Approximation Algorithms for the Capacitated Traveling Salesman Problem with Pickups and Deliveries

Shoshana Anily,[1] Julien Bramel[2]

[1] *Faculty of Management, Tel-Aviv University, Tel-Aviv, Israel 69978*

[2] *Columbia Business School, 406 Uris Hall, Columbia University, New York, NY 10027*

Received December 1997; revised December 1998; accepted 2 March 1999

Abstract: We consider the Capacitated Traveling Salesman Problem with Pickups and Deliveries (CTSPPD). This problem is characterized by a set of n pickup points and a set of n delivery points. A single product is available at the pickup points which must be brought to the delivery points. A vehicle of limited capacity is available to perform this task. The problem is to determine the tour the vehicle should follow so that the total distance traveled is minimized, each load at a pickup point is picked up, each delivery point receives its shipment and the vehicle capacity is not violated. We present two polynomial-time approximation algorithms for this problem and analyze their worst-case bounds. © 1999 John Wiley & Sons, Inc. Naval Research Logistics 46: 654–670, 1999

1. INTRODUCTION

The Capacitated Traveling Salesman Problem with Pickups and Deliveries (CTSPPD) consists of n pickup points (hereafter called "blue" points) and n drop-off (delivery) points (hereafter called "red" points) and one vehicle of limited capacity $k \geq 1$. One blue point is designated as the starting and ending point, from which any solution must start and end. At each blue point is a load of unit size that can be delivered to any red point, each of which requests a load of unit size. The problem is to determine a minimum length feasible tour that picks up and delivers all loads and does not violate the vehicle capacity of k units.

The problem of transporting a commodity from a set of suppliers to a set of demand points with a fleet of limited capacity vehicles is called the Capacitated Vehicle Routing Problem with Pickups and Deliveries (CVRPPD). This model can capture many real-life transportation and distribution problems. Casco Golden, and Wasil [6] discuss applications of a special case (the Vehicle Routing Problem with Backhauls) in the grocery industry where the supermarkets are the delivery points and the grocery suppliers are the pickup points. In fact, Casco et al. [6] report that combining deliveries and pickups on a route has led to industry wide savings in distribution costs upwards of $160 million a year. Clearly, problems involving multiple depots or pickup points (e.g., [12]) and multiple dropoff points (such as school bus routing, see [5], or express mail delivery and pickup, messenger services etc.) can be modeled using the CVRPPD. There

Correspondence to: J. Bramel

Contract grant sponsor: National Science Foundation; contract grant number: Career DMI-97-02596

 CCC 0894-069X/99/060654-17

are also applications in robotics, automated guided vehicle systems and industrial drilling (see [4, 7]).

One important application of the CVRPPD (and the CTSPPD) occurs in the context of inventory repositioning. Assume a set of retailers (owned and operated by the same firm) are geographically dispersed in a region. Often, due to the random nature of demands, some retailers have an excess of inventory while others have such strong sales that they are in need of additional stock. In many cases, the firm may decide to transfer inventory at retailers that have experienced below average sales to those that have experienced above average sales. For the one product problem, determining the cheapest way to execute a given stock transfer is exactly the CVRPPD (or, for the one vehicle case, the CTSPPD).

The assumption of unit size loads, in many cases, can be made without loss of generality. This is when each delivery or pickup load is allowed to be *split.* For instance, a delivery (or pickup) of size 7 units can be delivered (or picked up) in two parts, one for 3 units and later one for 4 units. Under this assumption, the CTSPPD (or CVRPPD) can handle essentially any load size since a delivery (pickup) of m units can be replaced by m red (blue) points at the same physical location. If we consider the inventory repositioning example above, since the retailers are owned by the firm, multiple visits may be tolerated. Imposing a constraint that a customer be visited only once is clearly more important when the customer is *external* to the firm. In those cases, multiple visits to the same customer is clearly an unsatisfactory service policy.

We will concentrate on the CTSPPD and point out in which instances our algorithm can be used for the CVRPPD. The CTSPPD has received scant attention in the literature in spite of its relevance in a variety of distribution systems. The worst-case analysis of heuristics for this problem has been limited to very special cases of the problem mainly because of tractability. For example, the case where the vehicle capacity is 1, or infinite, as well as the case where all blue (or red) points are at the same physical location have been analyzed. However, the more general problem of limited capacity and general locations for delivery and pickup customers has not been sufficiently studied mainly because of its complexity.

The CTSPPD is NP-hard since it includes the Traveling Salesman Problem (TSP) as a special case (i.e., if all red points are at the same location and $k \geq n$). This implies that the existence of a polynomial-time algorithm providing the optimal solution for each instance of the CTSPPD is unlikely to be found. We therefore concentrate our efforts on finding heuristics which have certain desirable properties. One such property is worst-case effectiveness. An *α-approximation algorithm* for a problem is an algorithm that guarantees that the length of the solution it creates is at most α times the length of the optimal solution. Alternatively, an α-approximation algorithm provides a solution with a *worst-case bound* of α. Our goal is to develop polynomial-time heuristics with worst-case bounds. Below, we review a number of well-known versions of the CTSPPD that can be solved by using the algorithms proposed in this paper. We also review some worst-case results that have been obtained for related versions of the CTSPPD.

Consider the Capacitated Vehicle Routing Problem (CVRP). In this problem, a central depot serves as the supply point for all shipments. Vehicles of limited capacity start and end their tours at the central depot and must bring product from the depot to the customers. Each customer requests a unit of product. This situation can be modeled using the CTSPPD by simply putting all the blue points at the depot's location and having each customer represented by a red point. Any solution to the CTSPPD corresponds to a solution to the CVRP and vice versa. For the CVRP, numerous worst-case results exist, including [1, 9]. In the conclusion, we describe the conditions under which our algorithms improve the best worst-case bounds known for the CVRP.

Consider the Multi-Depot Capacitated Vehicle Routing Problem (MCVRP) (see [12]). In this problem, in addition to a number of customers requesting product, there are several depots. At

each depot there are vehicles of limited capacity as well as a supply of product. The objective is to design routes for the vehicles such that each customer receives its shipment, the vehicle capacity is not exceeded and the total distance traveled is minimized. Most research on this problem (including the worst-case analysis of [12]) has assumed that there is an *unlimited* amount of product at each depot. If one models this problem using the CTSPPD, the model can capture the *more realistic* problem where each depot has a *limited supply* of product and (without loss of generality) total supply is equal to total demand. To do this, simply put s blue points at a depot with s units of supply. Each customer can be represented by a red point, or a number of red points equal to the size of the quantity demanded. In the conclusion, we describe the conditions under which our algorithms provide the best known worst-case bounds for the MCVRP.

The CTSPPD itself is a special case of the Swapping Problem (see [2]). There, the problem consists of a number of different commodities or product types. Each point is associated with the type of product currently at the point (if any) and the desired product type (if any). For each product, total demand is assumed to be equal to total supply. Anily and Hassin [2] present a 2.5-approximation algorithm for the Swapping Problem for the case where the vehicle capacity $k = 1$. This bound holds also when *drops* are allowed (a drop means that an object can be temporarily stored at an intermediate point on the vehicle route). Chalasani and Motwani [7] consider the special case of two product types which, in the context of the Swapping Problem, is equivalent to the CTSPPD. For $k = 1$, the problem is equivalent to finding an alternating tour between blue and red points of minimum length. The authors provide a 2-approximation algorithm, improving the above bound, based on solving a matroid intersection problem.

The CTSPPD with $k = \infty$ is studied in Anily and Mosheiov [3]. The problem in their paper is described in a somewhat different context. More specifically, the paper considers the Traveling Salesman Problem with Delivery and Backhauls. In this problem, a single vehicle of limited capacity, starting and ending at a depot, must serve customers that are partitioned into two groups: delivery and backhaul customers. At a delivery customer, the vehicle unloads one unit of product brought from the depot, while, at a backhaul customer, the vehicle loads one unit of product that is to be brought to the depot. Thus, when leaving the depot, the vehicle carries the *total* delivery requirement, whereas, when returning to the depot, the vehicle carries the total backhaul requirement. For the case where the number of delivery customers is identical to the number of backhaul customers and the vehicle capacity equals or exceeds the total delivery requirement, this problem is equivalent to the uncapacitated TSPPD (the CTSPPD with $k = \infty$); in order to see this, note that both problems reduce to finding a shortest closed tour where, at each point of the tour, the number of blue (delivery) customers visited so far does not fall below the number of red (back-haul) customers visited so far. Anily and Mosheiov provide a 2-approximation algorithm for this problem, based on doubling a minimum spanning tree.

The first to consider the general capacity case were Chalasani and Motwani [7]. They show that there exists a constant-approximation algorithm for general finite k. The algorithm they propose is a 9.5-approximation algorithm.

In this paper, we show that the same basic algorithm proposed in [7], with a slight refinement, yields a $(7 - 3/k)$-approximation algorithm for the CTSPPD. In our efforts to improve this bound, we were able to derive a second algorithm, called MATCH^k, whose worst-case bound is a (nonmonotone) function of k, and turns out to yield worst-case bounds which are smaller than $7 - 3/k$ for almost all practical values of k. For example, the worst-case bound for $k = 2$ is 2.5 (compared to 5.5), for $k = 4$ it is 3.25 (compared to 6.25), for $k = 8$ it is 3.875 (compared to 6.625), and for $k = 16$ it is 4.4375 (compared to 6.8125). The first value of k for which the worst-case bound of the second algorithm exceeds 7 is $k = 385$. Up to $k = 382$ (with

the exception of $k = 255$) the second algorithm provides better worst-case bounds than the first algorithm.

Finally, we note that our bounds are stronger for relatively small k. There are many cases where k is naturally not very large. The transportation of cars, large appliances, industrial machinery and handicapped children[1] are just a few examples where the size of the items being transported is a reasonable fraction of the vehicle capacity. There are obviously many examples where k is large. For those examples, alternate methods may be better suited.

The paper is organized as follows: In Section 2 we present notation and preliminaries, several useful bounds on the optimal solution value and structural properties which are used thereafter. In Section 3 we present our refinement of the algorithm proposed by Chalasani and Motwani [7], which we call the Iterated Tour Matching (ITM) Algorithm and prove its worst-case bound. In Section 4 we present the MATCH^2 Algorithm for the special case where the vehicle capacity is 2 and prove its worst-case bound. In Section 5 we present the MATCH^k Algorithm for a general vehicle capacity of k and prove its worst-case bound. In Section 6 we describe how our algorithm and worst-case bounds relate to other common vehicle routing problems.

2. NOTATION AND PRELIMINARIES

Let N^+ be the set of blue points (pickups) and let N^- be the set of red points (deliveries or dropoffs). Let $N \equiv N^+ \cup N^-$ be the set of all points, and let $|N^-| = |N^+| = n$ and thus $|N| = 2n$. We assume $k < n$, since the case $k \geq n$ is equivalent to $k = +\infty$ and was dealt with in [3]. From here on, we denote a blue point by b or b_i, and a red point by r or r_i. Let $b_0 \in N^+$ be the designated *starting and ending point*, i.e., the tour must start and end at b_0.

We define a complete undirected graph $\mathcal{G}$ with node set N and edge set $N \times N$. The length of arc (i, j) is denoted ℓ_{ij} and is equal to the distance between point i and point j. The distances ℓ_{ij} are assumed to be symmetric and satisfy the triangle inequality, i.e.,

$$\ell_{ij} \leq \ell_{ih} + \ell_{hj}, \qquad \forall\, h, i, j \in N.$$

For any set of arcs E, let $\ell(E) = \sum_{(i,j) \in E} \ell_{ij}$.

The Traveling Salesman Problem (TSP) will play a central role in our analysis. For any tour T, let $\ell(T)$ be the length of the tour. Denote by $T^*(N)$ a minimum length tour through N. If $T^\alpha(N)$ is a tour of the nodes N found by an α-approximation algorithm for the TSP, then $\ell(T^\alpha(N)) \leq \alpha \ell(T^*(N))$. The best known α-approximation algorithm for the TSP is Christofides' algorithm with $\alpha = 1.5$ (see [8]).

Let $\mathcal{P}_k$ denote an instance of the CTSPPD defined with vehicle capacity $k \geq 1$. Let OPT_k be the length of the optimal solution to $\mathcal{P}_k$. Let H_k be the length of the solution provided by a heuristic H_k on $\mathcal{P}_k$. The following will also be of interest in our analysis. Let M_2 be the arcs of a minimum weight (length) *bipartite* matching where the bipartition is (N^+, N^-). Each arc of M_2 is a blue-to-red arc. Among all minimum weight (length) *general* matchings on the nodes of N, let M_{all} be the arcs of a matching that contains the maximum number of arcs in common with M_2. Note that this solution may have arcs of any type (blue-to-red, blue-to-blue or red-to-red). Clearly $\ell(M_{all}) \leq \ell(M_2)$.

We now present several bounds on optimal solutions to various versions of the problem. In what follows, let $\lceil x \rceil$ denote the smallest integer greater than or equal to x.

[1] For example, in New York City vans holding at most 16 children are used (see [5]).

LEMMA 1: The following relations hold:

(1) $2\ell(M_{all}) \leq \ell(T^*(N)) \leq OPT_k, \quad \forall k \geq 1,$
(2) $\ell(M_{all}) + \ell(M_2) \leq OPT_2,$
(3) $OPT_j \leq \lceil k/j \rceil OPT_k, \quad \forall k \geq j \geq 1,$
(4) $2\ell(M_2)/k \leq OPT_k, \quad \forall k \geq 1.$

PROOF: (1) The optimal traveling salesman tour is clearly a lower bound on OPT_k for all $k \geq 1$. The optimal traveling salesman tour also defines two matchings on the nodes (by taking every other arc).

(2) Any feasible solution to $\mathcal{P}_2$ can be separated into a feasible solution to the bipartite matching problem and a feasible solution to the general matching problem. This is true since the loads on the vehicle over two consecutive arcs differ by exactly one unit and in any solution to $\mathcal{P}_2$ the loads cannot be greater than 2 or less than 0. Starting at an arc with load 1, it must be that every *other* arc, in either direction, has load 1. On the complement set of arcs the load is either 0 or 2. Each arc whose load is 0 or 2 must connect a red point and a blue point; thus this set of arcs is a bipartite matching between N^+ and N^-. The complement of this set is a general matching on N.

(3) Any solution to $\mathcal{P}_k$ can be traversed $\lceil k/j \rceil$ times with a vehicle of capacity j (for $1 \leq j \leq k$) creating a solution to $\mathcal{P}_j$. To see this, consider the solution to $\mathcal{P}_k$ and, starting from b_0, label each *blue* point encountered with the load of the vehicle immediately after serving the point. For example, points that are served when the vehicle is empty will have the label 1, points served when the vehicle has load 1 will be labeled 2, etc. Then in the first traversal of the solution to $\mathcal{P}_k$, pick up only blue points labeled $1, 2, \ldots, j$ and drop them off at the red points where they are dropped off at in the solution to $\mathcal{P}_k$. In the second traversal of the solution to $\mathcal{P}_k$, pick up only blue points labeled $j + 1, j + 2, \ldots, 2j$ and drop them off at the red points where they are dropped off at in the solution to $\mathcal{P}_k$. Continuing in this manner, the $\lceil k/j \rceil$ traversal will pick up only blue points labeled $(\lceil k/j \rceil - 1)j + 1, \ldots, k$ and drop them off similarly. It is clear that each of these traversals is feasible for a vehicle of capacity j.

(4) It is obvious that $2\ell(M_2) \leq OPT_1$ since any optimal solution to $\mathcal{P}_1$ can be separated into two feasible solutions to the bipartite matching problem. Also, from the previous bound, $OPT_1 \leq kOPT_k$.

The next lemma will be useful in the sequel.

LEMMA 2: Given m blue points and m red points located on a cycle and a vehicle of capacity $k \geq 1$, suppose that the vehicle can feasibly serve these points in a single clockwise (counterclockwise) traversal of the cycle starting from different blue points. Then,

(1) the load of the vehicle on each of the cycle's arcs on all feasible clockwise (counterclockwise) traversals of the cycle is independent of the starting point of the traversal, and
(2) there exists a feasible traversal for a vehicle of capacity k in the opposite direction.

PROOF: (a) Without loss of generality, we prove this for clockwise traversals. Take a feasible clockwise traversal of the cycle by a vehicle of capacity k. Let b_1 be its starting point. The last point of the traversal must be a red one, say r_1. The load of the vehicle on arc (r_1, b_1) is 0 and along the arcs of the cycle the load is between 0 and k. Suppose by contradiction that there exists a different feasible clockwise traversal of the cycle starting at a blue point $b_2 \neq b_1$ and this traversal associates loads with the cycle's arcs which are not identical to the ones of the first traversal. Since the load of the vehicle is solely determined by the difference between the

number of blue points and the number of red points visited so far, it must be that the second traversal assigns a load $x > 0$ to (r_1, b_1) and, moreover, that the load on any arc of the cycle in the second traversal is greater by x than the respective load in the first traversal. As a result, the vehicle never travels empty on the second traversal, even on the last arc of the traversal. This contradicts the assumption that the second traversal is feasible.

(b) Without loss of generality, suppose we have a feasible traversal in the clockwise direction. Start from the red point of the last arc of this feasible traversal and follow the counterclockwise direction. At each arc, record the cumulative load since the start. It is easy to see that the cumulative load on any arc in this traversal is minus the cumulative load on this arc in the clockwise traversal. Thus the arcs of this tour are associated with cumulative loads between $-k$ and 0 (inclusive). Find an arc with the minimum cumulative load. This arc must have one red and one blue end, call it (b^*, r^*). Therefore starting from b^* and traversing the cycle in the direction so that (b^*, r^*) is traversed last (which must be counterclockwise) is feasible for a vehicle of capacity k.

3. THE ITERATED TOUR MATCHING (ITM) ALGORITHM

We present here an algorithm whose worst-case error is bounded by 7. This is a refinement of a result of [7]. There, the authors present a 9.5-approximation algorithm.

We first present this heuristic in an informal manner. To simplify the presentation we assume, in this section, that n is divisible by k. This is without loss of generality, since if n is not divisible by k, it is always possible to add pairs of blue and red points located at the starting point (b_0) without increasing the length of any solution.

The heuristic starts by finding a minimum length tour $T^*(N^+)$ and a minimum length tour $T^*(N^-)$. Starting from the blue point b_0 [in $T^*(N^+)$], break up $T^*(N^+)$ into paths of exactly k nodes each by deleting the appropriate arcs. Starting from an arbitrary point of $T^*(N^-)$, break $T^*(N^-)$ into paths of exactly k nodes each by deleting the appropriate arcs. Each path of k nodes (called a k-path) is considered a *supernode.* Superimpose the matching M_2 onto this set of supernodes. Note that each supernode has degree exactly k in this graph; this property is called k-regularity. Also, there may be several arcs between two given super-nodes; thus it is a multi-graph.

In [7], the authors present a simple proof of the following property. Recall that a perfect matching is a matching where each node is matched to another.

LEMMA 3: The arcs of a k-regular bipartite multigraph can be partitioned into k perfect matchings.

If we choose one of these perfect matchings on the super-nodes, say $M_2^* \subseteq M_2$, then we can create a solution to the CTSPPD in the following manner. Starting from b_0, follow $T^*(N^+)$ in a clockwise direction picking up all k points in the k-path. Now go to the node in the k-path that is in M_2^* (this may require backtracking some steps) and follow the arc to a k-path of $T^*(N^-)$. Deliver all k points in this k-path and return through the same arc of M_2^*. Now continue following $T^*(N^+)$ in a clockwise direction until the next k-path and repeat this procedure. This is repeated until all points are served. An alternative tour is constructed by first skipping (not picking up) b_0 and traveling in a counterclockwise direction to the last k-path on the tour $T^*(N^+)$, and following $T^*(N^+)$ in the counterclockwise direction using the same rules as just performed. The better of the two solutions is kept as the solution.

In fact, the worst-case bound can be improved by iterating through k different starting points for $T^*(N^-)$ as well. This is what is done in the following.

The Iterated Tour Matching (ITM(α)) Algorithm:

Step 1: Using an α-approximation algorithm for the TSP, find a tour through N^+ [let $T^\alpha(N^+)$ be the tour], and find a tour through N^- [let $T^\alpha(N^-)$ be the tour].

Step 2: Starting at b_0, follow $T^\alpha(N^+)$ in a clockwise direction and break it into k-paths.

Step 3: Fix an orientation for $T^\alpha(N^-)$ and pick an arbitrary point of N^-, call it r_1, and label the points $r_1, r_2, \ldots, r_n$ in order of their appearance on $T^\alpha(N^-)$.

Step 4: For $j = 1, 2, \ldots, k$, do:

Step 4a: Starting from r_j, break $T^\alpha(N^-)$ into k-paths following the orientation of $T^\alpha(N^-)$.

Step 4b: Decompose the arcs of M_2 into k perfect matchings on the supernodes (the k-paths) of this graph. Select the cheapest of the k perfect matchings.

Step 4c: Construct a solution using these matching arcs by following $T^\alpha(N^+)$ clockwise from b_0. Construct another solution by starting from b_0, not picking up the load, and following $T^\alpha(N^+)$ in a counterclockwise direction while serving each of the k-paths encountered as described above (using the same matching arcs and the same k-paths as the solution of the clockwise direction). Keep track of the best solution found.

The worst-case bound is as follows.

THEOREM 4: Let $ITM_k(\alpha)$ designate the length of the solution provided by the Iterated Tour Matching Algorithm. Then, for each $k \geq 1$:

$$\frac{ITM_k(\alpha)}{OPT_k} \leq 1 + 2\alpha\left(2 - \frac{1}{k}\right).$$

PROOF: For each j in Step 4 of the ITM_k algorithm, two solutions are created, thus a total of $2k$ solutions are created. We calculate the total length of all $2k$ solutions and the best solution will have length less than or equal to the average. For this purpose, consider the following:

- For each j in Step 4, the two solutions created traverse each arc of $T^\alpha(N^+)$ at most 4 times [those arcs between the k-paths of $T^\alpha(N^+)$ are covered only twice]. Thus $T^\alpha(N^+)$ is covered a total of at most $4k$ times in the $2k$ solutions.
- Each arc of $T^\alpha(N^-)$ is covered $4(k-1)$ times in the $2k$ solutions.
- For each solution constructed, the length of the matching arcs chosen is at most $\ell(M_2)/k$ since there are k perfect matchings to choose from, their sum is $\ell(M_2)$, and the perfect matching with minimum length is selected each time; each of these arcs is covered twice in each solution.

The solution provided by the $ITM_k(\alpha)$ algorithm is at most the average of the $2k$ solutions, hence:

$$ITM_k(\alpha) \leq \frac{1}{2k}(4k\ell(T^\alpha(N^+)) + 4(k-1)\ell(T^\alpha(N^-))) + \frac{2\ell(M_2)}{k}$$

$$= 2\ell(T^\alpha(N^+)) + 2\left(1 - \frac{1}{k}\right)\ell(T^\alpha(N^-)) + \frac{2\ell(M_2)}{k}$$

$$\leq 2\alpha\ell(T^*(N^+)) + 2\left(1 - \frac{1}{k}\right)\alpha\ell(T^*(N^-)) + \frac{2\ell(M_2)}{k}$$

$$\leq 2\alpha\ell(T^*(N)) + 2\left(1 - \frac{1}{k}\right)\alpha\ell(T^*(N)) + \frac{2\ell(M_2)}{k}$$

$$\leq 2\alpha OPT_k + 2\left(1 - \frac{1}{k}\right)\alpha OPT_k + OPT_k,$$

where the last inequality follows from property four of Lemma 2.1.

When Christofides' heuristic is used to construct the initial traveling salesman tours (in Step 1) ($\alpha = 1.5$), the ITM algorithm is a $(7 - 3/k)$-approximation algorithm. In this case, the complexity of the algorithm is $O(n^3)$ since Step 1 (Christofides' heuristic) is $O(n^3)$ (see [11]) and solving the bipartite matching problem is also $O(n^3)$ (see [10]).

4. THE MATCH^2 ALGORITHM

We now present an algorithm called MATCH^2 for the special case where the vehicle capacity k is 2. The ideas here will be generalized in the next section to the case $k > 1$.

The MATCH^2 algorithm is based on superimposing the two matchings M_{all} and M_2. Let E represent the set of arcs defined by $E \equiv M_{all} \cup M_2$. Note E is a set of disjoint cycles, along with possibly some pairs of nodes, consisting of one blue and one red node, that are connected by two arcs (one from M_{all} and one from M_2). We call these *simple* cycles. We next study several structural properties of the set E.

LEMMA 5: Each cycle of E has zero total weight, i.e., the number of blue points equals the number of red points.

PROOF: Since the cycles of E are those formed by the sets M_{all} and M_2, every other arc is a blue-to-red arc.

LEMMA 6: A cycle of E is either a simple cycle or contains a blue-to-blue arc.

PROOF: A simple cycle of E must be made up of a blue and a red point hence cannot contain a blue-to-blue arc. If a cycle is not simple, then we show that it contains at least two consecutive blue points. Otherwise the cycle consists of the superposition of two different bipartite matchings (from blue points to red points) on the same set of nodes. If one of the two is cheaper than the other, it should appear in both the bipartite matching and the general matching. Therefore, the two must have the same length, but this contradicts the property assumed for M_{all} (that its intersection with M_2 was largest).

A blue point of a cycle is called a *feasible starting point* if it is possible to traverse the cycle (in at least one of two directions) starting with an empty vehicle from the point.

LEMMA 7: It is possible to feasibly traverse each cycle of E with a vehicle of capacity 2 in either direction. Moreover, each blue point in a nonsimple cycle is a feasible starting point for exactly one of the two directions.

PROOF: The lemma is clearly true for simple cycles. We now prove it for nonsimple cycles. By Lemma 6, any nonsimple cycle must contain an arc connecting two blue points, say (b_i, b_j). We will show that b_i and b_j are feasible starting points in opposite directions.

Let b_i be adjacent to b_j and the red point r_ℓ. Starting at b_i, it is clear that the last arc of the cycle must be (r_ℓ, b_i) in order for the vehicle to terminate the cycle empty. Select the direction according to which (r_ℓ, b_i) is the last arc of the cycle and suppose it is the clockwise direction. We will show now that starting at b_i and following the clockwise direction results in a feasible traversal. According to this traversal of the cycle the load on the first arc of the cycle $(b_i, b_j) \in M_{all}$ is 1. Also, every traversal of any other arc of the cycle in M_{all} will be with a vehicle of load 1 as well (since the change in load is zero between the traversals of arcs of M_{all}). Arcs of M_2 must be traversed with a vehicle of load either 0 or 2. Therefore, the above clockwise direction is feasible.

Starting at b_j, it is clear that the first arc to be traversed is $(b_j, b_i) \in M_{all}$ which, by following the same arguments as above, induces a counterclockwise feasible direction. This proves the first part of the lemma.

It remains to prove that any blue point on a nonsimple cycle which is adjacent to two red points is a feasible starting point in exactly one direction. Let b_i be adjacent to two red points r_ℓ and r_m in the cycle. Suppose without loss of generality that the arc $(r_\ell, b_i) \in M_{all}$. The load of the vehicle on (r_ℓ, b_i) is 1; therefore, starting at b_i and following the direction according to which (r_ℓ, b_i) is the last arc of the cycle is infeasible, the vehicle will be nonempty at the end of the cycle. The other direction must be feasible for the same reasons as above.

A consequence of the above proof is the following corollary.

COROLLARY 8: (a) In any feasible traversal of a cycle of E the load of the vehicle on arcs of M_{all} is 1 and on the arcs of M_2 is either 0 or 2.

(b) If the load on a given arc $(r, b) \in M_2$ of a nonsimple cycle of E is 0 (2) in all feasible clockwise traversals of the cycle, then

- the load is 2 (0) in all the feasible counterclockwise traversals of the cycle; and
- b (r) follows r (b) when the cycle is traversed in a clockwise direction.

PROOF: Part (a) follows directly from the proof of Lemma 7. Regarding (b), suppose that the load on an arc $(b, r) \in M_2$ is 0 in any feasible clockwise traversal of the cycle (see Lemma 2 for this assumption). According to Lemma 7, b is a feasible starting point for a clockwise traversal where (r, b) is the last arc traversed; therefore, the load on (r, b) is 0, and b immediately follows r (in the clockwise direction). Thus, in any feasible counterclockwise traversal b is visited before r; therefore, the load on (b, r) cannot be 0, and, since $(b, r) \in M_2$, the load must be 2. The proof is analogous for the case where the load on (b, r) is 2 in any feasible clockwise traversal.

We now present the MATCH^2 Algorithm.

The MATCH$^2(\alpha)$ Algorithm:

Step 1: Find a minimum weight (length) bipartite matching where the bipartition is (N^+, N^-), and let M_2 be the arcs of the matching.

Step 2: Find a minimum weight (length) general matching on the set N which contains the maximum number of arcs of M_2. Let the arcs of this matching be M_{all}. Let $E \equiv$

$\cup_{j\geq 0}\{C_j\}$, where C_j are the cycles (including simple cycles) of $M_2 \cup M_{all}$, and C_0 is the cycle containing b_0 (recall that b_0 is the designated starting point).

Step 3: For each cycle C_j ($j \geq 1$) of E, let b_j be an arbitrary blue point of C_j. Let $V \equiv \cup_{j\geq 0}\{b_j\}$. Using an α-approximation algorithm, find a tour $T^{\alpha}(V)$ through the set of points V.

Step 4: Starting at b_0, traverse C_0 in a feasible direction, and, after returning to b_0, follow $T^{\alpha}(V)$. When the vehicle encounters a point b_j ($j \geq 1$) on $T^{\alpha}(V)$, traverse that cycle (C_j) in a feasible direction. Upon completion of the cycle, continue along $T^{\alpha}(V)$ visiting each cycle until b_0 is reached.

It should now be clear that:

THEOREM 9: The MATCH^2 Algorithm constructs a feasible solution to $\mathcal{P}_2$.

The worst-case bound is given by the following theorem.

THEOREM 10: Let $MATCH^2(\alpha)$ be the length of the solution generated by the MATCH^2 Algorithm. Then,

$$\frac{MATCH^2(\alpha)}{OPT_2} \leq 1 + \alpha.$$

PROOF: Note that $\ell(T^{\alpha}(V)) \leq \alpha\ell(T^*(V)) \leq \alpha\ell(T^*(N))$. Then

$$\begin{aligned} MATCH^2(\alpha) &\leq \ell(M_{all}) + \ell(M_2) + \ell(T^{\alpha}(V)) \\ &\leq \ell(M_{all}) + \ell(M_2) + \alpha\ell(T^*(N)) \\ &\leq OPT_2 + \alpha OPT_2, \end{aligned}$$

where the last inequality uses properties one and two of Lemma 1.

Note that if the Christofides' heuristic is used for the TSP of Step 3, then the MATCH^2 Algorithm provides a 2.5-approximation. The complexity of the algorithm is $O(n^3)$, since it is dominated by the time taken to find the general matching and to find the traveling salesman tour, which are both $O(n^3)$ (see [10] and [11], respectively).

5. THE MATCH^k ALGORITHM

We now generalize this idea to the case $k > 2$. The algorithm is based on recursively applying the ideas from the MATCH^2 algorithm. The MATCH^k algorithm is as follows:

The MATCH$^k(\alpha)$ Algorithm:

Step 1: Let $m \equiv \lfloor \log_2 k \rfloor$.

Step 2: Find a minimum weight (length) bipartite matching on the bipartition (N^+, N^-). Let G_1 be the set of arcs in the matching and color them *green.* Note $G_1 = M_2$.

Step 3: Find a minimum weight (length) general matching on the set N which contains the maximum number of arcs of G_1. Let these arcs be A_1. Note $A_1 = M_{all}$. Let $E_1 \equiv G_1 \cup A_1$.

Step 4: Let $I = 1$ and $i = 1$. While $i \leq m - 1$ do begin:

Step 4a: Let t be the number of disjoint cycles making up E_i.

Step 4b: For each cycle C_j, $1 \leq j \leq t$, let B_j^+ be the green arcs of C_j that are oriented with the blue end immediately clockwise from the red end. Let B_j^- be the green arcs of C_j oriented in the opposite way. Let B_j be the set, either B_j^+ or B_j^-, with the *largest* total length (where ℓ is the measure used). That is, if $\ell(B_j^-) \leq \ell(B_j^+)$, then let $B_j = B_j^+$; else let $B_j = B_j^-$.

Step 4c: Let $B_i = \cup_{j=1}^{t} B_j$. When the green arcs B^i are removed from E_i, the set E_i is a set of paths, and if $i > 1$, also possibly a number of cycles with no green arcs. Consider the paths only and find an optimal general matching on the endpoints of the paths which has the maximum intersection with B^i. Let A_{i+1} be the arcs of this matching. Set $E_{i+1} \leftarrow (E_i \sim B^i) \cup A_{i+1}$ and $G_{i+1} \leftarrow G_i \sim B^i$. The edges of $B^i \cap A_{i+1}$ are no longer considered *green,* i.e., the only green arcs remaining are those of G_{i+1}. If $G_{i+1} = \emptyset$ (no green arcs are left) or $i = m - 1$, set $I = i + 1$ and go to Step 5. Otherwise, set $i = i + 1$.

Step 5: For each cycle of E_I, say C_j ($j \geq 1$), let b_j be a feasible starting point of C_j for a vehicle of capacity $2^I \leq k$. (If b_0 is a feasible starting point for its cycle, select it as the starting point for the cycle.) Let $V \equiv \cup_{j \geq 0} \{b_j\}$. Using an α-approximation algorithm, find a tour $T^\alpha(V)$ through the set of points V.

Step 6: Starting at b_0, follow $T^\alpha(V)$. When the vehicle encounters a point b_j ($j \geq 1$), traverse the cycle C_j in a feasible direction. Upon completion of the cycle, continue along $T^\alpha(V)$ visiting each cycle until b_0 is reached. If b_0 is a feasible starting point for its cycle then serve this cycle in the feasible direction until b_0 is reached.

We now prove that the above algorithm provides a feasible solution to $\mathcal{P}_k$.

DEFINITION 11: Given a cycle of E_i, for $i = 1, 2, \ldots, I$, with at least one green arc (b, r), define a *canonical traversal* to be one that starts from the blue point b and traverses the cycle in the direction so that (b, r) is traversed last.

LEMMA 12: Each cycle of E_i, for $i = 1, 2, \ldots, I$, has the following properties:

(a) Its total weight is 0.

(b) If the cycle has a green arc, then all canonical traversals are feasible for a vehicle of capacity 2^i. In addition, in any canonical traversal the load on a green arc is either 0 or 2^i (depending on its orientation[2]). Specifically, if the canonical traversal starts at b and ends at r, then all green arcs arcs with the same orientation as (r, b) are traversed with an empty vehicle. All other green arcs (with the opposite orientation), if such exist, are traversed with a load of 2^i.

(c) If the cycle has no green arcs, then there exists a feasible traversal for a vehicle of capacity 2^i.

[2] Here "orientation" is meant in the sense that the red end point is immediately clockwise from the blue end point, or vice versa.

PROOF: We prove this by induction on i. For $i = 1$, first note that Lemma 6 implies that all nonsimple cycles of E_1 have green arcs with both orientations. The claim then follows by Lemmas 5 and 7 and Corollary 8.

Suppose by induction that the claim holds for E_i, $1 \leq i \leq h < I$ and we prove it for E_{h+1}. Recall that in the set of cycles E_h, a subset of the green arcs (B^h) is removed and a new set of arcs (A_{h+1}) is added to form E_{h+1}. We concentrate for a moment on the graph $E_h \sim B^h$. The graph $E_h \sim B^h$ consists of a set of paths and possibly some cycles with no green arcs. We show the following:

1. Each path has a total weight of 0.
2. Each path has one blue and one red end point.
3. If there is a green arc on a path, it must be oriented in such a way that the arc's blue end is between the arc's red end and the path's blue end.
4. A vehicle of capacity 2^h can feasibly serve the path starting empty (full) from its blue (red) end.
5. If there is a green arc on a path, the cumulative load, starting empty, from the path's blue (red) end point to the blue (red) end point, inclusive, of each green arc of the path is exactly 2^h (-2^h).

Property 1 follows from part (b) of the inductive assumption on E_h. That is, if all green arcs in one orientation are traversed with the same load, then removing them results in paths with zero total weight. Property 2 follows from the definition of B^h. Property 3 holds since all green arcs having the opposite orientation were removed (they are in B^h). Given a path P in $E_h \sim B^h$ whose blue end point is b and its red end point is r, then the path is part of a cycle in E_h in which $(\tilde{r}, b)$ and $(r, \tilde{b})$ are green arcs for some nodes $\tilde{r}$ and $\tilde{b}$. (We note that $\tilde{r} = r$ and $\tilde{b} = b$ is possible.) The green arcs $(\tilde{r}, b)$ and $(r, \tilde{b})$ have the same orientation in E_h since both arcs are members of B^h. Consider a canonical traversal starting at b [such that $(\tilde{r}, b)$ is traversed last]; then the traversal of P is feasible for a vehicle of capacity 2^h [it follows from the inductive assumption part (b) on E_h]. Reversing the argument, it is clear that the path P can be feasibly traversed starting at r with a full vehicle (a load of 2^h). This proves 4. If P contains green arcs, then these green arcs must have the opposite orientation to $(\tilde{r}, b)$ and $(r, \tilde{b})$, and thus, according to part (b) of the inductive assumption on E_h, starting empty at b, the load on these green arcs is exactly 2^h. Reversing the argument, it is clear that starting at r with a full vehicle (a load of 2^h), the green arcs will be traversed with an empty vehicle. This proves 5.

We now prove (a)–(c) for the cycles of E_{h+1}: The cycles of E_{h+1} are of several types. There may be cycles in E_{h+1} with no green arcs that were also cycles in E_h. Induction with (a) and (c) confirms that the total weight of these cycles is 0 and that there exists a feasible traversal for a vehicle of capacity $2^h < 2^{h+1}$. The other cycles of E_{h+1} (those that were not cycles in E_h) are formed by matching the end points of paths in $E_h \sim B^h$ each having total weight 0. Let C be such a cycle of E_{h+1}, and let A be the arcs of C that are in A_{h+1}. The cycle C is made up by combining paths each of weight 0, and therefore C has total weight 0 which completes the proof of (a). We distinguish between the case where C contains green arcs and the case where C does not contain any green arcs.

To prove (b), consider a cycle C formed by combining paths with green arcs and possibly some paths with no green arcs. Pick a green arc (b, r) of C and let P_1 be the path of $(E_h \sim B^h) \cap C$ that contains (b, r). Now consider a canonical traversal starting at b. By 5, the first arc of A encountered is traversed with a load of exactly 2^h, and, by 1, each arc of A is traversed with

a load of exactly 2^h. We now show that the traversal is feasible for a vehicle of capacity 2^{h+1}. Consider the traversal of path P_i. If the first node of P_i is blue (red), then, by 4, the load on the vehicle is always at most $2^h + 2^h = 2^{h+1}$ (at least $2^h - 2^h = 0$). Upon completing the path, the load is again 2^h (on the arc of A). Thus the canonical traversal is feasible for a vehicle of capacity 2^{h+1}. We now consider the load on the green arcs of C. If P_i has a green arc in an opposite orientation to (b, r), then, by 3, the traversal of P_i starts at its blue end point; thus, by 5, the green arc will be traversed with a load of exactly $2^h + 2^h = 2^{h+1}$. If P_i has a green arc with the same orientation as (b, r), then, by 3, the traversal of P_i starts at its red end point, and thus, by 5, the green arc will be traversed with a load of exactly $2^h - 2^h = 0$. This proves (b) for cycles of E_{h+1} that contain green arcs.

To prove (c), consider a cycle C with no green arcs (that was not also a cycle of E_h). In the cycle, we need only find a feasible traversal for a vehicle of capacity 2^{h+1}. Starting from an arc (b, r) in A, follow the cycle in the direction such that b is served first and the arc (r, b) is the last traversed. On each arc record the total cumulative load since the moment prior to b. By 1, each arc of A in this cycle will be traversed with a cumulative load of zero. By 4, while traversing each path P_i the cumulative load always stays between -2^h and 2^h. After completing this (not necessarily feasible) traversal, note the arc that had the smallest cumulative load. This arc must have one red and one blue end, call it (b^*, r^*). Therefore, starting from b^* and traversing the cycle in the direction so that (b^*, r^*) is the last traversed is feasible for a vehicle of capacity 2^{h+1}. This proves (c).

As a direct consequence, we obtain the following theorem:

THEOREM 13: The MATCH^k Algorithm constructs a feasible solution for $\mathcal{P}_{(2^m)}$, for $m = \lfloor \log_2 k \rfloor$, and thus for $\mathcal{P}_k$.

Note that when the algorithm terminates with $I < m$ this signifies that MATCH^k terminates with a solution that does not use all the capacity of the vehicle and indeed the maximum load of the vehicle in the resulting solution is 2^I, where $2^I < 2^m \leq k < 2^{m+1}$.

The complexity of the algorithm is dominated by Step 4. Each iteration of Step 4 takes $O(n^3)$ since it is dominated by the time to perform the general matching of Step 4c. Step 4 is performed $\lfloor \log_2 k \rfloor$ times and thus the overall complexity of the MATCH^k Algorithm is $O(n^3 \cdot \log_2 k)$. Since we can assume $k \leq n$, the complexity is $O(n^3 \log n)$.

The next lemma is helpful in the derivation of the worst-case bound for MATCH^k.

LEMMA 14: In the application of MATCH^k, for any i, $1 \leq i \leq I$,

$$\ell(B^i) \geq \tfrac{1}{2}\ell(G_i).$$

PROOF: For each set E_i in Step 4b, by definition we have $B_j^+ \cap B_j^- = \emptyset$ for all j, and thus $G_i = \cup_{j=1}^t [B_j^+ \cup B_j^-]$. At the end of Step 4b, B_j is selected as either B_j^+ or B_j^- according to which has the largest length. Therefore, for each i and j,

$$\ell(B_j) \geq \tfrac{1}{2}\ell(G_i \cap C_j).$$

Since this is done for each j, $1 \leq j \leq t$, we obtain

$$\ell(B^i) = \ell(\cup_{j=1}^t B_j) \geq \sum_{j=1}^t \tfrac{1}{2}\ell(G_i \cap C_j) = \tfrac{1}{2}\ell(G_i).$$

Table 1. Worst-case bounds for MATCH^k and ITM when $\alpha = 1.5$.

k	2	3	4	5	6	7	8	10	16	32
MATCH^k (1.5)	2.5	3.5	3.25	3.75	3.75	4.25	3.875	4.125	4.438	4.969
ITM (1.5)	5.5	6.0	6.25	6.4	6.5	6.571	6.625	6.7	6.813	6.906

In the next theorem, we provide a worst-case bound for the MATCH^k Algorithm. Before presenting the theorem, Table 1 below lists the worst-case bounds for the MATCH^k and ITM algorithms for the case where Christofides' heuristic is used to find a traveling salesman tour ($\alpha = 1.5$):

THEOREM 15: Let $MATCH^k(\alpha)$ be the length of the solution generated by the MATCH^k Algorithm. Then, for each $k \geq 2$,

$$\frac{MATCH^k(\alpha)}{OPT_k} \leq \alpha + \frac{\lfloor \log_2 k \rfloor}{2} + \frac{2\lfloor k/2 \rfloor - 1}{2^{\lceil \log_2 k \rceil}}. \tag{1}$$

PROOF: Note G_I is the set of green arcs present in the solution at the end of the algorithm. This is the total length of the arcs of the matching M_2 (performed in Step 2) remaining in the solution at the completion of the algorithm. Then

$$MATCH^k(\alpha) \leq \ell(G_I) + \sum_{i=1}^{I} \ell(A_i) + \ell(T^\alpha(V)). \tag{2}$$

Recall $m \equiv \lfloor \log_2 k \rfloor$.

CASE 1: $I < m$. Then $G_I = \emptyset$. Since A_i is a set of arcs from a general matching on some subset of N, we have, for all $i \geq 1$:

$$\ell(A_i) \leq \tfrac{1}{2}\ell(T^*(N)) \leq \tfrac{1}{2}OPT_k. \tag{3}$$

Combining (2) and (3), we get

$$MATCH^k(\alpha) \leq \ell(G_I) + \sum_{i=1}^{I} \ell(A_i) + \ell(T^\alpha(V))$$

$$\leq OPT_k(\tfrac{I}{2} + \alpha)$$

$$\leq OPT_k\left(\frac{m-1}{2} + \alpha\right).$$

In this case, the bound is better than the one of (1).

CASE 2: $I = m$. Define

$$\beta_k = \left\lceil \frac{k}{2} \right\rceil - \frac{1}{2}.$$

Recall (from Lemma 1, property 4):

$$\ell(M_2) \leq \tfrac{k}{2} OPT_k.$$

We consider two subcases.

CASE 2a: $\ell(M_2) \leq \beta_k OPT_k$. Observe that

$$\ell(G_m) \leq \frac{\ell(M_2)}{2^{m-1}}.$$

This is true since in each round of Step 4 at least half of the total length of the remaining green arcs is removed from the solution. Thus, from (2),

$$\begin{aligned} MATCH^k(\alpha) &\leq \ell(G_m) + \sum_{i=1}^{m} \ell(A_i) + \ell(T^\alpha(V)) \\ &\leq \ell(M_2)/2^{m-1} + m\ell(T^*(N))/2 + \alpha\ell(T^*(N)) \\ &\leq \beta_k OPT_k/2^{m-1} + mOPT_k/2 + \alpha OPT_k. \end{aligned}$$

Hence

$$\frac{MATCH^k(\alpha)}{OPT_k} \leq \frac{\beta_k}{2^{m-1}} + \frac{m}{2} + \alpha,$$

and with some algebra we obtain (1).

CASE 2b: $\beta_k OPT_k < \ell(M_2) \leq k/2\ OPT_k$. Note this implies that k is even. Using Lemma 1 (properties 2 and 3):

$$\ell(M_{all}) + \ell(M_2) \leq OPT_2 \leq \left\lceil \tfrac{k}{2} \right\rceil OPT_k = \tfrac{k}{2} OPT_k.$$

Using the fact that the largest half of the green arcs (from M_2), i.e., B^1, is at least $\beta_k OPT_k/2$, and this, along with at least half of the remaining green arcs, i.e., $B^1 \cup B^2$, is at least $3\beta_k OPT_k/4$, etc.,

$$\ell(M_{all}) + \ell(G_m) \leq \left[\frac{k}{2} - \beta_k \left(1 - \frac{1}{2^{m-1}} \right) \right] OPT_k.$$

Thus the bound becomes:

Table 2. Worst-case bounds for MATCH^k on MCVRP when $\alpha = 1.5$.

k	2	3	4	5	6	7	8	10	16	32
MATCH^k (1.5)	1.0	2.0	1.75	2.25	2.25	2.75	2.375	2.625	2.938	3.469

$$MATCH^k(\alpha) \le \ell(G_m) + \sum_{i=1}^{m} \ell(A_i) + \ell(T^\alpha(V))$$

$$= \ell(G_m) + \ell(A_1) + \sum_{i=2}^{m} \ell(A_i) + \ell(T^\alpha(V))$$

$$= \ell(G_m) + \ell(M_{all}) + \sum_{i=2}^{m} \ell(A_i) + \ell(T^\alpha(V))$$

$$\le \left[\frac{k}{2} - \beta_k\left(1 - \frac{1}{2^{m-1}}\right)\right] OPT_k + (m-1)\frac{\ell(T^*(N))}{2}$$

By using $k/2 = \beta_k + 1/2$ (since k is even), we get

$$\frac{MATCH^k(\alpha)}{OPT_k} \le \frac{\beta_k}{2^{m-1}} + \frac{m}{2} + \alpha,$$

and we obtain (1).

6. CONCLUSION

We remark here that the MATCH^k algorithm also provides an improved worst-case bound for a special case of the CVRP discussed in the introduction. For the CVRP, the best approximation algorithm is the Iterated Tour Partitioning (ITP) Heuristic of Altinkemer and Gavish [1]. Their worst-case bound is $1 + \alpha(1 - 1/k)$, where k is the vehicle capacity and α is, again, the worst-case bound of the TSP heuristic used. For the CVRP, the worst-case bound of the MATCH^k Algorithm is reduced by α since there is no need to connect all cycles (of E_m) with a traveling salesman tour (all cycles include the depot). For this problem, the MATCH^k Algorithm strictly improves on the best worst-case bound for the case where $k = 4$. Our bound is 1.75, the ITP ($\alpha = 1.5$) heuristic provides a 2.125 worst-case bound. Incidentally, for the case of $k = 2$, the MATCH^k Algorithm provides the optimal solution to the CVRP (see [11]).

In the case of the MCVRP with limited supply (discussed in the Introduction), the worst-case bound of the MATCH^k Algorithm is reduced by α under the standard assumption that there is a vehicle at each depot and each vehicle must start and end its route at its depot (called the *fixed-destination case* in [12]). The bounds are then given in Table 2. If this assumption cannot be made, then the worst-case bounds are as in Sections 4 and 5. In either case, the MATCH^k Algorithm provides the best worst-case bound for the problem with *limited* supply.

ACKNOWLEDGMENTS

We would like to thank Moses Charikar of Stanford University for pointing out an error in an earlier version of this paper. This research was supported by an internal grant from the Columbia Graduate School of Business and NSF Career Award DMI-97-02596.

REFERENCES

[1] K. Altinkemer and B. Gavish, Heuristics for delivery problems with constant error guarantees, Transportation Sci 24 (1990), 294–297.
[2] S. Anily and R. Hassin, The Swapping Problem, Networks 22 (1992), 419–433.
[3] S. Anily and G. Mosheiov, The Traveling Salesman Problem with delivery and backhauls, Oper Res Lett 16 (1994), 11–18.
[4] E.M. Arkin, R. Hassin, and L. Klein, Restricted delivery problems on a network, Networks 29 (1997), 205–216.
[5] J. Braca, J. Bramel, B. Posner, and D. Simchi-Levi, A computerized approach to the New York City School Bus Routing Problem, IIE Trans 29 (1994), 693–702.
[6] D.O. Casco, B.L. Golden, and E.A. Wasil, "Vehicle routing with backhauls: Models, algorithms, and case studies," Vehicle routing: Methods and studies, B.L. Golden and A.A. Assad (Editors), North-Holland, Amsterdam, 1988, 127–147.
[7] P. Chalasani and R. Motwani, Approximating capacitated routing and delivery problems, Working Paper, Department of Computer Science, Stanford University, Palo Alto, CA, 1995.
[8] N. Christofides, Worst-case analysis of a new heuristic for the Traveling Salesman Problem, Report 388, Graduate School of Industrial Administration, Carnegie-Mellon University, Pittsburgh, PA, 1976.
[9] M. Haimovich and A.H.G. Rinnooy Kan, Bounds and heuristics for capacitated routing problems, Math Oper Res 10 (1985), 527–542.
[10] E.L. Lawler, Combinatorial optimization: Networks and matroids. Holt, Rinehart and Winston, New York, 1976.
[11] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, and D.B. Shmoys (Editors), The Traveling Salesman Problem: A guided tour of combinatorial optimization, Wiley, New York, 1985.
[12] C.L. Li and D. Simchi-Levi, Worst-case analysis of heuristics for the multi-depot capacitated vehicle routing problems, ORSA J Comput 2 (1990), 64–73.