1987

Efficient Solutions to Some Transpsortation Problems with Applications to Minimizing Robot Arm Travel

Mikhail J. Atallah
Purdue University, mja@cs.purdue.edu

S. Rao Kosaraju

Report Number:
87-666

EFFICIENT SOLUTIONS TO SOME TRANSPORTATION PROBLEMS WITH APPLICATIONS TO MINIMIZING ROBOT ARM TRAVEL

Mikhail J. Atallah
S. Rao Kosaraju

CSD-TR-666
March 1987
Revised October 1987

EFFICIENT SOLUTIONS TO SOME TRANSPORTATION PROBLEMS WITH APPLICATIONS TO MINIMIZING ROBOT ARM TRAVEL

Mikhail J. Atallah[1]
Dept. of Computer Science
Purdue University
West Lafayette, IN 47907

S. Rao Kosaraju[2]
Dept. of Computer Science
Johns Hopkins University
Baltimore, MD 21218

Abstract. We give efficient solutions to transportation problems motivated by the following robotics problem. A robot arm has the task of rearranging m objects between n stations in the plane. Each object is initially at one of these n stations and needs to be moved to another station. The robot arm consists of a single link that rotates about a fixed pivot. The link can extend in and out (like a telescope) so that its length is a variable. At the end of this "telescoping" link lies a gripper that is capable of grasping any one of the m given objects (the gripper cannot be holding more than one object at the same time). The robot arm must transport each of the m objects to its destination and come back to where it started. Since the problem of scheduling the motion of the gripper so as to minimize the total distance traveled is NP-hard, we focus on the problem of minimizing only the total angular motion (rotation of the link about the pivot), or only the telescoping motion. We give algorithms for two different modes of operation: (i) *no-drops*: no object can be dropped before its destination is reached, (ii) *with-drops*: any object can be dropped at any number of intermediate points. Our algorithm for case (i) runs in $O(m+n\log n)$ time for angular motion, in $O(m+n\,\alpha(n))$ time for telescoping motion. Our algorithm for case (ii) runs in in $O(m+n)$ time for angular motion, and with the same time bound for telescoping motion. The most interesting problem turns out to be that of minimizing angular motion for the with-drops mode of operation.

1 This research was supported by the Office of Naval Research under Grants N00014-84-K-0502 and N00014-86-K-0689, and the National Science Foundation under Grant DCR-8451393, with matching funds from AT&T.

2 This research was supported by the National Science Foundation under Grant DCR-856361.

1. Introduction

A robot arm has the task of rearranging m objects between n stations in the plane. Each object is initially at one of these stations and needs to be moved to another station (its destination). The robot arm consists of a single link that rotates about a fixed pivot (see Figure 1). The link can extend in and out (like a telescope) so that its length is a variable. At the end of this "telescoping" link lies a gripper that is capable of grasping any one of the m given objects.

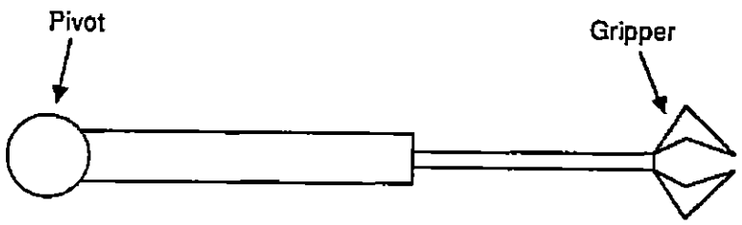


Figure 1. The robot arm can pivot, and can extend like a telescope

The gripper can pick up an object and drop it at another station, then, can move to another station and can continue with the transfers. Many objects can simultaneously be located at the same station, but the gripper cannot be holding more than one object at a time. When the gripper is empty and is at a station, it is free to pick up any of the objects at that station. We also require that the gripper must terminate at the station where it started. Scheduling the motion of the gripper so as to minimize the total distance it travels can be shown to be NP-hard from the NP-hardness of the Euclidean Traveling Salesperson problem [P2]. Here we focus on the problem of minimizing only the total angular motion (rotation of the link about the pivot), or only the total telescoping motion.

For the case of minimizing angular motion we henceforth assume, without loss of generality, that

(a) the n stations are positioned on a circular track centered at the pivot, and

(b) the motion of the gripper is always along the circumference of this circular track.

The problem is then to minimize the total length of the circular arcs traversed by the gripper. The input specification is by listing the destinations of the objects at each station on the circular track. The stations, in clockwise cyclic order, are denoted by the integers 1 to n, and one of them is designated as being the initial position of the gripper (we call it the *start* station). The input therefore describes a directed multigraph having n vertices and m edges (we draw a directed arc for each object -- the head and the tail corresponding to the destination and the source stations, respectively). Figure 2 illustrates a 4 station 4 object transfer problem.

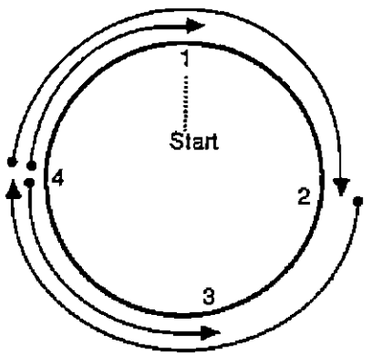


Figure 2. A transfer problem

The problem of minimizing the total telescoping motion rather than the angular one, can be viewed as a linear track problem rather than as a circular one. The circular track case is considerably more difficult than the linear track one.

We develop fast algorithms for two different modes of operation: (i) *no-drops*: once an object is picked up by the gripper, it cannot be dropped before its destination is reached, (ii) *with-drops*: any object can be dropped at any number of intermediate points. The algorithm for the no-drops case runs in $O(m+n\log n)$ time for a circular track (i.e. minimizing angular motion), in $O(m+n\,\alpha(n))$ for a linear track (i.e. minimizing telescoping motion); here $\alpha(n)$ is the extremely slowly growing functional inverse of Ackermann's function. The no-drops problem can be cast as a graph augmentation problem [ET,P1] -- augmentation of a graph to an eulerian graph. The with-drops problem is

more interesting and does not seem to translate into a natural graphical problem. Somewhat surprisingly, we are able to design a faster algorithm for this problem -- an $O(m+n)$ time algorithm (for either circular or linear track). One of the difficulties in the with-drops problem for a circular track is that an optimal transportation may have to transport an object through the longer of the two circular arcs between its source and destination (such an arc is henceforth called *major*, the other arc being *minor*). In Section 4 we give an example for which any optimal transportation must transport an object through the major arc. However, we prove that in a with-drops problem, an optimal transportation transports at most one object through the major arc. This nontrivial result is only one of the ingredients in our linear time solution to this problem; another ingredient is a method for quickly identifying which of the m objects (if any) should be transported through the major arc.

Throughout the paper, all graphs are actually multigraphs (i.e. can have many edges with same head and tail). A graph is directed unless we explicitly state that it is undirected. All the graphs we refer to are *embedded* on the (circular or linear) track, i.e. their vertices are the stations on the track and their edges are directed arcs drawn along the track. Therefore when we henceforth refer to an edge e of a graph G, we are really talking about a particular drawing of that edge (for a circular track, the edge can be drawn two ways). We use $|e|$ to denote the length of the portion of the track covered by e.

For a circular track we assume, without loss of generality, that the circle's circumference equals unity. The *complement* of an edge e is the edge e^c with the same source as e, same destination as e, and such that e and e^c together cover the complete circumference (see Figure 3). Note that $(e^c)^c=e$, and that $|e|+|e^c|=1$. An edge e is *major* iff $|e|>1/2$, and is *minor* otherwise (in Figure 3, e is minor and e^c is major). Note that if $|e|=1/2$ then both e and e^c are minor. *Shortening* a major edge means

replacing it with its complement.

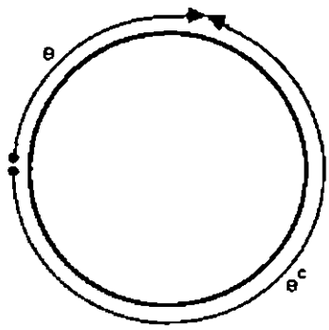


Figure 3. An arc and its complement

We adopt the convention that, when depicting a transportation, we draw an input source-to-destination pair as a directed (circular or linear) arc coinciding with the actual path that this transportation uses to take the object to its destination (in the with-drops case, the object transported along such an arc may be dropped many times on the way to its destination).

We assume that none of the n stations is useless, i.e. each is the source or destination of at least one object (useless stations are easily eliminated with an $O(m+n)$ preprocessing step). This implies that $n \leq 2m$.

2. No-drops problem

In this section we prove the following result.

Theorem 1. An optimal transportation for any no-drops problem can be calculated in $O(m+n\,\alpha(n))$ for a linear track, $O(m+n\log n)$ time for a circular track.

The rest of this section proves the above theorem.

First, observe that in a circular no-drops problem we never need to take an object to its destination using the major arc, and therefore we always draw the input edges so that they are minor.

The no-drops problem is a graph augmentation problem: we want to add edges to the input graph so as to make it eulerian [E], such that the total lengths of the added edges is minimum. Any euler tour of the resulting eulerian graph then gives an optimal transportation. These added edges are called *augmenting edges*, and correspond to motion of the gripper when it is not holding any object. In future drawings, we distinguish such augmenting edges by drawing their arrowhead dashed, whereas that of an input edge is drawn solid.

Since the minimum eulerian augmentation does not depend on the start vertex, the length of an optimal transportation does not depend on which vertex is the start (and hence finish) vertex. (In the with-drops case, considered in Section 3, the start vertex is significant.)

Recall that a graph G is eulerian if and only if (i) every vertex of G has its in-degree equal to its out-degree (we call this the *degree-balance* property), and (ii) the undirected version of G is connected. Condition (ii) can be replaced by "G is strongly connected", because if (i) holds then G is strongly connected if and only if its undirected version is connected [E]. In the rest of this paper we restrict the augmenting edges to be of the form $(i,i+1)$ or $(i+1,i)$, i.e. each augmenting edge covers only one of the n intervals (gaps) between adjacent stations. There is no loss of generality in doing so, since an augmenting edge that covers l intervals can always be broken into l smaller edges without increasing the total edge length, without disturbing degree balance, and without damaging undirected connectivity. Of course if there are many such augmenting edges covering an interval $(i,i+1)$ then we do not store each of them individually since this might take a total of $O(mn)$ space; instead, we store a count of the number of such edges going in each direction across that interval. Thus the total storage needed for augmenting edges is $O(n)$.

Observe that in any optimal augmentation, if any pair of antiparallel edges $(i,i+1)$

and $(i+1,i)$ are augmenting edges, then in between i and $i+1$ there cannot be any other augmenting edge (otherwise removal of $(i,i+1)$ and $(i+1,i)$ preserves the degree-balance and undirected connectivity, contradicting the optimality of the original augmentation).

1.1. Linear track

We first prove the linear track part of Theorem 1, an example of which is given in Figure 4a, where $n=8$ and $m=5$. We make a few trivial observations.

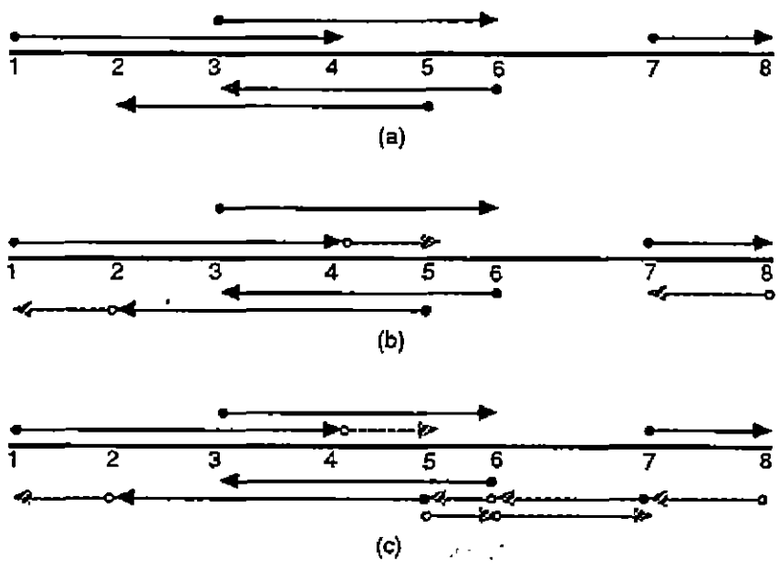


Figure 4. A linear track problem (a), its degree-balanced augmentation (b), and the optimal augmentation (c)

Observation 1. In any transportation, at any point, x, in between the leftmost and the rightmost stations, the number of times the gripper moves left to right across x is the same as the number of times the gripper moves right to left across x. In addition each of these crossings is ≥ 1.

Based on this observation, we add across each interval the smallest number of augmenting edges that will make the total number of edges that cross that interval from left to right equal to the number of edges that cross it from right to left. The "augmenting edges" needed for Figure 4a are shown in Figure 4b. When the graph is augmented in this manner, every vertex will have the degree-balance property. Let this augmentation

process be denoted as the *degree-balanced augmentation.*

Observation 2. If the resulting degree-balanced augmented graph is strongly connected, then it has an euler tour and hence it represents an optimal transportation.

However the augmented graph need not be strongly connected (sc). For example, Figure 4b has three strongly connected components (scc's): {1,4,5,2}, {3,6}, {7,8}. If the augmented graph is not sc then its scc's are disjoint in the sense that there is no edge between any two of them (because for a graph having the degree-balance property, the scc's are the connected components of the undirected version of the graph).

Now the problem reduces to adding more augmenting edges, with minimum total length, to make the graph sc without disturbing its degree-balance property. An example of this augmentation is shown in Figure 4c. In general, augmentation of a q-scc degree-balanced graph can be achieved by including $q-1$ antiparallel pairs of augmenting edges (we needed two such pairs to go from Figure 4b to 4c: one between 5 and 6, the other between 6 and 7). To find the $q-1$ antiparallel pairs needed to minimally make the degree-balanced graph sc, we create a q-vertex edge-weighted undirected graph, one vertex for each scc. In that undirected graph, an edge between i and j is present if and only if a station x in the ith scc is adjacent to a station y in the jth scc (i.e. $|x-y|=1$). The weight of this edge $\{i,j\}$ is the distance between stations x and y. If there are many such pairs x,y for a particular $\{i,j\}$, then the weight of $\{i,j\}$ is the minimum over all such pairs x,y.

Observation 3. The minimum total length augmenting pairs needed to make the degree-balanced graph sc correspond to the undirected edges of a minimum spanning tree (MST) of the above-mentioned q-vertex undirected graph of scc's. (Of course, we have to map each undirected edge $\{i,j\}$ of the MST into one antiparallel pair of augmenting edges: $(x,y),(y,x)$ in which x and y are the stations in scc's i and j, respectively, which contributed to the weight of the $\{i,j\}$ edge.)

The above discussion implies an $O(m+n\alpha(n))$ time algorithm for computing the minimum eulerian augmentation in the linear track version of the problem, using the MST algorithm of [FT]. Any euler tour of the resulting eulerian graph gives an optimal transportation. Such an euler tour can easily be found in an additional $O(m)$ time (exercise 5.9 in [AHU]).

Thus the overall time for the linear track case is $O(m+n\alpha(n))$. We now complete the proof of Theorem 1 by considering the circular track case.

1.2. Circular track problem

If we know that there exists an optimal transportation in which at least one interval is not covered by any augmenting edge, then we can solve n separate straight line problems: the ith one assuming that there is no augmenting edge in between stations i and $i+1$ (assume that station $n+1$ = station 1). Then the transportation corresponding to the minimum of these n solutions gives the optimal transportation. However, it is not hard to come up with an example in which any optimal solution must have augmenting edges covering the (complete) circumference.

The circular track equivalent of Observation 1 does not hold, i.e. for the circular track it is no longer true that at every point the number of clockwise crossings of the gripper is the same as the number of counterclockwise crossings. However, if we define the *flux* across an interval to be the number of clockwise crossings minus the number of counterclockwise crossings (counting both the input edges and the augmenting ones), then we have the following.

Lemma 1. For any augmentation, the degree-balance property is satisfied if and only if the flux is the same across all intervals.

Proof. It suffices to show that degree balance holds if and only if, for any i, the flux across interval $(i-1,i)$ is the same as that across interval $(i,i+1)$. The difference between these two fluxes equals the difference between the in-degree of i and its out-degree. □

The flux across an interval is the sum of two components. One component is the *augmenting* flux across that interval: the number of clockwise augmenting edges across that interval minus the number of counterclockwise augmenting edges across it. The other component is the *input* flux across that interval and is the number of clockwise input edges across it minus the number of counterclockwise input edges across it. Let $\phi(i)$ denote the input flux across the interval $(i,i+1)$. In Figure 2, $\phi(1)=\phi(2)=1,\phi(3)=0,\phi(4)=2$. Note that $\phi(i)$ is the number of counterclockwise augmenting edges that must be added to interval $(i,i+1)$ in order to make its total flux equal to zero (a negative value signifies adding clockwise edges).

The next two lemmas impose constraints on the augmenting edges and possible flux values that an optimal augmentation can have.

Lemma 2. There exists an optimal augmentation in which for some i the number of augmenting edges in between i and $i+1$ is no more than one.

Proof. Let an optimal augmentation result in at least two augmenting edges between every i and $i+1$. Among all such optimal augmentations, select one with fewest clockwise augmenting edges. Select any undirected circuit of n augmenting edges covering the circumference (ignoring the directions of these augmenting edges). We distinguish two cases.

Case 1. On this circuit, the total length of the clockwise edges is not equal to the total length of the counterclockwise edges. If it is larger (resp. smaller), then remove the clockwise (resp. counterclockwise) edges and duplicate the counterclockwise (resp. clockwise) edges one more time. This preserves undirected connectivity and also the degree-balance property. In addition, this transformation decreases the total length. This contradicts the optimality of the original augmentation.

Case 2. On this circuit, the total length of the clockwise edges is equal to the total length of the counterclockwise edges. Remove the clockwise edges and duplicate the counter-

clockwise edges one more time. This preserves undirected connectivity, the degree-balance property, and the total length. However it results in an optimal augmentation having fewer clockwise edges than the original one, a contradiction. □

Note that Lemma 1 implies that for every optimal transportation there exists a value such that the flux across every interval is that value. In addition, Lemma 2 implies that there are only $3n$ relevant values of flux worth considering, namely $\bigcup_{i=1}^{n}\{\phi(i)-1,\phi(i),\phi(i)+1\}$.

Lemma 3. There exists an optimal augmentation whose flux is between $-m-1$ and $m+1$.

Proof. Lemma 2 implies that there exists an optimal augmentation in which at least one interval has at most one augmenting edge across it. The absolute value of the flux of such an augmentation is no more than $1+\max_{1\le i\le n}|\phi(i)|\le 1+m$. □

It is easy to come up with examples in which there is a unique optimal augmentation and it has flux $\Theta(m)$. The range "$-m-1$ to $m+1$" of Lemma 3 can be narrowed to "$-m/2$ to $m/2$" but we avoid doing so for simplicity of exposition.

Observe that fixing the value of the flux (at, say, ψ) entirely determines the cost of the minimum augmentation achieving degree balance at that flux value, because every interval $(i,i+1)$ needs to add across it $\psi-\phi(i)$ clockwise augmenting edges in order for the flux across it to become ψ. The resulting graph, however, may not be sc, and additional pairs of antiparallel edges may need to be added in order to make it sc. For a given flux value, the antiparallel pairs needed to make the degree-balanced graph sc can be determined by a minimum-cost spanning tree computation similar to the one described for the linear track case. Our main problem is therefore that of determining which flux value ψ_0 is such that there is an optimal eulerian augmentation whose flux is ψ_0.

Let the *cost* of flux ψ be the total length of the minimum eulerian augmentation whose flux is constrained to be ψ. This cost consists of two components: (i) a degree-

balance component $db(\psi)$ equal to $\sum_{i=1}^{n} |\psi-\phi(i)|\, l_i$ where l_i is the length of the interval $(i,i+1)$, and (ii) a connectivity component $cc(\psi)$ which accounts for the length of the antiparallel pairs of augmenting edges needed to make sc the degree-balanced graph resulting from (i). (The $cc(\psi)$ results from the previously mentioned MST computation.)

If we knew $db(\psi)$ and $cc(\psi)$ values for all $-m-1 \le \psi \le m+1$, then the optimal ψ would be the one which minimizes $db(\psi)+cc(\psi)$. The next two lemmas show how to compute all the $db(\psi)$'s and $cc(\psi)$'s efficiently (the nontrivial part is computing the $cc(\psi)$'s).

Lemma 4. The $db(\psi)$'s $(-m-1 \le \psi \le m+1)$ can all be computed in $O(m)$ time.

Proof. It suffices to show that the description of the function $db(\cdot)$ can be computed in $O(m)$ time. Note that $db(\psi)=\sum_{i=1}^{n} |\psi-\phi(i)|\, l_i$ is piecewise linear and has at most n angular points (at $\psi=\phi(i)$, $1 \le i \le n$). It is easy to compute $\phi(1), \cdots, \phi(n)$ in $O(m)$ time. If we knew the slope of $db(\psi)$ at every value of ψ, $-m-1 \le \psi \le m+1$, then we could easily obtain all the $db(\psi)$'s with $O(m)$ additional work. The slope at $\psi=-m-1$ is equal to $-\sum_{i=1}^{n} l_i = -1$, and at $\psi=m+1$ it equals $\sum_{i=1}^{n} l_i = 1$; in between it changes only at values of ψ that belong to $\{\phi(1), \cdots, \phi(n)\}$. Therefore we sort $\{\phi(1), \cdots, \phi(n)\}$ in $O(m)$ time, and then we scan the resulting sorted sequence, updating the slope of $db(\cdot)$ as we go along. □

Lemma 5. The $cc(\psi)$'s $(-m-1 \le \psi \le m+1)$ can all be computed in $O(m+n \log n)$.

Proof. First observe that if $\psi \notin \bigcup_{i=1}^{n} \{\phi(i)\}$ then $cc(\psi)=0$ because in that case the degree-balanced graph of flux ψ is already sc (it has an augmenting edge across every one of the n intervals). We therefore need only concern ourselves with computing the $cc(\psi)$'s for all $\psi \in \bigcup_{i=1}^{n} \{\phi(i)\}$. By its definition, $cc(\psi)$ is equal to twice the cost of the MST of the

undirected graph $CC(\psi)$ whose vertices are the n stations, and whose edges consist of:

(i) the undirected versions of the input edges, and

(ii) one edge $\{i,i+1\}$ for each interval $(i,i+1)$.

The edges in (i) have zero cost in $CC(\psi)$, while an edge $\{i,i+1\}$ in (ii) has zero cost if $\psi \neq \phi(i)$ (because in that case the minimum degree-balanced augmentation for flux value ψ already places at least one augmenting edge across the interval $(i,i+1)$), and cost equal to the interval's length l_i if $\psi=\phi(i)$. Note that all $CC(\psi)$'s have the same set of edges, the only difference being in the weights of the n edges in (ii). Since the edges in (i) have zero cost in all $CC(\psi)$'s, we can "collapse" each connected component of the edges in (i) into a single vertex: let $v_1,\cdots,v_q$ be the vertices resulting from this collapsing operation; if the endpoints of an edge in (ii) collapse into a single v_i then the edge vanishes, otherwise it survives (of course its endpoints become the collapsed vertices rather than the original stations). This collapsing operation can easily be done in $O(m)$ time as a pre-processing step. Assume from now on that this has already been done, so that every $CC(\psi)$ is now a q-vertex multigraph having as edges the (at most n, at least q) edges in (ii) that survived the collapsing operation. Each of the (possibly many) edges between v_i and v_j corresponds to an interval between one of the stations that collapsed into v_i and one of the stations that collapsed into v_j. Of course $cc(\psi)$ is still twice the cost of the MST of the new (collapsed) $CC(\psi)$. Let CC be identical to the (collapsed) $CC(\psi)$, except that in CC the costs associated with the edges of $CC(\psi)$ are replaced by *labels*: the edge of CC that corresponds to interval $(i,i+1)$ is labeled by $\phi(i)$. Note that $CC(\psi)$ can be obtained from CC by assigning to each edge with label $\phi(i)$ a cost of zero if $\phi(i)\neq\psi$, a cost equal to the length of $(i,i+1)$ if $\phi(i)=\psi$. Let the intervals that correspond to edges of CC be denoted by $(i_1,i_1+1),\cdots,(i_r,i_r+1)$. Find the median of $\phi(i_1),\cdots,\phi(i_r)$ (call it ϕ_0), then partition the set $\{i_1,\cdots,i_r\}$ into A,B,C as follows: $A=\{i_j : \phi(i_j)<\phi_0\}$, $B=\{i_j : \phi(i_j)=\phi_0\}$, $C=\{i_j : \phi(i_j)>\phi_0\}$. (Note that each of A and C has

at most $r/2$ elements.) The important thing to notice is that, if $\psi \geq \phi_0$ (resp. $\neq \phi_0$, $\leq \phi_0$) and i belongs to A (resp. B, C), then in $CC(\psi)$ the edge corresponding to interval $(i, i+1)$ has zero cost. This suggests the following recursive procedure for computing the $cc(\psi)$'s for all $\psi \in \{\phi(i): i \in A \cup B \cup C\}$. First, create the undirected graph Q_A (resp. Q_B, Q_C) whose vertices are $v_1, \cdots, v_q$ and each of whose whose edges corresponds to an interval $(i, i+1)$ with $i \in A$ (resp. B, C). Let CC_A (resp. CC_B, CC_C) be obtained from CC by collapsing each connected component of Q_A (resp. Q_B, Q_C) into a single vertex. Note that CC_A (resp. CC_B, CC_C) has no more than $|A|$ (resp. $|B|, |C|$) edges, and no more vertices than it has edges. Recursively compute the $cc_A(\psi)$ values for all $\psi \in \{\phi(i): i \in A\}$.

Note. $cc_A(\psi)$ (resp. $cc_B(\psi), cc_C(\psi)$) denotes twice the cost of the MST of $CC_A(\psi)$ (resp. $CC_B(\psi), CC_C(\psi)$). Note that if $\psi = \phi(i)$ for some i in A (resp. B, C) then $cc(\psi)$ equals $cc_A(\psi)$ (resp. $cc_B(\psi)$, $cc_C(\psi)$).

Next, recursively compute the $cc_C(\psi)$ values for all $\psi \in \{\phi(i): i \in C\}$. Then find $CC_B(\phi_0)$, and compute its MST in $O(|B|\alpha(|B|))$ time [FT] ($cc_B(\phi_0)$ is twice the cost of this MST). If $T(r)$ denotes the overall time for this recursive procedure, then we have:

$$T(r) \leq T(|A|) + T(|C|) + c_1 r + c_2 |B| \alpha(|B|),$$

where $|A| \leq r/2$, $|C| \leq r/2$, and $|A| + |B| + |C| = r$. This implies that $T(r) = O(r \log r)$. □

This completes the proof of Theorem 1.

Now we consider the more interesting with-drops mode of operation.

3. With-drops Problem

The main result of this section is the following.

Theorem 2. When drops are allowed, an optimal transportation for the circular track problem (and hence for the linear track one as well) can be computed in $O(m+n)$ time.

The proof of the above theorem is developed through the end of this section, and involves several nontrivial insights into the structure of the with-drops problem. We concern ourselves with the circular track problem only, since an $O(m+n)$ time solution to the circular track problem automatically implies an $O(m+n)$ time solution to the linear track problem (by first embedding the linear track problem on a very small circular arc of a circular track and then using the circular track algorithm). Since $n \leq 2m$ it suffices to give an $O(m)$ time algorithm.

First, observe that every object can be moved to its target station by moving it in one direction only. However this observation still allows two possibilities for transporting an object: along the minor arc between its endpoints, or along the major arc. For example, Figure 5 proves that an optimal transportation for some problems must include transporting an object by the major arc. The (1,2), (1,4) and (2,4) distances are 1/3 each, and the (3,4) distance is very small. If we transport (1,2) and (2,1) by the minor arcs (Fig. 5a) then the complete transportation length is 4/3 (a pair of antiparallel augmenting edges between 2 and 3 is then needed). However if we transport (1,2) by the major arc, as in Figure 5b, we can drop it at station 4 (we henceforth call such a drop an *intermediate stop*), then complete the (4,3) and (3,4) transports and finally resume the transportation of the (1,2) arc. In this case the total path length is approximately one.

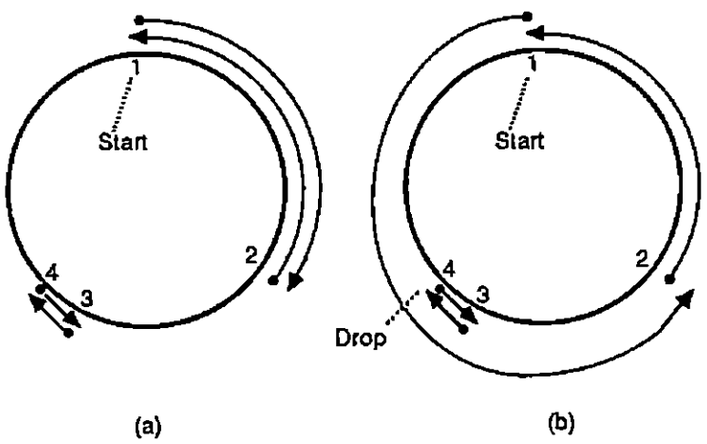


Figure 5. A with-drops problem (a), and its optimal transportation (b)

As in Section 2, each augmenting edge corresponds to motion of the gripper when it is not holding any object, and covers only one interval (if the motion spans more than one interval then each interval will get an augmenting edge). Also as before, if there are many augmenting edges across an interval then we store only a count of the number of such edges going in each direction across that interval.

Lemma 1 obviously still holds. Note that the fact that the transportation may move objects to their destinations using the major arc implies that there are 2^m possible ways to draw the m input source-to-destination pairs (whereas in Section 2 there was only one way to draw them). Lemma 10 will establish that we can restrict our attention to only m possibilities.

Let T be any transportation (with drops). We associate with T a graph $G(T)$ whose vertices are the n stations, and whose edges are the m edges corresponding to the input source-to-destination pairs, plus any augmenting edges. Each input edge in the transportation might have been covered by many intermediate stops, but in the graph $G(T)$ we simply draw the edge from its source to its destination. For example, in the transportation of Figure 5b, the edge (1,2) is a single edge even though this object gets dropped at station 4 and picked up from there later on. Note that $G(T)$ has the degree-balance property, and hence the flux corresponding to T is the same across every interval. Since $G(T)$ is degree-balanced, its scc's are also the connected components of the undirected version of $G(T)$. Therefore $G(T)$ is the union of disjoint scc's. For example, in figure 5b, the graph depicted has two scc's: {1,2}, {3,4}. A graph G is *transportable from vertex* x iff there exists a transportation T with x as its start (and hence finish) vertex, and such that $G(T)=G$. The graph shown in Figure 5b is not transportable from vertex 3, but it is transportable from vertex 1. This example also illustrates how the length of an optimal transportation now depends on where the start vertex is.

In the following, we first establish that the graph $G(T)$ of an optimal transportation

T can be computed in $O(m)$ time (Lemma 19). Then we show that T can be calculated from $G(T)$ in $O(m)$ time (Lemma 20).

For the next three lemmas, *BAL* is any degree-balanced graph, and the scc's of *BAL* are denoted by $scc_1, \cdots, scc_h$. Observe that any scc_i can individually be transported using any vertex in scc_i as the start vertex (even without drops).

Now we define the *reachability graph* of *BAL* to have vertices $scc_1, \cdots, scc_h$, and to have an edge from scc_i to scc_j iff there exists a vertex x of scc_j on some edge e of scc_i (i.e. x occurs on the circular arc covered by e). This is represented by $scc_i \rightarrow scc_j$, and we say that scc_j is *directly reachable* from scc_i.

The above definition of direct reachability implies that, if edges e_1 and e_2 of *BAL* overlap and neither one of them properly contains the other, then either e_1 and e_2 are in the same scc of *BAL* or the scc's containing e_1 and e_2 are directly reachable from each other.

We say that scc_j is *reachable* from scc_i iff there is a directed path from scc_i to scc_j in the reachability graph.

Lemma 6. In the reachability graph, if $scc_i \rightarrow scc_j$, then $scc_i \cup scc_j$ can be transported using any vertex of scc_i as the start vertex.

Proof. Since $scc_i \rightarrow scc_j$, there exists an edge e of scc_i that covers a vertex x of scc_j. Transport scc_i until point x of e is reached, drop the object, finish scc_j, and then complete scc_i. □

In fact the following generalization of Lemma 6 holds.

Lemma 7. If H is any directed spanning subtree of the reachability graph with scc_i as the root, then the union of all the scc's in H is transportable using any vertex of scc_i as the start vertex. In addition the total number of intermediate stops is no more than (the number of vertices of H)−1.

Proof. The transportation process resembles a depth-first search of H, begun at scc_i: First we mark every $scc_j \in H$ as being "new", then we mark scc_i as being "old" and begin transporting scc_i from any start vertex in scc_i. Whenever we are transporting an edge e of the scc currently being transported, we go through every edge f that has an endpoint covered by e: if f is in a child scc of the current scc, and if the scc of f is still marked "new", we mark the scc of f as being "old", interrupt the transportation of the scc of e, and recursively transport the scc of f using as start (and hence finish) vertex the endpoint of f covered by e. Every $scc_j \in H$ eventually gets transported, and every such scc_j (except the root, scc_i) causes one intermediate stop to occur during the transportation of its parent. □

Corollary 1. A graph is transportable from vertex x iff it has degree-balance and, in its reachability graph, every scc is reachable from the scc that contains x.

Proof. The "only if" part of the proof is trivial, the "if" part follows from Lemma 7. □

Corollary 2. If G is transportable from vertex x, then there is a transportation T of G (i.e. $G(T)=G$) from x such that, in T, the number of intermediate stops is no more than $n-1$.

Proof. Immediate consequence of Lemma 7. □

Lemma 8. Let S be a subset of the scc's of BAL such that every scc in S is reachable from scc_i. Suppose that the union of the scc's in S covers the circumference. Then for any degree-balanced graph G (G may be disconnected), $BAL \cup G$ is transportable from any vertex in scc_i.

Proof. Since the union of the scc's in S covers the circumference, every scc of BAL is reachable from at least one scc in S. This, and the fact that every scc in S is reachable from scc_i, implies that every scc of BAL is reachable from scc_i. Therefore (by Corollary 1) BAL is transportable from any vertex in scc_i. While transporting BAL, we are bound to reach a vertex in each of the scc's of G. At such vertices interrupt the main

transportation of BAL and finish the scc's of G. □

Lemma 9. Let T be an optimal transportation and let e be a major edge in $G(T)$. Then the scc of $G(T)$ that contains e must cover the circumference.

Proof. Let $scc(e)$ be the scc containing e, and scc_1 be the scc containing the start vertex. Since T is a transportation, $scc(e)$ is reachable in $G(T)$ from scc_1. If $scc(e)$ does not cover the circumference then its individual flux (the flux due to its edges only) is zero and hence any interval covered by $scc(e)$ is covered by at least two edges of $scc(e)$. Therefore, in $G(T)-e+e^c$, the scc containing e^c is still reachable from scc_1, and it now covers the circumference. Therefore, by Lemma 8, all the other scc's of $G(T)-e+e^c$ are transportable from the start vertex. Thus $G(T)-e+e^c$ is transportable from the start vertex, which contradicts the optimality of T (since e is longer than e^c). □

Lemma 10. In any optimal transportation, at most one object is moved to its destination along the major arc.

Proof. Let T be an optimal transportation, let $scc_1, \cdots, scc_h$ be the scc's of $G(T)$, and let the start vertex be in scc_1. Suppose that $G(T)$ has two major edges e_i and e_j, respectively in scc_i and scc_j. By Lemma 9, scc_i covers the circumference, and so does scc_j.

Case 1: $scc_i \neq scc_j$. Without loss of generality, we can assume that, in the reachability graph, scc_i is reachable from scc_1 without going through scc_j. Now, modify scc_j by shortening e_j. Because scc_i is still reachable from scc_1 and still covers the circumference, even this modified scc_j along with all the other scc's are transportable from the start vertex (by Lemma 8). Since we made e_j shorter, the new transportation has smaller length than T, a contradiction.

Case 2: $scc_i = scc_j$. We distinguish two sub-cases.

Sub-case 2.1: Every interval covered by e_i is covered by at least one other edge of scc_i. Consequently, modifying scc_i by shortening e_i leaves the circumference covered

by the new scc_j, which is still reachable from scc_1. Therefore, by Lemma 8, the graph obtained from $G(T)$ by shortening e_i is still transportable. This contradicts the optimality of T.

Sub-case 2.2: Some interval is covered by e_i and by no other edge of scc_i. Then the individual flux of scc_i is +1 or −1, and therefore every interval in the region covered by both e_i and e_j is also covered by at least one other edge of scc_i (because the individual flux of scc_i is an odd number). In this case simultaneously shortening both e_i and e_j leaves the new scc_i still covering the circumference, leading to a contradiction. □

Lemma 10 reduces from 2^m to m the number of possible drawings of the m input edges that need to be considered, but it does not yet give an $O(m)$ time algorithm. We must still identify, in $O(m)$ time, which edge (if any) needs to be drawn along the major arc. Even if we knew which drawing of the m input edges is best, it is not clear how to augment these into a minimum-length graph that is transportable (from the start vertex). All these nontrivial issues are addressed below.

Lemma 11. Let T be an optimal transportation, and let e be a major edge of $G(T)$. Then at least one interval covered by e is not covered by any other edge of $G(T)$.

Proof. Suppose to the contrary that the region covered by e is also covered in $G(T)-e$. Then there exists in $G(T)-e$ a sequence of edges $f_1, \cdots f_s$ such that $f_1 \cup \cdots \cup f_s$ covers e, and every f_i contains an endpoint of f_{i+1}, $1 \le i < s$. Figure 6 illustrates this ($s=5$), ignoring edge directions as well as the distinction between input edges and augmenting ones. We assume that the sequence $f_1, \cdots f_s$ has smallest number of elements ($=s$), among all such sequences covering e; this implies that neither one of f_i and f_{i+1} contains the other. Note that Lemma 10 implies that $s > 1$. Let scc_1 be the scc of $G(T)$ that contains the start vertex, and for any edge x, $scc(x)$ denotes the scc that contains x. We distinguish two cases.

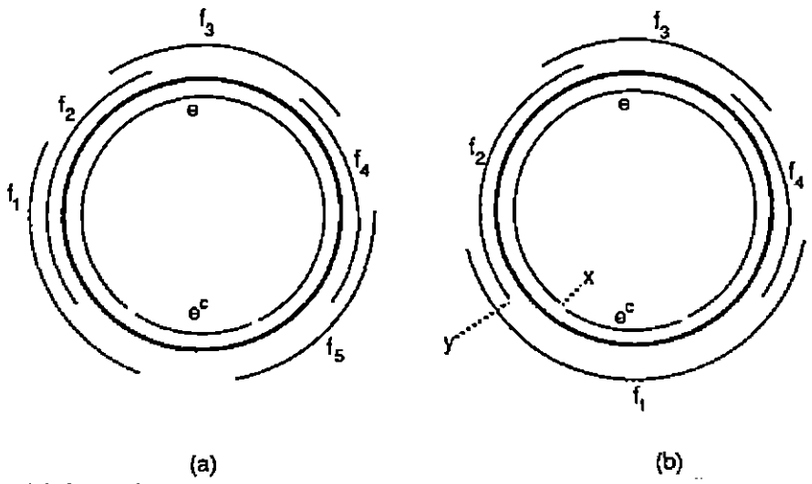


Figure 6. Illustrating the two cases of the proof of Lemma 11

Case 1: e^c, the complement of e, covers at least one endpoint of at least one of the f_i's (see Figure 6a). For every $1 \le i < s$, the fact that f_i and f_{i+1} overlap without containment implies that we have either $scc(f_i)=scc(f_{i+1})$, or $scc(f_i)$ and $scc(f_{i+1})$ are directly reachable from each other. Therefore the $scc(f_i)$'s are reachable from one another and from $scc(e)$. Now, in $G(T)-e+e^c$, $scc(e^c)$ is still reachable from scc_1, and every $scc(f_i)$ is still reachable from $scc(e^c)$ (because e^c contains an endpoint of some f_i). Since the union of the $scc(f_i)$'s with $scc(e^c)$ covers the circumference, it follows (by Lemma 8) that the graph $G(T)-e+e^c$ is transportable (from the start vertex). This contradicts the optimality of T.

Case 2: e^c does not cover any endpoint of any f_i (see Figure 6b). Note that this implies that some f_i, say f_1, contains e^c. Let l be the shortest distance between any endpoint of e (say, x) and any endpoint of any f_i (say, endpoint y of f_α). Now, in $G(T)$, simultaneously shorten e and add two antiparallel augmenting arcs of length l each between x and y. In the resulting graph, the two additional augmenting edges that were added make $scc(e^c)=scc(f_\alpha)$, and therefore every $scc(f_i)$ is reachable from scc_1. Since the union of the $scc(f_i)$'s covers the circle, Lemma 8 implies that the new graph is transportable. The decrease in cost with respect to the original $G(T)$ is given by

$$|e|-|e^c|-2\cdot l = 1-2\cdot(|e^c|+l) > 1-2|f_1|\geq 0$$

(note that $|e^c|+l<|f_1|\leq 1/2$, and recall that the circle's circumference is unity). Thus the new transportable graph is shorter than $G(T)$, a contradiction. □

Corollary 3. Let T be an optimal transportation, and let $\psi(T)$ denote the flux of T. If $G(T)$ contains a major edge, then $|\psi(T)|=1$.

Proof. The interval covered by e and by no other edge, in Lemma 11, has flux value of $+1$ or -1. □

We henceforth use D_0 to refer to the graph which consists of all m input edges, drawn so that each of them is minor.

Lemma 12. If D_0 covers the circumference, then no optimal transportation can contain a major edge.

Proof. Let T be an optimal transportation, e be a major edge of $G(T)$. Since e is major, its complement e^c is in D_0. Since D_0 covers the circumference, D_0-e^c covers e. Since $G(T)-e$ contains D_0-e^c, $G(T)-e$ covers the region covered by e. This contradicts Lemma 11. □

If two edges of D_0 overlap without either of them containing the other, then in every degree-balanced augmentation of D_0, these two edges either belong to the same scc or to two scc's that are directly reachable from each other. Based on this observation let us define a relation, $\simeq$, between any two edges of D_0, as follows:

(a) For any two input edges e_1 and e_2, $e_1 \simeq e_2$ iff either these two edges share a common endpoint, or they overlap but neither one of them contains the other.

(b) Transitively close the relation $\simeq$.

Note that $\simeq$ is an equivalence (eq.) relation, and, in addition, no two eq. classes of $\simeq$ have a vertex in common. Also note that in any degree-balanced augmentation of D_0, two input edges in the same eq. class of $\simeq$ belong to scc's that are reachable from each

other.

Define an ordering $<$ among the eq. classes of $\approx$ as follows: If C_i and C_j are any two distinct eq. classes of $\approx$, then $C_i < C_j$ iff some edge of C_j covers all the vertices of C_i (and hence no edge of C_i covers any vertex of C_j). Note that $\approx$ is independent of the drawing of the edges, whereas $<$ does depend on it. Based on this ordering we can draw a forest of trees F whose nodes are the eq. classes of $\approx$ (the parent of C_i is the "smallest" class above it according to the $<$ relation). An example of such a forest is shown in Figure 7. Let $\tau_1, \cdots, \tau_k$ be the trees of F, listed in clockwise cyclic order and such that τ_1 contains the start vertex. Let $root(\tau_i)$ denote the eq. class at the root of tree τ_i. Let $g_1, \cdots, g_k$ be the lengths of the gaps that separate, on the circumference, the regions covered by the τ_i's; g_i is the gap between τ_i and τ_{i+1} (in Figure 7, g_1 is the arc length from the head of edge h to the tail of edge z, and g_2 is the arc length from the head of edge j to the head of edge a).

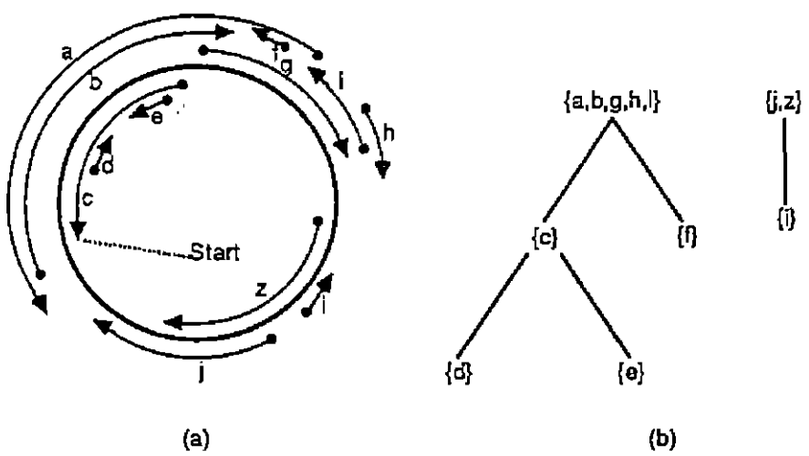


Figure 7. The forest (b) corresponds to the input edges shown in (a)

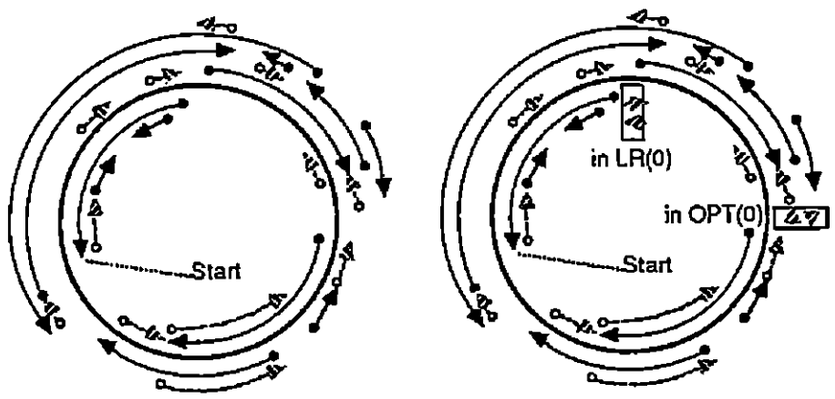
Lemma 13. The eq. classes of $\approx$, and the forest F of eq. classes, can be computed in $O(m)$ time.

Proof. See Appendix A. □

Let G be any degree-balanced augmentation of D_0. (Note that G is not necessarily

transportable and, since it is degree-balanced, its scc's are the same as the connected components of the undirected version of it.) We say that eq. classes C_i and C_j of $\equiv$ are *directly linked in* G iff at least one of the scc's of G contains vertices from both C_i and C_j; they are *linked* iff there is a sequence of eq. classes beginning with C_i and ending with C_j such that every two eq. classes in the sequence are linked.

Let $DB(\psi)$ be the graph corresponding to the minimum degree-balanced augmentation of D_0 that results in flux ψ; note that $DB(\psi)$ is unique but need not be transportable. Figure 8a shows $DB(0)$ for the D_0 of Figure 7. Let $db(\psi)$ be the total length of the augmenting edges in $DB(\psi)$ (i.e. edges in $DB(\psi)-D_0$). By Lemma 4, it takes $O(m)$ time to compute all the $db(\psi)$ values, $-m-1 \le \psi \le m+1$.



(a) (b)

Figure 8. Illustrating $DB(0)$, $LR(0)$, and $OPT(0)$ for the input edges of Figure 7

Let C_{start} be the eq. class containing the start vertex (recall that τ_1 is the tree of F that contains C_{start}). Let $LR(\psi)$ be a minimum augmentation of $DB(\psi)$ that makes C_{start} linked to $root(\tau_1)$ while keeping the flux equal to ψ. Hence $LR(\psi)$ consists of $DB(\psi)$ plus some augmenting pairs of antiparallel edges (in Figure 8b there is one such pair, marked "in $LR(0)$"). Let $lr(\psi)$ be the total length of the augmenting edges in $LR(\psi)$ but not in $DB(\psi)$.

Lemma 14. The $lr(\psi)$'s, for $-m-1 \le \psi \le m+1$, can all be computed in $O(m)$ time. For

any given flux value ψ_0, $LR(\psi_0)$ can be computed in $O(m)$ time.

Proof. See Appendix B. □

In the next two lemmas, we show how to compute the optimum among all transportations in which no edge is major.

Let $OPT_{minor}(\psi)$ be the graph corresponding to a transportation that is optimal among all transportations of flux ψ and that do not have any major edge. If $LR(\psi)$ is transportable from the start vertex, then $OPT_{minor}(\psi)=LR(\psi)$. Otherwise, pairs of antiparallel edges are needed across some of the gaps (intervals not covered by D_0) in order to turn $LR(\psi)$ into $OPT_{minor}(\psi)$. In Figure 8b there is one such antiparallel pair, and each of the two edges in it has length g_1. (Recall that $g_1, \cdots, g_k$ are the lengths of the gaps separating the τ_i's, listed in clockwise cyclic order, and such that τ_1 contains the start vertex.)

Lemma 15. Let $\hat{g}(\psi)$ be the total length of the augmenting pairs of antiparallel edges in $OPT_{minor}(\psi)-LR(\psi)$. If $\psi\neq 0$ then $\hat{g}(\psi)=0$, otherwise $\hat{g}(0)=2\cdot(\sum_{i=1}^{k} g_i-\max_i g_i)$.

Proof. $\hat{g}(\psi)$ is the length of the additional edges needed for linking all the $root(\tau_i)$'s together at flux ψ. No additional edges are needed if $\psi\neq 0$ because then every $root(\tau_i)$ is already linked to $root(\tau_{i+1})$ in $LR(\psi)$. If $\psi=0$ then we link the $root(\tau_i)$'s by adding an antiparallel pair across every gap except the longest. □

The above lemma immediately implies the following.

Corollary 4. The pairs of antiparallel augmenting edges in $OPT_{minor}(\psi)-LR(\psi)$ can be computed in $O(m)$ time.

Lemma 16. It is possible to compute, in $O(m)$ time, a graph $G(T_0)$ for some transportation T_0 which is optimal among all transportations none of whose edges is major.

Proof. Note that such a graph is simply an $OPT_{minor}(\psi)$ of minimum total length. The

total length of such a graph is equal to (length of edges of D_0) $+\min_{\psi}\{\, db(\psi)+lr(\psi)+g(\psi)\,\}$. Therefore lemmas 4, 14, 15 immediately imply that we can find a flux value ψ_0 such that the length of $OPT_{minor}(\psi)$ is minimum for $\psi=\psi_0$. We now show that, once we know ψ_0, $OPT_{minor}(\psi_0)$ itself can be computed in $O(m)$ time. It is trivial to obtain $DB(\psi_0)$ in $O(m)$ time. By Lemma 14, $LR(\psi_0)$ can be obtained from $DB(\psi_0)$ in $O(m)$ time. By Corollary 4, $OPT_{minor}(\psi_0)$ can be obtained from $LR(\psi_0)$ in $O(m)$ time. □

We now consider transportations that have exactly one major edge.

Lemma 17. Let E be any set of edges on the circle, exactly one of which is major (call it e). Let G be any degree-balanced augmentation of E. If E covers the circumference, then every scc of G is reachable from the scc that contains e.

Proof. Let $scc(e)$ denote the scc of G containing edge e. Since E covers the circumference, there exists in E a sequence of edges $f_1, \cdots f_s$ such that $f_1 \cup \cdots \cup f_s$ covers e^c, and every f_i contains an endpoint of f_{i+1}, $1 \le i < s$. We assume that the sequence has smallest number of elements ($=s$), and hence none of f_i and f_{i+1} contains the other. See Figure 9, and note that because e is the only major edge of E, it must contain at least one endpoint of one of the f_i's. Therefore at least one $scc(f_i)$ is reachable from $scc(e)$. For every $1 \le i < s$, the fact that f_i and f_{i+1} overlap without containment implies that they belong to scc's that are reachable from one another. Therefore every $scc(f_i)$ is reachable from $scc(e)$, and hence (by Lemma 8) any scc of G is also reachable from $scc(e)$. □

Recall that we have already stated (in Lemma 10) that in an optimal augmentation at most one edge is major. The next lemma refines this statement.

Lemma 18. There exists an optimal transportation T such that, if $e \in G(T)$ is major, then its complement $e^c \in D_0$ is in a root eq. class of $\simeq$ and is the longest edge in that class.

Proof. Let there exist an optimal transportation T in which edge e is major, and hence

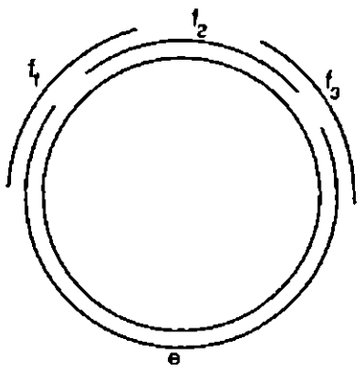


Figure 9. Illustrating the proof of Lemma 17

its complement e^c is in D_0). Note that $|e|=1-|e^c|$, and $|e^c|<1/2$. Let e^c be in eq. class C of tree τ_j (recall that the eq. classes as well as the τ_i's are defined using D_0, where all edges are minor).

By Lemma 12, D_0 does not cover the circumference. Lemma 11 implies that e is the only input edge that covers the k gaps not covered by D_0.

Let $G'=G(T)-e+e^c$, and note that G' is degree-balanced and contains D_0. No scc of G' covers the circumference, because otherwise G' is transportable (from the start vertex), contradicting the optimality of $G(T)$. Therefore every scc of G' has an individual flux of zero, which implies the following:

(*) If an interval is covered by an scc of G' then it is covered by at least two edges of that scc.

Now we prove that e^c is longest in C. Suppose to the contrary that edge $f \in C$ is longer than e^c. Consider the scc of G' that contains e^c (call it $scc(e^c)$). In G', $scc(e^c)$ and $scc(f)$ are mutually reachable from one another because G' contains D_0 and, in D_0, e^c and f are in the same eq. class. Now, since T is a transportation, the scc of $G(T)$ that contains e is reachable from the scc of the start vertex, and therefore in G' $scc(e^c)$ is reachable from the scc of the start vertex. Therefore in G', $scc(f)$ is reachable from the scc of the start vertex. Consequently, we have:

(**) In $G'-f+f^c$, $scc(f^c)$ is reachable from the scc of the start vertex.

Now, (*) implies that in G', every interval covered by f is also covered by at least one edge of $scc(f)$ other than f. Therefore $scc(f)-f+f^c$ covers the circumference, and hence by Lemma 17 every scc of $G'-f+f^c$ is reachable from $scc(f^c)$. This and (**) together imply that in $G'-f+f^c$, every scc is reachable from the scc of the start vertex. Thus the graph obtained from $G(T)$ by shortening e and simultaneously lengthening f is also transportable. This contradicts the optimality of T. Thus e^c must be longest in its eq. class.

Now we prove that there is always an optimal T in which C is root in its eq. class, i.e. $C=root(\tau_j)$. Suppose that $C\neq root(\tau_j)$. Then the parent of C in τ_j surely contains an edge f which is longer than e^c and covers it entirely. Let $\hat{G}=G(T)-e+e^c-f+f^c$. Let CYCLES be the set of augmenting edges defined as follows: across every interval covered by f but not by e^c, add two antiparallel edges. Thus the length of CYCLES is $2\cdot(|f|-|e^c|)$. Simple arithmetic shows that adding CYCLES to $\hat{G}$ results in a graph of total length no more than that of $G(T)$. We now show that $\hat{G}$+CYCLES is transportable. First note that $\hat{G}$+CYCLES has degree-balance. Now observe that, because of the presence of CYCLES, e^c and f^c are in the same scc of $\hat{G}$+CYCLES (call it scc_x). Furthermore, since T is a transportation, the scc of $G(T)$ that contains e is reachable from the scc of the start vertex, and therefore in $\hat{G}$+CYCLES, scc_x is reachable from the scc of the start vertex. This and the fact that scc_x covers the circle implies that $\hat{G}$+CYCLES is transportable. Since $\hat{G}$+CYCLES is transportable and has length no more than that of $G(T)$, we have obtained from $G(T)$ another transportable graph of optimal length, one in which the complement of the major edge is in an eq. class that is one level higher in τ_j. We can repeatedly do this until we end up with an optimal transportation whose major edge's complement belongs to $root(\tau_j)$. □

Lemma 19. It is possible to compute, in $O(m)$ time, a graph $G(T)$ for some optimal

transportation T.

Proof. Use Lemma 16 to compute $G(T_0)$ and let $Cost_0$ be its total length. Use Lemma 13 to compute F: if F has one tree only then return $G(T_0)$. Now, suppose that F has more than one tree, i.e. $k>1$. Let $Cost_1$ be the length of a transportation T_1 having exactly one major edge in it, and which is optimal among all transportations that have exactly one major edge. The optimal transportation T will have length $\min\{ Cost_0, Cost_1 \}$. If $Cost_1<Cost_0$ then Lemma 11 tells us that in our search for the value $Cost_1$, we can restrict our attention to transportations T_1 such that there exists at least one interval covered by only the major edge of $G(T_1)$. Within this class of transportations, any T_1 will have to be such that $G(T_1)$ has the following properties (i)-(iii), where e denotes the major edge and e^c its complement.

(i) $G(T_1)-e+e^c$ has flux equal to zero and thus contains $DB(0)$ (which contains D_0).

(ii) In $G(T_1)-e+e^c$, C_{start} is linked to the root eq. class of its tree (τ_1). Therefore $G(T_1)-e+e^c$ contains $LR(0)$ as well.

(iii) In $G(T_1)-e+e^c$, the root eq. class of the tree τ_j that contains e, is linked to $root(\tau_1)$ (simply because T_1 is a transportation). The augmenting edges that cause $root(\tau_j)$ and $root(\tau_1)$ to be linked are pairs of antiparallel edges across either all of gaps $\#1, \cdots, \#(j-1)$, or all of gaps $\#j, \cdots, \#k$ (depending on whichever of $g_1+\cdots+g_{j-1}$ or $g_j+\cdots+g_k$ is smaller).

Thus $G(T_1)-e+e^c$ equals $LR(0)$ plus the augmenting edges referred to in (iii). The length of $G(T_1)-e+e^c$ is therefore equal to

$$(\text{length of } D_0)+db(0)+lr(0)+2\cdot\min\{g_1+\cdots+g_{j-1}\ ,\ g_j+\cdots+g_k\}.$$

The length of $G(T_1)$ therefore equals (using $|e|+|e^c|=1$):

$$(\text{length of } D_0)+db(0)+lr(0)+2\cdot\min\{g_1+\cdots+g_{j-1}\ ,\ g_j+\cdots+g_k\}+1-2|e^c| \qquad (\dagger).$$

The first three terms of the sum (†) are the same for any such transportation T_1. Now, the edge e of $G(T_1)$ for which the sum of the last two terms of (†) is smallest should be the one to follow the major arc. It is trivial to identify this edge in $O(m)$ time and compute the corresponding sum (†): If this sum is smaller than $Cost_0$ then this $G(T_1)$ corresponds to the optimal transportation, otherwise it is $G(T_0)$. □

Lemma 20. Let G be transportable from a designated start vertex. A transportation T such that $G(T)=G$ can be found in $O(m)$ time.

Proof. The proof of Lemma 7 suggests an algorithm for obtaining such a transportation. However, we cannot afford to create the graph of the reachability relation among scc's, because such a graph can be dense. Instead, we create a graph *EQUIV'*, defined as *EQUIV* in the proof of Lemma 13 (appendix A) except that we now use the input edges as they are drawn in G rather than as they are drawn in D_0 (G may include a major edge). In addition, we compute the eq. classes of a relation $\simeq'$ and forest of eq. classes F' defined as $\simeq$ and F were before, except that we now use the input edges as they are in G rather than in D_0. For each eq. class C that is parent in F' of class C', we arbitrarily select an edge e in C and an edge f in C' such that e properly contains f (at least one such pair e,f exists); we call edge e the *parent* of f, and we call f a *child* of e. Note that the forest F' induces at most $n-1$ such parent-child pairs. For each edge e, let the list of edges $ADJ(e)$ be the union of (i) the children of e induced by F', and (ii) the adjacency list of e in the undirected graph *EQUIV'*. We are now ready to describe how to obtain T such that $G(T)=G$. The transportation process resembles a depth-first search of the scc's, begun at scc_1: First we mark all scc's of G as being "new", then we mark scc_1 as being "old" and begin transporting scc_1 from the start vertex of G (recall that any scc can individually be transported using any vertex in it as start and finish). Whenever we are transporting an input edge e of the scc currently being transported, we go through the list $ADJ(e)$: for each $f \in ADJ(e)$ that is in a "new" scc, we mark the scc of f as being

"old", interrupt the transportation of the scc of e, and recursively transport the scc of f using as start (and hence finish) vertex an endpoint of f covered by e. It is trivial to implement this procedure in $O(m)$ time. Correctness follows from the facts that (i) *EQUIV'*, even though it is sparse, captures all the "overlap without containment" relashionships between pairs of input edges, and (ii) the parent-child pairs induced by F' capture enough of the "proper containment" relationships between pairs of edges. More precisely, (i) guarantees that once an edge of eq. class C is reached by the transportation, eventually every edge of that class C will be transported. On the other hand, (ii) guarantees that once an edge of an eq. class C is reached by the transportation, eventually every eq. class in the subtree of C in F' will be transported. □

The last two lemmas imply Theorem 2, which is the main result of this section.

4. Concluding Remarks

It is easy to see that our solutions to the angular motion problem (with or without drops) also work in the presence of obstacles. A pre-processing step computes the visibility polygon from the fixed pivot point of the robot arm (of course all n stations must be visible from the pivot, since an invisible station is unreachable by the robot arm). The robot arm must remain within the visibility polygon while performing the transportation. While this does not affect the rotational distance function, the telescoping distance function has to be modified appropriately because the robot arm may have to be drawn in so as to clear an obstacle.

It would be interesting to investigate the with-drops circular track problem when the gripper can simultaneously hold c objects, where c is a constant larger than one. We conjecture that Lemma 10 generalizes to that case, i.e. no optimal transportation can transport more than c objects along the major arc. A special case of this problem for a linear track was treated in [K].

Acknowledgement. The authors are grateful to an anonymous referee for his comments on the case with obstacles.

References

[AHU] A.V. Aho, J.E. Hopcroft and J.D. Ullman, *The Design and Analysis of Computer Algorithms*, Addison-Wesley, Reading, Mass., 1974.

[ET] K. P. Eswaran and R. E. Tarjan, "Augmentation Problems," *SIAM J. on Computing*, 1976, 653-665.

[E] S. Even, *Graph Algorithms*, Computer Science Press, Potomac, Maryland, 1979.

[FT] M. L. Fredman and R. E. Tarjan, "Fibonacci Heaps and their Uses in Improved Network Optimization Algorithms," *Proc. 25th Annual IEEE Symp. on Foundations of Computer Science*, 1984, pp. 338-346.

[K] R. M. Karp, "Two Combinatorial Problems Associated with External Sorting," *Combinatorial Algorithms, ed. by R. Rustin*, Courant Computer Science Sym. 9, Algorithmics Press, New York, 1972.

[P1] C. H. Papadimitriou, "On the Complexity of Edge Traversing," *JACM*, 1976, 544-554.

[P2] C. H. Papadimitriou, "The Euclidean traveling salesman problem is NP-complete," *Theoretical Computer Science*, 1977, 237-244.

Appendix A

This appendix proves Lemma 13. Computing F in linear time when we know the eq. classes is easy and this construction is omitted. We give an $O(m)$ time algorithm for computing the eq. classes of $=$. For the purpose of this computation all the edges in D_0 can be considered undirected. An edge covers the circular region going clockwise from its *beginning* to its *end*. For an edge $e \in D_0$, let $CW(e)$ (resp. $CCW(e)$) be the set of edges of D_0 whose beginning (resp. end) is in the region covered by e and whose end (resp. beginning) is in the region not covered by e. Note that $f \in CW(e)$ iff $e \in CCW(f)$. The *clockwise* (resp. *counterclockwise*) *successor* of e is the edge of $CW(e)$ (resp. $CCW(e)$) whose beginning (resp. end) is encountered first by a clockwise (resp. counterclockwise) sweep starting at e's beginning (resp. end). In Figure 10a, $CW(e)=\{c,d\}$, $CCW(e)=\{a,b\}$, the clockwise successor of e is d and its counterclockwise successor is a.

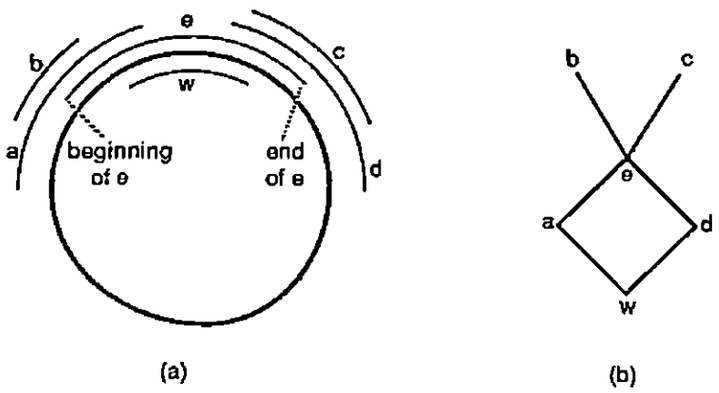


Figure 10. Illustrating *EQUIV*

The clockwise and counterclockwise successors of every edge of D_0 can easily be computed in $O(m)$ time. Assume this has already been done. Now create, in $O(m)$ time, the undirected graph *EQUIV* whose vertex set is D_0 and such that $\{e,f\}$ is an edge in it iff one of e and f is (clockwise or counterclockwise) successor of the other. Figure 10b shows the graph *EQUIV* corresponding to Figure 10a. Obviously, *EQUIV* has at most $2m$ edges, since every $e \in D_0$ has at most two successors (one clockwise, one counterclockwise). Hence the connected components of *EQUIV* can be computed in $O(m)$ time. Thus the lemma will follow immediately when we establish that the connected components of *EQUIV* are the equivalence classes of $\simeq$. To prove this, it suffices to show that, if any two edges e and f overlap without containment (i.e. without either one of them properly containing the other), then there is a path between e and f in *EQUIV*. If edges e and f overlap without containment, then we define the *overlap number* of the pair $\{e,f\}$ to be the number of stations covered by both e and f, not counting the endpoints of e and f. In Figure 10a, the overlap number of $\{e,d\}$ is 2, that of $\{a,w\}$ is 0, and that of $\{e,w\}$ is undefined because e properly contains w. We prove the desired result by contradiction: suppose there exist pairs of edges that overlap without containment and have no path between them in *EQUIV*. Among all such pairs, choose $\{e,f\}$ to have maximum overlap number. Without loss of generality, assume $f \in CW(e)$. Let g be the clockwise successor of e, and let h be the counterclockwise successor of f (see

Figure 11). Since $\{e,g\}$ and $\{f,h\}$ are edges in *EQUIV*, and since e and f are in different connected components of *EQUIV*, we must have $g \neq f$ and $h \neq e$. Note that, as shown in Figure 11, the end of g must occur after that of f in the clockwise direction, because otherwise the overlap number of $\{g,f\}$ would exceed the overlap number of $\{e,f\}$, a contradiction. Similarly, the beginning of h comes before that of e, since otherwise the overlap number of $\{e,h\}$ would exceed the overlap number of $\{e,f\}$, a contradiction. But then, the overlap number of $\{g,h\}$ exceeds the ovelap number of $\{e,f\}$, a contradiction. This completes the proof of Lemma 13.

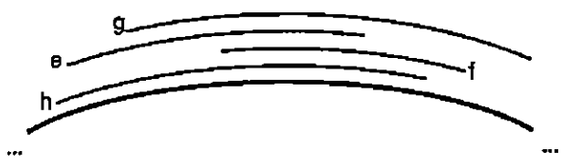


Figure 11. Illustrating clockwise and counterclockwise successors

Appendix B

This appendix proves Lemma 14.

For every eq. class $C \in \tau_1$, let $L(C)$ (resp. $R(C)$) be the vertex of C such that a clockwise sweep of the region covered by C starts at $L(C)$ (resp. ends at $R(C)$). In Figure 7, $L(\{c\})$ is the head of edge c, $L(\{j,z\})$ is the tail of edge z. If C is not $root(\tau_1)$, let the *left* (resp. *right*) *neighbor* of C be the first vertex of $\tau_1 - C$ that is is encountered by a counterclockwise (resp. clockwise) sweep begun at $L(C)$ (resp. $R(C)$). In Figure 7, the left neighbor of eq. class $\{c\}$ is the tail of edge b, its right neighbor is the tail of edge g. We use $LN(C)$ and $RN(C)$ to denote the left and right neighbors of C, respectively. Note that $LN(C)=L(C)-1$, and $RN(C)=R(C)+1$. Therefore we can talk about the *intervals* $(LN(C),L(C))$ and $(R(C),RN(C))$. Let the eq. class containing any vertex x be denoted by $Class(x)$. Note that $Class(LN(C))$ (resp. $Class(RN(C))$) is either the left (resp. right) sibling of C in τ_1, or the parent of C in τ_1. Also note that

$Class(LN(C))=Class(RN(C))$ is possible (if both are the parent of C). Now, imagine starting at C_{start} and repeatedly applying the function $Class(LN(\cdot))$ until $root(\tau_1)$ is reached. This defines a sequence S_{Left} of eq. classes from C_{start} to $root(\tau_1)$. Starting at C_{start} and repeatedly applying the function $Class(RN(\cdot))$ similarly defines a sequence of eq. classes S_{Right}. Let us draw a directed graph Q to depict what S_{Left} and S_{Right} might look like; i.e. the vertices of Q are the eq. classes of $S_{Left} \cup S_{Right}$ and Q has an edge from C_i to C_j iff C_j immediately follows C_i in S_{Left} or in S_{Right}. Figure 12 shows such a graph Q. Note that the two paths corresponding to S_{Left} and S_{Right} are not necessarily vertex-disjoint and may come together at *articulation points* more than once. Let these articulation points be $A_1, A_2, \cdots, A_t$, listed by increasing distance from C_{start} in Q. Let $A_0=C_{start}$, $A_{t+1}=root(\tau_1)$.

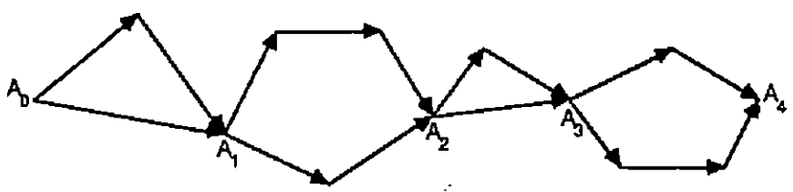


Figure 12. A typical graph Q

Let the interval that *corresponds to* the pair $\{C, Class(LN(C))\}$ be $(LN(C), L(C))$, and let the interval that corresponds to the pair $\{C, Class(RN(C))\}$ be $(R(C), RN(C))$. Now, let $Q(\psi)$ be the *undirected* graph obtained by removing from the undirected version of Q all the edges $\{C, Class(LN(C))\}$ and $\{C, Class(RN(C))\}$ whose corresponding intervals are not covered by any augmenting edge of $DB(\psi)$. Note that if $Q(\psi)$ does not contain a path between A_0 and A_{t+1}, then antiparallel pairs need to be added to $DB(\psi)$ to turn it into $LR(\psi)$; we next investigate which such antiparallel pairs should be added (the total length of these pairs is $lr(\psi)$). Let $E(\psi)$ be a subset of edges in the undirected version of Q such that adding these edges to $Q(\psi)$ causes a path to exist between A_0 to A_{t+1}; choose $E(\psi)$ to be such that the sum of the lengths of the intervals corresponding to its elements is minimum. It is not hard to see that $LR(\psi)$ is obtained by

adding to $DB(\psi)$ an antiparallel pair across every interval corresponding to an element of $E(\psi)$. Thus $lr(\psi)$ is twice the total length of the intervals corresponding to the elements of $E(\psi)$; let us call this the *cost of* $E(\psi)$. Now our problem is to keep track of the cost of $E(\psi)$ as ψ changes from $-m-1$ to $m+1$. Write $E(\psi)$ as $E_0(\psi)+E_1(\psi)+\cdots+E_t(\psi)$ where $E_i(\psi)$ is the subset of $E(\psi)$ that is in the biconnected component of the undirected version of Q having A_i and A_{i+1} as articulation points. Now, as ψ changes, $Q(\psi)$ changes as well, but it changes at no more than $3n$ values of ψ. Each such change in $Q(\psi)$ is due to the appearance or disappearance of an augmenting edge of $DB(\psi)$ across an interval, and it is trivial to update, in constant time per interval affected, the cost of each $E_i(\psi)$ affected (once one realize that the portion of Q between A_i and A_{i+1} consists of two disjoint paths, one can easily supply the other details). Thus it is possible to compute a description of the costs of all the $E_i(\psi)$'s in $O(m)$ time. Getting the description of the cost of $E(\psi)$ takes an additional $O(m)$ time, which completes the proof of the first part of the lemma.

For a given flux value ψ_0, computing the actual set $E(\psi_0)$ (and the intervals corresponding to its elements) is easily done in $O(m)$ time, in view of the preceding discussion: first we compute Q, then $Q(\psi_0)$, then $E(\psi_0)$. This completes the proof of Lemma 14.