A Cycle Augmentation Algorithm for Minimum Cost Multicommodity Flows on a Ring

Bruce Shepherd
bshep@research.bell-labs.com

Lisa Zhang
ylz@research.bell-labs.com

Bell Laboratories
600-700 Mountain Avenue
Murray Hill, NJ 07974

Abstract

Minimum cost multicommodity flows are a useful model for bandwidth allocation problems in telecommunications networks. These problems are arising more frequently as regional service providers wish to carry their traffic over some national core network. We describe a simple and practical combinatorial algorithm to find a minimum cost multicommodity flow in a ring network. Apart from 1 and 2-commodity flow problems, this seems to be the only such "combinatorial algorithm" for a version of exact mincost multicommodity flow. The solution it produces is always half-integral, and by increasing the capacity of each link by one, we may also find an integral routing of no greater cost. The "pivots" in the algorithm are determined by choosing an $\epsilon > 0$, *increasing* and *decreasing* sets of variables, and adjusting these variables up or down accordingly by ϵ. In this sense, it generalizes the cycle cancelling algorithms for single source mincost flow. Although the algorithm is easily stated, proof of its correctness and polynomially bounded running time are more complex.

Keywords: *Multicommodity flow, ring networks, linear programming, bandwidth allocation, VPN (virtual private network).*

1 Introduction

A *multicommodity flow* problem is determined by an edge-capacitated network together with a list of *demands* which need to be fulfilled between certain pairs of nodes in the network. We let the capacitated network be an undirected *supply* graph $G = (V, E)$ where $V = \{0, 1, 2, \ldots, n-1\}$ and let $u : E \to \mathbb{Z}_+$ be the edge-capacity vector. We represent demands by a simple undirected *demand graph* $H = (V, F)$ together with a positive integer demand function $h : F \to \mathbb{Z}_+$. Thus $h(ij)$ is the number of connection paths, or flow units, requested between nodes i and j.

We denote by $\mathcal{P}$ the collection of all such paths in G and $\mathcal{P}_{ij}$ denotes the set of paths with endpoints i, j. For any edge $\alpha \in E$, $\mathcal{P}^\alpha$ denotes the set of paths which contain α. A *multicommodity flow*, or *fractional routing*, in G is an assignment $x : \mathcal{P} \to \mathbb{R}_+$ of values to

the paths of G. A flow x *fulfils* h if

$$\sum_{P \in \mathcal{P}_{ij}} x(P) = h(ij) \quad \text{for each demand edge } ij \in F. \tag{1}$$

A flow x *satisfies the edge-capacities* if

$$\sum_{P \in \mathcal{P}^e} x(P) \leq u(e) \quad \text{for every edge } e \in E. \tag{2}$$

If x fulfils h and satisfies the edge-capacities, then it is *feasible* for the problem given by the quadruple (G, H, h, u). If each $x(P)$ is an integer, then x is called an *integer multicommodity flow*, or *routing*.

Multicommodity flow is a cornerstone optimization problem in network design. These problems can be solved in polynomial-time via general linear programming (LP) methods (whereas routing, or integer multicommodity flows, are NP-hard). This has the advantage that off-the-shelf software may be used as a solution technique for a broad range of applications. One disadvantage, however, is that products which rely on these methods must bundle this software with each copy. addition, resorting to a general purpose algorithm does not allow improvements in efficiency that may be possible by harnessing the special combinatorial structure of a problem. Unfortunately, there is presently no known "combinatorial" algorithm for general exact multicommodity flow (cf. [3]). Even for special instances there are few such algorithms, the prime exceptions being for 1 and 2 commodities (i.e., $|F| = 1, 2$) including algorithms of Ford and Fulkerson [5]. We mention that a similar state of affairs held for generalized-minimum-cost flows, even for a single commodity, until recent work of Wayne [17].

Ring optimization problems have attracted considerable interest in recent years due initially to the advent of SONET rings in modern telecommunication networks. In particular, the *ring loading problem* [2, 4, 9, 14, 18] has played a key role in determining the dimensions of the equipment offered. Formally, this is the problem of finding the minimum link capacity L for which the ring, with capacity L on each link, can support a given traffic demand graph. In the SONET setting, this meant that traffic was routed 'unsplittably', i.e., on a single path, without violating the link capacities. In some network design problems, however, the requirement that traffic for a given source-demand pair be unsplittably routed is not critical, or non-existent. Indeed, this can be the case for certain multi-ring networks. If this is the case, the problem of feasibility is to determine whether there is a multicommodity flow that fulfils a given demand. If this can be done, then a second step is to reserve capacity for such a flow at minimum cost. This problem can be faced by regional service providers leasing bandwidth from a national carrier. Alternatively, from the network operator's perspective, revenue maximization multicommodity flow models could be used in the design of virtual private networks (VPNs) which are carved out from the capacity in their core network - see [12] for work on general networks.

Our present purpose is to give a simple, efficient, combinatorial algorithm for the minimum cost multicommodity flow in a ring network. It is easily implemented and is the only known algorithmic alternative to a general linear programming approach for this problem. We also give a polynomial-time incarnation of our algorithm. To do this, we adopt a minimum mean

cost augmenting cycle approach of Goldberg and Tarjan [7]. In fact, we generalize this in the sense that we augment along minimum cost pairs of augmenting cycles for distinct commodities. (Note that in the ring, any augmenting cycle is of length n and so 'mean' loses its meaning.) Our algorithm resembles the classical cycle-cancelling algorithm of Klein for network flows. In particular, it uses (as in single commodity flows) "pivots" which are determined by a value $\epsilon > 0$ and two subsets of variables M, P such that the variables in P are raised by ϵ and the variables in M are decreased by ϵ.

It woud be interesting to examine more generally which LP's have this latter property. Namely, we would like to know when a region $P = \{x \in R^n_+ : A \cdot x \leq b\}$ has the property that for any adjacent vertices x, y of P we have that vector $|x - y|$ is $0, \beta$-valued for some $\beta > 0$. More specifically, can we characterize the $0, 1, -1$ matrices A with the following *Klein* property: for each pair $M, P \subseteq \{1, 2, \ldots, n\}$ of disjoint subsets, the cone

$$\{x \in \mathbb{R}^n : A \cdot x = 0, x_i \leq 0 \ \forall i \in M, x_i \geq 0 \ \forall i \in P\}$$

is generated by $0, 1, -1$ vectors. Many Klein matrices arise in combinatorial optimization, perhaps most notably the node-arc incidence matrix of any directed graph has this property. The present paper describes a different class of matrices with the Klein property.

2 Multicommodity Flows in Ring Networks

2.1 Feasibility

In the present paper, our supply graph is a ring (sometimes called a cycle) $G = (V, E)$ with $V = \{0, 1, \ldots, n-1\}$ and $E = \{\alpha_0, \ldots, \alpha_{n-1}\}$ where $\alpha_i = i(i+1)$, $i = 0, \ldots, n-1$ (arithmetic is modulo n). We let $P(i, j)$ denote the path $i, i+1, \ldots, j$ in G.

There has been considerable algorithmic work in recent years on a variety of ring routing problems, e.g., [4, 6, 9, 13, 14, 16, 18]. A common method of attack is to consider the LP relaxation of the routing problem, that is, we study multicommodity flows on the ring.

A demand edge *crosses* an edge-cut $\{\alpha_i, \alpha_j\}$ if one endnode of the demand is in $j+1, j+2, \ldots, i$ and the other endnode is in $i+1, i+2, \ldots, j$. Let $L(\alpha_i, \alpha_j)$ denote the sum of the demands that cross the edge-cut $\{\alpha_i, \alpha_j\}$; this is also called the *load* of cut $\{\alpha_i, \alpha_j\}$. A Theorem of Okamura and Seymour [13], when specialized to a ring G, asserts that there is a flow satisfying the edge-capacity constraints if and only if the edge-cut condition holds, that is,

$$u(\alpha_i) + u(\alpha_j) \geq L(\alpha_i, \alpha_j) \text{ for every } 0 \leq i < j < n \ . \tag{3}$$

Moreover, if u is integer-valued then there is a half-integral routing x. Schrijver, Seymour and Winkler gave an efficient combinatorial algorithm (cf. Proposition 4.1 in [14]) to obtain such a feasible flow if these conditions are satisfied. Thus we will assume that we start with a feasible initial flow.

2.2 A Minimum Cost Multicommodity Flow Algorithm

We now turn to finding a minimum cost multicommodity flow.

The Cost Function

We assume that we are given the ring G along with integral edge-capacities $u(\alpha)$. We also have a demand H, h which is equipped with path cost functions - one for each demand edge - for G. That is, for each $f \in E(H)$, there is a function w^f which assigns to a path P, a cost $w^f(P)$. (For example, these costs may arise from edge costs w^f_a in which case $w^f(P) = \sum_{a \in E(P)} w^f_a$.) Evidently we will only care about the f-cost of a path P if it joins the two endpoints of f. Since G is a ring, if H is a simple demand graph, we may assume that each path P has a unique cost which we denote by $w(P)$. (We assume H is simple, but note that non-simple demand graphs with varying commodity cost functions for the multiple demand edges may also be handled by these methods.) We require an extra condition on the path cost function w. Let $w(ij)$ be shorthand for $w(P(i,j))$. A path cost function w is *normal*, if for any nodes i, j, k, l appearing clockwise in this order on the ring we have:

$$w(ij) + w(kl) \leq w(ji) + w(lk),$$

For a normal cost function w, we now consider the multicommodity flow problem:

$$\begin{array}{l} \min\{\sum_{P \in \mathcal{P}} w(P)x(P) : \\ \quad x \geq 0 \text{ and satisfies the constraints (1),(2)}\}. \end{array} \tag{4}$$

Our aim is to describe a simple combinatorial algorithm, which given a feasible solution to (4), will produce a $\frac{1}{2}$-integral optimal solution (in case w is normal).

Algorithm Description

Let $x : \mathcal{P} \to \mathbb{R}$ be any feasible solution for (4), i.e., a fractional routing which fulfils the demand and satisfies the edge-capacities. We will define several operations which may be applied to such an x and which improve its cost. We then show, in Theorem 4, that if there is no more local improvements available to us, then the solution is optimal.

The *load* $l(\alpha)$ of edge α under x is the value $\sum_{P:\alpha \in E(P)} x(P)$. The *residual capacity* of an edge α is the value $u(\alpha) - l(\alpha)$. An edge α is *tight* if its residual capacity is zero. We say that a demand edge ij is *split* under x if both $x(ij)$ and $x(ji)$ are positive, where $x(ij)$ is the shorthand for the value $x(P(i,j))$.

We use (i,j) to denote a *directed arc* from i to j, and we identify the collection of directed arcs $\mathcal{P}(x) = \{(i,j) : x(ij) > 0\}$ with the support of the flow x. The *quality* of an arc $e = (i,j)$, denoted by $q(e)$, is the value $w(ji) - w(ij)$. Note that if $q(e) < 0$, then there is incentive to increase the flow on the path $P(j,i)$ which we call e's *positive section*, denoted by $pos(e)$, thus decreasing the flow on $P(i,j)$, or e's *negative section* denoted $neg(e)$. An arc $(i,j) \in \mathcal{P}(x)$ is *maximal* if there is no other arc $(k,l) \in \mathcal{P}(x)$ for which $P(i,j) \subset P(k,l)$. Let $\mathcal{M}(x)$ be the set of maximal arcs. A pair of arcs (k,ℓ) and (i,j) are *non-crossing* if k and ℓ are both in $\{i, i+1, i+2, \ldots, j-1, j\}$ or both in $\{j, j+1, \ldots, i\}$. Otherwise, (k,ℓ) and (i,j) are *crossing*. If e denotes a directed arc (i,j), we use $\bar{e}$ to denote the arc (j,i).

The algorithm will ultimately produce a certificate of optimality based on the following conditions derived from complementary slackness conditions (cf. [1]).

Proposition 1 *A feasible solution x for (4) is optimal if and only if there exists an edge weight function $z : E(G) \to \mathbb{R}_+$ such that*

- $z_\alpha > 0$ *implies that* α *is tight under* x.
- $x(ij) > 0$ *implies that* $w(ij) + z(ij) \leq w(ji) + z(ji)$.

This amounts to altering the costs of paths (incrementing by the values z_α) so that we only route flow along shortest paths under costs $w + z$. We call such a z, a *certificate of optimality* for x. The arcs (i, j) for which $q(ij) < 0$ identify precisely where this condition is violated when $z = 0$ and so we attack these edges first.

Swapping and Augmenting: For a multicommodity flow x, pair of arcs $e = (i, j), f = (k, \ell) \in \mathcal{P}(x)$ and $\epsilon > 0$, we denote by $Swap(x, e, f, \epsilon)$ the new routing obtained from x by decreasing the values $x(ij), x(kl)$ by ϵ and increasing the values of $x(ji), x(lk)$ by ϵ. Note that this new routing may or may not be feasible. The cost of the operation, denoted by $C(e, f) \equiv q(e)+q(f)$, is the increase in cost of the new vector obtained. For such a pair of arcs, their *positive intersection* denoted by $pos(e, f)$ is $pos(e) \cap pos(f)$; their *negative intersection* $neg(e, f) = neg(e) \cap neg(f)$. Note that if $pos(e, f)$ contains no tight edge, then there is an $\epsilon > 0$ for which $Swap(x, e, f, \epsilon)$ is feasible. In particular, define ϵ to be the minimum of (i) one half the minimum residual capacity of an edge in $pos(e, f)$ (equal to ∞ if the positive section is empty) and (ii) the minimum of $x(ij), x(k\ell)$. Then we let $Augment(x, e, f) \equiv Swap(x, e, f, \epsilon)$ and note that this is a new, distinct, feasible routing.

Opposing Pairs: For a pair of non-crossing arcs $e, f \in \mathcal{P}(x)$, there are four possible configurations as shown in Figure 1. The pair in configuration (a), having an empty positive intersection, is called an *opposing pair*. In this case, $Augment(x, e, f)$ produces a feasible routing of cost no worse than x, since $q(e) + q(f) \leq 0$ due to normality. In configuration (b), we have $q(e) + q(f) \geq 0$ due to normality. In configurations (c) and (d), either e or f is not maximal. This operation was first devised in [14] (reprinted in [15]).

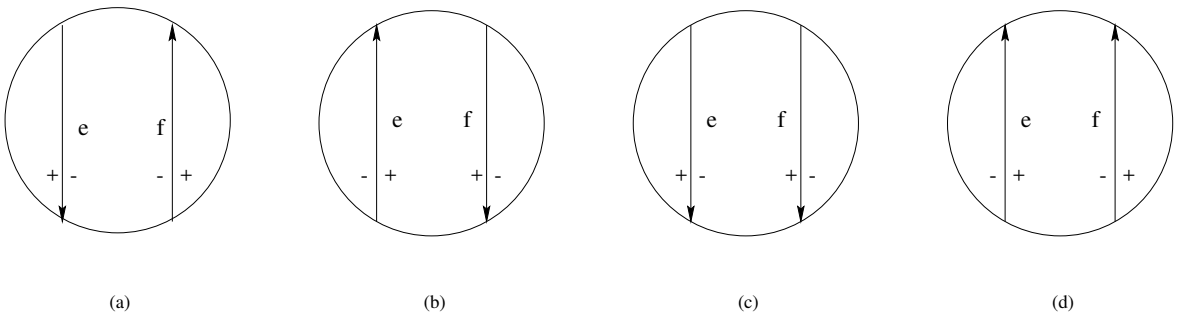

Figure 1: *Four possible configurations for two parallel arcs* e *and* f. *The pair in configuration* (a) *is opposing.*

Twisted Pairs: A pair of crossing arcs $e, f \in \mathcal{P}(x)$ is called twisted if there is no tight edge in $pos(e, f)$. In this case, $Augment(x, e, f)$ (i) increases the load of each edge in $pos(e, f)$ by 2ϵ, (ii) decreases the load of each edge in $neg(e, f)$ by 2ϵ and (iii) leaves the load of each other edge unchanged. A *negative pair* is either a twisted pair e, f for which $C(e, f) = q(e) + q(f)$ is negative or a pair e, e for which $e \in \mathcal{P}(x)$, $pos(e)$ has no tight edge and such that $q(e) < 0$.

We are now ready to describe our algorithm which consists of 3 routines, PACIFY, CLEANSE and MCMRR. The first routine, PACIFY, operates on opposing pairs only and turns a routing x into one with no opposing pairs. It is easy to verify that future operations do not introduce any opposing pairs. (See Fact 2.1.) The second routine, CLEANSE, turns the output of PACIFY into a 1/2-integral routing. (See Theorem 3.) We also show that future operations maintain 1/2-integrality. Finally, the third and main routine, MCMRR, produces an optimal solution. (See Theorem 4.)

In Section 2.3, we prove that all three routines can be made to terminate in polynomial time.

PACIFY(X)
$x :=$ a feasible routing
While ($\exists$ an opposing pair $e, f \in \mathcal{M}(x)$)
    $x := Augment(x, e, f)$
    Update $\mathcal{P}(x)$ and $\mathcal{M}(x)$
Endwhile
Output(x)

CLEANSE(X)
$x :=$ a feasible routing
$x :=$ PACIFY(x)
[Init]
$m :=$ number of split demands
If $m = 0$ Output(x)
    Identify the split demands arc set:
        $\{g_i = (v_i, v_{i+m})\}_{i=0}^{m-1}$
    where nodes appear clockwise as
        $v_0, v_1, \ldots, v_{2m-1}$
For ($i = 0, ..., m-1$)
    Choose e, f as either $\bar{g}_i, g_{i+1}$ or $g_i, \bar{g}_{i+1}$,
        so that $C(e, f) < 0$
    If (e, f) is twisted {
        $x := Augment(x, e, f)$
        If g_i or g_{i+1} becomes unsplit
            Goto *[Init]*
    }
EndFor
Output(x)

MINIMUM COST MULTICOMMODITY
RING ROUTING: MCMRR(X)
$x :=$ `a feasible routing`
$x :=$ PACIFY(x)
$x :=$ CLEANSE(x)
`While (`$\exists$ `a negative pair` $e, f \in \mathcal{M}(x)$`)`
    $x := Augment(x, e, f)$
    `Update` $\mathcal{P}(x)$ `and` $\mathcal{M}(x)$
`EndWhile`
`Output(`x`)`

Proof of Correctness

It is easy to see that all the operations in these three routine produce feasible routings. This is due to the choice of the arcs and the ϵ-value for the *Augment* operation.

We first summarize some properties of opposing pairs and the routine PACIFY. These observations are easy to verify, but they are also critical in the proofs later.

Fact 2 *If there is no opposing pair in $\mathcal{M}(x)$, then*

1. *There is no opposing pair in $\mathcal{P}(x)$. Furthermore, there is no opposing pair in $Augment(x, e, f)$ for any negative pair $e, f \in \mathcal{M}(x)$; i.e.* CLEANSE *and* MCMRR *never introduce opposing pairs.*
2. *If e corresponds to a split demand, then $e, \bar{e} \in \mathcal{M}(x)$.*
3. *If $e, f \in \mathcal{M}(x)$ and $q(e) + q(f) < 0$, then e and f are crossing.*

We now prove several facts about the routine CLEANSE. The proof follows closely to those in [14] for the $\{0, 1\}$ case.

Theorem 3 *The routine* CLEANSE *produces a solution x to (4) such that*

1. *x is half-integral*
2. *each edge in $E(G)$ has integral load*
3. *if we increase each edge-capacity by one, then there is an integral solution to (4) with cost at most that of x.*

Proof: By Fact 2.2, we have that the split demands may be described as set out in the Initialize step of the algorithm. If $m = 0$, then there is nothing to prove. If $m = 1$, then the while loop is performed only once and this turns the flow on each of $x(v_0v_1)$ and $x(v_1v_0)$ into an integer.

So we assume that $m \geq 2$. When the algorithm terminates, for each i, the pair (e, f) as defined is not twisted. Thus either $P(v_i, v_{i+1})$ or $P(v_{i+m+1}, v_{i+m+1})$ contains a tight edge. In fact, in either case we may deduce that for each i, the path $P(v_i, v_{i+1})$ contains

an edge f_i which has an integral load under x. Thus the load of each f_i under the path assignments for the split demands alone, is also equal to some integer $l(i)$. But note that $l(i) = l(i-1) + x(v_i v_{i+m}) - x(v_{i+m} v_i)$. Thus since $x(v_i v_{i+m}) + x(v_{i+m} v_i) = h(ij)$ is an integer we have that $2x(v_i v_{i+m}) = l(i) - l(i-1) + h(ij)$ and so $x(v_i v_{i+m})$ is half-integral.

Finally let x' be the new solution obtained from x as follows. For each $i \in \{0, 1, \ldots, m-1\}$ increase $x(v_i v_{i+m})$ by $(-1)^i/2$ and $x(v_{i+m} v_i)$ by $(-1)^{i+1}/2$. The load of any edge under this new solution is changed by at most one. One now sees that either x' or $2x - x'$ has cost at most that of x, and the proof is complete. □

Finally, we show that MCMRR maintains 1/2-integrality and produces an optimal solution to (4).

Theorem 4 MCMRR *produces a 1/2-integral and optimal fractional routing x. Moreover, there is a polytime algorithm which finds an optimality certificate z satisfying the conditions of Proposition 1 for x, w.*

Proof: By Theorem 3, CLEANSE produces a 1/2-integral where all edge loads are integral. Hence, when executing MCMRR the ϵ value for every $Augment(x, e, f)$ operation must be half-integral. Hence, from then on the routing remains 1/2-integral and the edge load remains integral.

It remains to show that MCMRR produces a routing of minimum cost. Fact 2.1 shows that after calling PACIFY, the routings will never have opposing pairs. For the remainder we assume that x is a feasible solution to (4) which has no opposing pair. If every arc $e \in \mathcal{P}(x)$ satisfies $q(e) \geq 0$, then by Proposition 1, x is optimal by taking z to be 0, so we also assume this is not the case.

Auxiliary Digraphs: We now build an auxiliary digraph $D(x) = (V, A)$ with arc costs c' as follows. For each $(i, j) \in \mathcal{M}(x)$ we include the arc (i, j) with cost $q(ij)$. These are called the *augment arcs*. Note that by maximality, each node has in-degree and out-degree at most one. Also, for any non-tight edge $\alpha_i \in E(G)$, include the *clockwise dummy arc* $(i, i+1)$ with cost 0.

We now add some *counter-clockwise dummy arcs*. Consider a pair of crossing augment arcs e, f. If the positive section $pos(e, f)$ contains no tight edge, then we include the counter-clockwise arcs which are parallel to the edges of $neg(e, f)$. Again, these arcs all have cost 0.

The following is a key structural lemma for the resulting graph $D(x)$.

Lemma 5 *Let $e, f \in \mathcal{M}(x)$ be a pair of crossing arcs. If $D(x)$ contains a counter-clockwise path parallel to $neg(e, f)$, then $pos(e, f)$ contains no tight edges.*

Proof: By construction, the counter-clockwise arcs parallel to the edges of $neg(e, f)$ must have been included by consideration of the negative intersections from a collection of crossing pairs of arcs in $\mathcal{M}(x)$. That is, there exists $(x_1, y_1), (x_2, y_2), \ldots, (x_t, y_t)$ such that (1) for each i, x_i, y_i is a pair of crossing arcs in $\mathcal{M}(x)$ (2) for each i, $pos(x_i, y_i)$ has no tight edge and (3) the union of the negative intersections of these pairs includes $neg(e, f)$.

Without loss of generality $e = (a, b), f = (c, d)$ where $neg(e, f) = P(c, b)$, i.e., the nodes appear in the clockwise order a, c, b, d. For each i let $x_i = (a_i, b_i)$ and $y_i = (c_i, d_i)$ where the nodes appear as a_i, c_i, b_i, d_i in clockwise order. Thus $neg(x_i, y_i) = P(c_i, b_i)$ and $pos(x_i, y_i) =$

$P(d_i, a_i)$. Note that the c_i's are distinct by maximality, as are the b_i's. We may assume that the pairs are numbered so that we have a clockwise order on the nodes $c_t, c, c_{t-1}, \ldots, c_1, b$. Moreover, we may assume that for each i, we visit in clockwise order the nodes $c_{i+1}, c_i, b_{i+1}, b_i$ (otherwise one of the pairs is redundant). Hence we also have the following clockwise order on the nodes: $c, b_t, b_{t-1}, \ldots, b_2, b, b_1$. It is then enough to show (i) $P(d_t, a_1)$ contains $P(d, a)$ and (ii) the union of the positive intersections is connected.

To see (i), note that $a_1 \in P(a, b)$, otherwise $P(a, b) \subset P(a_1, b_1)$. Similarly $d_t \in P(c, d)$. To see (ii), we show that the union of two consecutive positive intersections is connected. This follows since $x_{i+1} = (a_{i+1}, b_{i+1})$ must cross $y_i = (c_i, d_i)$ since there is no opposing pair, and since x_{i+1} crosses y_{i+1} by definition. □

We next show that if there is a negative circuit, then there is one arising from a pair.

Lemma 6 *If $D(x)$ contains a negative circuit, then there is a negative pair in $\mathcal{M}(x)$.*

Proof: Suppose $C = a_0, P_0, a_1, P_1, \ldots, a_{r-1}, P_{r-1}$ is a simple directed cycle of negative cost in $D(x)$, where each a_i is an augment arc and each P_i is a directed path of dummy arcs joining the head of a_i to the tail of a_{i+1} (subscripts will be modulo r). First suppose that $r = 1$ and $a_0 = (i, j)$. If P_0 is clockwise, then clearly $P(j, i)$ contains no tight edge. If P_0 is counter-clockwise, then Lemma 5 also implies that $P(j, i)$ contains no tight edge. Thus a_0, a_0 is a negative pair. So assume that $r \geq 2$.

Since $\sum_{i=0}^{r}(c'(a_i) + c'(a_{i+1}))$ is twice the cost of C, there must be some consecutive pair of augment arcs a_i, a_{i+1} such that $c'(a_i) + c'(a_{i+1}) < 0$. Let $a_i = (a, b), a_{i+1} = (c, d)$. By Fact 2.3, these two arcs are crossing. If P_i is a clockwise path, then clearly this is a negative pair. If P_i is counterclockwise, then the nodes appear as a, c, b, d around the ring and the arcs of P_i are parallel to the edges of $neg(a_i, a_{i+1})$. But then by Lemma 5, $pos(a_i, a_{i+1})$ has no tight edge and hence a_i, a_{i+1} is again a negative pair. This completes the proof. □

We now complete the proof of correctness.

Lemma 7 *If x is a feasible routing which has no opposing pair and $D(x)$ has no negative cycle, then x is an optimal solution to (4).*

Proof: Let x be a feasible routing with no opposing pair and for which $D(x)$ has no negative circuit. We may then find, for instance by the Bellman-Ford algorithm (cf. [3]), a feasible potential π for the digraph $D(x)$. Thus for each arc $a = (i, j)$ of $D(x)$ we have $\pi(j) \leq \pi(i) + c'(ij)$. From π we produce a vector of edge weights $z' : E(G) \to \mathbb{Q}$ as follows. For each $i \in V$, set $z'_{\alpha_i} = (\pi(i+1) - \pi(i))/2$. Note that for any $i \neq j$, we have

$$z'(ij) - z'(ji) = \pi(j) - \pi(i),$$

since $z'(ji) = \sum_{k=j}^{i-1}(\pi(k+1) - \pi(k))/2 = (\pi(i) - \pi(j))/2$. We now simulate z' by a nonnegative vector z with the property that $z_e > 0$ implies that e is tight under x. First if some $z'_{\alpha_i} > 0$, then clearly the dummy arc $(i, i+1)$ was not in $D(x)$ and hence e_i was tight under x. For any such arc we define $z_{\alpha_i} = z'_{\alpha_i}$. For any z_e not yet defined, we currently set $z_e = 0$; these values may be increased by the following procedure.

First, for each edge α_i we define an edge α'_i as follows. We consider all arcs $(k, j) \in \mathcal{M}(x)$ such that $P(i, j)$ contains α_i. (If there is no such arc, then α'_i is undefined.) Let (k_1, j_1) be

such an arc for which $P(k_1, i)$ is of maximum length, and (k_2, j_2) be such an arc for which $P(i+1, j_2)$ is of maximum length. Note that by maximality of the arcs, we have that the set of edges of $P(j_2, k_1)$ intersects neither $P(k_1, j_1)$ nor $P(k_2, j_2)$. If $P(j_2, k_1)$ has no tight edge, then α'_i is undefined. Otherwise, define α'_i to be one of the tight edges in this path.

Now suppose some $z'_{\alpha_i} < 0$. Then clearly $(i+1, i)$ was not an arc in $D(x)$ and so by definition of $D(x)$ and the (k_t, j_t)'s just discussed, α'_i must be defined. We thus set $z_{\alpha'_i} = z_{\alpha'_i} - z'_{\alpha_i}$.

Note that z satisfies the first condition of Proposition 1. We now show that it also satisfies the second optimality condition and hence x is optimal.

We first show that for any $(i, j) \in \mathcal{M}(x)$ we have that $z(ij) - z(ji) \leq z'(ij) - z'(ji)$. If this were not the case, then there was some arc $e_l \in P(i, j)$ with $z'_{e_l} < 0$ for which e'_l is also in $P(i, j)$. But the definition of e'_l guarantees precisely that $e'_l \notin P(i, j)$.

Since π is a potential in $D(x)$ we have that if $(i, j) \in \mathcal{M}(x)$, then $z(ij) - z(ji) = \pi(j) - \pi(i) \leq q(ij) = w(ji) - w(ij)$. Thus $w(ij) - w(ji) + z(ij) - z(ji) \leq w(ij) - w(ji) + z'(ij) - z'(ji) \leq 0$. Thus the second condition of Proposition 1 is satisfied for z and each $(i, j) \in \mathcal{M}(x)$.

Finally consider some $x(ij) > 0$ such that $(i, j) \notin \mathcal{M}(x)$. Then there exists some $kl \in \mathcal{M}(x)$ such that $P(i, j) \subseteq P(k, l)$. Since $z \geq 0$ we have $z(ij) - z(ji) \leq z(kl) - z(lk)$. Hence $w(ij) - w(ji) + z(ij) - z(ji) \leq w(ij) - w(ji) + z(kl) - z(lk)$. Normality now implies that this is at most $w(kl) - w(lk) + z(kl) - z(lk) \leq 0$ as required.

Thus x, w, z satisfy the optimality conditions from Proposition 1 and hence x is optimal. Note that we have thus described a polytime algorithm to produce the desired optimality certificate z. □

End Proof of Theorem 4.

2.3 A Version with Polynomial Running Time

We address the running time of each of the three routines in turn.

PACIFY:

During the execution of PACIFY, let $e, f \in \mathcal{M}(x)$ be an opposing pair for augmentation and let x' be the resulting routing $Augment(x, e, f)$. Without loss of generality, we assume that $e \notin \mathcal{P}(x')$. We show that from then on, $\bar{e}$ cannot participate in any augment operation in PACIFY. For $\bar{e}$ to be in an opposing pair, we need an arc g such that $neg(e) \subset neg(g)$. Such an arc g cannot be in $\mathcal{P}(x)$ due to the maximality of e under x. To reintroduce the arc g later, $\bar{g}$ must be maximal under some future routing. This is impossible since $neg(\bar{g}) \subset neg(\bar{e})$.

Therefore, after each augmentation the arcs for at least one demand "retire" from PACIFY. Hence, PACIFY terminates in polynomial time.

CLEANSE:

The routine CLEANSE terminates in polynomial time since any time we return to [*Init*], at least one split demand has disappeared.

MCMRR:

It is easy to see that MCMRR terminates in finite time. For each augmentation in MCMRR, the total routing cost is reduced by at least $2\epsilon \cdot \min_{e,f:C(e,f)<0} |C(e, f)|$. In Theorem 4, we have shown that ϵ is 1/2-integral. Hence, the routing cost is reduced by at least $\min_{e,f:C(e,f)<0} |C(e, f)|$.

To show that MCMRR terminates in polynomial time, we first assume for simplicity that the costs $C(e,f)$ are distinct for all arc pairs e,f with $C(e,f) < 0$. For each $Augment(x,e,f)$ operation in MCMRR, we choose the negative pair $e,f \in \mathcal{M}(x)$ that has the most negative cost $C(e,f)$. Note that this arc pair e,f is unique. Let $\mu(x)$ denote this cost $C(e,f)$. We show in Theorem 8 that either $\mu(x)$ increases or some demand edge *retires*, i.e., its corresponding arcs do not participate in any future augmentation. If no demand edge retires, each arc pair can only participate in the augment operation once due to the strict monotonicity of $\mu(x)$. Polynomial running time of MCMRR follows.

Theorem 8 *Suppose the $C(e,f)$'s are distinct for arc pairs e,f with $C(e,f) < 0$. If* MMCR *always chooses the most negative cost augmentation, then after each augmentation either $\mu(x)$ has increased or some demand edge retires.*

Proof: Suppose x is a solution and $e_1, e_2 \in \mathcal{M}(x)$ is a twisted pair under x such that $C(e_1,e_2) = \mu(x)$. Let x' be the resulting solution of $Augment(x,e_1,e_2)$. For the purpose of contradiction, suppose that there exists a twisted pair $e_3, e_4 \in \mathcal{M}(x')$ such that $C(e_3,e_4) < C(e_1,e_2)$. We begin with the following observations.

Lemma 9 *Let $i \in \{1,2\}$ and $j \in \{3,4\}$. If e_i and e_j are non-crossing, then $pos(e_i) \subset pos(e_j)$ and $q_i \leq q_j$.*

Proof: We prove this for $i = 1$ and $j = 3$; note that the desired configuration is given by Figure 1c with e set to e_1 and f set to e_3. We show that all three other configurations are not possible. If $pos(e_1,e_3)$ is empty as in Figure 1a, then e_1 and e_3 are opposing under x, contradicting Fact 2.1. If $neg(e_1,e_3)$ is empty as in Figure 1b, then e_3 is not maximal under x' due to $\bar{e}_1$. If $neg(e_1) \subset neg(e_3)$ as in Figure 1d, then $\bar{e}_1$ and e_3 are opposing under x'. The fact that $q_1 \leq q_3$ follows from normality. □

Case 1: e_1 and e_3 are non-crossing.
We have $q_3 \geq q_1$ by Lemma 9. Since $C(e_3,e_4) < C(e_1,c_2)$, we also have $q_4 < q_2$. This implies e_2 crosses e_4 again by Lemma 9. If e_1 crosses e_4, since $C(e_1,e_4) < C(e_1,e_2)$ and $Augment(x,e_1,e_2)$ is performed instead of $Augment(x,e_1,e_4)$, there must be a tight edge $f \in pos(e_1,e_4) \setminus pos(e_1,e_2)$ under x. By Lemma 9, $pos(e_1) \subset pos(e_3)$. Hence, f must be in $pos(e_3,e_4)$. Since f remains tight under x', this leads to a contradiction. We therefore conclude e_1 does not cross e_4.

To summarize the configuration under x, we have e_1 does not cross e_3 or e_4, and e_2 crosses both e_3 and e_4. (See Figure 2.) We also have $q_1 \leq q_3, q_4 < q_2$. We shall prove in Lemma 10 that in this case the demand corresponding to arc e_1 retires from now on.

The cases in which e_1, e_4 are non-crossing, or e_2, e_3 are non-crossing, or e_2, e_4 are non-crossing are similar to Case 1. Hence, in the remaining analysis, all arcs e_1, e_2, e_3 and e_4 cross one another.

Case 2a: e_1, e_3 and e_2, e_4 are both twisted pairs under x.
Then $C(e_1,e_3) \geq C(e_1,e_2)$ and $C(e_2,e_4) \geq C(e_1,e_2)$. Since $C(e_1,e_2)+C(e_3,e_4) = C(e_1,e_3)+C(e_2,e_4) \geq 2C(e_1,e_2)$, we deduce $C(e_3,e_4) \geq C(e_1,e_2)$, a contradiction.

Case 2b: e_1, e_3 are not twisted. (The case where e_2, e_4 are not twisted is similar.)

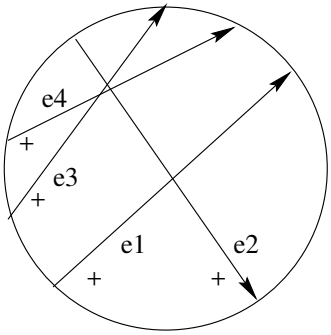

Figure 2: *Case 1 of Theorem 8. Under this configuration, the demand corresponding to* e_1 *retires after the operation* $Augment(x, e_1, e_2)$.

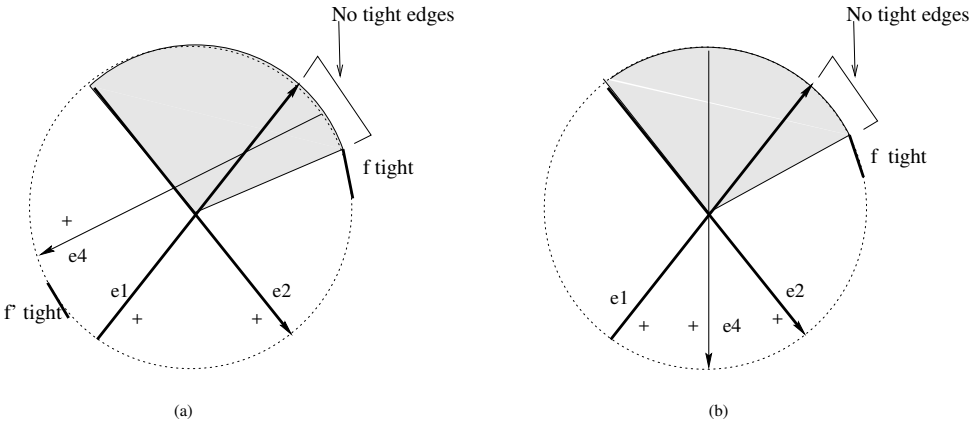

Figure 3: *Case 2b of Theorem 8.*

Since e_1 and e_3 are not twisted, there exists a tight edge f in $pos(e_1, e_3) \setminus pos(e_1, e_2)$ - see Figure 3a (or 3b). Since all pairs of arcs cross, $f \notin pos(e_4)$ (since f is still tight under x'), and since $pos(e_3, e_4)$ overlaps with $neg(e_1, e_2)$, the shadowed areas indicate the only places where e_4 could have an endpoint. There are two cases depending on whether e_4 has an endpoint in $neg(e_1, e_2)$ - Figure 3a, or not - Figure 3b. Note that in either case, the pair e_1, e_4 are twisted.

Next, note that if e_2, e_3 is twisted under x, then by relabelling we are in Case 2a, so we assume this is not the case. Thus there is a tight edge $f' \in pos(e_2, e_3)$. This edge cannot lie in $pos(e_4)$ for it would also be tight under x'. Since $pos(e_1, e_2)$ contains no tight edges, we have that $f' \in pos(e_2) \setminus (pos(e_4) \cup pos(e_1))$. Thus Figure 3b is impossible, and hence f' must lie as depicted in Figure 3a. This contradicts the fact that e_3 crosses e_4 but $pos(e_3)$ contains f, f' as well as an edge of $neg(e_1, e_2)$.

Lemma 10 *Under the configuration of Case 1, the demand corresponding to* e_1 *retires after*

operation Augment(x, e_1, e_2).

Proof: Let e_1, e_2, e_3, e_4 be as in Case 1. Note that $x'(e_1) = 0$ due to the maximality of e_3 and e_4. It suffices to show that no future operation involves arc $\bar{e}_1$. Let $N = \{e : q(e) < q(e_1) < 0\}$.

Case 1: We have $x(e) = 0$ for each $e \in N$. We claim that for each $e \in N$, $y(e) = 0$ under any future routing y. If not, let y' be the first routing after x under which $y'(f_1) > 0$ for some $f_1 \in N$. The operation that leads to y' must be some *Augment*(y'', f_1, f_2), where $q(\bar{f}_1) + q(f_2) < 0$. This implies $y''(f_2) > 0$ and $q(f_2) < q(f_1) < q(e_1)$. Hence, f_2 must be in N, contradicting the choice of y'.

Since $y(e) = 0$ for each $e \in N$ under any future routing y, then $q(\bar{e}_1) + q(f) > 0$ for any arc f with $y(f) > 0$. Hence, no future operation can involve $\bar{e}_1$.

Case 2: If $x(e) > 0$ for some $e \in N$, then let $f \in N \cap \mathcal{P}(x)$ be the arc such that $q(f)$ is minimized. Note that f must be maximal under x. Otherwise, there exists $f' \in \mathcal{P}(x)$ such that $neg(f) \subset neg(f')$. By normality, $q(f') < q(f)$ contradicting the fact that $q(f)$ is minimized over arcs in $N \cap \mathcal{P}(x)$.

Now f and e_1 are both maximal under x and $q(f)$ and $q(e_1)$ are both negative. Since there are no opposing pairs, Fact 2.3 implies that f and e_1 are crossing. Note also $pos(e_1)$ contains no tight edge under x, since $pos(e_1) \subseteq pos(e_3, e_4)$ and a swap on e_3, e_4 are performed under x'. Hence, we should have performed a swap on f and e_1, since $q(f) + q(e_1) < q(e_1) + q(e_2)$. This final contradiction completes the proof. □

End Proof of Theorem 8.

Our analysis is complete with a perturbation lemma. In Lemma 11, the quality $q(e)$ for each arc e is perturbed to $q'(e)$ to ensure the distinctness of $\mu(x)$. Furthermore, if we run our routines under this new cost q', the resulting routing is also optimal under the original cost q.

Lemma 11 *We can perturb the quality $q(e)$ of each arc e to $q'(e)$, such that*

1. *The values $q'(e_1) + q'(e_2)$ are distinct for all pairs e_1, e_2 with $q'(e_1) + q'(e_2) < 0$.*
2. *A routing which is optimal under the cost function q' is also optimal under q.*

Proof: Let δ_1 be $\min\{|C(e_1, e_2) - C(e_3, e_4)| : e_1, e_2$ and e_3, e_4 satisfy $C(e_1, e_2) \neq C(e_3, e_4)\}$. Let δ_2 be $\min |\sum_{e \in S} q(e)|$, minimized over all subsets of arcs S where $\sum_{e \in S} q(e) \neq 0$. Let $\delta > 0$ be $\min\{\delta_1/2, \delta_2/M\}$, where M is the total number of possible arcs. For each demand we pick one of its two corresponding arcs and put these arcs in an arbitrary order, $f_1, f_2, \ldots$. Let $q'(f_i) = q(f_i) + \delta \cdot 3^{-i}$ and $q'(\bar{f}_i) = q(\bar{f}_i) - \delta \cdot 3^{-i}$, for $i = 1, 2, \ldots$.

Let us verify the first property. If $q(e_1) + q(e_2) < q(e_3) + q(e_4)$, then $q'(e_1) + q'(e_2) < q(e_1) + q(e_2) + 2/3 \cdot \delta < q(e_3) + q(e_4) - 2/3 \cdot \delta < q'(e_3) + q'(e_4)$. The first and third inequalities follow from the definition of q', and the second inequality follows from the choices of δ_1 and δ. If $q(e_1) + q(e_2) = q(e_3) + q(e_4)$, then we show that $q'(e_1) + q'(e_2) \neq q'(e_3) + q'(e_4)$ unless (i) e_1, e_2 and e_3, e_4 are the same pair, or (ii) $e_1 = \bar{e}_2$ and $e_3 = \bar{e}_4$. By definition, let $q'(e_j) = q(e_j) + \delta \cdot (a_j 3^{-i_j})$, where $j = 1, 2, 3, 4$, $a_j = \pm 1$, and i_j are positive integers. It is a matter of case analysis to verify if $a_1 3^{-i_1} + a_2 3^{-i_2} \neq 0$ and if $a_1 3^{-i_1} + a_2 3^{-i_2} = a_3 3^{-i_3} + a_4 3^{-i_4}$, then e_1, e_2 and e_3, e_4 must be the same pair.

To verify the second property, let us recall from Lemma 7 that if the auxiliary digraph $D(x)$ does not have negative cycles, then the routing x is optimal. Hence, it suffices to show that if there is no negative cycle under the cost function q', then there is no negative cycle under q. Assume that $D(x)$ has no negative cost cycle under q'. Suppose C is the cost of a cycle under q. By definition of δ, its cost under q' is in $[C - \delta m, C + \delta m]$, where m is the number arcs in the cycle. As $C + \delta m \geq 0$ and $\delta m < |C|$ by definition of δ, we have that $C \geq 0$ as well.

□

References

[1] R.K. Ahuja, T. Magnanti, J. Orlin, *Network Flows* Prentice Hall, Englewood Cliffs, NJ, 1993.

[2] S. J. Clendening, Ring networks to the rescue, *Telephone Engineer and Management* February, (1992) 50-55.

[3] W. Cook, W. Cunningham, W. Pulleyblank, A. Schrijver, *Combinatorial Optimization*, John Wiley and Sons Inc., New York, 1998.

[4] S. Cosares, I. Saniee, An optimization problem related to balancing loads on SONET rings, *Telecommunications Systems*, **3**, (1994) 165-181.

[5] L. R. Ford jr., D. R. Fulkerson, Maximal flow through a network, *Canadian J. Math.*, **8**, (1956) 399-404.

[6] A. Frank, Edge-disjoint paths in planar graphs, *J. Combinatorial Theory B*, **39**, (1985) 164-178.

[7] A. V. Goldberg, R. E. Tarjan, Finding minimum-cost circulations by canceling negative cycles, *Journal of the ACM*, **36**, (1989) 873-886.

[8] T. C. Hu, Multi-commodity network flows, *Operations Research*, **11**, (1963) 344-360.

[9] S. Khanna, A polynomial time approximation scheme for the SONET ring loading problem, *Bell Labs Technical Journal,* Spring, (1997) 36-41.

[10] M. Klein, A primal method for minimal cost flows with application to the assignment and transportation problems, *Management Science*, **14**, (1967) 205-220.

[11] B. Korte, L. Lovász, H.J. Prömel, A. Schrijver Eds., *Paths, Flows and VLSI-Layout*, Springer, Berlin 1990.

[12] D. Mitra, J.A. Morrison, K.G. Ramakrishnan, Virtual private networks: joint resource allocation and routing design, *Proc. IEEE INFOCOM 99*, April (1999) 480-490.

[13] H. Okamura, P. Seymour, Multicommodity flows in planar graphs, *J. Combinatorial Theory B*, **31**, (1981) 75-81.

[14] A. Schrijver, P. Seymour, P. Winkler, The ring loading problem, *SIAM J. Discrete Math.*, **11**, (1) (1998) 1-14.

[15] A. Schrijver, P. Seymour, P. Winkler, The ring loading problem, *SIAM Review*, **41**, (4) (1999) 777-791.

[16] A. Frank, B. Shepherd, V. Tandon, Z. Vegh, Node-capacitated routing in a ring network, *submitted* (1998).

[17] K. Wayne, A polynomial combinatorial algorithm for generalized minimum cost flow, *Proceedings of STOC '99*, (1999) 11-18.

[18] G. Wilfong, P. Winkler, Ring routing in WDM networks, *Proceedings SODA*, (1998) 333-341.