AD-A035 918 TEXAS UNIV AT AUSTIN CENTER FOR CYBERNETIC STUDIES F/G 12/2
SOLVING CONSTRAINED GENERALIZED NETWORK PROBLEMS.(U)
NOV 76 J HULTZ, D KLINGMAN N00014-75-C-0616
UNCLASSIFIED CCS-257 NL

1 OF 1
AD-A
035 918

END
DATE
FILMED
3-25-77
NTIS

SOLVING CONSTRAINED GENERALIZED NETWORK PROBLEMS

TEXAS UNIVERSITY AT AUSTIN

NOVEMBER 1976

CENTER FOR CYBERNETIC STUDIES

The University of Texas
Austin, Texas 78712

Research Report CCS 257

SOLVING CONSTRAINED GENERALIZED NETWORK PROBLEMS

by

John Hultz
D. Klingman

November 1976

This research was partly supported by ONR Contract N00014-76-C-0383 with Decision Analysis and Research Institute and by Project NR047-021, ONR 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, 302E
The University of Texas
Austin, Texas 78712
(512) 471-1821

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)	2a. REPORT SECURITY CLASSIFICATION
Center for Cybernetic Studies The University of Texas	Unclassified 2b. GROUP

3. REPORT TITLE

"Solving Constrained Generalized Network Problems"

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

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

John Hultz
Darwin Klingman

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
November 1976	25	24

8a. CONTRACT OR GRANT NO	9a. ORIGINATOR'S REPORT NUMBER(S)
N00014-75-C-0569; 0616 b. PROJECT NO. NRO47-021 c. d.	Center for Cybernetic Studies Research Report CCS 257 9b. OTHER REPORT NO(S) (*Any other numbers that may be assigned this report*)

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 constrained generalized network problem is a linear programming problem in which the coefficient matrix contains m + q rows which are ordered such that each column has at most two non-zero entries in the first m rows. This paper describes highly efficient ways to modify and implement the steps of the simplex algorithm for such problems. The efficiency is the direct result of exploiting the generalized network portion (first m rows) of the coefficient matrix.

DD FORM 1 NOV 65 1473 (PAGE 1)

S/N 0101-807-6811

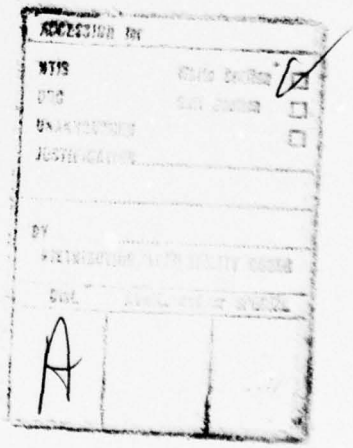
Unclassified.
Security Classification

A-31408

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Networks						
Generalized Networks						
Transportation						
Flow Management						
Linear Programming						

ABSTRACT

A constrained generalized network problem is a linear programming problem in which the coefficient matrix contains m + q rows which are ordered such that each column has at most two non-zero entries in the first m rows. This paper describes highly efficient ways to modify and implement the steps of the simplex algorithm for such problems. The efficiency is the direct result of exploiting the generalized network portion (first m rows) of the coefficient matrix.



1.0 INTRODUCTION

A constrained generalized network problem is a capacitated or uncapacitated linear programming (LP) problem in which the coefficient matrix contains m + q rows which are ordered such that each column has at most two non-zero entries in the first m rows. Note that by appropriately assigning values of m and q, any LP problem is a constrained generalized network problem. A major portion of the LP literature has been devoted either to problems in which m = 0 (standard LP problems) or to problems in which q = 0 (network related problems). The advantages of working with LP problems where m = 0 have been dramatically shown in the literature [2,7,8,17,21,24]. However, with the exception of generalized upper bounding (GUB) procedures, little research has been devoted to problems which are combinations of networks and standard LP problems.

This paper describes highly efficient ways to modify and implement the steps of the simplex algorithm for such problems. The efficiency is the direct result of exploiting the generalized network portion (first m rows) of the coefficient matrix. Thus, this research can be viewed as extending the concepts of exploiting special structure in LP problems, such as those presented in GUB procedures and in partitioning methods [3,4,18,22]. As stated, the term generalized network (GN) will be used to define the special structure evident in the constrained generalized network (CGN) problem. Any GN problem is characterized by a coefficient matrix which has at most two non-zero entries in each column. The term GN, therefore, is actually the broadest classification of network related problems. Depending upon the non-zero components of the coefficient matrix,

a GN problem could be a shortest path problem, an assignment problem, a transportation problem, a transshipment problem, a generalized transportation problem, a generalized transshipment problem, or even simple GUB constraints. The procedures presented subsequently are general enough to handle any of the network related problems listed above. To this extent, a higher degree of efficiency may be achieved in certain cases [9,18] by further exploiting the structure of the specific GN problem.

Some of the most efficient procedures for solving GN problems are based on viewing the problems in a graphical context. In particular, the simplex algorithm has been adapted [2,6,7,12,13,14,19,23] to solve problems in which the coefficient 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 the fundamental simplex operations and the amount of computer memory required to store essential data.

Due to the near-triangularity of bases for GN problems, the matrix operations of finding the representation of an entering vector and determining updated pseudo dual variable values can be performed by tracing paths within the basis graphs. Therefore, no explicit basis matrix is needed and the usual arithmetic operations required to update the inverse (and possible round-off error) are eliminated. In addition, the graphs contain only the non-zero matrix components which allows the list procedures [6,12,14] to eliminate checking and performing unnecessary arithmetic operations on zero elements. The advantages of using such techniques are illustrated by the fact that simplex codes for solving capacitated transshipment problems which employ graph processing techniques [7,17] are at least 75

times faster than state of the art commercial LP codes. In fact, a graph processing code for solving transportation problems [8] has been shown to be 150 times faster than the OPHELIE LP code.

The graph processing techniques also have the added advantage of requiring less computer memory to solve a problem. This allows larger problems to be solved without resorting to slower external storage. Even if the use of external storage is unavoidable, the graph procedures are normally able to maintain all basis information in central memory and are able to facilitate efficient input/output processing of nonbasic variable data.

Motivated by these advantages, this paper presents efficient procedures for solving constrained generalized network problems. A partitioning method is employed in conjunction with the simplex algorithm. In particular, the working basis for the CGN problem is partitioned in such a way that the near-triangularity of those portions relating to the GN problem can be fully exploited from both algebraic and graph processing viewpoints.

These solution procedures can be easily applied to GN problems to which extra constraints have been added. Such problems include multicommodity inventory scheduling problems, block diagonal GN problems with coupling constraints, and goal programming GN problems. However, one of the major objectives of this research is to draw attention to the critical nature of the formulation process. Every LP problem is a CGN problem, but the efficiency of the solution procedures depends on maximizing the size of the GN portion of the problem. In this regard, future research is needed to explore formulation and transformation procedures.

2.0 PROBLEM DEFINITION

The constrained generalized network (CGN) problem and its dual may be stated mathematically as follows:

Primal

$$\text{Minimize } c_1^T x + c_2^T y \tag{1}$$

subject to:

$$Gx + 0y = b \tag{2}$$

$$A_1 x + A_2 y = d \tag{3}$$

$$0 \leq x \leq u \tag{4}$$

$$0 \leq y \leq v \tag{5}$$

Dual

$$\text{Maximize } w^T b + \delta^T d - \gamma^T u - \psi v \tag{6}$$

subject to:

$$w^T G + \delta^T A_1 - \gamma \leq c_1^T \tag{7}$$

$$\delta^T A_2 - \psi \leq c_2^T \tag{8}$$

$$w, \delta \text{ -- unrestricted} \tag{9}$$

$$\gamma, \psi \geq 0 \tag{10}$$

where G is (m x n), A_1 is (q x n), and A_2 is (q x k). The remaining vectors are conformable column vectors.

Each column of the matrix G contains at most two non-zero entries which makes it possible to streamline the basic steps of the simplex algorithm. Any LP problem containing a coefficient matrix with this structure lies in the domain of the broad classification referred to in the literature as generalized network (GN) problems [1,10,12]. For this

reason, the (2) portion of the CGN problem will be referred to as the underlying GN problem. The procedures presented in this paper, as is true with much of the literature on GN problems [1,3,4,12,13,19,20], are based on visualizing and storing the information associated with G as a graph. Thus the essential concepts and definitions of elementary graph theory will be presented.

A simple graph G(V,E) is a finite set of vertices V and a finite set of edges E connecting the vertices. Each element of E is identified with an unordered pair of distinct elements of V. Visually, each edge connects two distinct vertices, which are then considered to be adjacent. If the edge set E is expanded to contain edges which have both endpoints incident on the same vertex (loops) or multiple edges connecting the same two vertices (parallel edges), then G(V,E) is called a general graph [5].

The underlying GN problem defines a general graph as follows. Each row of G corresponds to a vertex and each column corresponds to an edge of the general graph. The non-zero entries in a column will be referred to as the "multipliers" associated with the corresponding edge. The positions of the multipliers denote the vertices on which the edge is incident. If a column has only one multiplier, then the endpoints of the edge are incident on the same vertex and the edge will be called a loop. Associated with each edge is a variable, an upper bound, an objective function coefficient, and the non-zero multipliers. Each vertex has an associated right hand side component, called a vertex value. The edge multipliers govern how much of the variables are to be added to (or subtracted from) the appropriate vertex values.

It is customary in the literature on GN problems to assume that each edge of the general graph has an implied direction. In such cases, the graph is called a digraph, the vertices are called nodes, and the edges are called arcs. A variable is then considered to be a flow across an arc from one node to another. In generalized transportation problems, this is done by assigning certain vertices to be sources and all others to be destinations and scaling the multipliers of the source nodes to all be 1. If the problem is a generalized transshipment network, the coefficient matrix is transformed so that one of the multipliers on an edge is a -1 and the other is some non-zero quantity. The edge is then an arc directed from the -1 coefficient to the other. In certain cases the digraph formulation can be quite efficient since it reduces both the amount of data stored and the complexity of arithmetic operations. This is especially true if G does not have full row rank, since any GN problem with less than full row rank is equivalent to a disjoint set of minimum cost flow networks [10]. However, for the sake of generality the subsequent procedures do not assume that the underlying GN characterizes a digraph.

3.0 BASIS STRUCTURE

Using the standard bounded variable simplex algorithm, a basis B for the CGN problem will be a matrix composed of a linearly independent set of column vectors selected from the coefficient matrix $A = \begin{matrix} G & 0 \\ A_1 & A_2 \end{matrix}$. The variables associated with the column vectors of B are considered to be basic variables and all others are non-basic variables at their lower or upper bound.

For convenience, it will be assumed that the coefficient matrix A has full row rank. If this is not the case, artificial variables may be appended to the problem. Any basis B for the CGN problem will, therefore, be a non-singular matrix of order $(m + q) \times (m + q)$. Clearly, any basis matrix B may be partitioned as follows:

$$B = \begin{bmatrix} G_m & E_q \\ A_m & A_q \end{bmatrix}. \qquad (11)$$

The subscripts are used to denote the number of columns in each matrix. G_m is a non-singular $(m \times m)$ matrix which is composed of column vectors taken from G. E_q is $(m \times q)$ and may contain zero vectors and vectors from G. A_m, which is $(q \times m)$, and A_q, which is $(q \times q)$, are the continuations of the vectors in G_m and E_q, respectively. Based on the partitioning of matrices (11), the basis matrix inverse B^{-1} may be stated as follows:

$$B^{-1} = \begin{bmatrix} G_m^{-1} + G_m^{-1}E_qQ^{-1}A_mG_m^{-1} & -G_m^{-1}E_qQ^{-1} \\ -Q^{-1}A_mG_m^{-1} & Q^{-1} \end{bmatrix}. \qquad (12)$$

where $Q = A_q - A_mG_m^{-1}E_q$.

The motivation for the partitioning of (11) is to factor out the matrix G_m, which must by definition be a basis for the underlying GN problem. As noted earlier, a basis for a GN problem may be viewed and stored as a general graph which contains only the nonzero components of the basis matrix. Due to the near-triangularity of G_m, any operations involving G_m^{-1} may be performed by traversing the associated general graph. This fact implies that the only information required to generate the basis inverse for the CGN problem, as

stated in (12), is the partitioned basis B and the (q x q) non-singular matrix Q^{-1}. As a result, the simplex operations for solving the CGN problem can be streamlined to improve computational efficiency and computer storage. Each of these operations will be described in detail in the subsequent section. However, before proceeding, it will be helpful to further explore the structure of G_m.

The characteristics of the general graph corresponding to G_m have been outlined in the literature [12,19,20]. It contains all of the vertices but only a subset of the edges of the original general graph corresponding to G. For this reason, it is called a spanning subgraph. Due to the non-singularity of G_m, the spanning subgraph is composed of a finite set of disjoint quasi-trees [12]. Each quasi-tree is a simple tree to which a single edge has been added. The additional edge generates exactly one cycle (a circular path) in the quasi-tree. Note that a loop (as defined previously) is a cycle containing a single vertex and a single edge.

Several computer list structures have been developed for storing the basis general graphs for GN problems and for implementing the associated arithmetic operations. A thorough knowledge of these structures and procedures is not a necessary prerequisite for understanding the operations developed subsequently. However, a general insight into such methodologies will be an aid to realizing the implied efficiency involved. A typical graph processing technique is the extended augmented predecessor index (EAPI) method [12]. This method is based on the triple-label procedures proposed by Johnson [15,16]. The EAPI method orients each quasi-tree so that the cycle is located at the top (a rooted cycle). Any subtrees incident on cycle vertices hang down from this rooted cycle. Associated with each cycle is

a cycle factor [11,12]. The cycle factor is used to facilitate solving the normally non-triangular square system of equations that the cycle represents.

One of the advantages of the EAPI method is its ability to perform operations that normally require a basis inverse by traversing the quasi-trees of the basis general graph. There are actually two types of quasi-tree traversal required, one normally associated with pre-multiplication by the inverse and the other normally associated with post-multiplication by the inverse. Pre-multiplication is generally used to find the representation of a column vector(edge) to enter the basis. The EAPI method performs this operation by tracing the unique paths from the vertices on which the edge is incident "up" to the associated cycle(s). The cycle portion(s) of the representation can be obtained by solving a one variable equation and then tracing a directed path once around the cycle(s) from the appropriate vertex back to itself. The efficiency of this operation derives from the fact that at most two quasi-trees are traversed and only the non-zero components of the representation are generated. Post-multiplication by the basis inverse is normally used to calculate the values of the pseudo dual variables. The EAPI method associates each dual variable with a vertex of the basis general graph. A dual variable associated with a cycle vertex of each quasi-tree may be determined by solving a one variable equation using the cycle factors [11,12]. The remaining dual variables can be determined by exploiting the triangularity of the resultant systems of equations and by locating all vertices lying "below" the selected cycle vertices in the quasi-trees. This process is often referred to as "pricing-out" the basis general graph.

The subsequent procedures assume that G_m is stored as a general graph. Each operation will be described both in algebraic terms and in terms of the

two types of quasi-tree traversal described above. It will be assumed that the matrix Q^{-1} of (12) is available in some form after each basis exchange step. Other portions of B^{-1} will be generated only as they are needed. The result is a highly efficient simplex method that keeps storage requirements to a minimum.

4.0 ALGORITHMIC STEPS

This section describes in detail the basic simplex operations as they pertain to the CGN problem. These operations include initialization, checking for optimality, finding the representation of the vector entering the basis, the basis exchange step, and calculating updated pseudo dual variable values. As noted in section 3, it will be assumed that the basis is partitioned as in (11), that G_m is stored as a general graph, and that Q^{-1} is the only portion of the basis inverse that is being kept.

4.1 Initialization

An initial basis B for the CGN problem can be obtained by selecting a basis G_m (perhaps optimal) for the underlying GN problem. The matrices E_q and A_q can be obtained by appending q slack or artificial variables to the problem that result in satisfying the constraints (3). Since $E_q = 0$, Q from (12) is equal to A_q and, hence, $Q^{-1} = A_q^{-1}$.

Once a basis has been selected, the complementary slackness conditions can be used to obtain initial pseudo dual variable values which satisfy:

$$(w^T, \delta^T) \begin{pmatrix} G_m & E_q \\ A_m & A_q \end{pmatrix} = \left(c_m^T, c_q^T \right). \qquad (13)$$

where $\left(c_m^T, c_q^T \right)$ is an appropriately ordered vector of objective function coefficients corresponding to B. In expanded form, (13) becomes:

$$w^T G_m + \delta^T A_m = c_m^T \tag{14}$$

$$w^T E_q + \delta^T A_q = c_q^T . \tag{15}$$

Equations (14) and (15) may be rewritten as follows:

$$w^T G_m = c_m^T - \delta^T A_m \tag{16}$$

$$\delta^T (A_q - A_m G_m^{-1} E_q) = c_q^T - c_m^T G_m^{-1} E_q . \tag{17}$$

Noting that $Q = A_q - A_m G_m^{-1} E_q$, (17) can be stated as:

$$\delta^T = (c_q^T - c_m^T G_m^{-1} E_q) Q^{-1} . \tag{18}$$

Assuming that the initial basis has been selected as described above, (18) implies that $\delta^T = c_q^T Q^{-1}$.

δ could be recomputed at future iterations using (18) and graph operations to find $G_m^{-1} E_q$. However, as q increases, the number of operations involved in this process becomes prohibitive. Thus, for large q, it is better to treat δ as an extra row of Q^{-1} and update it at each iteration.

Given an initial value of δ, (16) provides an excellent framework for the calculation of w. The system of equations of (16) is, in fact, a series of disjoint, near-triangular systems of equations. Once the right hand side of (16) has been calculated, the "pricing-out" procedure discussed in section 3 can be used to determine the value of w at each iteration.

4.2 Determination of Optimality

Let $\bar{P}_i = \begin{matrix} P_{m,i} \\ P_{q,i} \end{matrix}$ denote any column vector of the coefficient matrix A, where $P_{m,i}$ is associated with constraints (2) and $P_{q,i}$ is associated with (3). Optimality of both the primal and dual solutions occurs when the pseudo

dual solution is feasible. A determination of this can be made by first calculating updated objective coefficients associated with each primal column vector.

Thus,

$$\bar{c}_i = w^T P_{m,i} + \delta^T P_{q,i} - c_i. \qquad (19)$$

Dual feasibility is achieved when the following conditions are satisfied for all i:

$$\begin{aligned} \bar{c}_i &\leq 0, \quad x_i = 0 \text{ or } y_i = 0 \\ \bar{c}_i &= 0, \quad x_i - \text{basic or } y_i - \text{basic} \\ \bar{c}_i &\geq 0, \quad x_i = u_i \text{ or } y_i = v_i. \end{aligned} \qquad (20)$$

If any one of the conditions in (20) is not satisfied, then the associated primal column vector may be selected to enter the basis.

4.3 Finding the Representation of the Entering Vector

Let $\bar{P}_r = \begin{matrix} P_{m,r} \\ P_{q,r} \end{matrix}$ denote the column vector selected to enter the basis matrix. The representation $\bar{Z}_r = \begin{matrix} Z_{m,r} \\ Z_{q,r} \end{matrix}$ of $\bar{P}_r$ in terms of B must be computed so that the vector to leave the basis can be determined. This computation involves solving the following system of equations:

$$Z_{m,r} = G_m^{-1} P_{m,r} + G_m^{-1} E_q Q^{-1} A_m G_m^{-1} P_{m,r} - G_m^{-1} E_q Q^{-1} P_{q,r} \qquad (21)$$

$$Z_{q,r} = Q^{-1}(-A_m G_m^{-1} P_{m,r} + P_{q,r}). \qquad (22)$$

Substituting $Z_{q,r}$ into (21) yields

$$Z_{m,r} = G_m^{-1}(P_{m,r} - E_q Z_{q,r}). \qquad (23)$$

It is thus efficient to first calculate $Z_{q,r}$ and then to use this result to find $Z_{m,r}$. If $P_{m,r} = 0$ in (22), the computation of $Z_{q,r}$ reduces to $Z_{q,r} = Q^{-1}P_{q,r}$. If not, the graph traversal method described in section 3 can be used to compute $G_m^{-1}P_{m,r}$. As noted, this is the representation of the edge $P_{m,r}$ in terms of the basis general graph G_m. The EAPI method can perform this calculation very efficiently and will only find the non-zero components of the representation. Simultaneously, appropriately scaled columns of A_m can be accumulated with $P_{q,r}$. Upon completion, a multiplication by Q^{-1} can be performed to yield $Z_{q,r}$. Using this method, a ratio test should be performed immediately on $Z_{q,r}$ before the computation of $Z_{m,r}$ is made since degenerate pivots occur frequently in network problems and, therefore, unnecessary computations could be avoided.

The computation of $Z_{m,r}$ in (23) may actually be performed in two different ways, depending upon the particular structure of the problem being solved. If q is a relatively small number or if E_q contains a large number of zero column vectors, then (23) should be rewritten as $Z_{m,r} = G_m^{-1}P_{m,r} - G_m^{-1}E_qZ_{q,r}$. $G_m^{-1}P_{m,r}$ is simply the representation of the edge associated with $P_{m,r}$ and may be found by graph traversal. Likewise, $G_m^{-1}E_q$ is a matrix whose column vectors are the representations of the q edges associated with the columns of E_q. If q is small or if the number of actual edges is small, then this may be performed very efficiently by the graph traversal method discussed in section 3.

If, however, calculating $G_m^{-1}E_q$ a column at a time is too lengthy, then (23) shall be restated and solved in the form $G_mZ_{m,r} = (P_{m,r} - E_qZ_{q,r})$. In this form, finding $Z_{m,r}$ is equivalent to determining basic variable values in a GN problem with a right hand side of $(P_{m,r} - E_qZ_{q,r})$. Again

due to the near-triangularity of G_m, this determination can be easily performed. Simply stated, this involves assigning values to terminal edges of the quasi-trees, successively working up toward the rooted cycles. Values associated with cycle edges can then be calculated using the appropriate cycle factors.

4.4 The Basis Exchange Step

Let $\bar{P}_s = \begin{matrix} P_{m,s} \\ P_{q,s} \end{matrix}$ denote the vector selected to leave the basis.

For notational convenience, it will be assumed that $\bar{P}_s$ occupies the s^{th} position in B. The position of $\bar{P}_s$ plays an important role in determining how the basis exchange is to be performed. An indiscriminate pivot may result in the destruction of the basis partitioning.

In order to update Q^{-1} and δ during a pivot, it is necessary to find the last q entries in the s^{th} row of B^{-1}. Let R_s denote this vector. If $s > m$, then R_s is simply the $(s-m)^{th}$ row of Q^{-1}. However, if $s \leq m$, then the s^{th} row of $(-G_m^{-1}E_qQ^{-1})$ must be determined. This can be efficiently achieved by first determining the s^{th} row of G_m^{-1}. The poly-w technique of Charnes and Cooper [3] can easily be applied to the basis general graph to generate the desired row of G_m^{-1}. This procedure uses the s^{th} unit vector as a pseudo cost vector and the "pricing-out" graph traversal method described in section 3. The resulting dual variable values are the appropriate entries in the s^{th} row of G_m^{-1}. This process only involves traversing a single quasi-tree, since all other costs are zero, and then only that portion of the quasi-tree lying below the edge associated with $P_{m,s}$ (all other dual variables will be zero). As these entries are determined, the appropriately scaled rows of E_q can be accumulated in a vector T_s (this vector may also be used in subsequent calculations). Upon completion, T_s is complemented

and multiplied times Q^{-1} to yield R_s.

Once R_s is known, consideration must be given to the problem of properly maintaining the basis partitioning. The following cases describe the appropriate basis exchange procedures based on the position of $\bar{P}_s$ in B.

Case 1: The minimum ratio occurred in $Z_{q,r}$ (i.e., $s > m$). In this case the basis exchange may be performed directly. Neither G_m nor A_{im} will be affected by this exchange. Q^{-1} and δ may be updated using $Z_{q,r}$ and R_s. E_q and A_q are modified by the insertion of $\bar{P}_r$ in place of $\bar{P}_s$.

Case 2: The minimum ratio occurs in $Z_{m,r}$ (i.e., $s \leq m$). This case actually must be divided into two subcases. The determining factor is whether or not the basis exchange will result in a G_m that is a basis for the underlying GN problem. This can be checked by looking at the representation of $P_{m,r}$ in terms of G_m. If the s^{th} position of this vector is zero, then the exchange will destroy the partitioning B.

Case 2(a): The s^{th} component of $G_m^{-1}P_{m,r}$ is zero. If the basis exchange is performed directly, the resulting G_m will be singular. The following remark is useful in resolving this issue.

Remark: If the s^{th} component of $G_m^{-1}P_{m,r}$ is zero, there must exist a basis vector $\bar{P}_t$, where t denotes the position of P_t in the basis and where $m < t \leq m + q$, such that when switched with $\bar{P}_s$ in the basis results in a non-singular G_m. This vector can be determined by assigning t to be the position of any non-zero component of T_s.

Proof: T_s corresponds to the s^{th} row of $G_m^{-1}E_q$. $G_m^{-1}E_q$ contains the representations of the q column vectors of E_q in terms of G_m. Using elementary linear algebra, it is known that the s^{th} vector of G_m can be

replaced by the t^{th} vector of E_q as long as the t^{th} component of T_s is non-zero. It is impossible for T_s to be a zero vector since this would imply (via (23)) that the s^{th} component of $Z_{m,r}$ is zero. This result would be a contradiction of the ratio test.

Once t has been determined, the basis must be reordered by switching $\bar{P}_t$ and $\bar{P}_s$ in B. The general graph G_m must be modified by removing the edge associated with $P_{m,s}$ and inserting the edge associated with $P_{m,t}$. Q^{-1} must also be modified by setting the $(t-m)^{th}$ row of Q^{-1} equal to R_s. After this interchange, the actual basis exchange of $\bar{P}_r$ for $\bar{P}_s$, where now $s > m$, may be performed as in case 1.

Case 2(b): The s^{th} component of $G_m^{-1}P_{m,r}$ is non-zero. In this case, the basis exchange may be performed directly since the resulting G_m will be non-singular. The basis graph can be modified by deleting the edge associated with $P_{m,s}$ and inserting the one associated with $P_{m,r}$. Q^{-1} and δ can be updated by a standard pivot using R_s and $Z_{q,r}$.

All of the basis exchange operations described above should involve both updating the variable vectors x and y and updating the partial vector of pseudo dual variables δ. The only other operation needed to complete the exchange is to compute the vector w. This can be done using (16) and the procedure discussed in section 4.1. Upon completion, the new pseudo dual variables would be checked for feasibility as in section 4.2.

4.5 Conclusion

Many of the advances in the solution of LP problems in recent years have been the result of exploiting special structures. One of the major areas of research has been in the solution of network related problems. A few of the reasons for this interest in networks are that they are easily visualized, that they occur frequently in real-world applications, and

that highly efficient techniques are being developed for their solution. This paper extends the usefulness of these procedures to a much broader class of problem.

The efficiency evident in these procedures for solving constrained generalized network problems draws attention to the importance of the formulation process. A large number of problems could be solved using these procedures, if they are properly formulated or transformed. To this end, future research is needed to identify general classes of problems to which these procedures can be applied and to explore transformation procedures that maximize the efficiency of these methods.

BIBLIOGRAPHY

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

2. G. Bradley, G. Brown, and G. Graves, "A Comparison of Storage Structure for Primal Network Codes," Presentation at the ORSA/TIMS conference, Chicago, April 1975.

3. A. Charnes and W. Cooper, Management Models and Industrial Applications of Linear Programming, Vols. I and II, Wiley, New York, 1961.

4. G. Dantzig, Linear Programming and Extensions, Princeton University Press, Princeton, New Jersey, 1963.

5. N. Deo, Graph Theory with Applications to Engineering and Computer Science, Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1974.

6. F. Glover, D. Karney, and D. Klingman, "The Augmented Predecessor Index Method for Locating Stepping Stone Paths and Assigning Dual Prices in Distribution Problems," Transportation Science, 6, 2 (1972), 171-179.

7. F. Glover, D. Karney, and D. Klingman, "Implementation and Computational Comparisons of Primal, Dual, and Primal-Dual Computer Codes for Minimum Cost Network Flow Problems," Networks, 4, 3 (1974), 191-212.

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

9. F. Glover, D. Karney, D. Klingman, and R. Russell, "Solving Singularly Constrained Transshipment Problems," Research Report CCS212, Center for Cybernetic Studies, University of Texas at Austin, 1974.

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

11. F. Glover and D. Klingman, "A Note on Computational Simplifications in Solving Generalized Transportation Problems," Transportation Science, 7, 4 (1973), 351-361.

12. F. Glover, D. Klingman, and J. Stutz, "Extensions of the Augmented Predecessor Index Method to Generalized Network Problems," Transportation Science, 7, 4 (1973), 377-384.

13. F. Glover, D. Klingman, and J. Stutz, "Implementation and Computational Study of a Generalized Network Code," Presentation at the 44th National ORSA Conference, San Diego, California, 1973.

Bibliography, Continued

14. F. Glover, D. Klingman, and J. Stutz, "The Augmented Threaded Index Method for Network Optimization," INFOR, 12, 3 (1974), 293-298.

15. E. Johnson, "Programming in Networks and Graphs," ORC Report 65-1, University of California at Berkeley, 1965.

16. E. Johnson, "Networks and Basic Solutions," Operations Research, 14, 4 (1966), 619-623.

17. D. Karney and D. Klingman, "Implementation and Computational Study on an In-Core Out-of-Core Primal Network Code," To appear in Operations Research, 24 (1976).

18. D. Klingman and R. Russell, "On Solving Constrained Transportation Problems," Operations Research, 23, 1 (1975), 91-107.

19. R. Langley, "Continuous and Integer Generalized Flow Problems," Unpublished Dissertation, Georgia Institute of Technology, 1973.

20. J. Maurras, "Optimization of the Flow Through Networks with Gains," Mathematical Programming, 4, 2 (1972), 135-145.

21. J. Mulvey, "Column Weighting Factors and Other Enhancements to the Augmented Threaded Index Method for Network Optimization," Joint ORSA/TIMS Conference, San Juan, Puerto Rico, 1974.

22. R. McBride, "Factorization in Large-Scale Linear Programming," Working Paper #220, Western Management Science Institute, University of California at Los Angeles, 1973.

23. V. Srinivasan and G. Thompson, "Accelerated Algorithms for Labeling and Relabeling of Trees with Applications for Distribution Problems," JACM, 19, 4 (1972), 712-726.

24. V. Srinivasan and G. Thompson, "Benefit-Cost Analysis of Coding Techniques for the Primal Transportation Algorithm," JACM, 20 (1973), 194-213.