FOREST ITERATION METHOD FOR STOCHASTIC TRANSPORTATION PROBLEM

by

Liqun Qi

Computer Sciences Technical Report #522

November 1983

FOREST ITERATION METHOD FOR STOCHASTIC TRANSPORTATION PROBLEM*

Dedicated to Professor George B. Dantzig on the occasion of his 70th Birthday

Liqun Qi

Computer Sciences Department
University of Wisconsin-Madison
Madison, Wisconsin 53706

ABSTRACT

The transportation problem with stochastic demands is a special version of the stochastic linear programming problem with simple recourse. It has many economic applications. In this paper we present a new algorithm to solve this problem. Instead of discretizing the distribution functions of the stochastic demands, we explore the problem's network aspects and propose a forest iteration method to solve it. This method iterates from one base forest triple to another base forest triple with strictly decreasing objective values. Therefore, it converges in finitely many steps. The nonlinear work in each step consists of solving a small number of one-dimensional monotone equations.

Key words: stochastic transportation problem; tree; forest; convexity; iteration method.

---

*Sponsored by the National Science Foundation under Grant No. MCS-8200632.

1. Introduction

Mathematically, the transportation problem with stochastic demands is a special version of stochastic linear programming problem with simple recourse [10] [11] [21] [22] [27]. However, because of its wide economical applications and special network structure, it has attracted special attention and consideration. Papers dealing with this problem inlcude [1] [3] [4] [5] [6] [7] [19] [23] [24] [25] [26].

There are already several approaches to solving this problem. One approach discretizes the distribution functions of the random demands and approximates the stochastic transportation problem by an ordinary transportation problem of large size. This entails an increase in the number of variables and an approximation error in case the distribution functions are continuous, as is pointed out in [3] [6]. Another main approach uses various convex programming algorithms and makes some modifications according to the structure of the stochastic transportation problem. The convex programming algorithms used there are the Frank-Wolfe convex programming algorithms [1] [3], the Arrow-Hurwicz gradient algorithm [6] and the Dantzig convex programming algorithm [23]. This approach also includes some general methods for the stochastic linear programming with simple recourse [27]. In general, this approach is infinitely convergent. Elmaghraby [6] claimed that his method is finitely convergent. However, at each step, a system of nonlinear simultaneous equations is needed to solve and even infinitely many tableaux may ensue at each iteration of his method. Some other approaches only give bounds and approximate estimates to the solution [24] [25]. We are not going to discuss all these approaches.

In this paper, we present a new method to solve this problem. This method combines results from stochastic simple recourse problems and nonlinear network optimizations. It iterates from one base forest triple to another base forest triple with strictly decreasing objective values. The concept of base forest triple and other forest triples will be defined later. Since the number of base forests is finite, this method, the forest iteration method, converges in finetely many steps. The nonlinear work in each step consists of solving a small number of one-dimentionsl monotone equations. The focus of this method is to find an optimal forest. The final error of the optimal solution depends only on the data of the optimal forest, and not on any intermediate iteration steps.

In Section 2, we state the formulation of the stochastic transportation problem, as well as the sufficient and the necessary conditions for its optimal solution. We prove that it has an optimal solution with nonzero components only on a subset of a spanning tree of the transportation tableau. In Section 3, we discuss the graphical properties of such subsets, i.e. forests. In Section 4, we discard the nonnegativity restriction and solve the minimization problem in a forest. The minimization problem is then split into k small problems if this forest is a k-tree forest, and the nonlinear work consists of solving a one-dimensional equation on each tree. If we know the optimal forest in advance, then this immediately yields an optimal solution. In general, we can make a guess about the optimal forest, solve the minimization problem on this forest, then use the information from this solution to make a better guess. This leads to the forest iteration. In Section 5, we discuss the general idea of forest iterations. In Sections

6 and 7, we discuss the techniques in forest iteration, i.e., cutting, connecting and pivoting. In Section 8, we give the algorithm and the convergence theorem. In Section 9, we discuss how to extend this method to the stochastic minimal cost network flow problem. Numerical examples are given in Section 10.

2. The Optimal Solutions of Stochastic Transportation Problem

The standard formulation of the stochastic transportation problem with a dummy node is as follows [3] [19] [23] [24] [25] [26]:

$$
\begin{aligned}
\min_{x,w} \quad & \sum_{i=1}^{m} \sum_{j=1}^{n} c_{ij}x_{ij} + \sum_{j=1}^{n} \phi_j(w_j) \\
\text{s.t.} \quad & \sum_{j=1}^{n+1} x_{ij} = a_i, \ i=1,\dots,m, \\
& \sum_{i=1}^{m} x_{ij} = w_j, \ j=1,\dots,n+1, \\
& x_{ij} \geq 0, \ \forall i \text{ and } j,
\end{aligned}
\tag{2.1}
$$

where n+1 is the dummy node,

$$\phi_j(w_j) = q_j^+\int_{\xi_j<w_j}(w_j-\xi_j)dF_j(\xi_j) + q_j^-\int_{\xi_j>w_j}(\xi_j-w_j)dF_j(\xi_j). \tag{2.2}$$

a_i: the total amount available at i, $a_i > 0$.

c_{ij}: the cost of shipping one unit form i to j, $c_{ij} \geq 0$.

x_{ij}: the quantity of items shipped from i to j.

w_j: the total amount supplied to j, this value should be determined.

ξ_j: the observed value of $\tilde{\xi}_j$.

$\tilde{\xi}_j$: the random variable for demand at j.

F_j: the marginal distribution function of $\tilde{\xi}_j$, which is known.

q_j^+: the salvage cost per unit of excess inventory at j, $q_j^+ \geq 0$.

q_j^-: the penalty cost per unit of inventory shortage at j, $q_j^- \geq 0$.

Let $a = \sum_{i=1}^{m} a_i$. a is the total amount available. We can suppose $\tilde{\xi}_j$ has support $\Omega_j \subset [0,a]$. Then (2.2) becomes:

$$\phi_j(w_j) = q_j^+\int_0^{w_j}(w_j-\xi_j)dF_j(\xi_j) + q_j^-\int_{w_j}^{a}(\xi_j-w_j)dF_j(\xi_j). \tag{2.3}$$

According to [11] [21], we know that ϕ_j is continuous and convex and that

$$\partial\phi_j(w_j) = [-q_j^+ + q_jF_j(w_j),\ -q_j^+ + q_jF_j^+(w_j)], \tag{2.4}$$

where $q_j = q_j^+ + q_j^-$. Here we assume left continuity of distribution functions. If F_j is continuous, then

$$\partial\phi_j(w_j) = \phi_j'(w_j) = -q_j^+ + q_jF(w_j) \tag{2.5}$$

Since the feasible region is compact and the object function is continuous, (2.1) is always solvable. According to convex program theory [16] [17], (x,w) is an optimal solution if and only if there exist $u \in \mathbb{R}^m$, $v \in \mathbb{R}^{n+1}$ such that:

$$\begin{aligned}
&\sum_{j=1}^{n+1} x_{ij} = a_i,\ i=1,\dots,m,\\
&\sum_{i=1}^{m} x_{ij} = w_j,\ j=1,\dots,n+1,\\
&x_{ij} \geq 0\ ,\ \forall i \text{ and } j,\\
&u_i + v_j \leq c_{ij},\ \forall i \text{ and } j,\\
&x_{ij}(c_{ij}-u_i-v_j) = 0,\ \forall i \text{ and } j,\\
&-v_j \in \partial\phi_j(w_j),\ j=1,\dots,n,\\
&v_{n+1} = 0,
\end{aligned} \tag{2.6}$$

where $c_{i,n+1} = 0$, $i=1,\dots,m$.

If we fix w in (2.1) and only minimize on x, we have an ordinary transportation problem of the original size. Denote it by $T(w)$. It can be easily solved by transportation algorithm [2] [8] if and only if

$$w \in S = \{w \mid w \geq 0,\ \sum_{j=1}^{n+1} w_j = a\}.$$

Theorem 2.1 (2.1) has an optimal solution (x^*,w^*), where $w^* \in S$ and $x^*_{ij} = 0$ except on a subset of a spanning tree of an $m \times (n+1)$ transportation tableau.

Proof Suppose $(\bar{x},w^*)$ is an optimal solution of (2.1). Solve the transportation problem $T(w^*)$; we get an optimal solution x^* whose components are zeros except on a subset of a spanning tree of $m \times (n+1)$ transportation tableau [2]. Checking the feasibility of (x^*,w^*) and comparing the objective values of (x^*,w^*) and $(\bar{x},w^*)$, we know (x^*,w^*) is also an optimal solution of (2.1). □

This suggests that we consider subsets of a spanning tree of a transportation tableau. We do this in the next section.

We use cx to denote $\sum_{i=1}^{m} \sum_{j=1}^{n+1} c_{ij}x_{ij}$ and $\phi(w)$ to denote $\sum_{j=1}^{n} \phi_j(w_j)$, and so on. If we use x and w without subscripts, then we always mean $x \in \mathbf{R}^{m\times(n+1)}$ and $w \in \mathbf{R}^{n+1}$.

3. Forests in a Transportation Tableau

Suppose we have an $m \times (n+1)$ transportation tableau T with m rows and $n + 1$ columns.

Definition 3.1 We call a pair of integer indices (i,j) a cell of T if $1 \leq i \leq m$, $1 \leq j \leq n + 1$. A graph is a set of cells, whose cells are connected if they are in the same row or column. A tree is a connected graph without cycles. A spanning tree is a tree whose row indices run throughout $\{1,\ldots,m\}$ and whose column indices run throughout $\{1,\ldots,n+1\}$. A forest is a graph each of whose components, i.e., connected parts, is a tree, and whose row indices run throughout $\{1,\ldots,m\}$ □

Notice that a group of trees does not necessarily form a forest. In Fig. 1, we have a forest of two trees. If we add two cells, say (1,2) and (3,1), to the left tree, it is still a tree but the whole graph is not a forest at all because we have a cycle {(1,1), (1,2), (2,2), (4,2), (4,7), (3,7), (3,1)}.

We know that the number of the cells of a tree is no more than $m + n$ and that a tree is a spanning tree if and only if it has $m + n$ cells [2]. Unsurprisingly, this is also true in general for a forest.

Theorem 3.2 The number of the cells of a forest is no more than $m + n$. More exactly, the number of the cells of a k-tree forest is no more than $m + n + 1 - k$. A forest is a spanning tree if and only if it has $m + n$ cells. A forest can always be expanded into a spanning tree.

Proof The key property of a forest is that there is no intersection between two row (column) index sets of two distinct component trees of a forest.

Suppose that we have a k-tree forest f, whose component trees are $t_1,\dots,t_k$. Suppose there are m_h row indices and n_h column indices for a component tree t_h. Then t_h is a spanning tree of an $m_h \times n_h$ transportation tableau. The number of cells of t_h is $m_h + n_h - 1$. Since we know

$$\sum_{h=1}^{k} m_h = m, \quad \sum_{h=1}^{k} n_h \leq n + 1,$$

and since the total number of nodes of f is $\sum_{h=1}^{k} (m_h + n_h - 1)$, we get our first conclusion. Since a k-tree forest has no more than $m + n + 1 - k$ cells, it is a spanning tree if and only if $k = 1$ and it is spanning. We can connect two component trees of a forest into one component tree of this forest if we add a cell (i,j) where i is a row index of the first tree and j is a column index of the second tree, or vice versa. In this way, we can connect (expand) a forest to a tree and finally expand it to a spanning tree. □

We now associate these concepts to points in $\mathbb{R}^{m\times(n+1)}$. Let $x \in \mathbb{R}^{m\times(n+1)}$. Let Gr x, the graph of x, be the graph associated with the set

$$\{(i,j) | x_{ij} \neq 0\}.$$

Then Theorem 2.1 can be rewritten in our forest terminology.

Theorem 3.3 (2.1) has an optimal solution (x^*,w^*), where $w^* \in S$ and Gr x^* is a forest of the transportation tableau. □

<u>Definition 3.4</u> If (x^*,w^*) is an optimal solution of (2.1) and Gr x^* is a forest, then we call this forest $f^* = \text{Gr } x^*$ an <u>optimal forest</u> of (2.1), and $(x^*,w^*;\ f^*)$ an <u>optimal forest triple</u> of (2.1). □

If we know such an optimal forest, then (2.1) becomes an equality constrained problem:

$$\begin{aligned} \min_{x,w} \quad & \sum_{(i,j)\in f^*} c_{ij}x_{ij} + \sum_{j\in N_{f^*}} \phi_j(w_j) \\ \text{s.t.} \quad & \sum_{(i,j)\in f^*}^{j} x_{ij} = a_i,\ i=1,\dots,m, \\ & \sum_{(i,j)\in f^*}^{i} x_{ij} = w_j,\ j \in N_{f^*}. \end{aligned} \tag{3.1}$$

Here we use N_f to denote the column index set of a forest f. We also use M_t and N_t to denote the row and column index sets of a tree t. Then

$$\begin{aligned} & \bigcup_{t\subset f} M_t = \{1,\dots,m\}, \\ & N_t \cap N_s = \phi,\ M_t \cap M_s = \phi \quad \text{if} \quad t,\ s \subset f,\ t \neq s,\ t,\ s \text{ trees.} \end{aligned} \tag{3.2}$$

If $j \notin N_{f^*}$, then $w_j = 0$ in (3.1).

Can (3.1) be easily solved? In the next section, we answer this question affirmatively.

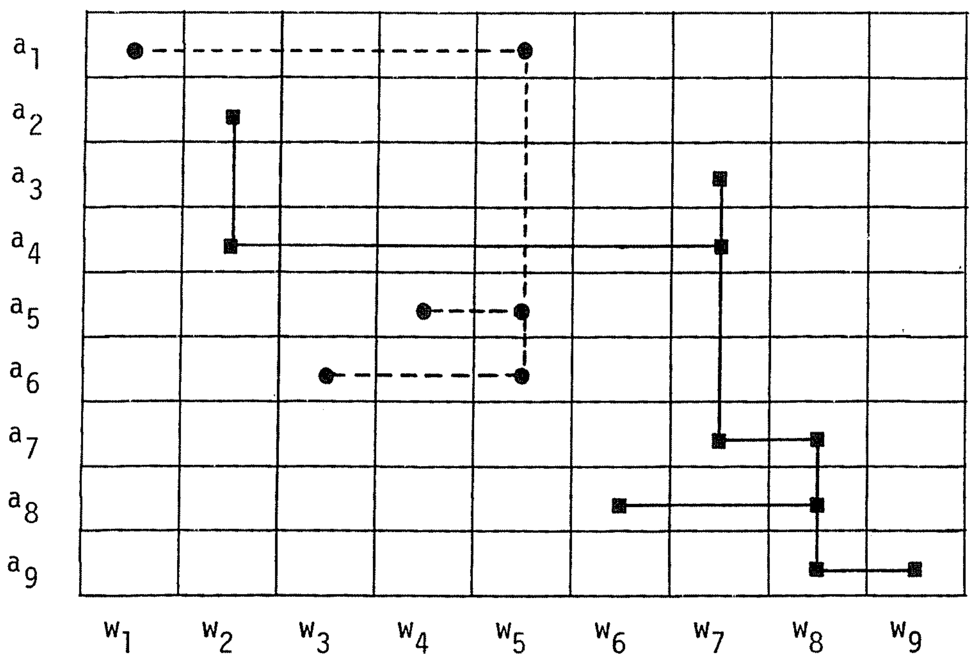

Figure 1. Forest

4. Minimization on a Forest

We now consider

$$\min_{x,w} \sum_{(i,j)\in f} c_{ij}x_{ij} + \sum_{j\in N_f} \phi_j(w_j)$$

$$\text{s.t.} \sum_{(i,j)\in f}^{j} x_{ij} = a_i, \; i=1,\ldots,m, \tag{4.1}$$

$$\sum_{(i,j)\in f}^{i} x_{ij} = w_j, \; j \in N_f,$$

where f is a forest.

Theorem 4.1 If f is a k-tree forest, $k > 1$, then (4.1) can be separated into k minimization problems:

$$\min_{x,w} \sum_{(i,j)\in t} c_{ij}x_{ij} + \sum_{j\in N_t} \phi_j(w_j)$$

$$\text{s.t.} \sum_{(i,j)\in t}^{j} x_{ij} = a_i, \; i \in M_t, \tag{4.2}$$

$$\sum_{(i,j)\in t}^{i} x_{ij} = w_j, \; j \in N_t,$$

where t's are component trees of f.

Proof Since there is no intersection of any two pair of row (column) index sets of two distinct component trees of a forest, we get the conclusion. □

We now discuss problem (4.2) for a tree t. There are two kinds of component trees of a forest. There is at most one tree with the dummy node $n + 1$ and the other trees do not involve the dummy node. We discuss them separately.

A. Minimization on a tree with the dummy node.

The right tree of Fig. 1 is an example. The necessary and sufficient conditions of optimal solutions on such a tree t are:

$$
\begin{aligned}
&\sum_{(i,j)\in t}^{j} x_{ij} = a_i, \; i \in M_t, \\
&\sum_{(i,j)\in t}^{i} x_{ij} = w_j, \; j \in N_t, \\
&u_i + v_j = c_{ij}, \; (i,j)\in t, \qquad (4.3) \\
&- v_j \in \partial\phi_j(w_j), \; j \in N_t, \\
&v_{n+1} = 0.
\end{aligned}
$$

According to the theory of the transportation problem [2] [5], the third and the fifth expressions of (4.3) form a triangular linear system in u_i and v_j . Therefore, we can easily get u_i and v_j from them. Then, we can use the fourth expression and v_j to determine w_j. Similarly, the first and the second expressions of (4.3) form a triangular system of x_{ij} . We can easily get x_{ij} from them now. The practical solution procedure for a tree can be found in [2].

B. Minimization on a tree without the dummy node.

The left tree of Fig. 1 is an example. The necessary and sufficient conditions of optimal solutions on such a tree t are:

$$
\begin{aligned}
&\sum_{(i,j)\in t}^{j} x_{ij} = a_i, \; i \in M_t, \\
&\sum_{(i,j)\in t}^{i} x_{ij} = w_j, \; j \in N_t, \qquad (4.4) \\
&u_i + v_j = c_{ij}, \; (i,j)\in t, \\
&- v_j \in \partial\phi_j(w_j), \; j \in N_t.
\end{aligned}
$$

We can fix a $j_0 \in N_t$ and assume $v_{j_0}^0 = 0$. This condition and the third expression of (4.4) form a triangular linear system. Using them, we can get a set of solutions $\{u_i^0, v_j^0 \mid (i,j) \in t\}$. For general v_{j_0}, the solutions are

$$u_i = u_i^0 - v_{j_0}, \; i \in M_t, \qquad (4.5)$$
$$v_j = v_j^0 + v_{j_0}, \; j \in N_t.$$

From the first and second expressions we know

$$\sum_{j \in N_t} w_j = \sum_{i \in M_t} a_i =: a_t. \qquad (4.6)$$

From the fourth expression of (4.4) and from (4.5), we know w_j, $j \in N_t$, are nonincreasing function of v_{j_0}. Expecially, for continuous distributions, from (2.5), we have $w_j = F_j^{-1}((q_j^+ - v_j^0 - v_{j_0})/q_j)$, $j \in N_t$. We can write $P(v_{j_0}) = \sum_{j \in N_t} w_j$ and (4.6) becomes

$$P(v_{j_0}) = a_t, \qquad (4.7)$$

for a nonincreasing function P. We can solve this one-dimentional equation (4.7) to get v_{j_0}. By (4.5), we get u_i and v_j. By the fourth expression of (4.4), we get w_j. Similarly to A, from the first and second expressions of (4.4), we get x_{ij}.

We see that the nonlinear work of (4.1) is no more than solving k one-dimensional equations.

Remark 4.2 Since the solution work of (4.7) comprises the main nonlinear work of our method, we analyse it more carefully:

(1). To solve one-dimensional monotone equations is not difficult even if the expression is implicit. We can even use the bisection method to solve them.

(2). In the next sections, we shall see that in the intermediate stage of our iterations, the solution of (4.1) is used in cutting to get a new forest triple. The focus is to get a new forest. Therefore, it is not necessary to solve (4.7) exactly in the intermediate stage of our iterations.

(3). When an optimal forest is in hand, we can use bisection method or other one-dimensional methods to solve (4.7) to any precision without much difficulty. □

5. Forest Iteration and Base Forests

Unfortunately, (4.1), though easily solved, usually will not give an optimal solution of (2.1). The optimal solutions of (4.1) may not be non-negative at all. Therefore, they may be infeasible for (2.1).

A reasonable way to solve this problem is as follows. We make a guess about f^* and solve the associated problem (4.1). If the solution is not optimal, we make a better guess about f^* using the knowledge we have gotten from the former guess. This is the general idea of the iteration. Then we get $f_1, f_2, \ldots, f_r, \ldots$ until $f_R = f^*$ for some optimal forest f^*.

Therefore, we have a TARGET: a certain optimal forest f^*. To reach this target, we need a travel PRINCIPLE. A simple principle is the strictly decreasing objective value principle, denoted by SDOVP. However, if (4.1) has no nonnegative optimal solution, it is nonsense to talk about its optimal objective value. Therefore, we consider nonnegative feasible points of (4.1). In general, such a point is called a forest point.

Definition 5.1 If (x,w) is a feasible point of (2.1) and $f = \text{Gr}\ x$ is a forest, then we call (x,w) a forest point and $(x,w;f)$ a forest triple. □

We move from one forest triple to another forest triple according to SDOVP. However, even if we move from one forest to another distinct forest in each step, it is still possible that we wander in several nonoptimal forests, i.e., a well-known phenomenon of zigzagging [9] may occur. To avoid such situation, we need some LANDMARKS which we shall not pass more than once. Here is such a landmark:

Definition 5.2 If $(x,w;f)$ is a forest triple and the corresponding part of (x,w) is an optimal solution of (4.1) associated with f, then f is called a base forest and $(x,w:f)$ is called a base forest triple.

Remark 5.3 We have

$$\begin{aligned} x_{ij} &> 0, \ \forall (i,j) \in f, \\ x_{ij} &= 0, \ \forall (i,j) \notin f, \end{aligned} \tag{5.1}$$

in this case, i.e., the corresponding part of x as an optimal solution of (4.1) is positive. □

Remark 5.4 According to Definitions 3.4 and 5.2, optimal forests are base forests with the lowest objective value and $(x^*,w^*;f^*)$ in that definition is a base forest triple with the lowest objective value. □

Remark 5.5 Since the number of base forests is finite, if we move from one base forest triple to another base forest triple according to SDOVP, we shall reach an optimal forest triple in finitely many steps. □

There are two other landmarks: complete forests and total forests. We shall introduce them in Section 6.

To make the forest iteration successful, we need some techniques:

1. Cutting ——— introducing more zero levels to prune trees, sometimes splitting trees.
2. Connecting —— cancelling some zero restrictions to connect trees.
3. Pivoting ——— pivoting from one tree to another tree.

We shall discuss them in the next two sections.

6. Cutting

The problem is that: if we have a forest triple $(x,w;f)$, which is not a base forest triple, how can we get a base forest triple $(\bar{x},\bar{w};\bar{f})$ with

$$c\bar{x} + \phi(\bar{w}) < cx + \phi(w)\ ? \tag{6.1}$$

From now on, we shall talk about an optimal solution $(\hat{x},\hat{w})$ of (4.1) and mean that $(\hat{x},\hat{w}) \in \mathbb{R}^{m\times(n+1)} \times \mathbb{R}^{n+1}$, that the corresponding part of $(\hat{x},\hat{w})$ is an optimal solution of (4.1) and that the other components are zeros.

Theorem 6.1 (Cutting) Suppose we have a forest triple $(x,w;f)$, which is not a base forest triple. Solve the problem (4.1) associated with f and get an optimal solution $(\hat{x},\hat{w})$. Then

$$c\hat{x} + \phi(\hat{w}) < cx + \phi(w)\ . \tag{6.2}$$

If $I = \{(i,j) | (i,j) \in f,\ \hat{x}_{ij} < 0\} = \phi$, then $(\hat{x},\hat{w};\hat{f} = \mathrm{Gr}\ \hat{x})$ is a base forest triple. If $I \neq \phi$, then

$$0 < \theta < 1, \tag{6.3}$$

where

$$\theta = \min_{(i,j)\in I} \frac{x_{ij}}{x_{ij}-\hat{x}_{ij}}\ . \tag{6.4}$$

Let

$$x'_{ij} = x_{ij} - \theta(x_{ij} - \hat{x}_{ij}),\ \forall i \text{ and } j\ , \tag{6.5}$$

and let w' correspond to x', $f' = \mathrm{Gr}\ x'$. Then $(x', w'; f')$ is a forest triple and

$$cx' + \phi(w') < cx + \phi(w), \tag{6.6}$$

$$f' \subsetneq f. \qquad \square \tag{6.7}$$

Proof Since $(x,w;f)$ is not a base forest triple, by definition (5.2), we get (6.2). Since

$$x_{ij} > 0, \ \forall (i,j) \in I \subset f,$$

$$\hat{x}_{ij} < 0, \ \forall (i,j) \in I,$$

we get (6.3). Hence $x' \geq 0$ and we know (6.7). By convexity and (6.4), we get (6.6). (See Fig. 2,3). $\square$

There is an interesting fact about cutting:

Theorem 6.2 Cutting always splits at least one tree except in the single case for which cutting makes only one cell to zero level and this cell is the only cell in the dummy node column.

Proof Use the notation in the proof of Theorem 6.1. We divide the cells of f into two categories: corner cells and non-corner cells. A cell $(i,j) \in f$ is called a corner cell of f if there are $i' \neq i$, $j' \neq j$ such that $(i',j) \in f$, $(i,j') \in f$. It is not difficult to see that if we delete from f only some corner cells, then f must be split. It suffices to prove that only corner cells can be deleted except the case mentioned in the theorem.

According to (6.4) and (6.5), only (i,j)'s for which $\hat{x}_{ij} < 0$ may be deleted. It therefore suffices to prove $(i,j) \in I \Rightarrow (i,j)$ is a corner cell except the mentioned case.

Since $\hat{w}_j$, $j \neq n+1$, is determined by $-\hat{v}_j \in \partial\phi_j(\hat{w}_j)$ and (2.3), we know that $\hat{w}_j \geq 0$, $j \neq n+1$.

If $(i,j) \in f$ and there is no j' such that $j' \neq j$ and $(i,j') \in f$, then $\hat{x}_{ij} = a_i > 0$. If $(i,j) \in f$, $j \neq n+1$ and there is no i' such that $i' \neq i$ and $(i',j) \in f$, then $\hat{x}_{ij} = \hat{w}_j \geq 0$. Therefore, if $(i,j) \in I$, (i,j) is not a corner cell, then

(1) $j = n+1$.

(2) there is $j' \neq n+1$ such that $(i,j') \in f$.

(3) there is no $i' \neq i$ such that $(i', n+1) \in f$, otherwise (i,j) will be a corner cell by (2).

This is exactly the exception case mentioned in the theorem. □

Theorem 6.3 Suppose we have a forest triple $(x,w;f)$, which is not a base forest triple. By repeating the cutting technique described in Theorem 6.1 at most n times, we obtain a base forest triple $(\bar{x},\bar{w};\bar{f})$ satisfying (6.1).

Proof By (6.7), the number of cells of the forest is strictly decreasing. Since the number of cells of a forest is no more than $m+n$ and no less than n, and since (6.6) holds, we get the conclusion. □

Remark 6.4 When we solve (4.1) in the cutting process, it is only necessary to solve (4.2) on those trees which are newly created. □

Suppose $(x,w;f)$ is a base forest triple and (u,v) are the corresponding multipliers of the optimal solution (x,w) of problem (4.1) associated with f. Then we have

$$\sum_{j=1}^{n+1} x_{ij} = a_i, \; i=1,\dots,m,$$

$$\sum_{i=1}^{m} x_{ij} = w_j, \; j=1,\dots,n+1,$$

$$x_{ij} \geq 0, \quad \forall i \text{ and } j,$$

$$u_i + v_j = c_{ij}, \; \forall (i,j) \in f, \qquad (6.8)$$

$$-v_j \in \partial\phi_j(w_j), \; j=1,\dots,n,$$

$$v_{n+1} = 0 .$$

Comparing (6.8) with (2.6), we know that $(x,w;f)$ is an optimal forest triple if

$$u_i + v_j \leq c_{ij}, \quad \forall (i,j) \notin f . \qquad (6.9)$$

We now define two further landmarks:

<u>Definition 6.5</u> A base forest triple $(x,w;f)$ is called a <u>total forest triple</u> if (6.8) holds and for every component tree t of f,

$$u_i + v_j \leq c_{ij}, \quad \forall i \in M_t, \; j \in N_t. \qquad (6.10)$$

A base forest triple $(x,w;f)$ is called a <u>complete forest triple</u> if (6.8) holds and there is a spanning tree $\bar{t} \supset f$ such that

$$u_i + v_j \leq c_{ij}, \quad \forall (i,j) \in \bar{t} . \quad \square \qquad (6.11)$$

In the next section, we shall see how we can get such a special base forest triple. Notice that an optimal forest triple is both a total and a complete forest triple.

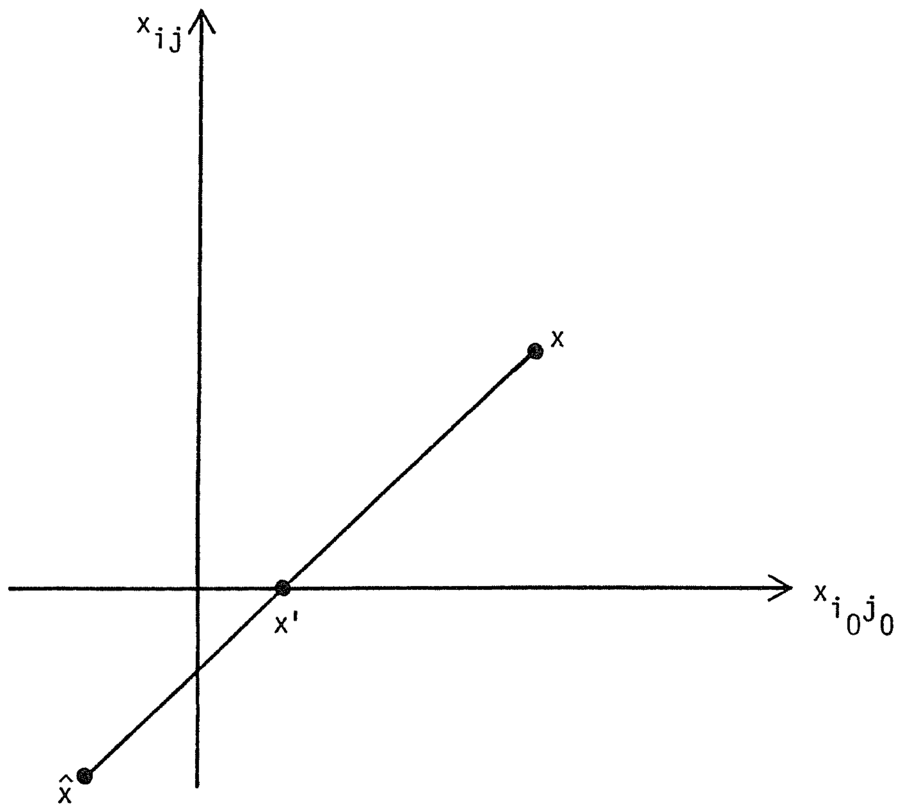

Figure 2. Cutting

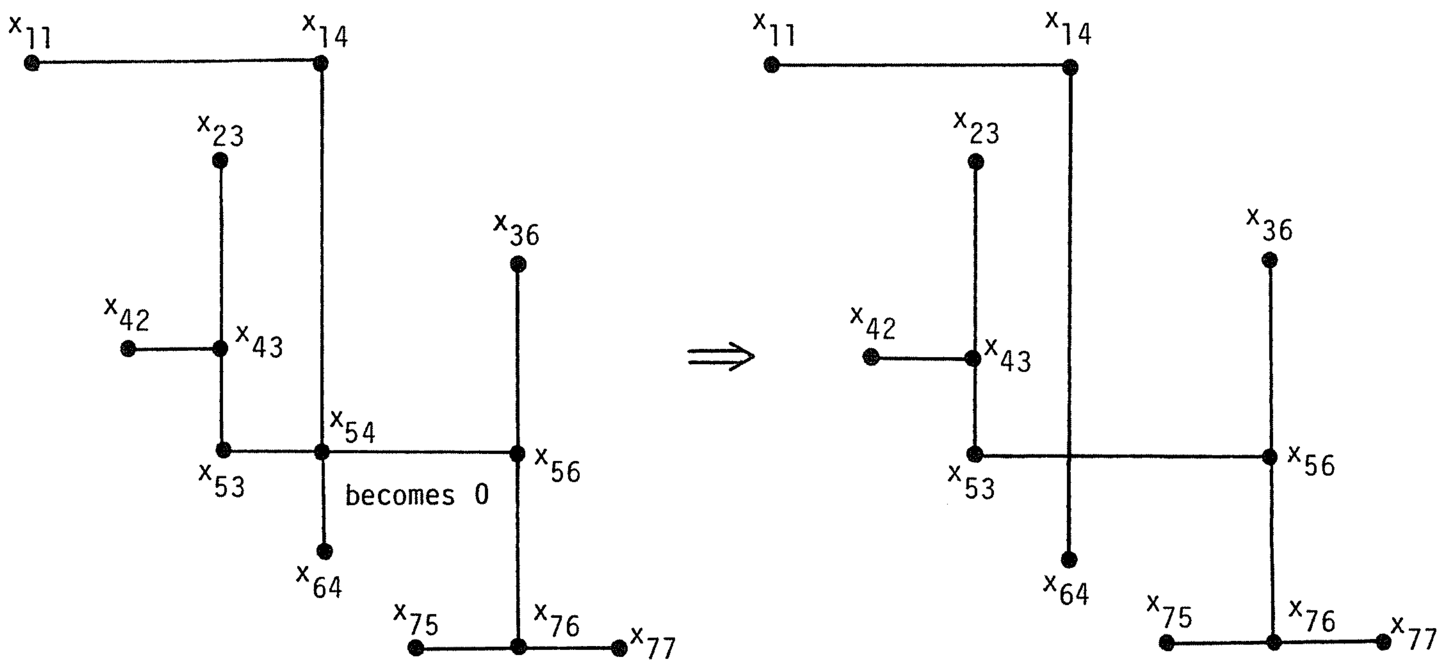

Figure 3. Cutting with Splitting

7. Pivoting and Connecting

Theorem 7.1 (Pivoting) Suppose $(x,w;f)$ is a base forest triple but not a total forest triple, i.e., there is a component tree t of f for which (6.10) does not hold. Solve the linear transportation problem with w fixed on t:

$$\begin{aligned} \min_{x} \quad & \sum_{i\in M_t} \sum_{j\in N_t} c_{ij}x_{ij} + \sum_{j\in N_t} \phi_j(w_j) \\ \text{s.t.} \quad & \sum_{j\in N_t} x_{ij} = a_i, \; i \in M_t , \\ & \sum_{i\in M_t} x_{ij} = w_j, \; j \in N_t , \\ & x_{ij} \geq 0, \; \forall i \in M_t, \; j \in N_t , \end{aligned} \tag{7.1}$$

and get a basic optimal solution $\{\bar{x}_{ij}\ i \in M_t, j \in N_t\}$. Let

$$\bar{x}_{ij} = x_{ij}, \quad \text{for other } (i,j) , \tag{7.2}$$

and let $\bar{f} = \text{Gr}\ \bar{x}$. Then $(\bar{x},w;\bar{f})$ is a forest triple with

$$c\bar{x} + \phi(w) < cx + \phi(w) . \tag{7.3}$$

Proof Since $x_{ij} > 0$, $\forall(i,j) \in t$ and since (6.10) does not hold, this is a nondegenerate case and x is not an optimal solution of (7.1). Therefore, (7.3) holds. □

Remark 7.2 If we do some cutting after a pivoting as described in the last section, we shall get a base forest triple with lower objective values. If we repeat this process, we shall get a total forest triple in finitely many steps since the number of base forests is finite. □

Theorem 7.3 (Connecting) Suppose $(x,w;f)$ is a base forest triple and

$$u_{i_0} + v_{i_0} > c_{i_0 j_0} \quad \text{for a pair } (i_0, j_0),\ i_0 \in M_s,\ j_0 \in N_t, \tag{7.4}$$

where s and t are distinct component trees of f, (u,v) is the multiplier described by (6.8). Pick j_1 such that $(i_0, j_1) \in s$. Since $a_{i_0} = \sum_{(i_0,j) \in s} x_{i_0 j} > 0$, such j_1 always exists. If F is continuous, then there exists $\delta_0 > 0$ such that for $0 < \delta < \delta_0$ and

$$\begin{aligned} \bar{x}_{i_0 j_0} &= \delta,\ \bar{w}_{j_0} = w_{j_0} + \delta, \\ \bar{x}_{i_0 j_1} &= x_{i_0 j_1} - \delta,\ w_{j_1} = w_{j_1} - \delta, \\ \bar{x}_{ij} &= x_{ij},\ \bar{w}_j = w_j \quad \text{for other } (i,j) \text{ and } j, \end{aligned} \tag{7.5}$$

and $(\bar{x},\bar{w};\bar{f} = Gr\,\bar{x})$ is a forest triple with

$$c\bar{x} + \phi(\bar{w}) < cx + \phi(w). \tag{7.6}$$

Proof Since $x_{i_0 j_1} > 0$, if δ_0 is small enough, $\bar{x} \geq 0$. This proves that $(\bar{x},\bar{w};\bar{f})$ is a forest triple. From (2.5), we know ϕ is differentiable.

$$\begin{aligned} e(\delta) &:= (c\bar{x} + \phi(\bar{w})) - (cx + \phi(w)) \\ &= (c_{i_0 j_0} - c_{i_0 j_1})\delta + \phi_{j_0}(w_{j_0} + \delta) - \phi_{j_0}(w_{j_0}) + \phi_{j_1}(w_{j_1} - \delta) - \phi_{j_1}(w_{j_1}). \\ e'(0) &= c_{i_0 j_0} - c_{i_0 j_1} + \phi'_{j_0}(w_{j_0}) - \phi'_{j_1}(w_{j_1}) \\ &= c_{i_0 j_0} - u_{i_0} - v_{j_1} + \phi'_{j_0}(w_{j_0}) - \phi'_{j_1}(w_{j_1}). \\ &= c_{i_0 j_0} - u_{i_0} - v_{j_0} + v_{j_0} + \phi'_{j_0}(w_{j_0}) - v_{j_1} - \phi'_{j_1}(w_{j_1}). \end{aligned}$$

The second equality of $e'(0)$ is due to $(i_0, j_1) \in s$, therefore

$$c_{i_0 j_1} = u_{i_0} + v_{j_1}.$$

From (2.5) and (2.6), we have

$$-v_{j_0} = \phi'_{j_0}(w_{j_0}), \quad -v_{j_1} = \phi'_{j_1}(w_{j_1}),$$

we have

$$0 > c_{i_0 j_0} - u_{i_0} - v_{j_0} = e'(0).$$

This implies for δ small enough, (7.6) holds. □

Remark 7.4 This connects s and t to a tree. We can connect several groups of trees at the same time. The δ can be found by using (7.5) and trying $\{\bar{\delta}, \frac{\bar{\delta}}{2}, \frac{\bar{\delta}}{4}, \ldots\}$. If we combine this technique with cutting, we shall get a complete forest triple in finitely many steps as in Remark 7.2 for total forests. □

Remark 7.5 Theorem 7.3 also covers the following case: $N_f \ddagger \{1,\ldots,n+1\}$, $N_t = \{1,\ldots,n+1\} \backslash N_f$, i.e., $w_{j_0} = 0$. In this case, we expand a tree instead of connecting two trees. □

$(i_0, j_0) = (1,3)$

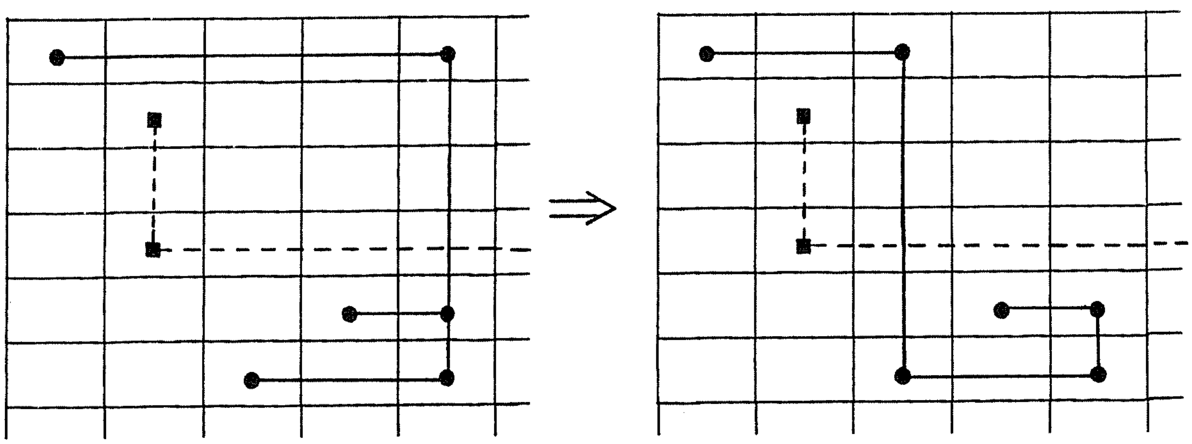

Figure 4. Pivoting

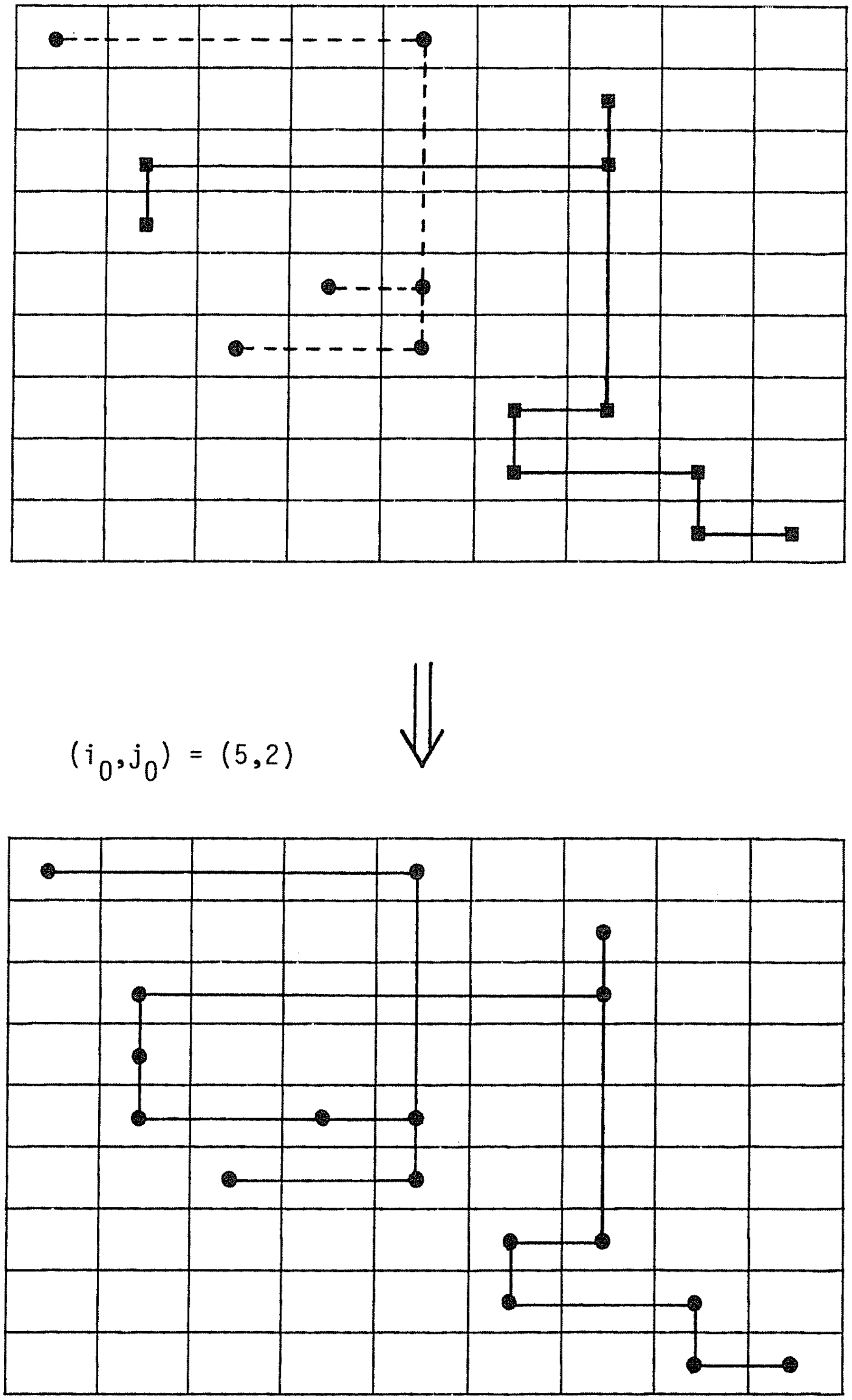

Figure 5. Connecting

8. Convergent Forest Iteration Methods

Now we can give the algorithm. Let $\bar{\xi}$ be the expectation of ξ.

Algorithm 8.1 (Forest Iteration Algorithm) Let w^0 be an estimate of the optimal solution w^*. For example, we may choose $w^0 = \bar{\xi}$ or some approximate values provided in [24] [25]. Solve $T(w^0)$ by the transportation algorithm. Suppose we get a basic optimal solution x^0 and $f^0 = Gr\ x^0$. Then $(x^0, w^0; f^0)$ is a forest triple.

1. Suppose we have a forest triple $(x^\ell, w^\ell; f^\ell)$ that is not a base forest triple. Use the cutting technique to get a base forest triple $(\bar{x}^\ell, \bar{w}^\ell; \bar{f}^\ell)$ such that

$$c\bar{x}^\ell + \phi(\bar{w}^\ell) < cx^\ell + \phi(w^\ell) . \tag{8.1}$$

   If $(x^\ell, w^\ell; f^\ell)$ is already a base forest triple, let $(\bar{x}^\ell, \bar{w}^\ell; \bar{f}^\ell) = (x^\ell, w^\ell; f^\ell)$.

2. Check (6.9) for the multipliers of $(\bar{x}^\ell, \bar{w}^\ell)$ of the problem (4.1) associated with $\bar{f}^\ell$. If (6.9) holds, then $(\bar{x}^\ell, \bar{w}^\ell; \bar{f}^\ell)$ is an optimal forest triple. Otherwise, use the pivoting and connecting techniques to get a forest triple $(x^{\ell+1}, w^{\ell+1}; f^{\ell+1})$ such that,

$$cx^{\ell+1} + \phi(w^{\ell+1}) < c\bar{x}^\ell + \phi(\bar{w}^\ell) . \tag{8.2}$$

   Go to step 1.

Theorem 8.2 (Convergence Theorem) If F is continuous, then Algorithm 8.1 converges in finitely many steps with SDOVP.

Proof This follows from (8.2) and the finiteness of the number of base forests. □

There are many choices in Step 2. We can choose pivoting and connecting according to $\max_{(i,j)\notin f} (u_i + v_j - c_{ij})$. Alternatively, we can do pivoting first until we get a total forest triple, or we can do connecting first until we get a complete forest.

In practice, some demands may be stochastic while the rest are deterministic. Also, not all of the cells may be available. The formulation then is

$$\min_{\substack{x,w_j \\ k+1\leq j\leq k+n}} \sum_{(i,j)\in S} c_{ij}x_{ij} + \sum_{j=k+1}^{k+n} \phi_j(w_j)$$

$$\text{s.t.} \quad \sum_{(i,j)\in S} {}_j \; x_{ij} \leq a_i, \; i=1,\ldots,m,$$

$$\sum_{(i,j)\in S} {}_i \; x_{ij} = w_j, \; j=1,\ldots,k+n,$$

$$x_{ij} \geq 0, \; \forall i \text{ and } j .$$

where S is the set of available cells and w_j, $j=1,\ldots,k$ are the known (deterministic) demands. Obviously, our forest discussion still holds with little change.

Remark 8.3 As pointed out in Remark 4.2, the focus of our method is to get an optimal forest. Before an optimal forest is obtained, the exact iteration value of (x,w) is not so important compared with the iterated forest f. When an optimal forest is in hand, we can use the method described in Section 4 to get an optimal solution to arbitrary precision without much difficulty. Furthermore, the error of the optimal solution depends only on the data of the optimal forest. This is another merit of our method.

9. Stochastic Minimal Cost Network Flow Problem

The ideas described above can be applied to the minimal cost network flow problem with stochastic demands. The formulation of a minimal cost network flow problem with stochastic demands is as follows:

$$\min_{x,w} \sum_{(i,j)\in S} c_{ij}x_{ij} + \sum_{k=m+1}^{n} \phi_k(w_k)$$

$$\text{s.t.} \quad \sum_{\substack{j\\(i,j)\in S}} x_{ij} - \sum_{\substack{j\\(i,j)\in S}} x_{ji} \leq a_i, \quad i=1,\ldots,\ell,$$

$$\sum_{\substack{j\\(i,j)\in S}} x_{ij} - \sum_{\substack{j\\(j,i)\in S}} x_{ji} = b_i, \quad i=\ell+1,\ldots,m, \tag{9.1}$$

$$\sum_{\substack{j\\(i,j)\in S}} x_{ij} - \sum_{\substack{j\\(j,i)\in S}} x_{ji} = w_i, \quad i=m+1,\ldots,n,$$

$$x_{ij} \geq 0, \quad \forall (i,j) \in S,$$

where S is the set of all arcs, a_i's are amounts of supplies available at $i=1,\ldots,\ell$, b_i's are amounts of supplies available or demands required, depending on their signs, at $i=\ell+1,\ldots,m$, $-w_i$'s are the total amounts supplied to $i=m+1,\ldots,n$, where the demands are stochastic, $w_i \leq 0$. The ϕ_k's have the same formula as in (2.2). We also can introduce a dummy node $n+1$, and discuss forests, base forests, optimal forests and in fact everything we have done for the stochastic transportation problem. Everything is similar. We use multipliers u instead of (u,v) and we can replace v_j by $-u_{j+m}$ wherever v_j appears; we use

$$\sum_{i\in L_t} a_i + \sum_{i\in M_t} b_i + \sum_{i\in N_t} w_i = 0 \tag{9.2}$$

instead of (4.6), and so on. In fact, the only change is that we use negative values to represent demands. The algorithms, theorems and results are all similar.

10. Numerical Examples

We first give a simple, small example to illustrate our algorithm, then briefly report our computational results for some medium and large examples.

In this example, $m = 4$, $n = 5$, cells (2,1), (3,1) and (3,3) are not available. Instead of using column 6, we use column 0 as our dummy node column. The other data are as in Table 1:

i	c					a
	j = 1	j = 2	j = 3	j = 4	j = 5	
1	18	21	18	16	10	10
2		15	16	14	9	19
3		10		9	6	25
4	17	16	17	15	10	15
D	22	20	12	10	13	

Table 1

We omit all subscripts in the tables of this section. Therefore, a, u, v, w, D, c and x in our table represent a_i, u_i, v_j, w_j, D_j, c_{ij} and x_{ij} correspondingly. The j-th random demand is uniformly distributed in $[0, D_j]$ and $q_j^- = 6D_j$, $q_j^+ = 0$. Therefore,

$$\phi_j(w_j) = \begin{cases} 3D_j^2 - 6D_j w_j, & \text{if } w_j < 0, \\ 3(D_j - w_j)^2, & \text{if } w_j \in [0, D_j], \\ 0, & \text{if } w_j > D_j, \end{cases} \qquad (10.1)$$

and

$$\phi'_j(w_j) = \begin{cases} -6D_j, & \text{if } w_j < 0, \\ 6(w_j - D_j), & \text{if } w_j \in [0, D_j], \\ 0, & \text{if } w_j > D_j, \end{cases} \qquad (10.2)$$

for $j = 1,2,3,4,5$.

We start from a poor starting point to illustrate our algorithms. Our starting point is $w = (9, 12, 12, 12, 12, 12)$. Solving $T(w)$, we get x_{ij} as in Table 2:

i	x						a
	j = 0	j = 1	j = 2	j = 3	j = 4	j = 5	
1	9					1	10
2				12		7	19
3			12		12	1	25
4		12				3	15
w	9	12	12	12	12	12	

Table 2

The objective value is 1228. In this table and the other x-tables in this section, we omit 0-entries to make the forest structure more obvious. The corresponding multiplier (u,v) and $u_i + v_j - c_{ij}$'s are as in Table 3:

i	u + v - c						u
	j= 0	j= 1	j= 2	j= 3	j= 4	j= 5	
1	0	-1	-7	-1	-3	0	0
2	-1		-2	0	-2	0	-1
3	-4		0		0	0	-4
4	0	0	-2	0	-2	0	0
v	0	17	14	17	13	10	

Table 3

According to (2.6), (4.3) and (10.2), we have

$$w_j = D_j - v_j / 6, \quad j = 1,2,3,4,5. \qquad (10.3)$$

We get an optimal solution of (4.1) on the current forest in Table 4.

i	x						a
	j= 0	j= 1	j= 2	j= 3	j= 4	j= 5	
1	23/6					37/6	10
2				55/6		59/6	19
3			53/3		47/6	-1/2	25
4		115/6				-25/6	15
w	23/6	115/6	53/3	55/6	47/6	34/3	

Table 4

We see that after cutting cell (4,5) leaves the forest and that the forest split to two trees: {(1,0), (1,5), (2,3), (2,5), (3,2), (3,4), (3,5)} and {(4,1)}. It is easy to see that the values of u_i and v_j in the first tree are the same as in Table 3. Therefore, the value of an optimal solution of (4.3) on the first tree is also the same as in Table 4. Therefore, the second cutting will take out cell (3,5). Now the forest has three component trees: {(1,0), (1,5), (2,3), (2,5)}, {(3,2), (3,4)} and {(4,1)}. In the first tree, the values of u, v, x and w are still unchanged. The third tree is a one-cell tree. We simply get $w_1 = x_{41} = a_4 = 15$. From (4.4) and (10.2), we know that $v_1 = 6(D_1 - w_1) = 42$. For the second tree, this is an example of the case B of Section 2.4. Let $k = 2$. We have $v_2^0 = 0$ and $v_4^0 = -1$. Therefore, we have

$$v_4 = v_2 - 1. \tag{10.4}$$

Similarly to (10.3), we have

$$\begin{aligned} w_2 &= D_2 - v_2/6 = 20 - v_2/6, \\ w_4 &= D_4 - v_4/6 = 61/6 - v_2/6. \end{aligned} \tag{10.5}$$

However, we have $w_2 + w_4 = a_3 = 25$. This gives us an equation of v_2:

$$P(v_2) := 181/6 - v_2/3 = 25.$$

We get $v_2 = 15.5$. From (10.4), we have $v_4 = 14.5$. From (10.5), we have $w_2 = 209/12$, $w_4 = 91/12$. These are also the values of x_{32} and x_{34}. Therefore, we get a base forest triple. The data of this forest triple are given in Table 5 and Table 6. A point here is that we haven't calculated two intermediate forest triples at all. The exact data in Table 4 are not so important either.

i	x						a
	j = 0	j = 1	j = 2	j = 3	j = 4	j = 5	
1	17/2					3/2	10
2				55/6		59/6	19
3			209/12		91/12		25
4		15					15
w	17/2	15	209/12	55/6	91/12	34/3	

Table 5

i	u + v - c						u
	j = 0	j = 1	j = 2	j = 3	j = 4	j = 5	
1	0	26	-5.5	-1	-1.5	0	0
2	-1		-1.5	0	-0.5	0	-1
3	-5.5		0		0	-1.5	-5.5
4	25	0	-25.5	-25	-25.5	-25	25
v	0	42	15.5	17	14.5	10	

Table 6

The objective value is 964.541666667. From Table 6, we see that this is not only a base forest triple, but also a total forest triple. The only place of violating (2.6) is cell (1,1). We do a connecting. We can take $j_1 = 0$ and $\delta = 1$. This leads to decreasing the objective value to 943.541666667 and a forest consisting of trees {(1,0), (1,1), (1,5), (2,3), (2,5), (4,1)} and

{(3,2), (3,4)}. Minimizing on this forest, we get new (x,w) and (u,v) in Table 7 and Table 8.

i	x						a
	j = 0	j = 1	j = 2	j = 3	j = 4	j = 5	
1	9/2	4				3/2	10
2				55/6		59/6	19
3			209/12		91/12		25
4		15					15
w	9/2	19	209/12	55/6	91/12	34/3	

Table 7

i	u + v - c						u
	j = 0	j = 1	j = 2	j = 3	j = 4	j = 5	
1	0	0	-5.5	-1	-1.5	0	0
2	-1		-1.5	0	-0.5	0	-1
3	-5.5		0		0	-1.5	-5.5
4	-1	0	-1.5	-1	-1.5	-1	-1
v	0	18	15.5	17	14.5	10	

Table 8

From Table 8, we know an optimal forest triple is at hand. The optimal objective value is 916.541666667. We see that the optimal solution is determined by solving (4.1) on the optimal forest. It will not be affected by the intermediate iteration error and data out of the optimal forest. In all, we have used three cuttings and one connecting.

We have calculated some medium and large examples in DEC VAX-11/780. We assume that each random demand is piecewise uniformly distributed in five successive intervals and that these five intervals, the probabilities in these five intervals and the penalty coefficient are distinct between different columns. We count getting a new base forest triple as one iteration. The following are the numbers of iterations for different problems with different sizes. The sparse extent is the approximate percentage of the unavailable cells. We use the mid point of the five intervals as the starting supply value for each demand node.

m	n	sparse extent	number of problems calculated	number of iterations for convergence
6-9	7-13	10%	10	3-9
9-13	14-18	10%	10	6-12
26-28	36	50%	4	21-33
29-34	44	50%	4	33-36

Table 9

Acknowledgment

I am thankful to Professor S. M. Robinson for his helpful suggestions and his careful reading of the manuscript. I am thankful to Professor R. R. Meyer for discussion of this problem. I am also thankful to the referee for his comments.

References

[1] V. Balachandran, "Generalized transportation networks with stochastic demands: an operator theoretic approach", Networks 9(1979)169-184.

[2] M. Bazaraa and J. Jarvis, Linear programming and network flows (John Wiley & Sons, New York, 1977).

[3] L. Cooper and L.J. LeBlanc, "Stochastic transportation problems and other network related convex problems", Naval Research Logistics Quarterly 24(1977)327-336.

[4] L. Cooper, "The stochastic transportation-location problem", Computers & Mathematics with Applications 4(1978)265-275.

[5] G.B. Dantzig, Linear programming and extensions (Princeton University Press, Princeton, NJ, 1963).

[6] S. Elmaghraby, "Allocation under uncertainty when the demand has a continuous distribution function", Management Science 6(1960)270-294.

[7] A.R. Ferguson and G.B. Dantzig, "The allocation of aircraft to routes", Management Science 3(1957)45-73.

[8] L.R. Ford and D.R. Fulkerson, Flows in networks (Princeton University Press, Princeton, NJ, 1962).

[9] P.E. Gill, W. Murray and M.H. Wright, Practical optimization (Academic Press, New York, 1980).

[10] P. Kall, Stochastic linear programming (Springer-Verlag, Berlin, 1976).

[11] S.C. Parikh, "Lecture notes on stochastic programming", Department of Industrial Engineering, University of California (Berkeley, CA, 1968).

[12] L. Qi, "Base set strategy for solving linearly constrained convex programs", Technical Report 505, Computer Sciences Department, University of Wisconsin-Madison (Madison, WI, 1983).

[13] L. Qi, "An alternating method to solve stochastic programming with simple recourse", Technical Report 515, Computer Sciences Department, University of Wisconsin-Madison (Madison, WI, 1983).

[14] L. Qi, "Total forest iteration method for stochastic transportation problem", forthcoming as Technical Report, Computer Sciences Department, University of Wisconsin-Madison (Madison, WI, 1984).

[15] L. Qi, "A-forest iteration method for stochastic generalized transportation problem", forthcoming as Technical Report, Computer Sciences Department, University of Wisconsin-Madison (Madison, WI, 1984).

[16] S.M. Robinson, "Convex Programming", unpublished notes, University of Wisconsin-Madison (Madison, WI,1981).

[17] R.T. Rockafellar, Convex analysis (Princeton University Press, Princeton, NJ, 1981).

[18] R.T. Rockafellar, "Monotropic programming: descent algorithms and duality", in: O.L. Mangasarian, R.R. Meyer and S.M. Robinson, eds., Nonlinear programming 4 (Academic Press, New York, 1981)pp.327-366.

[19] W. Szwarc, "The transportation problem with stochastic demands", Management Science 11(1964)33-50.

[20] S.W. Wallace, "Decomposing the requirement space of a transportation problem into polyhedral cones", manuscript, Chr. Michelsen Institute (Fantoft, Norway, 1984).

[21] R.J-B. Wets, "Solving stochastic programs with simple recourse, I", Department of Mathematics, University of Kentucky (Lexington, KY, 1974).

[22] R.J-B. Wets, "Solving stochastic programs with simple recourse", Stochastics 10(1983)219-242.

[23] A.C. Williams, "A stochastic transportation problem", Operations Research 11(1963)759-770.

[24] D. Wilson, "On a priori bounded model for transportation problem with stochastic demand and integer solutions", AIEE Transaction 4(1972)186-193.

[25] D. Wilson, "Tight bounds for stochastic transportation problems", AIEE Transaction 5(1973)180-185.

[26] D. Wilson, "A mean cost approximation for transportation problems with stochastic demand", Naval Research Logistics Quarterly 22(1975)181-187.

[27] W.T. Ziemba, "Computational algorithms for convex stochastic programs with simple recourse", Operations Research 18(1970)414-431.