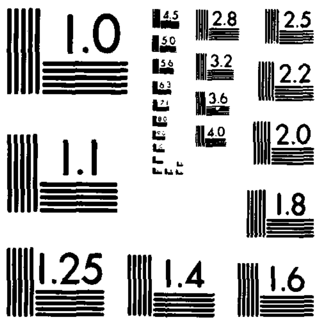

1.0
4.5
5.0
5.6
6.3
7.1
8.0
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

12 LEVEL II

CENTER FOR CYBERNETIC STUDIES

The University of Texas
Austin, Texas 78712

14 CCS - 364

6 A MULTI-PERIOD PRODUCTION, DISTRIBUTION,
AND INVENTORY PLANNING MODEL

by

10 Darwin/Klingman*
John/Mote**

12 44

11 Dec[redacted]979

* Professor of Operations Research and Computer Sciences, The University of Texas at Austin, Department of General Business, BEB 608, Austin, TX 78712.

** Systems Analyst, Analysis, Research, & Computation, Inc., P.O. Box 4067, Austin, TX 78765.

15
This research was supported in part by the Office of Naval Research contract no. N00014-78-C-0222. Reproduction in whole or in part is permitted for any purpose of the United States Government.

CENTER FOR CYBERNETIC STUDIES
A. Charnes, Director
BEB 203E
The University of Texas at Austin
Austin, Texas 78712
(512) 471-1821

406197

1. INTRODUCTION

Increasingly, today's manufacturing companies--both large and small--are finding it necessary to adopt computer-based planning systems to monitor their production, distribution, and inventory (PDI) functions [9]. This paper describes an optimization model that captures the multi-period aspect of many PDI planning systems. This model is based on the well-known pure network flow (transshipment) problem [18]. The model can be used to optimally select:

(1) production levels at multiple plant locations,

(2) (primary) distribution patterns between plants and storage facilities,

(3) inventory levels at multiple storage sites,

(4) (secondary) distribution patterns between storage facilities and customer demand points,

(5) customer satisfaction levels.

The model determines the optimal solution for each of these factors in each time period. By simultaneously considering all time periods (e.g., months) within the span of the planning horizon (e.g., fiscal year) the model is able to globally optimize the solution. That is, the model can tailor production and inventory schedules to account for fluctuating demands and prices over the planning horizon.

Since this PDI planning model is based on the pure network flow problem, it can be solved using any of the numerous capacitated transshipment computer codes such as PNET [1, 10, 11], GNET [6], RNET [13], or SUPERK [3]. However, in terms of computer memory requirements, each of these codes suffers considerably when the number of time periods is moderately large. This is due to the fact that multi-period planning models generally require a separate "copy" of the basic PDI network for each time period.

A specialized implementation of the primal simplex network code PNET has been developed that overcomes this computer memory handicap. This implementation, PNET-MP, is unique in that it only stores a single copy of the basic PDI network, thus minimizing the impact of a planning horizon with multiple time periods. PNET-MP generates the complete multi-period network as it is needed. Many of the efficient list structures that have been incorporated in the other primal simplex codes, such as PNET, GNET, and RNET, were adopted in this multi-period code.

PNET-MP was developed solely to reduce the computer memory requirements for this important class of planning problems. A degradation in solution speed, relative to PNET, was anticipated since extra data manipulations must be carried out in order to generate the multiple copies of the PDI network from the single copy that is actually stored. However, as will be illustrated in Section 6, PNET-MP is actually more efficient than its general-purpose parent code for this special class of planning problems. This is due to the modifications of the pivot strategy that were mandated by the code's restricted access to the complete multi-period network.

2. PROBLEM DESCRIPTION

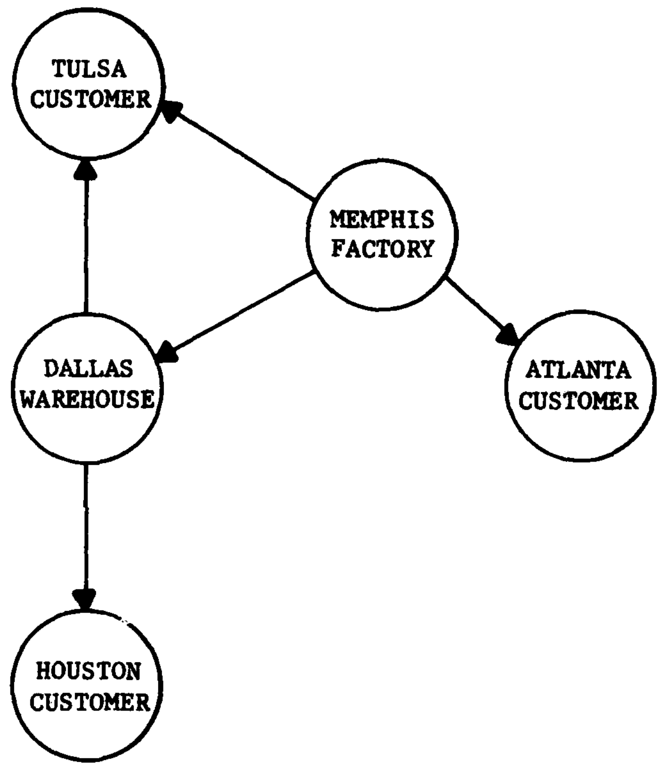
In this section a general statement of the specific problem class under consideration is presented. This type of problem arises in many production, distribution, and inventory planning systems. In general, the problem is to determine a minimum cost routing of units of a homogenous commodity through a distribution network over a (finite) multi-period planning horizon. The specific units being routed may be oil tankers, freight cars, aircraft, or military personnel. It must be assumed that the units are sufficiently similar to be treated within a single commodity framework. For instance, the units may actually be forty foot refrigerated box cars or second lieutenants with infantry skills.

Any distribution network consists of a set of nodes N and a set of arcs A (see Figure 1). Each node $i \in N$ of a distribution network represents a specific geographic location such as a production facility, a warehouse, a customer, or a military location. Each arc $k \in A$ represents an allowable distribution link between an ordered pair of nodes (i_k, j_k). That is, arc k represents a link from node $i_k \in N$ to node $j_k \in N$. Arcs can correspond to shipping channels, railroad tracks, flight path corridors, or personnel reassignments.

Each node of the distribution network has a schedule of

FIGURE 1

SIMPLE DISTRIBUTION NETWORK



supplies and demands for units at specific points in time over the planning horizon. Some or all of the nodes in the network can be used to store units until they are needed. These storage nodes have both a marginal storage cost and a capacity limitation associated with them. For instance, in Figure 1, the node representing the Dallas warehouse could have a storage cost of $10 per unit per period and a storage capacity of 5 units per period. Each arc of the network has a marginal distribution cost as well as a required traversal time. All arcs are assumed to be uncapacitated. In Figure 1, the arc from the Dallas warehouse node to the Houston customer node might have a marginal cost of $50 per unit and a traversal time of ten hours.

The only fundamental difference between this problem and the standard pure network problem is that the arcs have a delay or traversal time associated with them. This places a restriction on the movement of units over time. This restriction is not captured by the standard pure network model.

In order to simplify the problem analysis, the planning horizon will be partitioned into T equal length time periods. Depending upon the characteristics of the specific problem being solved, these periods may be hourly, daily, or even weekly.

The supplies and demands for units at each node in the network are assumed to take place at the end of the time periods. For instance, two empty freight cars could become available Tuesday morning in Chicago (supply), and one empty freight car could be required by Friday morning in Houston (demand). From period to period, the total number of available units may fluctuate as they are being added to and removed from the distribution system.

In order to simplify the problem analysis, it will be assumed that all units must arrive at a node by the end of each period. That is, a unit cannot be left stranded between nodes at the end of a period. In many cases this is a realistic assumption. In those cases where this assumption is not reasonable, long arcs can be broken up by introducing artificial nodes.

Many real-world distribution systems cover large geographic regions. This places severe limitations on the ability to meet the demand for units. For example, it may not be possible to use the empty freight cars in Chicago to meet the demand in Houston even though they are available three days in advance. This model seeks to capture this restriction on the length of allowable shipping routes through a distribution system.

Two different formulations of this problem are presented in the following sections. The first is an integer linear programming formulation of a multicommodity network model. The second is a capacitated transshipment formulation of a single commodity network model. Technically, both formulations are equivalent. However, in terms of computational feasibility, the second formulation is overwhemingly superior.

3. MULTICOMMODITY FORMULATION

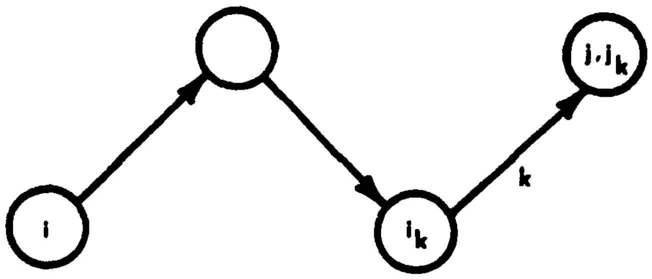
An integer linear programming formulation of the problem is presented in this section. The formulation is essentially an integer multicommodity model. The complexity of this approach is a direct result of the stipulation that the total traversal time of each shipping route cannot exceed the length of the time period.

This approach is a multicommodity model because it is necessary to keep track of the origin and destination of each unit as it is moved through the network. During any given period, the units that are being shipped across an arc may have originated (at the beginning of the period) at any of the nodes in the network, and may be destined (at the end of the period) for any of the other nodes. However, due to the traversal time restriction, many combinations of origin and destination nodes are not feasible. For instance, it may be physically impossible to move an empty freight car from Chicago to Houston in a single day.

In order to differentiate between the various units traversing an arc, an integer flow variable x_{kij}^{t} must be introduced. The index k denotes the arc number, i is the original origin node, j is the final destination node, and t is the time period. That is, i is the node where the unit was located at the start of period t, and j is the node where the unit will be located at the end of period t. It should be noted that nodes i and j are not necessarily the endpoints (i_k and j_k) of arc k. (See Figure 2.) That is, a unit may be shipped from node i to node j via many arcs during the same

FIGURE 2

SHIPPING ROUTE



period provided that the total traversal time of the shipping route does not exceed the length of the period.

The storage capability of a node can be modeled as a loop arc from the node to itself. Then the number of units stored at node v during period t is given by x_{kvv}^t where k is the index of the loop arc.

Each arc has a marginal cost c_k associated with it. For the distribution arcs ($i_k \neq j_k$) this is the marginal shipping cost, and for the loop arcs ($i_k = j_k$) this is the marginal storage cost. For some problems, these marginal costs may vary over the planning horizon. In these cases it is necessary to introduce a time index, c_k^t.

Each loop arc has an upper bound corresponding to the storage capacity of the associated node. The upper bound for x_{kvv}^t is given by S_v, or S_v^t if the storage capacity varies over time.

Each distribution arc has a traversal time ℓ_k associated with it. It is convenient to scale these times relative to the length of a period. For instance, if a daily period is used and arc k requires six hours to traverse, then $\ell_k = .25$.

Each node has a supply or demand for units at the end of each period. Let b_v^t be node v's net demand at the end of period t.

It is notationally convenient to introduce two arc index sets for each node. The first set

$$A(v) = \{k \mid i_k = v, j_k \neq v\}$$

is the set of distribution arcs that originate at node v, and the second set

$$B(v) = \{k \mid i_k \neq v,\ j_k = v\}$$

is the set of distribution arcs that terminate at node v. These sets do not include the loop arcs.

In addition to the integer flow variable x^t_{kij}, a binary variable L^t_{kij} is needed in order to capture the traversal time restriction on feasible shipping routes. This variable is used to identify the arcs that are used to ship units from node i to node j during period t.

The T time periods are numbered from 1 to T, and the T + 1 points in time at the ends of periods are numbered from 0 to T. That is, the point in time at the start of period t is numbered t - 1, and the point in time at the end of period t is numbered t.

The set of network nodes is denoted by N. The set of distribution arcs is denoted by A.

The objective of this problem is to minimize the total cost of distribution and storage over the T time periods. This objective can be stated as

$$\text{Minimize} \quad \sum_{t=1}^{T} \sum_{k \in A} c_k \sum_{i \in N} \sum_{j \in N-i} x^t_{kij} + \sum_{t=1}^{T} \sum_{i \in N} c_k x^t_{kii}$$

where the first term is the total distribution cost and the second term is the total storage cost. The quantity

$$\sum_{i \in N} \sum_{j \in N-i} x_{kij}^{t}$$

is the total number of units shipped across arc k during period t.

There are five basic types of constraints for this integer linear programming formulation of the problem. These constraint types are:

I. Conservation of flow originating or terminating at node v at time t,

II. Conservation of flow passing through node v during period t,

III. Flow variable/binary variable relationships,

IV. Shipping route total traversal time limitations, and

V. Node storage capacities.

Each constraint type will be presented separately.

Type I Constraints

These conservation of flow constraints affect only the units that start a period or end a period at node v. These constraints guarantee that the number of units that terminate at node v at the end of period t equals the number of units that originate at node v at the start of period t + 1 plus the net demand for units at node v at the end of period t. There is one type I constraint for each combination of network node and point in time. The general form of the constraint is

$$\begin{pmatrix}\text{Number of units}\\ \text{destined for}\\ \text{node v at the}\\ \text{end of period t}\end{pmatrix} = \begin{pmatrix}\text{Net demand for}\\ \text{units at node v}\\ \text{at the end of}\\ \text{period t}\end{pmatrix} + \begin{pmatrix}\text{Number of units}\\ \text{originating from}\\ \text{node v at the start}\\ \text{of period t + 1}\end{pmatrix}$$

For t = 0, the initial conditions are:

$$0 = b_v^o + \sum_{k\varepsilon A(v)} \sum_{j\varepsilon N} x_{kvj}^1$$

For t = 1, 2, ..., T - 1, the intermediate conditions are:

$$\sum_{k\varepsilon B(v)} \sum_{i\varepsilon N} x_{kiv}^t = b_v^t + \sum_{k\varepsilon A(v)} \sum_{j\varepsilon N} x_{kvj}^{t+1}$$

For t = T, the terminal conditions are:

$$\sum_{k\varepsilon B(v)} \sum_{i\varepsilon N} x_{kiv}^T = b_v^T$$

Type II Constraints

These constraints maintain the conservation of flow of all units that are not covered by the Type I constraint. That is, the units that simply pass through node v, as they are moved from node i to node j, are handled by the Type II constraints. There is one Type II constraint for each combination of network node and time period. The general form of the constraint is

$$\begin{pmatrix}\text{Number of units entering node}\\ \text{v, but not destined for it,}\\ \text{during period t}\end{pmatrix} = \begin{pmatrix}\text{Number of units leaving node}\\ \text{v, but not originating from}\\ \text{it, during period t}\end{pmatrix}$$

For t = 1, 2, ..., T, this constraint takes the form:

$$\sum_{k\varepsilon B(v)} \sum_{i\varepsilon N-v} \sum_{j\varepsilon N-i-v} x_{kij}^t = \sum_{k\varepsilon A(v)} \sum_{i\varepsilon N-v} \sum_{j\varepsilon N-i-v} x_{kij}^t$$

Type III Constraints

These constraints enforce the proper relationship between the flow variables x^t_{kij} and the binary variables L^t_{kij}. Specifically, L^t_{kij} must be equal to one unless x^t_{kij} is equal to zero. There is one Type III constraint for each combination of distribution arc, origin node, destination node, and time period. The constraint takes the form

$$x^t_{kij} \leq ML^t_{kij}$$

where M is an arbitrarily large number.

Type IV Constraints

These constraints restrict the total traversal time of each shipping route. The validity of these constraints relies on the following observation. Namely, since all distribution arcs are uncapacitated, all units being shipped from node i to node j during period t can travel along a single (least cost) route. This means that the total traversal time of the shipping route between nodes i and j during period t can be measured by

$$\sum_{k \in A} \ell_k L^t_{kij}$$

Therefore, the restriction that the total traversal time of the shipping route between nodes i and j is less than the length of the period can be enforced with the following constraint

$$\sum_{k \in A} \ell_k L^t_{kij} \leq 1$$

There is one Type IV constraint for each combination of origin node, destination node, and time period.

Type V Constraints

The storage capacity at each node can be handled as a simple upper bound on the flow on each loop arc. This constraint has the form

$$x_{kvv}^{t} \leq S_v$$

There is one Type V constraint for each combination of network node and time period.

This formulation, although mathematically correct, is ovbiously computationally infeasible. A simple five period problem with an underlying 100 node and 500 arc distribution network produces an integer linear programming problem with nearly 25 million constraints and 50 million variables. As stated earlier, the complexity of this approach is a direct result of the restriction on the total traversal time of each shipping route.

4. CAPACITATED TRANSSHIPMENT FORMULATION

In this section, the problem introduced in Section 2 is formulated as a multi-period capacitated transshipment problem. This single commodity formulation removes the majority of the complexity of the multi-commodity formulation of Section 3. Notably, the integrality requirements become a natural consequence of the unimodularity property of the network constraints [15].

This transshipment formulation of the probelm revolves around a simple transformation of the underlying distribution network. The transformed network contains the same nodes as the original network, but contains a different set of arcs. Roughly speaking, the transformed network has an arc corresponding to each feasible shipping route in the original distribution network.

The arcs of the transformed network can be generated by solving a set of simple shortest path problems. The first shortest path problem uses the traversal time ℓ_k as the length of each arc in the original network. The solution to this shortest path problem provides the minimum traversal time of the shipping route between each pair of nodes. Let T_{ij} be the minimum traversal time from node i to node j. The transformed network has a directed arc from node i to node j if and only if $T_{ij} \leqq 1$. That is, the transformed network has an arc between each pair of nodes that are less than one period apart in terms of traversal time.

The marginal cost of an arc in the transformed network is equal to the sum of the marginal costs of the arcs in the corresponding shipping route in the original network. If there are multiple shipping routes between a pair of nodes in the original network, then a shortest path problem with c_k as the length of arc k can be used to determine the marginal cost of the best shipping route. Caution must be exercised to make sure that the minimum cost shipping route has a total traversal time no longer than a period. For instance, if three shipping routes between nodes 1 and 2 in the original network have total traversal times of .4, .9, and 1.6, and have total marginal costs of 20, 30, and 15, then the transformed network would have a directed arc from node 1 to node 2 with a marginal cost of 20. Note that the best marginal cost (15) is associated with a shipping route that requires more than one time period (1.6). A transformation of this type was proposed by Wagner [19].

Since all of the distribution arcs in the original network are uncapacitated, all shipping routes in the transformed network are also uncapacitated. Since the arcs in the transformed network actually correspond to feasible shipping routes in the original network, all arcs in the transformed network are uncapacitated.

The transformation is completed by replacing the restriction that the total traversal time of each shipping route is less than one period, by the restriction that a unit of flow can traverse at most one arc per period in the transformed problem.

As before, the storage capability at node v is handled by adding a loop arc from node v to itself. The marginal cost of this loop arc is equal to the marginal storage cost at node v. This arc has an upper bound equal to the storage capacity of node v. This expanded transformed network is called the *basic network*.

Let N be the set of nodes, and A be the set of arcs in the basic network.

The number of units shipped across arc k during period t is denoted by x_k^t. The marginal cost of arc k is given by c_k (or c_k^t), and the upper bound on the flow on arc k is given by u_k (or u_k^t). So $u_k = S_v$ if arc k corresponds to the loop arc for node v. Otherwise, $u_k = \infty$.

The net demand for units at node v at the end of period t is denoted by b_v^t.

As before, it is convenient to introduce two arc index sets. The first set

$$A(v) = \{k \mid i_k = v\}$$

is the set of arcs that originate at node v in the basic network, and the second set

$$B(v) = \{k \mid j_k = v\}$$

is the set of arcs that terminate at node v in the basic network. The loop arcs are included in these sets.

The objective function for this formulation of the problem is given by

$$\text{Minimize} \quad \sum_{k=1}^{T} \sum_{k \in A} c_k x_k^t$$

This formulation has two basic constraint types:

A. Conservation of flow at node v at time t, and

B. Arc capacities.

These constraints are considered individually.

Type A Constraints

Since this formulation requires that the traversal of any arc of the basic network consumes a complete time period, no units will pass through an intermediate node during a period. This eliminates the need for a constraint analogous to the Type II constraint of Section 3. All conservation of flow is handled by the following version of the Type I constraint of Section 3. There is one Type A constraint for each combination of network node and point in time. The general form of the constraint is

$$\begin{pmatrix}\text{Number of units}\\ \text{entering node v}\\ \text{during period t}\end{pmatrix} = \begin{pmatrix}\text{Net demand for units}\\ \text{at node v at the end}\\ \text{of period t}\end{pmatrix} + \begin{pmatrix}\text{Number of units}\\ \text{leaving node v}\\ \text{during period}\\ \text{t + 1}\end{pmatrix}$$

For $t = 0$, the initial conditions are:

$$0 = b_v^0 + \sum_{k \varepsilon A(v)} x_k^1$$

For $t = 1, 2, \ldots, T - 1$, the intermediate conditions are:

$$\sum_{k \varepsilon B(v)} x_k^t = b_v^t + \sum_{k \varepsilon A(v)} x_k^{t+1}$$

For $t = T$, the terminal conditions are:

$$\sum_{k \varepsilon B(v)} x_k^T = b_v^T$$

Type B Constraints

The storage capability at the nodes can be handled as a simple upper bound constraint on the loop arcs. There is one Type

B constraint for each combination of network node and time period. The constraint has the form

$$x_k^t \leq u_k$$

The complete algebraic statement of the capacitated transshipment formulation of the problem is given as

$$\text{Minimize} \quad \sum_{t=1}^{T} \sum_{k \varepsilon A} c_k x_k^t$$

subject to:

$$\begin{aligned}
-\sum_{k \varepsilon A(v)} x_k^1 &= b_v^0 \quad \text{for all } v \ \varepsilon \ N \\
\sum_{k \varepsilon B(v)} x_k^t - \sum_{k \varepsilon A(v)} x_k^{t+1} &= b_v^t \quad \text{for all } v \ \varepsilon \ N \text{ and} \\
& \qquad\ t = 1, 2, \ldots, T-1 \\
\sum_{k \varepsilon B(v)} x_k^T &= b_v^T \quad \text{for all } v \ \varepsilon \ N \\
0 \leq x_k^t \leq u_k & \qquad\quad \text{for } k \ \varepsilon \ A \text{ and} \\
& \qquad\ t = 1, 2, \ldots, T
\end{aligned}$$

This problem is easily seen to be a capacitated pure transshipment problem if it is restated in matrix notation. In order to do this, two m x n matrices (F and T) will be introduced. Essentially, they represent the origin and destination nodes of the arcs of the m node and n arc basic network. The element in the vth row and kth column of the origin matrix F is defined by

$$F_{vk} = \begin{cases} -1 \text{ if } i_k = v \\ 0 \text{ otherwise} \end{cases}$$

and the element in the vth row and kth column of the destination matrix T is defined by

$$T_{vk} = \begin{cases} 1 \text{ if } j_k = v \\ 0 \text{ otherwise} \end{cases}$$

So each column of F and T has a single non-zero element.

Let X^t be the vector $(x_1^t, x_2^t, \ldots, x_n^t)^T$ of arc flow during period t. Let F_v and T_v denote the vth row of F and T, respectively. Then

$$F_v X^t = -\sum_{k\epsilon A(v)} x_k^t$$

and

$$T_v X^t = \sum_{k\epsilon B(v)} x_k^t$$

Let C be the vector $(c_1, c_2, \ldots, c_n)$, U be the vector $(u_1, u_2, \ldots, u_n)^T$, and b^t be the vector $(b_1^t, b_2^t, \ldots, b_m^t)^T$. Then the capacitated transshipment problem can be restated in matrix terms as

$$\text{Minimize} \quad CX^1 + CX^2 + CX^3 \ldots + CX^{T-1} + CX^T$$

$$\begin{array}{llllll} \text{subject to:} & FX^1 & & & & = b^0 \\ & TX^1 + FX^2 & & & & = b^1 \\ & & TX^2 + FX^3 & & & = b^2 \\ & & & \ddots & & \\ & & & & TX^{T-1} + FX^T & = b^{T-1} \\ & & & & TX^T & = b^T \end{array}$$

$$0 \leq X^t \leq U \text{ for } t = 1, 2, \ldots, T$$

Ignoring the simple upper bound restrictions on the flow variables, each column of the constraint matrix of this problem has two non-zero elements. Column $k + (t - 1)n$ is associated with variable x_k^t. This column has a -1 element in row $i_k + (t - 1)m$, and a $+1$ element in row $j_k + tm$. Therefore, this problem can be seen to be a capacitated pure transshipment problem with $M = (T + 1)m$ nodes and $N = Tn$ arcs. This problem will be referred to as the *master network problem*. The nodes and arcs of the master network problem are called *master nodes* and *master arcs*. The master network essentially consists of T copies of the original m node and n arc basic network. One master network node is generated for each combination of basic network node and point in time, and one master network arc is generated for each combination of basic network arc and time period.

Clearly this formulation of the original problem is superior to the integer multi-commodity formulation in Section 3. A direct comparison of the relative sizes of the two problem formulations is not particularly easy, since the capacitated transshipment formulation is based on a transformed network. The number of arcs in the transformed network, or basic network, is a function of the number of feasible shipping routes in the original distribution network. To simplify the analysis of the relative problem sizes, let θ denote the average number of arcs that originate at a node in the basic network. That is, θ is the average number of nodes that can be reached within one time period from a given node.

In Section 3, it was shown that a five period problem with a 100 node and 500 arc distribution system generated an integer linear programming problem with nearly 25 million constraints and 50 million variables. If $\theta = 10$, then this same problem can be formulated as a capacitated pure transshipment problem with a 600 node and 5,000 arc master network. Even if θ is as large as fifty, the master network only has 600 nodes and 25,000 arcs. Clearly, the single commodity formulation of the problem can be readily solved using any of the available capacitated transshipment computer codes.

A small example is given in the next section. In Section 6, a specialization of the efficient primal simplex network code PNET is presented. This specialized code PNET-MP can be used to solve much larger multi-period problems than can be solved by PNET or any of the other in-core network codes. This specialized code is extremely valuable when either T or θ is large.

5. EXAMPLE PROBLEM

A small three period example is considered in this section. The basic network, consisting of four nodes and ten arcs, is illustrated in Figure 3. Three of the nodes have loop arcs associated with them to capture their storage capability. The arc data for the basic network is summarized in Table 1, and the net demands for each node are presented in Table 2. Thirteen units are available in the distribution system during period one. Eight units are available during period two, and four units are available during period three.

FIGURE 3
BASIC NETWORK

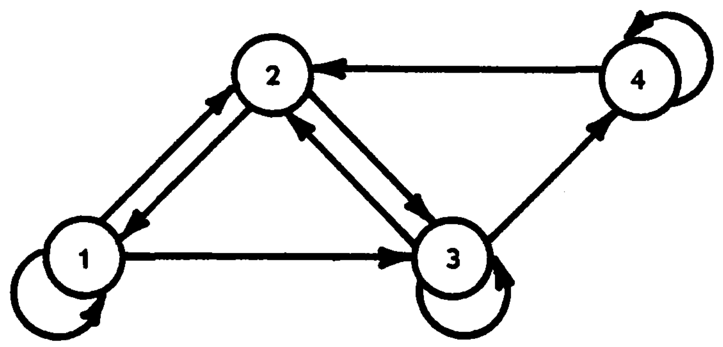


TABLE 1

ARC DATA

k	1	2	3	4	5	6	7	8	9	10
i_k	1	1	1	2	2	3	3	3	4	4
j_k	1	2	3	1	3	2	3	4	2	4
c_k	0	10	30	7	22	25	2	12	38	10
u_k	3	∞	∞	∞	∞	∞	4	∞	∞	8

TABLE 2

NET DEMANDS (b_v^t)

$v \backslash t$	1	2	3	4
1	-9	4	0	1
2	-1	-1	5	2
3	-3	0	0	1
4	0	2	-1	0

The matrix formulation of the capacitated transshipment problem is given in Figure 4, and the corresponding master network is illustrated in Figure 5. This figure highlights some of the important topological characteristics of the multi-period transshipment problem. The nodes of the master network are arranged in m parallel rows and T + 1 parallel columns. Row v of the master network corresponds to node v of the basic network. Column t of the master network corresponds to the t^{th} point in time, or the end of period t. Figure 5 illustrates that the master network is an acyclic graph. All master arcs are directed from a node in column t - 1 to a node in column t. The movement of units in the basic network during period t corresponds to the flows on the arcs directed from column t - 1 nodes to column t nodes. The pattern of arcs between columns t - 1 and t is repeated for each period. The marginal costs and upper bounds on these arcs are also identical from period to period.

Figure 6 shows a feasible solution to the master network problem, and Figure 7 shows the corresponding solution for the basic network for each period. This solution has an objective function value of 414.

FIGURE 4

CAPACITATED PURE TRANSSHIPMENT FORMULATION

Minimize

$$\left[\begin{array}{cccccccccc|cccccccccc|cccccccccc} 0 & 10 & 30 & 7 & 22 & 25 & 2 & 12 & 38 & 10 & 0 & 10 & 30 & 7 & 22 & 25 & 2 & 12 & 38 & 10 & 0 & 10 & 30 & 7 & 22 & 25 & 2 & 12 & 38 & 10 \end{array}\right] \begin{pmatrix} x^1 \\ x^2 \\ x^3 \end{pmatrix}$$

subject to:

$$\left[\begin{array}{cccccccccc|cccccccccc|cccccccccc}
-1&-1&-1&&&&&&& & &&&&&&&&& & &&&&&&&&& \\
&&&-1&-1&&&&& & &&&&&&&&& & &&&&&&&&& \\
&&&&&-1&-1&-1&& & &&&&&&&&& & &&&&&&&&& \\
&&&&&&&&-1&-1 & &&&&&&&&& & &&&&&&&&& \\
\hline
1&&&1&&&&&& & -1&-1&-1&&&&&&& & &&&&&&&&& \\
&1&&&&1&&&1& & &&&-1&-1&&&&& & &&&&&&&&& \\
&&1&&1&&1&&& & &&&&&-1&-1&-1&& & &&&&&&&&& \\
&&&&&&&1&&1 & &&&&&&&&-1&-1 & &&&&&&&&& \\
\hline
&&&&&&&&& & 1&&&1&&&&&& & -1&-1&-1&&&&&&& \\
&&&&&&&&& & &1&&&&1&&&1& & &&&-1&-1&&&&& \\
&&&&&&&&& & &&1&&1&&1&&& & &&&&&-1&-1&-1&& \\
&&&&&&&&& & &&&&&&&1&&1 & &&&&&&&&-1&-1 \\
\hline
&&&&&&&&& & &&&&&&&&& & 1&&&1&&&&&& \\
&&&&&&&&& & &&&&&&&&& & &1&&&&1&&&1& \\
&&&&&&&&& & &&&&&&&&& & &&1&&1&&1&&& \\
&&&&&&&&& & &&&&&&&&& & &&&&&&&1&&1
\end{array}\right] \begin{pmatrix} x^1 \\ x^2 \\ x^3 \end{pmatrix} = \left[\begin{array}{c} -9 \\ -1 \\ -3 \\ 0 \\ \hline 4 \\ -1 \\ 0 \\ 2 \\ \hline 0 \\ 5 \\ 0 \\ -1 \\ \hline 1 \\ 2 \\ 1 \\ 0 \end{array}\right]$$

$$\begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \leq \begin{pmatrix} x^1 \\ x^2 \\ x^3 \end{pmatrix} \leq \begin{pmatrix} U \\ U \\ U \end{pmatrix}$$

FIGURE 5

MASTER NETWORK

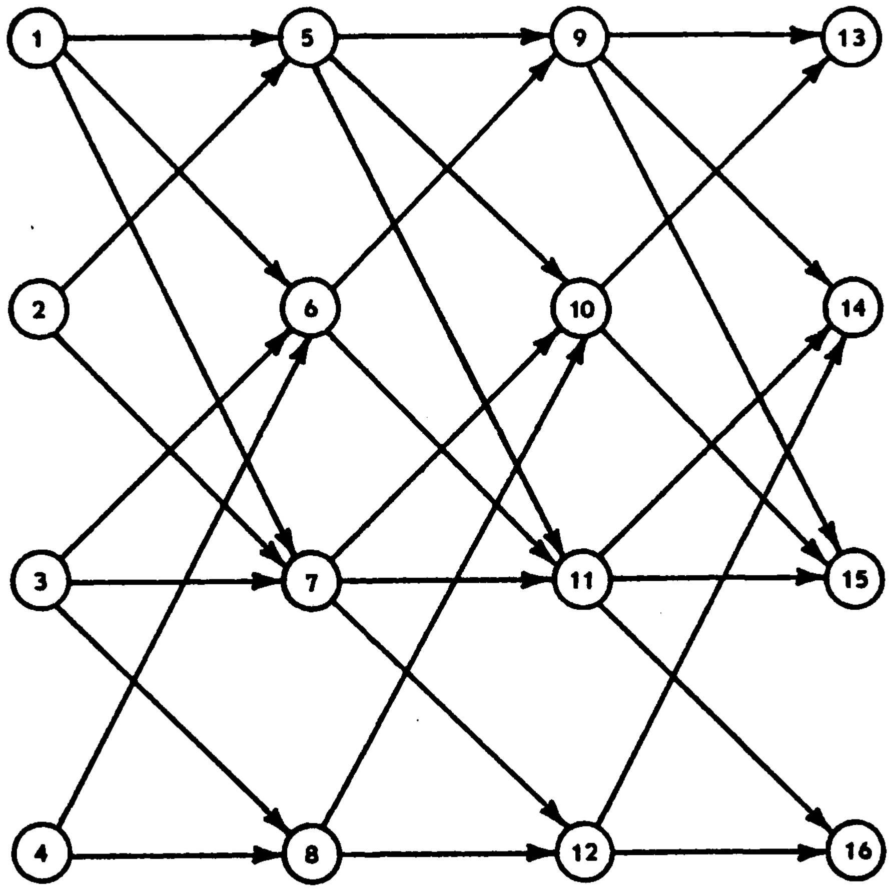


FIGURE 6

MASTER NETWORK FEASIBLE SOLUTION

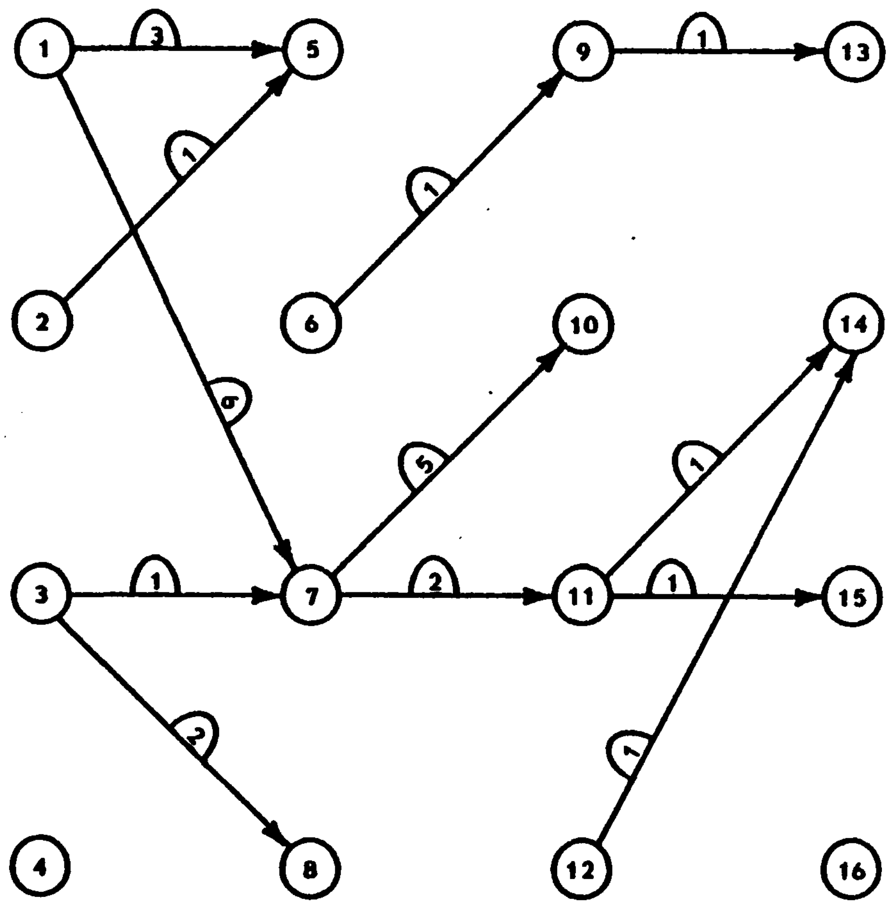
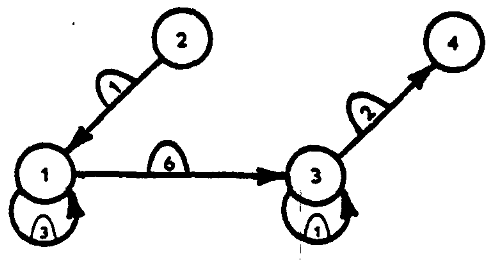


FIGURE 7

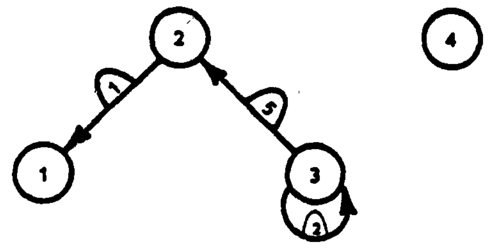
BASIC NETWORK FEASIBLE SOLUTIONS

PERIOD 1



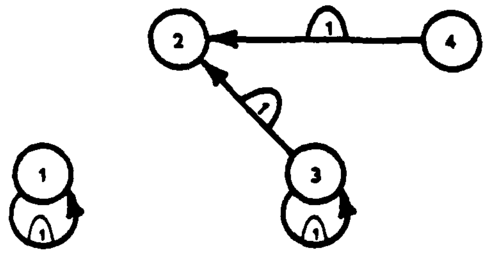
	UNITS	COST
SHIPPED	9	211
STORED	4	2
TOTAL	13	213

PERIOD 2



	UNITS	COST
SHIPPED	6	132
STORED	2	4
TOTAL	8	136

PERIOD 3



	UNITS	COST
SHIPPED	2	63
STORED	2	2
TOTAL	4	65

6. PNET-MP: A MULTI-PERIOD TRANSSHIPMENT CODE

A specialized version of the efficient primal simplex network code PNET [1] is presented in this section. This large-scale implementation uses the same principal list structures as PNET to store the tree representation of the basis matrix. Specifically, the M node and M - 1 arc triangular basis matrix is represented by six node functions.

The underlying structure of the basis tree is described by the predecessor node and predecessor arc functions. The thread and depth functions are used to improve the pivoting capabilities of the code. The primal basic solution and the associated dual solution are also maintained as node functions. Each of these node functions is implemented in PNET-MP as a node length array.

A basis tree corresponding to the feasible master network solution from Figure 6 is shown in Figure 8. Table 3 provides the six node functions that are used to describe the basis tree in Figure 8.

The principal difference between the general purpose code, PNET, and the specialized multi-period code, PNET-MP, involves the method of representing the original problem data. PNET uses three N length lists to store the arc data. That is, the destination node, marginal cost, and upper bound of every master arc is explicitly stored in the arc lists. This means that each basic network arc is duplicated T times in the arc lists. PNET-MP, on the other hand, uses two n length and one m length lists to store the arc data of

FIGURE 8

BASIS TREE ROOTED AT NODE 1

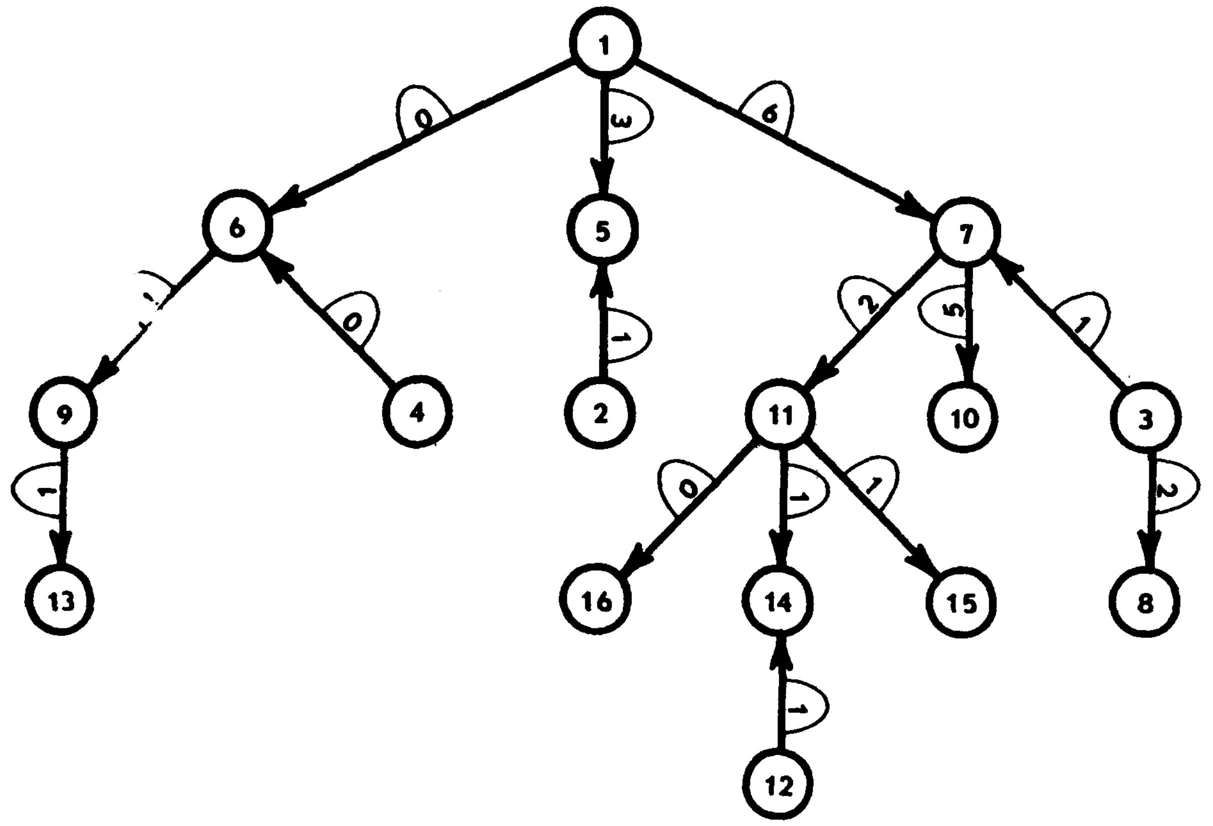


TABLE 3

NODE FUNCTIONS

I	PN(I)	PA(I)	THREAD(I)	DEPTH(I)	NETCAP(I)	POT(I)
1	0	0	6	0	0	0
2	5	4	7	2	∞ -1	-7
3	7	7	8	2	4 -1	28
4	6	9	5	2	∞ -0	-28
5	1	1	2	1	3	0
6	1	2	9	1	0	10
7	1	3	11	1	6	30
8	3	8	1	3	2	40
9	6	14	13	2	1	17
10	7	16	3	2	5	55
11	7	17	16	2	2	32
12	14	29	15	4	∞ -1	19
13	9	21	4	3	1	17
14	11	26	12	3	1	57
15	11	27	10	3	1	34
16	11	28	14	3	0	44

the basic network. The N master network arcs are implicitly stored by only explicitly storing the n basic network arcs.

Since only one copy of each basic network arc is physically stored, PNET-MP has the capability to solve extremely large multi-period network problems. For instance, PNET requires over two and a half times more core storage than PNET-MP to solve a five period problem with a 100 node and 500 arc basic network. For a ten period problem with a 100 node and 1000 arc basic network, PNET requires over six times more core storage than PNET-MP. As either the number of periods or the arc to node ratio increases, PNET-MP becomes more attractive than PNET in terms of total storage requirements.

As pointed out in Section 4, each column of the M x N constraint matrix corresponds to a master arc and each row corresponds to a master node. Master arc K, from master node I_K to master node J_K is related to arc k of the basic network for period t in the following manner:

$$K = k + (t - 1)n$$

$$I_K = i_k + (t - 1)m$$

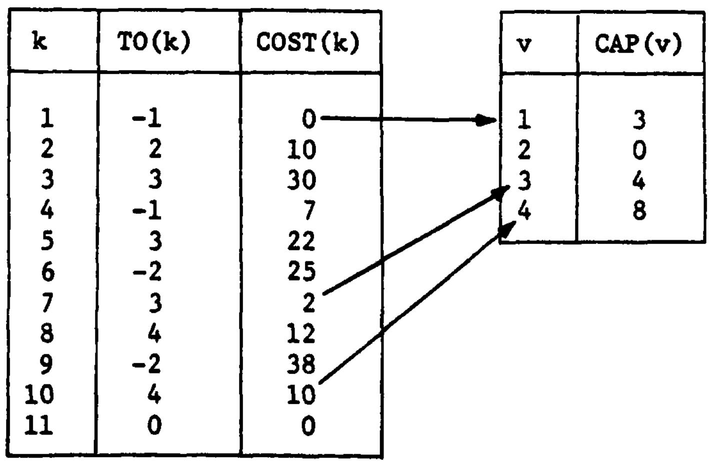
$$J_K = j_k + tm$$

PNET-MP uses this fundamental relationship between the master and basic arcs to generate the complete master network even though only the basic network is explicitly stored.

The arcs of the basic network are stored in two n length and one m length lists. The destination node of basic arc k is stored in TO(k). The arcs are stored according to common origin node. The destination node of the first arc out of each origin node is negated in the TO list. This is used to signal the start of the arc data for the next node. The marginal cost of basic arc k is stored as COST(k). The upper bound on the loop arc associated with basic node v is stored as CAP(v). This node length list is used to represent the finite arc capacities. That is, basic arc k has an infinite upper bound if $i_k \neq j_k$, and has a finite upper bound of $CAP(i_k)$ if $i_k = j_k$. Table 4 summarizes the arc data for the four node and ten arc basic network given in Section 5. Note that a trailer record (position 11) is used to signal the end of the arc data.

TABLE 4

ARC FUNCTIONS



k	TO(k)	COST(k)
1	-1	0
2	2	10
3	3	30
4	-1	7
5	3	22
6	-2	25
7	3	2
8	4	12
9	-2	38
10	4	10
11	0	0

v	CAP(v)
1	3
2	0
3	4
4	8

This particular data structure is well-suited for sequential processing of the arc data. Extensive computational testing has shown that sequential processing is highly effective for primal simplex network optimization codes [10, 11, 16, 17].

There are four fundamental operations that must be performed by a primal simplex network code. They are:

1. Determination of an initial solution,
2. Selection of an entering arc,
3. Selection of a leaving arc, and
4. Pivot, or change of basis.

The precise implementation in PNET-MP of each of these operations will be considered next.

An initial basis for the problem can be obtained by adding an artificial variable for each node constraint of the master network. The coefficient of the artificial variable in the I^{th} constraint is determined by the sign of the net demand for units at node I. If node I has a non-negative net demand, then the artificial variable has a +1 coefficient in constraint I. Otherwise, it has a -1 coefficient. Each artificial variable is assigned an infinite marginal cost. The expanded master network problem is given by:

$$\begin{array}{llllllllll}
\text{Minimize} & CX^1 + CX^2 + CX^3 \ldots + CX^{T-1} + CX^T + MY^0 + MY^1 + MY^2 \ldots + MY^{T-1} + MY^T \\
\text{subject to:} & FX^1 & + D^0Y^0 & & & & & = b^0 \\
& TX^1 + FX^2 & & + D^1Y^1 & & & & = b^1 \\
& \quad TX^2 + FX^3 & & & + D^2Y^2 & & & = b^2 \\
& \quad\quad \cdot & & & \cdot & & & \cdot \\
& \quad\quad\quad \cdot & & & \quad \cdot & & & \cdot \\
& \quad\quad\quad\quad \cdot & & & \quad\quad \cdot & & & \cdot \\
& \quad\quad\quad\quad TX^{T-1} + FX^T & & & & + D^{T-1}Y^{T-1} & & = b^{T-1} \\
& \quad\quad\quad\quad\quad\quad TX^T & & & & & + D^TY^T & = b^T \\
& & 0 \leq X^t \leq U \text{ for } t = 1, 2, \ldots, T \\
& & 0 \leq Y^t \quad\quad \text{for } t = 0, 1, \ldots, T
\end{array}$$

where M is an m dimensional infinite (Big M) vector, and D^t is a diagonal matrix whose v^{th} diagonal coefficient is given by

$$D^t_{vv} = \begin{cases} 1 \text{ if } b^t_v \geq 0 \\ -1 \text{ if } b^t_v < 0 \end{cases}$$

This expanded problem has full row rank since each D^t has rank m. An equivalent expanded pure network problem can be obtained by creating a redundant constraint equal to the negative of the sum of the M network constraints. This new constraint has the form

$$d^0Y^0 + d^1Y^1 + d^2Y^2 \ldots + d^{T-1}Y^{T-1} + d^TY^T = 0$$

where d^t is an m dimensional vector whose v^{th} coefficient is the negative of the v^{th} diagonal coefficient of D^t. This redundant constraint can be thought of as an artificial node in the expanded master network. The M artificial variables correspond to arcs directed into and out of the artificial node. Since $X^t = 0$

and $Y^t = D^t b^t$ is a feasible solution to the expanded network problem, the M artificial arcs can be used as an initial basis tree. For convenience, the artificial node, M + 1, is used as the root of the basis tree. Figure 9 illustrates this starting basis tree for the small example in Section 5.

PNET-MP uses this artificial arc basis tree as the initial feasible solution. The six node functions are extremely easy to initialize for this starting basis.

A modification of the outward node most negative pivot rule [11] is used by PNET-MP to select the arcs to enter the basis. The modification is motivated by the fact that each arc of the basic network generates T master arcs. Instead of selecting the most pivot eligible arc out of a single master node, PNET-MP selects the best arc out of all T related master nodes. That is, the arcs originating at master nodes i, i + m, i + 2m, ..., and i + (T - 1)m are examined together.

PNET-MP uses the standard upper bounded simplex minimum ratio test to determine the leaving arc. This is carried out by determining the maximum allowable flow change on the unique cycle, or basis equivalent path, created by adding the entering arc to the current basis tree. If more than one arc reaches a bound, then the first such arc identified is selected to leave the basis. A slightly more complicated selection rule has been developed [2, 4, 7, 8] that guarantees the finiteness of the network simplex

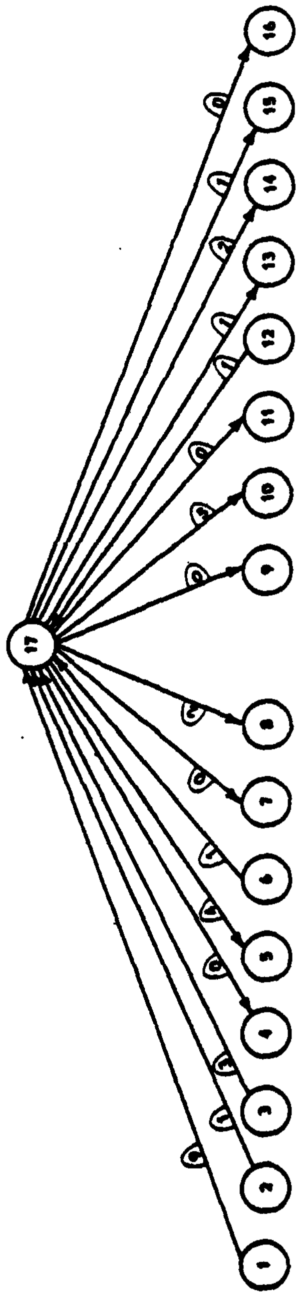


FIGURE 9

INITIAL BASIS TREE

algorithm, but it was not implemented in order to simplify comparisons with PNET, which uses the standard rule.

PNET uses a simple *flag* system to identify the arcs that are non-basic at upper bound. This system involves the negation of the arc length capacity array. PNET-MP cannot use this method since only the upper bounds of the loop arcs of the basic network are explicitly stored. Instead, PNET-MP uses a simple *bit map* technique to indicate the non-basic status of the master network arcs. Due to the special structure of this multi-period problem, the bit map can generally be implemented quite easily with only a single m length array.

PNET-MP performs the actual basis exchange operation, or pivot, in a manner identical to that used in PNET. The necessary updates of the various node functions are described in detail in [5, 12, 14].

In order to properly assess the performance of PNET-MP, twelve large multi-period test problems were generated. Table 5 shows the dimensions of both the basic and the master networks for each of the test problems. The table also presents the computational results on these problems for the specialized code, PNET-MP, as well as the general purpose pure network flow code, PNET.

In terms of core storage requirements, PNET-MP is clearly superior to PNET for this class of problems. For instance, on the largest problem in the test set, PNET requires more than four times the space of PNET-MP. In fact, due to excessive core requirements,

TABLE 5

PNET-MP VS. PNET

PROBLEM NUMBER	BASIC NETWORK		TIME PERIODS	MASTER NETWORK		PNET-MP					PNET				
	NODES	ARCS		NODES	ARCS	K WORDS	PIVOTS	OPTIMIZATION TIME	SOLUTION TIME	MEAN PIVOT TIME	K WORDS	PIVOTS	OPTIMIZATION TIME	SOLUTION TIME	MEAN PIVOT TIME
1	100	500	5	600	2500	4.7	1961	6.004	6.020	.003	11.1	2554	7.461	7.646	.003
2	100	500	10	1100	5000	7.7	4267	19.685	19.701	.005	21.6	6182	24.365	24.805	.004
3	100	500	15	1600	7500	10.7	7126	44.728	44.760	.006	32.1	13878	76.169	76.843	.005
4	100	1000	5	600	5000	5.7	2602	9.863	9.863	.004	18.6	3254	10.347	10.794	.003
5	100	1000	10	1100	10000	8.7	4430	26.662	26.678	.006	36.6	7223	30.528	31.309	.004
6	100	1000	15	1600	15000	11.7	7758	66.115	66.147	.009	54.6	16398	97.549	98.753	.006
7	200	1000	5	1200	5000	9.4	4995	22.325	22.357	.004	22.2	7338	32.091	32.589	.004
8	200	1000	10	2200	10000	15.4	11027	82.389	82.422	.007	43.2	20888	138.599	139.548	.007
9	200	1000	15	3200	15000	21.4	16168	162.791	162.840	.010	64.2*				
10	200	2000	5	1200	10000	11.4	5819	32.263	32.279	.006	37.2	7424	37.260	38.236	.005
11	200	2000	10	2200	20000	17.4	11934	104.415	104.447	.009	73.2*				
12	200	2000	15	3200	30000	23.4	17613	214.992	215.057	.012	109.2*				

All times are expressed in c.p.u. seconds on The University of Texas' CDC 6600 using the MNF FORTRAN compiler.

* CNR in 60,000 words.

PNET was unable to solve three of the problems on The University of Texas' CDC 6600.

Solution statistics reported in Table 5 include the number of pivots, optimization time, solution time, and mean time per pivot. All reported times are in c.p.u. seconds on the CDC 6600 using the MNF FORTRAN compiler. Times do not include any input or output operations. Solution time is equal to the sum of the start time (initialization and construction of the first basis tree) and the optimization time. The mean time per pivot is simply the ratio of the optimization time to the number of pivots.

At first it was surprising that PNET-MP actually solved the problems faster than PNET since PNET-MP was designed primarily to reduce core storage, not to execute faster. Upon closer examination it was determined that the success of PNET-MP revolves around the pivot strategy that it employs. Specifically, PNET-MP processes all of the master network arcs associated with a single basic network arc together, whereas PNET processes these master network arcs separately. As indicated by the number of pivots, the pivot strategy used by PNET-MP is superior, for this class of problems, to that used by PNET. As expected, PNET-MP requires more time per pivot than PNET since it must perform more work in order to construct the master network from its single copy of the basic network.

REFERENCES

1. Analysis, Research, and Computation, Inc., "PNET User's Guide," P.O. Box 4067, Austin, Texas 78765.

2. R. Barr, J. Elam, F. Glover, and D. Klingman, "A Network Augmenting Path Basis Algorithm for Transshipment Problems," Research Report CCS 272, Center for Cybernetic Studies, The University of Texas at Austin. To appear in *An International Symposium Volume on Extremal Methods and Systems Analysis.*

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

4. R. Barr, F. Glover, and D. Klingman, "The Alternating Basis Algorithm for Assignment Problems," *Mathematical Programming*, 13 (1977) 1-13.

5. R. Barr, F. Glover, and D. Klingman, "Enhancements of Spanning Tree Labeling Procedures for Network Optimization," *INFOR*, 17, 1 (1979) 16-34.

6. G. Bradley, G. Brown, and G. Graves, "Design and Implementation of Large-Scale Primal Transshipment Algorithms," *Management Science*, 24 (1977) 1-35.

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

8. W. H. Cunningham, "Theoretical Properties of the Network Simplex Method," *Mathematics of Operations Research*, 4 (1979) 196-208.

9. F. Glover, G. Jones, D. Karney, D. Klingman, and J. Mote, "An Integrated Production, Distribution, and Inventory Planning System," *Interfaces*, 9, 5 (1979) 21-35.

10. 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.

11. 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.

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

13. M. Grigoriadis and T. Hsu, "RNET: The Rutgers Minimum Cost Network Flow Subroutines," Department of Computer Science, Rutgers University, New Brunswick, New Jersey, 1978.

14. R. Helgason, J. Kennington, and J. Lall, "Primal Simplex Network Codes: State-of-the-Art Implementation Technology," Technical Report IEOR 76014, Department of Industrial Engineering and Operations Research, Southern Methodist University, 1976.

15. I. Heller and C. B. Tompkins, "An Extension of a Theorem of Dantzig's," in *Linear Inequalities and Related Systems*, H. W. Kuhn and A. W. Tucker (eds.), Princeton University Press, Princeton, New Jersey, 1956.

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

17. D. Klingman, A. Napier, and G. T. Ross, "A Computational Study on the Effects of Problem Dimensions on Solution Time for Transportation Problems," to appear in *Journal of the Association for Computing Machinery*.

18. A. Orden, "The Transshipment Problem," *Management Science*, 2, 3 (1956) 276-285.

19. H. Wagner, *Principles of Operations Research*, Prentice-Hall, Englewood Cliffs, New Jersey, 1969.

DAT
FILM