The Finite Capacity Dial-A-Ride Problem

MOSES CHARIKAR *
Stanford University
moses@cs.stanford.edu

BALAJI RAGHAVACHARI †
University of Texas, Dallas
rbk@utdallas.edu

Abstract

We give the first non-trivial approximation algorithm for the Capacitated Dial-a-Ride problem: given a collection of objects located at points in a metric space, a specified destination point for each object, and a vehicle with a capacity of at most k objects, the goal is to compute a shortest tour for the vehicle in which all objects can be delivered to their destinations while ensuring that the vehicle carries at most k objects at any point in time. The problem is known under several names, including the Stacker Crane problem and the Dial-a-Ride problem. No theoretical approximation guarantees were known for this problem other than for the cases $k = 1, \infty$ and the trivial $O(k)$ approximation for general capacity k. We give an algorithm with approximation ratio $O(\sqrt{k})$ for special instances on a class of tree metrics called height-balanced trees. Using Bartal's recent results on the probabilistic approximation of metric spaces by tree metrics, we obtain an approximation ratio of $O(\sqrt{k} \log n \log \log n)$ for arbitrary n point metric spaces. When the points lie on a line (line metric), we provide a 2-approximation algorithm.

We also consider the Dial-a-Ride problem in another framework: when the vehicle is allowed to leave objects at intermediate locations and pick them up at a later time and deliver them. For this model, we design an approximation algorithm whose performance ratio is $O(1)$ for tree metrics and $O(\log n \log \log n)$ for arbitrary metrics.

We also study the ratio between the values of the optimal solutions for the two versions of the problem. We show that unlike in k-delivery TSP in which all the objects are identical, this ratio is not bounded by a constant for the Dial-a-Ride problem, and it could be as large as $\Omega(k^{2/3})$.

*Computer Science Department, Stanford University, CA 94305. Research supported by the Pierre and Christine Lamond Fellowship and in part by an ARO MURI Grant DAAH04-96-1-0007 and NSF Award CCR-9357849, with matching funds from IBM, Schlumberger Foundation, Shell Foundation, and Xerox Corporation.

†Department of Computer Science, The University of Texas at Dallas, Richardson, TX 75083-0688. Research supported in part by NSF Research Initiation Award CCR-9409625.

1 Introduction

Vehicle-routing and delivery problems occur in many practical settings in varied areas such as transportation of goods and robotics (see the survey by Christofides [12] and the book by Golden and Assad [18].) Many important problems that arise are NP-hard and a lot of researchers in Computer Science, Management and Operations Research have concentrated on outlining heuristics to solve these problems.

The problem we consider is called the *Capacitated Dial-a-Ride* problem, and is defined as follows. The input is a collection of objects located at points in a metric space, a specified destination point for each object, and an integer k — the capacity of the vehicle used for making the deliveries. The goal is to compute a shortest tour for the vehicle in which all objects can be delivered to their destinations while ensuring that the vehicle carries at most k objects at any point in time. The problem is known under several names, including the Stacker Crane problem and the capacitated Dial-a-Ride problem. It generalizes the Traveling Salesman Problem (TSP) even for $k = 1$ and is therefore NP-hard. There are two variants of the problem: the non-preemptive case in which, once an object is loaded on the vehicle it stays on it until it is delivered to its destination, and the preemptive case in which an object may be dropped at intermediate locations and then picked up later by the vehicle and delivered. The performance ratio of the algorithms depend critically on whether "drops" are allowed or not. Henceforth, unless otherwise stated, the Capacitated Dial-a-Ride problem will refer to the non-preemptive case.

Dial-a-Ride systems involve dispatching a vehicle to satisfy demands from a set of customers who call a vehicle operating agency requesting that an item be picked up from a specific location and delivered to a specific destination. Such demand-responsive transportation systems have been in operation in several metropolitan areas. The Dial-a-Ride problem arises in several practical applications [26] such as the transportation of elderly and/or disabled persons, telebuses, shared taxi services [33], certain courier services and so on. Although single-vehicle Dial-a-Ride systems are not very common, single vehicle Dial-a-Ride algorithms are used as subroutines in large scale multivehicle Dial-a-Ride environments and this is why the single vehicle Dial-a-Ride problem is important.

Previous Work: For the past 20 years, several papers have presented exact exponential time dynamic programming algorithms [7, 13, 27, 28], heuristic algorithms [21, 22, 26, 29, 30] and algorithms based on the polyhedral structure of integer programming formulations [31]. Most research has focussed on empirical evaluation of algorithms. For a description of many of these approaches, see [8] and the recent survey by Savelsbergh and Sol [32].

The theoretical work on this problem has been restricted to algorithms for special cases, either $k = 1$ or $k = \infty$ or instances on simple metric spaces such as paths, cycles or trees. For the unit capacity non-preemptive case, Frederickson, Hecht and Kim [17] gave an algorithm with an approximation factor of 1.8. Several papers have discussed fast implementations of exact algorithms for special instances of the unit-capacity case [4, 14, 15, 16]. Frederickson and Guan [16] showed that the unit-capacity non-preemptive case is NP-hard on trees. A special case of the preemptive problem on the line for general k was considered by Karp [23] (see also Knuth [24, Section 5.4.8]) who called it the *static elevator scheduling problem.* Guan [19] improved Karp's result to give a fast algorithm for the preemptive case on the line for general k and proved NP-hardness for the non-preemptive problem on the line with $k = 2$ as well as the preemptive problem on trees with $k = 2$. Arkin, Hassin and Klein [2] investigated the problem with $k = \infty$ and considered the ratio of the optimal solution of the problem under various types of restrictions to the optimal solution of the unrestricted problem. Many papers have discussed implementation heuristics for the infinite-capacity problem. For restricted instances where all the objects are initially located at a central depot, Haimovich and Rinnooy Kan [20] gave a 3-approximation algorithm. For graphs induced by points in the plane with Euclidean distances as edge-weights, they gave a polynomial time approximation scheme for constant k. Asano *et al* [3] gave a polynomial-time approximation scheme for the same special case in the geometric setting, when k is $O(\log n/\log\log n)$.

A closely related problem is the k-delivery TSP in which the objects are identical and the algorithm can deliver any object to any destination. The first constant factor approximation algorithm for the general k-delivery TSP was given by Chalasani, Motwani and Rao [9], which was subsequently improved [1, 11]. Charikar, Khuller and Raghavachari [11] studied the ratio of the cost of the optimal nonpreemptive solution to that of the optimal preemptive solution and showed that when all the objects are identical, this ratio is at most 4.

Our Results: Let n be the number of nodes in the input graph (number of points in which there is an object initially or is a destination point). Let k be the capacity of the vehicle. We use the recent results on probabilistic approximation of metric spaces by tree metrics [5, 6, 10] to reduce the problem on a general metric space to an instance of a special form on a class of trees called *height-balanced* trees. We give an algorithm for the Capacitated Dial-a-Ride problem with approximation ratio $O(\sqrt{k})$ for special instances on *height-balanced* trees. This gives us a randomized $O(\sqrt{k}\log n\log\log n)$ approximation algorithm in arbitrary metric spaces. The algorithm can be derandomized using the results of [10]. When the points lie on a line (line metric), we provide a 2-approximation algorithm.

We also consider the preemptive case of the Capacitated Dial-a-Ride problem — when the vehicle is allowed to leave objects at intermediate locations and pick them up at a later time and deliver them. For this model, we design an approximation algorithm whose performance ratio is $O(1)$ for tree metrics and $O(\log n\log\log n)$ for arbitrary metrics.

Following [11], we study the ratio of the cost of the optimal non-preemptive solution to the cost of an optimal preemptive solution. We show that unlike k-delivery TSP in which all the objects are identical, this ratio is not bounded by a constant for the Dial-a-Ride problem, and it could be as large as $\Omega(k^{2/3})$.

Our Contribution: Before our work, no theoretical approximation guarantees were known for this problem for general k other than the trivial $O(k)$ approximation, which is obtained by simply taking the $O(1)$ approximation algorithm for the unit capacity case. All previous work has been restricted to either the unit-capacity case ($k = 1$) or the infinite-capacity case ($k = \infty$) or line/cycle/tree metrics. In fact, prior to our work, no non-trivial approximation guarantee was known even for tree metrics for general k. We show the first non-trivial bounds for the Capacitated Dial-a-Ride problem for arbitrary metric spaces by developing new lower bounds for the optimal non-preemptive solution. We also show that there could be a significant gap between the values of the optimal solutions for the preemptive and non-preemptive cases.

We briefly describe the remainder of the paper. In Section 2, we explain how an instance of the problem in an arbitrary metric space can be reduced to an instance of a special form on a *height-balanced* tree by losing a factor of $O(\log n\log\log n)$ in the approximation ratio. In Section 3, we study the preemptive Dial-a-Ride problem. For instances on a tree, we develop simple lower bounds on the value of the optimal solution called the *Steiner tree* bound and the *flow* bound. We use these to develop a 2-approximation algorithm for preemptive Dial-a-Ride on trees. In Section 4, we study the non-preemptive Dial-a-Ride problem. We develop a new lower bound on the value of the optimal solution called the *wait* lower bound. We use this to design an $O(\sqrt{k})$ approximation algorithm for special instances on *height-balanced* trees. In Section 5, we demonstrate that the analysis of our algorithm for height-balanced trees is tight. We also construct an instance for which the value of the optimal is a factor of $\Omega(k^{1/6})$ away from all our three lower bounds. This emphasizes the need to develop a better understanding of the structure of the optimal tour in order to significantly improve on our results. We also prove that the ratio between the values of the optimal non-preemptive and preemptive solutions could be as high as $\Omega(k^{2/3})$ and is bounded above by $k+1$.

2 Approximating General Metric Spaces by Tree Metrics

Let V be a set of n points. Consider a metric space M over V. For $u, v \in V$, the distance between u and v in the metric M is denoted by $d_M(u, v)$.

We recall some definitions from [5].

Definition 1 *A metric space N over V, α-approximates a metric space M over V if for every $u, v \in V$, $d_M(u, v) \leq d_N(u, v) \leq \alpha \cdot d_M(u, v)$.*

Definition 2 *A set of metric spaces $\mathcal{S}$ over V, α-probabilistically approximates a metric space M over V, if* **(1)** *for all $u, v \in V$ and $N \in \mathcal{S}$, $d_N(u, v) \geq d_M(u, v)$, and* **(2)** *there exists a probability distribution $\mathcal{D}$ over metric spaces $N \in S$ such that for every $u, v \in V$, $\mathbf{E}_{\mathcal{D}}[d_N(u, v)] \leq \alpha \cdot d_M(u, v)$.*

Definition 3 *A k-hierarchically well-separated tree (k-HST) is defined as a rooted weighted tree such that* **(1)** *The edge weight from any node to each of its children is the same.* **(2)** *The edge weights along any path from the root to a leaf are decreasing by a factor of at least k.*

Theorem 1 (Bartal [6]) *Every metric space M over V can be α-probabilistically approximated by the set of 2-HSTs, where $\alpha = O(\log n \log \log n)$. Here the points in V occur as the leaves of the 2-HST, the internal nodes (and the root) are dummy nodes.*

Actually, we will use a slightly different kind of a tree, which we call a *height-balanced* tree.

Definition 4 *A* height-balanced *tree is a rooted weighted tree such that:* **(1)** *The distance from the root to any leaf is the same for all leaves.* **(2)** *For all leaves, the path from the root to the leaf consists of the same number m of edges.* **(3)** *For $i = 1, \ldots, m$, define level i to be all the edges which are the ith edge on the (shortest) path from the root to some leaf. Then edges in the same level have the same length.*

Clearly, properties 2 and 3 imply property 1. In fact, all we need is property 1. The other two properties can be satisfied by breaking edges and introducing new vertices.

Lemma 2 *Every 2-HST can be 4-approximated by a* height-balanced *tree.*

Proof: Consider a 2-HST. First round up all edges to the nearest power of 2. Edge lengths (and all distances) are at most doubled (and none reduced) in this process. Let L be the set of (rounded up) edge lengths in the tree. Let $m = |L|$. We will ensure that the path from the root to any leaf consists of m edges, one corresponding to each element of L, such that the length of the edge corresponding to $x \in L$ is x.

We walk down the path from the root to every leaf and add edges according to the following rule. First, all edge lengths must be strictly decreasing as we go from the root to a leaf because we started with a 2-HST. Suppose e, e' are consecutive edges on the path. Let l be the length of e and let l' be the length of e'. If there are no elements in L whose values are between l and l', we proceed. Else, let $l_1 > l_2 > \ldots > l_r$ be the elements of L strictly between l and l'. We add edges of lengths $l_1, l_2, \ldots, l_r$ (in that order) between e and e'. We can think of this as increasing the length of e by an amount $l_1 + l_2 + \ldots + l_r$. But this is at most $l/2 + l/4 + \ldots \leq l$. Thus we at most double the length of edge e. Again, all distances are at most doubled (and none reduced) by this transformation. The tree we finally obtain is *height-balanced* and it is a 4-approximation of the 2-HST with which we started. □

By Theorem 1 and Lemma 2, we can take an instance I of Capacitated Dial-a-Ride on a general metric space and convert it into an instance I' on a *height-balanced* tree such that the sources and destinations of all items are at the leaves of the tree. Let *OPT* be the cost of the optimal solution to I and let OPT' be the cost of the optimal solution to I'. Then $E(OPT') \leq \alpha \cdot OPT$, where $\alpha = O(\log n \log \log n)$. (Here the expectation is over the random choice of a 2-HST.) Also a solution S' to I' can be converted to a solution S of I, such that $cost(S) \leq cost(S')$. If we have a β approximation algorithm for instances of the form I', this gives us an $\alpha \cdot \beta$ approximation algorithm for general instances.

We now proceed to study instances of Capacitated Dial-a-Ride on *height-balanced* trees such that the sources and destinations of all items are at leaves of the tree.

3 Preemptive Dial-a-Ride

Consider an instance I of Capacitated Dial-a-Ride on a tree. We will first develop two easy lower bounds on the optimal solution. We then present a simple 2-approximation algorithm for Capacitated Dial-A-Ride on trees when preemption is allowed. The results of this section hold for instances for any tree, not just the restricted class of height-balanced trees.

Every item x is associated with a (unique) path $p(x)$ from the source of x to the destination of x. Let the *turning point* of x be the highest node in the tree on the path $p(x)$; this is the least common ancestor of the source and destination nodes of x. Let v be the turning point of x. The path $p(x)$ first moves upwards from the source of x to v, and then downwards from v to the destination of x.

A vertex in the tree is called *interesting* if it lies on a path $p(x)$ for some item x. Let T' be the Steiner tree spanning all interesting points. Clearly any tour that performs all the pickups and deliveries must visit every interesting point. Thus any tour must traverse every edge in T' at least 2 times. We refer to this as the *Steiner tree* lower bound. For tree metrics, there is no loss of generality in assuming that the algorithm is restricted to the subtree T', and edges not in T' are discarded. We will therefore assume that the tree T is in fact the Steiner tree spanning all interesting points.

For an edge $e \in T$, let $flow_u(e)$ be the total number of paths $p(x)$ that pass upwards through the edge e, i.e., the total number of items whose sources are in the subtree rooted at e and their destinations lie outside this subtree. Similarly, let $flow_d(e)$ be the total number of paths that pass downwards through the edge e. Since the vehicle can carry at most k items at a time, any tour that performs all the pickups and deliveries must traverse the edge e at least $2\max(\lceil \frac{flow_u(e)}{k} \rceil, \lceil \frac{flow_d(e)}{k} \rceil)$ times. We call this the *flow* lower bound.

Note that both the *Steiner tree* and the *flow* lower bounds hold for the preemptive as well as the nonpeemptive case. We now describe a simple algorithm that gives a 2-approximation for preemptive Dial-a-Ride on a tree. We will perform two depth-first traversals of the tree T. In the first, called the *up sweep*, each item x is moved *up* from its source to its turning point. In the second traversal, called the *down sweep*, each item x is moved *down* from its turning point to its destination.

The usual depth-first traversal traverses every edge e twice. First e is traversed downwards, the subtree rooted at e is explored, then e is traversed upwards. We modify the usual depth first search as follows. In the *up sweep*, we will recursively ensure that after the subtree rooted at e is explored, all $flow_u(e)$ items that need to move up the edge e are collected at the lower end point of e. We then traverse the edge e back and forth carrying at most k items at a time and moving all these items up the edge e. Since preemption is allowed, we can drop the items off at the upper end point of e and proceed. It is easy to see that an edge e is traversed $2\max(1, \lceil \frac{flow_u(e)}{k} \rceil)$ times in the *up sweep*. Similarly, in the *down sweep*, we recursively ensure that before the subtree rooted at e is explored, all edges $flow_d(e)$ items that need to be moved down the edge e are collected at the upper end point of e. We then traverse the edge e back and forth carrying at most k items at a time and moving all these items down the edge e. Since preemption is allowed, we can drop the items off at the lower end point of e and proceed. It is easy to see that an edge e is traversed $2\max(1, \lceil \frac{flow_d(e)}{k} \rceil)$ times in the *down sweep*. It is fairly straightforward to check the correctness of the algorithm and verify that the assumptions made in the two sweeps are recursively satisfied.

Note that in both the *up sweep* and the *down sweep*, the total number of traversals of edge e is bounded above by the maximum of the *Steiner tree* lower bound and the *flow* lower bound for e. Thus the tour produced by the algorithm is at most 2 times the length of an optimal tour, giving us the following theorems.

Theorem 3 *There is a polynomial-time 2-approximation algorithm for preemptive Capacitated Dial-a-Ride on trees.*

Theorem 4 *There is a polynomial time $O(\log n \log\log n)$-approximation algorithm for preemptive Capacitated Dial-a-Ride on arbitrary metrics.*

4 Non-preemptive Dial-a-Ride

For the non-preemptive case, there can be a significant gap between the value of the optimal and the lower bounds developed in the previous section. In fact we will prove later that there are instances for which this gap is $\Omega(k^{2/3})$. This means that if our analysis uses only these bounds, our approximation guarantee will be $\Omega(k^{2/3})$. In this section, we develop a better lower bound for non-preemptive Dial-a-Ride on *height-balanced* trees and then present an approximation algorithm for non-preemptive Dial-a-Ride.

4.1 The Wait Lower Bound

Given an instance of Capacitated Dial-a-Ride on a *height-balanced* tree, we derive a lower bound on the optimal cost for this instance. Intuitively, the idea behind the lower bound is as follows. Consider a vertex v that has several items that need to be delivered to different destinations. The optimal tour visits v a certain number of times. If a large number of items at v are picked up in a particular visit, then since none of the items can be dropped off at intermediate locations and the destinations of each of these items is distinct, the average wait time (i.e. time spent in the vehicle) for these items must be large. Now, if the tour visits v a large number of times, the contribution of v to the length of the tour is large. If, one the other hand, the tour visits v a few times, a lot of items must be picked up on each visit. In that case, the average wait time for the items is large, and again the contribution to the length of the tour is large. This idea can be used to derive a lower bound on the optimal, called the *wait* bound.

For every level l, we will prove a lower bound on the total number of traversals of level-l edges for any tour that satisfies the instance. It is useful to think of the lower bound in the following manner. We will associate two counters with every edge in the tree. For edge e in the tree, let $U(e)$ and $D(e)$ be the two counters associated with e. Consider any tour that satisfies the instance. Initially all counters are set to 0. We will traverse the tour from start to finish and while doing so, increase the counter values in a certain way. At the end of the traversal, the counter values will give us a lower bound on the length of the tour.

The counter values are incremented as follows. The counters for an edge at level l are only affected by traversals of edges at level l. Consider an edge e at level l. When e is traversed in the upward direction, $U(e)$ is incremented by 1. When e is traversed in the downward direction, several counters are increased, corresponding to the items carried by the vehicle while traversing the edge. Let x be an item carried by the vehicle at that time. Suppose x originates from the subtree rooted at edge e_x at level l. Then $D(e_x)$ is incremented by $1/k$. Since the vehicle carries at most k items, the net increase of the counters is at most 1. Thus, each time a level-l edge is traversed, the sum of counter values of edges at level l is increased by at most 1. This ensures that the sum of the counter values of level-l edges is at most the number of times its edges were traversed by the tour.

We will now give a lower bound on the sum of counter values of edges at level l. For edges e, e' at level l, define an (e, e')-item to be an item that must be transported from the subtree rooted at e to the subtree rooted at e'. Let $\mathcal{C}_{e,e'}$ be the set of (e, e')-items. Let $y_{e,e'}$ be the number of (e, e')-items. Two items are said to be distinct if they belong to distinct sets $\mathcal{C}_{e,e'}$.

Consider an edge e at level l. Let T_e be the subtree rooted at e. We will focus on the set S of items that originates in T_e and needs to be delivered to vertices outside T_e. Any tour must visit the subtree T_e. For every visit to T_e, $U(e)$ is incremented by 1. Consider the portion of the tour between visit i and visit $i+1$ to T_e. We refer to this as interval i.

Let n_i be the number of distinct items from S that are delivered to their destinations during interval i. Since the items are distinct, they have to be delivered to distinct subtrees $T_{e'}$. Thus the tour must visit some n_i distinct subtrees $T_{e_j}, j = 1 \ldots n_i$ during interval i. Suppose these subtrees are traversed in the order $T_{e_1}, \ldots, T_{e_{n_i}}$. When T_{e_1} is visited, the vehicle contains at least n_i distinct elements from S. Then $D(e)$ is incremented by at least n_i/k. In general, when T_{e_j} is visited, the vehicle contains at least $n_i - j + 1$ distinct commodities from S, and $D(e)$ is incremented by at least $(n_i - j + 1)/k$. Thus the total increase in $D(e)$ during interval i is at least $n_i(n_i + 1)/2k$.

We will charge the total increase in $U(e) + D(e)$ during interval i to the n_i distinct items delivered during interval i. The total increase is at least $1 + n_i(n_i+1)/2k$. Thus, the average contribution of each item is $1/n_i + (n_i+1)/2k$. This is minimized for $n_i = \sqrt{2k}$. Thus the average contribution of each item is at least $1/\sqrt{k}$.

Suppose that T_e is visited by the tour r times. At the end of the tour, the value of $U(e) + D(e)$ is at least

$$\frac{1}{\sqrt{k}} \sum_{i=1}^{r} n_i .$$

Note that the number of items delivered in interval i is at most k. Thus, the $y_{e,e'}$ items in $\mathcal{C}_{e,e'}$ must be delivered in at least $\lceil \frac{y_{e,e'}}{k} \rceil$ intervals, i.e., there must be at least $\lceil \frac{y_{e,e'}}{k} \rceil$ intervals in which some (e, e')-item is delivered. Hence

$$\sum_{i=1}^{r} n_i \geq \sum_{e' \in \text{ level } l} \left\lceil \frac{y_{e,e'}}{k} \right\rceil .$$

This implies that

$$U(e) + D(e) \geq \frac{1}{\sqrt{k}} \sum_{e' \in \text{ level } l} \left\lceil \frac{y_{e,e'}}{k} \right\rceil .$$

Thus the sum of the counter values for all edges at level l is at least

$$\frac{1}{\sqrt{k}} \sum_{e,e' \in \text{ level } l} \left\lceil \frac{y_{e,e'}}{k} \right\rceil .$$

This gives us the following lemma.

Lemma 5 *For any solution that satisfies the instance, the total number of traversals of level-l edges is at least*

$$\frac{1}{\sqrt{k}} \sum_{e,e' \in \textit{ level } l} \left\lceil \frac{y_{e,e'}}{k} \right\rceil .$$

We call this the *wait* lower bound.

4.2 The Approximation Algorithm

We now present our approximation algorithm for the Capacitated Dial-a-Ride problem in height-balanced trees. Recall that all edges of the tree at the same level have equal length and all items are located at the leaves of the tree. Let $S(v)$ be the set of all items whose turning point is v. We can think of $S(v)$ as a collection of paths $p(x)$ for every item $x \in S(v)$.

Consider all vertices of the tree that are turning points of some item. The algorithm visits them in a depth-first order. When a vertex v is visited, it delivers all items in

$S(v)$. We refer to this process of delivering all items in $S(v)$ as *servicing* v.

The algorithm services a node v as follows: let m be the number of levels in the tree. Suppose the subtree rooted at v consists of edges in levels $h+1,\ldots,m$. Then, every path in $S(v)$ has exactly two edges in each of the levels $h+1,\ldots,m$. For each object $x \in S(v)$, the path from the source to the destination traverses one level-l edge upwards (towards the root) and the other level-l edge downwards (away from the root). The former is called the *upward* level-l edge, and the latter the *downward* level-l edge, of x.

We associate with every item x, an edge sequence $E(x)$, consisting of the edges of $p(x)$ sorted in the order of increasing level, and within a level the upward edges are placed before the downward edges. Formally, $E(x) = (e_1, e_2, \ldots e_{2i-1}, e_{2i}, \ldots e_{2(m-h)})$, where e_{2i-1} is the *upward* level-$(h+i)$ edge in $p(x)$ and e_{2i} is the *downward* level-$(h+i)$ edge in $p(x)$.

Now order the edges of the tree arbitrarily and sort the items in $S(v)$ in lexicographic order of their edge sequences $E(x)$ defined above. Group the items according to this sorted order, taking the first k to be the first group, the next k to be the next group and so on. For each group, we start from v, perform all pickups, return to v, perform all deliveries and end at v. The pickups are done by visiting all the sources of items in the group in a depth-first order. Similarly, the deliveries are done by visiting all the destinations of items in the group in a depth-first order.

We now analyze the cost of the solution returned by the algorithm. We can divide the traversals of edges into two types: (1) edge traversals while performing the depth first search of turning points, and (2) edge traversals while *servicing* nodes. Observe that each edge is traversed at most 2 times while performing the depth first search. This is bounded by the *Steiner tree* lower bound on optimal. For edge traversals while *servicing* nodes, we will bound the total number of level-l edges traversed for each level l and compare this with the *wait* bound.

For each vertex v that is a turning point of some item, the algorithm groups the items in $S(v)$ by sorting the edge sequences. We can prove the following property of the sorted order.

Lemma 6 *Let e, e' be edges at level $l \in [h+1, m]$, such that v is the turning point of the shortest path from e to e'. Then, the number of groups that contain some (e, e')-item is at most $2\lceil \frac{y_{e,e'}}{k} \rceil$.*

Proof: We will first prove that all the (e, e')-items occur consecutively in the sorted order of edge sequences in $S(v)$. Consider any (e, e')-item $x \in S(v)$. The path $p(x)$ traverses edge e upwards and edge e' downwards. The portion of the path from e to e' must be the shortest path from e to e'. Consider the edge sequence $E(x)$ associated with x. $E(x) = (e_1, \ldots, e_{2i-1}, e_{2i}, \ldots, e_{2(m-h)})$. Then for $1 \le i \le 2(l-h)$, e_{2i-1} must be the upward level-$(h+i)$ edge in the path from e to e' and e_{2i} must be the downward level-$(h+i)$ edge on this path. Thus, for any (e, e')-item x, the first $2(l-h)$ edges in the edge sequence are uniquely determined by the edges on the shortest path from e to e'. This means that the first $2(l-h)$ edges in the edge-index sequence are the same for all (e, e')-items. Also, for any item that is not an (e, e')-item, at least one of the first $2(l-h)$ edges in the edge index sequence must be different. Hence, all the (e, e')-items occur consecutively in the sorted order.

Now, the number of (e, e')-items is $y_{e,e'}$. Recall that the groups are formed by taking the first k items in the sorted order, the next k in the sorted order and so on. Hence the number of groups that contain some (e, e')-item is at most $1 + \lceil \frac{y_{e,e'}}{k} \rceil \le 2\lceil \frac{y_{e,e'}}{k} \rceil$. □

Lemma 7 *The total number of times level-l edges are traversed while servicing nodes is at most*

$$8 \sum_{e,e' \in \text{ level } l} \left\lceil \frac{y_{e,e'}}{k} \right\rceil .$$

Proof: The algorithm gathers items in $S(v)$ into groups of size at most k, picks up all the items by visiting the sources in a depth-first order, and then delivers the items by visiting the destinations in a depth-first order.

Consider the process of picking up all the items in a group. The algorithm performs a depth-first search of the tree formed by taking the union of the shortest paths from each source to v. Each edge traversed while doing this is traversed exactly twice, once downwards and once upwards. Further, an edge e at level l is traversed iff there is some item in the group such that e is on the path from the source of x to v, i.e., iff there is an (e, e')-item in the group for some edge e' at level l.

Similarly, for the process of delivering all the items in a group, an edge e at level l is traversed iff there is an (e', e)-item in the group for some edge e' at level l.

Let $n_{e,e'}$ be the number of groups that contain an (e, e')-item. Thus, an edge e at level l is traversed at most $2 \sum_{e' \in \text{ level } l} (n_{e,e'} + n_{e',e})$ times. Summing over all edges e at level l, we get the total number of traversals of edges at level l is at most

$$2 \sum_{e,e' \in \text{ level } l} (n_{e,e'} + n_{e',e}) = 4 \sum_{e,e' \in \text{ level } l} n_{e,e'} .$$

Now, given edges e, e' at level l, there is a unique vertex v such that v is the turning point of the shortest path from e to e'. All (e, e')-items are delivered only when v is serviced. Thus all (e, e')-items can only belong to the groups formed when v is serviced. By Lemma 6, $n_{e,e'} \le 2\lceil \frac{y_{e,e'}}{k} \rceil$. The lemma follows. □

Comparing the upper bound in Lemma 7 with the *wait* bound in Lemma 5, we see that the total cost of servicing nodes is at most $8\sqrt{k}$ times the optimal. As observed earlier, the total cost of the depth first search is bounded by the optimal. This gives us the following theorem.

Theorem 8 *There is an polynomial time $O(\sqrt{k})$-approximation algorithm for the capacity k Dial-a-Ride problem on* height-balanced *trees.*

Theorem 9 *There is a polynomial-time approximation algorithm to solve the capacity k Dial-a-Ride problem on* any *n-point metric space with an approximation ratio of $O(\sqrt{k}\log n \log\log n)$.*

Remark: We can use both the *flow* and *wait* lower bounds for *height-balanced* trees to prove that, if the algorithm uses a vehicle of capacity αk and we compare the cost of the solution obtained with the cost of an optimal solution for a capacity-k vehicle, then the approximation ratio is $O(\max(\sqrt{k}, 1/\alpha))$. Thus, the algorithm achieves an $O(\sqrt{k})$ approximation even by using only a capacity of $\sqrt{k}$. Also, the current analysis does not take advantage of higher capacity; i.e., for any capacity $> \sqrt{k}$, the approximation ratio is still $O(\sqrt{k})$.

4.3 Line metrics

For points on a line, where the distances are measured along the line, we provide an algorithm with approximation ratio of 2 for the Capacitated Dial-a-Ride problem (non-preemptive case).

Guan [19] showed that the problem can be solved optimally in polynomial time if drops are allowed (preemptive case). Due to lack of space, we omit the proof of the following theorem.

Theorem 10 *There is an algorithm that runs in polynomial time for the non-preemptive Capacitated Dial-a-Ride problem with a vehicle of capacity k, in which all points are on a line (line metric) and outputs a tour that is at most twice the optimal tour.*

5 Gaps and lower bounds

We consider the lower bound we obtained on the cost of a solution to the Dial-a-Ride problem for several instances and prove results about the gap between this bound and the optimal cost.

5.1 Tightness of analysis

Let $\mathcal{I}_1$ be an instance on a star with $k+1$ nodes (in addition to a root); as usual k is the capacity of the vehicle. For every pair of distinct nodes u and v, there is an item at u that needs to be delivered to v.

Lemma 11 *For $\mathcal{I}_1$, the optimal cost is $\Theta(k\sqrt{k})$, and this is within $O(1)$ of the* wait *lower bound.*

The following theorem shows that the analysis of the algorithm is tight.

Theorem 12 *For $\mathcal{I}_1$, the approximation ratio achieved by the algorithm is $\Omega(\sqrt{k})$.*

Proof: For the instance $\mathcal{I}_1$, our algorithm returns a solution whose cost is $\Omega(k^2)$. By Lemma 11, the cost of an optimal solution is $\Theta(k\sqrt{k})$. Thus the approximation ratio is $\Omega(\sqrt{k})$. □

5.2 Gaps

We now show gaps between the lower bound and an optimal solution, showing that the performance guarantee that we derive is almost the best possible using this lower bound technique. In addition, we also explore the ratio between the cost of the optimal preemptive and non-preemptive routing schemes for the same instance, and show that there are instances for which the ratio is $\Omega(k^{2/3})$.

In order to define instance I_2, we need to use a *finite projective plane* [25]. We review the definition of a finite projective plane, viewed as a set system.

Definition 5 *A finite projective plane of order m is a system of m^2+m+1 sets of $\{1, 2, \ldots, m^2+m+1\}$, such that*
(1) *Every pair $\{i, j\}$, $1 \le i < j \le m^2+m+1$ is contained in exactly one set,*
(2) *Every pair of sets intersects in exactly one element,*
(3) *Every element is contained in $m+1$ sets, and*
(4) *Every set contains $m+1$ elements.*

It is known that finite projective planes exist when their order m is a prime power. Consider a finite projective plane of order m. It consists of m^2+m+1 sets $S_1, S_2, \ldots, S_{m^2+m+1}$. We define instance $\mathcal{I}_2$ on a star with $n = m^2+m+1$ nodes (besides the root), where the vehicle capacity is $k = m+1$. Let the nodes be numbered $1, \ldots, m^2+m+1$. At vertex i, we place $m+1$ items initially, one for each element of S_i. The destination of the item corresponding to $j \in S_i$ is chosen to be j.

Theorem 13 *For $\mathcal{I}_2$, the cost of the optimal solution is at least a factor of $\Omega(k^{1/6})$ from the* Steiner tree, flow *and* wait *lower bounds.*

Proof: The *wait* lower bound for $\mathcal{I}_2$ is $\Theta(nk^{1/2})$. The other two lower bounds are $\Theta(n)$. We will prove that the optimal cost for $\mathcal{I}_2$ is $\Omega(nk^{2/3})$, using a better lower bound proof that uses the special structure of $\mathcal{I}_2$. The proof appears in Section 6. $\square$

Theorem 13 implies that any analysis that uses only these three lower bounds cannot establish an approximation ratio of $o(k^{1/6})$.

Theorem 14 *For $\mathcal{I}_2$, the ratio of the cost of the optimal solution with no drops allowed to the cost of the optimal solution with drops allowed is $\Omega(k^{2/3})$.*

Proof: The proof of Theorem 13 shows that the cost of the optimal solution without drops for $\mathcal{I}_2$ is $\Omega(nk^{2/3})$. On the other hand, the cost of an optimal solution when drops are allowed is $O(n)$, because all deliveries can be performed by visiting each vertex at most twice. First visit each vertex in turn, pick up all the items there (there are at most k of them) and drop them off at the root. Then visit each vertex again, this time carrying the items destined for that vertex from the root (there are at most k such items). The cost of this solution is clearly $O(n)$. $\square$

5.3 Simulation

Similar to k-delivery TSP [11], we show that a vehicle of capacity k with drops (preemptive) can be simulated by a vehicle of capacity 1 without drops (non-preemptive) where the cost increases by at most $O(k)$.

Theorem 15 *For an arbitrary instance of the Dial-a-Ride problem, the length of an optimal capacity one tour without drops is at most $k+1$ times the cost of the optimal capacity k tour with drops allowed.*

As a corollary, we get the following upper bound that complements Theorem 14.

Corollary 16 *For any instance of capacity k Dial-a-Ride, the ratio of the cost of the solution with no drops allowed to the cost of the solution with drops allowed is at most $k+1$.*

6 A Better Lower Bound

Consider an instance of Capacitated Dial-a-Ride on the star. For vertex v, let $S(v)$ denote the set of sources of the items that need to be delivered to v. An instance is called *almost disjoint* if for all pairs of vertices $u \neq v$, it is the case that $|S(u) \cap S(v)| \leq 1$.

We will prove a stronger lower bound on the length of the optimal tour for *almost disjoint* instances of Capacitated Dial-a-Ride on the star. To this end, we associate with every item x, a set of four counters $c_1(x), \ldots, c_4(x)$. All the counters are initially zero, and they are modified as follows: consider the ith visit to a particular vertex v by the tour. Suppose that a total of t items that originate from v are delivered to their destinations between the ith and $i+1$th visits to v. (Note that the t items need not have been picked up at the ith visit to v.) Then the c_1 value of each of the t items is set to $1/t$. When t items are dropped off at a vertex, the c_2 value for each of them is set to $1/t$. When a vertex is visited, the c_3 value for all items carried by the vehicle at that point is incremented by $1/k$. Thus, for an item x, $c_3(x)$ equals $1/k$ times the time spent by x in the vehicle (here time is measured by number of vertices visited.) When t items are dropped off at a vertex, the c_4 counter for each of them is assigned the average of the c_3 counter values for the t items. Thus the sum of the c_4 counter values for the t items equals the sum of their c_3 counter values. It can be shown that for each i, $i = 1, \ldots 4$, the sum of the c_i values for all the items is at most the length of the tour.

For any item x, we will prove a lower bound on $\sum_{i=1}^{4} c_i(x)$. This gives us a lower bound on the length of the tour. Consider the items that originate from vertex v. We will focus on the subset of the items that are delivered between the qth and $(q+1)$th visit to v. Suppose r items $x_1, \ldots, x_r$ are delivered to vertices $v_1, \ldots v_r$ (visited in that order.) Let X_i be the set of items delivered to v_i when x_i is delivered to it. Let S_i be the set of sources of the items in X_i. Then $S_i \cap S_j = \{v\}$ for all $1 \leq i < j \leq r$. Let $|X_i| = |S_i| = 1 + t_i$. Let $X'_i = X_i - \{x_i\}$ and $S'_i = S_i - \{v\}$. Then $|X'_i| = |S'_i| = t_i$.

Note that $c_2(x_i) = \frac{1}{1+t_i}$. The tour must visit, for each i, the t_i vertices in S'_i before it visits v_i.

Intuitively, the analysis proceeds as follows: If r is small, then the value of the c_1 counters is large. If not, a lot of items must have been delivered. If the t_i's are small, then the value of the c_2 counters will be large. If not, then the tour has to visit a lot of vertices. This means that one of two things must occur. Either the wait time for the x_i's is large, or the total wait time for all the items in the X_i's is large. In the first case, the sum of the c_3 counter values is large and otherwise the sum of the c_4 counter values is large.

Suppose $\alpha_i t_i$ of the items in X'_i are picked up before the qth visit to v. Note that all the items in the sets X'_i come from distinct destinations. Thus, a total of $\sum_{i=1}^{r} \alpha_i t_i$ vertices are visited before the qth visit to v, such that at least one item is picked up from each of these and none of them is delivered before the qth visit to v. The first of these items picked up must have a wait time of at least $\sum_{i=1}^{r} \alpha_i t_i$, the second must have a wait time of at least $\sum_{i=1}^{r} \alpha_i t_i - 1$, and so on. Consider the contribution to $k \sum_{i=1}^{r} c_4(x_i)$. The jth item picked up must have a wait time of at least $\sum_{i=1}^{r} \alpha_i t_i - j + 1$. Suppose it belongs to set X'_i. Then its contribution is at least $(\sum_{i=1}^{r} \alpha_i t_i - j + 1)/(1 + t_i)$. The sum of the contributions is minimized when the items in sets

X_i' with higher t_i values are picked up before items in sets X_i' with lower t_i values. Let $\pi : \{1,\ldots,r\} \Rightarrow \{1,\ldots,r\}$ be a permutation (the sorted order of the t_i's) such that $t_{\pi(1)} \geq t_{\pi(2)} \geq \ldots t_{\pi(r)}$. Then the sum of the contributions is minimized when the $\alpha_{\pi(1)}t_{\pi(1)}$ items of $X'_{\pi(1)}$ are picked up first, followed by the $\alpha_{\pi(2)}t_{\pi(2)}$ items of $X'_{\pi(2)}$ and so on. Note that the wait time of each of the $\alpha_{\pi(i)}t_i$ items of $X'_{\pi(i)}$ is at least $\sum_{j=i+1}^{r} \alpha_{\pi(j)}t_{\pi(j)}$. Thus,

$$k\sum_{i=1}^{r} c_4(x_i) \geq \sum_{i=1}^{r-1} \alpha_{\pi(i)}t_i \cdot \frac{1}{1+t_i} \sum_{j=i+1}^{r} \alpha_{\pi(j)}t_{\pi(j)}$$
$$\geq \frac{1}{2} \sum_{1\leq i<j\leq r} \alpha_{\pi(i)}\alpha_{\pi(j)}t_{\pi(j)}$$

Now, since $\alpha_i t_i$ of the items in X_i' are picked up before the qth visit to v, the remaining $(1-\alpha_i)t_i$ items must be picked up after the qth visit to v and before the visit to v_i where x_i and all the items in X_i' are delivered. Between the qth visit to v and the visit to v_i, all of the items $x_i,\ldots,x_r$ must be present in the vehicle. Thus the pickups of the $(1-\alpha_i)t_i$ items contribute to the wait times of the items $x_i,\ldots,x_r$. The total contribution to $k\sum_{i=1}^{r} c_3(x_i)$ is at least $(1-\alpha_i)(r-i+1)t_i$. It follows that

$$k\sum_{i=1}^{r} c_3(x_i) \geq \sum_{i=1}^{r}(1-\alpha_i)(k-i+1)t_i$$

For fixed $t_1,\ldots,t_r$, we wish to derive a lower bound on the sum $\sum_{i=1}^{r} c_3(x_i) + \sum_{i=1}^{r} c_4(x_i)$. We know that

$$k\sum_{i=1}^{r} c_3(x_i) + k\sum_{i=1}^{r} c_4(x_i) \geq \sum_{i=1}^{r}(1-\alpha_i)(r-i+1)t_i$$
$$+\frac{1}{2} \sum_{1\leq i<j\leq r} \alpha_{\pi(i)}\alpha_{\pi(j)}t_{\pi(j)}$$

Without loss of generality, the RHS is minimized when all the α_i are either 0 or 1. We will choose a subset $L \subseteq \{1,\ldots,r\}$ of size at least $\frac{r}{4}$ such that the value of the RHS is at least $\frac{r}{4}\sum_{i\in L} t_i$. Let $A_0 = \{i|\alpha_i = 0\}$ and $A_1 = \{i|\alpha_i = 1\}$. Let $A_0' = A_0 - \{3r/4,\ldots,r\}$. Note that $|A_0'| \geq |A_0| - r/4$. Also, for each $i \in A_0'$, the contribution to the RHS is at least $\frac{r}{4}t_i$. For $j \in A_1$ define $p(j)$, the set of *predecessors* of j to be $p(j) = \{i \in A_1 | \pi^{-1}(i) < \pi^{-1}(j)\}$. Note that there is some $j_0 \in A_1$ such that $|p(j_0)| = 0$, some $j_1 \in A_1$ such that $|p(j_1)| = 1$ and so on. For $j \in A_1$, the contribution to the RHS is $\frac{|p(j)|}{2}t_j$. Let $A_1' = A_1 - \{j \in A_1 : |p(j)| < r/2\}$. Observe that $|A_1'| \geq |A_1| - r/2$. For each $i \in A_1$, the contribution to the RHS is at least $\frac{r}{4}t_i$. Now, let $L = A_0' \cup A_1'$. Then $|L| \geq r/4$ and the value of the RHS is at least $\frac{r}{4}\sum_{i\in L} t_i$.

Now,

$$\sum_{i=1}^{r} c_2(x_i) + \sum_{i=1}^{r} c_3(x_i) + \sum_{i=1}^{r} c_4(x_i) \geq \sum_{i\in L} \frac{1}{t_i+1} + \frac{r}{4k}t_i$$
$$\geq \sum_{i\in L}\sqrt{\frac{r}{4k}} \geq \frac{r}{4}\sqrt{\frac{r}{4k}} = \frac{r\sqrt{r}}{8\sqrt{k}}$$

Also, $\sum_{i=1}^{r} c_1(x_i) = 1$. Thus the sum of the 4 counters, $\sum_{i=1}^{4} c_i(x)$, averaged over the r items $x_1,\ldots,x_r$ is at least

$$\frac{1}{r} + \frac{1}{8}\sqrt{\frac{r}{k}} \geq \frac{1}{3k^{1/3}}$$

Thus the average contribution of every item to the lower bound on optimal is at least $\Omega(1/k^{1/3})$.

Recall that we considered an instance on a star. Every vertex of the star (other than the root) can be identified by the edge e connecting it to the root. For edges e, e', let $y_{e,e'}$ be the number of items that need to be transported from the vertex at the end of e to the vertex at the end of e'. Then the preceding discussion implies the following lower bound on the optimal solution.

Theorem 17 *For an* almost-disjoint *instance of Capacitated Dial-a-Ride on a star, the value of the optimal solution is at least*

$$\Omega\left(\frac{1}{k^{1/3}}\sum_{e,e'}\left\lceil\frac{y_{e,e'}}{k}\right\rceil\right).$$

References

[1] S. Anily and J. Bramel, "Approximation algorithms for the capacitated traveling salesman problem with pick-ups and deliveries," *manuscript*, (1997).

[2] E. Arkin, R. Hassin and L. Klein, "Restricted delivery problems on a network," *Networks*, 29:205-216, (1997).

[3] T. Asano, N. Katoh, H. Tamaki and T. Tokuyama, "Covering points in the plane by k-tours: towards a polynomial time approximation scheme for general k," *Proc. 29th STOC* (1997), pages 275-283.

[4] M. J. Atallah and S. R. Kosaraju, "Efficient solutions to some transportation problems with applications to minimizing robot arm travel," *SIAM J. Comput.*, 17(5):569-591, (1988).

[5] Y. Bartal, "Probabilistic approximations of metric spaces and its algorithmic applications," *Proc. 37th FOCS*, (1996), pages 184-193.

[6] Y. Bartal. "On approximating arbitrary metrics by tree metrics", *Proc. 30th STOC*, (1998), pages 161-168.

[7] L. Bianco, A. Mingozzi, S. Riccaiardelli and M. Spadoni, "Exact and heuristic procedures for the traveling salesman problem with precedence constraints, based on dynamic programming", *INFOR*, 32(1):19-31, (1994).

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

[9] P. Chalasani, R. Motwani and A. Rao, "Algorithms for robot grasp and delivery," *2nd International Workshop on Algorithmic Foundations of Robotics*, (1996).

[10] M. Charikar, C. Chekuri, A. Goel, S. Guha and S. Plotkin, "Approximating a finite metric by a small number of tree metrics", *these proceedings*.

[11] M. Charikar, S. Khuller and B. Raghavachari, "Algorithms for capacitated vehicle routing," *Proc. 30th STOC*, (1998), pages 349-358.

[12] N. Christofides, "Vehicle routing," in *The traveling salesman problem*, edited by E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan and D. B. Shmoys, John Wiley & Sons, New York, Pages 431-448, (1985).

[13] J. Desroisers, Y. Dumas and F. Soumis, "A dynamic programming solution of the large scale single-vehicle dial-a-ride problem with time windows", *The American Journal of Mathematical and Management Sciences*, 6:301-325, (1986).

[14] G. N. Frederickson, "A note on the complexity of a simple transportation problem," *SIAM J. Comput.*, 22(1):57-61, (1993).

[15] G. N. Frederickson and D. J. Guan, "Preemptive ensemble motion planning on a tree," *SIAM J. Comput.*, 22(1):1130-1152, (1992).

[16] G. N. Frederickson and D. J. Guan, "Non-preemptive ensemble motion planning on a tree," *Journal of Algorithms*,15(1):29-60, (1993).

[17] G. N. Frederickson, M. S. Hecht and C. E. Kim, "Approximation algorithms for some routing problems," *SIAM J. Comput.*, 7(2):178-193, (1978).

[18] B. L. Golden and A. A. Assad, *Vehicle Routing: Methods and Studies*, North Holland, Amsterdam (1988).

[19] D. J. Guan, "Routing a vehicle of capacity greater than one," *Discrete Applied Mathematics*, 81:41-57, (1998).

[20] M. Haimovich and A. H. G. Rinnooy Kan, "Bounds and heuristics for capacitated routing problems," *Math. Oper. Res.* 10:527-542, (1985).

[21] P. Healy and R. Moll, "A new extension of local search applied to the dial-a-ride problem", *European Journal of Operations Research*, 83:83-104, (1995).

[22] J. Jaw, A. R. Odoni, H. N. Psaraftis and N. H. M. Wilson, "A heuristic algorithm for the multi-vehicle advance request dial-a-ride problem with time windows", *Transportation Research*, 20B(3):243-257, (1986).

[23] R. M. Karp, "Two combinatorial problems associated with external sorting," *Combinatorial Algorithms, Courant Comp. Sci. Symp.*, pages 17-29, Algorithmics Press, New York, (1972).

[24] D. E. Knuth, *The art of computer programming, vol 3: Sorting and Searching*, Addison Wesley, (1973).

[25] J. H. van Lint and R. M. Wilson, *A course in Combinatorics*, Cambridge University Press, (1992).

[26] O. B. G. Madsen, H. F. Ravn and J. M. Rygaard, "A heuristic algorithm for a dial-a-ride problem with time windows, multiple capacities, and multiple objectives", *Annals of Operations Research*, 60:193-208, (1995).

[27] H. N. Psarfatis, "A dynamic programming solution to the single vehicle many-to-many immediate request dial-a-ride problem", *Transportation Science*, 14(2):130-154, (1980).

[28] H. N. Psarfatis, "An exact algorithm for the single vehicle many-to-many dial-a-ride problem with time windows", *Transportation Science*, 17(3):351-357, (1983).

[29] H. N. Psarfatis, "k-interchange procedures for local search in a precedence-constrained routing problem", *European Journal of Operations Research*, 13:391-402, (1983).

[30] H. N. Psarfatis, "Analysis of an $O(N^2)$ heuristic for the single vehicle many-to-many Euclidean dial-a-ride problem", *Transportation Research*, 17B:133-145, (1983).

[31] K. S. Ruland and E. Y. Rodin, "The pickup and delivery problem: faces and branch-and-cut algorithm", *Computers Math. Applic.*, 33(12):1-13, (1997).

[32] M. W. P. Savelsbergh and M. Sol, "The general pickup and delivery problem", *Transportation Science*, 29(1):17-29, (1995).

[33] L. Suen, A. Ebrahim and M. Oksenhendler, "Computerised dispatching for shared-ride taxi operations in Canada", *Transportation Planning and Technology*, 7:33-48, (1981).