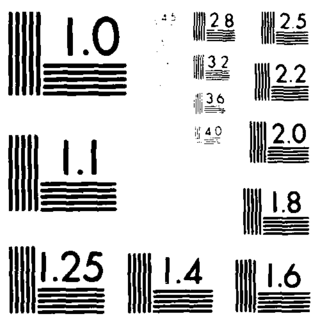
AD-A088 032 STANFORD UNIV CA DEPT OF OPERATIONS RESEARCH F/G 12/1
NEAR-EQUIVALENCE OF NETWORK FLOW ALGORITHMS.(U)
DEC 79 N ZADEH N00014-75-C-0493
UNCLASSIFIED TR-36 NL

1 OF 1
AD A088032

END
DATE FILMED
9-80
DTIC



MICROCOPY RESOLUTION TEST CHART

LEVEL II

AD A088032

NEAR-EQUIVALENCE OF NETWORK FLOW ALGORITHMS

BY

NORMAN ZADEH

TECHNICAL REPORT NO. 36
DECEMBER 1, 1979

PREPARED UNDER
OFFICE OF NAVAL RESEARCH CONTRACT
N00014-75-C-0493 (NR-042-264)

DEPARTMENT OF OPERATIONS RESEARCH
STANFORD UNIVERSITY
STANFORD, CALIFORNIA

NEAR-EQUIVALENCE OF NETWORK FLOW ALGORITHMS*

by

Norman Zadeh

Technical Report No. 36

Dec[redacted]79

Prepared Under
Office of Naval Research Contract N00014-75-C-0493

Department of Operations Research
Stanford University
Stanford, California

*Also partially supported by National Science Foundation Grant ENG 76-12266.

Near-Equivalence of Network Flow Algorithms

Abstract

Network problems arising in practice typically have non-negative arc costs. On such problems we show that the following algorithms perform, modulo ties, the same sequence of flow augmentations:

Simplex (with the standard pivot rule and Big-M start),

Out-of-Kilter (Primal-Dual),

Dual Simplex (with the standard pivot rule),

Lemke's Complementary Pivot Algorithm.

All methods compute a shortest path tree by mimicking the Dijkstra algorithm and then send flow along a sequence of minimum cost paths. Differences in implementation are discussed. It becomes clear that Dantzig's simplex method with the best empirical pivot rule (not the standard rule) will outperform other methods (variations of Simplex with the standard rule). A simple reason is given why Dual Simplex (best empirical) cannot do as well as Simplex (best empirical). It is noted that network flow problems from [18] represent pathological examples for all the above methods.

INTRODUCTION

In most practical transshipment or transportation problems, the initial network has zero flow and non-negative arc costs. Problems with some negative arcs generally arise in sensitivity studies and may be put in non-negative form by a transformation [8] [13].

In this paper we argue that the best known algorithms for solving problems with non-negative costs are basically different implementations of the same general method, which we call the efficient path technique:

1. Start with zero flow.
2. Compute a shortest path tree (or partial tree) using Dijkstra.
3. Send a maximum flow along the shortest s-t path(s) efficiently.
4. Recompute new shortest path(s) using portions of the previous shortest path tree.
5. Go to 3 and repeat steps 3 and 4 until the desired level of flow is reached.

In the context of the simplex method, step 4 corresponds to determining the forward unblocked arc of minimum modified cost in a cut. This arc is then joined to the existing basis tree, yielding a shortest s-t path. Step 3 corresponds to sending flow across that path until an arc is blocked which leaves the basis. The result is a new cut.

To determine the incoming arc, Simplex uses node numbers $\{\pi_i\}$, (π_i represents the distance from ⓢ to ⓘ via the basis tree) and modified costs $\pi_i + d_{ij} - \pi_j$, where d_{ij} is the cost of arc (i,j). The <u>relevant</u> modified costs in Big-M Simplex differ from those in

Out-of-Kilter and Dual Simplex by a constant π_t - M, where π_t is the length of the current shortest s-t path. As a result the methods can "bring in" the same arc. The process of revising costs results in a change in the node numbers and modified costs (dual variables) similar to a dual pivot. By taking $M = \pi_t$, one sees that the node numbers on the sink side of the cut are increased, decreasing the modified costs of forward arcs in the cut until one becomes zero. The same process occurs in the Out-of-Kilter method.

The dual simplex method may be viewed as a procedure for manipulating infeasible primal trees. Its variables are non-negative modified costs and unrestricted node numbers which generate the same modified costs as Out-of-Kilter. The "complement" of the dual basis is a spanning tree containing the current shortest s-t path. To determine its incoming variable, Dual Simplex sends flow across the shortest s-t path until the required level of s-t flow is reached, even though this violates capacities of arcs on the shortest path. The arc whose capacity is most violated (equivalently, the first arc to be blocked) leaves the complement of the dual basis. In other words a dual variable corresponding to that arc is entered, forming (with the other dual basic variables) the same cut as that found by Big-M Simplex and Out-of-Kilter. Potential is increased across the cut until some modified cost goes to zero and leaves the basis. The arc corresponding to that modified cost effectively enters the infeasible primal complement forming a new shortest path.

To summarize, when viewed as a primal method Dual Simplex sends flow along the shortest s-t path until the required level of s-t flow is reached, thus creating an infeasible solution. It does the same operations as Simplex but in the reverse order. Determining the outgoing dual variable is essentially equivalent to determining the incoming primal variable.

Lemke's algorithm and Simplex (most negative) differ only in that Lemke's method uses a simplex pivot in the dual to determine the entering variable in the primal. Basically this dual pivot determines the forward unblocked arc of minimum modified cost in a cut. Simplex and Out-of-Kilter use the same process but it is not formally described as a pivot.

All methods can be made to implement Dijkstra with approximately the same degree of efficiency, although naive implementations might yield $O(n^3)$ procedures for the first shortest path computation [19]. For subsequent shortest path computations, Simplex keeps most of the shortest path tree, whereas the standard version of Out-of-Kilter recomputes it using a max flow labelling procedure.

The relative efficiency of Simplex (most negative) vs. Out-of-Kilter on networks with many paths of equal cost is unclear. Out-of-Kilter essentially computes all shortest paths at once and then uses a labeling method to compute the augmenting paths one at a time. The textbook version of the labeling method (Ford-Fulkerson) throws away all labeling informaion after each augmentation which would seem inefficient.

Simplex computes the shortest paths one at a time and simultaneously does the augmentation (if possible). The best way to implement this process is not known. Also presently unavailable are published results comparing best known implementations of Simplex (most negative) vs. Out-of Kilter. What is known is that other pivot rules (not most negative) are faster than an improved version of Out-of-Kilter [1]. This coincides with empirical results for general linear programs. It apparently doesn't pay to find the "cheapest" arc in a cut when flow will probably be sent on the cheap arc one has already found.

Dual Simplex with the best empirical pivot rule can not do as well as Simplex (best empirical), because regardless of the pivot rule, Dual Simplex must decide which modified cost will go to zero, which is a $O(n^2)$ computation, whereas Simplex (best empirical) can avoid this computation and must decide which arc will become blocked, which is $O(n)$. This observation explains the results of computational studies [3] [11].

Starting methods such as Phase I - Phase II which involve a non-zero starting flow and consequently large numbers of negative cycles seem to be inferior [16] to starts with zero flow such as the Big-M method, at least on general minimum cost flow problems. When started with non-zero flow, equivalently, negative arc costs, we show that Simplex, Out-of-Kilter, and Lemke's method behave differently.

Because of the essential equivalence of Simplex (most negative) and Lemke's algorithm on non-negative network problems, it follows that the examples from [18] are pathological for both.

A minimum cost flow or transshipment problem requires that one send a specified flow v from a source s to a sink t at minimum cost. Lower bounds on the flow in each arc are zero. In [19], we showed that any minimum cost circulation problem with a possibly infeasible starting solution may be simplified to a problem of the above form. In the following, we will assume that arc costs are non-negative unless stated otherwise.

The minimum cost flow problem may be written as

$$\text{minimize } \sum d_{ij} f_{ij}$$

$$\text{subject to } F(N,s) - F(s,N) = -v$$

1) $$F(N,i) - F(i,N) = 0 \qquad i \neq s,t$$

$$F(N,t) - F(t,N) = v$$

$$0 \leq f_{ij} \leq c_{ij}$$

where d_{ij} = cost per unit flow on (i,j)

f_{ij} = flow on (i,j)

c_{ij} = capacity of (i,j)

v = required s-t flow

N = node set, A = arc set, $\mathfrak{N}$ = the network

and

$$F(i,N) = \sum_{j \in N} f_{ij} = \text{flow out of } i ,$$

$$F(N,i) = \sum_{j \in N} f_{ji} = \text{flow into } i .$$

For the sake of simplicity, we will assume unless otherwise stated that there exists at most one arc between any two nodes.

Pivots

When slack variables are added to the L.P. formulation of the minimum cost flow problem and a redundant equation removed, the resulting system has $n - 1 + |A|$ equations in $2|A|$ unknowns, where $|A|$ is the number of arcs, and n is the number of nodes. Any simplex basis may be represented as a union of a) two variables (flow and slack) for each arc in a spanning tree and b) exactly one variable (flow or slack) for every other arc. This representation is hard to find explicitly stated in the literature and was communicated to the author by Richard Stone.

For the purpose of understanding, one can think of the basis as consisting of just those arcs (the tree) with both variables in the basis.

Observation 1: If $0 < f_{ij} < c_{ij}$, then arc (i,j) is automatically in the basis.

To bring a non-basic arc into the basis, one brings in that arc's non-basic variable. This operation forms a unique cycle of basic arcs (see Figure 1). In a non-degenerate pivot, flow is sent around the cycle in the direction determined by the incoming variable. For example, if (i,j) has $f_{ij} = c_{ij}$, then entering s_{ij} (the slack variable) would cause f_{ij} to decrease (flow is sent around the cycle in the direction from j to i). The amount of flow sent is determined by the first slack (residual capacity) or flow to go to zero. The pivot is completed by removing one of the zero variables from the basis, making its associated arc non-basic. In practice, many basic arcs may have flow at upper or lower bounds and degenerate pivots are common.

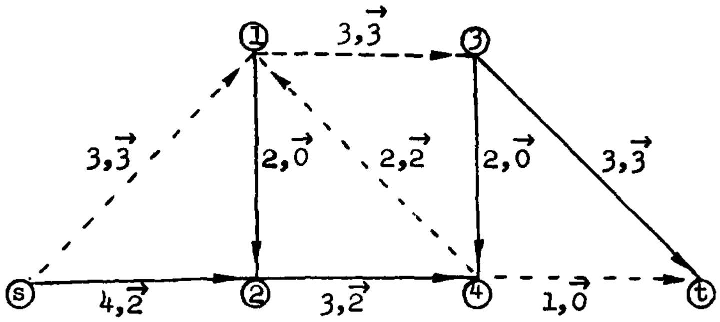


Figure 1a: Example of a basis. Basic arcs are darkened. Numbers on arcs represent capacity and flow.

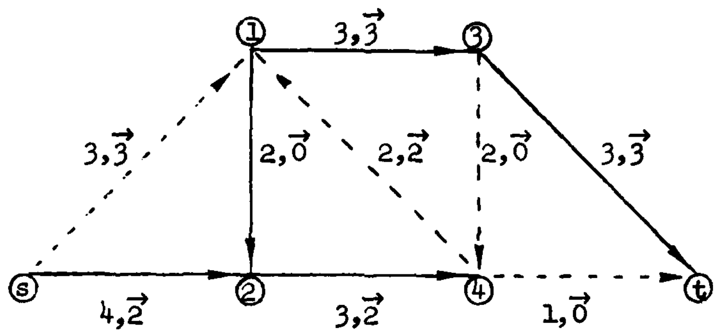


Figure 1b: Resulting basis after the slack variable for arc (1,3) is entered and a degenerate pivot performed which removes arc (3,4).

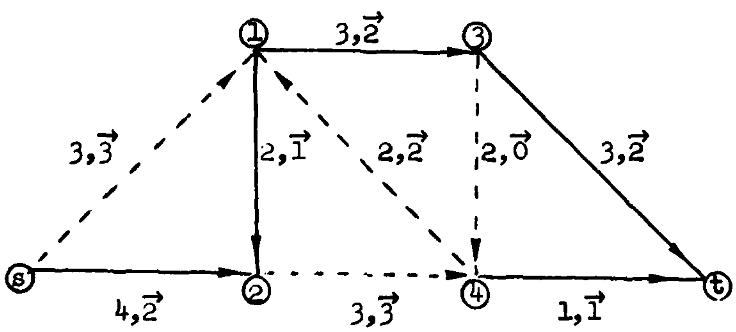


Figure 1c: Resulting basis after the flow variable for arc (4,t) is entered and the slack for (2,4) is removed. This operation sent one unit about the cycle 4t3124. The slack for (4,t) could have been removed instead, leaving the tree of basic arcs unchanged.

Modified Costs and Augmentation Networks

The <u>dual variable</u> π_i may be taken to be the cost of the unique path from s to i in the basis tree, whether this path is blocked or not (see Figure 3).

The <u>modified cost</u> of arc (i,j), denoted by d'_{ij}, is defined to be $\pi_i + d_{ij} - \pi_j$. This is the cost of sending one unit of flow in the direction from i to j about the cycle formed when f_{ij} is entered into the basis, and is equal to the <u>simplex modified cost coefficient</u> for f_{ij}.

If (i,j) is non-basic with flow at capacity, it can only become basic by entering s_{ij}, or equivalently, decreasing f_{ij} and sending flow around the cycle in the opposite direction. The cost of this operation per unit flow is $-(\pi_i + d_{ij} - \pi_j)$, which is the simplex modified cost coefficient for s_{ij}.

The following definition is essential. Let f be a flow in a network $\mathfrak{n}$. Then the <u>augmentation network</u> relative to f, denoted by $\mathfrak{n}^f$, describes the ways in which additional flow may be sent on each arc. $\mathfrak{n}^f$ is constructed as follows:

If $f(i,j) > 0$, put an arc $(j,i) \in \mathfrak{n}^f$ with $c_{ji} = f_{ij}$,
$d_{ji} = -d_{ij}$.

If $f(i,j) < c_{ij}$, put an arc $(i,j) \in \mathfrak{n}^f$ with $c_{ij} = c_{ij} - f_{ij}$,
d_{ij} is unchanged.

All arcs in $\mathfrak{n}^f$ have zero flow. Figure 2 gives an example.

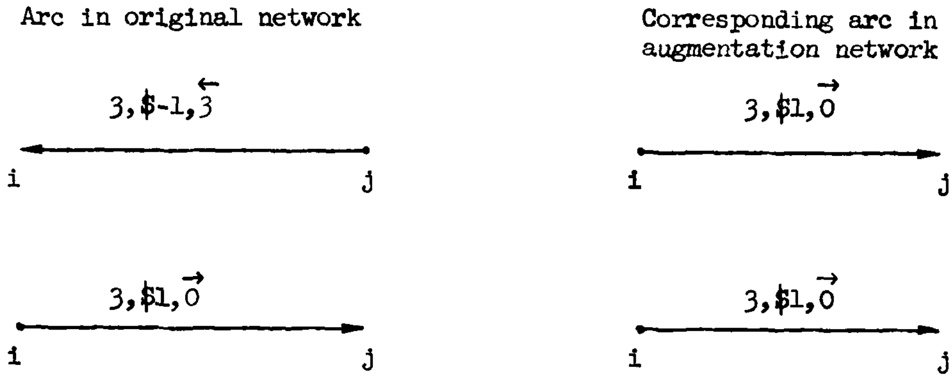


Figure 2: Constructing an Augmentation Network

Observation 2: The simplex modified cost coefficient for $f_{ij}(s_{ji})$ equals the modified cost of (i,j) in h^f. Thus a flow f is of minimum cost iff each arc in h^f has non-negative modified cost.

In Figure 3, costs indicated are modified costs, i.e., $\pi_i + d_{ij} - \pi_j$. The actual cost per unit flow of "entering" a non-basic arc is equal to the modified cost of the associated arc in h^f. Figure 3 shows how node numbers (dual variables) and modified arc costs change as pivots in Figure 1 are performed. Note in Figure 3c that a fraction of the node numbers and modified arc costs must be updated during each pivot.

Figure 3: Node numbers and modified costs associated with the pivots in Figure 1.

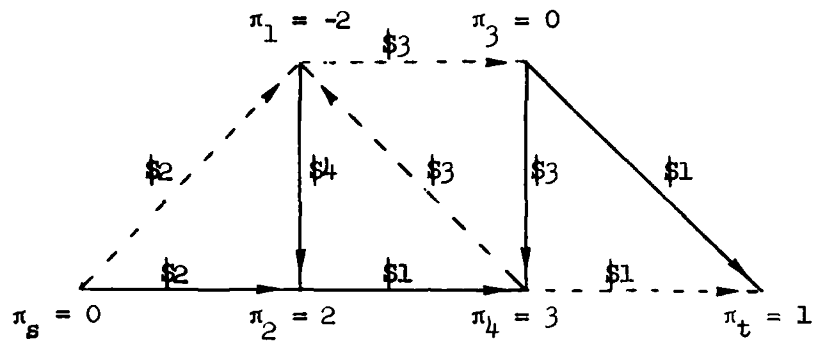


Figure 3a: Initial costs and node numbers (dual variables).

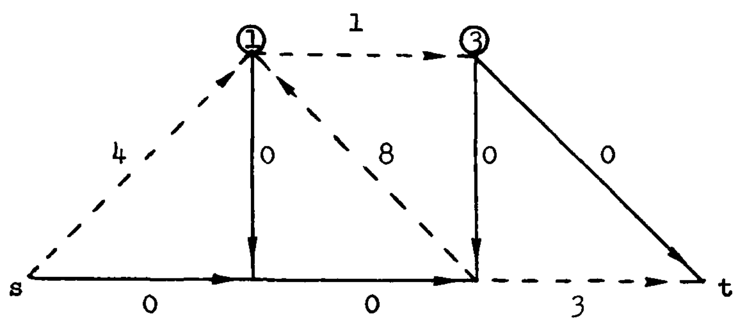


Figure 3b: Modified costs. This basis is not optimal because the modified cost coefficient of s_{13} is -1.

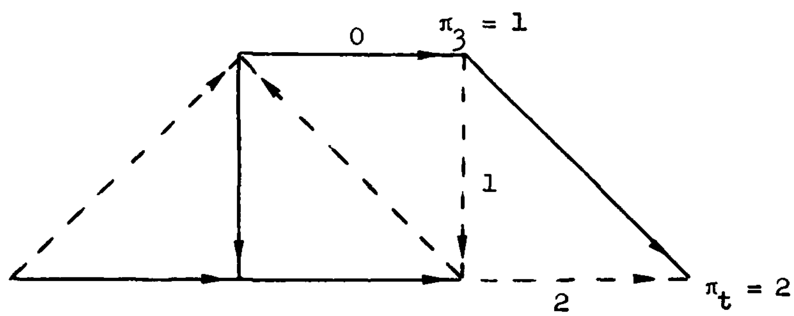


Figure 3c: Modified costs and dual variables when arc (1,3) becomes basic and arc (3,4) is removed. (Numbers not shown are unchanged.)

Artificial Starts

To apply the Simplex method, one generally uses an artificial start because an initial basic feasible solution can not be found otherwise.

In the Phase I - Phase II form of artificial start, the Simplex method is first used to compute a flow of value v without concern for cost. This flow is used as the starting solution, and Simplex reapplied.

It would appear that a better technique would be to apply a fast max flow algorithm (which might be no better than an efficient implementation of Simplex), to a network consisting of the cheaper arcs. How cheap would depend on the ratio between v and the maximum flow. This and other advanced starting procedures are currently being investigated.

In the Big-M method, tested by Mulvey [16] to be 73% faster than current versions of Phase I - Phase II, an artificial arc (s,t) is added with a large cost M and a flow of value v. The initial flow in the network is zero. Simplex proceeds by sending flow along cheap s - t paths (cheapest s - t paths with the most negative pivot rule), and reducing the flow in the artificial arc. When the flow in (s,t) reaches zero, the solution is optimal when the most negative rule is employed. Otherwise additional iterations may be necessary to eliminate all negative cycles.

Note that negative cycles initially introduced by entering "cheap" rather than "cheapest" arcs might well be blocked by subsequent augmentations, and thus not exist when the desired level of flow is reached.

An example of a Big-M start is shown in Figure 4a. The starting basis tree for Simplex consists of artificial arcs running from s to each node. Each artificial arc has cost M and flow zero, except for (s,t) which has flow v. Using the most negative pivot rule, Simplex will develop a basis similar to that shown in Figure 4b prior to the first cyclic augmentation.

Note that the starting artificial basis tree could just as easily be rooted backwards from t, in which case Simplex (with a most negative pivot rule) would develop a shortest path tree backwards from t.

Computational results in [1] suggest that it may not be profitable to start with a complete shortest path tree. This could easily be because many nodes are so "out of the way" that they never carry flow.

Figure 5 gives the node numbers and modified arc costs for the first two pivots which occur when Simplex is started with the basis shown in Figure 4c. Dotted arcs are non-basic. Only non-basic arcs whose modified costs involve M are shown.

Observation 3: The simplex basis in Figures 5a, b, and c consists of the artificial arc (s,t) together with two trees, one rooted from s with node set S, the other rooted into t with node set T. The entering arc is determined by looking in h^f for the forward arc of minimum modified cost in the cut (S,T). The corresponding arc in h is entered.

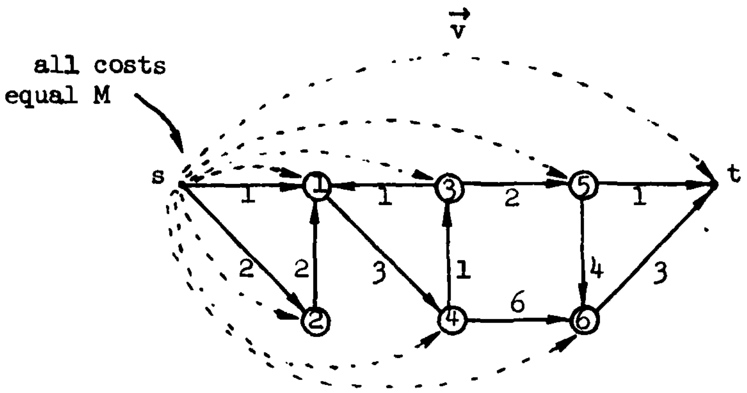


Figure 4a: Dotted arcs represent starting artificial basis for Simplex. Flows on all arcs other than (s,t) are zero.

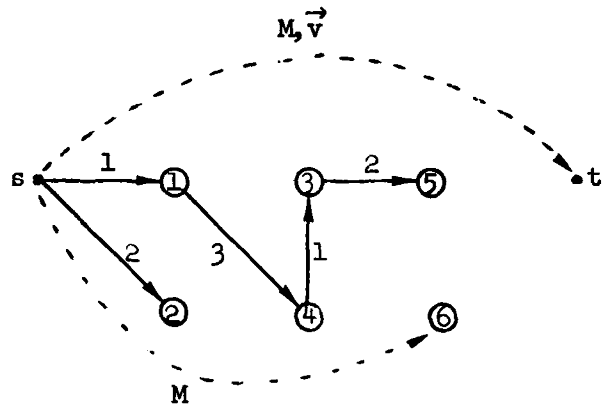


Figure 4b: Basis for Simplex preceding first augmentation.

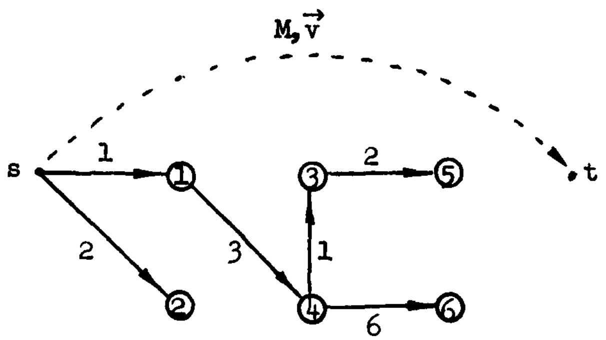


Figure 4c: Equivalent basis for Simplex containing entire shortest path tree.

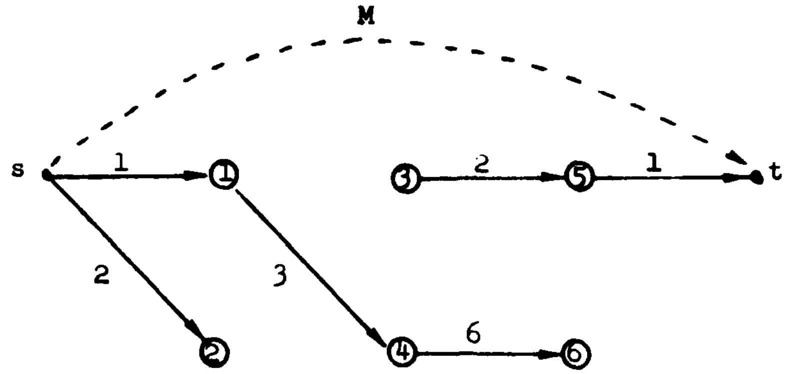


Figure 5a: Simplex basis after the first augmentation in Figure 4c. Arc (4,3) was saturated and left the basis.

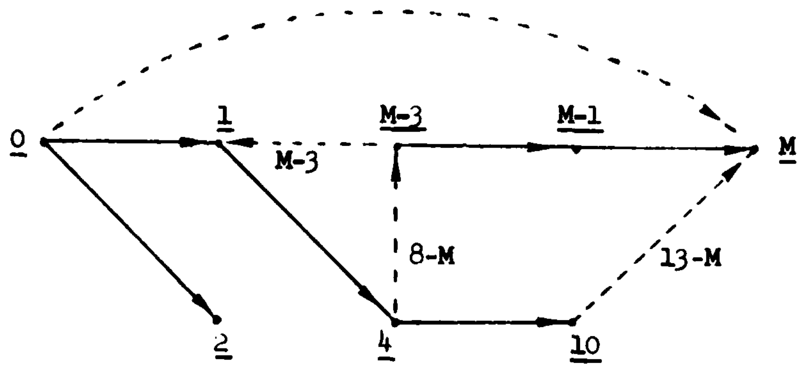


Figure 5b: New node numbers and modified costs across the cut (S,T). Note that reverse arcs in the cut have high cost.

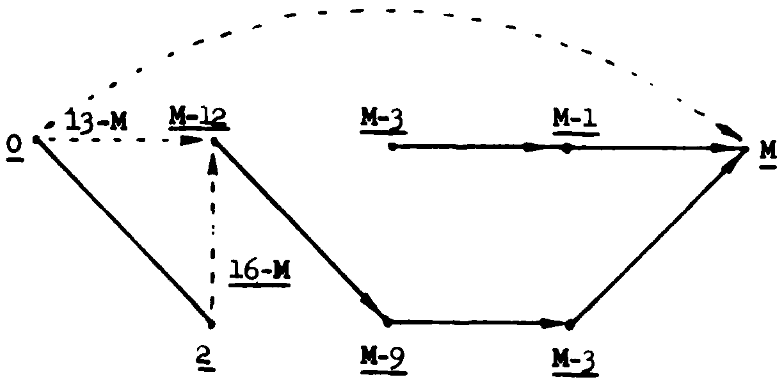


Figure 5c: Node numbers and relevant modified costs after the pivot in which (6,t) replaces (s,1). Arc (4,3) could not enter because it was saturated.

<u>Definition</u>: Let $\underline{t_{ij}}$ be the cost of the unique path from i to j in the basis tree whether or not that path is blocked. Let $\underline{p_{ij}}$ represent the cost of the cheapest augmenting path from i to j in the network.

Theorem 1: Given a Big-M start, the simplex method (with a most negative pivot rule) will construct a shortest path tree in the same order as Dijkstra to all nodes i such that $p_{si} < p_{st}$. Nodes with $p_{si} > p_{st}$ will be connected by artificial arcs.

The simplex method will then solve the problem by augmenting along a sequence of shortest paths, reducing flow in the artificial arc (s,t) during each augmentation. The optimal solution is obtained when the flow in (s,t) becomes zero.

Proof: The first part follows by induction. Initially, $\pi_s = 0$ and $\pi_i = M \; \forall \, i \neq s$. Note that the modified cost of arc (i,j) equals d_{ij} for $i \neq s$, and for $i = s$ equals $d_{sj} - M$. Thus the first arc entered will be the cheapest adjacent to s, as in Dijkstra and the artificial arc adjacent to that node will be pivoted out (by a degenerate pivot).

Suppose inductively that Simplex has computed a shortest path tree to a set of permanently labeled nodes P, and has pivoted out the artificial arcs initially associated with nodes in $P \sim s$. Then the current modified cost of (i,j) for $i \in P$, $j \notin P$, will be $p_{si} + d_{ij} - M$. The modified cost of arcs outside the cut $(P,\overline{P})$ will not involve M. It follows that Simplex will enter the arc (i,j) which achieves $\min_{i \in P, j \notin P} \{p_{si} + d_{ij}\}$ as in Dijkstra.

To prove the second assertion, no   at the first augmentation will occur along a shortest s-t path. In general, the simplex basis will contain the arc (s,t), artificial arcs from s to a set of nodes J, and two trees $\mathcal{S}$ and $\mathcal{T}$, $\mathcal{S}$ rooted at s with node set S, and $\mathcal{T}$ rooted into t, with node set T. The proof will be given first for the case where J is empty.

Assume inductively that Simplex has augmented along a sequence of shortest paths, and that $t_{sj} \leq p_{sj}$ $\forall j \in S$, and $t_{jt} \leq p_{jt}$ $\forall j \in T$. We wish to show that Simplex will continue to satisfy these conditions, and that the next augmentation will be along a shortest path.

In what follows, let the unique arc between i and j in A be denoted by $(\overline{i,j})$ if "entering that arc in the basis" causes flow to be sent from i to j. Otherwise the arc is denoted $(\overline{j,i})$. Given a flow f, let $\bar{d}_{ij}$ represent the cost of $(i,j) \in h^f$. Thus $\bar{d}_{ij} = d_{ij}$ if $f_{ij} < c_{ij}$, and $\bar{d}_{ij} = -d_{ji}$ if $f_{ji} > 0$.

Suppose $(i,j) \in h^f$ is a forward arc in the cut (S,T). The modified cost of (i,j) in h^f is $t_{si} + \bar{d}_{ij} + t_{jt} - M$, which is just the cost of the unique s-t path formed when $(\overline{i,j})$ enters the basis minus M. Since $t_{si} \leq p_{si}$ $\forall i \in S$, and $t_{jt} \leq p_{jt}$ $\forall j \in T$, the path selected by Simplex must be at least as cheap as $\min_{i\in S, j\in T} \{p_{si} + \bar{d}_{ij} + p_{jt}\}$, which is the cost of the cheapest augmenting path. It suffices to show that after the pivot is made (which may be degenerate), we still have $t_{sj} \leq p_{sj}$ $\forall j \in S$, $t_{jt} \leq p_{jt}$ $\forall j \in T$.

Suppose that $(\overline{i,j})$ enters the basis and $(\overline{k,\ell}) \in \mathcal{S}$ leaves. (The case $(\overline{k,\ell}) \in \mathcal{T}$ is similar.) Let S' be the portion of S connected to T via (i,j) after the pivot. Since an augmentation of any amount (including zero) along a shortest path will not decrease

p_{sx} or p_{xt} (see [8]), it suffices to show that $t_{xt}^{new} \leq p_{xt}^{old}$ $\forall x \in S'$. Choose $x \in S'$. Let P be an augmenting path from x to t of length p_{xt}. Let y be the node on the previous augmenting path closest to x, in the sense that the number of basic arcs in the unique path between y and x is minimum. Then since the previous augmentation occurred over a shortest path, $t_{yx} + p_{xt} \geq t_{yt}$ or $p_{xt} \geq t_{yt} - t_{yx} = t_{yt} + t_{xy} = t_{xt}$.

We will now extend the induction to non-empty J. It suffices to show that a node $z \in J$ cannot lie on <u>the cheapest</u> augmenting $s - t$ path, and that when z leaves J, we have $p_{sz} \geq t_{sz}$.

Given a simplex basis $\mathcal{S} \cup \mathcal{J}$, with $(\overline{p,q})$ a forward arc of minimum modified cost in the cut (S,T) in η^f, let t'_{it} denote the cost of the unique path from i to t in $\mathcal{S} \cup \mathcal{J} \cup (\overline{p,q})$.

We want to show that as long as z remains in J, $d_{iz} \geq t'_{it}$ $\forall i \in N$. This will imply that $p_{sz} \geq \min_i \{p_{si} + d_{iz}\} \geq \min_i \{t_{si} + t'_{it}\} = t'_{st}$, and since $p_{zt} \geq 0$ by [8], the result will follow.

Consider the first pivot for which $d_{xz} > t'_{xt}$ for some x. We wish to show that z will join S, i.e., some arc of the form (i,z) has minimum modified cost. Suppose instead that (w,u) has minimum modified cost (see Figure 6). We have $t'_{xt} = t_{xy} + t'_{yt} > d_{xz} \Rightarrow t'_{yt} > t_{yx} + d_{xz} \Rightarrow (w,u)$ has modified cost greater than (x,z). ⊠

<u>Corollary 1:</u> The Simplex Method with a Big-M start and most negative pivot rule will yield the same pivot sequence if M is initially set at zero and then increased.

<u>Comment:</u> Corollary 1 establishes the relationship between Big-M Simplex and primal dual methods such as Lemke's algorithm and Out-of-Kilter. In those methods, the essential difference is that M, which

may be viewed as the cost of the current cheapest s-t path, is slowly increased by a sequence of "increases in potential across cuts" instead of being fixed at one value throughout.

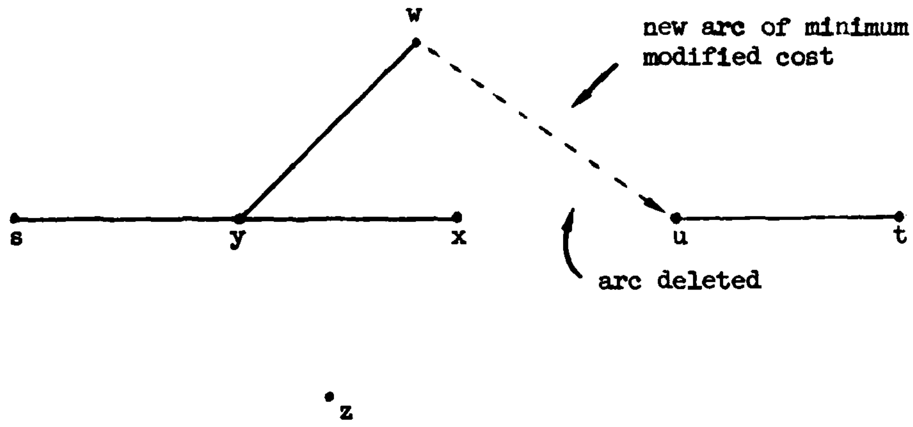


Figure 6: An explanation of the last argument in Theorem 1

Homotopy Principle

The Homotopy principle has wide applicability and is closely related to the concept of artificial variables. For network flows it is similar to the idea behind efficient path methods: start with zero flow and build up to the desired level. Essentially one deforms a problem by adding one artificial variable so that the deformed problem has a trivial solution. The problem is then solved repeatedly as the artificial variable is driven to zero.

To deform a minimum cost flow problem using the Homotopy principle, an artificial arc (s,t) is added, along which a flow θv is sent, where v is the required level of $s - t$ flow. The deformed problem then is

$$\begin{aligned} \text{minimize} \quad & \sum d_{ij} f_{ij} \\ \text{subject to} \quad & F(N,s) - F(s,N) = -v + \theta v \\ & F(N,i) - F(i,N) = 0 \qquad i \neq s,t \\ & F(N,t) - f(t,N) = v - \theta v \\ & 0 \leq f_{ij} \leq c_{ij} \end{aligned}$$

For $\theta = 1$, we have a trivial solution $f_{ij} \equiv 0$, which is the same as the starting solutions for Big-M simplex and efficient path methods. For $\theta = 0$, we have the original problem. As θ travels from 1 to 0, the solutions obtained are optimal for the flow level $v(1 - \theta)$.

Complementary Pivot Theory

Complementary pivot theory can be used to solve linear programs as well as quadratic programs and bimatrix games. Basically the idea is to find a point which satisfies complementary slackness and is feasible for the primal and dual systems, and hence optimal. Lemke's algorithm is one of several methods [4], [15], [17], for solving linear complementarity problems. We will show that Lemke's procedure yields a sequence of solutions similar to those obtained by Dantzig's simplex method when both are applied to network problems with non-negative costs. This will imply that network problems in [18] are pathological for Lemke's algorithm as well as for the simplex method.

Application of Lemke's Algorithm to Network Flow Problems

Consider the problem of finding a minimum cost s - t flow of value 2 in the following network. To write this as a Linear

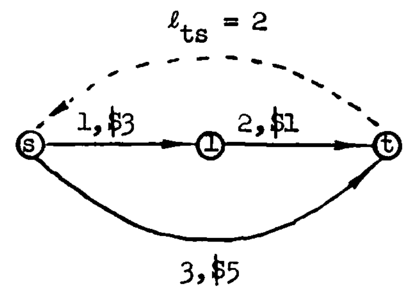


Figure 7: Network for the minimum cost flow problem. Arc (t,s) is added with a lower bound of 2 to allow replacement of node conservation constraints by inequalities.

Complementarity Problem suitable for solution by Lemke's algorithm, the node conservation equations must be made into inequalities. This may be accomplished using a trick suggested by Richard Stone. A reverse arc (t,s) is added with a lower bound $\ell_{ts} = 2$, and each conservation equality constraint is replaced by an inequality. By following the inequalities around any cycle, the reader can verify that the new system forces equality at all nodes. An equivalent formulation of the minimum cost flow problem suitable for complementary pivot algorithms is then

$$\text{minimize} \quad 0 \cdot f_{ts} + \Sigma\, d_{ij} f_{ij}$$

$$\text{subject to} \quad F(N,i) - F(i,N) \leq 0$$

$$-f_{ts} \leq -v$$

$$f_{ij} \leq c_{ij} \qquad (i,j) \neq (t,s)$$

$$f_{ij} \geq 0 \text{ .}$$

The dual problem is

$$\begin{aligned} \text{maximize} \quad & -\lambda_{ts} v + \sum c_{ij} \sigma_{ij} \\ \text{subject to} \quad & \pi_s - \pi_t - \lambda_{ts} \leq 0 \\ & -\pi_i + \pi_j + \sigma_{ij} \leq d_{ij} \qquad (i,j) \neq (t,s) \\ & \pi_i, \sigma_{ij}, \lambda_{ts} \leq 0 \end{aligned}$$

The primal problem for the network in Figure 7 is

$$\begin{aligned} \text{minimize} \quad & 3f_{s1} + f_{1t} + 5f_{st} + 0 \cdot f_{ts} \\ \text{subject to} \quad & -f_{s1} - f_{st} + f_{ts} \leq 0 \\ & f_{s1} - f_{1t} \leq 0 \\ & f_{1t} + f_{st} - f_{ts} \leq 0 \\ & f_{s1} \leq 1 \\ & f_{1t} \leq 2 \\ & f_{st} \leq 3 \\ & - f_{ts} \leq -2 \end{aligned}$$

$$f_{ij} \geq 0$$

The dual problem is

$$\begin{array}{llllllll}
\text{maximize} & & & \sigma_{s1} & + 2\sigma_{1t} & + 3\sigma_{st} & - 2\lambda_{ts} & \\
\text{subject to} & -\pi_s + \pi_1 & & + \sigma_{s1} & & & & \leq 3 \\
& & -\pi_1 + \pi_t & & + \sigma_{1t} & & & \leq 1 \\
& -\pi_s & + \pi_t & & & + \sigma_{st} & & \leq 5 \\
& \pi_s & - \pi_t & & & & - \lambda_{ts} & \leq 0
\end{array}$$

$\pi_i \leq 0,\ \sigma_{ij} \leq 0,\ \lambda_{ts} \leq 0.$

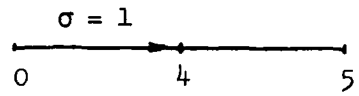
We have chosen to use $(\leq)$ constraints in the primal for two reasons. First, it is desirable to have as many slack variables as possible in the primal so as to minimize the number of constraints affected by artificial variables. Secondly, we want initial dual pivots to form a shortest path tree rooted from s. Using $(\leq)$ constraints accomplishes both objectives but requires the use of negative node numbers. Using $(\geq)$ constraints would yield positive node numbers but causes initial dual pivots to form a shortest path tree routed backwards from t. Figure 8 explains how the optimal dual solution is affected by the primal formulation for the network shown in Figure 7. In Figure 8a, the conservation equations are written as (flow in) - (flow out) ≥ 0, and the resulting optimal dual solution has non-negative variables. Tracing through the sequence of dual pivots, one would find that π_t was first increased to 1, then π_1 and π_t increased to 3 and 4 respectively. In other words, initial pivots would build a shortest path tree backwards from t.

By changing the conservation equations to (flow out) - (flow in) ≥ 0 as in Figure 8b, the values of the node numbers change, but the differences $\{\pi_i - \pi_j\}$, in absolute value, remain the same.

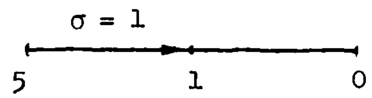
The formulation we use in Figures 9a-f is shown in Figure 8d. Note that adding 5 to each node number would yield the same values as in Figure 8a.

Primal Formulation | Optimal Dual Solution

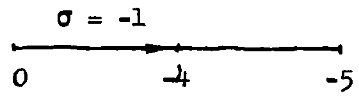
$$
\begin{array}{rcl}
f_{ts} - f_{s1} \qquad - f_{st} & \geq & 0 \\
f_{s1} - f_{1t} \qquad & \geq & 0 \\
-f_{ts} \qquad + f_{1t} + f_{st} & \geq & 0 \\
(- f_{ij}) & \geq & (-c_{ij}) \\
(\text{flow in}) - (\text{flow out}) & \geq & 0
\end{array}
$$



$$
\begin{array}{rcl}
-f_{ts} + f_{s1} \qquad + f_{st} & \geq & 0 \\
- f_{s1} + f_{1t} \qquad & \geq & 0 \\
f_{ts} \qquad - f_{1t} - f_{st} & \geq & 0 \\
(- f_{ij}) & \geq & (-c_{ij}) \\
(\text{flow out}) - (\text{flow in}) & \geq & 0
\end{array}
$$



$$
\begin{array}{rcl}
f_{ts} - f_{s1} \qquad - f_{st} & \leq & 0 \\
f_{s1} - f_{1t} \qquad & \leq & 0 \\
-f_{ts} \qquad + f_{1t} + f_{st} & \leq & 0 \\
(f_{ij}) & \leq & (c_{ij}) \\
(\text{flow in}) - (\text{flow out}) & \leq & 0
\end{array}
$$



$$
\begin{array}{rcl}
-f_{ts} + f_{s1} \qquad + f_{st} & \leq & 0 \\
- f_{s1} + f_{1t} \qquad & \leq & 0 \\
f_{ts} \qquad - f_{1t} - f_{st} & \leq & 0 \\
(f_{ij}) & \leq & (c_{ij}) \\
(\text{flow out}) - (\text{flow in}) & \leq & 0
\end{array}
$$

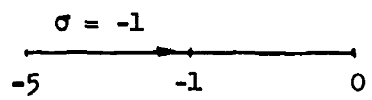


Figure 8: An example of how the dual solution varies as a function of the primal formulation

An Example

Recall that Lemke's algorithm seeks a point satisfying both primal and dual systems, and complementary slackness. To find such a point for the network in Figure 7, the primal and dual systems are written in a tableau, as shown in Figure 9a. The first three primal slacks are denoted n_s, n_1, and n_t, standing for node slacks, which may be thought of as artificial arcs (from s). The next three primal slacks are denoted c'_{s1}, c'_{1t}, c'_{st}, and are residual capacities. There is one surplus variable, ℓ'_{ts}. The four dual slacks are denoted d'_{s1}, d'_{1t}, d'_{st}, d'_{ts}, standing for modified costs. Note that the first four rows contain the dual system, written in terms of $-\pi_s,\ldots,-\lambda_{st}$, to keep everything positive, and the last seven rows contain the primal system. An artificial variable z_0 must be added to the primal system to make all components of q, the vector of basic variables non-negative.

Column i is complementary to column $i + 11$, $i = 1,\ldots,11$, meaning that both columns cannot appear in the same basis (this is the complementary slackness condition). The initial infeasible basis consists of slack variables with one surplus variable, ℓ'_{ts}.

	d'_{sl}	d'_{lt}	d'_{st}	d'_{ts}	n_s	n_l	n_t	c'_{sl}	c'_{lt}	c'_{st}	ℓ'_{ts}	f_{sl}	f_{lt}	f_{st}	f_{ts}	$\bar{\pi}_s$	$\bar{\pi}_l$	$\bar{\pi}_t$	$\bar{\sigma}_{sl}$	$\bar{\sigma}_{lt}$	$\bar{\sigma}_{st}$	$\bar{\lambda}_{ts}$	z_0	q
d'_{sl}	1															1	-1		-1					3
d'_{lt}		1															1	-1		-1				1
d'_{st}			1													1		-1			-1			5
d'_{ts}				1												-1		1				(1)		0
n_s					1							-1		-1	1									0
n_l						1						1	-1											0
n_t							1						1	1	-1									0
c'_{sl}								1				1												1
c'_{lt}									1				1											2
c'_{st}										1				1										3
ℓ'_{ts}											1				-1								(-1)	-2

Figure 9a: Initial tableau

	d'_{sl}	d'_{lt}	d'_{st}	d'_{ts}	n_s	n_l	n_t	c'_{sl}	c'_{lt}	c'_{st}	ℓ'_{ts}	f_{sl}	f_{lt}	f_{st}	f_{ts}	$\bar{\pi}_s$	$\bar{\pi}_l$	$\bar{\pi}_t$	$\bar{\sigma}_{sl}$	$\bar{\sigma}_{lt}$	$\bar{\sigma}_{st}$	$\bar{\lambda}_{ts}$	z_0	q
d'_{sl}	1															(1)	-1		-1					3
d'_{lt}		1															1	-1		-1				1
d'_{st}			1													1		-1			-1			5
λ_{ts}				1												-1		1				1		0
n_s					1							-1		-1	(1)									0
n_l						1						1	-1											0
n_t							1						1	1	-1									0
c'_{sl}								1				1												1
c'_{lt}									1				1											2
c'_{st}										1				1										3
z_0											-1				1								1	2

Figure 9b: The first pivot. Dual variables in the tableau are

$-\pi_s$, $-\pi_l$, $-\pi_t$, $-\sigma_{sl}$, $-\sigma_{lt}$, $-\sigma_{st}$, $-\lambda_{ts}$

	d'_{sl}	d'_{lt}	d'_{st}	d'_{ts}	n_s	n_l	n_t	c'_{sl}	c'_{lt}	c'_{st}	ℓ'_{ts}	f_{st}	f_{lt}	f_{st}	f_{ts}	$\bar{\pi}_s$	$\bar{\pi}_l$	$\bar{\pi}_t$	$\bar{\sigma}_{sl}$	$\bar{\sigma}_{lt}$	$\bar{\sigma}_{st}$	$\bar{\lambda}_{ts}$	z_0	q
$\bar{\pi}_s$	1															1	-1		-1					3
d'_{lt}		1															(1)	-1		-1				1
d'_{st}	-1		1														1	-1	1		-1			2
$\bar{\lambda}_{ts}$	1			1													-1	1	-1			1		3
f_{ts}					1							-1		-1	1									0
n_l						1						(1)	-1											0
n_t					1		1					-1	1											0
c'_{sl}								1				1												1
c'_{lt}									1				1											2
c'_{st}										1				1										3
z_0					-1						-1	1		1									1	2

Figure 9c

	d'_{sl}	d'_{lt}	d'_{st}	d'_{ts}	n_s	n_l	n_t	c'_{sl}	c'_{lt}	c'_{st}	ℓ'_{ts}	f_{sl}	f_{lt}	f_{st}	f_{ts}	$\bar{\pi}_s$	$\bar{\pi}_l$	$\bar{\pi}_t$	$\bar{\sigma}_{sl}$	$\bar{\sigma}_{lt}$	$\bar{\sigma}_{st}$	$\bar{\lambda}_{ts}$	z_0	q
$\bar{\pi}_s$	1	1														1		-1	-1	-1				4
$\bar{\pi}_l$		1															1	-1		-1				1
d'_{st}	-1	-1	1																(1)	1	-1			1
$\bar{\lambda}_{ts}$	1	1		1															-1	-1		1		4
f_{ts}					1	1							-1	-1	1									0
f_{sl}						1						1	-1											0
n_t					1	1	1																	0
c'_{sl}						-1		1					(1)											1
c'_{lt}									1				1											2
c'_{st}										1				1										3
z_0					-1	-1					-1		1	1									1	2

Figure 9d

	d'_{sl}	d'_{lt}	d'_{st}	d'_{ts}	n_s	n_l	n_t	c'_{sl}	c'_{lt}	c'_{st}	ℓ'_{ts}	f_{sl}	f_{lt}	f_{st}	f_{ts}	$\bar{\pi}_s$	$\bar{\pi}_l$	$\bar{\pi}_t$	$\bar{\sigma}_{sl}$	$\bar{\sigma}_{lt}$	$\bar{\sigma}_{st}$	$\bar{\lambda}_{ts}$	z_0	q
$\bar{\pi}_s$			1													1		-1						5
$\bar{\pi}_l$		1															1	-1		-1				1
σ_{sl}	-1	-1	1																1	1	-1			1
$\bar{\lambda}_{ts}$			1	1																		1		5
f_{ts}					1			1						-1	1									1
f_{sl}								1				1												1
n_t					1	1	1																	0
f_{lt}						-1		1					1											1
c'_{lt}									1															1
c'_{st}										1				1										3
z_0					-1			-1			-1			(1)									1	1

	d'_{sl}	d'_{lt}	d'_{st}	d'_{ts}	n_s	n_l	n_t	c'_{sl}	c'_{lt}	c'_{st}	ℓ'_{ts}	f_{sl}	f_{lt}	f_{st}	f_{ts}	$\bar{\pi}_s$	$\bar{\pi}_l$	$\bar{\pi}_t$	$\bar{\sigma}_{sl}$	$\bar{\sigma}_{lt}$	$\bar{\sigma}_{st}$	$\bar{\lambda}_{ts}$	z_0	q
$\bar{\pi}_s$																								5
$\bar{\pi}_l$																								1
σ_{sl}																								1
$\bar{\lambda}_{ts}$																								5
f_{ts}											-1				1								1	2
f_{sl}								1				1												1
n_t					1	1	1																	0
f_{lt}						-1		1					1											1
c'_{lt}									1															1
c'_{st}					1			1		1	1												-1	2
f_{st}					-1			-1			-1			1									1	1

Sequence of Pivots

Entering Variable	Exiting Variable	Comment
z_0	ℓ'_{ts}	z_0 is entered to make q non-negative.
λ_{ts}	d'_{ts}	Degenerate pivot.
f_{ts}	n_s	The primal begins to construct a shortest path tree by pivoting out the node slacks.
π_s	d'_{s1}	The dual begins to determine node numbers corresponding to the shortest path tree. π_s is now -3.
f_{s1}	n_1	Another node slack leaves in a degenerate pivot.
π_1	d'_{1t}	Potential is increased across the cut (s1,2t) by 1, making $\pi_s = -4$ and $\pi_1 = -1$.
f_{1t}	c'_{s1}	Augmentation of one unit across the path s1t saturates arc (s,1).
σ_{s1}	d'_{st}	Increase in potential of 1 across the cut (s,12t) makes d'_{st} zero.
f_{st}	z_0	Augmentation of one unit across the path (s,t) drives out z_0.

Summary

The complementary pivot algorithm alternates between primal and dual pivots, which are disjoint. The dual pivot may be viewed as a process for determining which non-basic variable in the primal has the minimum modified cost coefficient. The algorithm begins by entering an artificial variable $\dot{z}_0$ to make the initial basis of slacks non-negative. This determines a unique sequence of alternating primal and dual pivots in a non-degenerate system.

The variable which enters in each case is the complement of the leaving variable. Thus when z_0 enters, knocking out ℓ'_{ts}, the complement of ℓ'_{ts}, namely λ_{ts}, enters in the dual. This eliminates the dual slack d'_{ts} which allows f_{ts} into the primal, which in turn knocks out n_s, allowing π_s to enter the dual system. The value of π_s becomes -3, which means the potential across the cut (s,12t) is increased by 3, the cost of the cheapest arc in that cut, equivalently, the cheapest arc adjacent to s. The same computation occurs in the Out-of-Kilter, Primal-Dual, Big-M Simplex, and Dijkstra methods. Setting π_s to -3 makes the modified cost of arc (s,1) zero (it becomes "admissible"), and f_{s1} is allowed to enter the primal basis.

Subsequent dual pivots determine values of π_i (node numbers) which correspond to a shortest path tree rooted at s. The node numbers are introduced in the same order as in Dijkstra. In the primal, node slacks (which may be thought of as artificial arcs) are eliminated and replaced by arcs pivoted in at a zero level forming a shortest path tree.

Neglecting n_t, which corresponds to the added constraint, the sequence of primal basis "trees" associated with tableaus 9c-9f is $\{n_1, z_0\}$, $\{(s,1), z_0\}$, $\{(1,t), z_0\}$, and $\{(s,t),(1,t)\}$. This is the same sequence that would appear in the Big-M Method.

The augmentation process may be described as follows: The last node number to enter the dual system forces the modified cost of some arc (j,t) to zero, allowing f_{jt} to enter the primal. This causes an augmentation which saturates some arc (u,v), thereby allowing an increase in potential across the dual "cut" containing (u,v). The increase in potential drives the modified cost of some arc (p,q) in that cut to zero, allowing f_{pq} to enter in the primal. Some primal pivots may be degenerate, meaning that increases in potential across several cuts must be made before additional flow can be sent. Successive augmentations continue in this fashion, reducing the value of z_0 until it becomes zero, at which point the solution is optimal.

In order to prove the essential equivalence of Lemke's algorithm and the Simplex method on non-negative network problems, it is necessary to study the application of the Simplex method to the dual problem in more detail. Some of the material in the following section, notably Theorem 2 and the essence of Figure 12, was communicated to the author by Robert Bland, and will be appearing in one of his forthcoming papers.

Interpretation of the Dual

In this section we will for the purposes of comprehension deal with non-negative π_i by rewriting the dual problem as

$$\begin{aligned} \text{maximize} \quad & \lambda v - \sum c_{ij}\sigma_{ij} \\ \text{subject to} \quad & \pi_s - \pi_t + \lambda \leq 0 \\ & -\pi_i + \pi_j - \sigma_{ij} \leq d_{ij}, \qquad (i,j) \in A \\ & \pi_i,\ \sigma_{ij},\ \lambda \geq 0 . \end{aligned}$$

With this understanding, the optimal value of π_i may be interpreted as the length of the shortest augmenting path from s to i during the last network augmentation, or afterwards if an augmenting path still exists. To see that an augmenting path to each node i exists prior to the last augmentation, note that such a path from s to t exists. If all direct paths from s to i have been blocked, then the flow on some path from i to t may be reduced, yielding an augmenting path $s \to t \to i$.

To interpret σ_{ij}, note that the dual equation $-\pi_i + \pi_j - \sigma_{ij} \leq d_{ij}$ may be rewritten as $-\sigma_{ij} \leq \pi_i + d_{ij} - \pi_j$. The quantity $\pi_i + d_{ij} - \pi_j$ is just the modified cost of arc (i,j). In a basic dual solution, σ_{ij} and the slack for the above equation, namely s_{ij}, cannot both be in the basis. In other words,

$-\sigma_{ij}$ = the modified cost of arc (i,j) when negative,

s_{ij} = the modified cost of arc (i,j) when positive.

The objective function is maximized when $\lambda = \pi_t - \pi_s$. In general λ may be interpreted as the length of the current shortest s-t augmenting path.

Dual Bases and Pivots

Consider the problem of finding a minimum cost $s - t$ flow of value 5 in the network in Figure 10b. The arc with capacity 1 and cost 6 will be referred to as $(\tilde{s,2})$. A reverse arc (t,s) is added with a lower bound of 5 to yield a circulation. Non-negative dual variables are obtained by writing the primal problem as

$$\begin{array}{llr}
\text{minimize} & 0 \cdot f_{ts} + 2f_{s1} + 3f_{s2} + 6f_{\tilde{s2}} + 2f_{12} + 4f_{1t} + f_{2t} & \\
\text{subject to} & f_{ts} - f_{s1} - f_{s2} - f_{\tilde{s2}} & \geq 0 \\
& f_{s1} - f_{12} - f_{1t} & \geq 0 \\
& f_{s2} + f_{\tilde{s2}} + f_{12} - f_{2t} & \geq 0 \\
& -f_{ts} + f_{1t} + f_{2t} & \geq 0 \\
& f_{ts} & \geq 5 \\
& -f_{s1} & \geq -3 \\
& -f_{s2} & \geq -1 \\
& -f_{\tilde{s2}} & \geq -1 \\
& -f_{12} & \geq -1 \\
& -f_{1t} & \geq -3 \\
& -f_{2t} & \geq -2 \\
& f_{ij} \geq 0 &
\end{array}$$

The dual is

$$\begin{array}{lllllllllllll}
\text{maximize} & & & & 5\lambda & -3\sigma_{s1} & -\sigma_{s2} & -\sigma_{\tilde{s2}} & -\sigma_{12} & -3\sigma_{1t} & -2\sigma_{2t} & & \\
\text{subject to} & \pi_s & & & -\pi_t + \lambda & & & & & & & \le 0 \\
& -\pi_s + \pi_1 & & & & -\sigma_{s1} & & & & & & \le 2 \\
& -\pi_s & +\pi_2 & & & & -\sigma_{s2} & & & & & \le 3 \\
& -\pi_s & +\pi_2 & & & & & -\sigma_{\tilde{s2}} & & & & \le 6 \\
& -\pi_1 & +\pi_2 & & & & & & -\sigma_{12} & & & \le 2 \\
& -\pi_1 & & +\pi_t & & & & & & -\sigma_{1t} & & \le 4 \\
& & -\pi_2 & +\pi_t & & & & & & & -\sigma_{2t} & \le 1
\end{array}$$

$$\pi_i, \lambda, \sigma_{ij} \ge 0$$

Figure 10 shows the initial tableau with λ and six slacks corresponding to positive modified costs (their columns not shown) in the basis. The first pivot enters π_t at a value of 1, driving d'_{2t} to zero, and increasing λ to 1. The second and third pivots yield $\pi_1 = 1$, $\pi_2 = 3$, $\pi_t = \lambda = 4$ determining s2t as the cheapest $s - t$ path. The dual objective function is $5 \cdot 4 = 20$, which is a lower bound, since 5 units must be sent and the cost per unit will be <u>at least 4</u>.

The variable λ corresponds to an arc between s^* and t^* in the dual graph. Each successive π_i which enters eliminates a slack variable until in Figure 10h, the union of λ and the slacks forms a tree in the dual graph. Further pivots will involve only variables representing $\pm$ modified costs; π_1, π_2, and π_t will not leave, and π_s being zero throughout, will not enter.

π_s	π_1	π_2	π_t	λ	σ_{s1}	σ_{s2}	$\sigma_{\tilde{s}2}$	σ_{12}	σ_{1t}	σ_{2t}		
1			-1	1							0	λ
-1	1				-1						2	s_{s1}
-1		1				-1					3	s_{s2}
-1		1					-1				6	$s_{\tilde{s}2}$
	-1	1						-1			2	s_{12}
	-1		1						-1		4	s_{1t}
		-1	(1)							-1	1	s_{2t}
-5			5	0	-3	-1	-1	-1	-3	-2		

Figure 10a: Starting tableau

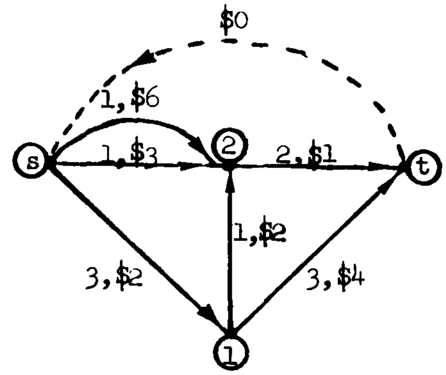


Figure 10b: Initial arc capacities and costs

π_s	π_1	π_2	π_t	λ	σ_{s1}	σ_{s2}	$\sigma_{\tilde{s}2}$	σ_{12}	σ_{1t}	σ_{2t}		
1		-1		1						-1	1	λ
-1	1				-1						2	s_{21}
-1		1				-1					3	s_{s2}
-1		1					-1				6	$s_{\tilde{s}2}$
	-1	(1)						-1			2	s_{12}
	-1	1							-1	1	3	s_{1t}
		-1	1							-1	1	π_t
-5		5	0	0	-3	-1	-1	-1	-3	3		

Figure 10c: π_2 enters, s_{12} leaves

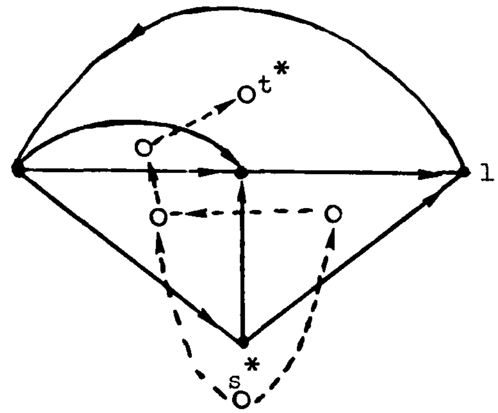


Figure 10d: Dual slacks and positive node numbers after first pivot

π_s	π_1	π_2	π_t	λ	σ_{s1}	σ_{s2}	$\sigma_{\tilde{s}2}$	σ_{12}	σ_{1t}	σ_{2t}		
1	-1			1				-1		-1	3	λ
-1	1				-1						2	s_{s1}
-1	(1)					-1		1			1	s_{s2}
-1	1						-1	1			4	$s_{\tilde{s}2}$
	-1	1						-1			2	π_2
								1	-1	1	1	s_{1t}
	-1		1					-1		-1	3	π_t
-5	5	0	0	0	-3	-1	-1	4	-3	3		

Figure 10e: π_1 enters, s_{s2} leaves

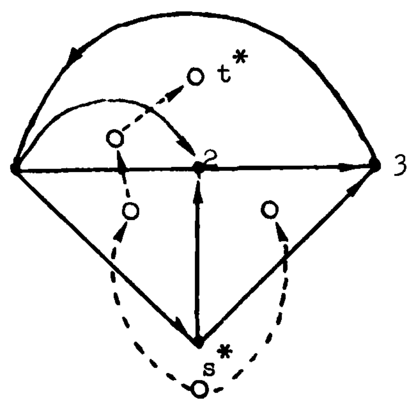


Figure 10f: Dual slacks and node numbers after second pivot

π_s	π_1	π_2	π_t	λ	σ_{s1}	σ_{s2}	$\sigma_{\tilde{s}2}$	σ_{12}	σ_{1t}	σ_{2t}		
				1		-1				-1	4	λ
					-1	(1)		-1			1	s_{s1}
-1	1					-1		1			1	π_1
						1	-1				3	$s_{\tilde{s}2}$
-1		1				-1					3	π_2
								1	-1	1	1	s_{1t}
-1			1			-1				-1	4	π_t
0	0	0	0	0	-3	4	-1	-1	-3	3		

Figure 10g: σ_{s2} enters, s_{s1} leaves

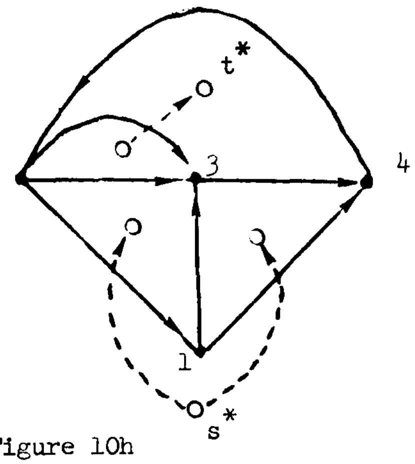


Figure 10h

π_s	π_1	π_2	π_t	λ	σ_{s1}	σ_{s2}	$\sigma_{\tilde{s}2}$	σ_{12}	σ_{1t}	σ_{2t}		
				1	-1			-1		-1	5	λ
					-1	1		-1			1	σ_{s2}
-1	1				-1						2	π_1
					1		-1	1			2	$s_{\tilde{s}2}$
-1		1			-1			-1			4	π_2
								(1)	-1	1	1	s_{1t}
-1			1		-1			-1		-1	5	π_t
0	0	0	0	0	1	0	-1	3	-3	3		

Figure 10i: σ_{12} enters, s_{1t} leaves

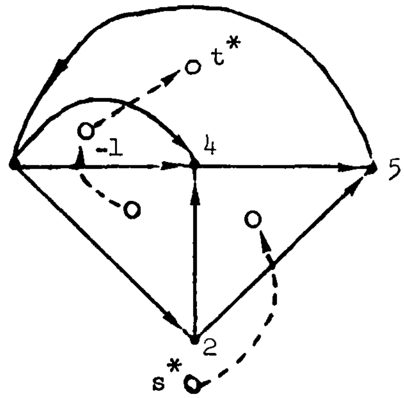


Figure 10j

π_s	π_1	π_2	π_t	λ	σ_{s1}	σ_{s2}	$\sigma_{\tilde{s}2}$	σ_{12}	σ_{1t}	σ_{2t}		
				1	-1				-1		6	λ
					-1	1			-1	1	2	σ_{s2}
-1	1				-1						2	π_1
					(1)		-1		1	-1	1	$s_{\tilde{s}2}$
-1		1			-1				-1	1	5	π_2
								1	-1	1	1	σ_{12}
-1			1		-1				-1		6	π_t
0	0	0	0	0	1	0	-1	0	0	0		

Figure 10k: σ_{s1} enters, $s_{\tilde{s}2}$ leaves

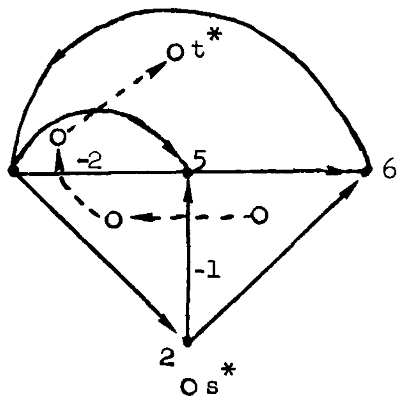


Figure 10ℓ

π_s	π_1	π_2	π_t	λ	σ_{s1}	σ_{s2}	$\sigma_{\tilde{s}2}$	σ_{12}	σ_{1t}	σ_{2t}		
				1			-1			-1	7	λ
						1	-1				3	σ_{s2}
-1	1						-1		1	-1	3	π_1
					1		-1		1	-1	1	σ_{s1}
-1		1					-1				6	π_2
								1	-1	(1)	1	σ_{12}
-1			1				-1			-1	7	π_t
0	0	0	0	0	0	0	0	0	-1	1		

Figure 10m: σ_{2t} enters, σ_{12} leaves

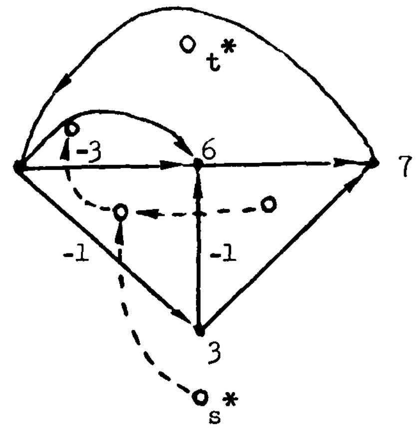


Figure 10n

π_s	π_1	π_2	π_t	λ	σ_{s1}	σ_{s2}	$\sigma_{\tilde{s}2}$	σ_{12}	σ_{1t}	σ_{2t}		
				1			-1	1	-1		8	λ
						1	-1				3	σ_{s2}
-1	1						-1	1			4	π_1
					1		-1	1			2	σ_{s1}
-1		1					-1				6	π_2
								1	-1	1	1	σ_{2t}
-1			1				-1	1	-1		8	π_t
0	0	0	0	0	0	0	0	-1	0	0		

Figure 10o: Optimal tableau

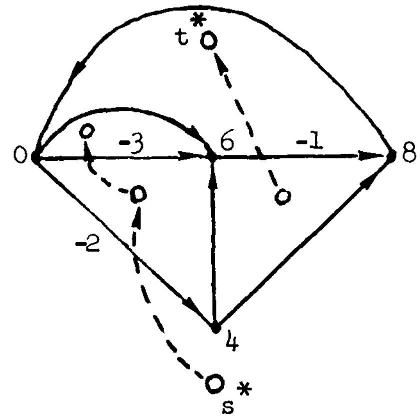


Figure 10p: Optimal non-zero modified costs

Remaining pivots	Comment
1. $\sigma_{s2} \to 1$ $d'_{s1} \to 0$	Increase in potential of 1 across cut $(s,12t)$ determines new incoming primal variable f_{s1}.
2. $\sigma_{12} \to 1$ $d'_{1t} \to 0$	Increase in potential of 1 across $(s1,2t)$ determines new incoming primal variable f_{1t}.
3. $\sigma_{s1} \to 1$ $d'_{\tilde{s}2} \to 0$	Allows $f_{\tilde{s}2}$ to enter primal but primal pivot is degenerate.
4. $\sigma_{2t} \to 1$ $\sigma_{12} \to 0$	Increase in potential of 1 across $(s2,1t)$.

Comparison of Dual Simplex to Big-M Simplex

Figures 11a, c, and e show the infeasible primal bases associated with the feasible dual solutions in Figures 10g, i, and k. Note in Figure 11a that the capacities of arcs (s,2) and (2,t) are exceeded by 4 and 3 respectively, and these are precisely the modified cost coefficients of σ_{s2} and σ_{2t} in Figure 10g. In other words, the modified cost coefficient of σ_{ij} equals $f_{ij} - c_{ij}$, which is just $-c'_{ij}$. A comparison of Figures 10h and 11a, 10j and 11c, etc. shows that primal and dual basis trees do not intersect (complementary slackness holds).

Figure 11b gives the Big-M solution corresponding to Figure 11a. Note that the infeasible primal solution associated with the dual simplex method is obtained from the Big-M solution by routing the flow on (s,t) along the current shortest s - t path in the primal, thus violating the capacities of arcs along that path. The arc whose capacity is most violated is the first arc to be saturated as flow is increased. This arc is pivoted out, yielding the same cut as in the Big-M method.

The Big-M node numbers and modified costs across the cut after (s,2) is pivoted out are shown in Figure 11d. The Big-M modified costs for forward arcs across the cut in h^f differ from the modified costs for the associated dual simplex solution shown in Figure 11f by a constant $(\pi_t - M)$. Therefore the methods can select the same incoming arc, and will perform the same pivot sequence on non-degenerate problems.

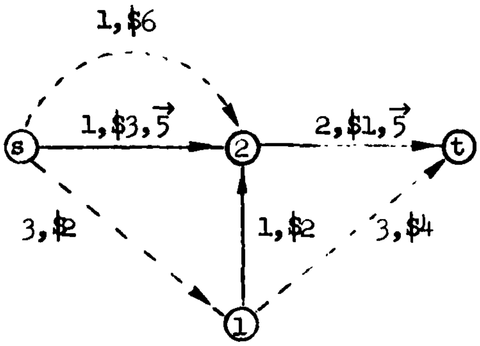


Figure 11a: Infeasible primal basis corresponding to Figure 10g.

Dotted arcs are non-basic with zero flow.

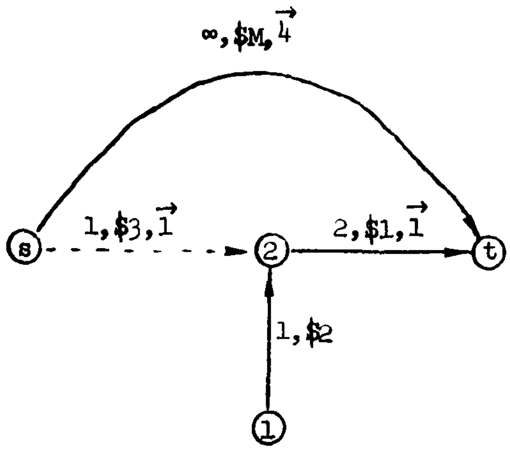


Figure 11b: Big-M basis corresponding to Figure 11a.

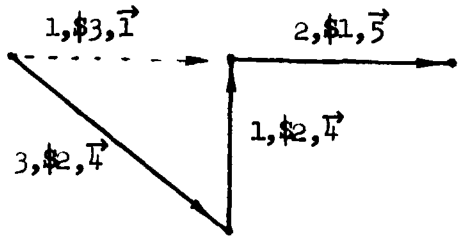


Figure 11c: Infeasible primal basis corresponding to Figure 10h.

Only arcs with flow are shown.

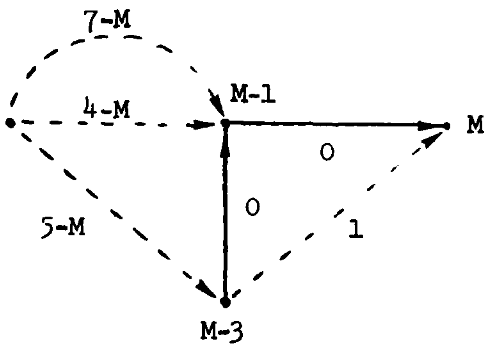


Figure 11d: Node numbers and modified costs for Figure 11b.

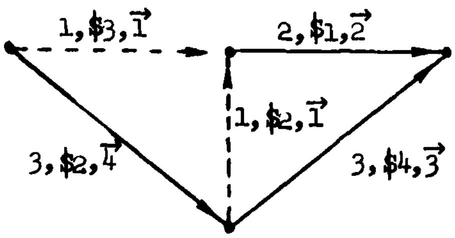


Figure 11e: Infeasible primal basis corresponding to Figure 10i.

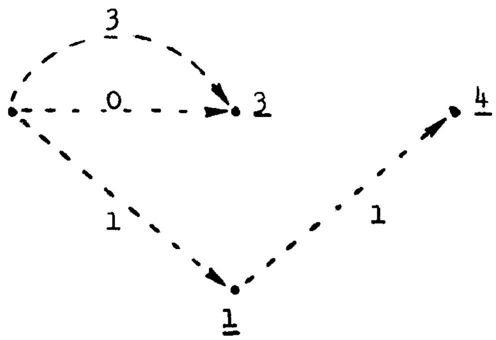


Figure 11f: Node numbers and modified costs when $M = 4$, the length of the current shortest path.

Observe that the computation required to determine the outgoing arc is order $O(n)$, where n is the number of nodes, whereas the computation required to determine the incoming arc is order $O(n^2)$, since there may be as many as $\frac{n^2}{4}$ arcs in a cut. This explains why Dual Simplex (best empirical) does not seem to be as good as Simplex (best empirical). By not searching for the best arc to remove, one saves at most $O(n)$.

Dual Simplex should be computationally similar to Out-of-Kilter since both find the best arc in each cut. This coincides with computational results [11].

Figure 12 shows how the dual problem may be formulated as a minimum cost flow problem when the primal network is planar [2]. Increases in potential across cuts may be viewed as augmentations of potential from s^* to t^* in the dual graph. Saturations in the dual graph occur when a positive modified cost goes to zero. Note that each arc in the primal has two arcs associated with it in the dual, a slack arc with zero cost and capacity equal to d_{ij}, and an arc with cost equal to c_{ij} and infinite capacity corresponding to σ_{ij}.

The following theorem allows a rigorous characterization of dual bases, and was communicated to the author by Robert Bland.

Theorem 2: Let $\hat{\pi}_i$, $\hat{\sigma}_{ij}$, and $\hat{s}_{ij}$ represent the dual column corresponding to the variable π_i, σ_{ij}, and s_{ij} respectively. Then a set of columns $\{\hat{\pi}_i\}_{i\neq s} \cup \{\hat{\sigma}_{ij}\}_{(i,j)\in I} \cup \{\hat{s}_{ij}\}_{(i,j)\in K\subset I^c}$ is independent in the dual system if the edges in $I \cup K$ do not contain any cutsets.

Proof: Note that the column corresponding to π_i contains 1's for arcs leading into i and -1's for arcs leading out. In other words, it represents the cut $(N \sim i, i)$. Similarly, $\hat{\sigma}_{ij}$ and $\hat{s}_{ij}$ have one 1 or -1, they represent the arc (i,j). Any non-zero linear

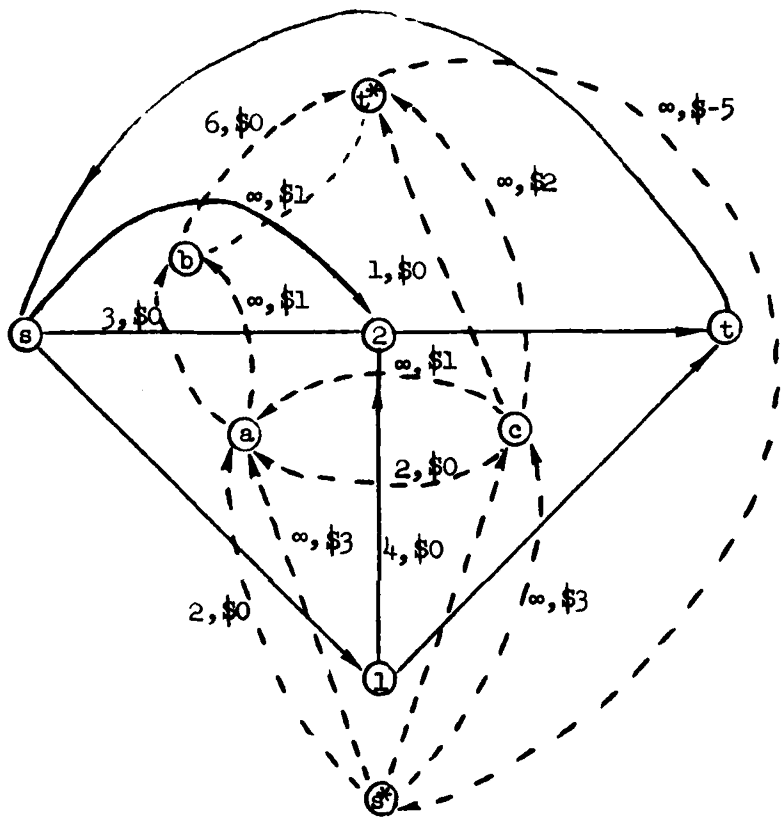


Figure 12: Viewing the Dual of a Planar Problem as a Minimum Cost Circulation Problem

combination $\sum_{i \in X} \alpha_i \hat{\pi}_i$ of columns in $\{\hat{\pi}_i\}_{i \neq s}$ will have non-zero elements for every forward and reverse arc in the cut $(X,\bar{X})$. Hence a set of columns $\{\hat{\pi}_i\} \cup \{\hat{\sigma}_{ij}\} \cup \{\hat{s}_{ij}\}$ can not be dependent if the edges corresponding to $\{\hat{\sigma}_{ij}\} \cup \{\hat{s}_{ij}\}$ do not contain any cutsets. ⊠

The following definition gives a precise correspondence between primal and dual bases. Recall that A is the set of arcs and that at most one arc is assumed to exist between any pair of nodes.

Definition: Let $B = \mathcal{S} \cup \mathcal{T} \cup (s,t)$ be a basis tree for the Big-M method with (s,t) the only artificial arc and let ℓ_1 denote the arc which just left the basis.

The corresponding primal infeasible basis associated with the Dual Simplex method is defined to be $D = \mathcal{S} \cup \mathcal{T} \cup \ell_1$. Let $\pi_T(i)$ denote the shortest distance from s to i in a tree T.

The basic dual feasible solution associated with D is denoted by $\tilde{D}$, and defined as the tuple $\{\lambda,\{\pi_i\},\{s_{ij}\},\{\sigma_{ij}\}\}$, where

$\pi_i = \pi_D(i) \quad \forall\, i \in N \sim s.$

$\lambda = \pi_t.$

(*) $s_{ij} = \pi_i + d_{ij} - \pi_j$ if positive and $(i,j) \in A.$

$\sigma_{ij} = -\pi_i - d_{ij} + \pi_j$ if positive and $(i,j) \in A.$

$\sigma_{ij} = 0$ if $\pi_i + d_{ij} - \pi_j = 0,\ f_{ij} = c_{ij}$ and $(i,j) \notin D.$

$s_{ij} = 0$ if $\pi_i + d_{ij} - \pi_j = 0,\ f_{ij} = 0$ and $(i,j) \notin D.$

All other variables are non-basic and equal zero.

Remark: It is easy to verify that λ, $\{\pi_i\}$, $\{s_{ij}\}$, $\{\sigma_{ij}\}$ is feasible for the dual system. It is basic feasible because the columns associated with it contain no columns of D and hence are independent by Theorem 2.

The following lemma observes that modified costs relative to a Big-M basis differ from modified costs associated with the corresponding dual solution only along the cut (S,T), where they differ by a constant.

<u>Lemma 1</u>: Let D be a tree with node numbers $\{\pi_D(i)\}$. Suppose that arc ℓ is deleted and arc (s,t) is added with cost M, yielding a new tree $B = \mathcal{S} \cup \mathcal{T} \cup (s,t)$. Let $d'_D(i,j)$, respectively $d'_B(i,j)$ denote the modified cost of arc (i,j) with respect to $\{\pi_D\}$ respectively $\{\pi_B\}$. Then we have for $(i,j) \in A$,

$$i \in S,\ j \in T \Rightarrow d'_B(i,j) = d'_D(i,j) + (\pi_D(t) - M) .$$
$$i \in T,\ j \in S \Rightarrow d'_B(i,j) = d'_D(i,j) - (\pi_D(t) - M) .$$

and for every $(i,j) \notin (S,T)$,

$$d'_B(i,j) = d'_D(i,j) \geq 0 .$$

This implies that for every forward arc (i,j) in the cut (S,T) in N^f, we have

$$d'_B(i,j) = d'_D(i,j) + (\pi_D(t) - M) .$$

<u>Proof</u>: The first part follows by noting that for $M = \pi_D(t)$, the node numbers and modified costs relative to respective trees are identical. As M is increased to $\pi_D(t) + \delta$, $\delta > 0$, the node numbers of nodes in T are increased by δ, and the modified costs of arcs across the cut are affected as claimed. For the second part, note

that if $(i,j) \in N^f$, $(i,j) \in (S,T)$, and $(j,i) \in A$, then

$$d'_B(i,j) = -d'_B(j,i) = -d'_D(j,i) + (\pi_D(t) - M) = d'_D(i,j) + (\pi_D(t) - M) . \quad \boxtimes$$

<u>Theorem 3</u>: Let $B_1 = \mathcal{S}_1 \cup \mathcal{T}_1 \cup (s,t)$ be a basis tree for the primal with (s,t) as the only artificial arc, and let ℓ_1 denote the arc which just left. Suppose that e_1 now enters, ℓ_2 leaves and the new basic tree is $B_2 = \mathcal{S}_2 \cup \mathcal{T}_2 \cup (s,t)$. Let $D_1 = \mathcal{S}_1 \cup \mathcal{T}_1 \cup \ell_1$, and let $D_2 = \mathcal{S}_2 \cup \mathcal{T}_2 \cup \ell_2$. Then $\tilde{D}_2$ may be obtained from $\tilde{D}_1$ by pivoting the appropriate dual variable associated with ℓ_1 into the dual and pivoting out the variable associated with e_1.

<u>Proof</u>: Since ℓ_1 is in D_1, the appropriate dual variable associated with ℓ_1, namely s_{ℓ_1} if $f_{\ell_1} = 0$, σ_{ℓ_1} if $f_{\ell_1} = c_{\ell_1}$, can enter $\tilde{D}_1$. Combining Observation 3 with Lemma 1, the dual variable associated with e_1 will be first to leave.

The primal tree associated with the current dual solution is $D_2 = \mathcal{S}_2 \cup \mathcal{T}_2 \cup \ell_2$, and the dual solution may be verified to be $\tilde{D}_2$. $\boxtimes$

<u>Theorem 4</u>: On problems with non-negative arc costs, the simplex method with a most negative pivot rule and Big-M start and Lemke's Complementary Pivot algorithm act similarly in the following sense. Given the same start, the methods can generate the same sequence of primal solutions, and will do so when the problem is non-degenerate.

<u>Proof</u>: Suppose $\mathcal{S} \cup \mathcal{T} \cup (s,t)$ is the current Big-M basis with ℓ_1 having just left and (s,t) is the only artificial arc. Then an equivalent primal basis for Lemke's algorithm may be obtained by adding f_{ts}, c'_{ts}, and setting $z_0 = \frac{f_{st}}{v}$.

The node numbers to the dual complementary system are taken equal to $\pi_D(i)$, where $D = \mathcal{S} \cup \mathcal{J} \cup \ell_1$, and s_{ij} and σ_{ij} defined according to (*).

Both methods can now determine the same incoming variable and execute the same primal pivot. The result follows by repeated application of Theorem 3. ⊠

Remark: Any primal solution with artificial arcs other than (s,t) may be replaced by an equivalent solution with those arcs eliminated as in Figure 4c. Lemke's algorithm will simulate the Big-M Method when there are multiple artificial arcs if the primal formulation is changed to that of Figure 8d and the corresponding dual solution modified accordingly.

Negative Costs

The following example shows that Big-M Simplex (most negative), Out-of-Kilter, and Lemke's Complementary Pivot Algorithm act differently on problems with negative costs.

Big-M Simplex (most negative) first augments over path s2t. Out-of-Kilter, if trying to bring (t,s) into Kilter, would set all negative costs to zero [19], and augment along s1t. Lemke's method sets the initial costs of (s,2) and (1,t) to $\theta - 3$ and $\theta - 1$, respectively, where θ is initially 8. Consequently, Lemke's method augments over s12t first.

As noted by George Dantzig, Lemke's Method is equivalent to the Self Dual Algorithm [6] on linear programs.

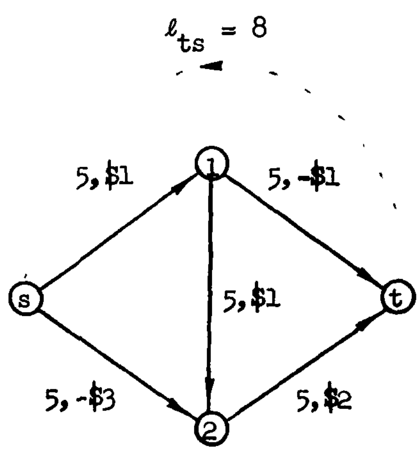


Figure 13: A network problem for which Big-M Simplex (most negative), Out-of-Kilter, and Lemke's method act differently.

ACKNOWLEDGEMENT

The author would like to thank Robert Bland, B. Curtis Eaves, and Richard Stone for their helpful comments.

REFERENCES

1. Barr, R. S., F. Glover, and D. Klingman, "An Improved Version of the Out-of-Kilter Method and a Comparative Study of Computer Codes," Mathematical Programming 7(1974), 60-86.

2. Bland, R., "A Class of Cographic Linear Programming Problems Related to Production Planning," to be published.

3. Bradley, G. H., G. G. Brown, and G. W. Graves, "Design and Implementation of Large Scale Primal Transhipment Algorithms," Management Science, 24, No. 1 (1977), 1-34.

4. Cottle, R.W., and G. B. Dantzig, "Complementary Pivot Theory of Mathematical Programming," Linear Algebra Appl., 1(1968), 103-125.

5. Cunningham, W. C., "A Network Simplex Method," Mathematical Programming, 11(1976), 105-106.

6. Dantzig, G. B., Linear Programming and Extensions, Princeton University Press, Princeton, N.J. (1963).

7. Eaves, B. C., "A View of Complementary Pivot Theory," Constructive Approaches to Mathematical Models, (Coffman and Fix, eds), Academic Press, (1980).

8. Edmonds, J., and R. M. Karp, "Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems," J. A. C. M. 19 (1972), 248-264.

9. Ford, L. R., Jr., and D. R. Fulkerson, Flows in Networks, Princeton University Press, Princeton, New Jersey, 1962.

10. Fulkerson, D. R., "An Out-of-Kilter Method for Minimal Cost Flow Problems," SIAM J. Appl. Math., 9(1961), 18-27.

11. Glover, F., D. Karney, and D. Klingman, "Implementation and Computational Comparisons of Primal, Dual, and Primal-Dual Computer Codes for Minimum Cost Network Flow Problems," Networks, 20(1974), 191-212.

12. Johnston, E. L., "Networks and Basic Solutions," Operations Research, 14(1966), 619-624.

13. Johnson, E. L., "Flows in Networks," in Handbook of Operations Research, J.J. Moder and S.E. Elmaghraby editors, Van Nostrand Reinhold & Co. (1978), 183-206.

14. Lawler, E., Combinatorial Optimization, Holt, Rinehart, and Winston, (1976), 206-207.

15. Lemke, C. E., "Bimatrix Equilibrium Points and Mathematical Programming," Management Science 11(1965), 681-689.

16. Mulvey, J. V., "Testing of a Large-Scale Network Optimization Program," Mathematical Programming 15(1978), 291-314.

17. Van De Panne, C., "A Complementary Variant of Lemke's Method for the Linear Complementarity Problem," Math. Programming 7 (1974), 283-310.

18. Zadeh, N., "A Bad Network Problem for the Simplex Method and other Minimum Cost Flow Algorithms," Mathematical Programming, 5(3), (1973), 255-266.

19. Zadeh, N., "A Simple Alternative to the Out-of-Kilter Algorithm," Stanford Operations Research Technical Report No. 25, May 1979.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: 36	2. GOVT ACCESSION NO.: AD-A088 032
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): NEAR-EQUIVALENCE OF NETWORK FLOW ALGORITHMS	5. TYPE OF REPORT & PERIOD COVERED: TECHNICAL REPORT
	6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): NORMAN ZADEH	8. CONTRACT OR GRANT NUMBER(s): N00014-75-C-0493
9. PERFORMING ORGANIZATION NAME AND ADDRESS: DEPARTMENT OF OPERATIONS RESEARCH, STANFORD UNIVERSITY, STANFORD, CALIFORNIA	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: NR-042-264
11. CONTROLLING OFFICE NAME AND ADDRESS: LOGISTICS AND MATHEMATICAL STATISTICS BRANCH, OFFICE OF NAVAL RESEARCH, ARLINGTON, VIRGINIA	12. REPORT DATE: DECEMBER 1, 1979
	13. NUMBER OF PAGES: 50
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

This document has been approved for public release and sale. Its distribution is unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Also partially supported by National Science Foundation Grant ENG76-12266

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Network Flows, Nonnegative unit costs, Equivalence of Algorithms, Simplex method, Out-of-kilter method, Dual simplex method, Lemke's method

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

See Reverse Side

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

S/N 0102-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Near-Equivalence of Network Flow Algorithms

Abstract

Network problems arising in practice typically have non-negative arc costs. On such problems we show that the following algorithms perform, modulo ties, the same sequence of flow augmentations.

Simplex (with the standard pivot rule and Big-M start),

Out-of-Kilter (Primal-Dual),

Dual Simplex (with the standard pivot rule),

Lemke's Complementary Pivot Algorithm.

All methods compute a shortest path tree by mimicking the Dijkstra algorithm and then send flow along a sequence of minimum cost paths. Differences in implementation are discussed. It becomes clear that Dantzig's simplex method with the best empirical pivot rule (not the standard rule) will outperform other methods (variations of Simplex with the standard rule). A simple reason is given why Dual Simplex (best empirical) cannot do as well as Simplex (best empirical). It is noted that network flow problems from [18] represent pathological examples for all the above methods.

ILMED

-8