An Analysis of the LOGAIR Distribution System Using Optimization--ETC(U)

Nov 81  J L Kennington

AFOSR-TR-82-0029

UNCLASSIFIED
AN ANALYSIS OF THE LOGAIR DISTRIBUTION SYSTEM USING OPTIMIZATION PRINCIPLES

Jeff L. Kennington
Department of Operations Research
Southern Methodist University
Dallas, Texas 75275

November 1981
# AN ANALYSIS OF THE LOGAIR DISTRIBUTION SYSTEM USING OPTIMIZATION PRINCIPLES

**Authors:** Jeff L. Kennington

**Performing Organization:** Department of Operations Research, Southern Methodist University, Dallas TX 75275

**Contract or Grant Number:** AFOSR-77-3151

**Program Elements, Projects, Tasks, Areas & Work Unit Numbers:**
- 61102F; 2304/A6

**Controlling Office:** Mathematical & Information Sciences Directorate, Air Force Office of Scientific Research, Bolling AFB DC 20332

**Report Date:** Nov 81

**Number of Pages:** 40

**Security Class. (of this Report):** UNCLASSIFIED

**Distribution Statement (of this Report):** Approved for public release; distribution unlimited.

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

**Supplementary Notes:**

**Key Words:**

**Abstract:**

This report presents a summary of the research sponsored under the grant. The major objective of the research was to build mathematical programming models and specialized software to assist Air Force personnel at Wright-Patterson AFB in the design of the LOGAIR Distribution System. A description of the problem, the mathematical models developed, and the software developed is presented. The software has been documented and installed at Wright-Patterson and is currently being used by Air Force Logistics Command personnel. (CONTINUED)
ITEM #20, CONT.

A secondary objective of the principal investigator was to address the issue of how one can automatically convert general linear programs into network models. The underlying mathematical results which can be used to develop either exact transformation algorithms or heuristic transformation algorithms are presented in Section III. This section concludes with a heuristic algorithm which, it is believed, holds the best hope for routinely converting linear programs into network programs or network programs with extra constraints.
I. INTRODUCTION

This report presents a summary of the research sponsored by AFOSR under Grant Number 77-3151. This research was performed during the years 1976 through 1981 by the principal investigator and a group of graduate students.

The major objective of the research was to build mathematical programming models and specialized software to assist Air Force personnel at Wright-Patterson AFB in the design of the LOGAIR Distribution System. A description of the problem, the mathematical models developed, and the software developed is presented in Section II. The software has been documented and installed at Wright-Patterson and is currently being used by Air Force Logistics Command personnel.

A secondary objective of the principal investigator was to address the issue of how one can automatically convert general linear programs into network models. The underlying mathematical results which can be used to develop either exact transformation algorithms or heuristic transformation algorithms are presented in Section III. This section concludes with a heuristic algorithm which, I believe, holds the best hope for routinely converting linear programs into network programs or network programs with extra constraints.

The list of technical reports which have been either fully or partially supported by this grant are as follows:

AIR FORCE OFFICE OF SCIENTIFIC RESEARCH (AFOSR)
NOTICE OF TRANSMITTAL TO DTIC
This technical report has been reviewed and is approved for public release JAN 1981. Distribution is unlimited.
MATTHEW J. KERKER
Chief, Technical Information Division


II. THE LOGAIR DISTRIBUTION SYSTEM

This section reports on the successful application of mathematical programming in a decision support system for the Air Force Logistics Command. A complementary pair of multicommodity network flow models is used to aid Air Force personnel in designing the air cargo network and shipment plan (LOGAIR) utilized by the Air Force to support sixty bases in the continental U. S. A. State-of-the-art software was developed to solve these models and this software is currently being used at Wright-Patterson AFB in an integrated man-machine system to aid Air Force personnel in making annual design changes in the route structure.

2.1 Problem Description

The U. S. Air Force has major repair facilities which are responsible for the maintenance of serviceable spares for all aircraft, missiles, and ground radar systems. When a subsystem fails, it is removed and replaced by an operating subsystem. The failed system is shipped to the repair facility, repaired, and returned to either the base of origin or to inventory. During 1980, for example, over 2700 tons of serviceable spares were shipped from Wright-Patterson AFB in Ohio to Tinker AFB in Oklahoma and for the entire system of 60 bases, well over 100,000 tons of cargo was moved. Due to the magnitude of these shipping requirements, the Air Force maintains a separate air cargo system for shipment of these serviceable spares. Each year Air Force Logistics Command (AFLC) personnel develop a daily air cargo shipment plan to be used for the entire fiscal year. This section reports on a complementary pair of multicommodity network flow models used to aid Air Force personnel in designing the air cargo network and shipment plan.
2.2 Survey of Literature

Due to numerous applications, routing and scheduling problems have been extensively studied in the operations research literature. Invariably, simplifying assumptions are made to specialize a problem for a given situation (e.g. [19, 20, 41]). Many characteristics of the Air Force cargo shipment plan design problem are also present in the areas of bus, train, and ship routing.

In school bus routing problems one is concerned with routing in a single period and with only a single destination. Problems in this class are almost always approached with a heuristic method based on a modification of the nearest unvisited city procedure developed for the traveling salesman problem (e.g. [6, 48, 56]).

Silman, Barzily, and Passy [50] present heuristic procedures for developing schedules for city buses. They propose a two phase approach for devising bus routes and schedules. Phase I obtains a set of potential routes while the second phase gives the frequency of travel. Their general approach is adaptable to the Air Force problem but their specific heuristic is specialized for only routing buses. Billheimer and Gray [12] address the general fixed-charge multicommodity network flow problem in the context of Mass Transportation Network Design. However, they assume that all arcs have infinite capacity which greatly simplifies the solution procedure.

Ferguson and Dantzig [18] present a model for assigning aircraft to routes. However, they assume the routes given and ignore all fixed charges. Bellmore, Bennington, and Lubore [9] present a model for assigning tankers to shipping routes to maximize a utility function. They view the tankers as the commodities and assume a possible loading after the tankers have been assigned to routes. Again the routes are assumed given and there are no fixed charges incurred for using a shipping lane. A similar study on the movement of train cars over a rail system was conducted by White.

-4-
and Wrathall [58]. Unlike the Air Force problem, the network topology and schedules are input for their system. Geoffrion and Graves [21] solved a large warehouse location distribution problem for Hunt-Wesson Foods, Inc. and; Marsten and Muller [45] solved some special models for the Flying Tiger Line, but these models are not applicable for the Air Force problem.

Demmy and Brant [15], in an early paper, were the first to model the Air Force problem. Their model was a large linear program with GUB constraints. Agin and Cullen [1] present a model for the general vehicle routing and scheduling problem, and Richardson [49] presents a routing model for commercial airline schedule planning. Unfortunately, these models when applied to the Air Force problem produce mixed integer programs for which there is little hope of finding an efficient solution procedure.

2.3 The General Decision Support System

The AFLC defines a cargo-route as a sequence of bases and an aircraft type such that the first base and the last base in the sequence are the same. This guarantees that both the aircraft and crew are returned to the home base. Two major types of aircraft, the Lockheed L100 and L188, are currently being used in the air cargo system. The characteristics of these aircraft are given in Table 1. The sequence of bases {Tinker, Hill, Travis, Robbins, Tinker} along with the Lockheed L100 is a cargo-route in which the fixed costs (cost for flying the route with an empty L100), the variable cost (fuel consumption cost as a function of cargo weight for the L100), and cargo capacity are known. A set of cargo-routes for the 60 base system is called an air cargo plan.
TABLE 1. AIRCRAFT CHARACTERISTICS (1980 DATA)

<table>
<thead>
<tr>
<th>Aircraft Characteristics</th>
<th>L100</th>
<th>L188</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transportation Cost ($/mile)</td>
<td>3.5260</td>
<td>2.4128</td>
</tr>
<tr>
<td>Fuel Consumption Cost ($/mile)</td>
<td>0.9936</td>
<td>1.0304</td>
</tr>
<tr>
<td>Empty Weight (lbs)</td>
<td>74,746</td>
<td>56,013</td>
</tr>
<tr>
<td>Full Weight (lbs)</td>
<td>120,746</td>
<td>86,538</td>
</tr>
<tr>
<td>Usable Cargo Capacity (lbs)</td>
<td>43,160</td>
<td>28,640</td>
</tr>
<tr>
<td>Total Cargo Capacity (Cargo plus pallets) (lbs)</td>
<td>46,000</td>
<td>30,525</td>
</tr>
<tr>
<td>Variable Component of Cost ($/mile)</td>
<td>0.87703x10^{-5}</td>
<td>1.2691x10^{-5}</td>
</tr>
<tr>
<td>Fixed Component of Cost ($/mile)</td>
<td>3.1475</td>
<td>2.0493</td>
</tr>
</tbody>
</table>

-6-
Based on forecasted cargo shipment data, each year AFLC personnel develop an air cargo plan to be flown on a daily basis for the next fiscal year. The routes in the plan are flown by civilian carriers and are not subject to change during the fiscal year. **The objective of AFLC personnel is to select the least cost set of cargo-routes which satisfy the point-to-point demands for cargo movement among 60 Air Force Bases.**

Following the work of Agin and Cullen [1] a global optimization model of this planning problem can be developed. This global model is a mixed integer program with over 3 million continuous variables and over 60 thousand binary variables. In contrast to the above approach, we have chosen to develop an integrated man-machine system which may be used in the development of an air cargo plan. The three inputs of this system are as follows:

(i) **aircraft characteristics (Table 1),**
(ii) a 60 by 60 cargo forecast matrix, and
(iii) a 60 by 60 distance matrix which gives the flight distance between all pairs of bases.

Using only the cargo forecast matrix and the distance matrix as inputs, a nominal set of cargo-routes is produced. The nominal set is selected in such a way that the system pound-miles is a minimum. These routes use hypothetical aircraft and may violate constraints on the length of time a crew travels before it returns to the home base.

Using the aircraft available and taking into consideration other system constraints, Air Force personnel modify the nominal cargo-routes to form a set of potential routes. An integer programming problem is
then solved to develop an air cargo plan from the set of potential routes. The analytical tools, input data, and the man-machine interaction is illustrated in Figure 1.

2.4 The Flow Generator and Route Selector Models

We now present the mathematical notation used to define a cargo-route. A network \( G = [N, A] \) consists of a node set \( N \) and a set of ordered pairs of nodes \( A = \{e_1, \ldots, e_t\} \). A circuit is defined to be a finite sequence of the form \( (s_1, s_2, s_3, \ldots, s_m, s_1) \) where \( s_i \in N \) and each pair \( (s_i, s_j) \in A \). A circuit along with an aircraft type specifies a cargo-route.

Let \( A \) denote the node-arc incidence matrix for a network and let \( C \) denote the set of arcs in some circuit in the network. Let \( y \) be any vector such that \( Ay = 0 \). Such a vector has been referred to as a flow by Berge and Ghouila-Houri [11]. Let

\[
z_j = \begin{cases} 
1, \text{ if the } j\text{th arc is a member of } C, \\
0, \text{ otherwise.}
\end{cases}
\]

Then the vector \( z \) is a flow and will be referred to as a vector-circuit corresponding to \( C \).

For the Air Force problem, we use a linear program to obtain a vector \( y \) satisfying \( Ay = 0 \) and \( y > 0 \). We then apply a simple labeling algorithm to decompose \( y \) into a set of vector-circuits and nonnegative multipliers such that \( y = \sum_{i=j}^{p} a_i z^i \). The problem of finding a basis of cycles in a graph has been extensively studied in the literature (see [2, 10, 46, 47, 56]). The \( z^i \) may be viewed as a set of nominal cargo-routes each with aircraft having capacity of at least \( a_i \).
Figure 1. Data and codes used to develop an Air Cargo Plan.
Letting \( N \) denote the number of bases, there are a potential of \( N(N-1) \) arcs (i.e., total arcs in a complete network). Suppose \( M < N(N-1) \) arcs are selected for consideration and let \( A \) denote the corresponding \( N \) by \( M \) node-arc incidence matrix. Let \( c_j \) for \( j = 1, \ldots, M \), denote the flying distance associated with each of these arcs, and let \( c \) denote the vector of distances. For each pair of bases, \((i,j)\), let \( d_{ij} \) denote the total quantity of serviceable spares to be shipped from base \( i \) to base \( j \) in units of pounds per day. Since the capacity of the aircraft must be shared by all goods with various origin-destination pairs, these must be distinguished in the model. For our models the commodities are associated with the nodes of origin. We let the node length vector \( r^k \) denote the requirement vector for commodity \( k \). If \( r^k_i > 0 \), then node \( i \) is called a supply point for commodity \( k \) with supply of \( r^k_i \). If \( r^k_i < 0 \), then node \( i \) is called a demand point for commodity \( k \) with demand of \( |r^k_i| \). For the Air Force problem the requirement vectors are defined as follows:

\[
 r^k_i = \begin{cases} 
 -d^k_{ki}, & k \neq i \\
 \sum_{j=1}^{N} d_{ij}, & k = i \end{cases} 
\]

\( i = 1, \ldots, N, \) \( k = 1, \ldots, N, \) \( j = 1, \ldots, N, \) \( k \neq i \)

Letting \( x_j^k \) denote the flow of commodity \( k \) in arc \( e_j \) with corresponding vector \( x^k \), the Flow Generator Model seeks a flow \( x \) which satisfies the demand while minimizing system pound-miles. Mathematically the Flow Generator Model may be stated as follows:

-10-
Given an optimal solution to (1), say \( (\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_N, \bar{y}) \), we decompose \( \bar{y} \) into a set of vector-circuits, \( \bar{z}_1, \ldots, \bar{z}_p \), and positive multipliers \( a_1, \ldots, a_p \), such that \( \bar{y} = \sum_{i=1}^{p} a_i \bar{z}_i \). The vector-circuits define nominal routes for the system. The algorithm used to obtain the vector-circuits may be found in [5].

Using the nominal routes as a guide, a set of approximately 25 feasible routes are input to an integer program for final route selection. Suppose there are \( L \) routes in the feasible set. Let the set \( R_\ell = \{ e_{j_1}, e_{j_2}, \ldots, e_{j_q} \} \) denote the arcs in route \( \ell \). Let the arc set be given by

\[
A_\ell = \bigcup_{\ell=1}^{L} R_\ell.
\]

Then the network used in the Route Selector Model is \([N, \hat{A}]\)

where \( N = \{1, \ldots, N\} \). Let \( \hat{A} \) denote the node-arc incidence matrix associated with \([N, \hat{A}]\). Letting \( f_\ell \) and \( b_\ell \) denote the fixed charge and aircraft capacity for route \( \ell \), respectively; the Route Selector Model is given by

\[
\min \sum_{k=1}^{N} c_k x_k + \sum_{\ell=1}^{L} f_\ell y_\ell
\]

s.t. \( \hat{A} x = z_k \), \( k = 1, \ldots, N \) \hspace{1cm} (3)

\[
\sum_{k=1}^{N} x_j^k \leq b_\ell, \quad \text{for all } e_j \in R_\ell, \quad \text{and } \ell = 1, \ldots, L
\]

(4)
where $M^*$ is a large positive number. Constraint (4) insures that the aircraft capacity is not exceeded and constraint (5) forces the binary route variable $y_{\ell}$ to assume the value of 1 if route $\ell$ is used. The above model is a multicommodity fixed charge network flow problem. Solution of (2) - (7) provides a set of optimum routes from the set of $L$ feasible routes which if flown daily will guarantee that the daily demand is met subject to aircraft capacity constraints. The underlying assumptions associated with this model are as follows:

(i) All cargo has the same priority.

(ii) Loading and unloading costs have been ignored.

(iii) Cargo volume restrictions have been ignored. (However, these can be incorporated into the model at the expense of increasing the number of constraints).

(iv) Circuitous routing is allowed to meet the demand constraints.

The integrated man-machine system used to develop an air cargo plan is illustrated in Figure 2.
Flow Generator Model
Linear Program
Rows = 4000
Cols = 18,000

Flow Vector: $y$

Circuit Decomposer

Nominal Cargo-Routes: $z_1, \ldots, z_p$
Nominal Aircraft Capacities: $a_1, \ldots, a_p$

Air Force Personnel

Set of Feasible Routes (20 to 25 cargo-routes)

Route Selector Model
Mixed 0-1 Integer Program
Rows = 3400
Cols = 9300
Binary Var = 20 to 25

Air Cargo Plan (15-18 Routes)

Figure 2. Procedure used to develop an Air Cargo Plan.
2.5 Computational Experience

The primal partitioning code for solving multicommodity network flow problems reported in [4] has been specialized for the Flow Generator Model (see [31] for a complete description of the primal partitioning algorithm). This system carries the inverse of the working basis in product form using the technique described in [31]. The reinversion routine is based on the work of Hellerman and Rarick [36] and uses the spike swapping procedure described in [32]. A simple circuit identifier algorithm has also been coded. Both codes are written in standard FORTRAN and have been run on a CDC Cyber 73.

The Civil Aeronautics Board provided the distance matrix for the 60 Air Force Bases in the continental U. S. A. and the Air Force Logistics Command provided the point-to-point demands \((d_{ij})\) for the fiscal years 1979 and 1980. From this data two test problems for each year were generated. The two test problems differ only in the number of arcs used to define the network used in the Flow Generator Model.

The termination criterion used for problems 2 and 4 was to check the objective function every 1000 iterations and terminate if the objective function value became less than \(5 \times 10^8\). This number was selected arbitrarily, though keeping in mind that the routes generated by AFLC personnel for 1980 yielded a cost of \(222.5 \times 10^8\) pound-miles. The two smaller problems, problems 1 and 3, were solved to optimality. Table 2 summarizes relevant information obtained in solving these problems. Note that the vector-circuit generator takes only a few seconds while the multicommodity code requires more than 20 minutes to obtain an acceptable solution.
TABLE 2. COMPUTATIONAL EXPERIENCE WITH ROUTE GENERATOR MODEL

<table>
<thead>
<tr>
<th>Row Description</th>
<th>Problem Number</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>PROBLEM CHARACTERISTICS</td>
<td></td>
</tr>
<tr>
<td>Nodes</td>
<td>60</td>
</tr>
<tr>
<td>Arcs</td>
<td>249</td>
</tr>
<tr>
<td>Commodities</td>
<td>61</td>
</tr>
<tr>
<td>LP rows</td>
<td>3967</td>
</tr>
<tr>
<td>LP columns</td>
<td>15009</td>
</tr>
<tr>
<td>Linking Constraints</td>
<td>246</td>
</tr>
<tr>
<td>Data Source Year</td>
<td>1979</td>
</tr>
<tr>
<td>SOLUTION STATISTICS</td>
<td></td>
</tr>
<tr>
<td>Objective function value x 10^9</td>
<td>.36235</td>
</tr>
<tr>
<td>Termination Criterion *</td>
<td>optimal</td>
</tr>
<tr>
<td>Time in CP minutes</td>
<td>21</td>
</tr>
<tr>
<td>Iterations</td>
<td>11,605</td>
</tr>
<tr>
<td>Reinversions</td>
<td>77</td>
</tr>
<tr>
<td>Time for vector-circuit selector in seconds</td>
<td>6.3</td>
</tr>
</tbody>
</table>

*Conditional termination when obj value < 5 x 10^9.
We also designed and implemented a large-scale FORTRAN computer code to be used in obtaining the solution of the Route Selector Model. The code employs a branch-and-bound scheme with separation and candidate selection guided by heuristic rules. The free integer variable furthest from an integral value is chosen for separation. The candidate subproblems most recently created are chosen first with preference given to those whose separation variable was fixed at 1 when created. The branch-and-bound tree was kept on disk in groups of 16 nodes, making use of the CDC mass storage input/output subroutines. The system was designed to allow the user to terminate a run with the current branch-and-bound tree and later restart with that tree.

It is shown in [33] that the continuous relaxation of a candidate subproblem can be formulated as a minimum cost multicommodity network flow problem. Thus we make use of a specialization of the primal partitioning code of [4] for efficient solution of the relaxed candidate subproblems. The route selector system was tested on the 1980 data using 17 routes supplied by AFLC personnel and 8 routes developed using the Flow Generator Model. This yielded a fixed charge multicommodity model having 25 binary variables, 9349 continuous variables, and 3355 constraints.

Beginning with an initial feasible solution supplied by AFLC personnel, the system was used to generate a branch-and-bound tree having 1023 nodes. This required 46 restarts and took approximately 23 hours of computer time over a 3 week period. At the termination of the run, there were 15 nodes remaining in the candidate list. Only one new incumbent was developed during the computation but the estimated cost savings of this incumbent was approximately $800,000. The new route structure involved the substitution of one of the 8 routes generated by the Flow Generator Model for one of the original 17 supplied by the Air Force.
2.6 Implementation

The two models and specialized software systems described above evolved over the period 1976 - 1980. All code development was done at Southern Methodist University by the authors for the Directorate of Transportation located at Wright-Patterson Air Force Base in Dayton, Ohio. Transportation personnel had many years of experience with the Air Force system, but they had little background in mathematical analysis and no background in either mathematical or computer programming. Even though AFLC personnel were unfamiliar with optimization models, the problem was ideally suited for operations research analysis. The important characteristics which made this study feasible are as follows:

(i) The problem was well-defined.

(ii) It was a planning (as opposed to an operational) problem in which the plan was reevaluated annually.

(iii) The problem involved a large cash outlay, $50,000,000. Hence a 1% savings was very significant.

(iv) Most of the data was already being collected and stored on magnetic tape. There was essentially no new data which had to be collected by the client.

(v) The client had been attempting to solve the problem manually and had an appreciation for the complexity of the problem.

Rather than implement both models simultaneously, we chose to install only the Route Selector Model in which all binary variables are fixed by the user. The user selects the routes and the system loads the routes to
optimally satisfy the demand. An elaborate report generator was attached to this system to provide the client with detailed information about flow in the system. In particular, legs of routes running at 100% capacity and underutilized legs are highlighted. This system has been implemented at Wright-Patterson and was used to develop the air cargo plan for fiscal year 1981. The client was very pleased with this basic system and was able to run the system and interpret the results without the aid of the authors. The system is currently being used to develop the annual routing plan.
Since the development of the primal simplex method by George B. Dantzig in 1947, linear programming has been used as a fundamental planning tool for solving a wide variety of problems in industry and government. Due to the development of extremely efficient solution algorithms, a special class of linear programs known as network models have emerged as one of the most important models available to operations research analysts. Since the constraint matrices of real world linear programs usually have only a few nonzero elements (i.e. more than 98% of the matrix elements are zero) we are convinced that these problems either contain large embedded networks or can be transformed to a problem which contains a large network. If this is the case, then techniques which combine linear programming technology with network technology can be used to solve such problems.

The underlying hypothesis of this project is that general linear programs can best be solved by transforming them to networks with side constraints. Unlike the theory of linear programming which is based on mathematical results from linear algebra and convex analysis, this investigation is cast in the mathematical framework of matroid theory.
3.1 Linear Network Models

A network is composed of two types of entities: arcs and nodes. The arcs may be viewed as unidirectional means of commodity transport, and the nodes may be interpreted as locations or terminals connected by the arcs. Hence, arcs may represent aircraft flights in a distribution system, streets and highways in an urban transportation network, pipes in a water distribution network, telephone lines in a communication network, and so on. The structure of a network can be displayed by means of a labeled drawing in which nodes are represented by circles and arcs are represented by line segments incident on two nodes. An arrowhead on the line segment indicates the arc direction.

The structure of a network may also be described by a matrix, defined as follows:

$$ A_{ij} = \begin{cases} +1, & \text{if arc } j \text{ is directed away from node } i \\ -1, & \text{if arc } j \text{ is directed toward node } i \\ 0, & \text{otherwise} \end{cases} $$

The matrix $A$ defined above is called a node-arc incidence matrix. A characteristic of this matrix is that each column has exactly two non-zero entries, one being $+1$ and the other a $-1$. Any matrix (regardless of origin) having this characteristic is called a node-arc incidence matrix.

The minimal cost network flow problem is a linear program whose constraint matrix is a node-arc incidence matrix. Mathematically this problem may be stated as follows:
\[
\begin{align*}
\text{min} \quad & cx \\
\text{s.t.} \quad & Ax = r \\
& 0 \leq x \leq u
\end{align*}
\]

where

- \( c \) is a known \( 1 \times n \) vector,
- \( u \) is a known \( n \times 1 \) vector,
- \( 0 \) is an \( n \times 1 \) vector of zeroes,
- \( r \) is a known \( m \times 1 \) vector,
- \( A \) is an \( m \times n \) node-arc incidence matrix, and
- \( x \) is an \( n \times 1 \) vector of decision variables.

Since the expository papers by Ellis Johnson [39, 40] in 1965, tremendous advances have been made in the area of solution techniques for network related problems. This work was led primarily by Glover and Klingman and their colleagues at the University of Texas (see [3, 17, 22, 23, 24, 25, 26, 27, 28, 29]). Contributions have also been made by Srinivasan and Thompson [51, 52] and by Bradley, Brown, and Graves [14]. The author and his colleagues at Southern Methodist University have been actively extending these ideas to the more complicated network structure found in multicommodity network flow problems (see [3, 4, 32, 42, 43, 44]). Computational experimentation has shown that the new methodology is approximately 200 times faster on pure networks and as much as 25 times faster on more complex embedded network problems.

We call a matrix \( M \) a \textbf{network matrix} if it has the following three properties:
(P1) The nonzero entries of $M$ are either +1 or -1.

(P2) No column of $M$ has more than two nonzero entries.

(P3) If a column of $M$ has two nonzero entries, then one is a +1 and the other is a -1.

A network matrix $M$ can be transformed to a node-arc incidence matrix by simply appending a row which is the negative of the sum of all other rows. Consider

$$
M = \begin{bmatrix}
1 & 1 & & & -1 & -1 \\
& -1 & -1 & 1 & 1 \\
& & & & & \\
& & & & & \\
\end{bmatrix}
$$

Appending a row which is the negative of the sum of the other rows yields the node-arc incidence matrix,

$$
A = \begin{bmatrix}
1 & 1 & & & -1 & -1 \\
-1 & -1 & 1 & 1 & & \\
& & & & & \\
& & & & & \\
\end{bmatrix}
$$

which corresponds to the network
Hence, any linear program whose constraint matrix is a network matrix may be solved as a minimal cost network flow problem.

3.2 Reducibility of a Linear Program

Since pure network problems are at least two orders of magnitude easier to solve than general linear programs, several researchers have addressed the following problem:

"When is a general linear program reducible (i.e. transformable) to a minimal cost network flow problem?"

Consider the general linear program in the following form:

\[ \begin{align*}
\text{min } & \quad cx \\
\text{s.t. } & \quad \tilde{A}x = b \\
& \quad 0 \leq x \leq u,
\end{align*} \]

where

- \( c \) is a known \( 1 \times n \) vector,
- \( u \) is a known \( n \times 1 \) vector,
- \( 0 \) is an \( n \times 1 \) vector of zeroes,
- \( \tilde{A} \) is an \( m \times n \) matrix, and
- \( x \) is an \( n \times 1 \) vector of decision variables.

Suppose \( \tilde{A} \) takes the form \( \tilde{A} = [I; A] \). This form is always obtainable by the addition of artificial variables with corresponding bound, \( u_i \), equal
Let $T$ be an $m \times m$ nonsingular matrix, let $R$ be an $m \times m$ nonsingular diagonal matrix, and let $D$ be an $n \times n$ nonsingular diagonal matrix. Letting $x = Dy$ and premultiplying (9) by $TR$ yields the equivalent linear program,

\[
\min \quad cv \\
\text{s.t.} \quad Ax = b \\
0 \leq Dy \leq u
\]

where
\[
c = cD, \\
\hat{A} = TRAD, \quad \text{and} \\
\hat{b} = TRb.
\]

The problems (8) - (10) and (11) - (13) are equivalent in the sense that if $x^*$ solves (8) - (10), then $y^* = D^* x^*$ solves (11) - (13) and if $y^*$ solves (11) - (13), then $x^* = Dy^*$ solves (8) - (10). Furthermore, we say that (8) - (10) is reducible to a network problem if and only if $TRAD$ is a network matrix (see properties (P1) - (P3) on page 22). But

\[
TRAD = TR[I \mid A] D_1 \\
\quad \\
\quad D_2
\]

Without loss of generality we may require that $D_1 = R^{-1}$. Then $T[RD_1 \mid RAD_2] = T[I \mid RAD_2]$. Therefore, the Reducibility Problem may be stated as follows:

"Given a matrix $\tilde{A} = [I \mid A]$, does there exist a nonsingular matrix $T$ and nonsingular diagonal matrices $R$ and $D$ such that $[T \mid TRAD]$ is a network matrix?"

-24-
3.3 Necessary Conditions for Reducibility

Several researchers have addressed the reducibility problem and necessary conditions on T, R, and D are known. It is clear that if T[RAD] is a network matrix, then T is itself a network matrix. It is shown in Bartholdi and Ratliff [8] that a nonsingular network matrix corresponds to a tree (i.e., a connected graph having one less arc than node). For example, the nonsingular matrix

\[
T = \begin{bmatrix}
  a & b & c & d \\
  -1 & & & \\
  1 & -1 & -1 & \\
  & 1 & & \\
  & & 1 &
\end{bmatrix}
\]

\text{row 1} \quad \text{row 2} \quad \text{row 3} \quad \text{row 4}

corresponds to the tree

```
1 -- a -- 2
  |     |   |
  b     c   \\
2     |

4 -- b --> 2
    |   |
    c   \\
3     |

4 -- c -- 2
    |   |
    d   \\
5
```

where node (row 5) does not appear in T. Therefore, T[RAD] is a network matrix only if T is a tree.

A cut-set for a connected graph G is a set of edges whose removal results in a disconnected graph and is minimal with respect to this property. For example in the graph

```
1 -- a -- 2
  |     |   |
  b     c   \\
2     |

4 -- b --> 2
    |   |
    c   \\
3

4 -- c -- 2
    |   |
    d   \\
5
```

-25-
\{a, b\}, \{a, c, e\}, and \{a, c, d\} are all cut-sets. There is a dual relationship between the spanning tree of a graph and a cut-set. Recall that a spanning tree is a minimal set of edges which connects all vertices of a graph, whereas a cut-set is a minimal set of links which disconnects some vertices from others. From this observation it is obvious that any spanning tree must have at least one link in common with every cut-set. The set of fundamental cut-sets associated with a spanning tree having \( n \) arcs is composed of the \( n \) cut-sets each having one of the \( n \) edges from the tree. For example, for the graph

\[
\begin{array}{c}
1 \\
| \\
| a \\
| \\
| \\
2 \\
| \\
| c \\
| \\
| b \\
| \\
3 \\
| \\
| e \\
| \\
| \\
4 \\
\end{array}
\]

with spanning tree

\[
\begin{array}{c}
1 \\
| \\
| a \\
| \\
| \\
2 \\
| \\
| c \\
| \\
| b \\
| \\
3 \\
| \\
| e \\
| \\
| \\
4 \\
\end{array}
\]
the fundamental cut-sets are \{a, b\}, \{d, b, c\}, and \{e, b, c\}. The corresponding cut-set matrix is given by

\[
\begin{bmatrix}
  a & d & e & b & c \\
  1 & 0 & 0 & 1 & 0 \\
  d & 0 & 1 & 0 & 1 \\
  e & 0 & 0 & 1 & 1 \\
\end{bmatrix}
\]

where the rows correspond to cut-sets and the columns correspond to edges. The following result gives a relationship between a cut-set matrix and the corresponding network matrix.

**Proposition 1.**

Let \(T\) be a nonsingular network matrix corresponding to a tree. Then \(T[I \mid E]\) is a network matrix only if \([I \mid E]\) mod 2 is a fundamental cut-set matrix corresponding to \(T\).

We now state another necessary condition for reducibility.

**Proposition 2.**

Let \(T\) be a nonsingular network matrix corresponding to a tree. Then \(T[I \mid R_1AD_1]\) is a network matrix only if the nonzero elements of \(R_1AD_1\) are \(\pm 1\).

The above two propositions provide the basis for a transformation algorithm given below.

**TRANSFORMATION ALGORITHM**

0. Begin with the constraint matrix \([I \mid A]\).

1. Does there exist nonsingular diagonal matrices \(R_1\) and \(D_1\) such that the nonzero elements of \(R_1AD_1\) are \(\pm 1\)?

   No - Then \([I \mid A]\) is not transformable by Proposition 2.

   Yes - Continue with step 2.
2. Is $[I_1; R_1, AD_1] \mod 2$ a fundamental cut-set matrix for some graph, say $G$?
   No - Then $[I_1; A]$ is not transformable by Proposition 1.
   Yes - Continue with step 3.

3. Direct the arcs of $G$ arbitrarily and define the corresponding node-arc incidence matrix by $N_1$. Let $N_2$ be formed from $N_1$ by omitting one row. Let $T$ correspond to the columns of $N_2$ associated with some spanning tree of $G$. Partition $N$ as $[T; N_3]$. Convert $N_2$ to standard form by premultiplying by $T^{-1}$. This gives $[I_1; T^{-1}N_3]$.

4. Do there exist diagonal nonsingular matrices $R_2$ and $D_2$ having nonzero elements $\pm 1$ such that $R_2[I_1; R_1, AD_1] \begin{bmatrix} R_2^{-1} & \cdot \\ & \\
& D_2 \end{bmatrix} = [I_1; T^{-1}N_3]$?
   Yes - Then $[I_1; A]$ is transformable to a network matrix using $R = R_2R_1$, $D = D_1D_2$ and $T$. That is, $TR_2R_1[I_1; A]$
   \[
   \begin{bmatrix}
   R_2^{-1} & \\
   & \\
   & D_2
   \end{bmatrix}
   \begin{bmatrix}
   R_1^{-1} & \\
   & \\
   & D_1
   \end{bmatrix}
   \]
   is a network matrix.
   No - Transformation algorithm is not successful.

3.4 Matroid Theory

The question of when a matrix, $[I_1; E]$ is the cut-set matrix for some graph has been addressed by Tutte [54] in the mathematical framework known as matroid theory. A matroid (Welsh [57]) is a mathematical structure consisting of a finite set $E$ and a finite set $C$ of nonempty subsets of $E$ such that two properties hold.

(M1) If $X \neq Y \subseteq C$, then $X \not\subseteq Y$.

(M2) If $X$, $Y$ are distinct members of $C$ and $a \in X \cap Y \neq \emptyset$ there exists $Z \subseteq C$ such that $Z \subseteq (X \cup Y) - \{a\}$.  

-28-
For example $E = \{1, 2, 3, 4, 5\}$ and $C = \{1, 4\}, \{2, 4, 5\}, \{3\}, \{1, 2, 5\}$ is a matroid.

An $m \times n$ matrix $R$ is called a binary matrix if the nonzero elements of $R$ are ones. Let $F$ denote the set of binary $m$-vectors generated by all modulo 2 row sums of the binary matrix $R$. That is, $F = \{r = \alpha R \text{ and } \alpha 	ext{ is integer for all } i\}$. The elements of $F$ are called chains by Tutte [55]. Define the support of a chain, $f \in F$, denoted by $|f|$ as follows:

$$|f| = \{i : f_i \neq 0\}.$$  

A chain, $f \neq 0$, of $F$ is said to be elementary if there is no chain of $F$ which is a proper subset of $f$. Letting $E = \{1, \ldots, n\}$ and $C$ equal the set of elementary supports, we have what Tutte [55] calls a binary matroid, $M(R)$. For example, let

$$R = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}.$$  

Then the set of binary vectors generated by the mod 2 sum of rows of $R$ is given by

| Number | $a$         | $f$            | $|f|$  |
|--------|-------------|----------------|-------|
| 1      | $[1 \ 0 \ 0]$ | $[1 \ 0 \ 0 \ 1 \ 0]$ | $\{1, 4\}$ |
| 2      | $[0 \ 1 \ 0]$ | $[0 \ 1 \ 0 \ 1 \ 1]$ | $\{2, 4, 5\}$ |
| 3      | $[0 \ 0 \ 1]$ | $[0 \ 0 \ 1 \ 0 \ 0]$ | $\{3\}$ |
| 4      | $[1 \ 1 \ 0]$ | $[1 \ 1 \ 0 \ 0 \ 1]$ | $\{1, 2, 5\}$ |
| 5      | $[1 \ 0 \ 1]$ | $[1 \ 0 \ 1 \ 1 \ 0]$ | $\{1, 3, 4\}$ |
| 6      | $[0 \ 1 \ 1]$ | $[0 \ 1 \ 1 \ 1 \ 1]$ | $\{2, 3, 4, 5\}$ |
| 7      | $[1 \ 1 \ 1]$ | $[1 \ 1 \ 1 \ 0 \ 1]$ | $\{1, 2, 3, 5\}$ |

The elementary supports are $\{1, 4\}$, $\{2, 4, 5\}$, $\{3\}$, and $\{1, 2, 5\}$, and
the matroid is given by \( E = \{1, 2, 3, 4, 5\} \) and \( C = \{(1, 4), (2, 4, 5), (3), (1, 2, 5)\} \).

Another procedure for generating a binary matroid is to consider the cut sets of a given finite graph having \( n \) edges. Letting \( E = \{1, \ldots, n\} \) and \( C = \) the set of cut sets for \( G \), we also have a binary matroid.

For example, suppose \( G \) is

![Graph Image]

Then \( E = \{a, b, c, d, e\} \) and \( C = \{\{a, d\}, \{b, d, e\}, \{c\}, \{a, b, e\}\} \) forms a binary matroid. A matroid formed in this way is also called the bond matroid of the graph \( G \). A binary matroid is said to be graphic if it can be represented as the bond matroid of some graph. All graphic matroids are binary but the converse is false. The question of whether a given binary matroid is graphic is equivalent to the question, when is a binary matrix, \( [I | E] \), the cut-set matrix some graph? Tutte [54] provided an algorithm to determine when a binary matroid is graphic and Bixby and Cunningham [13] described a precise procedure based on Tutte's work to generate the corresponding graph. Heller [34, 35], and Iri [37, 38], have also discussed the reducibility problem but no one has ever experimentally tested any of these procedures on a set of large real world linear programs.

### 3.5 Practical Significance

We conjecture that at least 90% of the real world linear programs are not reducible to minimal cost network flow problems. Therefore,
from a strictly practical point of view, the rich theory developed to
date can not be applied directly to help solve a large class of real
world linear programs. However, modification of these results to ob-
tain networks with side constraints could be extremely valuable.

3.6 A Heuristic Algorithm

The reducibility algorithms of Bixby-Cunningham [13] and Iri [38]
either find the matrices $T$, $R$, and $D$ required for the transformation
or conclude that no such matrices exist. We have modified these ideas
so that they can be used to produce an embedded network. This work
could drastically change the way we view linear programs and could
substantially improve the state-of-the-art software for solving linear
programs.

Consider the following linear program which has been called the
network problem with side constraints,

$$\begin{align*}
\text{min} & \quad cx + dz \\
\text{s.t.} & \quad Ax = r \\
& \quad Sx + Pz = b \\
& \quad 0 < x < u \\
& \quad 0 < z < r
\end{align*}$$

where

- $c$ is $1 \times n_1$,
- $d$ is $1 \times n_2$,
- $r$ is $m_1 \times 1$,
- $b$ is $m_2 \times 1$,
- $u$ is $n_1 \times 1$,
\( I \) is \( n_2 \times 1 \),

\( A \) is \( m_1 \times n_1 \),

\( S \) is \( m_2 \times n_1 \),

\( P \) is \( m_2 \times n_2 \), and

\( A \) is a network matrix. If \( m_1/(m_1 + m_2) \geq .5 \), then we believe that a special algorithm applied to (14) - (18), such as the Simplex Son [27], will be substantially superior to the standard primal simplex method applied to this problem.

We shall say that an \( m \times n \) matrix \([ I \mid C ] \) is \( p \)-reducible if there exists a nonsingular \( p \times p \) matrix \( T \), and there exists nonsingular diagonal \( p \times p \) matrices \( R \) and \( D \), such that for some \( p \) rows of \([ I \mid C ] \), say \([ I \mid C_1 ] \), \( TR[ I \mid C_1 ] \) is a network matrix. That is, the matrix \([ I \mid C ] \) can be transformed as follows:

\[
\begin{bmatrix}
T & R_1 \\
0 & I
\end{bmatrix}
\begin{bmatrix}
R_1 & I \\
0 & I
\end{bmatrix}
\begin{bmatrix}
I & I \\
I & 0
\end{bmatrix}
\begin{bmatrix}
C_1 \\
C_2 \\
C_3
\end{bmatrix} =
\begin{bmatrix}
R_1^{-1} & I \\
-I & D \\
-T & I
\end{bmatrix}
\begin{bmatrix}
I \\
0 \\
0
\end{bmatrix}
\begin{bmatrix}
T & TR_2C_1D \\
0 & I \\
C_2D & C_3
\end{bmatrix}.
\]

The matrix (19) is, of course, the constraint matrix for a \( p \)-node network with side constraints.

Theoretically, the problem of partial network transformation can be described as follows:

"Given any matrix \( \lambda = [ I \mid A ] \), find the largest \( p \) such that \( \lambda \) is \( p \)-reducible."
The above described problem is NP-complete and we believe that there is little hope of finding an exact solution which could be used to enhance linear programming software. However, a good heuristic based upon the rich theory available could prove to be extremely useful.

We now present a heuristic algorithm which we believe holds the best hope for development of an automatic procedure for converting linear programs to networks with side constraints.

0. Initialization

Begin with the linear system
\[ A_1^1 x = b_1^1 \] (20)

1. Scale To \( \pm 1 \)'s

Using only row and column scaling, let
\[ A_2^2 x = b_2^2 \] (21)
denote a subset of rows of (20) which have been scaled to the elements 0, \( \pm 1 \).

2. Apply Brown-Wright Heuristic

Let \[ A_3^3 x = b_3^3 \] (22)
denote the rows of (21) which correspond to a network matrix as obtained by the Brown-Wright heuristic.

3. Transformation

Let
\[ A_4^4 x = b_4^4 \] (23)
denote the rows of (21) not appearing in (22). Apply a modification of the Bixby-Cunningham algorithm to attempt to build a tree which transforms part of (23) to a network matrix.
The above algorithm will be fast; however, no results concerning the size of the network generated are available. No computational experience with this algorithm is available at this time.
REFERENCES


-37-


