A heuristic algorithm for the multivehicle advance request dial-a-ride problem with time windows

Jaw, Jang-Jei; Odoni, A. R.; Psaraftis, Harilaos N.; Wilson, N. H. M.

Published in:
Transportation Research. Part B: Methodological

Link to article, DOI:
10.1016/0191-2615(86)90020-2

Publication date:
1986

Document Version
Publisher's PDF, also known as Version of record

Link back to DTU Orbit

Citation (APA):
Jaw, J-J., Odoni, A. R., Psaraftis, H. N., & Wilson, N. H. M. (1986). A heuristic algorithm for the multivehicle advance request dial-a-ride problem with time windows. *Transportation Research. Part B: Methodological*, *20B*(3), 243-257. https://doi.org/10.1016/0191-2615(86)90020-2

Transpn. Res.-B Vol. 20B, No. 3, pp. 243–257, 1986
Printed in Great Britain.
0191-2615/86 $3.00+.00
© 1986 Pergamon Journals Ltd.

A HEURISTIC ALGORITHM FOR THE MULTI-VEHICLE ADVANCE REQUEST DIAL-A-RIDE PROBLEM WITH TIME WINDOWS

JANG-JEI JAW, AMEDEO R. ODONI, HARILAOS N. PSARAFTIS and NIGEL H. M. WILSON
Massachusetts Institute of Technology, Cambridge, MA 02139, U.S.A.
(*Received* 14 *March* 1984; *in revised form* 15 *October* 1984)

Abstract—A heuristic algorithm is described for a time-constrained version of the advance-request, multi-vehicle, many-to-many Dial-A-Ride Problem (DARP). The time constraints consist of upper bounds on: (1) the amount of time by which the pick-up or delivery of a customer can deviate from the desired pick-up or delivery time; (2) the time that a customer can spend riding in a vehicle. The algorithm uses a sequential insertion procedure to assign customers to vehicles and to determine a time schedule of pick-ups and deliveries for each vehicle. A flexible objective function balances the cost of providing service with the customers' preferences for pick-up and delivery times close to those requested, and for short ride times. Computational experience with the algorithm is described, including a run with a real database of 2600 customers and some 20 simultaneously active vehicles. The scenario for the application of the algorithm is also discussed in detail.

1. INTRODUCTION

In this paper we describe a heuristic algorithm, Advanced Dial-A-Ride with Time Windows (ADARTW), for the advance-request version of the multi-vehicle, many-to-many Dial-A-Ride Problem (DARP) with service quality constraints. DARP is concerned with developing a set of routes for a fleet of vehicles carrying customers between specified origins and destinations. "Many-to-many" means that each customer can have a distinct origin and destination, while "advance-request" means that all the requests are received well before the time of vehicle dispatching. Finally, "service quality constraints" guarantee that (1) customers' ride times will not exceed a prespecified maximum and (2) the time of pick-up or delivery of customers will not deviate from the most-desired pick-up or delivery time of these customers by more than pre-specified amounts ("the time windows").

The algorithmic approach to be described is interesting for two reasons. First, it addresses what is probably the most applicable and realistic version of the real-world problem in a way that avoids excessive abstraction and simplification and emphasizes flexibility and convenience to the algorithm's user. Second, it can generate at low computation cost what are apparently high-quality solutions to much larger problems (e.g. 2500 customers and 30 simultaneously active vehicles) than have hitherto been attempted.

Many North American systems serving elderly and handicapped passengers require reservations a day in advance, while in other cases return trips may be scheduled as immediate requests. This second scenario can also be handled with an extension of the algorithm presented in this paper.

The DARP was first examined by Wilson *et al.* (1971, 1976, 1977) in connection with the development of real-time algorithms for the Dial-A-Ride systems of Haddonfield, NJ, and Rochester, NY. ADARTW derives some of its fundamental concepts (building tours through sequential insertion of customers, general form of the objective function) from that work. A paper by Roy *et al.* (1983) which appeared as this paper was being completed also uses similar concepts and solves essentially the same version of DARP. Hung *et al.* (1982) have been tackling the same problem but have adopted an entirely different approach involving processing (scheduling and routing) one vehicle at a time rather than all vehicles simultaneously, as is done by ADARTW. Psaraftis (1983a) has developed a dynamic programming algorithm for solving the *single-vehicle* DARP optimally for a small number of customers in the presence of time windows, and Solomon (1983) has recently published an excellent review of the vehicle routing problem (not DARP) with time-window constraints.

Descriptions of several versions of DARP without time-window constraints can be found in Bodin *et al.* (1983), and many authors have published related papers during the last decade. Hendrickson (1978) and Daganzo (1978) have developed approximate models to evaluate the performance of Dial-A-Ride systems; Stein (1978a, 1978b) has presented a probabilistic analysis of the problem; Sexton (1979) and Bodin and Sexton (1982) have developed approximate algorithms based on Benders decomposition applied to a subscriber Dial-A-Ride system in Baltimore, MD; Psaraftis has described an exact approach based on dynamic programming (1980) as well as several polynomial-time heuristics (1983b, 1983c) for solving the single vehicle problem; and, finally, the authors of this paper have previously developed (Jaw *et al.*, 1982) an approach to solving the multi-vehicle, advance-request DARP in the absence of "hard" time-window constraints, using a three-step (grouping, clustering and routing) algorithm.

The purpose of this paper is to present an overview of the structure of ADARTW and of the heuristic techniques that it uses. These are to a large extent dictated by the operating scenario (described in Section 2) within which the algorithm has been conceived. Section 3 then presents an overview of ADARTW; Section 4 describes the search for feasible insertions of customers into vehicle work-schedules; and Section 5 describes the optimization procedure used to assign customers to vehicles and to fix pick-up and delivery times. Due to space limitations, we omit throughout the paper many details not only on the procedures used but also on such topics as data structures, list processing and updating. These details are nonetheless very important in ensuring the good performance and computational efficiency of the algorithm, and can be found in Jaw (1984). Section 6 summarizes computational experience to date with ADARTW, while Section 7 contains some brief concluding remarks. Table 1 defines mathematical notation which will be used throughout the paper.

2. OPERATING SCENARIO

The Dial-A-Ride system for which ADARTW is designed assumes that each of the system's customers specifies *either* a desired pick-up time (*DPT*) *or* a desired delivery time (*DDT*). Most individuals are constrained in the morning by a desired "delivery" time (e.g. work start time) and select their trip starting time accordingly. Such a Dial-A-Ride customer will be a "*DDT*-specified" customer and will rely on the system to tell him at what time he will be picked up so that he will be delivered by time *DDT*. The reverse is, of course, true for *DPT*-specified customers.

It is further assumed that the system operates under three types of service quality constraints:

1. No *DPT*- (*DDT*-) specified customer will be picked up (delivered) earlier (later) than his *DPT* (*DDT*).
2. No customer's actual ride time will exceed a given maximum.
3. The difference ("time deviation") between the actual pick-up (delivery) time and the desired pick-up (delivery) time of a customer will not exceed a given maximum for *DPT*-specified (*DDT*-specified) customers.

Table 1. Mathematical notation

N:	the number of customers requesting service
n:	the number of available vehicles
$DPT_i(DDT_i)$:	the desired pick-up (delivery) time of customer i
$EPT_i(EDT_i)$:	the earliest pick-up (delivery) time for customer i
$LPT_i(LDT_i)$:	the latest pick-up (delivery) time for customer i
$APT_i(ADT_i)$:	the *actual* (scheduled) pick-up (delivery) time for customer i
$D(x, y)$:	vehicle travel time from point x to point y using the shortest route between x and y
$+i(-i)$:	the event "pick-up (deliver) customer i"; "$+i$"("$-i$") also denotes the *point of origin* (*destination*) of customer i
DRT_i:	the direct ride time of customer i, i.e. $DRT_i = D(+i, -i)$
MRT_i:	the maximum acceptable ride time for customer i
DV_i:	the actual deviation for customer i from his desired pick-up (delivery) time [for DPT-specified customers $DV_i = APT_i - DPT_i$; for DDT-specified customers $DV_i = DDT_i - ADT_i$; see eqns (2) and (3)].
WS_i:	the maximum acceptable deviation of customer i from his desired pick-up or delivery time ($DV_i \leq WS_i$)
d:	the number of stops (pick-ups and deliveries) in a schedule block

The values of the maximum ride time and of the maximum time deviation can either be determined unilaterally by the system's operator and applied universally or, alternatively, can be left open to negotiation between the operator and each customer. In the former case, an operator might advertise, for example, that a customer's ride time would "under no circumstances" exceed twice his direct ride time and that he would be delivered (picked up) no earlier (later) than 20 minutes prior to (after) his desired delivery (pick-up) time.

The problem

The version of DARP solved by ADARTW can now be summarized as follows: Given a subscription list of N customers, each specifying either a DPT_i or DDT_i ($i = 1, 2, \ldots, N$) and a fleet of n vehicles, find an effective allocation of customers among vehicles and an associated time schedule of pick-ups and deliveries such that:

1. For all customers i:

$$ADT_i - APT_i \leq MRT_i \tag{1}$$

2. For DPT-specified customers:

$$DPT_i \leq APT_i \leq DPT_i + WS_i \tag{2}$$

3. For DDT-specified customers:

$$DDT_i - WS_i \leq ADT_i \leq DDT_i. \tag{3}$$

Several comments are in order concerning this formulation:

(a) We have not yet specified a measure of effectiveness (see Section 5).

(b) It may prove *infeasible* to serve some of the N customers with the given vehicle resources and service-quality constraints.

(c) From now on we shall assume for convenience that $WS_i = WS$, a constant, for *all* customers. This is not a necessary condition in ADARTW.

(d) MRT_i, the maximum ride time for customer i, will normally be specified as a function of the direct ride time, DRT_i. In our work we have used:

$$MRT_i = A + B \times DRT_i \tag{4a}$$

where A and B are user-specified constants (e.g., $A = 5$ minutes, $B = 1.5$). Other functional forms can, of course, be used to specify MRT_i, if desired.

The result of application of ADARTW to a list of service requests is a detailed *work-schedule* for each vehicle, listing times and locations for each of the pick-ups and deliveries. Each customer may be called and given an (approximate) APT and ADT, satisfying constraints (1) and (2) or (1) and (3), as appropriate. Those customers, if any, whom it is infeasible to serve will also be notified.

Before proceeding to the description of the algorithm, the following additional assumptions should also be noted:

(i) Vehicle capacity is assumed to be finite and is not necessarily the same for all vehicles.

(ii) Dwell times—the amounts of time needed to pick up and deliver customers—can be nonzero, and different customers may have different dwell times. For discussion of the nonzero dwell times, see Jaw (1984).

(iii) A vehicle is not allowed to idle when it is carrying passengers, since it is felt that such idle waiting would not be tolerated by Dial-A-Ride customers. Such idle periods by nonempty vehicles are accepted by some vehicle-routing algorithms with time-window constraints (see Baker, 1981, and Christofides *et al.*, 1980).

Finally, it is useful to distinguish between *active* periods and *slack* periods for vehicles. At any time, an available vehicle can be either in a slack period (i.e. idling) or active (on the way to picking up a customer during an active period; transporting, picking up, or delivering customers; or returning to a depot).

3. OVERVIEW OF THE ALGORITHM

ADARTW is a heuristic algorithm that processes ride requests sequentially, inserting one customer at a time into the work-schedule of some vehicle until all ride requests have been processed. Central to the algorithm are: a search for *feasible* insertions of customers into work-schedules and an *optimization* step to find the best feasible insertion. An insertion of a customer into the work-schedule of a vehicle is feasible only if it does not lead to violation of any service-quality constraints for the newly assigned customer and for all other customers *already* assigned to that vehicle. The optimization step deals with minimizing the additional "cost" due to inserting the customer into a vehicle's work-schedule. The cost function used is a weighted sum of disutility to the system's customers (due to excess ride times and to deviations from the most desirable pick-up or delivery times) and of system costs, as represented by a function that quantifies the "consumption" of available vehicle resources.

Consider now the case in which there are N customer demands for service and n available Dial-a-Ride vehicles. ADARTW begins by indexing customers in the order of their "earliest pick-up times", EPT_i $(i = 1, \ldots, N)$, i.e. according to the earliest time at which they are expected to be available for a pick-up. Section 4 shows how EPT_i is computed.

The algorithm then processes each customer in the list in sequence, and assigns each customer to a vehicle until the list of customers is exhausted. The processing of a customer i goes as follows:

Step 1. For each vehicle j $(j = 1, 2, \ldots, n)$:

(i) Find all the feasible ways in which customer i can be inserted into the work-schedule of vehicle j (details in Section 4). If it is infeasible to assign customer i to vehicle j, examine the next vehicle $j + 1$ (restart Step 1); otherwise:

(ii) Find the insertion of customer i into the work-schedule of vehicle j that results in minimum additional cost (details in Section 5). Call this additional cost COST_j.

Step 2. If it is infeasible to insert i into the work-schedule of any vehicle, then declare i a "rejected customer"; otherwise, assign i to a vehicle j^* for which $\mathrm{COST}_{j^*} \leq \mathrm{COST}_j$ for all $j = 1, \ldots, n$.

The above is only the "generic" version of the algorithm, and a number of options are available at various points in the procedure:

(a) Customers can be indexed and processed according to criteria other than EPT. For example, one can also process customers "backward" by ordering them according to their latest delivery time, LDT—with the customer having the largest LDT being processed first. Such alternative processing orderings can be used to generate several alternative solutions.

(b) Instead of processing one customer at a time, the user can specify how many (yet unassigned) customers should be considered in Step 1, above. For example, if the number 5 is specified, the top five (in terms of their EPT index) unassigned customers will be considered on each occasion as candidates to be assigned next to a vehicle. It should be emphasized that each of the candidates is considered *separately* in Step 1, so that in Step 2 the candidate (among the five, in our example) with the smallest COST_j would be assigned to vehicle j^*, while the remaining customers stay unassigned. This multi-candidate option is provided for the purpose of improving the performance of the algorithm by making it less "myopic", i.e. by giving it an opportunity to select among more than one customer for the next assignment. The penalty, of course, is that as the number of candidates that are considered each time increases, the efficiency of the algorithm decreases. These points will be discussed further in Section 7.

(c) If the user so desires, ADARTW will not reject any customers. Instead, if it ever proves infeasible to assign a customer i to any of the n initially available vehicles, ADARTW will introduce an additional vehicle to serve that customer. This additional vehicle will join the fleet of vehicles from that time on and will be available to serve subsequent customers.

4. SEARCH FOR FEASIBLE INSERTIONS

We now turn to an outline of the steps taken to identify feasible insertions of customers into vehicle work-schedules. A notion which plays an important role in this respect is that of a "schedule block". This can be illustrated through Fig. 1, which shows part of the work-schedule of vehicle j. A schedule block is a continuous period of active vehicle time between

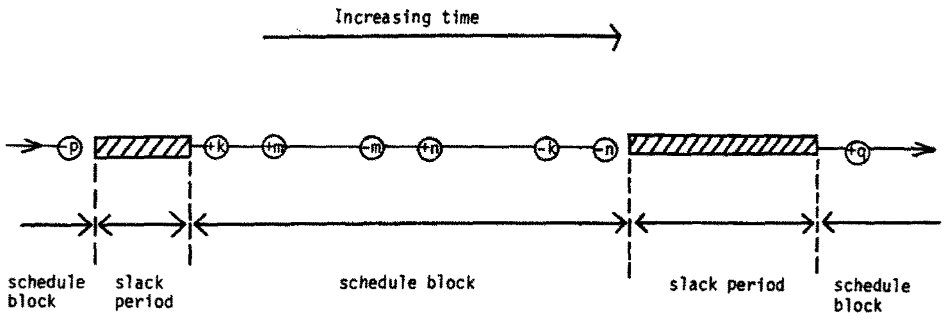


Fig. 1. Part of the work schedule of a vehicle j.

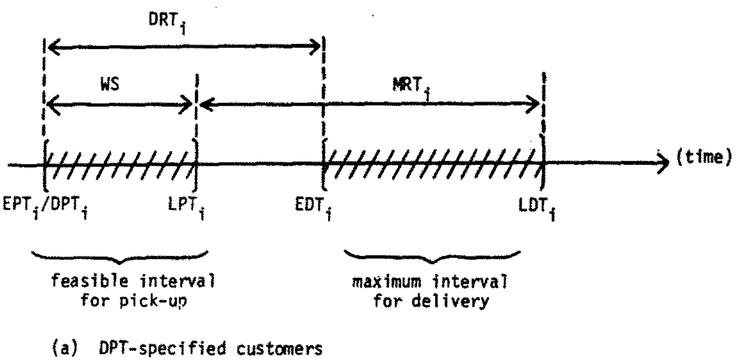
two successive periods of vehicle slack time. A schedule block always begins with a vehicle starting (possibly from a depot) on its way to pick up a customer and ends either after the vehicle discharges its last customer on board or after the vehicle returns to a depot. Associated with a schedule block is a "schedule sequence" indicating the sequence of stops in the block, and a "time schedule" indicating the time when each stop is scheduled to take place. For example, in Fig. 1 the schedule sequence associated with the middle schedule block is $\{APT_k, APT_m, ADT_m, APT_n, ADT_k, ADT_n\}$.

Suppose now that we wish to examine whether an unassigned customer i can be inserted into the work schedule of vehicle j. The objectives of the search for feasible insertions are:

(i) To identify feasible schedule sequences;

(ii) For each feasible schedule sequence to find upper and lower bounds for APT_i, and ADT_i, i.e. bounds for the times when it is feasible to schedule the pickup and delivery of customer i.

ADARTW systematically examines all possible schedule sequences associated with each and every schedule block for vehicle j. For example, with respect to the middle schedule block of Fig. 2, the possible schedule sequences involving the insertion of customer i are $\{+i, -i,$


DRT_i
WS
MRT_i
(time)
EPT_i/DPT_i
LPT_i
EDT_i
LDT_i
feasible interval for pick-up
maximum interval for delivery
(a) DPT-specified customers

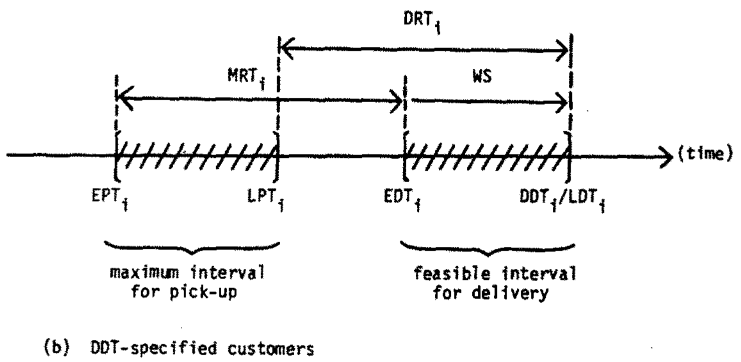

DRT_i
MRT_i
WS
(time)
EPT_i
LPT_i
EDT_i
DDT_i/LDT_i
maximum interval for pick-up
feasible interval for delivery
(b) DDT-specified customers

Fig. 2. Time windows

$+k, +m, -m, +n, -k, -n\}, \{+i, +k, -i, +m, \ldots, -n)\}, \ldots, \{+k, +m, \ldots, -n, +i, -i\}$. In all, if there are d stops already on a schedule block, there are $(d + 2)(d + 1)/2$ possible schedule sequences that involve the insertion of a new customer into that schedule block, while adhering to the constraint that stop "$+i$" must precede stop "$-i$." In view of the maximum ride time and maximum time-deviation constraints of DARP (eqns (1)–(3)), some (and perhaps all) of the above possible sequences may be infeasible.

It should be noted that in addition to inserting customer i into one of the already existing schedule blocks of any vehicle j, ADARTW will consider creating an entirely *new* schedule block for vehicle j in order to accommodate customer i. For example, the first customer ever assigned to a vehicle will obviously always create a new schedule block. Such new schedule blocks are added to the list of existing schedule blocks to which ADARTW attempts to add new customers through the insertion procedure.

A fast screening test

To facilitate the search, ADARTW includes a procedure that greatly increases its efficiency and is fundamental to its viability in dealing with large-scale problems. This procedure involves defining *two* time windows for each customer, as follows:

For *DPT*-specified customers, we define

$$EPT_i = DPT_i \tag{5}$$

$$LPT_i = EPT_i + WS \tag{6}$$

$$EDT_i = EPT_i + DRT_i \tag{7}$$

$$LDT_i = LPT_i + MRT_i \tag{8}$$

These earliest and latest pick-up and delivery times are shown in Fig. 2(a). Note that (5) and (6) contain the same information as (2).

Similarly, for each *DDT*-specified customer, we define (see Fig. 2(b)):

$$LDT_i = DDT_i \tag{9}$$

$$EDT_i = LDT_i - WS \tag{10}$$

$$LPT_i = LDT_i - DRT_i \tag{11}$$

$$EPT_i = EDT_i - MRT_i \tag{12}$$

Equations (9) and (10) contain the same information as (3).

For any customer i, whether *DPT*- or *DDT*-specified, a set of *necessary, but not sufficient* conditions for feasibility is then provided by (13) and (14):

$$EPT_i \leq APT_i \leq LPT_i \tag{13}$$

$$EDT_i \leq ADT_i < LDT_i \tag{14}$$

It can be seen that (13) and (14) are not sufficient because it is possible that APT_i and ADT_i satisfy them, but that the ride time is greater than MRT_i. It should also be noted that LDT_i need not be smaller than EDT_i, i.e. the pick-up and delivery time windows may overlap.

The quantities EPT_i, LPT_i, EDT_i, and LDT_i are computed from eqns (5)–(12), as appropriate, for all customers and define "fixes" on the time axis within which the customer must be picked up and delivered. To understand the value of these fixes, let us return to the problem of checking the feasibility of inserting customer i into a particular schedule block for vehicle j. Let us index the successive stops on the schedule sequence with the subscript $\alpha = 1, 2,$

. . . , d. Note that from (13) and (14) we have an upper and lower bound for each element in the time schedule associated with our schedule block.

For convenience let us now drop the indication of whether a particular stop on the schedule block is a pick-up or a delivery and use ET_α, AT_α, and LT_α to denote earliest, actual (scheduled) and latest time, respectively, for stop α. For instance, in Fig. 1, we would indicate EPT_k, APT_k, LPT_k by ET_1, AT_1 and LT_1, respectively, and EDT_m, ADT_m and LDT_m by ET_3, AT_3 and LT_3, respectively, for the middle schedule block.

In ADARTW, we compute and store four statistics for each stop α on each schedule block, defined as follows:

$$BUP_\alpha = Min\,[\underset{1\le l\le\alpha}{Min}\,(AT_l - ET_l),\, SKT_p] \tag{15}$$

$$BDOWN_\alpha = \underset{1\le l\le\alpha}{Min}\,(LT_l - AT_l) \tag{16}$$

$$AUP_\alpha = \underset{\alpha\le l\le d}{Min}\,(AT_l - ET_l) \tag{17}$$

$$ADOWN_\alpha = Min\,[\underset{\alpha\le l\le d}{Min}\,(LT_l - AT_l),\, SKT_q] \tag{18}$$

where SKT_p and SKT_q denote, respectively, the duration of the slack period immediately preceding and immediately following the schedule block in question.

There is a real intuitive meaning associated with the four quantities defined by (15)–(18). Specifically, BUP_α ($BDOWN_\alpha$) represents the maximum amount of time by which every stop preceding but not including stop $\alpha + 1$ can be advanced (delayed) without violating the time-window constraints. Similarly, AUP_α($ADOWN_\alpha$) represents the maximum amount of time by which every stop following but not including stop $\alpha - 1$ can be advanced (delayed). Essentially, the quantities BUP, $BDOWN$, AUP and $ADOWN$ indicate by how much, at most, each segment of a schedule block can be displaced in order to accommodate (pick up, deliver, or both) an additional customer. As an example, in the situation shown in Fig. 3 (where an attempt is made to insert the pick-up of customer i between the second and third stop of a schedule-block containing four stops), the extra time required to visit $+i$ between $+m$ and $-m$ is given by $DETOUR = D(+m, +i) + D(+i, -m) - D(+m, -m)$. If $DETOUR \le BUP_2 + ADOWN_3$, then it is feasible to insert "$+i$" in the schedule-block at the point indicated without violating any time-windows for the customers already in the schedule-block (k and m). Should this be the case, it is then necessary to check whether it is also feasible to insert "$-i$" at the point indicated in Fig. 3, taking into consideration the fact that an extra amount of time equal to $DETOUR$ has already been spent to pick up customer i. This second check can also be performed using the quantities, BUP, $BDOWN$, AUP and $ADOWN$.

Finally, once it is determined that it is feasible, as far as the time-window constraints are concerned, to insert customer i in a particular way within a schedule-block, a check must be performed to ascertain that no maximum-ride-time constraints are violated for the newly inserted customer and for the customers already in the schedule block. This can be done very easily by scanning through the list of these customers and comparing the respective actual ride-times and maximum ride-times.

In concluding this section, we note that the details of the logic of the feasibility check in ADARTW differ, depending on where in the schedule-block the pick-up and delivery of the newly inserted customer are. ADARTW handles four different possibilities which account for the total of $(d + 2)(d + 1)/2$ combinations mentioned earlier:

(i) Both the pick-up and delivery of customer i are inserted at the end of the last schedule-block, i.e. they become the last two stops on the expanded vehicle work-schedule.

(ii) Both the pick-up and delivery of customer i are inserted between two consecutive stops on the schedule-block.

(iii) The pick-up of customer i takes place somewhere within the schedule-block, while his delivery is inserted at the end of the work-schedule.

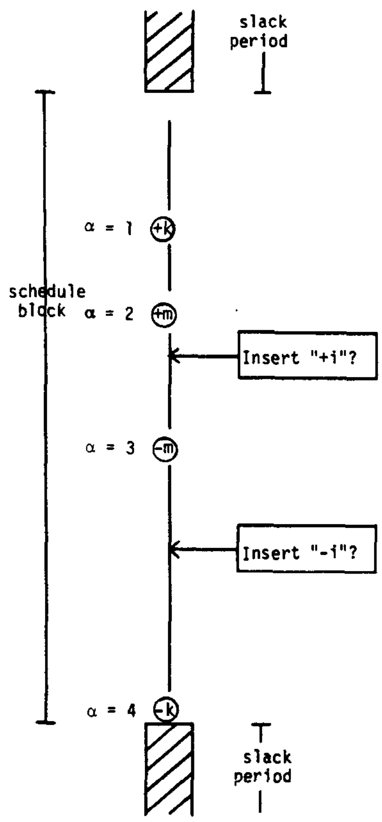


Fig. 3. Insertion of a customer *i* into an existing schedule block.

(iv) The pick-up and delivery of customer *i* are separated by at least one other stop and the delivery of *i* is not the last stop on the expanded work-schedule.

The details of the algorithm's logic for each one of these four possibilities are provided in Jaw (1984). A fifth possibility obviously exists: insert the pick-up and delivery of customer *i* into different schedule blocks. This possibility is not included in the existing version of ADARTW and is discussed further in Section 7.

5. THE OPTIMIZATION PROCEDURE

In order to select among all feasible insertions of customer *i* into the work-schedules of the available vehicles, we use an objective function designed to evaluate the incremental "cost" of each insertion. This cost is taken to be a weighted sum of disutility to the system's customers and of operator costs—the latter measured in terms of "consumption" of available vehicle resources.

Assume that it is feasible to insert customer *i* into the work-schedule of vehicle *j*. Then the incremental disutility of that insertion to the system's customers consists of the sum of two parts: the disutility to customer *i*, i.e. the customer being assigned to a vehicle; and the additional disutility suffered by all *other* customers *already assigned to that vehicle* because of the insertion of customer *i*. The first part (disutility to customer *i*) is given by

$$DU_i = DU_i^d + DU_i^r \tag{19}$$

where

$$\begin{aligned} DU_i^d &= \text{disutility due to deviation from most desired time} \\ &= C_1 x_i + C_2 x_i^2 \quad 0 \le x_i \le WS \end{aligned} \tag{20}$$

and

$$\begin{aligned} DU_i^r &= \text{disutility due to excess ride time} \\ &= C_3 y_i + C_4 y_i^2 \quad 0 \leq y_i \leq MRT_i. \end{aligned} \tag{21}$$

In (20) and (21), C_1, C_2, C_3 and C_4 are user-specified constants and

$$x_i = \begin{cases} APT_i - DPT_i & \text{for } DPT\text{-specified customers} \\ DDT_i - ADT_i & \text{for } DDT\text{-specified customers} \end{cases} \tag{22}$$

and

$$y_i = ART_i - DRT_i. \tag{23}$$

The quadratic terms allow the modeling of situations in which disutilities are believed to increase nonlinearly with x_i and/or y_i. Clearly, by varying C_1, C_2, C_3 and C_4 (including assigning the value of zero to some of them) many different types of behavior can be represented.

The second part (disutility to other customers) is given by:

$$DU_i^o = \sum_{k \text{ on } j} \left(DU_k^{\text{new}} - DU_k^{\text{old}} \right) \tag{24}$$

where DU_k^{new} and DU_k^{old} are, respectively, the disutilities to customer k after and before insertion of customer i into the schedule of vehicle j. The summation is over all customers k who were already assigned to vehicle j prior to the assignment of customer i.

The incremental cost, VC_i, to the system's operator due to inserting customer i into the work-schedule of some vehicle is quantified in somewhat unusual terms by our objective function. We have:

$$VC_i = C_5 z_i + C_6 w_i + U_i (C_7 z_i + C_8 w_i) \tag{25}$$

where C_5, C_6, C_7 and C_8 are externally set constants, z_i is the *additional active vehicle time* required to serve the customer i, w_i is the change in vehicle slack time due to the insertion and U_i is an indicator of system workload defined as:

$$U_i = \frac{(\text{No. of system customers in } [EPT_i - T_a,\ EPT_i + T_b])}{(\text{No. of vehicles available in } [EPT_i - T_a,\ EPT_i + T_b])}. \tag{26}$$

In (26), T_a and T_b are externally specified constants. For example, if $T_a = T_b = 60$ minutes, U_i is equal to the ratio of the number of customers demanding service to the available number of vehicles during the two-hour time period that has the earliest pick-up time of customer i as its midpoint. Obviously U_i will be larger during periods of heavy demand. Since the total "cost" of an insertion is given by $DU_i + VC_i$—i.e. by the sum of (19), (24) and (25)—it is clear that during heavy-demand periods, the objective function places more emphasis on the system operator's cost (relative to service quality to customers) than during low-demand periods. This is as it should be, since during periods of high utilization, vehicle resources are scarcer and it is thus important to use these resources as efficiently as possible.

Clearly the most difficult part of the optimization is to find the times APT_i and ADT_i which minimize the incremental cost of assigning customer i. However, it can be shown that, for any feasible insertion sequence for customer i, ***the problem of finding the "optimal" insertion times APT_i and ADT_i is equivalent to minimizing a single-variable convex function***, the variable being the amount by which the current time schedule of the schedule-block in question should be shifted. This is proven in Jaw (1984), where an efficient procedure for finding the minimum of this function and the corresponding shift time is also presented.

6. COMPUTATIONAL EXPERIENCE

We have made a large number of runs to gain insight into the performance and computational efficiency of ADARTW. All runs were performed on a VAX 11/750. The runs can be divided into two major categories:

(a) runs using simulated data

(b) runs using real data.

In this section we present some examples of these two categories of tests.

(a) *Runs using simulated data.* The basic scenario examined was the following:

Service area: A square of 36 square miles
Distance metric: Euclidean
Location of customer origins/destinations: Uniformly and independently distributed in the area
Depot location: Center of area
Initial fleet size: 4 vehicles
Customer service option: Add more vehicles if necessary
Vehicle speed: 15 mph
Total number of customers: 250
Time simulated: 9 hours.
Demand pattern (number of customers per hour): 20, 20, 30, 40, 40, 30, 30, 20, 20
Request time distribution within each hour: Uniform and independent
Percent of customers specifying a desired pickup time: 50%
Time-window size: 20 minutes or 10 minutes
Maximum ride time: 5 minutes + 2× (direct ride time)
Parameters, T_a and T_b, used in evaluating average vehicle workload (eqn (26) of Section 5): 60 minutes

In all runs, the locations of customer origin/destinations as well as their desired pick-up/delivery times were the same. The values of the objective function coefficients were set to $C_1 = C_2 = C_3 = C_4 = C_5 = C_6 = 0$, $C_7 = 1$, $C_8 = 1.8$ in the "base-case" run and were

Table 2. Coefficient tests with 20-minute time window

	PARAMETERS	RESULTS					
Run No.	(Only non-zero coefficients are shown)	Vehicles Used	Vehicle Prod.[1]	Average Dev.[2]	Ride time Ratio	Vehicle Slack Time[3]	Vehicle Capacity Required[4]
1	$C_7 = 1$, $C_8 = 0.8$	11	3.84	9.34	1.53	230	4
2	$C_1 = 1$, $C_7 = 1$, $C_8 = 0.8$	11	3.81	7.59	1.50	222	5
3	$C_1 = 3$, $C_7 = 1$, $C_8 = 0.8$	12	3.35	5.57	1.53	368	4
4	$C_1 = 5$, $C_7 = 1$, $C_8 = 0.8$	12	3.33	4.90	1.53	254	5
5	$C_2 = 0.1$, $C_7 = 1$, $C_8 = 0.8$	12	3.71	7.47	1.48	170	4
6	$C_2 = 0.3$, $C_7 = 1$, $C_8 = 0.8$	11	3.34	5.71	1.48	335	5
7	$C_2 = 0.5$, $C_7 = 1$, $C_8 = 0.8$	13	3.22	4.40	1.44	260	4
8	$C_3 = 1$, $C_7 = 1$, $C_8 = 0.8$	10	3.87	9.01	1.45	207	6
9	$C_3 = 1$, $C_7 = 1$, $C_8 = 0.8$	10	3.65	8.50	1.31	292	4
10	$C_3 = 5$, $C_7 = 1$, $C_8 = 0.8$	11	3.56	8.62	1.26	408	4
11	$C_4 = 0.1$, $C_7 = 1$, $C_8 = 0.8$	11	3.76	9.37	1.41	252	4
12	$C_4 = 0.2$, $C_7 = 1$, $C_8 = 0.8$	11	3.66	9.62	1.24	158	5
13	$C_4 = 0.3$, $C_7 = 1$, $C_8 = 0.8$	11	3.54	9.73	1.22	253	4
14	$C_7 = 1$, $C_8 = 0.4$	12	3.69	9.06	1.49	323	4
15	$C_7 = 1$, $C_8 = 0.6$	10	3.79	8.76	1.52	273	5

(1 = in passengers/hour; 2 = in minutes; 3 = in total minutes for the simulated period; 4 = in number of seats per vehicle).

varied one at a time in the remaining runs. Tables 2 and 3 display results for the 20-minute and 10-minute time-window cases respectively. Statistics displayed in those tables are defined as follows:

$$\text{Vehicle Productivity} = N/\text{Total Vehicle Time}$$

$$\text{Average Deviation} = (1/N) \sum_{i=1}^{N} DV_i$$

$$\text{Ride Time Ratio} = (1/N) \sum_{i=1}^{N} (ADT_i - APT_i)/DRT_i,$$

where N is the total number of passengers and "vehicle time" for each vehicle is measured from the time the vehicle is sent to pick up its first customer of the day to the time that vehicle delivers its last customer of the day. The purpose of recording ride time ratio (although this term is not directly compatible with our objective function) is to measure ride "circuity" under alternative configurations.

In addition, Fig. 4 and 5 display the utilization of each individual vehicle throughout the simulation period (base-case runs, 20-minute and 10-minute time windows respectively).

Several comments can be made about these results:

1. Tables 2 and 3 show that the behavior of the algorithm when C_1, C_2, C_3 and C_4 and C_8 are varied is predictable. Specifically, *vehicle productivity* generally seems to be a decreasing function of C_1, C_2, C_3 and C_4. Vehicle productivity seems to be most sensitive to variations in C_2 (the coefficient of the quadratic term that represents pick-up or delivery deviation) and least sensitive to variations in C_8 (the coefficient associated with simultaneous changes in system workload and vehicle slack time). In fact, vehicle productivity rises slightly with C_8. Increases in C_3 (the coefficient of the linear term associated with excess ride time) can lead to either an increase or a decrease in productivity.

2. With respect to *average deviation*, there is the expected strong negative correlation between this statistic and both C_1 and C_2, the parameters associated with pick-up or delivery deviation. There is somewhat greater sensitivity in the case of the *wider* (20-minute) time window, where average deviation can be reduced by almost 50% by an appropriate choice of C_1 and C_2.

Table 3. Coefficient tests with 10-minute time window

	PARAMETERS	RESULTS					
Run No.	(Only non-zero coefficients are shown)	Vehicle Used	Vehicle Prod.	Average Dev.	Ridetime Ratio	Vehicle Slack Time	Vehicle Seat Capacity Required
1	C_7 = 1, C_8 = 0.8	13	3.39	4.55	1.43	335	4
2	C_1 = 1, , C_7 = 1, C_8 = 0.8	14	3.23	4.23	1.50	368	4
3	C_1 = 3, C_7 = 1, C_8 = 0.8	12	3.07	3.44	1.45	551	4
4	C_1 = 5. C_7 = 1, C_8 = 0.8	13	2.83	3.01	1.41	689	5
5	C_2 = 0.1, C_7 = 1, C_8 = 0.8	13	3.24	3.98	1.45	355	4
6	C_2 = 0.3, C_7 = 1, C_8 = 0.8	14	3.05	3.20	1.40	540	4
7	C_2 = 0.5, C_7 = 1, C_8 = 0.8	13	3.05	3.20	1.45	461	4
8	C_3 = 1, C_7 = 1, C_8 = 0.8	12	3.14	4.64	1.37	576	4
9	C_3 = 3, C_7 = 1, C_8 = 0.8	13	3.18	4.45	1.28	458	4
10	C_3 = 5, C_7 = 1, C_8 = 0.8	13	3.09	4.58	1.20	491	4
11	C_4 = 0.1, C_7 = 1, C_8 = 0.8	14	3.16	4.48	1.29	489	4
12	C_4 = 0.2, C_7 = 1, C_8 = 0.8	14	3.08	4.85	1.20	402	4
13	C_4 = 0.3, C_7 = 1, C_8 = 0.8	13	3.11	4.79	1.19	413	3
14	C_7 = 1, C_8 = 0.4	13	3.27	4.59	1.46	529	4
15	C_7 = 1, C_8 = 0.6	12	3.27	4.60	1.44	463	4

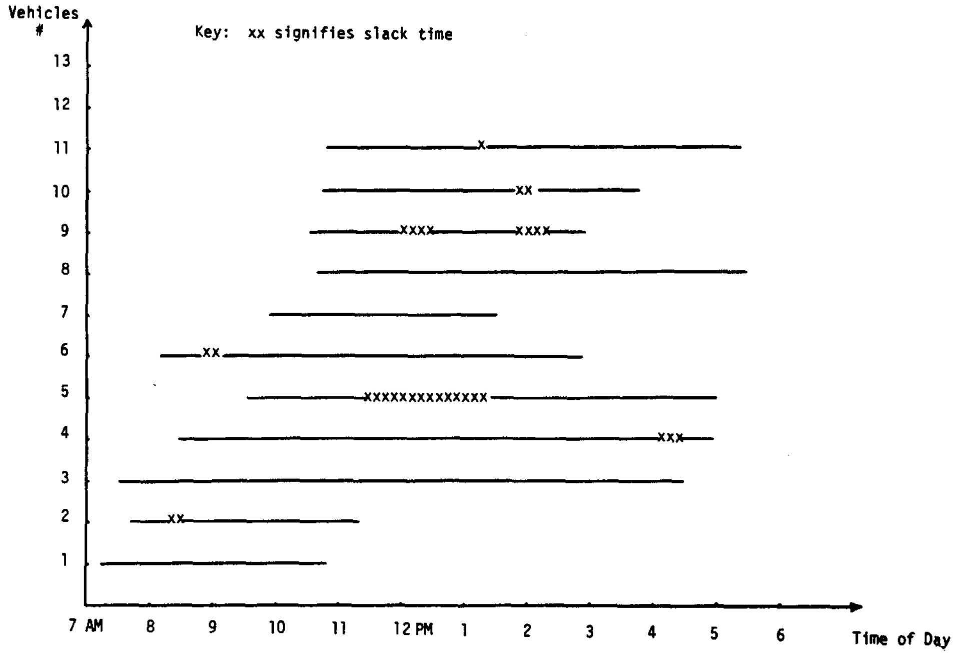


Fig. 4. Base-case vehicle utilization (20-minute time windows)

3. We can see that the *ride time ratio* again, as expected, is most strongly influenced by C_3 and C_4, the coefficients representing excess ride time.

4. The *total number of vehicles* used is never observed to be a strictly monotonic function of any of the above five coefficients, but seems to depend on the width of the time window, although such dependency is less significant than it appears to be at first glance. Comparing for instance Fig. 4 and 5, the two extra vehicles that are required when the time window is halved from 20 to 10 minutes are used for only a very short period.

5. Another interesting effect of reducing the width of the time window (everything else being equal) is the increase in both the *number* and the *total duration* of vehicle slack periods. The increase in number is particularly striking in the base-case runs. Comparing Figs. 4 and

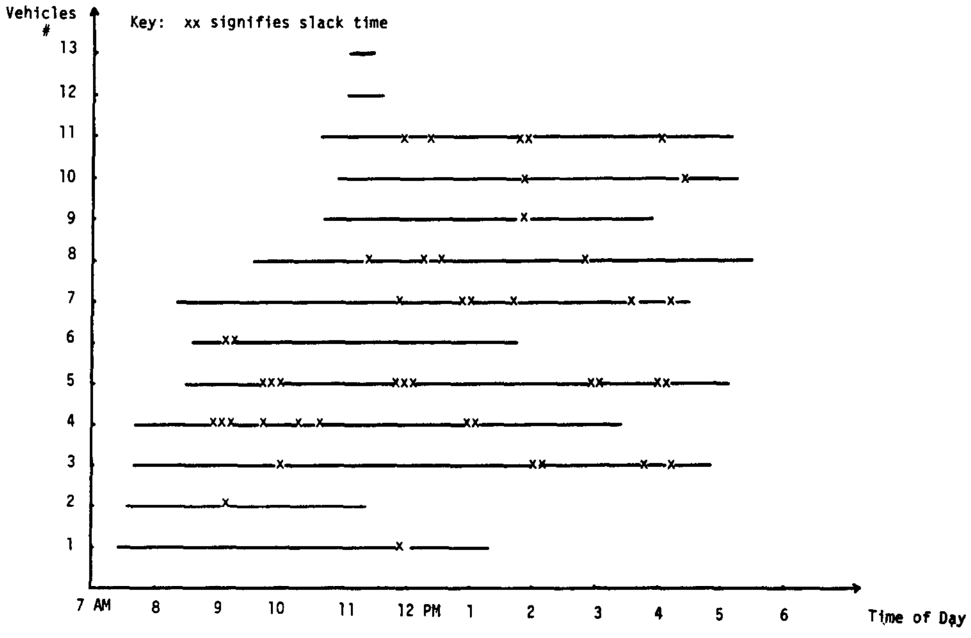


Fig. 5. Base-case vehicle utilization (10-minute time windows)

5, one can see that the number of slack periods almost quadruples, from 8 to 31, when the time window is halved. This observation follows directly from the lower vehicle productivity associated with narrower time windows.

6. Finally, all service attributes were relatively insensitive to changes in the remaining three coefficients C_5, C_6 and C_7. We did not examine simultaneous variations of any of the objective function coefficients.

Each of the 30 runs of Tables 2 and 3 took about 20 seconds of CPU time on the VAX 11/750.

(b) *Runs using real data.* These involved testing the algorithm with data associated with the flexible-route system operated by Rufbus Gmbh Bodenseekreis in the city of Friedrichshafen, West Germany. The particular database with which we tested this algorithm included information covering approximately 16 hours of system operation. Of the 2617 customers of the database, 2397 were *DPT*-specified, with the remaining 220 *DDT*-specified. Some of the *DPT*-specified customers in the database were actually requesting service as soon as possible ("immediate requests") but each customer was converted to an "advance-request" customer by defining the *DPT* to be the time of their service request (such a conversion of course limits out ability to compare results obtained from ADARTW schedules with results from the actual Rufbus operation—see below).

Vehicle stops were located at "checkpoints" across the entire Rufbus system, and the direct trip times between all possible vehicle stop pairs were part of the database. It should be noted here that due to one-way streets and other "peculiarities" of the system, the direct-trip time matrix was not symmetric. The database also included information on the actual vehicle schedules for the period of interest with a 28-vehicle fleet, consisting of one 33-passenger vehicle, four 9-passenger vehicles and twenty-three 17-passenger vehicles.

The schedules in the database had been obtained by Rufbus schedulers by use of an interactive man–machine procedure, details of which were not available. Finally, we were told that Rufbus operators tried to keep a 15-minute time window, although at times this constraint was relaxed to 60 minutes to avoid rejecting customers.

Table 4 highlights statistics from one of our runs, together with statistics of the actual scheduling. Options exercised for this run were the following:

Time window:	15 minutes
Maximum Ride Time:	5 minutes + 1.5 × (Direct Ride Time)
Vehicle Capacity:	17 customers (all vehicles)
Initial fleet size:	10 vehicles
Customer service option:	Add more vehicles if necessary
Non-zero Parameters:	$C_1 = 3$, $C_7 = 1$, $C_8 = 0.8$.

Since the Rufbus procedure was essentially applied to a "mixed" demand scenario (where some of the requests were "immediate" and others were "advance"), while ADARTW was applied to a case where all demands were treated as advance-request, it is essential to stress that no direct comparison of the two procedures can be made from Table 4. However, Table 4 demonstrates that ADARTW performed efficiently in this data set. In addition, ADARTW performed generally better than the Grouping/Clustering/Routing Algorithm that was applied

Table 4. Rufbus scheduling test

	Rufbus Scheduling ("mixed" case)	ADARTW Scheduling ("advance-request"case)
Vehicles used	28	17
Vehicle productivity (Pass./veh. hour)	8.87	12.06
Average deviation (minutes)	11.9	6.6
Ride time ratio	N.A.	1.54

to the same "converted" database (Jaw *et al.*, 1982). In a variety of runs of the latter algorithm with the same database, 21 or 22 vehicles were used; productivity ranged from 8.97 to 10.52 customers per vehicle hour, and average deviation ranged from 6 to 15 minutes. The *CPU* time for this run of ADARTW was about 12 minutes on the *VAX* 11/750.

7. CONCLUSION

ADARTW is an algorithm designed to address much of the complexity of a real problem while offering flexibility and several options to its users. The principal challenge in this instance was obviously the service-quality constraints, particularly the time windows for pick-ups and deliveries. ADARTW seems sufficiently efficient from the computational point of view to deal with multi-vehicle time-constrained problems of size equal to the largest ones encountered in practice.

As in the case of most heuristic algorithms that solve large-scale and complex routing and scheduling problems, it is difficult to state in quantitative terms how good the solutions provided by ADARTW are. There are no "optimal" solutions to compare with, since, first, no exact algorithms to solve problems of similar size exist, and, second, we lack a precise closed-form objective function (the objective of Dial-a-Ride systems can be viewed as that of "satisficing" both operator and customers). About all that can be said is that, as Dial-a-Ride systems go, the solutions found by ADARTW for simulated or real databases are at least as good as or superior to those encountered in practice in all respects (vehicle productivity, vehicle utilization, ride circuity, deviation from desired pick-up/delivery times). In addition, of course, ADARTW provides strict guarantees on the minimum level of service quality to be provided. Through appropriate adjustment of the parameters C_1 through C_8, users of the algorithm can also give more or less emphasis to customer- or operator-oriented objectives, as desired.

None of the variations of the algorithm that we have attempted to date have resulted in significant and consistent improvements to the solution obtained through the basic version of ADARTW described above. For example, quite surprisingly, we found that the consideration of a group of two or more customers as candidates for the next insertion (see Section 3) did not result in large improvements, while at the same time leading to considerable increases in computation cost—especially when more than 5 customers at a time are considered.

Another change that we are currently working on is the consideration of insertions of the pick-up and delivery of customer i into different—not necessarily consecutive—schedule blocks. Of course, this requires merging and manipulating simultaneously all the schedule blocks between (and including) the ones where the pick-up and delivery are inserted. All the slack time contained between these schedule-blocks must be eliminated since, otherwise, the vehicle would idle with customer i on board. This change may conceivably lead to significant improvements in the quality of the solutions.

Finally, another important direction for further research would be to include a "dynamic" updating capability for the algorithm, so that some customers (a small fraction in the most likely operating scenario) could be added to the vehicle schedules on a "real-time" basis. In other words, while most customers would still be scheduled on an advance-request basis, the system would allow some customers to request immediate service, which would be provided to them through reconfiguration of existing vehicle routes and schedules with assistance from such a dynamic version of ADARTW.

Acknowledgements—Work for this paper was supported in part by the Paratransit Integration Program of the Urban Mass Transportation Administration, U.S. Department of Transportation. We are particularly grateful to the project's technical monitor, Ed Neigut, Director of the Paratransit Integration Program, for his general support and for his many specific suggestions on the formulation of the problem. The latter include: (a) the definition of "time windows" for service acceptability (Section 4); and (b) the notion that poorly served (infeasible) customers should be rejected.

REFERENCES

Baker E. (1981) An Algorithm for Vehicle Routing with Time Window Constraints. Management Science Department Report, University of Miami, Coral Gables, Florida.

Bodin L. D., Golden B. L., Assad A. and Ball M. O. (1983) Routing and scheduling of vehicles and crews: The state of the art. *Computers and Operations Research* **10**, 63–211.

Bodin L. D. and Sexton T. R. (1982) The multi-vehicle subscriber dial-a-ride problem. Working Paper No. 82-005, Management Science and Statistics, University of Maryland at College Park, Maryland.

Christofides N., Mingozzi A. and Toth P. (1980) Exact Algorithms for the TSP with Additional Constraints. Euro-IV Congress, Cambridge, England.

Daganzo C. (1978) An approximate analytic model of many-to-many demand responsive transportation systems *Transportation Research* **12**, 325–333.

Hendrickson C. T. (1978) Approximate, Analytic Performance Models of Integrated Transit System Components. Ph.D. Thesis, Dept. of Civil Engineering, M.I.T., Cambridge, MA.

Hung H. K., Chapman R. E., Hall W. G. and Neigut E. (1982) A Heuristic Algorithm for Routing and Scheduling Dial-a-Ride Vehicles. ORSA/TIMS National Meeting, San Diego, California.

Jaw, J. J. (1984) Heuristic Algorithms for Multi-Vehicle, Advance-Request Dial-A-Ride Problems, Ph.D. Thesis, Dept. of Aeronautics and Astronautics, M.I.T., Cambridge, MA.

Jaw J. J., Odoni A. R., Psaraftis H. N. and Wilson N. H. M. (1982) A heuristic algorithm for the Multi-Vehicle Many-to-Many Advance-Request Dial-a-Ride Problem. Working Paper MIT-UMTA-82-3, M.I.T., Cambridge, MA.

Psaraftis H. N. (1980) A dynamic programming solution to the single vehicle many-to-many immediate request dial-a-ride problem *Transportation Science* **14**, 130–154.

Psaraftis H. N. (1983a) An exact algorithm for the single vehicle many-to-many dial-a-ride problem with time windows. *Transportation Science*, **17**, 351–357.

Psaraftis H. N. (1983b) k-interchange procedures for local search in a precedence-constrained routing problem. *European Journal of Operational Research* **13**, 391–402.

Psaraftis H. N. (1983c) Analysis of an $O(N^2)$ heuristic for the single vehicle many-to-many Euclidean dial-a-ride problem. *Transportation Research* **17B**, 133–145.

Roy S., Chapleau L., Ferland J., Lapalme G. and Rousseau J.-M. (1983) The construction of routes and schedules for the transportation of the handicapped. Working Paper, Publication No. 308, Centre de recherche sur les transports, University of Montreal, Canada.

Sexton T. R. (1979) The Single Vehicle Many-to-Many Routing and Scheduling Problem. Ph.D. Thesis, State University of New York at Stony Brook, New York.

Solomon M. M. (1983) Vehicle routing and scheduling with time window constraints: Models and algorithms. Paper 83-02-01, Department of Decision Sciences, Wharton School, University of Pennsylvania, Philadelphia, PA.

Stein D. M. (1978a) An asymptotic, probabilistic analysis of a routing problem: *Mathematics of Operations Research* **3**, 89–101.

Stein D. M. (1978b) Scheduling dial-a-ride transportation systems. *Transportation Science* **12**, 232–249.

Wilson N. H. M., Sussman J. M., Wang H.-K. and Higonnet B. T. (1971) Scheduling algorithms for dial-a-ride systems. Urban Systems Laboratory Report USL TR-70-13, M.I.T., Cambridge, MA.

Wilson M. H. M. and Weissberg H. (1976) Advanced dial-a-ride algorithms research project: Final report. Report R76-20, Dept. of Civil Engineering, M.I.T., Cambridge, MA.

Wilson N. H. M. and Colvin N. H. (1977) Computer control of the Rochester dial-a-ride system. Report R77-31, Dept. of Civil Engineering, M.I.T., Cambridge, MA.