Available online at www.sciencedirect.com

ScienceDirect

Transportation Research Procedia 27 (2017) 632–639

Transportation Research Procedia

www.elsevier.com/locate/procedia

20th EURO Working Group on Transportation Meeting, EWGT 2017, 4-6 September 2017, Budapest, Hungary

A double dynamic fast algorithm to solve multi-vehicle Dial a Ride Problem

Pasquale Carotenuto [a] *, Fabio Martis [a]

[a] *National Research Council – Istitute for Computer Application "M. Picone"*
Via dei Taurini, 19 – 00185, Rome, Italy, carotenuto@iac.cnr.it

Abstract

In this work a two level heuristic algorithm is described for a nearly real-time multi-vehicle many-to-many Dial-A-Ride Problem (DARP). This algorithm is ready to support a Demand Responsive Transportation System in which we face the problem of quickly evaluate a good-quality schedule for the vehicles and provide fast response to the users. The insertion heuristic is double dynamic nearly real-time and the objective function is to minimize the variance between the requested and scheduled time of pickup and delivery. In the first level, after a customer web-request, the heuristic returns an answer about the possibility to insert the request into the accepted reservations, and therefore in a vehicle schedule, or reject the request. In the second level, during the time elapsed between a request and the following, and after a reshuffling of the order of the incoming accepted requests, the same heuristic works for the whole set of accepted requests, trying to optimize the solution. We intensively tested the algorithm with a requests-generating software that has allowed us to show the competitive advantage of this web-based architecture.

Peer-review under responsibility of the scientific committee of the 20th EURO Working Group on Transportation Meeting.

Keywords: Vehicle routing; Heuristics; Dial-A-Ride; Transportation planning; Public transport

1. Introduction

A Demand Responsive Transport System (DRTS) requires the planning of travel paths (routing) and customer pick-up and drop-off times (scheduling) on the basis of received requests (*Cubillos et al*. 2004). The problem of working out optimal service paths and times is called a Dial-a-Ride Problem (DaRP), which derives from the well-

* Corresponding author. Tel.: +3-906-499-37600; fax: +3-906-440-4306.
E-mail address: carotenuto@iac.cnr.it

2352-1465
Peer-review under responsibility of the scientific committee of the 20th EURO Working Group on Transportation Meeting.
10.1016/j.trpro.2017.12.131

known Vehicle Routing Problem (VRP) (*Toth and Vigo*, 2002; *Barrie et al*., 2003; *Prins*, 2004), with the addition of precedence constraints between pick-up and drop-off locations (*Cordeau and Laporte*, 2007).

Their computational complexity makes both DaRP and VRP as NP-hard problems, so attempts to develop optimal solutions have been limited to simple and small-size problems.

In the DaRP, customers formulate transportation requests from a given origin-destination pair (i.e., from a pick-up point to a delivery point). Transportation is carried out by vehicles that provide a shared service. Due to this fact, several customers may be in the same vehicle at the same time.

Clearly, the DaRP has to take into account specific constraints as we are considering people instead of goods. A DRTS may operate according to a static or to a dynamic mode. In the static setting, all the customer requests are known beforehand, and the DRTS produces, by solving a DaRP instance, the tour each vehicle has to make, respecting the pick-up and delivery time windows while minimising the solution cost (*Bergvinsdottir et al*., 2004; *Uchimura et al*., 1999; *Jaw et al*., 1986; *Jorgensen et al*., 2007). In the dynamic mode, the customer requests arrive over time to a control center and processed immediately and independently to the arrival of a next request. Consequently, the solution may change over time (*Jih and Hsu*, 1999; *Carotenuto et al*., 2006; *Beaudry et al*., 2010; *Berbeglia et al*., 2010). For our system in the dynamic mode, we also use the definition: "Booking mode" and "Running mode". The dynamic DRT in "Booking mode" is used when the system accepts requests that refer to rides to be carried out in the next days. In the "Running mode", a request is received when the service is running.

This work improves on previous approaches as it operates dynamically without interrupting the optimization cycle until the end of the service, providing the best solution found each time the system is modified. We address a DRTS capable of managing incoming transport demand solving a DaRP instance using a solution architecture based on a two-stage algorithm double dynamic. The first one is a constructive heuristic algorithm that quickly provides a feasible solution. The second stage operates between a request and the following one trying to maximize the quality of the solution with the reshuffling of the accepted arrival requests order: reinsert the entire set of accepted request with a minimized overall cost function.

The paper is organized as follows. In Section 1, we describe the DaRP by introducing the assumptions to be considered when a Demand Responsive Transportation System is being designed. In Section 2, we introduce the algorithms used in solving the DaRP instances. In particular, in Section 3, we describe the heuristic algorithm to compute an initial feasible DaRP plan. Section 4 describes the Optimization stage, and Section 5 shows some results related to a real case study and finally conclusions are given.

2. Darp definition methodology

In DaRP (*Bodin et al*., 1983), each customer transportation request is dynamically formulated and specifies a single origin and a single destination as well as the number of passengers and a time of pick-up and delivery. Also a time window is defined in order to represents the maximum deviation respect to the agreed requested time of pick-up and delivery and the maximum time each passenger can remain on board the vehicle. The related time windows defined as maximum deviation by the time agreed and the maximum time that each passenger can remain on board the vehicle. The capacity of all vehicles is limited. For brevity, we do not provide the model formulation here, but the complete mathematical structure is presented in *Savelsbergh and Sol*, 1995, *Cordeau and Laporte*, 2007. We would like to point out that the DaRP has to notify customers as soon as possible the acceptance or denial of their requests (*Horn*, 2002, *Coslovich, et al.*, 2006, *Quadrifoglio et al.*, 2007). Moreover, according to the dynamic mode, the DaRP has to be capable of easily inserting new customer requests into one of the already initiated tours without violating any previously accepted customer requests. The following notations are used throughout the paper: let N be the set of N_i ($i = 1,\ldots, n$) customer requests, M be the set of m_k vehicles ($k = 1,\ldots, m$) operating the tours, and $G(V,A)$ be a symmetric graph representing the road network. The depot of the vehicles and the vehicle stops are represented as nodes in V. The cost c_{hj} denotes the distance between nodes h and j belonging to set V. The cost matrix $\{c_{hj}\}$ satisfies the triangle inequality. Regarding the maximum time that each passenger can remain on board the vehicle (Maximum Ride Time), there is an additional parameter to consider. To set an upper limit to the ride time within which the user has to reach its destination, the DRTS system manager can define a constant parameter of time (or a parameter as function of τ_{hj}), so given the pick-up time, the delivery time can be calculated as follows:

$$t_i^j = t_i^h + \tau_{hj} + \vartheta \text{ (or as } t_i^j = t_i^h + \tau_{hj} + \vartheta(\tau_{hj})\text{)} \quad (1)$$

Where τ_{hj} represents the minimum time required to travel from the pick-up point (h) to the destination point (j), both defined on graph G. If customer i specifies the pick-up time (the delivery time), the DRTS controls the delivery time (the pick-up time), that must be as follows:

$$t_i^j \geq t_i^h + \tau_{hj} \text{ (Alternatively } t_i^h \geq t_i^j - \tau_{hj}\text{)}. \quad (2)$$

The goal is to construct a set of routes in order to satisfy the set N of customer transportation requests by using at most m vehicles. Let t_i^h and t_i^j be the preferred pick up time (on node $h \in V$), and delivery time (on node $j \in V$), of customer i, respectively. T_k represents the tour associated to vehicle k, where tour $T_k = \{v_{1k}, \ldots, v_{r_k k}\}$ is a set indicating the ordered list of r_k nodes in V visited by the vehicle k, for all $k = 1,\ldots, m$. Therefore, a solution for the DaRP is a set of routes T_k satisfying the pick-up and delivery time windows. In particular, Let $s_{i,k}^h$ and $s_{i,k}^j$ be the pick-up and delivery times (at nodes $h, j \in V$), respectively, assigned to customer i whose request has been inserted into tour T_k. We aim to identify the DaRP plan (i.e. a solution S) minimising cost z, representing customer delays, defined as follows:

$$z = \sum_{i=1,\ldots,n} | s_{i,k}^h - t_i^h | + | s_{i,k}^j - t_i^j |, \text{ for all } k = 1,\ldots, m, \text{ and } h, j \in V. \quad (3)$$

We emphasize the following additional statement and hypothesis: each customer request has to specify the number of people requiring the transportation service and the vehicle capacity is a constraint of the number of customers that can be on board the vehicle at any given time. The following sections aim to identify the algorithms adopted for the DaRP solution and describe how they interact.

3. General algorithmic architecture

This Section defines the general algorithmic architecture implemented in solving the DaRP. We consider as customer of our DRTS, which operate in a suburb area, a preregistered user of a web based application. The customers are served by the drivers of a vehicles fleet that pick up and deliver them to requested stops. The drivers pick up the customers on the vehicle at bus stops, only if they were previously booked, and correctly scheduled. We also assume that the problem is characterized by hard time window constraints where each customer i specifies the pick-up time or the delivery time, but not both. As a result, the DRTS has to pre-process the customer requests in order to verify the pick-up and delivery time feasibility and to define the time windows, thus the DaRP instance. Then the DRTS produces a feasible requests instance for the DaRP, and in order to solve this requests instance we implement a two-level algorithm. In the first level an 'Insertion' heuristic algorithm is implemented, that quickly produces a feasible solution S for the requests instance. The second algorithm uses the same heuristics of the first level but processing requests after a reshuffling of the arrival order. Therefore, the best solution S constitutes the input for the second level that produces and evaluates new improved solutions. Figure 1 represents the DaRP general schema.

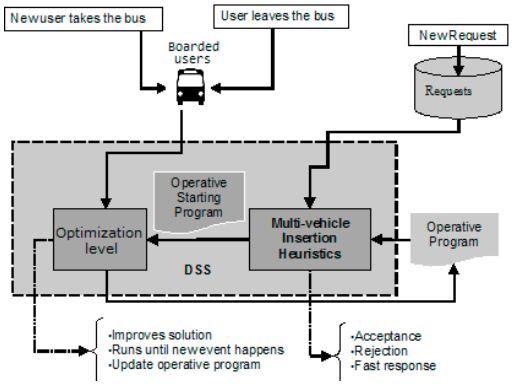

Figure 1 - Algorithm architecture

We emphasize two critical aspects. The first one concerns the 'Insertion' algorithm. In fact, one of the main requirements of a DaRP is its capability to quickly notify customers of the acceptance or denial of their requests. In this setting, the computational time of the algorithm used to solve the DaRP instance could be relevant for a web-based DRTS algorithm (*Fiona Fui-Hoon Nah*, 2004). This is the main reason why we have decided to implement a fast heuristic algorithm. Obviously, the capability of this DRTS will also be exploited in the dynamic mode, where customer requests arrive over time and it is necessary to provide notification as soon as possible.

The second critical aspect deals with the second level algorithm having two main goals. First of all, it has to improve the solutions given by the 'first' level algorithm by allowing it to obtain a more profitable solution from the customer's point of view. Secondly, in the dynamic mode, once the 'Insertion' algorithm accepts a new request, it has to produce a new solution, even if the tours have already been initiated.

4. Insertion algorithm

The algorithm called 'Insertion' is a constructive heuristic that considers one customer request at a time. Initially, the tour T_k associated to each vehicle is empty, for each, $k = 1,\ldots,m$ and at each iteration the algorithm enlarges the tours by selecting one of the customer transportation requests. Therefore, the 'Insertion' algorithm finds a solution in n iterations, one for each customer requests.

To insert a new customer request, selected by using a Fist In First Out structure, the algorithm evaluates the request on the basis of previous solution, and if feasible a new solution S is given. The algorithm ends when all the requests have been evaluated. Clearly, the set of tours identifies an operative plan. In the following, we define the steps of the 'Insertion' algorithm (*Nanry and Barnes*, 2000):

STEP 1: Initially, each tour $T_k = \varnothing$.

STEP 2: For customer request N_i, the algorithm first examines the pick-up node $h \in V$. It analyses one tour T_k (k=1,…,m) at a time and finds all the possible insertions of the pick-up node respecting the pick-up-time window. Then, for all the tours, which the pickup time window has been respected, the algorithm also tries to insert the delivery node $j \in V$, respecting the delivery time window. Clearly, the insertion of customer request N_i can be accepted if the previously inserted customer requests are still feasible, and if the vehicle capacity is not violated.

STEP 3: Among the possible insertions providing a feasible pair (s_{ik}^{h} , s_{ik}^{j}), the algorithm selects the best one, that is the one minimizing the value of z (see(3)). If a feasible insertion does not exist, the customer request is rejected, and the algorithm goes to step 4.

STEP 4: If all the customer requests have been evaluated, then the 'Insertion' algorithm stops. Otherwise, it goes to step 2 by analyzing the next customer request N_{i+1} .

The 'Insertion' algorithm allows a rapid computation of the DaRP plan. Figure 2 shows the scheme of the insertion algorithm.

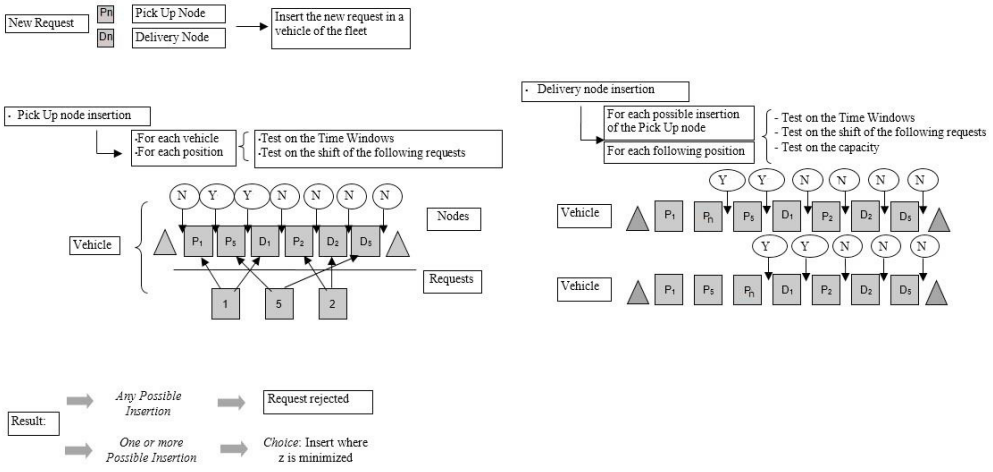

Figure 2 - The insertion heuristic scheme.

5. Test on the network of Tor Vergata University

To verify the answers of the architectural solution proposed in a real case, a series of tests using a real net have been performed. The net on which the tests have been conducted is referred to the zone of Tor Vergata University, a suburban zone of Rome, Italy (Figure 3).

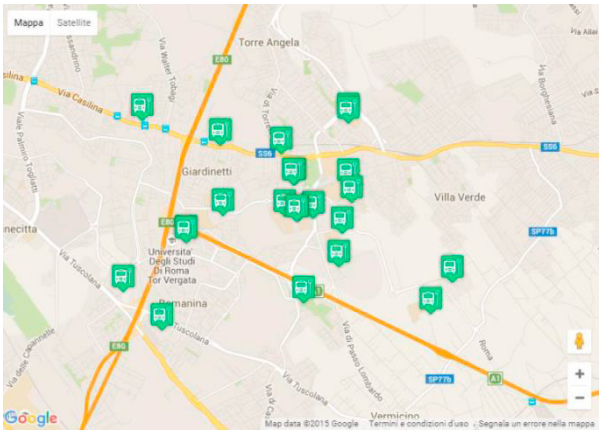

Figure 3 - The suburban zone of Rome "Tor Vergata".

The net used for the test is composed by 25 nodes and the tests have been performed considering:

- percentage of refused requests
- response speed
- "re-optimization" level
- vehicle capacity

5.1 Test on the percentage of refused requests

The first test that has been performed is related to the percentage of refused requests. These have been conducted analyzing 3 different scenarios in which the number of requests reaching the central system increases from 1 to a maximum of 800. Figure 4 shows the difference in the percentage of refused requests for a different number of vehicles of the fleet.

The requests are accepted with a good percentage until a point of break-up is reached. After that point the number of

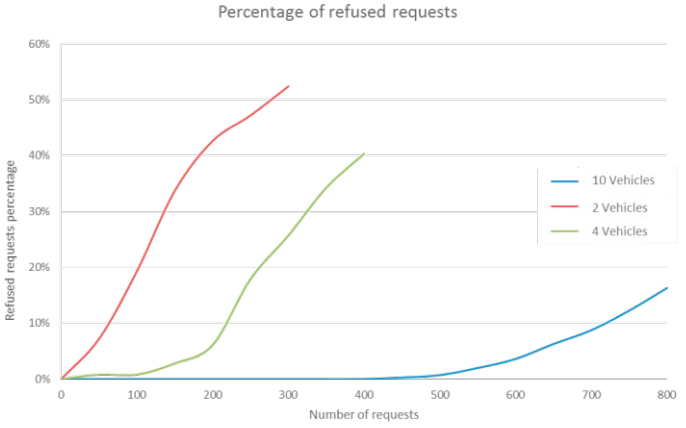

Figure 4 - Percentage of refused requests: network of Tor Vergata University

refused requests increases. This test can be useful in the planning phase of a real service: for a specific forecasted demand the fleet size can be chosen.

5.2 Test on response speed

The following is an example of the computation time achieved by the insertion heuristics in the case of 1000 requests on the network of Tor Vergata University with 10 vehicles. In this case, the *n-th* request is inserted in the n-1 previously accepted requests, with an average time of computation of 1.77 secs. The graph in figure 5b also shows how the insertion time changes by increasing the set of accepted requests in which the next one is inserted.

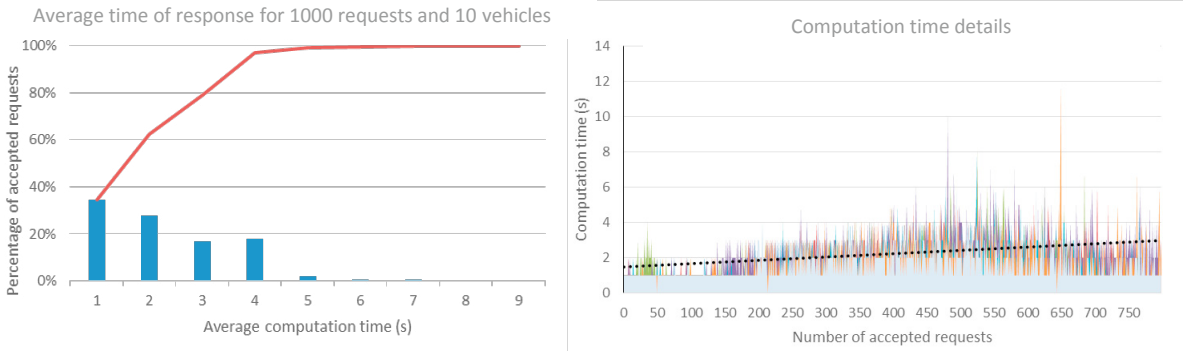

Figure 5 - (a) Computation time for the insertion of a new request (with 1000 requests). (b) Computation time details

This result is encouraging, with 98% of accepted requests within 4 seconds. Our aim is to improve the algorithm in order to apply it to an analogous real case.

5.3 Test on "re-optimization"

This test compares the performances provided by the heuristic in the first level with those provided by the second level. The tests have been performed on multiple scenarios. In Table 1 the achieved results are reported for 10 vehicles:

Table 1 - Comparison of the results between the heuristic and the re-optimization level. Booking and Running mode for 10 vehicles.

AVERAGE OPTIMIZATION PERFORMANCE case: 10 Vehicles "Booking mode"

	NOT OPT	OPT	%
Objective Function (OF)	298,3	321,7	-7,84
Requests(T)	816,1	816,1	
Accepted(A)	628,3	662,8	5,49
Rejected(R)	187,8	153,3	-18,37
R/A	0,2989	0,2313	-22,62
R/T	0,2301	0,1878	-18,37
OF/A	0,4747	0,4853	

AVERAGE OPTIMIZATION PERFORMANCE case: 10 Vehicles "Running mode"

	NOT OPT	OPT	%
Objective Function (OF)	331,1	322,4	2,63
Requests(T)	804,1	804,1	
Accepted(A)	642,5	659,1	2,58
Rejected(R)	161,6	145	-10,27
R/A	0,2515	0,2200	-12,53
R/T	0,2010	0,1803	-10,27
OF/A	0,5153	0,4891	

In this case the optimization level succeeds in improving the results achieved by the heuristic of the first level, but we highlight that for the Running mode we have an objective function improvement and also a greater number of

accepted requests while in the Booking mode we have a reduction of the objective function explained with the increase number of accepted requests.

5.4 Test on vehicle capacity

We tested how our system responds compared with the alternative to increase the vehicle capacity. In table 2 we report results for 10 vehicles with different capacity:

Table 2 - Comparison of the results between different vehicles capacity. Booking mode for 10 vehicles

AVERAGE OPTIMIZATION PERFORMANCE case: 10			
	10 seats	15 seats	%
Objective Function (OF)	207,7	183,5	-11,65
Requests(T)	570,9	570,9	
Accepted(A)	463,7	464,9	0,26
Rejected(R)	107,2	106	-1,12
R/A	0,2312	0,2280	-1,37
R/T	0,1878	0,1857	-1,12
OF/A	0,4479	0,3947	

This result is interesting and leaves good perspectives for future developments, because it highlights that the optimization level is effective. Comparing results between table 1 and table 2 in terms of accepted requests and user utility we can see that the re-optimization phase produces better results than increase the vehicle capacity.

6. Conclusions

In this paper, we proposed a double dynamic algorithm for the DaRP and described its implementation. The DaRP is part of a DRTS which goal is to support a public Service Provider in the management of transport activities. In particular, for the DRTS we approached the problem using a solution architecture based on a two-stage algorithm for a web-based application. We have implemented some computational experiments and obtained some interesting preliminary results showing how effective can be such algorithm in a real application case.

In the test phase, the algorithm, responded with 98% of accepted requests within 4 seconds, encouraging us that it could be easily applied to real case systems. The re-optimization phase produces better schedules that allow us to use smaller fleet of vehicles. Moreover the test proves that the algorithm can evaluate the percentage of rejected incoming requests for a specific forecasted demand and can be useful also in the planning phase of a real service.

The web-based system can be easily adapted to different scenarios changing the dimension of the bus fleet or using different networks.

In the near future, we plan to explore how the second level of the algorithm can be reinforced to obtain better results in both time and objective function.

References

Ambrosino G., J.D. Nelson and M. Romanazzo (2004). *Demand Responsive Transport Services: Towards the Flexible Mobility Agency*. Published by ENEA, Rome.

Barrie M. B. and M.A. Ayechew, (2003). A genetic algorithm for the vehicle routing problem. *Computers & Operations Research*, 30, 787-800.

Beaudry, A., G. Laporte, T. Melo and S. Nickel, (2010). Dynamic transportation of patients to hospitals. *OR Spectrum*, 32, 77-107.

Berbeglia, G., J.-F. Cordeau, G. Laporte, (2010). Dynamic pickup and delivery problems. *European Journal of Operational Research*, 202, 8-15.

Bergvinsdottir K. B., J. Larsen and R. M. Jørgensen (2004). *Solving the Dial-a-Ride Problem using Genetic algorithms*, IMM-Technical report-2004-20, Informatics and Mathematical Modelling, Technical University of Denmark.

Bodin, Lawrence, Bruce Golden, Arjang Assad, and Michael Ball. (1983). Routing and Scheduling of Vehicles and Crews: The State of the Art, *Computers & Operations Research*, 10(2), 63–210.

Carotenuto P., C. Cis and Storchi G. (2006). "Hybrid genetic algorithm to approach the DaRP in a demand responsive passenger service", *"INFORMATION CONTROL PROBLEMS IN MANUFACTURING 2006" Proceedings of the 12th IFAC International Symposium*, Elsevier Science, ISBN: 978-0-08-044654-7, v.3, 349-354.

Coslovich L., R. Pesenti and W. Ukovich (2006). A two-phase insertion technique of unexpected customers for a dynamic dial-a-ride problem, *European Journal of Operational Research*, 175, 1605-1615.

Cordeau J.-F. , G. Laporte, (2007). The Dial-a-Ride Problem: Models and Algorithms, *Annals of Operations Research*, 153, 1, 29-46.

Cubillos C., F. Guidi-Polanco and C. Demartini, (2004). Multi-Agent Infrastructure for Distributed Planning of Demand-Responsive Passenger Transportation Service, *Proceeding IEEE International Conference on Systems, Men and Cybernetics*, 2013-2017.

De Jong K.A., (1975). *An analysis on the behaviour of a class of genetic adaptive systems*. (Doctoral dissertation, University of Michigan). Dissertation Abstract International 36(10), 5140B. (University microfilms n.76-9381).

Fiona Fui-Hoon Nah (2004). *A study on tolerable waiting time: how long are Web users willing to wait?*

Glover F. and G. A. Kochenberger (2003). *Handbook of Metaheuristics*, Kluwer.

Goldberg D.E. (1989). *Genetic Algorithms in Search Optimization, and Machine Learning*. Addison-Wesley, Reading, Massachusetts.

Holland, J. H. (1975). *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor.

Horn M. E.T. (2002). Fleet scheduling and dispatching for demand-responsive passenger services, *Transportation Research part C*, 10, 35-63.

Jaw J.J., A.R. Odoni, H.N. Psaraftis and N.H.M. Wilson (1986). A heuristic algorithm for the multi-vehicle advance request dial-a-ride problem with time windows, *Transportation Research Part B*, 20, 243-257.

Jih W. J. and Y. Hsu (1999). Dynamic Vehicle Routing using Hybrid Genetic Algorithms. *Proceedings of the 1999 IEEE Conference on Robotics and Automation*, 453-458.

Jorgensen RM, J. Larsen and KB. Bergvinsdottir (2007). Solving the dial-a-ride problem using genetic algorithms. *Journal of the Operational Research Society*, 58:1321-31.

Nanry W. and J.W. Barnes (2000). Solving the pickup and delivery problem with time windows using reactive tabu search. *Transportation Research part B*, 34, 107-121.

Prins C. (2004). A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research*, 31, 1985-2002.

Quadrifoglio L., M.M. Dessouki and K. Palmer (2007). An insertion heuristic for scheduling Mobility Allowance Shuttle Transit (MAST) services, *Journal of Scheduling*, 10, 25-40.

Savelsbergh M.W.P. and M. Sol (1995). The General Pickup and Delivery Problem. *Transportation Research*, 29, 17-29.

Taniguchi E., R. G. Thompson, T. Yamada and R. van Duin (2001). *City Logistics: network modelling and intelligent transport system*. Pergamon, Amsterdam.

Toth P. and D. Vigo (2002). The Vehicle Routing Problem. *SIAM Monographs on Discrete Mathematics and Applications*, Philadelphia, Pennsylvania.

Uchimura K., T. Saitoh and H. Takahashi (1999). The dial-a-ride problem in a public transit system. *Electronics and Communication in Japan*, part III 82, n. 7, 30-38.