Approximation Algorithms for k-Delivery TSP

PRASAD CHALASANI* RAJEEV MOTWANI†

August 1, 1995

Abstract

We provide $O(1)$ approximation algorithms for the following NP-hard problem called *k-Delivery TSP*: We have at our disposal a truck of capacity k, and there are n depots and n customers at various locations in some metric space, and exactly one item (all of which are identical) at each depot. We want to find an optimal tour using the truck to deliver one item to each customer. Our algorithms run in time polynomial in both n and k. The 1-Delivery problem is one of finding an optimal tour that alternately visits depots and customers. For this case we use matroid intersection to show a polynomial-time 2-approximation algorithm, improving upon a factor 2.5 algorithm of Anily and Hassin [1]. Using this approximation combined with certain lower bounding arguments we show a factor 11.5 approximation to the optimal k-Delivery tour. For the infinite k case we show a factor 2 approximation.

1 Introduction

Consider the following k-Delivery TSP. There are n depots and n customers at various locations in some metric space. There are n identical items (say refrigerators), exactly one at each depot, and one must be delivered to each customer. We have a truck of capacity k at our disposal and our problem is to compute an optimal route to do these deliveries (starting at one of the depots, say). Note that some of the depots/customers can be at the same location, so that our algorithm also works for any collection of depots and customers with integral supplies and demands respectively (and the algorithms would run in time polynomial in the total supply/demand).

This problem is easily seen to be NP-hard by reducing from the TSP: set up a depot and customer at each point of the TSP problem, and set $k = 1$; now an optimal solution to this 1-Delivery problem will be an optimal solution to the TSP.

Other TSP variants that constrain the order of visiting the vertices have been considered in the OR literature, although not from an approximation viewpoint: (a) the Dial-a-Ride problem [8, 7, 9]: compute an optimal route for a k-capacity van to pick-up and drop off n persons between different origin-destination pairs, and (b) the Precedence-Constrained TSP [7]: For each vertex i there is a set $P(i)$ of vertices that must be visited before visiting i, and we are required to find an optimal tour that satisfies these constraints.

In the rest of the paper we will refer to the depots as blue points and the customers as red points.

*Los Alamos National Laboratory, New Mexico. E-mail: `chal@lanl.gov`.

†Department of Computer Science, Stanford University. E-mail: `rajeev@cs.stanford.edu`. Supported by an Alfred P. Sloan Research Fellowship, an IBM Faculty Development Award, an OTL grant, and NSF Young Investigator Award CCR-9357849, with matching funds from IBM, Schlumberger Foundation, Shell Foundation, and Xerox Corporation.

2 Unit Capacity Truck: 1-Delivery TSP

When the truck has unit capacity, any delivery route must alternately pick up and deliver one item at a time. The corresponding graph problem is the following.

> **Bipartite Traveling Salesman Problem.** Given an edge-weighted graph G satisfying the triangle inequality, with n blue vertices (depots) and n red vertices (customers), find the optimal *bipartite* tour starting and ending at a designated blue vertex s and visiting all vertices. A tour is bipartite if no two consecutively visited vertices have the same color.

Anily and Hassin [1] have shown a 2.5-approximation algorithm for a generalization of this problem known as the Swapping Problem. Their algorithm finds a perfect matching M consisting of edges that connect red and blue vertices, and uses Christofides' heuristic [3] to find a tour T of the blue vertices. The final delivery route consists of visiting the blue vertices in the sequence specified by the tour T, using the matching edges in M to deliver an item to a customer and return to the blue vertex (or "short-cut" to the next blue vertex on T). If OPT is the optimal delivery tour, clearly $T \leq 1.5\text{OPT}$ and $M \leq 0.5\text{OPT}$, whereas the total length of the delivery tour is at most $T + 2M \leq 2.5\text{OPT}$. We exploit some combinatorial properties of bipartite spanning trees and matroid intersection to improve this factor to 2.

2.1 Matching Trees

One naive approach to achieve a factor 2 approximation that does not work is to mimic the well-known factor-2 approximation algorithm for the TSP problem: pick a *bipartite* spanning tree of G, perform a depth-first traversal followed by short-cutting. A spanning tree of G is bipartite if each edge connects a red and blue vertex. Given a bipartite spanning tree T, we can think of it as a tree rooted at s, and do a depth-first traversal of T with short-cuts (there may in general be several ways to short-cut) and obtain a tour of G. However, such a tour may not be bipartite; there are bipartite spanning trees that do not yield a bipartite tour regardless of how we do the depth-first traversal and short-cuts. Fig. 1 shows an example of such a tree. When does a depth-first traversal (with short-cuts) of a bipartite spanning tree yield a bipartite tour? Theorem 1 shows a sufficient condition (and Fig. 2 shows that the condition is not necessary).

Theorem 1 *If a bipartite spanning tree T (rooted at s) of G contains a perfect matching of G, then there is a depth first traversal of T that visits red and blue vertices alternately (and this DFT therefore yields a bipartite tour of G).*

Proof: Consider T as a tree rooted at the designated vertex s. For any vertex v of T, let $T(v)$ denote the subtree of T rooted at v. Let $\mathbf{r}(v)$ and $\mathbf{b}(v)$ denote the number of red and blue vertices (including v) in $T(v)$ (see Fig. 3). For any set of vertices S, a vertex v is a *neighbor* of S if there is an edge of T connecting v to some vertex in S.

Suppose WLOG that v is red. Clearly the number of neighbors of the red vertices in $T(v)$ equals $\mathbf{b}(v)$ if v is the root of T and equals $\mathbf{b}(v) + 1$ otherwise. By Hall's theorem, since T contains a perfect matching, $\mathbf{b}(v) \geq \mathbf{r}(v) - 1$ must hold. On the other hand the number of neighbors of the blue vertices in $T(v)$ is exactly $\mathbf{r}(v)$, and by Hall's theorem we must have $\mathbf{r}(v) \geq \mathbf{b}(v)$. Thus for any vertex v, either $\mathbf{r}(v) = \mathbf{b}(v)$ (we will say v is *even* in this case) or the number of the vertices of $T(v)$ having the color of v

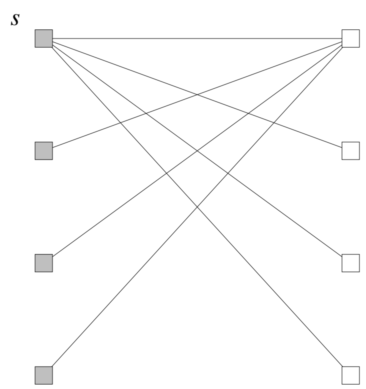

Figure 1: A bipartite spanning tree for which no depth-first traversal yields a bipartitie tour.

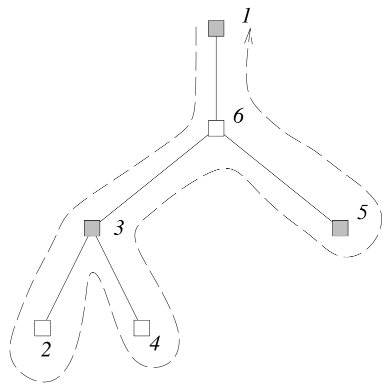

Figure 2: A bipartite spanning tree that does not contain a perfect matching and yet there is a depth-first traversal that can be short-cut to yield a bipartite tour.

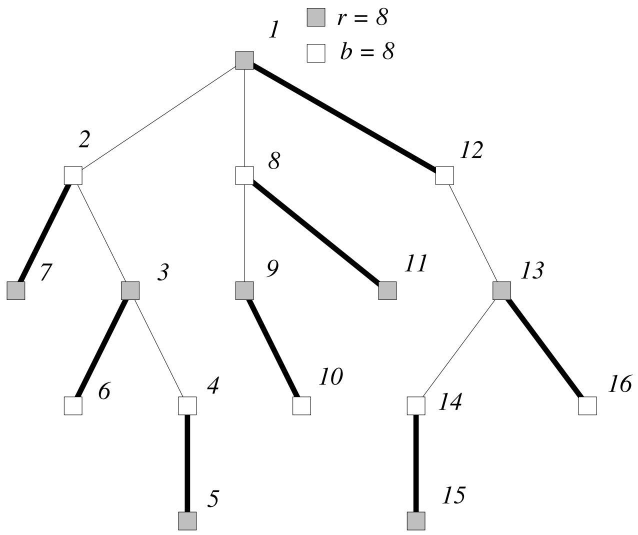

Figure 3: A bipartite spanning tree that contains a perfect matching, with the matching edges shown in dark. The values $\mathbf{r}(v)$ and $\mathbf{b}(v)$ of the root node are shown. The vertices are numbered in the depth-first order specified above;this is an alternating-color ordering of the vertices.

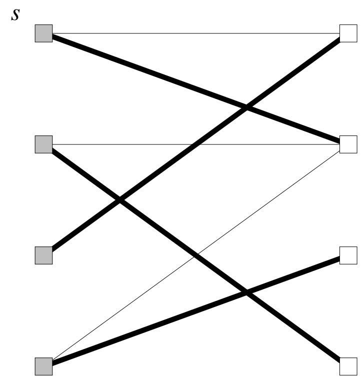

Figure 4: A bipartite spanning tree where the blue (shaded) vertices have degree at most 2. The edges of the perfect matching are shown in dark.

outnumber the other vertices by exactly 1 (we will say v is *odd*). Clearly if v is even then exactly one child of v is odd, and if v is odd, then all of its children are even.

Consider the DFT of T specified by the following rule:

> **Depth First Traversal Rule.** After visiting a vertex v, traverse the subtrees rooted at the even children (if any) before visiting the odd child (if any).

We claim that this DFT visits the vertices of T in alternating-color order. The proof is by induction on the number of vertices visited. The claim is trivially true for the very first vertex s (the root) visited. For any vertex v other than s, assume inductively that the vertices visited before v were visited in alternating color-order. Suppose WLOG that v is red. We must show that the last vertex visited before visiting v is blue. Let the parent of v be u. Since even children are visited before the (at most one) odd child, the set S of descendants of u visited before v contains an equal number of red and blue vertices (this number could be zero). Since u is blue, and by induction hypothesis the vertices in $S \cup \{u\}$ were visited in alternating-color order, the last vertex of $S \cup \{u\}$ visited must have been blue. ■

We refer to a bipartite spanning tree that contains a perfect matching as a *matching tree*. Clearly, the optimal bipartite tour OPT contains a matching tree, so the weight of a *minimum-weight* matching tree is a lower bound on the length of OPT. Theorem 1 implies that a depth-first traversal (with short-cuts) of any matching tree yields a bipartite tour, and by the triangle inequality, the length of this tour is at most twice the length of the tree. Thus, *if* one could find the minimum-weight matching tree T in polynomial time, a depth-first traversal (with short-cuts) of T would yield a tour whose length is at most $2.OPT$. However, finding the minimum-weight matching tree is NP-hard!

The trick is to use the following sufficient (but not necessary, see Fig. 3) condition for a bipartite spanning tree to contain a perfect matching (see Fig. 4).

Lemma 2 *Any bipartite spanning tree T whose blue vertices have degree at most 2 must contain a perfect matching. (That is a bipartite spanning tree where the blue nodes have degree at most 2 must be a matching tree)*

Proof: The spanning tree T must have $2n - 1$ edges, and since blue vertices have degree at most 2, exactly one blue vertex has degree 1, and all other blue vertices have degree 2. Consider any subset S of blue nodes, and let $|S| = m$. There are at least $2m - 1$ edges going out of S, since at most one of the vertices of S has degree 1. Let $N(S)$ denote the set of neighbors of S, and suppose $|N(S)| < m$. Thus $|N(S) \cup S| < 2m$ and the number of tree edges connecting vertices of $N(S) \cup S$ is at least $2m - 1$, which means there is a cycle. This is impossible since T is a spanning tree. Thus any subset S of blue vertices has at least $|S|$ red neighbors. Hall's theorem then implies that T contains a perfect matching. ■

Clearly, the OPT bipartite tour contains a bipartite spanning tree where all blue vertices have degree at most 2. Therefore the weight of the minimum-weight bipartite spanning tree whose blue vertices have degree at most 2, is a lower bound on OPT. Again, if we can find (in polynomial time) the minimum-weight bipartite spanning tree T whose blue vertices have degree at most 2, then a depth first traversal of T with short-cuts will yield a tour whose length is at most 2.0 times OPT.

2.2 Matroid Intersection

It turns out that the problem of finding T can be viewed as that of finding the minimum-weight, maximum-cardinality subset in the *intersection* of two matroids, and polynomial-time algorithms exist for this problem [6, 4, 5]. The two matroids in this case are: M_1, the matroid of all bipartite forests, and M_2, the matroid of all bipartite subgraphs whose blue vertices have degree at most 2.

For completeness we review here the definition of a matroid, following the standard text [6].

Definition. A *matroid* $M = (E, \mathcal{I})$ is a structure in which E a finite set of elements and $\mathcal{I}$ is a family of subsets (called *independent sets*) of E, such that

- $\phi \in \mathcal{I}$ and all proper subsets of a set $I \in \mathcal{I}$ are in $\mathcal{I}$.
- If I_p and I_{p+1} are sets in $\mathcal{I}$ containing p and $p+1$ elements respectively, then there exists an element $e \in I_{p+1}$ such that $I_p \cup \{e\} \in \mathcal{I}$

An example of a matroid is the *graphic matroid* $M = (E, \mathcal{I})$ where E is the set of edges of an undirected graph, and a subset $I \subset E$ is in $\mathcal{I}$ if and only if I is cycle-free. Another example of a matroid is the *matrix matroid* $M = (C, \mathcal{I})$ where C is the set of columns of a fixed matrix A and a subset S of columns is in $\mathcal{I}$ if and only if the columns of S are linearly independent. A maximal-cardinality independent subset of a matroid is called a *base* of a matroid; all bases of a matroid have the same cardinality.

Returning to our problem, let E be the set of all edges that connect red vertices to blue vertices. Let $\mathcal{F}$ denote the collection of all subsets of E that are cycle-free, and let $\mathcal{D}$ denote the collection of subsets S of E such that no more than two edges of S are incident on any blue vertex. Then it is easily seen that $M_1 = (E, \mathcal{F})$ and $M_2 = (E, \mathcal{D})$ are matroids. In addition, the problem of finding

> a minimum-weight bipartite spanning tree where the blue vertices have degree at most two,

is equivalent to the problem of finding

> a minimum-weight common base of M_1 and M_2.

This is a special case of the *matroid intersection* problem, which was first solved in polynomial time by Edmonds [4, 5]. Other authors [2] have exploited the special structure of problems such as ours to improve the running times.

Thus we have our main theorem:

Theorem 3 *There is a polynomial time 2-approximate algorithm for the Bipartite Traveling Salesman problem*

3 Finite Capacity Truck: k-Delivery TSP

3.1 Lower bounds

We now derive some useful lower bounds. Let C_k denote the (length of the) optimal k-Delivery tour, and let C_r and C_b denote the (length of the) optimal tours on the red and blue points respectively.

Lemma 4 $\frac{1}{2k}C_1 \leq C_k$.

Proof: Start with an optimal k-Delivery tour C_k: this defines an ordering $r_1, r_2, \ldots, r_n$ on the red points and an ordering $b_1, b_2, \ldots, b_n$ on the blue points. We then construct a 1-Delivery tour T starting at the blue vertex b_1 as follows: Consider the blue vertices in the order imposed by C_k, connecting the ith blue vertex b_i to the earliest red vertex in the C_k-ordering that has not already been connected to a blue vertex; then add another edge connecting this red vertex to the next blue vertex b_{i+1} (if this red vertex is the last one, connect it to the starting blue vertex b_1). By the triangle inequality, each edge e of T is no longer than the sum of the C_k-edges connecting the endpoints of e; we can thus "charge off" each edge of T to a collection of edges of C_k. Since there is never a sequence of more than k consecutive red or consecutive blue points in the tour C_k, it follows that no edge of T is charged more than $2k$ times. Thus $T \leq 2kC_k$, from which the lemma follows since $C_1 \leq T$. ∎

The following lower bounds are easy to see:

Lemma 5 $C_r \leq C_k$ *and* $C_b \leq C_k$.

3.2 An $O(1)$ approximation

We now use the two lower bounds just presented to design a constant-factor approximation algorithm for the k-Delivery problem. Assume for simplicity of exposition that n is a multiple of k. We first use Christofides' heuristic to obtain a 1.5-approximate tour T_r of the red vertices and a 1.5-approximate tour T_b of the blue vertices. Next we (arbitrarily) break up T_r and T_b into paths of k vertices each, by deleting edges appropriately. It will be convenient to view each k-path as a "super-node" in the following. We now overlay our 2-approximate 1-Delivery tour T_1 (from the previous section) on this graph. Note that any (red or blue) super-node now has degree exactly $2k$, and that there may be several edges between two given super-nodes. So what we have now is a 2k-regular bipartite multi-graph. The following lemma is crucial:

Lemma 6 *The edges of a d-regular bipartite multi-graph can be partitioned into d perfect matchings.*

Proof: Firstly note that the d-regularity implies there are an equal number of vertices on each side of the bipartition. Consider any subset S of the vertices on the left side of the bipartite graph and say $|S| = m$. Clearly the number of edges emanating from S is dm and by the Pigeonhole Principle if the set S has fewer than m neighbors on the right side then some vertex on the right side has degree greater than d, which is a contradiction. Thus any subset S of the left-vertices has at least $|S|$ neighbors.

Clearly this remains true even if replace each multi-edge by one of the edges, so by Hall's theorem it follows that the multi-graph contains a perfect matching. If we delete this perfect matching, we are left with a $(d-1)$-regular bipartite multi-graph, and the same argument can be repeated, each time removing a perfect matching. This proves the lemma. ■

We can thus partition T_1 into $2k$ perfect matchings on the super-nodes. We pick the least-weight matching M out of these and delete all other edges of T_1. Clearly $M \leq \frac{1}{2k}T_1$. At this stage we have a collection of n/k subgraphs $H_1, H_2, \ldots, H_{n/k}$, each consisting of a red supernode connected via an edge of M to a blue supernode. Now we re-introduce the edges of T_b that were removed when breaking T_b into k-paths; this imposes a cyclic ordering on the subgraphs H_i; let us relabel them $H_1, H_2, \ldots, H_{n/k}$ with this cyclic ordering, where H_1 contains the start blue vertex. We now traverse the subgraphs $H_1, H_2, \ldots, H_{n/k}$ in sequence as follows. Within each subgraph H_i first visit all the blue vertices, then use the edge of M to go to the red side and visit all the red vertices, and then return to the blue side, and go to the blue vertex that is connected via an edge e of T_b to the next subgraph H_{i+1}, and use the edge e to go to H_{i+1} (or H_1 if $i = n/k$). We claim that this tour T is within a constant factor of the optimal k-Delivery tour:

Theorem 7 *The above algorithm produces a tour T whose length is within a factor of 11.5 of the optimal k-Delivery tour C_k.*

Proof: Notice that in short-cutting, by triangle inequality, we charge each edge of T_b no more than 3 times, each edge of T_r no more than 2 times and each edge of M at most 2 times. So

$$\begin{aligned} T &\leq 3T_b + 2T_r + 2M \\ &\leq 3 \times 1.5C_b \ + \ 2 \times 1.5C_r + \frac{1}{k}T_1 \\ &\leq 7.5C_k \ + \ \frac{2.0}{k}C_1 \\ &\leq 7.5C_k \ + \ \frac{2.0}{k}2kC_k \\ &\leq 11.5C_k \end{aligned}$$

■

4 Infinite Capacity Truck

When the truck has infinite capacity, the only restriction is that at any stage during the delivery, the number of items picked up so far be at least as many as the number of customers visited so far. The corresponding graph problem is this:

> **Blue-Dominant TSP.** Given an edge-weighted graph G satisfying the triangle inequality, with n blue vertices (depots) and n red vertices find an optimal tour that starts at a blue vertex s and visits all the vertices (and returns to s) in *blue-dominant* order. An ordering of vertices is blue-dominant if in each prefix of the ordering, there are at least as many blue vertices as red vertices.

We show below a 2-approximation algorithm for this problem. It turns out that one can obtain a 2-approximation for this problem by first finding a minimum-weight spanning tree (not necessarily bipartite) and then using a *specific* depth first traversal and short-cutting scheme. As Fig 2 shows, for a fixed depth-first traversal (DFT) there may be several ways to perform short-cuts. Let us make precise the distinction between a DFT and a specific short-cutting scheme. Note that we can view a depth-first traversal as traversing edges rather than as visiting vertices. In the edge-traversing viewpoint, each edge is traversed exactly twice: first downward (away from the root) and later upward (toward the root). Thus each vertex may be "visited" several times by a DFT. The short-cutting scheme *marks* exactly one out of these several visits to a vertex. For a fixed DFT and short-cutting scheme, the marking order defines a tour of all the vertices. We will consider two types of short-cutting schemes in particular: when the DFT reaches a subtree rooted at v, we may mark v either *before* or *after* visiting all its descendants. In the former case we say v is *pre-marked*, and in the latter case we say v is *post-marked*.

Theorem 8 *For any spanning tree T rooted at the blue vertex s, there is a depth-firs t traversal and a short-cutting scheme that marks the vertices in blue-dominant order.*

Proof: For any vertex v, let $\mathbf{r}(v)$ and $\mathbf{b}(v)$ be defined as before. For a marking sequence defined by a DFT and a short-cutting scheme, for any vertex v let $R(v)$ denote the number of red vertices marked just before the DFT *first* reaches v, and let $B(v)$ denote the

number of blue vertices marked just before the DFT *first* reaches v.

Consider a depth-first traversal (DFT) of T satisfying the following:

> **Depth First Traversal Rule.** After visiting a vertex v, if there are any children w of v such that $\mathbf{b}(w) \geq \mathbf{r}(w)$ then visit them (and their descendants) before visiting any remaining children.

The short-cutting scheme is:

> **Short-Cutting Rule.** Pre-mark blue-vertices (depots) and post-mark red vertices.

Figure 5 illustrates a use of these rules.

We would like to show that $B(v) \geq R(v)$ holds at all times. We will in fact claim that the following invariant holds whenever the DFT first reaches a vertex v:

$$B(v) \geq R(v), \quad \text{and}$$

$$B(v) + \mathbf{b}(v) \geq R(v) + \mathbf{r}(v), \quad \text{or equivalently} \quad B(v) - R(v) \geq \mathbf{r}(v) - \mathbf{b}(v).$$

The proof of the invariant is by structural induction on the rooted tree T. It is trivially true at the start vertex s since $B(s) = R(s) = 0$ and $\mathbf{b}(s) = \mathbf{r}(s) = n$. Assume inductively that the invariant holds just before the DFT reaches some vertex v. We argue that it will hold at each child of v, visited say in the order $w_1, w_2, \ldots, w_k$. For any vertex v, let $c(v)$ denote the quantity $\mathbf{r}(v) - \mathbf{b}(v)$, the "surplus of customers", and let $D(v)$ denote $B(v) - R(v)$, the "depot surplus". Our inductive assumption $D(v) \geq c(v)$ implies, if v is blue,

$$D(v) \geq \sum_{i=1}^{k} c(w_i) - 1,$$

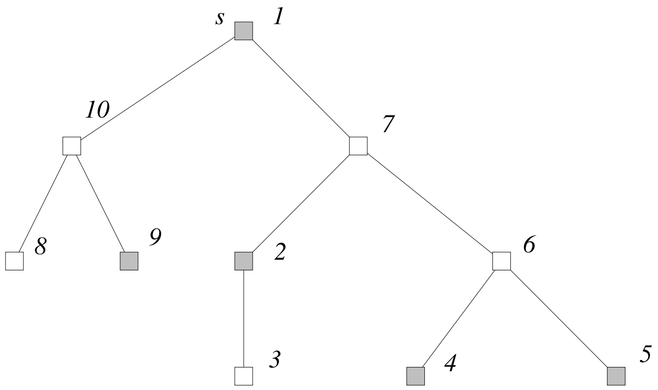

Figure 5: When the vertices of this spanning tree are marked in depth-first order using the short-cutting scheme from above, we get a valid delivery tour for an infinite-capacity truck. The depots are shown shaded.

and if v is red,

$$D(v) \geq \sum_{i=1}^{k} c(w_i) + 1.$$

If v is blue, it is marked before the DFT visits w_1, so $D(w_1) = D(v) + 1$. If v is red, it is not marked before visiting w_1, so $D(w_1) = D(v)$. In either case we have $D(w_1) \geq 0$ and

$$D(w_1) \geq \sum_{i=1}^{k} c(w_i).$$

Note that for each $i = 1, 2, \ldots, k-1$,

$$D(w_{i+1}) = D(w_i) - c(w_i) \geq c(w_{i+1}) + c(w_{i+2}) + \ldots + c(w_k).$$

Our DFT rule specifies that children with $c(w_i) \leq 0$ must be visited first, and the above equation implies that the $D(.)$ value does not decrease after traversing the subtree rooted at such a child and therefore remains positive. So our invariant holds at each such child. After the subtree under the last such child (say w_j) has been traversed, we have $D(w_{j+1}) \geq 0$ and

$$D(w_{j+1}) \geq c(w_{j+1}) + c(w_{j+2}) + \ldots + c(w_k),$$

where each term on the right hand side is positive. It is now easy to see that $D(w_i) \geq 0$ and $D(w_i) \geq c(w_i)$ will hold for each $i = j+1, \ldots, k$. ∎

Acknowledgement

We are grateful to Alan Frieze for suggesting the problem and his encouragement.

References

[1] S. Anily and R. Hassin. The swapping problem. *Networks*, 22:419–433, 1992.

[2] C. Brezovec, G. Cornuejols, and F. Glover. A matroid algorithm and its application to the efficient solution of two optimization problems on graphs. *Math. Programming*, 42:471–487, 1988.

[3] N. Christofides. Worst-case analysis of a new heuristic for the traveling salesman problem. Technical Report 62-75-76, Graduate Sch. of Ind. Adm., Carnegie Mellon University, 1976.

[4] J. Edmonds. Submodular functions, matroids and certain polyhedra. In *Combinatorial Structures and their Applications, Proc. Calgary Int'l Conf.*, pages 69–87, 1970.

[5] J. Edmonds. Matroid intersection. *Annals of Discrete Math.*, 4:39–49, 1979.

[6] E. Lawler. *Combinatorial Optimization: Networks and Matroids.* Holt, Reinhart and Winston, New York, 1976.

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

[8] M. Kubo and H. Kagusai. Heuristic algorithms for the single-vehicle dial-a-ride problem. *J. O.R. Society of Japan*, 30(4):354–365, Dec. 1990.

[9] H. Psaraftis. Scheduling large-scale advance-request dial-a-ride systems. *Amer. J. Math. Manage. Sci.*, 6(3-4):327–367, 1986.