Approche décentralisée pour résoudre le problème du transport à la demande

Michel Nabaa, Besma Zeddini, Pierrick Tranouez

▶ **To cite this version:**

Michel Nabaa, Besma Zeddini, Pierrick Tranouez. Approche décentralisée pour résoudre le problème du transport à la demande. Majestic 2007, Oct 2007, Caen, France. pp.Proceedings sur CD. hal-00422630

HAL Id: hal-00422630
https://hal.science/hal-00422630

Submitted on 8 Oct 2009

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

MajecSTIC 2007

Caen

Approche décentralisée pour résoudre le problème du transport à la demande

Michel Nabaa, Besma Zeddini, Pierrick Tranouez

Laboratoire d'Informatique, de Traitement de l'Information et des Systèmes(LITIS)

Michel.Nabaa@gmail.com, Besmile.Zed@gmail.com, Pierrick.Tranouez@univ-lehavre.fr

Résumé:

Les systèmes de transports publics actuels sont déterministes. Dans de tels systèmes, c'est la demande qui s'adapte à l'offre. Dans cet article, nous proposons un modèle d'auto-organisation pour la répartition de véhicules dans un système de transport à la demande (TAD) purement dynamique. Notre proposition s'appuie sur une approche décentralisée et une modélisation de l'environnement dans un système multiagent (SMA). Il s'agit de limiter la concentration des véhicules ou la famine dans certaines régions de la ville. Nous montrons que notre modèle répond à la problématique posée dans ce domaine en permettant aux agents véhicules, pour une demande donnée, de prendre la décision finale au moyen d'un processus de négociation et de calculer des surcoûts suivant une heuristique d'insertion originale.

Mot-clés : Transport à la demande, optimisation de tournées de véhicules, intelligence collective, auto-organisation.

Abstract:

Public transport systems are deterministic. In such systems, the demand must be adapted to the offer. In this paper, we propose a model based on self-organization in order to dispatch a fleet of vehicles in a purely dynamic transport on demand system (ODT). Our proposal consists in a decentralized approach and a multi-agent system (MAS) to model the environment.This will avoid the problem of vehicles concentration or the lack of service in certain areas of the city. We demonstrate that our model addresses these problems by allowing vehicle agents, for a given request, to make the final decision thanks to a negotiation process and to calculate overcosts according to an original insertion heuristic.

Keywords : transport on demand, vehicule routing problem, collective intelligence, self - organization.

1 Introduction

Des préoccupations environnementales grandissantes conjuguées à des difficultés de gestion de trafic notamment urbain poussent à la recherche de nouvelles solutions améliorant à ces égards la mobilité en agglomération. Les systèmes de transports publics actuels sont déterministes et basés sur des fréquences et des itinéraires fixés à l'avance. Ils sont construits à partir d'études d'opportunité et sont peu adaptatifs à une demande très changeante, c'est la demande qui s'adapte à l'offre. Nous proposons un système de transport à la demande qui doit à l'opposé adapter l'offre, en temps réel, à la demande. Il doit permettre à terme de générer une forte diminution du trafic et d'offrir une qualité de service maximale à un moindre coût d'exploitation. Enfin, il constituera un outil d'aide à la décision permettant de calculer en temps réel des problèmes de construction de tournées de véhicules, ce qui n'est pas offert par les systèmes de transport classiques. L'enjeu de cet article est d'étudier la possibilité de la mise en place d'un système de transport à la demande (TAD) que l'on compte appliquer sur la ville de Rouen pour essayer de satisfaire, au mieux et à tout instant, les demandes des clients, tout en répartissant la charge entre la flotte de véhicules afin d'atteindre les objectifs cités précédemment. Ce système ajustera dynamiquement à une fréquence optimisée l'offre à la demande. Le scénario d'exécution commence avec la première demande d'un usager qui apparaît aléatoirement dans un endroit de la ville. Il envoie une requête en désignant son point de départ et son point d'arrivée. La résolution consiste à choisir le véhicule le mieux placé pour satisfaire les demandes de déplacement des passagers déjà existants dans ce véhicule ainsi que la nouvelle demande ; et ceci en essayant d'optimiser son taux de remplissage et son temps de parcours.

2 Positionnement scientifique

Le problème général de construction de tournées des véhicules est connu sous le nom de Vehicle Routing Problem (VRP) et représente un problème d'optimisation combinatoire multi-objectif qui a fait l'objet de nombreux travaux et de nombreuses variantes dans la littérature. Il appartient à la catégorie NP-difficile [Christofides 79, Lenstra 81]. Dans sa version de base, le problème VRP (voir figure 1) modélise un problème de transport très répandu qui consiste à livrer (et/ou collecter) des produits auprès d'un ensemble de clients à l'aide d'une flotte de véhicules. La résolution consiste à déterminer un ensemble de tournées qui minimise au mieux des objectifs comme la distance totale parcourue, le nombre de véhicules utilisés, la somme des retards des clients [Laporte 99]. Un état de l'art complet

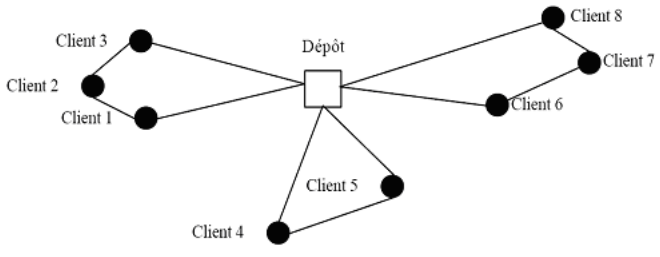

FIG. 1 – Problème de tournées de véhicules

des problèmes de VRP dans un contexte statique, en particulier, dynamique (DVRP) et de leurs applications est donné dans [Larsen 00]. Quand il s'agit du problème de VRP avec des collectes et livraison de marchandises on parle de PDP (Pickup & Delivery Problem ou Problème de Ramassage & Livraison). Le cas d'un PDP dynamique a été étudié dans

[Koźlak 06]. Les auteurs ont adopté une approche multi-agents. La conversation entre les agents est basée sur le protocole Contract Net. L'arrivée en temps des clients est calculée selon la loi de Poisson avec un coefficient pouvant changer pendant la simulation. Un DARP (Dial-a-Ride Problem) est une extension du PDP dans lequel le transport de marchandises est remplacé par le transport des personnes [Krumke 06, Hauptmeier 00]. Puisqu'il s'agit d'un problème de transport de personnes, le DARP se focalise plus sur la satisfaction de ces personnes. Le DARP se décline à la fois de façon statique et dynamique. Souvent, ce cloisonnement n'est pas toujours strict, car par exemple, dans le cas statique, des réservations peuvent être annulées au dernier moment, ce qui implique un certain degré de dynamisme tandis que dans le cas dynamique, beaucoup de réservations sont connues à priori avant la planification [Attanasio 04]. Le problème est alors en général traité comme une succession de sous problèmes statiques [Cordeau 04]. Une application dans le contexte des transports urbains concerne le bus "à la demande" : les clients peuvent donner une fenêtre de temps au lieu de départ ou au lieu d'arrivée, mais pas les deux en même temps. Des demandes sont connues par avance et d'autres peuvent arriver pendant le jour même. Les auteurs ont adopté un algorithme d'insertion qui a donné des résultats intéressants avec des durées d'exécution courtes [Madsen 95]. Ce problème est connu sous le nom de DDARP : Dynamic Dial-A-Ride Problem dans un contexte Many-to-Many (i.e plusieurs lieux de collectes et de dépôts). Dans [Lipmann 04], le problème de DARP a été traité en ligne en considérant une flotte de véhicules homogène de capacité unitaire, i.e un véhicule ayant un passager à bord, ne peut servir un autre passager que s'il est arrivé à la destination du premier. Le problème de DARP dynamique est étudié d'une façon marginale faute d'applications et il n'existe pas, à notre connaissance, en version purement dynamique. Le problème qui se pose dans cet article est original, du fait que les demandes sont traitées en temps réel : nous traitons un cas purement dynamique. Cette dynamicité vient du fait qu'aucune réservation ne peut être anticipée. De plus, les techniques classiques supposent d'avoir un central qui connaît en permanence la position, les trajectoires et les dessertes prévues de toute la flotte (le central reçoit la demande, choisit le transporteur et l'informe de sa nouvelle trajectoire). En outre, ces techniques supposent une connaissance parfaite par le central de l'actualité des véhicules (y compris pauses, pannes, problèmes de communication . . .), ce qui est peu réaliste. Ajoutons que ce fait peut entraîner des calculs coûteux en temps. Ce qui nous pousse à adopter une méthode de résolution décentralisée pour faire face à ce genre de problèmes, et qui pourraît être supportée par de l'informatique embarquée amenée à se développer de plus en plus.

3 Modèle dynamique proposé

3.1 Description du problème

Le modèle qu'on va proposer essaie d'arbitrer entre différentes contraintes.

Chaque usager souhaite :

- Être pris rapidement en charge une fois sa demande acceptée,
- Être amené à destination dans des délais n'excédant "'pas trop"' le temps minimal pour aller de leur provenance à leur destination,
- ne pas voir son temps de transport augmenter trop brutalement (écarts trop manifestes par rapport à sa perception d'un bon itinéraire, demi-tours . . .) .

Pour chaque véhicule, les transporteurs essaient :

- D'optimiser le taux de remplissage en dérivant d'un itinéraire déjà planifié pour prendre en charge une nouvelle demande,

– De prendre en charge l'évolution de la charge du trafic et surtout les événements inattendus (accident, route bouchée, nouvelle route) et historiques : le système doit avoir une capacité d'apprentissage,
– De négocier avec les autres véhicules afin de choisir celui qui est le mieux positionné pour servir une nouvelle demande.

Pour l'ensemble, le système s'efforce d'apparier au mieux usagers et transporteur en arbitrant et ajustant les contraintes précédentes. Le système n'est pas centralisé en un serveur ou assimilé, mais émergeant de la flotte des véhicules.

Nous proposons une approche agentifiée. Le système est composé des agents suivants : `agent Véhicule`, `agent Interface` et `agent Client`. Le scénario d'exécution est décrit comme suit : un usager se connecte au système via un support donné (central d'appel, serveur Web...), il est alors instancié par un `agent Client` qui a pour fonction de le représenter au sein du système. L'utilisateur désigne son point de départ et son point d'arrivée comme indiqué précédemment. Ces informations seront traduites en des données convenables à l'ontologie du service. Ainsi, `l'agent Client` entre en interaction avec l'`agent Interface` (voir figure 2). Ce dernier, diffuse la demande de l'usager aux `agents Véhicules` situés dans un environnement qui est modélisé dans la section suivante. Ce qui diffère dans notre modèle des systèmes de TAD dynamiques existants,

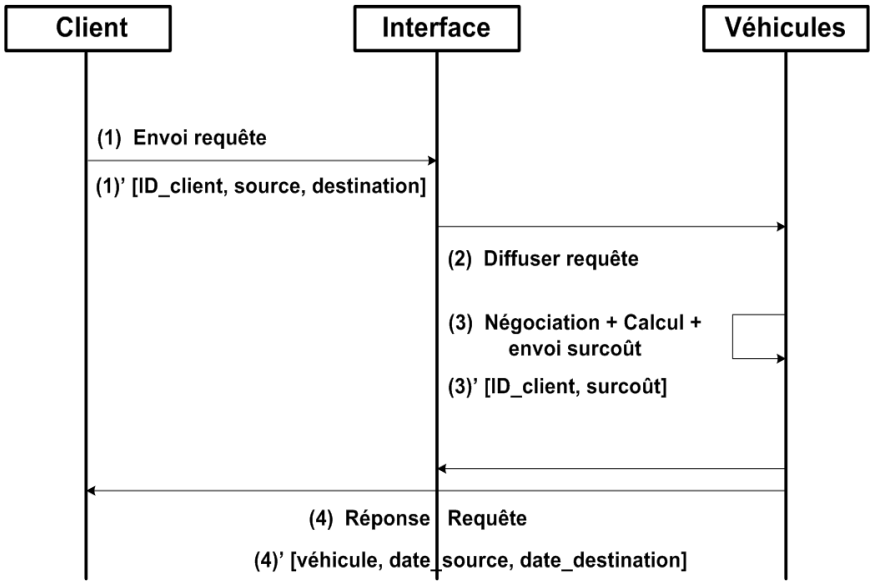

FIG. 2 – Diagramme UML montrant la séquence d'interactions entre les acteurs du système

c'est que les demandes ne sont pas traitées par lot mais en "temps réel".

3.2 Modélisation de l'environnement

Nous représentons le réseau urbain par un graphe dynamique $G(t) = (V(t), E(t))$ orienté et pondéré où $V(t)$ est l'ensemble de nœuds et $E(t)$ l'ensemble des arcs :

– Les nœuds de ce graphe représentent des lieux topologiquement intéressants du réseau : intersections, gares, cinémas, centre commerciaux ...
– Les arcs représentent les voies empruntables par les véhicules menant d'un lieu à l'autre,
– La valuation des arcs représente le temps nécessaire au franchissement de celui-ci, vue la charge actuelle du trafic,
– La dynamique porte sur la valuation des arcs, qui est susceptible d'évoluer dans le temps, en fonction de l'évolution de la fluidité de la circulation par exemple. La

dynamique pourrait également s'étendre ultérieurement à l'existence des sommets et des arcs.

Les usagers sont attribués à un sommet ou à un transporteur. Ces derniers sont sur des sommets ou des arcs. La taille des populations d'usagers et de transporteurs sont variables au cours du temps, pour simuler au minimum un mécanisme jour/nuit par exemple. Une fois un modèle temporel de la population d'usagers établi, la population de transporteurs doit être établie en fonction, afin d'avoir un taux moyen d'occupation satisfaisant.

3.3 Offres et choix dynamiques

Notre modèle est basé sur deux phases simultanées, une phase d'offres et une phase de choix. On cherche à établir un accord entre les propositions de transport et les intérêts du client. Un élément clef du système est l'appariement entre transporteur et usager. Quel est le meilleur transporteur pour une demande donnée ? Qui le détermine et comment ? Comment le transporteur sait-il qu'il a été choisi ? Ces questions ne sont d'ailleurs pas indépendantes : si par exemple le transporteur a pu déterminer lui-même qu'il était le meilleur, la question de le mettre au courant ne se pose même pas. Le transporteur qui correspondra à l'usager sera choisi en essayant de minimiser l'effort supplémentaire $\triangle C$ pour le transporteur. Pour connaître cet effort supplémentaire, on calcule la longueur (en temps) de l'itinéraire actuel (le coût) d'une part, celle de l'itinéraire pour décharger les actuels passagers plus la prise et la desserte de l'usager ayant fait une demande d'une autre part, et on soustrait la première à la seconde. Cette différence est l'effort supplémentaire ou *surcoût*.

Déterminer le transporteur choisi consiste alors à sélectionner celui de moindre surcoût parmi ceux que prendre en charge le nouveau demandeur n'obligerait pas à bafouer la date limite de desserte d'au moins un usager déjà embarqué. la demande est diffusée à tous les transporteurs. Ceux-ci calculent alors chacun leurs surcoûts qu'ils diffusent à tous. Ils ordonnent les réponses reçues puis diffusent la tête de liste. Le vainqueur déterminé par chaque transporteur est celui qui aura été le plus souvent classé premier dans les réponses reçues. Ce vainqueur gagne le nouvel usager. Idéalement on pourrait se dispenser de ces dernières étapes : si la diffusion est parfaite, tous les transporteurs obtiendront directement le bon classement. Mais c'est parce qu'en réalité la diffusion ne sera pas parfaite qu'on procède à cette étape supplémentaire, pour être sûr que tous les transporteurs soient d'accord sur qui prendra le passager.

3.4 Algorithme d'ordonnancement

Les agents véhicules effectuent le travail principal de planification par évaluation de l'insertion d'un voyage (source et destination) de telle façon à respecter le plus possible les échéances des passagers existants dans les véhicules. La politique d'insertion est similaire à ADARTW [Jaw 86]. Pour chaque véhicule, un bloc d'ordonnancement commence toujours par le premier client sur son chemin et se termine quand le dernier client arrive à destination. La figure suivante illustre l'ajout d'un client dans un bloc d'ordonnancement d'un véhicule contenant déjà trois clients ($C1, C2, C3$) dont l'itinéraire est déjà planifié. Chacun a un lieu de départ (+) et de destination (-). A un bloc relatif à un véhicule et contenant déjà n clients,

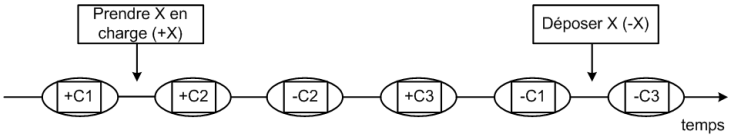

FIG. 3 – Heuristique d'insertion

correspond $k = 2n$ arrêts (prise en charge et dépôt) et il y a $(k+1)(k+2)/2$ possibilités

d'insertion en admettant qu'une prise en charge d'un client doit précéder son dépôt.
La complexité dépend de l'algorithme choisi pour déterminer les plus courts chemins d'un sommet à un autre - nous avons considéré l'algorithme de Djikstra ($O(m+log(n))$ pour m arcs et n nœuds). Elle dépend également de l'heuristique d'insertion qui pour un véhicule de capacité Q_v nous donne dans le pire des cas un coefficient multiplicateur de $Q_v(2Q_v-1)^2$.

La fonction objectif $MinZ = \triangle C$ mesure le surcoût minimal dû à l'insertion d'un nouveau client. Pour le terme $\triangle C$, plusieurs variables ont été considérées : temps supplémentaire, capacité actuelle, proximité du client. Si deux ou plusieurs véhicules ont la même valeur du surcoût pour un même client à un instant t, celui qui a la capacité minimale doit prendre en charge ce client. S'ils ont aussi la même capacité, celui qui est plus proche(en distance) va ramener le client.

3.5 Auto-organisation

La nature dynamique et non déterministe du problème peut conduire à des concentrations de l'offre dans certaines zones plus attractives et de ce fait à la création de zones de famines. En effet le centre-ville sera par exemple une zone d'attraction forte à certaines heures de la journée alors que certaines zones périurbaines dîtes dortoirs seront délaissées. Le temps d'attente dans une telle zone sera alors très important, nous avons donc choisi d'éventuellement dégrader les performances dans les zones attractives et d'avoir une meilleure couverture du domaine par la flotte de véhicules afin d'éviter des zones orphelines.

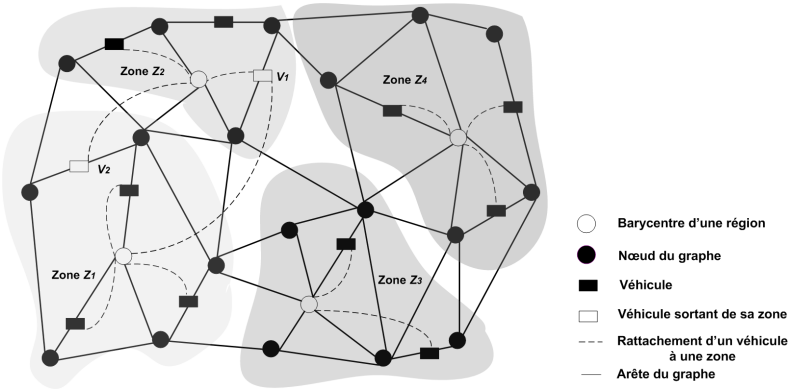

FIG. 4 – Attraction des zones

On définit donc des zones d'attraction au sein du graphe auxquelles les véhicules sont rattachées, ces zones peuvent évoluer par la suite soit au niveau de leur couverture du graphe, soit au niveau du nombre de véhicules rattachés.

Lorsqu'un véhicule quitte sa zone il est pénalisé dans une fonction de coût qui joue comme une force de rappel exercée par un ressort accroché à sa zone (voir figure 4). Si le véhicule n'est plus dans la couverture géographique de sa zone, il a la possibilité de négocier un changement de zone de rattachement avec d'autres. Par exemple, dans la figure 4, les véhicules V_1 et V_2 échangent leur zone. Cette partie de la modélisation est en cours d'implémentation.

4 Simulation

L'architecture multi-agent proposée a été développée en utilisant la plateforme d'agent REPAST Simphony (Recursive Porus Agent Simulation Toolkit) écrite en Java [North 05].

Cette plateforme développée par l'université de Chicago, hérite le principe de fonctionnement de la plateforme SWARM

Algorithme 1 : Comportement de la flotte lors d'un pas de simulation

```
pour chaque vehicule v ∈ flotte faire
    tant que non fin de tournée faire
        message = consultation de boîte à lettre;
        si message! = vide alors
            si message ∈ demande_client alors
                calcul surcoût selon ADARTW pour effectuer l'insertion temporaire du
                client dans le trajet;
                diffuser surcoût ;
            sinon
                si message ∈ surcoût alors
                    classer les surcoûts pour le client;
                    si tous les surcoûts sont arrivés ou timeout declenché alors
                        vainqueur = tête (liste des surcoûts);
                        diffuser le vainqueur;
                sinon
                    si message ∈ un vainqueur alors
                        si tous les vainqueurs sont arrivés ou timeout declenché
                        alors
                            gagnant = vainqueur le plus souvent cité;
                            si gagnant == v alors
                                modifier itinéraire;
        avancer d'un pas sur l'itinéraire prévu
```

Comme déjà cité avant, le calcul de surcoût vis à vis d'une nouvelle demande se fait au niveau des véhicules. À un pas de temps, chaque véhicule reçoit une demande(s'il y en a), collecte les réponses (surcoût des autres véhicules) et compare les surcoûts pour savoir s'il est le mieux placé, puis met au courant l'ensemble du système (demandeur, central et véhicule) avant d'aller à la recherche du demandeur. Dans le cas où il ne reçoit pas la totalité des offres, un véhicule attend un certain délai pour réagir s'il aperçoit qu'il est le vainqueur actuel de l'offre. Toutefois, le fait d'informer les autres avant de répondre à une demande peut éviter que deux véhicules répondent à la même demande positivement.

5 Premiers résultats

Nous avons effectué les premières simulations des fonctionnalités de base de notre modèle sur un graphe à 50 sommets, sur 7200 pas de temps, avec des véhicules à 4 places. Les clients apparaissent à des lieux et heures aléatoires, avec des destinations aléatoires.

Dans le tableau 1, la colonne *coût optimal* indique le temps total minimal obtenu à l'aide du plus court chemin pour emmener tous les clients de leur lieu de prise en charge jusqu'à leur destination. La colonne *Coût réel* indique le temps effectivement mis en prenant en compte la contrainte de remplissage. *Écart* est le pourcentage de différence entre les deux coûts. *Remplissage* indique sommairement entre quelle et quelle valeur a oscillé le remplissage.

Nb véhicules	Nb clients	Coût optimal en temps	Coût réel en temps	Écart	Remplissage
4	225	4422	5693	22%	0–>4
6	225	4575	5628	16%	0–>4
8	225	4708	5475	14%	0–>3

TAB. 1 – Quelques résultats de simulation

L'*écart* est déjà intéressant, car plutôt faible, mais la simulation doit être raffinée : modèle probabiliste plus fin pour la population d'usager, meilleures statistiques pour le remplissage etc. Les autres fonctionnalités du modèle (auto-organisation) doivent être implémentés. Ces premiers résultats sont néanmoins encourageants.

6 Conclusion et perspectives

Dans cet article, nous avons présenté un système de transport à la demande purement dynamique dans un environnement en changement perpétuel. Pour cela, nous avons adopté une approche décentralisée basée sur l'optimisation et la négociation entre les véhicules. Pour faire face aux zones de famine ou, au contraire, de concentration de véhicules dans certaines régions, nous avons proposé un modèle d'auto-organisation pouvant s'adapter aux changements environnementaux (pannes, demandes clients. . .). Les premiers résultats sont encourageants et la phase d'auto-organisation est en cours de développement. Il nous reste à compléter notre travail par la validation complète du modèle proposé.

Références

[Attanasio 04] Andrea Attanasio, Jean-François Cordeau, Gianpaolo Ghiani & Gilbert Laporte. *Parallel Tabu search heuristics for the dynamic multi-vehicle dial-a-ride problem*. Parallel Comput., vol. 30, no. 3, pages 377–387, 2004.

[Christofides 79] N. Christofides, A. Mingozzi & P. Toth. *The vehicle routing problem*. In Combinatorial Optimization, volume 11, pages 315–338. John Wiley, 1979.

[Cordeau 04] Jean-François Cordeau, Gilbert Laporte, Jean-Yves Potvin & Martin W.P. Savelsbergh. *Transportation on Demand*. Handbooks in Operations Research and Management Science, 2004.

[Hauptmeier 00] D. Hauptmeier, S. O. Krumke & J. Rambau. The online dial-a-ride problem under reasonable load, in proceedings of the 4th italian conference on algorithms and complexity, volume 1767. Springer, 2000.

[Jaw 86] J. J. Jaw, Odoni A. R., Psaraftis H. N. & Wilson N. H. M. *A heuristic algorithm for the multi-vehicle many-to-many advance request dial-a-ride problem with time windows*. Pergamon Journals Ltd., vol. 20B, no. 3, pages 243–257, 1986.

[Koźlak 06] Jaroslaw Koźlak, Jean-Charles Créput, Vincent Hilaire & Abder Koukam. *Multi-agent approach to dynamic pick-up and delivery problem with uncertain knowledge about future transport demands*. Fundam. Inf., vol. 71, no. 1, pages 27–36, 2006.

[Krumke 06] S. O. Krumke, W. E. de Paepe, D. Poensgen, M. Lipmann, A. Marchetti-Spaccamela & L. Stougie. On minimizing the maximum flow time in the online dial-a-ride problem, volume 3879. Springer Berlin / Heidelberg, 2006.

[Laporte 99] G. Laporte, M. Gendreau, J.-Y. Potvin & F. Semet. *Classical and Modern Heuristics for the Vehicle Routing Problem*. International Transactions in Operational Research, vol. 7, pages 285–300, 1999.

[Larsen 00] A. Larsen. *The dynamic vehicle routing problem*. PhD thesis, Technical University of Denmark, 2000.

[Lenstra 81] J. Lenstra & A. H. G. Rinnooy Kan. *Complexity of the vehicle routing and scheduling problems*. In Networks, volume 11, pages 221–228. Springer, 1981.

[Lipmann 04] M. Lipmann, X. Lu, W.E. de Paepe, R.A. Sitters & L. Stougie. *On-Line Dial-a-Ride Problems Under a Restricted Information Model*. In Algorithmica, volume 40, pages 319–329. Springer, 2004.

[Madsen 95] O. B. G. Madsen, H. F. Ravn & J. M. Rygaard. *A heuristic algorithm for the a dial-a-ride problem with time windows, multiple capacities, and multiple objectives.* Annals of Operations Research, vol. 60, pages 193–208, 1995.

[North 05] M.J. North, T.R. Howe, N.T. Collier & R.J. Vos. *The Repast Simphony Runtime System*. In Agent 2005 Conference on Generative Social Processes, Models, and Mechanisms, Argonne National Laboratory, Argonne, IL USA, 2005.