# *Technical Note*

# An Exact Algorithm for the Single Vehicle Many-to-Many Dial-A-Ride Problem with Time Windows

HARILAOS N. PSARAFTIS

*Massachusetts Institute of Technology, Cambridge, Massachusetts*

*This paper modifies the exact Dynamic Programming algorithm developed by the author for the single vehicle many-to-many immediate request Dial-A-Ride problem to solve the problem where each customer has specified upper and lower bounds for his pickup and delivery times and where the objective is to minimize the time needed to service all customers. The major difference between the two algorithms is the substitution of backward recursion with forward recursion. The new algorithm requires the same computational effort as the old one* ($0(N^2 3^N)$ *for* $N$ *customers*) *and is able to recognize infeasible problem instances.*

Although single-vehicle Dial-A-Ride systems do not exist in practice, single-vehicle Dial-A-Ride algorithms can be used as *subroutines* in large scale multivehicle Dial-A-Ride environments. It is mainly for this reason that one's ability to "solve" the single-vehicle Dial-A-Ride problem is considered important. In fact, the single-vehicle version of this problem has received significant attention from many researchers over the past few years (for instance, References [1–4] among others).

In Reference [1], this author developed an exact algorithm, based on Dynamic Programming, for solving the single vehicle many-to-many, immediate-request Dial-A-Ride problem. In that version of the problem, $N$ customers were assumed to request immediate (as-soon-as-possible) service from $N$ distinct origins to $N$ distinct destinations, by a single vehicle whose initial location is known (e.g. a depot). A generalized objective function was examined, consisting of a weighted combination of



**Transportation Science**
Vol. 17, No. 3, August 1983

0041-1655/83/1703-0351 $01.25

the time to service all customers and the total degree of dissatisfaction experienced by customers while waiting for service. This dissatisfaction was assumed to be a linear function of each customer's waiting and riding times. In addition, vehicle capacity constraints and special priority rules were included, and both "static" and "dynamic" versions of the problem were examined.

Some other important variants of this problem exist and have been examined by various researchers. Perhaps the most important of those variants is what is known as the "Dial-A-Ride problem with desired pickup or delivery times," in which each customer specifies a desired time for his pickup and/or delivery. The work of BODIN AND SEXTON[2, 5] exemplifies efforts in this area. In addition, BAKER[6] and CHRISTOFIDES, MINGOZZI AND TOTH[7] have examined Traveling-Salesman-type vehicle routing problems with time windows on delivery times.

This short technical note will show that a procedure similar to the one developed in Reference [1] can be applied without difficulty to the single-vehicle many-to-many Dial-A-Ride problem where each customer has specified time windows for both his pickup *and* his delivery times. Specifically, we shall assume that customer $i$ ($i = 1, \cdots, N$) has requested to be picked up within the interval $[a_i, b_i]$, *and*, delivered within the interval $[c_i, d_i]$. This formulation is more general than the one presented by some other researchers (e.g. References [2, 5]) who typically impose *either* desired pickup time *or* desired delivery time constraints. Of course, it is understood that setting $a_i$ or $c_i$ equal to $-\infty$, and/or $b_i$ or $d_i$ equal to $+\infty$ can make any one (or all) of the above four time constraints superfluous. In this paper, we shall assume that $a_i$, $b_i$, $c_i$ and $d_i$ are known inputs for each customer $i$, inputs which satisfy certain necessary conditions to avoid trivial infeasibility. In general, those conditions will not be also sufficient for feasibility, but the procedure that will be developed will be able to identify infeasible problem instances, should those instances occur.

## 1. FEASIBILITY CONSIDERATIONS

FOR $N$ customers, there will be $2N + 1$ points to be visited by the vehicle. Let $L$ be an index representing the point which is currently visited by the vehicle. By convention, $L = 0$ means the vehicle is at the starting point, $1 \leq L \leq N$ means the vehicle picks up customer $L$ and $N + 1 \leq L \leq 2N$ means the vehicle delivers customer $L - N$. It is assumed that the "direct time" matrix $[t(x, y)]$ with $x$ and $y$ between 0 and $2N$ is known and not necessarily symmetric. The convention for $x$, $y$ is the same with the one introduced for $L$ above.

The problem will be trivially infeasible if any one of the conditions listed below is true:

(a) $a_i > b_i$ for some customer $i$
(b) $c_i > d_i$ for some customer $i$
(c) $d_i - a_i < t(i, i + N)$ for some customer $i$.

For instance, the third condition states that if the difference between the latest delivery time ($d_i$) and the earliest pickup time ($a_i$) of any customer ($i$) is less than the direct time from the origin ($i$) to the destination ($i + N$) of that customer, then the entire problem is infeasible.

Therefore, to avoid trivially infeasible cases such as the above, and without loss of generality, we shall assume that the inputs $a_i$, $b_i$, $c_i$, $d_i$ and $[t(x, y)]$ satisfy the following conditions:

$$-\infty \leq a_i \leq b_i \leq +\infty \qquad i = 1, \cdots N \qquad (1)$$

$$-\infty \leq c_i \leq d_i \leq +\infty \qquad i = 1, \cdots N \qquad (2)$$

$$d_i - a_i \geq t(i, i + N) \qquad i = 1, \cdots N. \qquad (3)$$

Of course, (1), (2) and (3) are only *necessary* conditions for feasibility. Whether or not the problem inputs guarantee a feasible vehicle route and schedule cannot be predicted with certainty before the algorithm for solving the problem is executed. Details on that point appear later.

Three additional assumptions concerning feasibility are the following:

1. If the vehicle arrives at any point (either origin or destination) *later* than the *upper* bound or that point's time constraint ($b_i$ or $d_i$), then *the entire* vehicle route and schedule is *infeasible*. In other words, those upper bounds constitute *hard* constraints that should be met by the vehicle. Of course, in a real-world setting this assumption could be relaxed but the scope of this note is theoretical rather than implementation-oriented, hence the assumption.

2. If the vehicle arrives at any point (either origin or destination) *earlier* than the *lower* bound on that point's time constraint ($a_i$ or $c_i$), then the vehicle *will stay idle* at that point and depart immediately at $a_i$ or $c_i$. In case the point is a pickup point, the vehicle will simply wait for customer $i$ and then depart at time $a_i$, hence the assumption is reasonable for pickup points. The *main* reason this assumption is extended to delivery points as well is for uniformity. Indeed, if the vehicle arrives at a destination earlier than time $c_i$, there would be usually little reason to force customer $i$ to remain inside the vehicle until $c_i$ (except perhaps to punish him by proving that the service agency intends to enforce to the letter the constraints *he* himself set!).

Of course, there may very well be valid reasons why a customer may not want to arrive at his destination earlier than $c_i$, hence we regard both lower bound time constraints as hard constraints.

3. Finally, no other constraints (capacity or priority) will be considered. Again, this causes no loss of generality, for the method of including those constraints is known (see Reference [1]).

## 2. THE ALGORITHM

To SOLVE the time-window version via Dynamic Programming, one has to abandon the *backward* recursion scheme of Reference [1], because backward recursion cannot keep track of time *from* a specified instant (e.g. $t = 0$) *onward*. It is straightforward to overcome this difficulty by developing a *forward* recursion scheme. Thus, define $V(L, k_1, \cdots k_N)$ as the minimum achievable time from the time the vehicle departs from the depot ($t = 0$) until state $(L, k_1, \cdots, k_N)$ is reached, with the understanding that if $V(L, k_1, \cdots k_N) = +\infty$, then state $(L, k_1, \cdots, k_N)$ is infeasible.

We use the same state representation as in Reference [1], that is:

$L$ = index representing the point the vehicle is currently visiting) ($0 \leq L \leq 2N$)

$k_i$ = status of customer $i$($i = 1, \cdots,$ ): $k_i = 3$ means the customer is awaiting for the vehicle, $k_i = 2$ means the customer is in the vehicle and $k_i = 1$ means the customer has been delivered.

As in Reference [1], we assume that $(L, k_1, \cdots, k_N)$ is *consistent*, that is, if $L = 0$, then $k_1 = \cdots = k_N = 3$, if $1 \leq L \leq N$, then $k_L = 2$ and if $N + 1 \leq L \leq 2N$, then $k_{L-N} = 1$. In other words, the recursion will examine only consistent states.

For any particular consistent state $(L, k_1, \cdots, k_N)$, let $X$ denote the set of points that are potential immediate predecessors of $L$ on the route from the starting point to $L$. It is straightforward to check that $X$ is described by:

$$\begin{aligned} X = \{0 \text{ if } k_1' = \cdots k_N' &= 3\} \\ U\{i{:}1 \leq i \leq N \text{ and } k_i' &= 2\} \\ U\{i{:}N + 1 \leq i \leq 2N \text{ and } k_{i-N}' &= 1\} \end{aligned} \tag{4}$$

where $U$ means "union" and : means "such that" and where $(k_i', \cdots k_N')$ is the "predecessor-$k$" vector, given by

$$k_i' = \begin{cases} k_i + 1 \text{ if } i = L \quad \text{or} \quad i = L - N \\ k_i \qquad \text{otherwise} \qquad (i = 1, \cdots N) \end{cases} \tag{5}$$

For a particular element $x$ of $X$ we also define the following:

$$u(x) = t(x, L) + V(x, k_1', \cdots, k_N') \tag{6}$$

and

$$v(x)=\begin{cases} u(x)\begin{cases}\text{if} & (1 \le L \le N \quad \text{and} \quad a_L \le u(x) \le b_L)\\ \text{or} & (N+1 \le L \le 2N \quad \text{and} \quad c_{L-N} \le u(x) \le d_{L-N})\end{cases} \\ a_L \quad \text{if} \quad (1 \le L \le N \quad \text{and} \quad u(x) < a_L) \\ c_{L-N} \quad \text{if} \quad (N+1 \le L \le 2N \quad \text{and} \quad u(x) < c_{L-N}) \\ +\infty \begin{cases}\text{if} & (1 \le L \le N \quad \text{and} \quad u(x) > b_L)\\ \text{or} & (N+1 \le L \le 2N \quad \text{and} \quad u(x) > d_{L-N}).\end{cases}\end{cases} \tag{7}$$

Then the (forward) optimality recursion is

$$V(L, k_1, \cdots, k_N) = \text{Min}_{x \in X} v(x) \qquad (L \neq 0) \tag{8}$$

with $X$, $k_i'$, $v(x)$ and $u(x)$ defined by (4), (5), (6) and (7).

The boundary condition of the problem is

$$V(0, 3, 3, \cdots 3) = 0 \tag{9}$$

The optimal value of the problem is

$$V_{\min} = \text{Min}_{N+1 \le L \le 2N} V(L, 1, \cdots, 1). \tag{10}$$

If $V_{\min} = +\infty$, the problem is infeasible.

## 3. DISCUSSION

THE SCOPE of this note has been rather limited: We have shown that the approach developed in Reference [1] can be extended, with some straightforward modifications, to tackling the "time-window" version of the single-vehicle DARP, as this was defined earlier. However, there are some additional issues regarding this algorithm that merit discussion.

### 3.1. On the Loss of Generality of the Objective Function

The approach developed in the previous section, although valid if the objective function is to minimize the time needed to service all customers, may not be necessarily generalizable to objective functions dealing with the minimization of total customer dissatisfaction (however that is defined). In fact, the possibility of such an extension, although straightforward in the absence of time windows,[1] is an open question when time windows are present. Looking at this issue from another viewpoint, it is quite conceivable that in many of those problems the number of *feasible* vehicle schedules becomes so limited (if actually that number is not equal to zero), that the difference in the optimal vehicle schedule under alternative objective functions (total dissatisfaction versus time to deliver all customers) may not be that significant. It is speculated that in such

"highly feasible" cases, either objective will produce more or less the same vehicle route and schedule.

### 3.2. On the Computational Complexity of the Algorithm

The computational effort of this algorithm is $0(N^2 3^N)$ and its storage requirement is $0(N3^N)$, the same as in the algorithm of Reference [1]. As noted in Reference [1], this rapid exponential growth sets the practical limit on the size of problems that can be solved to be around $N = 8$ to 10 customers (or 17 to 21 points). In fact, it was observed in Reference [1] that the CPU time associated with the *unconstrained* version of the problem was of the order of 2.7 seconds for $N = 5$, 46.8 seconds for $N = 7$ and 591.4 seconds for $N = 9$ (on a VAX 780/II). Shorter execution times were observed in the presence of capacity or priority constraints, but the general growth pattern was similar. Since the structure of the algorithm of this paper is virtually identical to that of Reference [1] (as far as its number of iterations are concerned), we expect a similar growth pattern and similar order-of-magnitude values for its CPU time. Because of the way the time window constraints are handled within the algorithm, we do not expect a significant variation in CPU time with respect to how wide or how narrow those windows are. This is in contrast to other time-window algorithms (e.g. Reference [6]) where the time to find a feasible (or optimal) solution depends significantly on how narrow the time windows are.

How viable can this algorithm be in a real-world environment? The answer to this question depends crucially on the environment itself. A discussion of this issue in an immediate-request environment was presented in Reference [1] and need not be repeated here. For an advance-request environment (to which the algorithm presented here is more applicable) the need of very fast execution times is less important than before. On the other hand, in an advance-request environment the sizes of problems that are likely to be imposed on the algorithm are likely to be larger than before. Hence the computational burden associated with an exact approach cannot be overly deemphasized, and this has prompted many research groups to use heuristic routing algorithms instead of exact.

### 3.3. On Using This Algorithm as a Subroutine in a Multivehicle Environment

In References [3, 4] this author developed some new fast, polynomial-time heuristic algorithms for the single-vehicle many-to-many Dial-A-Ride problem *without* time constraints. Those algorithms are currently used as subroutines in a multivehicle advance-request algorithm developed by the author and his colleagues in Reference [8]. Time constraints in the multivehicle version do exist, but are treated indirectly and in a "soft" way, that is, there are no guarantees that desired pickup or delivery

times will be satisfied "to the minute." It was realized that trying to provide *hard* constraint guarantees would likely render most of those problems infeasible (unless the time windows were suspiciously chosen), and would not make too much sense anyway for real-world problems. It would be interesting however to attempt to use the algorithm described in this note as a subroutine instead of the heuristic algorithms. The fact that the number of customers handled by each vehicle in each time group turns out to be rather small (for instance, around 5 customers per vehicle per time group for a demand rate of 100 customers per hour, 10 vehicles and a 30-minute time group size) would alleviate the potential computational obstacles associated with this algorithm for the majority of cases. No computational experience with this algorithm along the above lines exists to date.

## ACKNOWLEDGMENTS

THIS WORK has been supported by the Urban Mass Transporation Administration. I would like to thank the editor and two anonymous referees for their comments.

## REFERENCES

1. H. N. PSARAFTIS, "A Dynamic Programming Solution to the Single-Vehicle, Many-to-Many, Immediate Request Dial-A-Ride Problem," *Trans. Sci.* **14,** 130–154 (1980).
2. T. R. SEXTON AND L. R. BODIN, "The Single Vehicle Many-to-Many Routing and Scheduling Problem with Desired Delivery Times," submitted to *Transportation Science.*
3. H. N. PSARAFTIS, "Analysis of an O($N^2$) Heuristic for the Single Vehicle Many-to-Many Euclidean Dial-A-Ride Problem," *Trans. Res.* **178,** 133–145 (1983)
4. H. N. PSARAFTIS, "$k$-Interchange Procedures for Local Search in a Precedence-Constrained Routing Problem," forthcoming in *European Journal of Operational Research.*
5. L. D. BODIN AND T. R. SEXTON, "The Multi-Vehicle Subscriber Dial-A-Ride Problem," Working Paper MS/S 82-005, University of Maryland at College Park, 1982.
6. E. BAKER, "An Algorithm for Vehicle Routing with Time Window Constraints," Working Paper, University of Miami at Coral Gables, 1981.
7. N. CHRISTOFIDES, A. MINGOZZI AND P. TOTH, "State-Space Relaxation Procedures for the Computation of Bounds to Routing Problems," *Networks* **11,** 145–164 (1981).
8. J. J. JAW, A. R. ODONI, H. N. PSARAFTIS AND N. H. M. WILSON, "A Heuristic Algorithm for the Multi-Vehicle Many-to-Many Advance-Request Dial-A-Ride Problem," Working Paper MIT-UMTA-82-2, Massachusetts Institute of Technology, 1982.