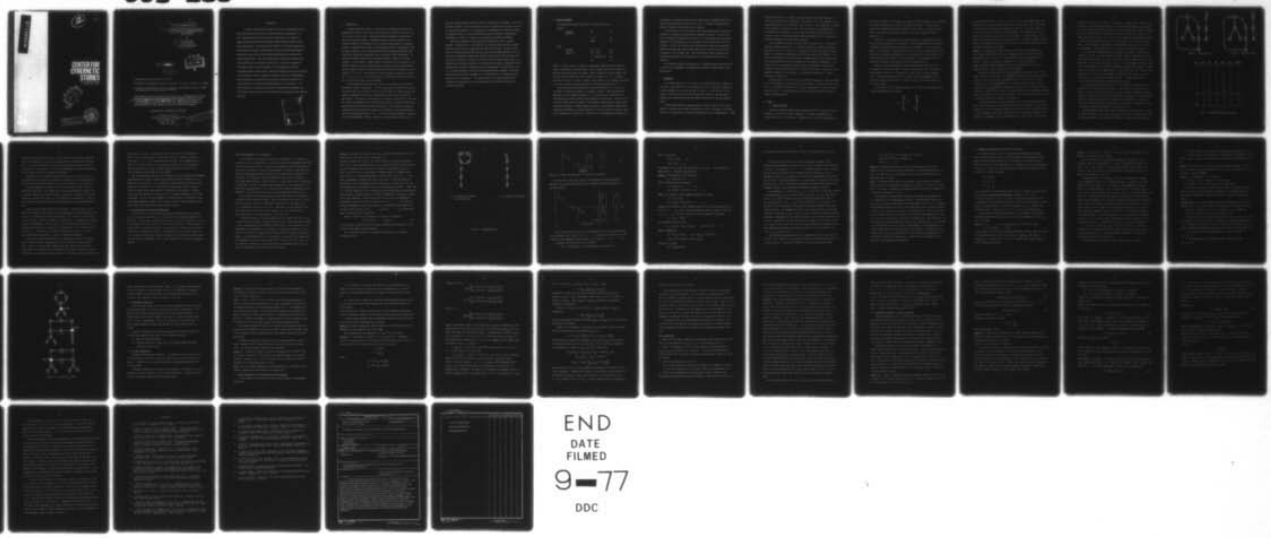
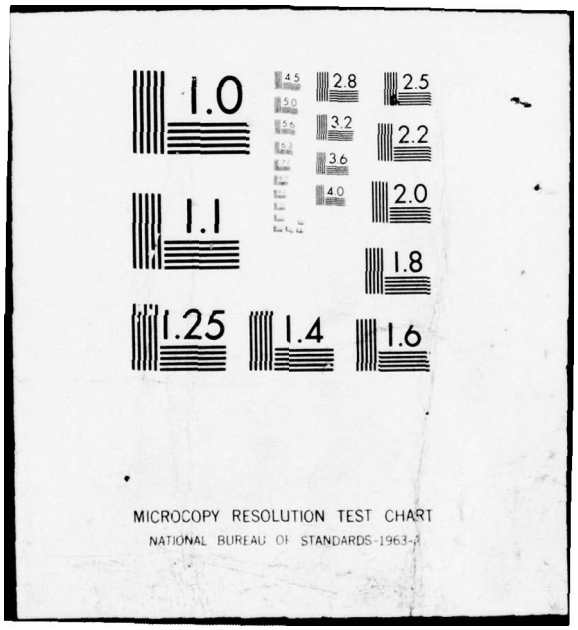
AD-A044 115 TEXAS UNIV AT AUSTIN CENTER FOR CYBERNETIC STUDIES F/G 9/2
A STRONGLY CONVERGENT PRIMAL ALGORITHM FOR GENERALIZED NETWORKS--ETC(U)
JAN 77 J ELAM, F GLOVER, D KLINGMAN N00014-75-C-0616

UNCLASSIFIED CCS-288 NL

| OF |
AD
A044115



END
DATE
FILMED
9-77
DDC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

(12)

CENTER FOR CYBERNETIC STUDIES

The University of Texas
Austin, Texas 78712

(9) (14) Research Report CCS-288

(6) A STRONGLY CONVERGENT PRIMAL ALGORITHM FOR GENERALIZED NETWORKS,

by

(10) Joyce/Elam*
Fred/Glover**
Darwin/Klingman***

(11) January 1977

(12) 47 p.

*University of Texas, Austin, TX 78712.

**Professor of Management Science, University of Colorado, Boulder, CO 80302.

***Professor of Operations Research, Statistics, and Computer Sciences, BEB 608, University of Texas, Austin, TX 78712.

This research was partly supported by ONR Contract N00014-76-C-0383 with Decision Analysis and Research Institute and by Project NR047-021, CNR (15) Contracts N00014-75-C-0616 and N00014-75-C-0569 with the Center for Cybernetic Studies, The University of Texas. Reproduction in whole or in part is permitted for any purpose of the United States Government.

CENTER FOR CYBERNETIC STUDIES

A. Charnes, Director
Business-Economics Building, 203E
The University of Texas
Austin, TX 78712
(512) 471-1821

406 197

ABSTRACT

A major computational problem that arises in the attempt to solve generalized network and network-related problems is degeneracy. In fact, using primal extreme point solution techniques, the number of degenerate pivots performed frequently ranges as high as 90% in large-scale applications. The purpose of this paper is to develop a special set of structural and logical relationships for generalized network problems that yields a new primal extreme point algorithm which avoids and/or exploits degeneracy. The major mathematical differences between this new algorithm and the simplex method are (1) each basis examined is restricted to have a special topology, (2) the algorithm is finitely convergent without reliance upon "external" techniques such as lexicography or perturbation, and (3) a special screening criterion for nondegenerate basis exchanges is available that allows some of these exchanges to be recognized prior to finding the representation of an incoming arc. For these reasons, this algorithm has several computational advantages over the simplex-based codes recently developed for solving generalized network problems.

1. INTRODUCTION

A generalized network (GN) is a capacitated or uncapacitated linear programming (LP) problem in which the coefficient matrix contains at most two non-zero entries in each column. Generalized networks encompass a broad range of important practical problems. Depending upon the non-zero components of the coefficient matrix, a GN problem can be a shortest path problem, an assignment problem, a transportation problem, a transshipment problem, a generalized transportation problem, or even an optimization problem involving generalized upper bound (GUB) conditions coupled with a set of disjoint linear constraints. The procedures presented subsequently are general enough to handle any of these problems. To this extent, a higher degree of efficiency may be achieved for the simpler problems by further exploiting their special structural characteristics. The more complex variants of the GN class have received the attention of several studies [2, 3, 4, 11]. The prevalence of these problems in practical settings [6, 7, 8, 9, 10, 12, 17, 20, 21] has created a need for new theoretical results that can provide efficient solution methods.

A major computational problem that arises in the attempt to solve GN and GN-related problems is degeneracy. In fact, using primal extreme point solution techniques, which are currently the most efficient known for these problems, the number of degenerate pivot steps is frequently as high as 90% in large-scale applications. The purpose of this paper is to develop a special set of structural and logical relationships for GN problems that yields a new primal extreme point algorithm which avoids and/or exploits degeneracy. The conceptual foundations underlying this algorithm may be viewed as an extension of those underlying the recently developed algorithms of [2, 3, 4, 11] for solving assignment, transportation, and transshipment problems. However, the more complex structure of gen-

eralized networks requires several theoretical departures and advances. One of the principal features of the new algorithm, shared in common with its earlier cousins for less general problems, is a strong form of convergence that limits the number of degenerate steps in a far more powerful way than achieved by "lexicographic improvement" (as used, for example, in customary LP perturbation schemes).

Each basis examined by this algorithm is restricted to have a special topology. We show that if a GN problem has an optimal solution, then an optimal solution can be found by considering only bases of this type. The major mathematical differences between this new algorithm and the simplex method are (1) the rules of the algorithm automatically (without search) ensure that all bases have the special topological structure, and bypass all other bases normally given consideration by the simplex method; (2) the algorithm is finitely convergent without reliance upon "external" techniques (such as lexicography or perturbation); and (3) a special screening criterion for nondegenerate basis exchanges is available that allows some of these exchanges to be recognized prior to finding the representation of an incoming arc. For these reasons, this algorithm has several computational advantages over the simplex-based codes recently developed for solving GN problems.

2. PROBLEM STATEMENT

The *generalized network* problem may be defined as follows:

Primal

Minimize	$C^T X$	(1)
Subject to		
	$AX = b$	(2)
	$0 \leq X \leq U$	(3)

Dual

Maximize	$\pi^T b - \gamma^T U$	(4)
Subject to	$\pi^T A - \gamma \leq C^T$	(5)
	π - unrestricted	(6)
	$\gamma \geq 0$	(7)

where $U > 0$ and A is an m x n matrix in which each column contains at most two nonzero coefficients of opposite sign. We will assume that one coefficient is -1 and the other any positive real number. When a column has only one nonzero coefficient, we will assume that this coefficient is either 1 or -1. A variable associated with such a column is called a *slack variable* if its nonzero coefficient is 1, and is called a *surplus variable* if its nonzero coefficient is -1.

The most efficient procedures for solving GN problems [13, 14, 19, 22, 24] are based on viewing the problem in a graphical context. These procedures are adaptations of the simplex algorithm [1, 7, 12] in which the A matrix and the basis matrix are stored as graphs using computer list structures. The use of such structures reduces both the amount of work needed to perform basic simplex operations and the amount of computer memory required to store essential problem data. Computation and memory are further reduced if A does not have full row rank, since such GN problems are equivalent to a disjoint set of transshipment problems [15].

In addition, the graphs contain only the nonzero matrix components which allow special graph labeling procedures [18, 22, 24] to eliminate checking and unnecessary arithmetic operations on zero elements.

The development of the algorithm in this paper demonstrates an additional advantage of representing and solving GN problems graphically. We will show that the graphical representation allows one to fully characterize the nonzero elements of the representation of any nonbasic vector and the signs of these elements. We will also show that the representation allows one to explicitly characterize which dual values change and how these values are altered after performing a basis exchange step. These mathematical characterizations and visualizations provide the cornerstones of the development and proof of our algorithm.

The essential concepts and definitions of elementary graph theory that will be used in our development are presented and related to the GN problem in the next section.

3. BACKGROUND

A *directed graph* G(N, E) is a finite set of nodes N and arcs E connecting the nodes. Each element of E is an ordered pair (i, j) of distinct elements of N which represents an arc from the node i to node j. If the arc set E is expanded to contain arcs which have both endpoints incident on the same node, or multiple arcs connecting the same two nodes, then G(N, E) is called a *general graph.*

The GN problem defines a general graph as follows. Each row of A corresponds to a node and each column corresponds to an arc of the graph. The nonzero entries in a column will be called the multipliers of the corresponding arc. Asso-

ciated with each arc is a variable, an upper bound, an objective function coefficient, and the nonzero multipliers. The variable (i.e., the component of X) associated with the arc is called the *flow* on the arc. (Or, frequently, the *value* of this variable is called the flow.) The row positions of the multipliers in the column that corresponds to the arc identify the nodes on which the arc is incident. The arc is directed from the node associated with the -1 multiplier to the node associated with the positive multiplier.

An arc directed from node i to node j will be denoted (i, j). The positive multiplier associated with arc (i, j) will be denoted a_{ij}. (Since there may be more than one arc (i, j) from i to j, technically a third index should be used; however, for notational convenience, the third subscript will be omitted. The method as subsequently described provides an organization by which multiple arcs with unique multipliers, costs and capacities, are readily accommodated.) If a column has only one multiplier (nonzero entry) both endpoints of the arc are incident on the same node and the arc is said to form a *loop*. Such an arc, with endpoints incident on the same node i, is customarily denoted (i, i) and its multiplier is denoted a_{ii}.

The right-hand side vector b for the GN problem associates a *node requirement* b_k (the k^{th} component of b) with node k of the network. (Each unit of flow on an arc (i, j) therefore "contributes" -1 and a_{ij}, respectively, to the node requirements b_i and b_j.) In addition, each node k has an associated dual variable π_k, called its *node potential*.

4. BASIS

4.1 Basis Structure

Using the standard bounded variable simplex algorithm, a basis B for the GN problem is a full row rank matrix composed of a linearly independent set of column vectors of A (or possibly of A augmented by unit vectors corresponding to

artificial variables). The variables associated with the columns of B are considered to be basic variables and all others are nonbasic variables. A *basic solution* consists of assigning each nonbasic variable a value equal to its lower or upper bound. Thereupon, a unique value results for each basic variable to satisfy equation (2).

A basis for a GN problem may be viewed and stored as a graph which contains only the nonzero components of the matrix B. The characteristics of the graph corresponding to B have been fully outlined in the literature [1, 12, 23] when the graph of A is bipartite. Several authors [18, 22, 24] have further noted that the structure of the graph corresponding to B is essentially the same (except for arc orientations) when the graph of A is not bipartite. For completeness we will rigorously characterize the structure for the non-bipartite case.

Clearly, by the correspondences already indicated, any basis for the GN problem may be viewed as a graph. The basis graph contains all of the nodes but only a subset of the arcs of the graph of A, and, hence is called a *spanning graph*. A spanning graph consists of one or more connected subgraphs, where a *connected graph* (hence subgraph) is a graph in which a path exists between any two distinct nodes.

Let B_i denote the i^{th} connected subgraph of B. Then B can be represented as a block diagonal matrix as follows:

$$B = \begin{bmatrix} B_1 & & & \\ & B_2 & & \\ & & \ddots & \\ & & & B_n \end{bmatrix}$$

By the non-singularity of B, each B_i is a square $n_i \times n_i$ non-singular matrix and consequently is a connected graph consisting of n_i nodes and n_i arcs. Therefore, it follows that the subgraph corresponding to each B_i is a *quasi-tree* [18], i.e., a simple tree to which a single arc has been added. The additional arc creates exactly one *loop* (a circular path) in the quasi-tree. As noted earlier, an arc having both endpoints incident on the same node (i.e., a slack or surplus variable) forms a loop henceforth referred to as a *self-loop*. The basis structure may thus be simply characterized as follows.

Remark 1: By arranging rows and columns, any basis B for a GN problem is a block diagonal matrix whose blocks are triangular except for rows and columns corresponding to loop arcs (i.e., arcs belonging to loops associated with the blocks). A block is completely triangular if its loop is a self-loop.

Proof: The proof follows immediately from the fact that a basis B for a GN problem consists of one or more disjoint quasi-trees.

An efficient method called the *extended augmented predecessor index* (EAPI) method [18] has been developed for storing and updating the basis graph of the GN problem. One of the advantages of the EAPI method is its ability to find the representation of an entering vector and to determine updated values for dual variables by tracing appropriate segments of the quasi-trees of the basis graph. Therefore, no explicit basis inverse is needed, and the usual arithmetic operations required to update the inverse are eliminated. (The accumulation of roundoff errors is likewise substantially reduced or eliminated).

The EAPI method also plays an important role in the statement and development of the new extreme point algorithm presented in this paper. Its special updating rules can be conveniently expressed in terms of a set of operations applied to Ellis Johnson's triple label representation for recording and tracing a tree, adapted to accommodate structural peculiarities of a quasi-tree. The approach

assigns three labels to each node of a quasi-tree: a predecessor, a successor, and a brother. This triple-label scheme is first applied to the portion of each quasi-tree that results by suppressing all loop arcs of the quasi-tree so that the labeled portion is a disjoint set of rooted trees. The root of each tree consists of the node that lies on the loop; that is, the arcs are oriented by the predecessor labels so that the unique path from any node to the root node of the tree is a directed path. (In this manner the predecessor labels uniformly orient the tree to create an arborescence.) Notationally, we let p(i) denote the predecessor of node i.

The triple-label representation may be viewed as inducing an "ancestry relationship" on a tree, each node carrying three node indexes, which name the father (predecessor), the eldest son ("first" successor), and next younger brother of the given node. A node is taken to be the father of all its immediate successors, these latter constituting a set of brothers, arbitrarily sequenced from eldest to youngest. Thus, the root node is the common ancestor of all nodes in the tree. Nodes at the extremities of the tree have no immediate successors and the "last" of a set of successors of a given node has no younger brother. In these extreme cases, a "dummy" name which communicates their nonexistence is used.

The EAPI method connects these trees to the remainder of the quasi-trees in which they lie by reinstating the loop arcs. Then the root of each tree is assigned a predecessor which orients these loop arcs uniformly clockwise or counterclockwise. Each node on the loop thus becomes its own ancestor. (A self-loop may simply be assigned the orientation it receives in the graph of A.) Hence, each loop node has an equal status as an ancestor of all nodes in the quasi-tree. This is called the *rooted loop* orientation of a quasi-tree [18].

The basis graph illustrated in Figure 1 consists of two quasi-trees. The arcs are drawn according to their *actual direction* in the graph of A rather than to the orientation imparted by the predecessor indexing. The basis matrix B associated

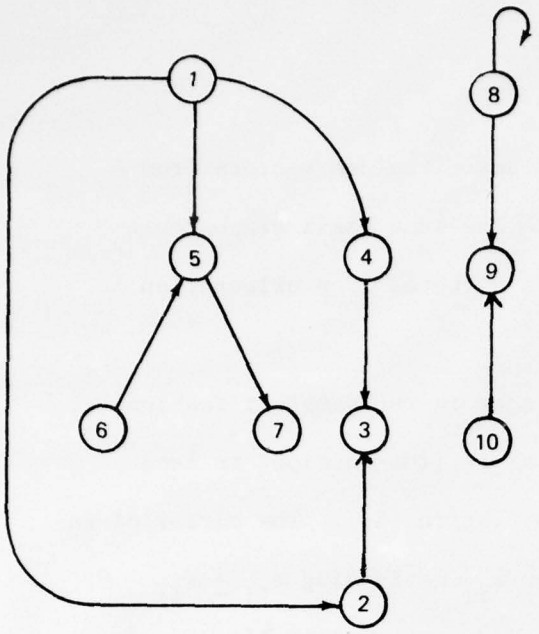


Figure 1 - Basis Graph of A

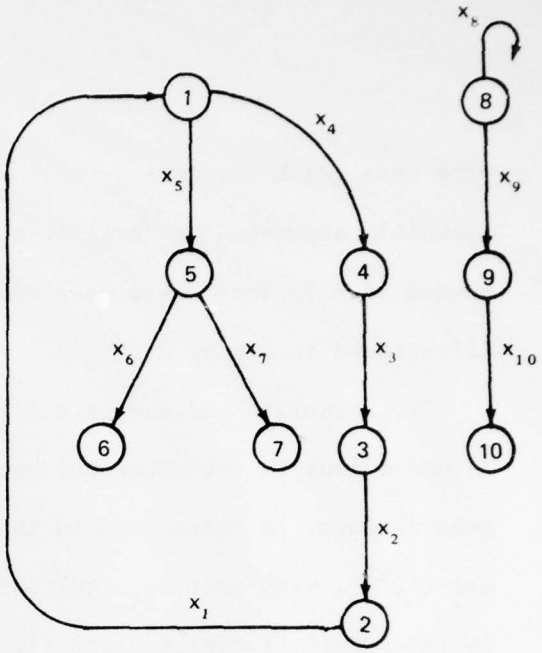


Figure 2 - EAPI Basis Graph

NODE	PRED	SUC	BRO	FLOW	MULT	MIRROR
1	2	5	0	0	1/3	0
2	3	1	0	-1	1	1
3	4	2	0	0	1	0
4	1	3	0	8	2	0
5	1	6	4	1/2	1/4	0
6	5	0	7	-2	1	1
7	5	0	0	1/2	2	0
8	8	9	0	1/3	1	0
9	8	10	0	4	1/2	0
10	9	0	0	-2	1	1

Table I - EAPI Representation of a Basis

with this graph consists of a linearly independent set of column vectors from A (possibly augmented by "artificial" unit vectors). The same basis graph represented in EAPI form where each quasi-tree possesses a rooted loop orientation is illustrated in Figure 2.

To formulate and derive the results of this paper in the simplest fashion, it is convenient to introduce the notion of a *mirror arc*. (This notion, in less general form, is introduced in the pure network setting in [5]). The mirror of an arc (i, j), with cost c_{ij}, multiplier a_{ij}, and flow x_{ij} satisfying $u_{ij} \geq x_{ij} \geq 0$, is an oppositely directed arc (j, i) with cost $-c_{ij}/a_{ij}$, multiplier $1/a_{ij}$, and flow x_{ji} satisfying $-u_{ij}\, a_{ij} \leq x_{ji} \leq 0$. The mirror arc is clearly the result of a simple linear transformation by which $x_{ji} = -x_{ij} \cdot a_{ij}$. Viewed alternatively, the mirror arc arises by a scaling operation that divides the column of A associated with the variable x_{ij} by the quantity $-a_{ij}$. Thus the multiplier a_{ij} that lies in row j is changed to -1, indicating that the mirror arc will be directed out of, rather than into, node j. Similarly, the -1 in row i becomes $1/a_{ij}$, identifying this to be the multiplier of the new arc. The scaling is applied correspondingly to the appropriate components of the vectors c and u. Note that there is a convenient reciprocal relationship between an arc and its mirror. That is, if the same type of scaling process is applied a second time, we discover that *the mirror of the mirror is the original arc*, or in other words, each arc is the mirror of the other. (By further employing the linear translation $x_{ij} = -x_{ji}/a_{ij} + u_{ij}$, both arcs would be constrained to nonnegative flows. We do not bother with the translation since it requires an extra step and is artificial for arcs with infinite capacities.)

As a consequence of these remarks, an arc may be replaced (conceptually or literally) at any time by its mirror, provided the appropriate relationships are

observed. Thus, for example, a flow increase of 1 unit on arc (i, j) corresponds to a flow decrease of a_{ij} units on its mirror arc (j, i), and if the flow on arc (i, j) is y units below its upper bound, then the flow on the mirror arc (j, i) is ya_{ij} units above its lower bound. (As noted earlier, we allow the existence of multiple arcs between the same two endpoints, but continue to refer to a *given arc* from i to j as simply (i, j), suppressing a third subscript for notational convenience. [The word "given" implicitly provides the third subscript.])

The notion of a mirror arc can be applied to slack and surplus arcs by means of a refinement of the self-loop concept. A self-loop at a node i is conceived as joining node i to an *implicit copy* of i, denoted i#. A surplus arc at node i (the self-loop that has a -1 in the i^{th} position of its column) will be designated as (i, i#), and a slack arc at node i (the self-loop that has a 1 in the i^{th} position of its column) will be designated as (i#, i). *The multiplier for both a slack arc and surplus arc is taken to be 1.* Consequently, as a result of this refinement and the previous definitions, the mirror of an arc (i, i#), with cost $c_{ii\#}$, and flow $x_{ii\#}$ satisfying $u_{ii\#} \geq x_{ii\#} \geq 0$, is the arc (i#, i) with cost $c_{i\#i} = -c_{ii\#}$ and flow $x_{i\#i}$ satisfying $-u_{ii\#} \leq x_{i\#i} \leq 0$. (The mirror of an arc (i#, i) is an arc (i, i#) characterized in a like fashion.)

The advantage of introducing the notion of a mirror arc is twofold. The mirror arc concept makes it possible to ensure: (1) all nonbasic arcs have flows equal to their lower bounds, and (2) all basic arcs have the same direction as imparted by the orientation of the predecessor indexing. A nonbasic arc whose flow equals its upper bound or a basic arc whose direction is contrary to the predecessor orientation may simply be replaced by its mirror arc, thereby enabling both (1) and (2) to hold. It may be noted that a slack arc (i#, i) in the basis automatically is directed in accordance with the orientation of the predecessor indexing, while a surplus arc (i, i#) is directed contrary to this orientation, requiring the

latter to be replaced by its mirror. (This follows from the natural convention that the condition p(i) = i for a self-loop implicitly represents the condition p(i) = i#.) This replacement of arcs by their mirrors to give them a desired orientation or a flow equal to a lower bound, has important conceptual advantages, as will become clear in the subsequent development.

Hereafter, we will continue to use the term *mirror of an arc* to refer to an arc obtained from any given arc (original or otherwise) by the mirror operation, but will restrict the term *mirror arc* to refer to the mirror of an original arc.

From the foregoing, the basis arc (p(i), i), directed to node i from its predecessor node p(i), is an original arc if the associated basic column of A has a positive coefficient in position i and is a mirror arc, otherwise. The scaled matrix that results from a basis B in this fashion will be denoted B'. (That is, each column of B' gives the appropriate representation of the corresponding basis arc.)

Some notational simplification is now possible. Since each node i in the basis graph associated with B' has a unique predecessor, the unique multiplier for arc (p(i), i) (which may be either an original arc or a mirror arc) can be associated with node i and will, therefore, be denoted by a_i. In addition, the unique variable corresponding to each arc (p(i), i) can be associated with node i by reordering the columns of B' so that the i^{th} column vector in B' is associated with arc (p(i), i). Thus the basic variable associated with arc (p(i), i) in the basis graph will be denoted by x_i. This is illustrated in Figure 2.

Due to this scaling and reordering, the information required to represent a basis in EAPI form can be organized in an efficient manner as shown in Table I. Thus, in Table I, the array NODE contains node names and would not be stored. Arrays PRED, SUC, and BRO contain the predecessor, successor, and brother labels, respectively, associated with each node name. A zero label is used to indicate the

nonexistence of a label. Associated with each node i in the arrays FLOW and MULT is the flow and multiplier, respectively, on the arc (p(i), i) from the predecessor of i to i. Also associated with each node i in the MIRROR array is a value of 1 if the arc (p(i), i) is a mirror arc and 0 if it is an original arc. This organization is in effect a compact storage scheme for B since B can be created using the arrays PRED, MULT, and MIRROR.

The conventions described not only facilitate an effective computer programming implementation, but greatly simplify the notation required to express and prove a number of our main results. It is important to keep in mind that results derived graphically (in terms of arcs of the EAPI basis graph) have a direct algebraic analog (in terms of the basis matrix B' associated with this graph). Likewise, results derived algebraically by reference to B' translate immediately to the EAPI basis graph. We will find it convenient, at various points, to alternate the type of derivation employed, though we will continue to rely on the basis graph as the chief vehicle for motivating and elucidating our results.

4.2 Exchange Steps and Basis Representations

The procedures of an adjacent extreme point algorithm identify an *incoming arc* (p, q) and an *outgoing arc* (r, s), respectively, from among the nonbasic and basic arcs. It should be pointed out that (p, q) can refer to a slack arc (p = q#) or a surplus arc (q = p#), and (r, s) can refer to a slack arc. (The outgoing arc cannot refer to a surplus arc while it is in the basis, in order to retain proper orientation, as observed previously.) The addition of (p, q) and the removal of (r, s) from the current basis produces a changed set of quasi-trees, and this operation constitutes a fundamental step or "basis exchange" step of linear programming methods.

Basis Representation of an Incoming Arc

In any quasi-tree possessing a rooted loop orientation, it is clear that a sequential trace of predecessors of a given node generates a *predecessor path* which contains all arcs on the loop. The arcs of a predecessor path are encountered in the sequence determined by tracing from descendant to ancestor, rather than from ancestor to descendant (though the latter trace accords with the orientation of the basis). For simplicity, we shall suppose that such a path is simple; i.e., not traced beyond necessity. Hence, as soon as a node on this path is encountered a second time, the trace is stopped (which applies also to the situation $p(i) = i$, implicitly representing $p(i) = i\#$). The path itself therefore duplicates no arcs. Henceforth, P_i will denote the set of arcs in the predecessor path of node i.

We will follow the convention that the predecessor path for a node which is an implicit copy of another is empty. Thus, in particular, if an arc (i, j) is a self-loop then $P_j = \emptyset$ if (i, j) represents a slack arc ($i = j\#$) and $P_i = \emptyset$ if (i, j) represents a surplus arc ($j = i\#$). (Note that P_i is never empty for any node i that is not an implicit copy of another, since the predecessor path must always end in a loop that contains at least one arc.)

An important aspect of a basis exchange step is to identify the set of arcs on which flows will change, or more precisely, the set of basic arcs which provide a linear representation of the nonbasic incoming vector. The column vector corresponding to the incoming arc (p, q) when (p, q) does not correspond to a self-loop is of the form $-E^p + a_{pq}E^q$, where E^i denotes the i^{th} column of the identity matrix. If the incoming arc (p, q) corresponds to a self-loop, the column vector corresponding to this arc is of the form E^q (if $p = q\#$) or $-E^p$ (if $q = p\#$). It is, therefore, convenient to first characterize the representation of any unit vector E^i with respect to the arcs in the basis graph.

Remark 2: The basic arcs (vectors) that have a nonzero coefficient in the representation of a unit vector E^i are contained in P_i.

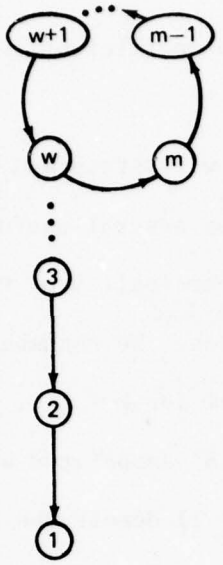
Instead of proving this remark in its present form, we will state and constructively prove a more encompassing result that illustrates several useful principles. To lay the foundation for this result, we suppose the cardinality of P_i is m. Then P_i is a set of m arcs which connect m distinct nodes. We renumber these nodes so that i = 1 and the predecessor of node k is k+1 for k = 1, . . . , m-1. (This corresponds to reordering the rows of the matrix B' associated with the basis graph.) For the renumbered nodes let node w ($w \geq 1$) denote the first loop node encountered in a sequential trace of the predecessors from node i. (Note that $m \geq w$ and the predecessor of m is the duplicate node encountered in the predecessor path from node 1 and thus is node w.) Figure 3 graphically illustrates P_i subject to this node numbering. We also suppose that the arcs in P_i are numbered correspondingly, so that the k^{th} arc is (p(k), k) for k = 1, . . . , m. Using the notation established in the previous section, the multipliers and columns of B' associated with these arcs are $a_1, \ldots, a_m$ and $B'_1, \ldots, B'_m$, respectively, where B'_k denotes the k^{th} column of B'.

Remark 2 asserts that there exists a solution to the equation

$$B'_1 Y_1 + B'_2 Y_2 + \ldots . B'_m Y_m = E^1 \qquad (8)$$

For our purposes, however, we will go beyond establishing the existence of such a solution and identify its form exactly.

For a self-loop, where m = w, equation (8) represents the following triangular system:



a) P_i contains a loop which is not a self-loop

b) P_i contains a self-loop

Figure 3. Illustration of P_i

$$\begin{bmatrix} a_1 & & & & & \\ -1 & a_2 & & & & \\ & -1 & \ddots & & & \\ & & \ddots & & & \\ & & & -1 & a_{w-1} & \\ & & & & -1 & \boxed{a_w} \\ & & & & & \text{loop portion} \end{bmatrix} \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ \vdots \\ \vdots \\ y_w \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ \\ \\ \\ 0 \end{pmatrix} \qquad (9)$$

Where $a_w = 1$ (since the multiplier of a self-loop is always 1).

For other than a self-loop, equation (8) represents the following which is identical to the preceding in the first w-1 components, but which differs over the "loop portion:"

$$\begin{bmatrix} a_1 & & & & & & & \\ -1 & a_2 & & & & & & \\ & \ddots & \ddots & & & & & \\ & & \ddots & -1 & a_{w-1} & & & \\ & & & & -1 & \boxed{\begin{matrix} a_w & & & -1 \\ & a_{w+1} & & \\ & & \ddots & \\ & & & -1 \quad a_m \end{matrix}} \\ & & & & & \text{loop portion} \end{bmatrix} \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_{w-1} \\ y_w \\ y_{w+1} \\ \vdots \\ y_m \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix} \qquad (10)$$

Let S_k = the set of the first k arcs encountered by a trace of the predecessor path from node 1 ($S_k = \{(p(i), i), i = 1 \ldots, k\}$) and let L = the set of arcs on the loop of the predecessor path from node 1 ($L = \{(p(i), i), i = w, \ldots, m\}$).

In general, for any set of arcs S, we define

$\theta(S)$ = 1/(the product of the multipliers of the arcs in S).

Thus, in particular

$$\theta(S_k) = 1/a_1 a_2 \ . \ . \ . \ a_k$$

$$\theta(L) = 1/a_w \ . \ . \ . \ a_m$$

The quantity $\theta(L)$ is called the *loop factor* of the loop L. (By convention, we define $\theta(\emptyset) = 1$, where $\emptyset$ is the empty set.)

Remark 3: The solution to (9) is given by

$$y_k = \theta(S_k) \text{ for } k = 1, \ . \ . \ . \ , w$$

and the solution to (10) is given by

$$y_k = \theta(S_k) \text{ for } k = 1, \ . \ . \ . \ , w-1$$

$$y_k = \theta(S_k)/(1 - \theta(L)) \text{ for } k = w, \ . \ . \ . \ , m$$

Proof: The solution to the triangular system (9) is clearly

$$y_1 = 1/a_1 = \theta(S_1)$$

$$y_k = y_{k-1}/a_k = \theta(S_k) \text{ for } k = 2, \ . \ . \ . \ , w$$

Similarly, the solution to the triangular portion of (10) is the same as for (9) for $k = 1, \ . \ . \ . \ , w-1$, and it is only required to show $y_k = \theta(S_k)/(1 - \theta(L))$ for $k = w, \ . \ . \ . \ , m$. The first equation of the loop portion of (10) yields

$$y_w = (y_{w-1} + y_m)/a_w$$

and subsequent equations yield

$$y_k = y_{k-1}/a_k = (y_{w-1} + y_m)/a_w \ . \ . \ . \ a_k \text{ for } k = w+1, \ . \ . \ . \ , m$$

Hence in particular,

$$y_m = (y_{w-1} + y_m)/a_w \ . \ . \ . \ a_m = (\theta(S_{w-1}) + y_m) \cdot \theta(L)) =$$

$$\theta(S_{w-1}) \ \theta(L) + y_m \theta(L) = \theta(S_m) + y_m \theta(L)$$

Solving for y_m gives

$$y_m = \theta(S_m)/(1-\theta(L))$$

The remark then follows from the fact that $\theta(S_k) = \theta(S_{k-1})/a_k$ and $y_k = y_{k-1}/a_k$.

In terms of the EAPI graph, the solution identified in Remark 3 can be viewed as *the determination of flows on the arcs of* P_i *to satisfy a requirement of 1 at node i*. The values y_k, k = 1, . . . , m, identify these arc flows. Further, the way in which the value of y_k is obtained from the value of y_{k-1} over the triangular portion of the system can be viewed as the *transmission of the requirement at node i to successive nodes of* P_i *as a consequence of assigning flow values along the path*. Thus, in particular, the graphical interpretation applied to the triangular portion of the system can be expressed in general as follows:

The assignment of a flow value of y_{k-1} to the $(k-1)^{st}$ arc of the path transmits a requirement equal to y_{k-1} to the k^{th} node of the path. In turn, this requirement of y_{k-1} at the k^{th} node of the path determines the unique flow value of $y_k = y_{k-1}/a_k$ (hence = $1/a_1$ . . . $a_k = \theta(S_k)$) for the k^{th} arc of the path. The solution for the loop portion of the system (in the non-triangular case) can be viewed similarly. The assignment of flows to the non-loop portion of the system transmits a requirement of $y_{w-1} = \theta(S_{w-1})$ to node w (equation w) and this must be satisfied by assigning flows to the arcs (values to the variables) associated with the loop. Specifically, as shown in the proof of Remark 3, this yields a flow of $y_w = \theta(S_w)/(1 - \theta(L))$ on the first arc of the loop, whereupon the remaining flows and transmitted requirements are identified exactly as in the triangular case, i.e., $y_k = y_{k-1}/a_k$ for k = w+1, . . . , m.

The flows thus determined provide the representation for the unit vector E^i relative to B'. These must be re-scaled to obtain the representation relative to the basis matrix B. Thus the k^{th} element of the re-scaled solution equals

y_k if the k^{th} arc is an original arc, and equals

$-a_k y_k$ if the k^{th} arc is a mirror arc

for $k = 1, \ldots, m$.

Remark 4: The predecessor paths P_p and P_q, for two distinct nodes p and q, contain all basic arcs with a nonzero coefficient in the basis representation of arc (p, q), where arc (p, q) represents any one of the possible multiple arcs connecting node p to node q.

Proof: This remark follows directly from Remark 2 since finding the representation of arc (p, q) is equivalent to finding $-E^p + a_{pq}E^q$ if (p, q) does not correspond to a self-loop. Otherwise, the representation is equivalent to finding E^q if $p = q\#$ and to finding $-E^p$ if $q = p\#$.

The foregoing procedures make it possible to determine the representation of a nonbasic arc (p, q) graphically as follows: Find appropriate flows on the arcs in P_p (if $p \neq q\#$) which satisfy a node requirement at node p of -1. Likewise, find appropriate flows on arcs in P_q (if $q \neq p\#$) which satisfy a node requirement at node q of a_{pq}. Assign zero flows to all other basic arcs. Add the resulting sets of flows to yield the coefficients which provide the representation of a nonbasic arc (p, q) with respect to the arcs in the basis graph. The calculation can be accelerated by tracing predecessor paths to their intersection, or to loop nodes (if the intersection occurs on a loop), and performing a single aggregated calculation (rather than a sum) from the point of intersection. (If desired, one may re-scale arcs in P_p and P_q that are mirror arcs to obtain the representation coefficients in terms of the original arcs. However, this is unnecessary in the execution of the extreme point algorithm.)

4.3 Updating and Maintaining the EAPI Basis Structure

To develop the special rules and relationships underlying the complete basis exchange operation, subject to maintaining the EAPI orientation, we will make use of the following definitions. From Remark 4, the basic arcs with a nonzero coefficient in the representation of a nonbasic arc (p, q) are contained in P_p and P_q. These arcs can be separated into three mutually exhaustive sets at least one of which is nonempty:

(1) $P_p \cap P_q$

(2) $P_p - P_q$

(3) $P_q - P_p$

If i is some node in the predecessor path of node j, we define P_{ij} to be the set of arcs in the predecessor path from node j to node i. If i = j or i is not an ancestor of j, $P_{ij} = \emptyset$. Then, by the previous definition of $\theta(S)$ (= 1/product of the multipliers of the arcs in set S)), we have $\theta(P_{ij})$ = 1/(the product of the multipliers of the arcs in P_j and not in P_i). (Recall by the convention $\theta(\emptyset) = 1$, that $\theta(P_{ij}) = 1$ if i = j or i is not an ancestor of j.)

We may further elaborate the properties of the function θ (to facilitate subsequent derivations involving sets of arcs on path segments, such as P_i, P_{ij}, etc.) as follows.

Remark 5: If $R_1, \ldots, R_v$ is any partition of a set of arcs R, then

$$\theta(R) = \theta(R_1)\theta(R_2) \ldots \theta(R_v).$$

In particular, if k is a node on the predecessor path of node j, and i is a node on the predecessor path of node k, so that $P_{ij} = P_{ik} \cup P_{kj}$, then $\theta(P_{ij}) = \theta(P_{ik})\theta(P_{kj})$. (We may allow k = i or k = j, in which case $\theta(P_{ik})$ or $\theta(P_{kj}) = 1$ and the relation still holds.)

Remark 6: If R* is the set of mirror arcs of the arcs in R, then $\theta(R^*) = 1/\theta(R)$. (Hence in particular $\theta(P_{ij}^*) = 1/\theta(P_{ij})$.

We will continue to use the asterisk to indicate a "mirror set" in the following. Also, for completeness, it is useful to define $\emptyset^* = \emptyset$ (hence $\theta(S) = \theta(S^*) = 1$ if $S = \emptyset$). However, to avoid cluttered notation such as $\theta(\{(i, j)\})$ and $\theta(\{(i, j)\}^*)$, we will let (i, j) represent both a specified arc from i to j and the set consisting of this arc.

The representation of an incoming arc (p, q) is by definition a weighted linear combination of the arcs in an EAPI basis graph. By means of this representation, any flow change in (p, q) induces a "corresponding" flow change in each arc of its representation. Thus, in particular, if Δ is the flow change in (p, q), and γ is the coefficient of a given arc, then $-\Delta\gamma$ is the flow change on the basic arc. This means that the flow change on the basic arc will have the same sign as the flow change on (p, q) if the coefficient of the basic arc is negative, and will have a flow change of the opposite sign if the coefficient of the basic arc is positive. An arc whose flow is among those first driven to an upper or lower bound by an attempted change of the value of (p, q) (away from the bound it currently equals) is called a *blocking arc*. (By our convention of employing mirror arcs, (p, q), being nonbasic, always equals its lower bound prior to the attempted flow change. It should be noted that (p, q) can be a blocking arc.)

In the case of a degenerate pivot, the flow on a blocking arc currently equals one of its bounds and would violate that bound with any change in the flow of the incoming arc (p, q). To maintain primal feasibility, the outgoing arc must always be one of the blocking arcs.

We consolidate these and several earlier observations by defining:

Q = the set of basic arcs with positive representation coefficients for (p, q)

P = the set of basic arcs with negative representation coefficients for (p, q)

(where, as will be seen, Q is associated with node q and P is associated with node p).

Remark 7: Q is the set of arcs whose flows are decreased and $P \cup (p, q)$ is the set of arcs whose flows are increased when the flow of the incoming arc (p, q) is increased. Further, $P \cup Q \subset P_p \cup P_q$.

Remark 8: The outgoing arc (r, s) satisfies:

$(r, s) \in Q$ if (r, s) is blocking at its lower bound.

$(r, s) \in P \cup (p, q)$ if (r, s) is blocking at its upper bound.

We are now in a position to state the major result of [18] which reduces a diverse collection of basis updating configurations to two simple cases and identifies the appropriate reorientation required to maintain the EAPI basis structure.

Theorem 1 [18]: For a basis exchange involving the incoming arc (p, q) and the outgoing arc (r, s), the EAPI rooted loop orientation is maintained as follows:

Reverse the orientation of the arcs of $P_{sp} \cup (p, q)$ if $(r, s) \in P \cup (p, q)$

Reverse the orientation of the arcs of P_{sq} if $(r, s) \in Q$.

It may be noted that reversing the orientation of a set of arcs S corresponds to replacing S by its mirror set S*. To translate and extend the implications of this important result still more fully to the mirror arc setting, and set the stage for further developments, we define for the situation in which $P_p \cap P_q \neq \emptyset$

x = the unique node on the predecessor path of node p such that

$P_{xp} = P_p - P_q$

y = the unique node in the predecessor path of node q such that

$P_{yq} = P_q - P_p$.

z = the first intersection node of P_{sp} and P_{sq}, i.e., the unique node of these paths such that $P_{sp} \bigcap P_{sq} = P_{sz}$, or equivalently $P_{zp} \bigcap P_{zq} = \emptyset$ and $P_{zq} \bigcup P_{zp} = P_{sq} \bigcup P_{sp} - (P_{sq} \bigcap P_{sp})$

Figures 4 and 5 illustrate these definitions.

Note that when x = y, node z = node x = node y.

When $x \neq y$, $z = \begin{cases} x \text{ if } (r, s) \varepsilon P_{yx} \\ y \text{ if } (r, s) \varepsilon P_{xy} \end{cases}$

Remark 9: If $P_p \bigcap P_q \neq \emptyset$, $P = P_{xp} \bigcup P_o$ and $Q = P_{yq} \bigcup Q_o$ for some P_o, Q_o, possibly empty and contained in $P_p \bigcap P_q$.

The significance of the identification of node z is demonstrated by the following consequence of Theorem 1 which refines the specification of the sets P and Q as follows:

Corollary 1: A new loop is created by adding the incoming arc (p, q) and dropping the outgoing arc (r, s) if and only if $(r, s) \varepsilon P_p \bigcap P_q$ and the composition of the new loop in the EAPI basis is given by

$(p, q)^* \bigcup P_{zp}^* \bigcup P_{zq}$ if $(r, s) \varepsilon (P_p \bigcap P_q) \bigcap P$

$(p, q) \bigcup P_{zp} \bigcup P_{zq}^*$ if $(r, s) \varepsilon (P_p \bigcap P_q) \bigcap Q$

Note that the composition of the loop when $(r, s) \varepsilon P$ is exactly the mirror of its composition when $(r, s) \varepsilon Q$.

5. THE NEW ALGORITHM

The preceding development consolidates many of the fundamental results for GN problems that can be found piecemeal in a variety of references [1, 7, 12, 16, 18, 22, 24]. By means of the generalized mirror arc concept, several useful interrelationships have been made visible that are not explicit in the literature. Our

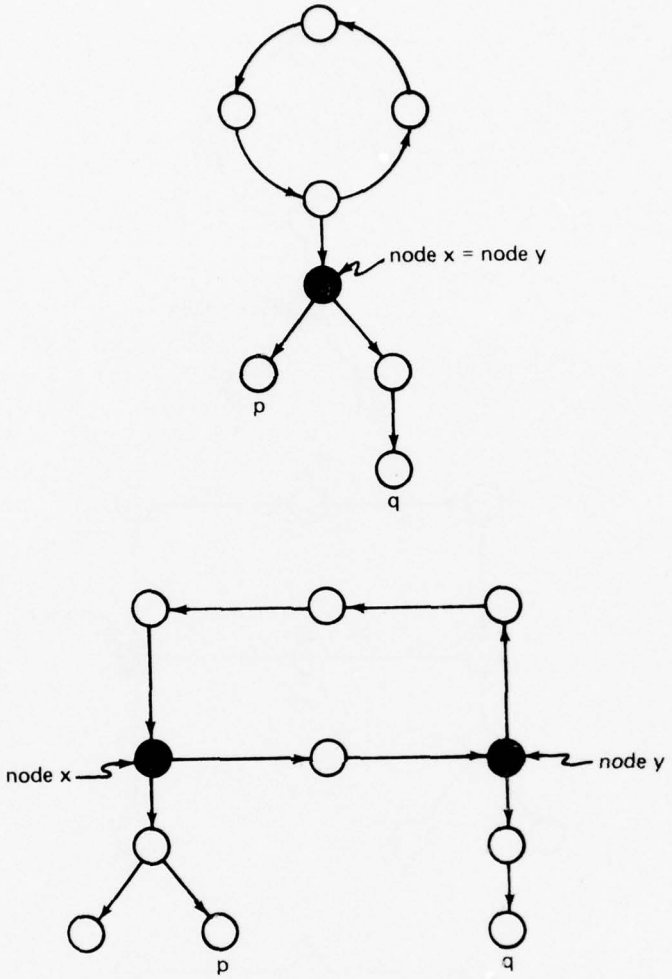


Figure 4 - Illustration of Nodes x and y

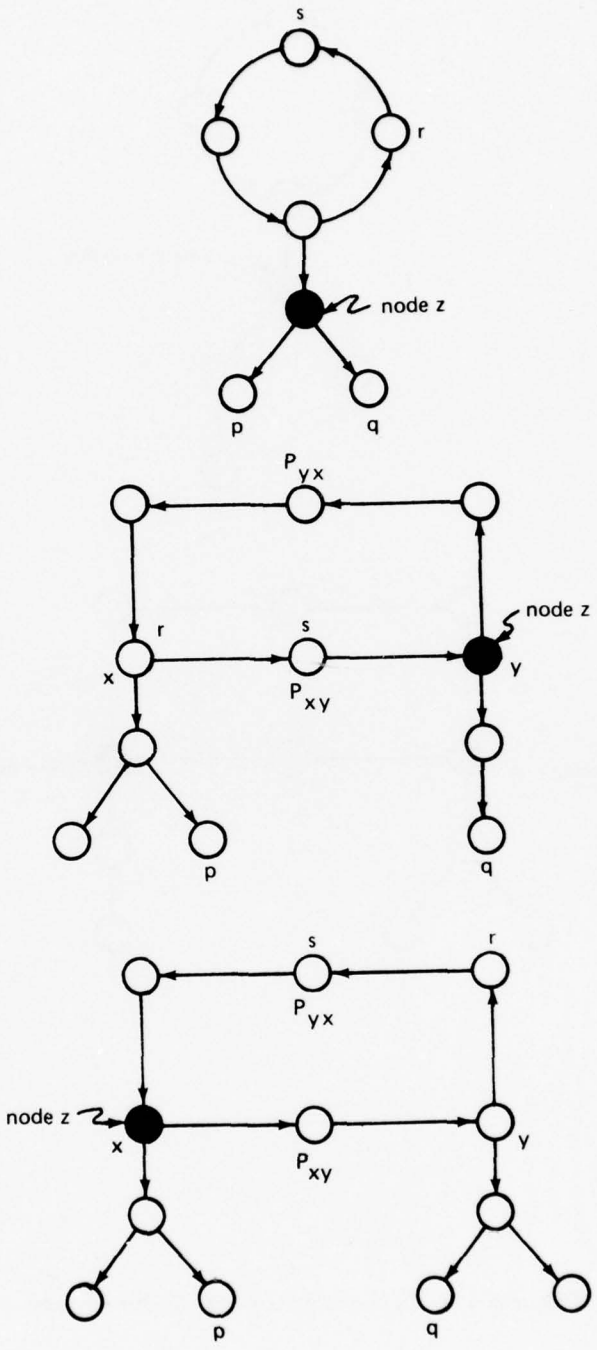


Figure 5 - Illustration of Node z

primary motivation for this development, however, is not simply to bring together scattered concepts in a unified format, but to lay a foundation for characterizing the special structural attributes and convergence properties of the new algorithm. The following sections are devoted to this end.

5.1 Fundamental Definitions

An arc has *lower leeway* if its flow is strictly greater than its lower bound, and has *upper leeway* if its flow is strictly less than its upper bound. An arc which has both lower and upper leeway will be called a *double leeway* arc. It follows that an arc has lower leeway if and only if its mirror has upper leeway, and vice versa. Consequently, if an arc is a mirror arc, then it has lower or upper leeway according to whether the original arc has the opposite type of leeway.

A basis B is defined to be a *strongly convergent (SC) basis* if and only if the EAPI basis graph satisfies each of the following conditions:

(1) Every arc has lower leeway.

(2) The loop factor $\theta(L)$ of each loop L of the basis graph other than a self-loop satisfies $\theta(L)<1$.

5.2 SC Basis Equivalence

Consider the basis shown in Figure 2. At least one of the arcs with 0 flow is an original arc, thus this is not an SC basis since not all arcs have lower leeway. In general, it is clear that many bases for generalized networks will not be SC bases.

A necessary condition for a basis to be equivalent to an SC basis is that all nonloop arcs satisfy Condition 1 of the SC basis definition. A sufficient condition for equivalence is given by the following remark.

Remark 10: A basis that satisfies Condition 1 of the SC basis definition can be reoriented to create an SC basis if and only if the arcs of each loop L with $\theta(L)>1$ are double leeway arcs.

Proof: If $\theta(L) = 1$ and L is not a self-loop, the vectors associated with the arcs in L form a linearly dependent set as a direct consequence of the solution identified in Remark 3, which is undetermined when $\theta(L) = 1$. Consequently such an L cannot be in the basis. Thus, we need only consider the case where $\theta(L)>1$. When L is oriented in the reverse direction, L is replaced by its mirror set L* and $\theta(L^*) = (1/\theta(L))<1$ by Remark 6. Since the mirror of an arc has lower leeway if and only if the arc has upper leeway, the conclusion follows at once.

By the same reasoning, if the only arcs without lower leeway are loop arcs, the basis can be reoriented to create an SC basis if and only if every basis loop L that contains such an arc satisfies $\theta(L)>1$ and consists entirely of arcs with upper leeway.

It should be pointed out, however, that it is always possible to find an initial feasible SC basis (which may involve artificial vectors for a GN problem). The following remark establishes this statement.

Remark 11: A basis that consists of positive and negative unit vectors E^i and $-E^i$, where E^i represents the i^{th} column of the identity matrix, is an SC basis provided E^i is basic if $b_i > 0$ and $-E^i$ is basic if $b_i \leq 0$ (where b_i is the i^{th} component of the right-hand vector b of (3)).

Proof: The proof follows immediately from the definition of an SC basis and the comments regarding the EAPI orientation of surplus arcs.

5.3 Special Properties of an SC Basis and the SC Algorithm

In this section, we will show that every SC basis exhibits two fundamental properties:

(1) The signs of the nonzero coefficients in the representation of a nonbasic arc with respect to an SC basis can be determined prior to actually finding the representation (by means of a complete characterization of P and Q).

(2) There exists a unique arc in the EAPI basis graph which must be chosen to leave the basis at each basis exchange step if the SC basis structure is to be maintained.

These properties have important implications for the computational efficiency of the new algorithm founded on the SC bases (hereafter called the SC algorithm).

A further useful property that is immediately available from the definition of an SC basis and the preceding results follows.

Remark 12: A basis exchange step executed relative to an SC basis will be non-degenerate if the outgoing arc $(r, s) \varepsilon (p, q) \cup Q$.

We now characterize the composition of P and Q for an SC basis.

Theorem 2: In an SC basis, if $P_p \cap P_q = \emptyset$, $P = P_p$ and $Q = P_q$. Otherwise, P consists of consecutive arcs in the predecessor path of node p and Q consists of consecutive arcs in the predecessor path of node q, such that

$$P = P_{xp} \cup P_o$$

$$Q = P_{yq} \cup Q_o$$

where,

$$P_o = \emptyset \text{ or } P_{yx} \text{ or } P_p \cap P_q$$

$$Q_o = \emptyset \text{ or } P_{xy} \text{ or } P_p \cap P_q$$

Further, for $x \neq y$

$$P_{xy} \subset \begin{cases} P_o \text{ if and only if } a_{pq}\theta(P_{yq}) < \theta(P_{yp}) \\ Q_o \text{ if and only if } a_{pq}\theta(P_{yq}) > \theta(P_{yp}) \end{cases}$$

$$P_{yx} \subset \begin{cases} P_o \text{ if and only if } a_{pq}\theta(P_{xq}) < \theta(P_{xp}) \\ Q_o \text{ if and only if } a_{pq}\theta(P_{xq}) > \theta(P_{xp}) \end{cases}$$

and for $x = y$

$$P_p \bigcap P_q = \begin{cases} P_o \text{ if and only if } a_{pq}\theta(P_{yq}) < \theta(P_{yp}) \\ Q_o \text{ if and only if } a_{pq}\theta(P_{yq}) > \theta(P_{yp}) \end{cases}$$

Proof: The essential content of the theorem not previously established, is the identification of the form of P_o and Q_o for the SC basis when $P_p \bigcap P_q \neq \emptyset$. Let L denote the unique loop contained in $P_p \bigcap P_q$. Allowing either or both of P_{xp} and P_{yq} to be empty (i.e., for $x = p$ or $y = q$, in which case $\theta(P_{xp})$ or $\theta(P_{yq}) = 1$), the representation coefficient γ for any arc $(i, j) \varepsilon P_p \bigcap P_q)$ can be characterized from Remarks 3 and 4, as follows:

$\gamma = a_{pq}\theta(P_{iq}) - \theta(P_{ip})$ if $(i, j) \notin L$ (which can occur only if $x = y$) or if (i, j) is the slack arc of a self-loop.

$\gamma = ((a_{pq}\theta(P_{iq}) - \theta(P_{ip}))/(1 - \theta(L))$ if $(i, j) \varepsilon$ L and L is not a self-loop. Because $\theta(L)<1$ for the SC basis, γ can be written $\gamma=\alpha\ (a_{pq}\theta(P_{iq})-\theta(P_{ip}))$ where $\alpha>0$, in both instances. If $x = y$, the results follow immediately since (1) $P_{xy} = P_{yx} = \emptyset$ and (2) the sign of the coefficients on all the arcs in $P_p \bigcap P_q (=P_x=P_y)$ correspond to the sign of the transmitted requirement to node $x = y$ which is $a_{pq}\theta(P_{xq})-\theta(P_{xp}) = a_{pq}\theta(P_{yq})-\theta(P_{yp})$. If $x \neq y$, first suppose $(i, j) \varepsilon P_{xy}$.

Then we can write $P_{iq} = P_{iy} \bigcup P_{yq}$ and $P_{ip} = P_{iy} \bigcup P_{yp}$. Hence

$$\gamma = \alpha\theta(P_{iy}) \left[a_{pq}\theta(P_{yq}) - \theta(P_{yp})\right]$$

from which it follows that the sign of γ depends only on the sign of the quantity in braces, and is independent of the choice of arc (i, j) itself, as long as $(i, j) \varepsilon P_{xy}$. Hence $P_{xy} \subset P_o$ or $P_{xy} \subset Q_o$ according to the inequality stipulated in the theorem.

Next, if $(i, j) \varepsilon P_{yx}$, we can write $P_{iq} = P_{ix} \bigcup P_{xq}$ and $P_{ip} = P_{ix} \bigcup P_{xp}$. Consequently,

$$\gamma = \alpha\theta(P_{ix}) \left[a_{pq}\theta(P_{xq}) - \theta(P_{xp})\right]$$

and by corresponding reasoning we conclude $P_{yx} \subset P_o$ or $P_{yx} \subset Q_o$ by the inequalities specified in the theorem.

Finally, it is necessary to show that P_o and Q_o (hence P and Q) are indeed sets of consecutive arcs in predecessor paths, i.e.,

$$P_o = \emptyset \text{ or } P_{yx} \text{ or } P_p \bigcap P_q \text{ and } Q_o = \emptyset \text{ or } P_{xy} \text{ or } P_p \bigcap P_q \ .$$

This assertion is equivalent to showing $P_o \neq P_{xy}$ and $Q_o \neq P_{yx}$ (on the basis of the foregoing results and the fact that $P_o \bigcap Q_o = \emptyset$). To do this, we multiply the expression for γ when $(i, j) \varepsilon P_{yx}$ by $\theta(P_{yx})$. Noting that

$$\theta(P_{yx})\theta(P_{xq}) = \theta(P_{yx})\theta(P_{xy})\theta(P_{yq}) = \theta(L)\theta(P_{yq}), \text{ and}$$

$$\theta(P_{yx})\theta(P_{xp}) = \theta(P_{yp}), \text{ we obtain}$$

$$\theta(P_{yx})\gamma = \alpha\theta(P_{ix}) \left[a_{pq}\theta(P_{yq})\theta(L) - \theta(P_{yp})\right]$$

Thus the sign of γ for $(i, j) \varepsilon P_{yx}$ depends on the quantity in the braces in this latter expression. Comparing this quantity to the quantity in the braces in the expression for γ when $(i, j) \varepsilon P_{xy}$, and noting $\theta(L) < 1$, it follows that $\gamma \geq 0$ for $(i, j) \varepsilon P_{yx}$ implies $\gamma > 0$ for $(i, j) \varepsilon P_{xy}$, and this together with its contrapositive

leads to the conclusion of the theorem.

The preceding characterizations make it possible to state the SC algorithm and prove its special properties. For this purpose, we suppose the arcs of P and Q are indexed in ascending order, in the same sequence as they are encountered by a trace of the predecessor indexing, starting at nodes p and q. (Thus, if P is not empty, (p(p), p) is the first arc of P, and if Q is not empty, (p(q), q) is the first arc of Q.) In addition, we will augment the set P by the addition of arc (p, q), to give the set $(p, q) \cup P$, and treat arc (p, q) as the first arc of the augmented set (whereupon the first arc of P is the second arc of this set, etc.).

The complete form of the SC algorithm may be described as follows. (The specifications for determining and updating node potentials in this description can be implemented efficiently by reference to the procedures described in [16, 18].)

The SC Algorithm

1. Begin with an SC Basis (Remark 11) and determine node potentials π_k for each node k, so that reduced cost $a_{ij}\pi_j - \pi_i - c_{ij} = 0$ for each basic arc (i, j).
2. Select the incoming arc (p, q) to be any nonbasic arc whose reduced cost $a_{pq}\pi_q - \pi_p - c_{pq}$ is profitable (i.e., positive). If no such arc exists (and all artificial arcs have zero flow), the problem is solved and the current arc flows are optimal.
3. Select the outgoing arc (r, s) to be:
   (1) the last blocking arc in Q if any blocking arc is a member of that set.
   (2) the first blocking arc of $(p, q) \cup P$ if there is no blocking arc in Q.
4. Execute a basis exchange step, reorienting the basis by Theorem 1 and determining the updated flows and node potentials. Then return to instruction 2.

Recall that the incoming and outgoing arcs can be self-loops (i.e., (r, s) can be a slack arc and (p, q) can be either a slack or a surplus arc). By convention, for these cases, we define $\pi_{j\#} = 0$ and $\pi_{i\#} = 0$ (for slack and surplus arcs, respectively). This yields correct reduced cost expressions for self-loops.

Theorem 3: The SC algorithm maintains the SC basis structure at each basis exchange step. Moreover, any other choice of (r, s) destroys this structure.

Proof: First, we show that all arcs of $(p, q) \in P \bigcup Q$ (the only arcs whose flows change) will have lower leeway after the exchange step if and only if (r, s) is selected from this set as specified. If $(r, s) \in Q$, the flow change is nondegenerate by Remark 12. None of the arcs in Q indexed higher than (r, s) are blocking arcs. Hence these arcs retain their lower leeway and further are not reoriented by the basis exchange step. If a blocking arc existed that did have a higher index than (r, s), it would not have lower leeway after the pivot and would also not be reoriented, thereby violating the SC structure. The arcs of P_{sq}, which are reoriented (Theorem 1) must have upper leeway before reorientation due to the fact that $P_{sq} \subset Q$ and the flow change is nondegenerate. Consequently, after reorientation they have lower leeway. Finally, arcs in $(p, q) \in P$, which are not reoriented, have lower leeway because their flows are increased. If $(r, s) \in (p, q) \bigcup P$, then none of the arcs in $(p, q) \bigcup P$ indexed lower than (r, s) are blocking, and hence must have upper leeway. These are precisely the arcs that are reoriented (Theorem 1) and hence have lower leeway after reorientation. If any arc indexed lower than (r, s) had been blocking, then after reorientation such an arc would have had no lower leeway, thereby violating the SC structure. No arcs in Q are blocking, and hence retain lower leeway, and arcs in P not reoriented have lower leeway since their flows are only increased or remain the same.

Now, we must establish that $\theta(L) < 1$ for any loop L created by the basis ex-

change step, other than a self-loop. By Corollary 1, we need only consider the case where $(r, s)\varepsilon(P_p \bigcap P_q) \bigcap P$ and $(r, s)\varepsilon(P_p \bigcap P_q) \bigcap Q$. From Section 4.3, if $x = y$, $z = x = y$. If $x \neq y$, $z = x$ if $(r, s)\varepsilon P_{yx}$ and $z = y$ if $(r, s)\varepsilon P_{xy}$. Assuming that $(r, s)\varepsilon P$, the new loop L formed is $(p, q)^* \bigcup P_{zp}{}^* \bigcup P_{zq}$.

If $x = y$, by Theorem 2 and the substitution of z, $a_{pq}\theta(P_{zq}) < \theta(P_{zp})$ which implies $\theta(L) < 1$. If $x \neq y$, by Theorem 2, we need only consider the case $(r, s)\varepsilon P_{yx}$. By Theorem 2 and the substitution of z, $a_{pq}\theta(P_{zq}) < \theta(P_{zp})$ which again, implies $\theta(L) < 1$. Similar reasoning applies when $(r, s)\varepsilon Q$.

5.3 Convergence Properties of the SC Algorithm

Having established the way to create and maintain an SC basis, and having characterized the sufficiency condition by which a pivot in this basis is non-degenerate, we will now show that the SC algorithm is finitely convergent without any reliance upon "external" techniques such as perturbation. Moreover, we will show that the convergence through a succession of degenerate pivots is of a much stronger form than that established by reference to "perturbation" by showing that the dual variables (node potentials) which change after a basis exchange step change in a uniform direction throughout any sequence of degenerate pivots.

Removing the outgoing arc (r, s) before adding the incoming arc (p, q) always creates a unique tree in the EAPI basis graph. (This tree will contain all nodes of a former quasi-tree if the outgoing arc is a loop arc. The tree may also consist of only a single node.) We let T denote the set of nodes in this tree. The only node potentials which change in the new basis are those associated with the nodes in T. A change in the node potential of any particular node of T induces a change in the node potentials of all its successor nodes. This change is characterized as follows.

Remark 13: In order to maintain complementary slackness (reduced arc costs of 0), a change of α in the node potential of node i in T induces a change of $\alpha\theta(P_{ij})$ in the node potential of any successor node j in T.

Proof: Assume that the nodes in the predecessor path from node j to node i are numbered beginning at node i as i, i+1, . . . , j. Let π_k denote the current node potential and α_k denote the change in this node potential for node k. In order for complementary slackness to hold,

$$-(\pi_k+\alpha_k)+a_{k+1}(\pi_{k+1}+\alpha_{k+1})-c_{k,k+1} = 0 \tag{11}$$

where $c_{k,k+1}$ denotes the cost attached to the arc (k,k+1), k = i, . . . , j-1. The current node potentials also satisfy complementary slackness:

$$-\pi_k+a_{k+1}\pi_{k+1}-c_{k,k+1} = 0 \tag{12}$$

for each arc (k,k+1), k = i, . . . , j-1.

Combining (11) and (12) we obtain

$$\alpha_{k+1} = \frac{\alpha_k}{a_{k+1}}$$

The remark follows at once.

Remark 14: A change in the node potential of node i induces a change of the same sign in the node potentials of all its successors in T if the conditions of complementary slackness are maintained.

Proof: A direct consequence of Remark 13.

Using the above remarks, we will now prove that the SC algorithm is finitely convergent and this convergence has a particularly strong character. Let π denote the vector of node potentials. An extreme point method is said to be *uniformly converging* if every component of π that changes must strictly increase as a result of a degenerate pivot. Clearly this form of "uniform" convergence is substantially stronger than the usual lexicographic convergence" in which only the least indexed component of π which changes must increase.

Theorem 4: Every pivot of the SC algorithm causes the potentials of the nodes in T to change in the following ways:

(1) If $(r, s) \in Q$, the potentials of the nodes in T decrease.

(2) If $(r, s) \in P$, the potentials of the nodes in T increase.

Further, the SC algorithm is uniformly converging through every sequence of degenerate pivots.

Proof: The incoming arc (p, q) is eligible to enter the basis if its reduced cost is positive, i.e.,

$$-\pi_p + a_{pq}\pi_q - c_{pq} = \alpha > 0 \tag{13}$$

First suppose $(r, s) \notin P_p \bigcap P_q$. By Theorem 1, the tree associated with T will be rooted after re-orientation at node q (node p) if the outgoing arc belongs to $P_q - P_p$ ($P_p - P_q$). If node q becomes the root (hence $(r, s) \in Q$ by Theorem 1) the potential of q must change by δ to maintain complementary slackness, where

$$-\pi_p + a_{pq}(\pi_q + \delta) - c_{pq} = 0 \tag{14}$$

Rewriting (13) using (14) we obtain

$$\delta = \frac{-\alpha}{a_{pq}} < 0.$$

Hence by Remark 14, the potentials of all nodes in T decrease. If node p becomes the root (hence $(r, s) \in P$), similar reasoning discloses the potentials of nodes in T increase.

Next suppose $(r, s) \in P_p \bigcap P_q$. Both node p and node q are then in T and thus both node potentials will change. Let δ represent the change in π_p and Δ represent the change in π_q. To maintain complementary slackness in the updated basis:

$$-(\pi_p + \delta) + a_{pq}(\pi_q + \Delta) - c_{pq} = 0 \tag{15}$$

If $(r, s)\varepsilon Q$, the arcs in P_{sq} are reoriented in the updated basis such that node p is a successor of node q (before adding arc (p, q), which also makes q a successor of p). Thus from Remark 13, a change of Δ at node q transmits a change of $\Delta\theta(P_{qp})$ to node p for the predecessor path that connects node p and node q after reorienting. However, from (13), we obtain $\delta = \alpha+a_{pq}\Delta$ which implies $\alpha+a_{pq}\Delta = \Delta\theta(P_{qp})$.

Solving for Δ,

$$\Delta = \alpha/(\theta(P_{qp}) - a_{pq}) \tag{16}$$

By Corollary 1, the composition of P_{qp} in terms of the basis arcs before reorienting is $P_{zp} \bigcup P_{zq*}$ so that (16) becomes upon rearranging terms

$$\Delta = \alpha\ \theta(P_{zq})/\theta(P_{zp})-a_{pq}\ \theta(P_{zq}).$$

Using the appropriate substitution for z when $(r, s)\varepsilon P_p \bigcap P_q$, $x = y$; $(r, s)\varepsilon P_{xy}$; and $(r, s)\varepsilon P_{yx}$, we obtain from Theorem 2 $\Delta<0$.

Since node q is a loop node in the updated basis, all nodes whose potentials change are successors of node q. Thus by Remark 14, all node potentials which change will decrease.

If $(r, s)\varepsilon P$,

$$\delta = \frac{\alpha}{1-a_{pq}\theta(P_{pq})}$$

Using the same reasoning as above, it follows that $\delta>0$ when $(r, s)\varepsilon P_p \bigcap P_q$, $x = y$; when $(r, s)\varepsilon P_{xy}$; and when $(r, s)\varepsilon P_{yx}$ and thus all node potentials which change will increase. Finally, since in a degenerate pivot $(r, s)\varepsilon P$, the uniformly converging property is established and the theorem is proved.

6.0 Computational Advantages

In the SC algorithm, the number of *possible* degenerate pivots is reduced since the special structure of the basis eliminates many of the alternative basis representations for extreme points normally examined by other algorithms. This combined with the strong convergence properties of the algorithm should result in fewer total pivots. Further, since no computational testing is required to ensure that only bases with the SC structure are examined, there will be no accompanying increase in execution time per pivot.

Hultz, et al [21] showed that by incorporating degeneracy checks into a GN code, solution times were substantially decreased. In this code, degeneracy checks are applied to each arc traversed. When executing the SC algorithm, degeneracy can occur only in a portion of the basis graph (i.e., the set P). Thus, degeneracy checks need only be applied to the arcs in P. Further by the pivot rules of the algorithm, the first arc encountered in P which will cause a degenerate pivot is also the outgoing arc; thus no further tree traversal or additional degeneracy checks are required once degeneracy is detected.

For these reasons, the SC algorithm when implemented should result in several computational advantages over currently available codes [19, 22, 24] for solving GN problems.

7.0 Limits of Generality

As noted earlier, methods with the uniform convergence property have previously been developed in the pure network setting for assignment, transportation and transshipment problems [2, 3, 4, 11]. There is a major leap from the generality of the pure network setting to that of the generalized network setting which, successfully accomplished by the preceding development, raises the question of whether it is possible to discover a primal extreme point algorithm with the

uniform convergence property for the still more general case where both ends of a generalized arc may have multipliers (i.e., nonzero entries in the associated column of A) of the same sign. In fact, our results make it possible to conclusively answer this question in the negative, establishing the precise limits to the level of problem generality for which such a method exists, and identifying the SC algorithm as the method that attains that level (relative to linear programming problems with at most two entries in each column of A).

To demonstrate this, consider first of all the required change in the definition of $\theta(S)$ for a set of arcs S when some arcs may have multipliers of the same sign. Retracing the proof of Remark 3, which motivates this definition (to give the form of the representation coefficients), it is clear that S must specifically refer to a set of arcs with an implied orientation, as induced by the predecessor indexing. Hence, if S consists of k arcs, with multipliers a_j, u_j for $j = 1, \ldots, k$, where a_j is the multiplier at the successor node and u_j is the multiplier at the predecessor node under the implied orientation (hence u_j occupies the role of the -1 multiplier in the preceding development) then the appropriate definition is

$$\Theta(S) = \begin{cases} 1 \text{ if } S = \emptyset \\ 1/a_1 \text{ if } k = 1 \\ (1/a_1)(-u_1/a_2)(-u_2/a_3) \ldots (-u_{k-1}/a_k) \text{ if } k > 1 \end{cases}$$

In addition, however, the loop factor must be redefined to be $\Theta(L)$ instead of $\theta(L)$, where

$$\Theta(L) = (-u_1/a_1) \ldots (-u_k/a_k)$$

for L composed of the arcs specified above to be in S (hence, in general, $\Theta(S) = -u_1 \ldots -u_{k-1}\theta(S)$). By these conventions, replacing $\theta(L)$ by $\Theta(L)$, the statement of Remark 3 is valid for arcs with arbitrary multipliers.

Note that when all of the u_k are -1, as earlier, then $\Theta(S) = \theta(S)$. Also, for a self-loop L, $\Theta(L) = \theta(L) = 1$ by these definitions. Finally, by an obvious adaptation of the mirror arc concept to arcs whose multipliers have the same sign, it can be assured that $u_k = -1$ for all basic arcs, and the previous expressions recover their validity.

The chief consequence of these observations is that representation coefficients for the unit vector E^i may have both positive and negative signs on the arcs of the predecessor path of node i. Further, the representation coefficient of a particular arc on this path can change its sign simply by shifting the location of node i on the path. Thus, the composition of P and Q, whose special form is essential to Theorems 3 and 4, can no longer be characterized in relation to the sets P_p and P_q. Finally, after a basis exchange, the change in the node potential of node j induced by a change of α in the potential of an ancestor, node i, becomes $\alpha\ \Theta(P_{ij})$ (Remark 13), which may or may not have the same sign as α, and consequently it is impossible for the node potentials to change in any uniformly specifiable manner as required by Theorem 4.

Still more simply (though at a less rigorous level) it is possible to argue that a single arc whose multipliers are of the same sign can prevent the uniform convergence property from being established. If such an arc is nonbasic and becomes the incoming arc (p, q), then both P_p and P_q belong either to P or Q, and the proof of Theorem 3, which establishes the uniqueness of the outgoing arc (r, s), discloses that there may in this case be irreconcilable competing choices for (r, s) in both $P_p - P_q$ and $P_q - P_p$. Regardless of the choice of (r, s), the lower leeway requirement is no longer satisfied by all basic arcs, and a subsequent step can therefore yield a degenerate pivot when $(r, s) \varepsilon Q$, invalidating the requirement needed to establish Theorem 4.

REFERENCES

1. E. Balas and P. L. Ivanescu (Hammer) (1964). On the Generalized Transportation Problem. *Management Sci.* 11 188-202.

2. R. Barr, J. Elam, F. Glover, D. Klingman (1976). A Network Alternating Path Basis Algorithm for Transshipment Problems. Research Report CCS 272, Center for Cybernetic Studies, University of Texas at Austin.

3. R. Barr, F. Glover, and D. Klingman (1977). The Alternating Basis Algorithm for Assignment Problems. to appear in *Math. Programming*.

4. R. Barr, F. Glover, and D. Klingman (1977). The Generalized Alternating Path Algorithm for Transportation Problems. Research Report CCS 282, Center for Cybernetic Studies, University of Texas at Austin.

5. R. Barr, F. Glover, and D. Klingman (1974). An Improved Version of the Out-of-Kilter Method and a Comparative Study of Computer Codes. *Math. Programming* 7, 1 60-87.

6. G. Bhaumik (1973). Optimum Operating Policies of a Water Distribution System with Losses. Ph.D. Thesis, University of Texas at Austin.

7. A. Charnes and W. Cooper (1961). *Management Models and Industrial Applications of Linear Programming,* Vols. I and II. John Wiley, New York, N. Y.

8. A. Charnes, F. Glover, D. Karney, D. Klingman, and J. Stutz (1975). Past, Present, and Future Development, Computational Efficiency, and Practical Use of Large Scale Transportation and Transshipment Computer Codes. *Comput. and Ops. Res.* 2, 2 71-81.

9. R. Crum (1976) Cash Management in the Multinational Firm: A Constrained Generalized Network Approach. Working paper, The University of Florida, Gainesville, Florida.

10. R. Crum, D. Klingman, and L. Tavis (1976). Implementation of Large-Scale Financial Planning Models: Solution Efficient Transformations. Research Report CCS 267, Center for Cybernetic Studies, University of Texas at Austin.

11. Cunningham, W. H. (1976). A Network Simplex Method. *Math. Programming* 11 105-116.

12. G. Dantzig (1963). *Linear Programming and Extensions.* Princeton University Press, Princeton, New Jersey.

13. F. Glover, J. Hultz, D. Klingman, J. Stutz (1977). Implementation and Computational Study on Generalized Network Codes. Research Report CCS 258. Center for Cybernetic Studies, University of Texas at Austin.

14. F. Glover, D. Karney, D. Klingman, and A. Napier (1974). A Computational Study on Start Procedures, Basis Change Criteria, and Solution Algorithms for Transportation Problems. *Management Sci.* 20, 5 793-813.

15. F. Glover and D. Klingman (1973). On the Equivalence of Some Generalized Network Problems to Pure Network Problems. *Math. Programming* 4, 3 369-378.

16. F. Glover and D. Klingman (1973). A Note on Computational Simplifications in Solving Generalized Transportation Problems. *Trans. Sci.* 7, 4 351-361.

17. F. Glover and D. Klingman (1975). Network Applications in Industry and Government. MSRS 75-20, University of Colorado, Boulder, Co.

18. F. Glover, D. Klingman, and J. Stutz (1973). Extensions of the Augmented Predecessor Index Method to Generalized Network Problems. *Trans. Sci.* 7, 4 377-384.

19. F. Glover, D. Klingman, and J. Stutz (1973). Implementation and Computational Study of a Generalized Network Code. 44th National Meeting of DRSA, San Diego, Ca.

20. F. Glover and J. Mulvey (1975) Equivalence of the 0-1 Integer Programming Problem to Discrete Generalized and Pure Networks. MSRS 75-19, University of Colorado, Boulder, Co.

21. J. Hultz, F. Glover, and D. Klingman (1977). A New Computer-Based Planning Tool. Research Report CCS 258, Center for Cybernetic Studies, University of Texas at Austin.

22. R. Langley (1973). Continuous and Integer Generalized Flow Problems. Ph.D. Thesis, Georgia Institute of Technology.

23. J. Lourie (1964). Topology and Computation of the Generalized Transportation Problem. *Management Sci.* 11, 1 177-187.

24. J. Maurras (1972). Optimization of the Flow through Networks with Gains. *Math. Programming* 3 135-144.

Unclassified

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author)

Center for Cybernetic Studies
The University of Texas

2a. REPORT SECURITY CLASSIFICATION

Unclassified

2b. GROUP

3. REPORT TITLE

A Strongly Convergent Primal Algorithm for Generalized Networks

4. DESCRIPTIVE NOTES (Type of report and, inclusive dates)

5. AUTHOR(S) (First name, middle initial, last name)

Fred Glover
Joyce Elam
Darwin Klingman

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
January 1977	43	24

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
N00014-76-C-0383;N00014-75-C-0616;0569	Center for Cybernetic Studies Research Report CCS 288
b. PROJECT NO. NR047-021	
c.	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)
d.	

10. DISTRIBUTION STATEMENT

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

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Office of Naval Research (Code 434) Washington, D.C.

13. ABSTRACT

A major computational problem that arises in the attempt to solve generalized network and network-related problems is degeneracy. In fact, using primal extreme point solution techniques, the number of degenerate pivots performed frequently ranges as high as 90% in large-scale applications. The purpose of this paper is to develop a special set of structural and logical relationships for generalized network problems that yields a new primal extreme point algorithm which avoids and/or exploits degeneracy. The major mathematical differences between this new algorithm and the simplex method are (1) each basis examined is restricted to have a special topology, (2) the algorithm is finitely convergent without reliance upon "external" techniques such as lexicography or perturbation, and (3) a special screening criterion for nondegenerate basis exchanges is available that allows some of these exchanges to be recognized prior to finding the representation of an incoming arc. For these reasons, this algorithm has several computational advantages over the simplex-based codes recently developed for solving generalized network problems.

DD FORM 1 NOV 65 1473 (PAGE 1)

S/N 0101-807-6811

Unclassified

Security Classification

A-31408

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Linear Programming						
Generalized Networks						
Weighted Networks						