Dial-a-ride problem for disabled people using vehicles with reconfigurable capacity

Oscar Tellez, Samuel Vercraene Vercraene, Fabien Lehuédé, Olivier Péton, Thibaud Monteiro

Dial-a-ride problem for disabled people using vehicles with reconfigurable capacity

Oscar Tellez * **Samuel Vercraene** * **Fabien Lehuédé** **
Olivier Péton ** **Thibaud Monteiro** *

* *Laboratoire DISP, INSA de Lyon, Bât Léonard de Vinci, 21 avenue Jean Capelle, CO 69621 Villeurbanne, France (e-mail: {oscar.tellez, samuel.vercraene, thibaud.monteiro}@insa-lyon.fr).*
** *IMT Atlantique, 4 rue Alfred Kastler, F-44307 Nantes Cedex, France (e-mail: {fabien.lehuede, olivier.peton}@imt-atlantique.fr)*

Abstract: The aim of this paper is to address the dial-a-ride problem with heterogeneous users in which the vehicle capacity can be modified en-route by reconfiguring its internal layout. The work is motivated by the daily transport of children with disabilities performed by a private company based in Lyon Métropole, France. Every day, a fleet of configurable vehicles is available to transport children to medico-social establishments. The objective of this work is then to help route planners with the fleet dimensioning and take reconfiguration opportunities into consideration in the design of routes. Due to the number of passengers and vehicles, real-size instances are intractable for mix-integer programing solvers and exact solution methods. Thus, a large neighborhood search meta-heuristic combined with a set covering component is proposed. The resulting framework is evaluated on real life instances from the transport company.

Keywords: Transportation logistics, optimization, dial-a-ride problem, large neighborhood search meta heuristic, set covering problem.

1. INTRODUCTION

The standard Dial-a-Ride Problem (DARP) consists in designing vehicle routes in order to serve transportation demands scattered through a geographic area. The global objective is to minimize the transportation cost while satisfying demands.

In contrast to the Pickup and Delivery Problem (PDP), DARP applications concern the transportation of persons. Hence constraints or objectives related to the quality of service should be taken into consideration. In the context of door-to-door transportation of elderly and disabled people, the number of applications has considerably grown recently. Population is aging in developed countries, and many people with disabilities cannot use public transport. As a result, new transport modes, public and private, arise to satisfy their transportation needs. Demand from people with disabilities differ in their need of special equipments such as wheelchair spaces or stretchers thus obliging the use of adapted vehicles.

Parragh (2011) introduced the DARP with heterogeneous users and vehicles, and solved instance with up to 96 user requests. Qu and Bard (2013) extended this problem by considering vehicles with configurable capacity. Different categories of users express special needs such as regular seats or wheelchair spaces. These demands are served by configurable vehicles. The goal is to find the most convenient vehicle configuration for each route.

In this paper we present a generalization of the PDP with configurable vehicle capacity. Contrary to the work by Qu and Bard (2013), we allow vehicles to be reconfigured en-route on. The other difference is that we determine the fleet dimension instead of starting with limited fleet. We call this variant dial-a-ride problem with reconfigurable vehicle capacity (DARP-RC). Note it is not an heterogeneous variant because only one vehicle type is considered.

The use of hybrid methods or matheuristics has become quite popular in recent years for routing problems. Our solution method has been inspired by the framework of Grangier et al. (2017) to solve the vehicle routing problem with cross-docking. We combine a Large Neighborhood Search (LNS) metaheuristic with a Set Covering Problem (SCP).

The contribution of this paper is therefore to introduce and solve the DARP-RC. Moreover, we compare the results of the combined LNS-SCP approach with pure LNS and adaptive large neighborhood search (ALNS).

2. INDUSTRIAL CONTEXT

This work is motivated by the daily transport of people with disabilities at Lyon Métropole. One segment of this service is operated by the GIHP company on regular basis. Every day, a fleet of configurable vehicles transport around 500 children from and to Medico-Social Establishments (MSE) for rehabilitative treatment. GIHP serves around 60 MSEs with around 180 adapted vehicles. One of the particularities of the vehicle fleet is the capacity to reconfigure its internal layout to trade off seats by wheelchair spaces as per convenience.

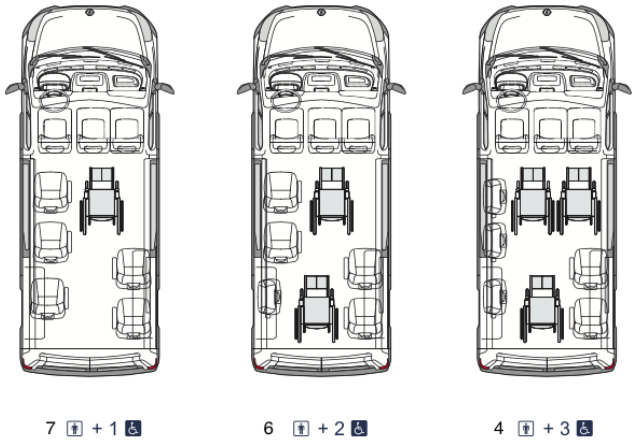

Fig. 1. Example of reconfigurable vehicle (source: www.handynamic.fr)

For MSEs, transportation is often considered the second biggest expense after wages. As a consequence, optimizing the transport becomes a priority. Every year, the company makes strategic choices in the definition and constitution of this fleet. Then, routing decisions are re-revalued daily by route planners.

These decisions are often taken without help of decision making tools like a vehicle routing software. This is why route planners conceive suboptimal assumptions such as designing separate routes for each MSE or ignore vehicle reconfiguration possibilities. As such, this measures can reduce pooling gains and increase operating cost.

3. PROBLEM DEFINITION

In the classic DARP, a homogeneous vehicle fleet is assumed. All vehicles have the same single capacity type and are located at a single depot (Cordeau and Laporte, 2003). The proposed DARP-RC constitutes an extension for DARP problem considering more realistic assumptions such us heterogeneous users (e.g. seats, wheelchairs, stretchers) and vehicles with reconfigurable capacity.

3.1 Reconfigurable vehicles

Vehicles with configurable capacity were introduced in Qu and Bard (2015). In their problem, vehicles were not allowed to change configurations all along the route. This assumption grant configurations to be treated as vehicles types with some extra dimensioning constraints. DARP-RC instead, by allowing reconfigurations, introduces the challenge of tracking each configuration for every visited node. Vehicles can have one or several configurations. Each configuration is characterized by the capacity for each user type. Consider for example vehicle in Fig. 1. In the first configuration it can handle 7 seated people and 1 wheelchair; in the second one, 6 seated people and 2 wheelchairs; in the third one, 4 seated people and 3 wheelchairs. Note that there is no linear relationship between the capacity in the two types of users (one wheelchair cannot be simply converted into one or two seat spaces). Also that unused chairs are folded and not removed from the vehicle.

Example. The following example illustrates how vehicles with reconfigurable capacity can reduce operating cost. Consider a vehicle with two configurations $\{c_1, c_2\}$ as shown in Fig. 2. The first configuration consists of 2 seats and 1 wheelchair space. The second configuration has 4 seats only. Users a and c go to destination $M1$ while b, d, e and f go to $M2$. In order to satisfy all user demands, the reconfigurable vehicle can follow the route $D \rightarrow a \rightarrow b \rightarrow c \rightarrow M1$ using configuration c_1 and $d \rightarrow e \rightarrow f \rightarrow M2 \rightarrow D$ with configuration c_2. Performing the same route without reconfiguring capacity would imply making an extra detour $d \rightarrow M2 \rightarrow d$ using configuration c_1 only (see dotted line); therefore increasing transportation costs.

3.2 Problem Description

The DARP-RC is defined as a network composed of a set $\mathcal{V}$ of vertices containing the set $\mathcal{O}^+$ of vehicle starting depots, the set $\mathcal{O}^-$ of vehicle arrival depots, the set $\mathcal{P}$ of pickup locations and the set $\mathcal{D}$ of delivery locations. Without loss of generality, we address the case of morning routes where the set $\mathcal{P}$ corresponds to people home or any nearby address, and the set $\mathcal{D}$ corresponds to MSEs.

The set of users is partitioned into several categories $u \in \mathcal{U}$, i.e. seat, wheelchairs. An user request $r \in \mathcal{R}$ is composed of a pickup at location $p_r \in \mathcal{P}$, a delivery at location $d_r \in \mathcal{D}$ and the number q_{ru} of persons of each type $u \in \mathcal{U}$ to be transported. Moreover, each request has a maximum ride time $\bar{T}_r$ that guarantees a given quality of service.

The fleet of vehicles is homogeneous. Each vehicle has the set $\mathcal{C}$ of possible configurations that provides different capacity vectors. Hence, the vehicle capacity is expressed by quantities Q_u^c representing the maximal number of users type $u \in U$ that can be transported at a time by the same vehicle when using configuration $c \in \mathcal{C}$.

The sequence of nodes visited by a vehicle forms a route called ω. It is characterized by a starting at depot $o_k^+ \in \mathcal{O}^+$, an arrival at depot $o_k^- \in \mathcal{O}^-$, and the sequence of visited nodes (pickup and delivery nodes).

Each node $i \in \mathcal{P} \cup \mathcal{D}$ is associated with a time window $[a_i, b_i]$, representing the earliest and latest possible visit

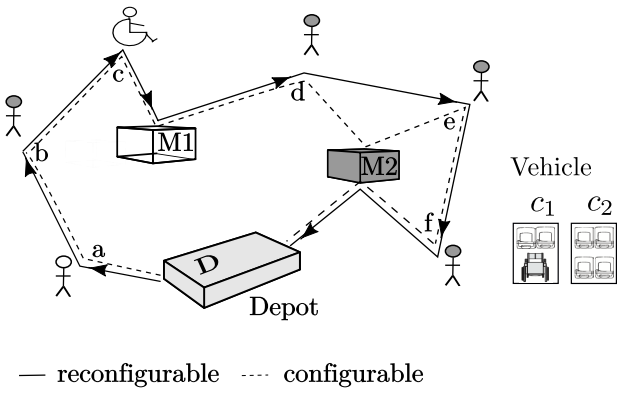

Fig. 2. Comparison of routes with and without capacity reconfiguration

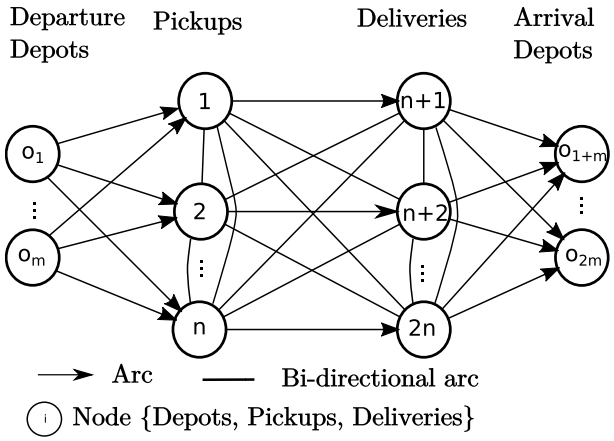

Fig. 3. Directed graph representation of DARP

time. There is also a service time s_i counted as the time required to get in and out of the vehicle.

Without loss of generality, we consider that if $i \in \mathcal{P}$ represents a pickup node, then the node $i+n$ is the delivery node associated with the same request. It is assumed that different nodes can share the same geographical location.

Consider the directed graph $G = (\mathcal{A}, \mathcal{V})$ shown in Fig. 3. $\mathcal{V} = \mathcal{P} \cup \mathcal{D} \cup \mathcal{O}$ corresponds to the set of nodes $\mathcal{A}$ is the set of arcs connecting the nodes. Every arc (i, j) in the graph represents the shortest path in time between nodes i and j. Its travel time is denoted as t_{ij} while its length is denoted as d_{ij}.

Note that reconfiguration time is not considered because it is negligible compared with service times and it can be performed at MSEs while massive drop-offs take place.

The objective function is to minimize the total transportation cost. Three main costs are considered: a fixed cost associated with each vehicle (amortization cost), a time related cost proportional to the total route duration (driver cost) and a distance related cost (vehicle fuel and maintenance costs).

4. RESOLUTION METHOD

An exact method has been proposed in Qu and Bard (2015) to solve the heterogeneous PDP with configurable vehicle capacity for up to 50 requests. For large scale applications usually found in real-life situations, metaheuristics provide a good alternative. We propose a matheuristic that combines the metaheuristic LNS with SCP. It will be denoted as LNS-SCP.

4.1 Matheuristic framework (LNS-SCP)

The matheuristic framework consists of an LNS with a nested SCP solved periodically. LNS was first proposed by Shaw (1998) and introduced under the name *ruin and recreate* in Schrimpf et al. (2000). In LNS the current solution is improved following an iterative process of destroying (i.e. removing parts of the solution) and repairing the current solution. This process is repeated until a stopping criterion is reach. In our case the stopping criterion is a maximal number of iterations or a maximum computational time.

The potential of the approach was revealed by Ropke and Pisiuger Ropke and Pisinger (2006) who proposed an ALNS consisting in multiple search operators adaptively selected according to their past performance in the algorithm.

Algorithm 1 shows the general structure of the LNS-SCP.

Algorithm 1. (The LNS-SCP framework).

Input: Σ^-: set of destroy operators,
Σ^+: set of repair operators,
η: nb. of iterations between two calls to the SCP.
Output: best solution found s^*.
–Begin–
Pool of routes: $\Omega \leftarrow \emptyset$
Request bank: $\mathcal{B} \leftarrow \emptyset$
s, s': current and temporary solutions
$it \leftarrow 0$
While (termination criterion not met) {
  $s' \leftarrow s$
  Destroy quantity: randomly select a value Φ
  Operator selection: select $\sigma^- \in \Sigma^-$ and $\sigma^+ \in \Sigma^+$
  Destroy: apply σ^- to remove Φ requests from s'
  Copy the Φ requests into $\mathcal{B}$
  Repair: apply σ^+ to insert requests from $\mathcal{B}$ into s'
  $\Omega \leftarrow \Omega \cup routes(s')$ */* add routes into the pool */*
  If (acceptance criterion is met)
    $s \leftarrow s'$
  If (cost of s' is better than cost of $s^\star$)
    $s^* \leftarrow s'$
  If (it modulo $\eta = 0$) { */*set covering component*/*
    $s' \leftarrow$ solve set covering problem(Ω)
    Update $s^* \leftarrow s'$ if s' is cheaper than s*
    Update $s \leftarrow s'$ if s' is cheaper than s
    Perform pool management
  }
$it \leftarrow it + 1$
}
Return s^*
–End–

LNS manages 3 solutions at a time: the *current solution* s, the *temporary solution* s' generated after destroying and repairing a copy of s, and the *best solution* found so far s^*. LNS consists in 3 fundamental steps:

(1) Determine the number of requests to remove Φ. We first randomly select the percentage of requests to be removed in the interval $[\alpha, \beta]$.

(2) Destroy and repair the *current solution* with a randomly selected operator $\sigma^- \in \Sigma^-$ and $\sigma^+ \in \Sigma^+$ respectively. This step results in a new *temporary solution.*

(3) Accept or reject the new *temporary solution* s', according to the record-to-record criterion of Dueck (1993): if $objective(s') \leq (1+\chi) * objective(s^*)$, where χ is a small positive value, s' is accepted as the new current solution.

The SCP component is performed every η iterations. The purpose of the SCP component is to correct the LNS bias of discarding good routes that are part of costly solutions. Every new route is a candidate to be stored into a pool Ω of routes. Implementation details are presented in Section 4.3.

4.2 LNS operators

A key aspect of the LNS are the set of destroy and repair operators. With the goal of keeping the framework as simple as possible, we try to reduce the number of used operators without sacrificing the solution quality. At the end, only 2 destroy and 2 repair operators were kept in the framework based on the performance obtained in benchmark instances in Section 5.

Destroy operators determine the set of requests to be removed from the solution according to a certain criterion. In the framework of Pisinger and Ropke (2007) 7 destroy operators are proposed. After testing our framework on literature instances we find out that *random removal* and *historical node-pair removal* were sufficient to obtain competitive results. For details about the implementation of these operators, please refer to Pisinger and Ropke (2007).

Repair operators rebuild a partially destroyed solution in order to restore a complete feasible solution. This operation consists in reinserting nodes one by one from the request bank into the solution according to a specific criterion. Every insertion must satisfy all problem constraints. In our case time windows, ride times and capacity constraints have to be respected. If the operator does not fully repair the solution due to feasibility requirements, the objective function is strongly penalized by multiplying it by a big constant value. The two most common repair operators for the DARP are the *cheapest insertion* and the *k-regret heuristics*, both employed in the LNS-SCP. We implemented the *k-regret heuristics* with values of k varying from 2 to 4.

4.3 Set Covering Problem (SCP)

In LNS, a solution is rejected solely based on its cost with respect to the best solution so far. A rejected solution may contain some good routes, which are also removed. This issue is addressed by storing the routes found by LNS and using them in a SCP to find new solutions. In the following lines, we present the mathematical model and key components for its implementation.

Let Ω be the set of routes in the pool collected through the LNS iterations, and $V_\omega \in \mathbb{R}^+$ the cost of route $\omega \in \Omega$. To describe the itinerary followed by each route, we define the values $R_{r\omega}$, which are set at 1 if request $r \in \mathcal{R}$ is served by route $\omega \in \Omega$, and 0 otherwise.

The set covering problem aims at determining the value of binary variables y_ω, where $y_\omega = 1$ if route $\omega \in \Omega$ is part of the solution and 0 otherwise. The set covering problem is defined by the following model.

$$\min \sum_{\omega \in \Omega} V_\omega y_\omega, \tag{1}$$

s.t.

$$\sum_{\omega \in \Omega} R_{r\omega} y_\omega \geq 1 \quad \forall r \in \mathcal{R}, \tag{2}$$

$$y_w \in \{0,1\} \quad \forall \omega \in \Omega. \tag{3}$$

In every iteration, after calling the repair operator, the current routes are memorized in the pool Ω. In order to reduce the number of variables for the SCP, only non-dominated routes are saved according to Proposition 1.

Proposition 1. (Route dominance). Route ω_1 dominates route ω_2, if ω_1 visits the same set of nodes as ω_2 at a lower cost.

The SCP is solved with a MILP solver every η iterations. The solver is initialized with the best known solution and solved given a time limit T_{limit}. This implies the SCP is not always solved to optimality. If the obtained solution is better that the *current solution* s' then the *current solution* is updated. Otherwise s remains unchanged.

Similarly, the *best solution* s^* is updated if a better solution is found. As constraint (2) of the set covering model allows request duplicates in the output solution, all duplicates are removed to obtain a consistent solution. The cheapest one is always conserved.

If the solver fails to find an optimal solution within the time limit T_{limit}, the pool is cleared and filled again with the routes of the best known solution. This step is refereed as pool management in Alg.1.

5. RESULTS

In order to determine the added value of the set covering component, we implemented three LNS variants: a classic LNS (using operators: k-regret, random removal, historical node-pair removal, worse removal, time related removal and distance related removal), an Adaptive LNS using the same set of operators and the proposed LNS-SCP (using only best insertion, k-regret, random removal and historical node-pair removal). In all cases k=2,3,4 regret was used. The MILP solver for solving the SCP is IBM Ilog Cplex 12.6, running on a single thread. The SCP is solved every $\eta = 1000$ iterations with a time limit of $T_{limit} = 3$ seconds. The acceptance criteria is set to $\chi = 5\%$ and the percentages used in destroy operators are $\alpha = 10\%$ and $\beta = 40\%$ as in Pisinger and Ropke (2007).

In order to test the framework, a set of 9 instances of different sizes is proposed. The first number in the instance name refers to the number of considered requests. The service time for valid people was set to 2 minutes for pickup and 1 minute for delivery. While the service time for wheelchairs was 2 minutes for pickup and 5 minutes for delivery. There is not time window in pickup locations. There is a single depot for vehicles with infinite capacity. Vehicles are all of the type illustrated in Fig 1.

Two experiments were completed. One limiting the number of iterations (Table 1) and the other one limiting the computation time (Table 2). In both cases it is compared LNS, ALNS, and LNS-SCP metaheuristics. 5 runs per instance type were considered. The average gap (Gap_{avg}) was computed with respect to the best known solution (BKS) found throughout both experiments.

In Table 1 can be observed that in most of cases, SCP component improves the objective function on average in 2.96% (= 3.99% − 1.03%) with regard to the LNS and in 1.21% (= 2.24% − 1.03%) with respect to the ALNS version of the algorithm. By limiting the computational

time to 30 minutes it can be observed a similar behavior as shown Table 2, however with a higher gap for the lasts instances. This is expected as the computation time is not big enough as to obtain good results in big instances.

Looking at very large instances, in both benchmarks, ALNS outperforms the other methods which may indicate that ALNS-SCP may be relevant for evaluation. On going experimentation also aim to establish the best value of parameter T_{limit}. In particular a greater value for this parameter may be beneficial for large size instances.

		LNS	ALNS	LNS-SCP
instance	BKS	Gap_{avg}	Gap_{avg}	Gap_{avg}
M75-1	1896.32	1.42%	1.39%	***0.86%***
M70-2	916.67	0.52%	0.37%	***0.14%***
M70-3	1225.17	4.78%	4.68%	***2.01%***
M140-4	2491.77	5.69%	2.70%	***0.46%***
M141-5	2517.88	6.03%	4.41%	***0.48%***
M146-6	2391.74	4.56%	2.01%	***0.84%***
M150-7	2158.22	4.47%	2.43%	***0.49%***
M281-8	4734.87	4.20%	***0.80%***	1.03%
M295-9	5072.44	4.28%	***1.34%***	3.00%
Avg	2600.56	3.99%	2.24%	1.03%

Table 1. Performance comparison of metaheuristics 50000 iterations over 5 runs

		LNS	ALNS	LNS-SCP
instance	BKS	Gap_{avg}	Gap_{avg}	Gap_{avg}
M75-1	1896.32	1.37%	1.25%	***0.28%***
M70-2	916.67	0.35%	0.30%	***0.12%***
M70-3	1225.17	4.65%	4.67%	***1.99%***
M140-4	2491.77	4.46%	2.59%	***0.61%***
M141-5	2517.88	5.64%	4.32%	***0.55%***
M146-6	2391.74	4.15%	2.01%	***0.85%***
M150-7	2158.22	4.25%	2.59%	***0.88%***
M281-8	4734.87	4.16%	***0.85%***	2.51%
M295-9	5072.44	4.08%	***3.02%***	4.60%
Avg	2600.56	3.68%	2.40%	1.38%

Table 2. Performance comparison of metaheuristics in 30 minutes over 5 runs

Table 3 shows detailed information of the best solution found for each instance. It shows computational time (t), the best cost among the five runs (obj), the number of reconfigurations (rec), the number of routes in the solution(routes) and the average number of requests per route (requests/route). The maximum number of reconfigurations was 1 for most of instances. These results is influenced by the vehicle fixed cost and maximum ride time constrains. Nevertheless, a deeper study should be done to determine precisely the key factor for reconfiguration. Finally we can observe the average number of requests per route of 7.31 that is higher than the vehicle capacity.

6. CONCLUSION

Through the paper we have described the dial-a-ride problem using vehicles with en-route reconfigurable capacity and a solution procedure based on LNS and SCP. By analyzing the performance of the LNS-SCP on real life instances, we could observe some significant gains when compared with LNS and ALNS.

We also observed that best LNS-SCP solutions reconfigure en-route at most once for most of the instances. The

instance	t (minutes)	obj	rec	routes	requests /route
M75-1	15,43	1 896,32	1	14	5,00
M70-2	26,49	916,67	1	6	11,67
M70-3	7,37	1 225,17	0	7	10,00
M140-4	30,00	2 491,77	1	16	4,38
M141-5	24,86	2 517,88	1	17	8,29
M146-6	28,90	2 391,74	1	14	10,43
M150-7	48,24	2 158,22	1	13	11,54
M281-8	75,08	4 734,87	1	31	2,26
M291-9	78,85	5 072,44	1	31	2,26
Avg	**37,24**	**2 600,56**	**0,89**	**16,56**	**7,31**

Table 3. Best results

perspectives are therefore to extend the study with heterogeneous vehicles and to make an exhaustive evaluation to characterize the key factors of vehicle reconfiguration.

From the solution method side, we aim to establish the best value for the SCP time limit in particular for large size instances and include the ALNS-SCP for evaluation.

REFERENCES

Cordeau, J.F. and Laporte, G. (2003). A tabu search heuristic for the static multi-vehicle dial-a-ride problem. *Transportation Research Part B: Methodological*, 37(6), 579–594.

Dueck, G. (1993). New optimization heuristics. *Journal of Computational Physics*, 104(1), 86 – 92.

Grangier, P., Gendreau, M., Lehuédé, F., and Rousseau, L.M. (2017). A matheuristic based on large neighborhood search for the vehicle routing problem with cross-docking. *Computers & Operations Research*, 84, 116–126.

Parragh, S.N. (2011). Introducing heterogeneous users and vehicles into models and algorithms for the dial-a-ride problem. *Transportation Research Part C: Emerging Technologies*, 19(5), 912–930.

Pisinger, D. and Ropke, S. (2007). A general heuristic for vehicle routing problems. *Computers & Operations Research*, 34(8), 2403–2435.

Qu, Y. and Bard, J.F. (2013). The heterogeneous pickup and delivery problem with configurable vehicle capacity. *Transportation Research Part C: Emerging Technologies*, 32, 1–20.

Qu, Y. and Bard, J.F. (2015). A Branch-and-Price-and-Cut Algorithm for Heterogeneous Pickup and Delivery Problems with Configurable Vehicle Capacity. *Transportation Science*, 49(2), 254–270.

Ropke, S. and Pisinger, D. (2006). An Adaptive Large Neighborhood Search Heuristic for the Pickup and Delivery Problem with Time Windows. *Transportation Science*, 40(4), 455–472.

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

Shaw, P. (1998). Using constraint programming and local search methods to solve vehicle routing problems. In *International Conference on Principles and Practice of Constraint Programming*, 417–431. Springer.