scale dial-a-ride vehicle routing and scheduling problems (Jang-Jei Jaw) 1984

Dans son doctorat, Jang-Jei Jaw décrit *« an algorithm for DARP with stric service constraints »*. Cet algorithme "multi-véhicule" se veut être optimal tout en respectant les contraintes des clients. Il garantit donc que les fenêtres de temps spécifiées par les clients soient inviolées. Il s'agit d'une méthode heuristique, ce qui implique que les solutions proposées ne sont pas forcément optimales mais elles ne sont pas mauvaises pour autant, au contraire. Les méthodes exactes résolvant ce problème optimalement ont été montrées comme étant de complexité *NP-hard*. Cette procédure heuristique permet, en conséquence, de générer des solutions à coût nettement plus faible.Cet algorithme exploite les concepts d'un algorithme à insertion permettant de résoudre des cas plus simples de ce problème. Il va donc chercher à insérer un par un les clients dans les plages horaires de chaque véhicule tout en respectant les contraintes du client ainsi que de chaque autre client présent dans le véhicule au moment de l'insertion. S'en suivra une série d'optimisations réalisées de manière à minimiser le coût de l'insertion. Ce coût est quant à lui calculé grâce à une fonction de désutilité. Ici, « les clients sont des repas ». C'est-à-dire qu'au lieu de transporter des clients, ce sont des repas qui doivent être acheminés du cuisinier jusqu'au clients qui les ont commandés. Dans le problème original, le client a le choix entre spécifier soit l'heure de départ soit l'heure d'arrivée mais pas les deux. Dans cette version modifiée, le client ne peut que préciser l'heure d'arrivée des repas, ce qui fait de cette requête une requête *outbound*.
Nous avons choisi d'implémenter une version de l'algorithme d'insertion de J-J. Jaw 1984, car il s'agit d'une approche heuristique permettant de résoudre de large problèmes en considérant des fenêtres de temps, tout en produisant une bonne solution, bien que non-optimale. Cet algorithme est référencé par un grand nombre d'articles scientifiques traitant le DARP, ce qui en fait une source très fiable.

A tabu search heuristic for the static multi-vehicle dial-a-ride problem (Jean-François Cordeau et Gilbert Laporte) 2002

Dans cet article est décrit un algorithme de recherche tabou ('Tabu Search'). Leur approche consiste à établir un ensemble de solutions initiales sans se préoccuper des contraintes de temps et de charge. Est ensuite déterminé une fonction de coût régie par plusieurs paramètres ajustables ce qui permettra de construire les solutions. Le fait d'être souple quant à la violation des contraintes initiales est caractéristique de cet algorithme décrit dans cet article. Il se base sur l'évaluation de voisinages pour construire un trajet optimal minimisant le temps total de trajet ainsi que la durée d'un trajet pour un client. L'article décrit un algorithme 'Tabu Search'.

A new regret insertion heuristic for solving large-scale dial-a-ride problems with time windows (Marco Diana et Maged M. Dessouky) 2002

Dans leur article, les auteurs décrivent une solution du DARP en tenant compte de contraintes horaires (time windows) et de la situation géographique des différents clients. Les time windows admissibles de chaque requête sont définis à l'aide des données des clients (heure de départ souhaitée ou heure d'arrivée souhaitée) et d'un temps de service à chaque arrêt, vu que les auteurs se concentrent particulièrement sur le transport de personnes à mobilité réduite. Cet article est intéressant pour notre projet car on y décrit une façon comment choisir un bon point de départ de chaque route pour les différents véhicules et on y décrit une façon comment insérer de nouvelles requêtes dans des routes déjà établies en profitant du système des time windows.

A variable neighborhood search algorithm for the optimization of a dial-a-ride problem in a large city (Santiago Muelas et al. 2013)

Amélioration d'un VNS de Parragh et al (2010) Le principe de cette métaheuristique est d'explorer des solutions voisines à une solution initiale générées aléatoirement. L'algorithme doit cependant partir d'une solution initiale possible, et les résultats obtenus sont très dépendants des conditions initiales. Pour de grands problèmes (1000 requêtes), partir d'une situation initiale aléatoire peut ne pas mener du tout à une solution faisable.

À chaque itération, on génère une solution aléatoire voisine à la solution initiale. Un Local Search algorithm est appliqué aux solutions les plus prometteuses pour générer une solution propre. Si elle est meilleure, on relance l'algorithme sur elle. On arête la génération de solutions voisines quand on a une solution que l'on trouve satisfaisante.

Comparé à un Cordeau Tabu Search 2003 et à un VNS précédent, cette méthode trouve toujours les solutions faisables (même avec 1000 requêtes), à condition de partir d'une solution initiale presque faisable (générée par insertion), et trouve en général de meilleures solution.

A distributed VNS algorithm for optimizing dial-a-ride problems in large-scale scenarios (Santiago Muelas, Antonio LaTorre et José-María Peña) 2014

Dans leur article, les auteurs décrivent une façon pour résoudre des DARP de grandes tailles. Pour maîtriser la taille du problème, les différents requêtes qui sont proches d'un point de vue géographique et horaire sont regroupés, puis ces groupes sont traités comme sous-problèmes. Chaque sous-problème est résolu et la route optimale de sa solution est renvoyé à l'algorithme appelant. Chaque groupe est traité indépendant des autres groupes. La construction de la solution finale est construite à partir des différentes sous-solutions du problème. L'article est pertinent pour notre projet parce qu'il montre une façon de traiter un DARP de grande taille.

The dial-a-ride problem : models and algorithms (Jean-François Cordeau et Gilbert Laporte) 2007

Cet article résume la littérature scientifique à propos des DARP. Il donne un aperçu des différents algorithmes qui ont été publiés pour attaquer ce problème, chacun avec une description des contraintes (time windows, véhicules, ...) et une nombre n d'utilisateurs que l'algorithme est capable de gérer dans un temps raisonnable. Cet aperçu peut nous servir pour apprécier quelle structure l'algorithme convient le plus pour résoudre notre problème. Par ailleurs les différents modèles mathématiques sont présentés et la différence entre le modèle statique et le modèle dynamique est expliquée.

A Granular Tabu Search algorithm for the Dial-a-Ride Problem (Dominik Kirchler et Roberto Wolfer Calvo) 2013

Dans cet article, un algorithme 'Granular Tabu Search' est présenté. C'est une amélioration de l'algorithme 'Tabu Search' décrit par l'article de référence. 'Granular', granulaire en français, parce que les voisinages qui sont utilisés pour l'amélioration de la solution initiale sont sous forme réduite, c'est-à-dire que les trajets qui ont une faible probabilité d'appartenir à de bonnes solutions sont exclus. L'article peut nous fournir des informations pour améliorer notre algorithme.

Recent Models and Algorithms for One-to-One Pickup and Delivery Problems revised (Jean-Francois Cordeau, Gilbert Laporte et Stefan Ropke) 2007

Dans cet article sont décrits des algorithmes (2007) pour résoudre les PDP (Pickup & Delivevy Problem), dont un branch-and-cut pour DARP développé par Cordeau et Ropke en pages 16-19. Branch-and-cut est présenté comme plus efficace pour donner une solution optimale sur de petits problèmes. Dans ce modèle où tous les véhicules sont identiques, on pose une variable à minimiser avec une série de nouvelles contraintes et d'inégalités sur lesquelles on applique un algorithme branch-and-cut. C'est une version améliorée du Branch-and-cut Cordeau de 2003. L'article semble présenter l'état de l'art de l'époque en matière de Branch-and-cut pour DARP.

A Branch-and-Cut Algorithm for the Dial-a-Ride Problem (Jean-Francois Cordeau) 2003

Dans cet article est décrit la première version d'un Branch-and-cut pour DARP par Cordeau peu après son article sur la méthode heuristique tabu. Sont présentés une somme à minimiser sous de nombreuses contraintes et inégalités qui décrivent les limitations du problème. L'article peut nous informer sur les débuts de la méthode Branch-and-cut.

Large Neighborhood Search For Dial-a-Ride Problems (Siddhartha Jain et Pascal Van Hentenryck) 2011

Cet article présente une variante "First Feasible Probabilistic Acceptance" de la méthode appelée Large Neighborhood Search qui fait une recherche de voisinage sans fixer de contrainte d'amélioration des solutions partielles jusqu'à trouver des solutions faisables qui sont choisies sur bases probabilistes. La méthode est dite plus efficace dans l'article pour des recherches à contraintes à temps limité que les méthodes tabu de l'époque (2011), le rendant approprié pour du DARP semi-dynamique, et donne des résultats non-optimaux comparables aux algorithmes état-de-l'art de l'époque. Cet article peut nous donner une autre piste dans la recherche d'un algorithme efficace de résolution de DARP.

Bibliographie

[1] Jang-Jei Jaw, Solving large-scale dial-a-ride vehicle routing and scheduling problems, *FTL report (Massachusetts Institute of Technology. Flight Transportation Laboratory)*, R84-3 (1984), 90-131

[2] A tabu search heuristic for the static multi-vehicle dial-a-ride problem (Jean-François Cordeau, Gilbert Laporte) 2002

[3] A new regret insertion heuristic for solving large-scale dial- a-ride problems with time windows (Marco Diana , Maged M. Dessouky) 2002

[4] A variable neighborhood search algorithm for the optimization of a dial-a-ride problem in a large city (Santiago Muelas et al. 2013)

[5] A distributed VNS algorithm for optimizing dial-a-ride pro- blems in large-scale scenarios (Santiago Muelas , Antonio LaTorre , José-María Peña) 2014

[6] The dial-a-ride problem : models and algorithms (Jean- François Cordeau et Gilbert Laporte) 2007

[7] A Granular Tabu Search algorithm for the Dial-a-Ride Problem (Dominik Kirchler, Roberto Wolfer Calvo) 2013

[8] Recent Models and Algorithms for One-to-One Pickup and Delivery Problems revised (Jean-Francois Cordeau, Gilbert Laporte et Stefan Ropke) 2007

[9] A Branch-and-Cut Algorithm for the Dial-a-Ride Problem (Jean-Francois Cordeau) 2003

[10] Large Neighborhood Search For Dial-a-Ride Problems (Siddhartha Jain et Pascal Van Hentenryck) 2011