Mathematical Programming 12 (1977) 372–391.
North-Holland Publishing Company

COST OPERATOR ALGORITHMS FOR THE TRANSPORTATION PROBLEM*

V. SRINIVASAN
Stanford University, Stanford, CA, U.S.A.

G.L. THOMPSON
Carnegie–Mellon University, Pittsburgh, PA, U.S.A.

Received 10 November 1974
Revised manuscript received 22 December 1976

A primal transportation algorithm is devised via post-optimization on the costs of a modified problem. The procedure involves altering the costs corresponding to the basic cells of the initial (primal feasible) solution so that it is dual feasible as well. The altered costs are then successively restored to their true values with appropriate changes in the "optimal" solution by the application of cell or area cost operators discussed elsewhere. The cell cost operator algorithm converges to optimum within $(2T-1)$ steps for primal nondegenerate transportation problems and $[(2T+1)\cdot\min(m,n)]-1$ steps for primal degenerate transportation problems, where T is the sum of the (integer) warehouse availabilities (also the sum of the (integer) market requirements) and m and n denote the number of warehouses and markets respectively. For the area cost operator algorithm the corresponding bounds on the number of steps are T and $(T+1)\cdot\min(m,n)$ respectively.

Key words: Transportation Problem, Networks, Cost Operator Algorithm, Post-optimizing, Primal Network Algorithm.

1. Introduction

One of the interesting results concerning the Ford–Fulkerson method [5, pp. 93–101] for solving transportation problems is the fact that fairly tight upper bounds on the number of steps the algorithm takes to reach an optimum can be stated. The Balinski–Gomory primal algorithm [1] which is "dual" to the Ford–Fulkerson method also has approximately the same upper bound on the number of steps. In contrast, for primal *basic* methods such as the stepping-stone [2] or MODI [3] methods, no tight bounds (other than the maximum number of feasible bases) have so far been stated.

In the present paper we shall describe two new primal basic methods – the *cell* and *area* cost operator algorithms for solving transportation problems. They are based on our previous work on an operator theory [8] of parametric program-

*This report was prepared as part of the activities of the Management Sciences Research Group, Carnegie–Mellon University, under Contract N00014-67-A-0314-0007 NR 047-048 with the U.S. Office of Naval Research.

ming for transportation problems. These methods begin by finding a primal basic feasible starting solution, then altering the costs corresponding to the basic cells so that the initial solution is dual feasible, and hence optimal, for the altered problem. The altered costs are then restored to their original values (with appropriate changes in the "optimum" solution) by the application of cell or area cost operators.

Besides providing new algorithms for transportation problems, it turns out that very strong bounds can be stated for the number of steps needed to obtain an optimum solution. For primal nondegenerate transportation problems, the upper bound on the number of steps for the cell cost operator algorithm is much stronger (by a factor of approximately $\min(m, n)/2$, where m and n are the number of warehouses and markets respectively) than those for the Ford–Fulkerson and Balinski–Gomory algorithms. (The typical step for each of these three algorithms involves approximately the same number of elementary operations.) However, for primal degenerate problems the cell cost operator algorithm is slightly weaker (by a factor of approximately 2) compared to these other two methods. The bounds for the area cost operator method are approximately one half of the bounds for the cell cost operator method. Since the basic step of the area cost operator algorithm is more involved than the cell cost operator algorithm, the area operator bounds are not necessarily stronger than the cell operator bounds.

The cost operator algorithms are, to the best of our knowledge, the first primal basic algorithms for the transportation problem to have polynomial bounds. The reason we are restricting our attention to primal *basic* methods rather than other primal methods (such as [1]) is that the former involve less computational storage requirements (the values for the primal variables need to be stored only for basic variables, the others are necessarily zero) and the obtained solutions are more readily suited for parametric programming calculations [8]. But, basic methods are generally poorer in terms of handling degeneracy. The cell cost operator algorithm, however, does not have such problems with degeneracy. In fact, unlike the MODI method, it can be shown to converge without perturbation.

The superiority of the cost operator method compared to the MODI method in terms of upper bounds on the number of calculations should not be confused with computational superiority in terms of mean computation times. In fact, our results in Section 4 show that the MODI method is better than the cell cost operator method in terms of mean computation times.

The rest of the paper is organized as follows. In Section 2 some preliminary notation and definitions are given. A description of the cell cost operator algorithm together with the solution of a small example is given in Section 3, computational results are given in Section 4, the convergence proof in Section 5 and the derivation of the bounds for the cell operator method are given in Section 6. A description of the area cost operator algorithm together with the solution of the same example is given in Section 7 and the area cost operator bounds are derived in Section 8.

For expositional ease, the algorithms provided in this paper apply only to

uncapacitated transportation problems. These algorithms can be extended to the capacitated case by referring to [8]. Although this paper is based on our earlier work [8] it is largely self-contained (except for some of the proofs) and can be read independently.

2. Notation and definitions

Consider an $m \times n$ transportation problem P defined as follows:

$$\text{Minimize} \quad \sum_{i\in I}\sum_{j\in J} c_{ij}x_{ij} = Z,$$

$$\text{subject to} \quad \sum_{j\in J} x_{ij} = a_i \quad \text{for } i \in I = \{1, 2, \ldots, m\},$$

$$\sum_{i\in I} x_{ij} = b_j \quad \text{for } j \in J = \{1, 2, \ldots, n\},$$

$$x_{ij} \geq 0 \quad \text{for } i \in I \text{ and } j \in J.$$

We assume without loss of generality (see [2, 3, 7, 8]):

(i) $\Sigma_{i\in I} a_i = \Sigma_{j\in J} b_j = T$

(ii) $m \leq n$

(iii) a_i and b_j are positive integers for $i \in I$, $j \in J$.

Let us define

$$s = \operatorname*{Min}_{I_1\subseteq I, J_1\subseteq J} \left| \sum_{i\in I_1} a_i - \sum_{j\in J_1} b_j \right| \tag{1}$$

where at least one of I_1 or J_1 is a *proper* subset of I or J, respectively. If the transportation problem P is *primal nondegenerate* [8] then $s \geq 1$; if it is *primal degenerate* then $s = 0$. But by adding $1/m$ to every supply a_i and 1 to one of the demands b_j we can make $s \geq 1/m > 0$. The latter is one of the standard degeneracy prevention techniques for transportation problems and will be considered in more detail in Section 6.

The dual problem D, to problem P is:

$$\text{Maximize} \quad \sum_{i\in I} a_i u_i + \sum_{j\in J} b_j v_j,$$

$$\text{subject to} \quad u_i + v_j \leq c_{ij} \quad \text{for } i \in I \text{ and } j \in J$$

where u_i and v_j are the dual variables associated with rows and columns of the cost matrix.

Next we define p_i and q_j by

$$p_i = \operatorname*{Min}_{j\in J} c_{ij} \quad \text{for } i \in I, \tag{2}$$

$$q_j = \operatorname*{Min}_{i\in I} (c_{ij} - p_i) \quad \text{for } j \in J. \tag{3}$$

Given these an obvious lower bound L to the optimum objective function Z is

given by

$$L = \sum_{i \in I} p_i a_i + \sum_{j \in J} q_j b_j \tag{4}$$

since the objective function Z can be rewritten as

$$Z = \sum_{i \in I} \sum_{j \in J} (c_{ij} - p_i - q_j) x_{ij} + \sum_{i \in I} p_i a_i + \sum_{j \in J} q_j b_j$$

and $(c_{ij} - p_i - q_j) \geq 0$ from (2) and (3).

Let $X = \{x_{ij}\}$ be any primal basic feasible solution with basis B to problem P. Define

$$A = \{(i, j) \mid (i, j) \in B \text{ and } c_{ij} - p_i - q_j > 0\}. \tag{5}$$

If $A = \emptyset$, then X is optimal for problem P since then $Z = L$ and L was a lower bound for Z.

Assume $A \neq \emptyset$. Then define the *modified transportation problem* $\hat{\mathrm{P}}$ with costs

$$\begin{cases} \hat{c}_{ij} = c_{ij} & \text{for } (i, j) \notin A, \\ \hat{c}_{ij} = (p_i + q_j) & \text{for } (i, j) \in A \end{cases}$$

and the same rim conditions as for problem P. The solution X is primal feasible and $u_i = p_i$, $v_j = q_j$ is dual feasible, hence these are optimal primal and dual feasible solutions for problem $\hat{\mathrm{P}}$.

In order to describe the algorithms we introduce some definitions from [8]. By a *cell* (i, j) we mean an ordered index pair with row $i \in I$ and column $j \in J$. A *line* refers to a row or column.

Definition 1. Let Ω be a set of cells (i, j). Line g is said to be *connected* to line h in Ω if and only if there exists a *path* S of distinct cells in Ω,

$$S = \{(i_1, j_1), (i_2, j_2), \ldots, (i_k, j_k)\}$$

such that

(a) (i_1, j_1) is the only cell of S in line g and (i_k, j_k) is the only cell of S in line h;
(b) for each t in $1 < t < k$ either
  (i) $i_t = i_{t-1}$ and $j_t = j_{t+1}$, or
  (ii) $j_t = j_{t-1}$ and $i_t = i_{t+1}$.

A basis B for the problem P (or $\hat{\mathrm{P}}$) consists of a set of $m + n - 1$ cells such that every line is (uniquely) connected to every other line in B.

Definition 2. Let B be a basis, $(p, q) \in B$ and $\Omega = B - \{(p, q)\}$. Then we define the sets

$I_R = \{p\} \cup \{i \in I \mid i \text{ is connected to row } p \text{ in } \Omega\}$,
$I_C = I - I_R$,
$J_C = \{q\} \cup \{j \in J \mid j \text{ is connected to column } q \text{ in } \Omega\}$,
$J_R = J - J_C$.

The "scanning routine" described in Remark 9 on p. 218 of [8] can be used to easily find these sets.

Definition 3. A (plus) *cell cost operator* [8] δC_{pq}^{+} transforms the optimal solution ($\{x_{ij}\}$, $\{u_i\}$ and $\{v_j\}$) of a problem P to that for a transformed problem P^{+} whose *data* (i.e., $\{c_{ij}\}$, $\{a_j\}$ and $\{b_j\}$) are the same as those of P except for the single cell (p, q) whose cost is changed from c_{pq} to $c_{pq} + \delta$ ($\delta \geq 0$). The *area cost operator* δC_A [8] transforms the optimal solution of P to that for a transformed problem P^A whose data are the same as those of P except c_{ij} is changed to $c_{ij} + \delta\gamma_{ij}$ ($\delta \geq 0$) for all i and j.

We now use the operator theory of parametric programming developed in [8] to parametrically increase the costs $\hat{c}_{ij}$ for $(i, j) \in A$, at the same time making appropriate changes in the primal and dual feasibility, until they equal the original costs c_{ij} for problem P; at this point the current primal and dual solutions are optimal for P. We shall first discuss the *cell cost operator algorithm* in which the costs are changed one cell (i.e., index pair (i, j)) at a time, and then describe the *area cost operator algorithm* in which the costs are changed on several cells simultaneously.

3. The cell cost operator algorithm

The cell cost operator algorithm to be described below directly follows from the algorithms described in pp. 209–211, pp. 215–221 and pp. 240–248 of [8] for parametrically increasing (one by one) the costs $\hat{c}_{ij}$ to c_{ij} for $(i, j) \in A$.

(A1) *Cell cost operator algorithm*

(0) Set up initial tableau with costs c_{ij}, supplies a_i and demands b_j.

(1) Find $p_i = u_i$ and $q_j = v_j$ using equations (2) and (3). (This can be done in two passes through the cost matrix.) Compute L using eq. (4). Set $Z = L$.

(2) Find a primal basic feasible starting solution $X = \{x_{ij}\}$ with basis B by any technique. (Several starting techniques are given in [10].)

(3) Find the set A defined in eq. (5).

(4) For each (i, j) in A store the value of c_{ij} as c_{ij}^{*}; then replace c_{ij} by $p_i + q_j = u_i + v_j$.

(5) If $A = \emptyset$ the optimum solution has been found; stop. Else choose any cell $(p, q) \in A$. (In Section 4 the rules for choosing the next cell are discussed.)

(6) Find the sets I_R, I_C, J_R and J_C corresponding to cell (p, q) (Definition 2).

(7) Determine the maximum extent μ^{+} of the basis preserving cell cost operator from the formula

$$\mu^{+} = \underset{(i,j)\in\Psi}{\text{Minimum}} (c_{ij} - u_i - v_j)$$

where $\Psi = [I_R \times J_C] - \{(p, q)\}$.

Record the *entering cell* (e, f) at which this minimum is taken on.

(8) Let $\delta = \text{Min}(\mu^{+}, c_{pq}^{*} - c_{pq})$. If $\delta = 0$ go to (11). Else go to (9).

(9) Change the dual variables as follows: Replace u_i by $u_i + \delta$ for $i \in I_R$. Replace v_j by $v_j - \delta$ for $j \in J_R$.

(10) Replace c_{pq} by $c_{pq} + \delta$ and Z by $Z + \delta x_{pq}$. If $c_{pq} = c^*_{pq}$ remove (p, q) from A and go to (5). Else go to (11).

(11) Bring (e, f) into the basis, determining the *leaving cell* (r, s) (i.e., determine the 'giver' cell in the cycle created when (e, f) is added to B for which x_{ij} is a minimum). Whenever possible select (p, q) as (r, s). Change the shipping amounts x_{ij} on the cycle in $B + \{(e, f)\}$. Replace B by $B - \{(r, s)\} + \{(e, f)\}$.

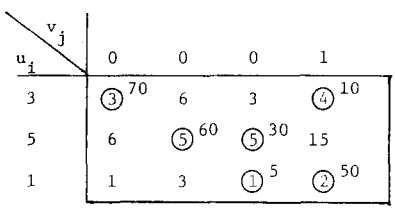
(12) If $(r, s) \in A$, replace c_{rs} by c^*_{rs} and remove (r, s) from A.
If $(r, s) = (p, q)$ go to (5). Else go to (6).

Example 1. Fig. 1 shows the tableau of a 3×4 nondegenerate transportation problem with the values of p_i and q_j marked on the left and top. Since

p_i \ q_j	0	0	0	1	a_i
3	3	6	3	4	80
5	6	5	11	15	90
1	1	3	10	5	55
b_j	70	60	35	60	

s=5, L=(3x80)+(5x90)+(1x55)+(1x60)=805

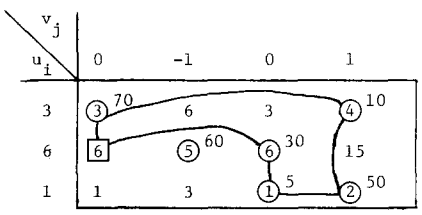
Figure 1



A = {(2,3), (3,3), (3,4)}

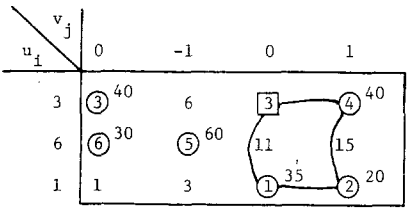
$c^*_{23} = 11$, $c^*_{33} = 10$, $c^*_{34} = 5$, Z = 805

Figure 2



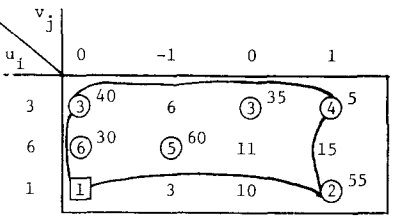
Z = 805+(1x30) = 835

Figure 3



A = {(3,3), 3,4)}

Figure 4



A = {(3,4)}

Figure 5

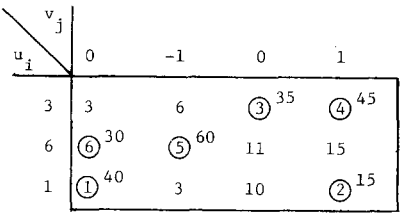
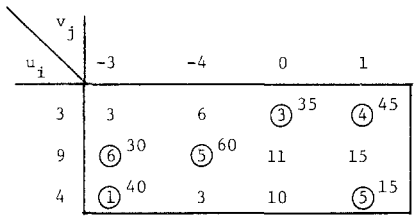


Figure 6



Z = 835+(3x15) = 880. Optimum.

Figure 7

$60-55=5$ it is easy to see that $s=5$. Also $Z=L=805$ by the calculation shown. Fig. 2 shows the modified problem together with a primal basic feasible starting solution obtained by using the Row Minimum Rule [10], and the set A. In Fig. 2 the costs in A have been changed as indicated in operation (4) of the algorithm. We now indicate the solution of the problem using the algorithm.

Operation number	Result
(5)	$A \neq \emptyset$. Choose $(p, q) = (2, 3)$.
(6)	$I_R = \{2\}$, $J_C = \{1, 3, 4\}$.
(7)	$\mu^+ = 6-5-0 = 1$, $(e, f) = (2, 1)$.
(8)	$\delta = 1$.
(9), (10)	$u_2 = 6$, $v_2 = -1$, $c_{23} = 6$. $Z = 805 + (1 \times 30) = 835$ (Fig. 3).
(11), (12)	Shipping amounts altered as shown in Fig. 4. Since $(r, s) = (2, 3)$ we remove $(2, 3)$ from A and set $c_{23} = 11$.
(5)	Choose $(p, q) = (3, 3)$.
(6)	$I_R = \{1, 2, 3\}$, $J_C = \{3\}$.
(7)	$\mu^+ = 0$, $(e, f) = (1, 3)$.
(8)	$\delta = 0$.
(11), (12)	Shipping amounts altered as shown in Fig. 5. Since $(r, s) = (3, 3)$ we remove $(3, 3)$ from A and set $c_{33} = 10$.
(5)	$(p, q) = (3, 4)$.
(6)	$I_R = \{3\}$, $J_C = \{1, 2, 3, 4\}$.
(7)	$\mu^+ = 0$, $(e, f) = (3, 1)$.
(8)	$\delta = 0$.
(11)	Shipping amounts shown in Fig. 6. $(r, s) = (1, 1)$.
(6)	$I_R = \{2, 3\}$, $J_C = \{3, 4\}$.
(7)	$\mu^+ = 5$, $(e, f) = (2, 3)$.
(8)	$\delta = \text{Min}\{5, 3\} = 3$.
(9)	$u_2 = 9$, $u_3 = 4$, $v_1 = -3$, $v_2 = -4$.
(10)	$c_{34} = 5$, $Z = 835 + (3 \times 15) = 880$, $A = \emptyset$.
(5)	Optimum solution shown in Fig. 7.

4. Computational experience

The algorithm of the previous section has been programmed in FORTRAN V and extensive tests have been run on the UNIVAC 1108 computer.

In programming the algorithm many experiments were carried out with the choice in operation (5) of the algorithm for the next cell in the set A. Of all the experiments the only conclusive experimental result obtained was that once cell (p, q) is chosen then the same choice should be made until (p, q) has been driven out of A. (Any rule that we tried that involved switching to another cell invariably gave worse results. For the algorithm to converge, it is essential that a switch to another cell be made only after a strictly positive increase in the cost of the present cell (p, q).) The statement of the algorithm in Section 3 already uses this result.

For the choice of the next (p, q) in A the following rules were tested:

(1) Select (p, q) in A such that x_{pq} is maximum.

(2) Select (p, q) in A such that x_{pq} is minimum.

(3) Select, if possible, a cell (p, q) in A such that (p, q) is the only basis cell in its row or column. If no such cell exists, use Rule (1).

(4) Select, if possible, a cell (p, q) in A such that (p, q) is the only basis cell in its row or column. If no such cell exists, use Rule (2).

Rule (1) is a steepest ascent rule in the sense that when the cost of cell (p, q) is increased by δ the objective function value increases by δx_{pq} (Theorem 5a, [8]). Consequently this rule could be expected to require fewer steps in reaching the optimum objective function value for the original problem. Rule (2), by choosing x_{pq} as low as possible, increases the probability that (p, q) would be chosen as (r, s) in operation (11) of the algorithm thus driving (p, q) out of A as quickly as possible. Rules (3) and (4) use the fact that if (p, q) is the only basis cell in row p (column q) then the area $[I_R \times J_C]$ is simply all the cells in row p (column q) (Remark 8, [8]). Thus the computational effort in identifying the sets I_R, I_C, J_R, and J_C and determining μ^+ (operations (6) and (7) of the algorithm) is considerably reduced.

In testing each of the above rules, once a cell was selected it was retained until it had been driven out of A. The result of extensive tests with these four rules was that there was no significant difference in their performance for transportation problems. However, for assignment problems, Rule (2) performed significantly better than the other rules. This happens because for assignment problems with no perturbation the amount x_{ij} is 0 or 1. (As will be seen in Section 5, the cell cost operator algorithm converges even for degenerate problems, i.e., when the amount s defined in eq. (1) is equal to zero.) Consequently, if a (p, q) is chosen with $x_{pq} = 0$ it is easily shown that in operation (11) of the algorithm, the cell (p, q) will be chosen as (r, s) so that (p, q) leaves the set A in one step. For these reasons our final code made use of Rule (2) exclusively for both transportation and assignment problems.

The mean solution times for the cell cost operator algorithm were compared with the authors' 1971 primal basic transportation code [10] using the MODI method [3] which is currently one of the fastest existing codes for solving such problems. The results are shown in Table 1 where mean solution times and number of pivots are exhibited for randomly generated problems ($m = n = 100$) with costs and *rim conditions* (i.e., a_i and b_j) chosen as integers distributed uniformly in the range 0–100. (The effects of changes in these ranges are

Table 1[a]
Computational performance of the cell cost operator algorithm

(A) 100×100 transportation problems

Probability of infeasibility		$p = 0$	$p = 0.2$	$p = 0.4$	$p = 0.6$	$p = 0.8$
Primal algorithm [10]	Mean pivots	386	395	393	436	442
	Mean solution time (sec.)	2.093	2.121	2.033	2.149	2.185
Cell cost operator algorithm	Mean pivots	337	356	396	398	406
	Mean solution time (sec.)	7.654	7.715	8.635	8.622	8.805

(All costs and rims chosen to be integers uniformly distributed between 0 and 100)

(B) 100×100 assignment problems

Probability of infeasibility		$p = 0$	$p = 0.2$	$p = 0.4$	$p = 0.6$	$p = 0.8$
Primal algorithm [10]	Mean pivots	612	609	651	618	658
	Mean solution time (sec.)	2.090	2.093	2.189	2.057	2.165
Cell cost operator algorithm	Mean pivots	177	171	169	180	189
	Mean solution time (sec.)	3.681	3.674	3.541	3.825	3.900

(All costs chosen to be integers uniformly distributed between 0 and 100)

[a] All computations were performed on UNIVAC 1108 (FORTRAN V); mean solution times in seconds are exclusive of input and output and are each based on runs of 21 randomly generated problems.

considered subsequently.) The term 'pivots' refers to the total number of basis changes needed to reach an optimum. The parameter p at the top of the table indicates the probability with which some of the cells are chosen to be infeasible, i.e., have infinite shipping cost. Note that the solution times increase only slightly as p increases. Note also that the cost operator algorithm requires somewhat fewer pivots than the primal algorithm for transportation problems and many fewer pivots for assignment problems. However, the cost operator algorithm took longer to solve both kinds of problems although it was still faster than the computational times for the Ford–Fulkerson algorithm [10, Section 3.1].

Table 2 shows the effect of changing the intervals in which the rim conditions were chosen. As in the primal algorithm (see [10], Table IV) there are only minor changes in solution time.

Table 3 shows the effect on solution times for assignment problems of changing the range in which the costs were chosen. Note that as the range for the costs increases from 0–10 to 0–100 the solution time increases markedly, which was also observed with our primal algorithm (see [10], Table IV). In [10] we called this the Minimum Cost Effect and is explained in more detail there.

Table 2[a]
Effect of parameter changes on the performance of cell cost operator algorithm for transportation problems

Mean solution times (sec.) for 100×100 transportation problems

Probability of infeasibility		$p = 0$	$p = 0.2$	$p = 0.4$	$p = 0.6$	$p = 0.8$
Rims chosen in the interval	1–10	6.623	6.767	7.290	7.857	8.284
	1–100	7.654	7.715	8.635	8.622	8.805
	1–1,000	7.880	7.841	7.351	8.526	9.007
	1–10,000	7.902	7.978	8.417	8.821	8.945

[a]All computations were performed on UNIVAC 1108 (FORTRAN V); mean solution times in seconds are exclusive of input and output and are each based on 21 randomly generated problems. The costs were chosen as integers uniformly distributed between 0 and 100.

Table 3[a]
Effect of parameter changes on the performance of cell cost operator algorithm for assignment problems

Mean solution times (sec.) for 100×100 assignment problems ($p = 0$)

Costs chosen in interval	0–10	0–100	0–1,000	0–10,000
Mean solution time (sec.)	1.355	3.681	4.278	4.081

[a] All computations were performed on UNIVAC 1108 (FORTRAN V); mean solution times in seconds are exclusive of input and output and are each based on 21 randomly generated problems.

Essentially what happens is that when the costs are chosen in the range 0–10, the starting solution is very likely to be optimal or nearly optimal so that the total solution time is only a little longer than the time to get the initial solution. For larger cost ranges such as 0–100, 0–1000, or 0–10,000 this is no longer true for 100×100 problems, and many more pivots are needed to obtain the optimal solution. As in our code for primal algorithm [10, Table IV], the mean computation time for the cell cost operator algorithm increases rapidly at first and then tends to saturate as the range for c_{ij} is increased.

5. Convergence of the cell cost operator algorithm

In the case that problem P is nondegenerate, i.e., the quantity s defined by (1) is strictly positive, the convergence of the cell cost operator is assured by Theorem 14 of [8]. Let us define a *step* of the cell cost operator algorithm A1 as

the computational loop consisting of operations (6) through (12) of that algorithm. Then Lemma 6(a) in [8] shows that if $x_{pq}^{(k)}$ and $x_{pq}^{(k+1)}$ denote the shipments by cell (p, q) at steps k and $k+1$ of the algorithm then

$$x_{pq}^{(k+1)} = x_{pq}^{(k)} - \Delta_{pq}^{(k)} \tag{6}$$

where $\Delta_{pq}^{(k)} \geq s > 0$ and s is defined by (1). Hence at most $x_{pq}^{(1)}/s$ steps will suffice to remove (p, q) from A.

However, even if problem P is degenerate the algorithm will converge *without* any perturbation. We prove this by using an elegant proof technique due to W. Szwarc [11] for proving a related result in his time transportation algorithm.

Theorem 1. *The cell cost operator algorithm converges in a finite number of steps.*

Proof. We already know the algorithm converges if P is nondegenerate. Hence assume P is degenerate ($s = 0$) and that Algorithm (A1) cycles. In other words, assume that in applying the operator the algorithm generates a finite sequence of bases: $B_1, B_2, \ldots, B_t = B_1$.

Clearly if this can happen the method will cycle and conversely. We know that $x_{pq}^{(1)} > 0$ or else using operation (11) of (A1) we would remove (p, q) in one step. Also because $x_{pq}^{(h+1)} \leq x_{pq}^{(h)}$ for every h and since $x_{pq}^{(t)} = x_{pq}^{(1)}$ it follows from (6) that $\Delta_{pq}^{(h)} = 0$ for $h = 1, 2, \ldots, t$ and hence $x_{ij}^{(h)} = x_{ij}^{(1)}$ for every (i, j) and $h = 1, 2, \ldots, t$.

Let us now consider the set

$$\Omega = B_1 \cap B_2 \cap \cdots \cap B_t.$$

If we can show that $B_h = \Omega$ for $h = 1, 2, \ldots, t$, then we would have proved that all bases are the same, (i.e., $t = 1$) and hence no cycling is possible. To prove this, let us consider one such B_h. There is a unique path in B_h from (p, q) to any other cell (i, j) in B_h. By the *distance* of (i, j) we shall mean the number of cells on this path, including (p, q) and (i, j).

We now give an inductive argument that $B_h = \Omega$. We know $(p, q) \in \Omega$. Assume we have shown that k cells of B_h are in Ω; we must show that $k+1$ cells of B_h are in Ω. Let (i, j) be any cell in B_h not yet shown to be in Ω but such that (i, j) has a line in common with a cell in Ω. (This is always possible since the cells in B_h are connected.) If $x_{ij} > 0$, then (i, j) is in Ω because the cells with positive shipments stay the same and hence cannot leave the basis. If $x_{ij} = 0$ and the distance from (p, q) is even then (i, j) is a getter cell and cannot leave the basis so $(i, j) \in \Omega$. (Note that (p, q) is always a giver cell because of eq. (6).) Finally if $x_{ij} = 0$ and the distance from (p, q) is odd then if (i, j) should leave the basis as a giver it could never again enter the basis since it would remain a giver because its distance from (p, q) does not change. Thus in all cases $(i, j) \in \Omega$ completing the induction proof that $B_h = \Omega$.

Since the above proof holds for each $h = 1, 2, \ldots, t$ it follows that $B_h = \Omega$ for $h = 1, 2, \ldots, t$ and hence no cycling is possible. Since there are only a finite

number of bases and since no basis is repeated the algorithm will converge in a finite number of steps.

6. Theoretical bounds for the cell cost operator algorithm

Although the finite convergence of the cell cost operator algorithm has now been established for degenerate problems even without perturbation, it is necessary to consider the solution of the perturbed problem to derive bounds on the number of steps to reach the optimum.

Remark 1. Given a degenerate transportation problem P, we define the perturbed problem P′ to have the same costs $\{c_{ij}\}$ as P but with $a'_i = a_i + 1/m$ for $i \in I$. $b'_j = b_j$ for $1 \leq j < n$ and $b'_n = b_n + 1$. Then P′ is nondegenerate with $s \geq 1/m$ (Dantzig [3, p. 314, Problem 14] and Orden [6]). *If* P *is nondegenerate, however, we define* $P' = P$.

Remark 2. Let $\{x'_{ij}\}$ be a basic feasible solution to the perturbed problem with basis B. It is well known [7, 9] that $T = [I \cup J, B]$ is a tree with nodes being the $m + n$ rows and columns of the tableau and edges the $m + n - 1$ cells of B. A node of degree 1 of T corresponds to a row or column that contains a unique cell of B. Since T is a tree it has at least two nodes of degree 1 provided it has at least one edge (i.e., at least two nodes). We now define a *crossing out procedure* to express x'_{ij} in terms of the rims a'_i and b'_j. Suppose cell (i, j) is unique in row i; then make the following replacements.

Replace I by $I - \{i\}$,
Replace B by $B - \{(i, j)\}$,
Replace b'_j by $b'_j - a'_i$,

set $x'_{ij} = a'_i$ and cross out row i. Because B is assumed to be feasible $b'_j \geq a'_i$ so that the new problem with row i omitted is a transportation problem with one fewer row and a feasible basis. Similarly, suppose cell (i, j) is unique in column j; then make the following replacements.

Replace J by $J - \{j\}$,
Replace B by $B - \{(i, j)\}$,
Replace a'_i by $a'_i - b'_j$,

set $x'_{ij} = b'_j$ and cross out column j. We again obtain a smaller problem with a feasible basis. Since we reduce B by one element on each step we will find the feasible solution $\{x'_{ij}\}$ in exactly $m + n - 1$ steps.

Remark 3. The preceding crossing out routine also makes each cell of B correspond to a unique row or column – the one that is crossed out at the time the cell is removed from B. Since there are $m + n$ rows and columns and $m + n - 1$ basis cells it is clear that exactly one row or column is left out of this correspondence. Because as noted above there *always* are at least two lines (rows and columns) that contain a unique basis cell we can *always select any row or column* we wish to be left out of this correspondence.

Theorem 2 below shows that any basic optimal solution to P′ also defines a basic optimal solution to P. (This proof is essential to justify substituting the degenerate problem P by the perturbed problem P′. We have not been able to find a proof for this important result in the literature.)

Theorem 2. *Let X' be a basic optimal solution to* P′ *with basis B. Then the solution X corresponding to basis B is optimal to the degenerate transportation problem* P.

Proof. As per Remark 3, we can select column n not to be crossed out in the above routine. It follows from the crossing out procedure (see also Simmonard [7], p. 242) that we can uniquely write

$$x'_{ij} = \delta_{ij}\left(\sum_{h\in I_{ij}} a'_h - \sum_{k\in J_{ij}} b'_k\right)$$

where $\delta_{ij} = \pm 1$, $I_{ij} \subseteq I$ and $J_{ij} \subseteq J - \{n\}$. Using the definitions for a'_i and b'_j in Remark 1 we have

$$x'_{ij} = \delta_{ij}\left(\sum_{h\in I_{ij}} a_h + \tfrac{1}{m}|I_{ij}| - \sum_{k\in J_{ij}} b_k\right) \tag{7}$$

where $|I_{ij}|$ is the number of elements in the set I_{ij}.

In the crossing out procedure of Remark 2 with basis B, if we had used a_i, b_j, and x_{ij} of problem P instead of a'_i, b'_j and x'_{ij} for problem P′, respectively, we would have gotten

$$x_{ij} = \delta_{ij}\left(\sum_{h\in I_{ij}} a_h - \sum_{k\in J_{ij}} b_k\right). \tag{8}$$

It is clear that the x_{ij}'s so obtained are integers and that they satisfy the first two sets of constraints of problem P. To show that $x_{ij} \geq 0$, assume the contrary, i.e., that $x_{ij} \leq -1$ for some i and j. Since the corresponding $x'_{ij} > 0$ (because basis B is feasible for P′ and P′ is nondegenerate), it follows from (7) and (8) that $|I_{ij}| > m$, which is a contradiction. Thus, given a feasible basis B for P′ the corresponding solution x_{ij} defined by (8) is feasible for P. Furthermore since the costs have not been changed in going from P to P′ any dual feasible basis to P′ is also dual feasible to P. Thus any optimal (i.e., primal and dual feasible) basis to P′ is also optimal to P. Consequently, to solve problem P we can equivalently solve problem P′.

In the results to be proved next *we assume that we solve the perturbed problem* P′ by Algorithm A1 and introduce the following definitions:

$$T' = \sum_{i\in I} a'_i = \sum_{j\in J} b'_j, \tag{9}$$

$$R' = \operatorname{Max}\left(\max_{i\in I} a'_i, \max_{j\in J} b'_j\right), \tag{10}$$

that is, T' is the total amount shipped and R' is the largest rim entry for the perturbed problem P'. From the definition of T for problem P and Remark 1, it follows that

$$T' = T \quad \text{if P is primal nondegenerate,} \tag{11}$$

$$T' = T + 1 \quad \text{if P is primal degenerate.} \tag{12}$$

Let A be the set of cells defined in step (3) of Algorithm (A1). For any cell $(p, q) \in A$, let $\hat{x}_{pq}$ be the value of x_{pq} *at the moment* (p, q) *is chosen in step* (5) *of the algorithm.* (Some $(p, q) \in A$ may never be chosen since they may leave A in step (12) of the algorithm (A1): we will set such $\hat{x}_{pq} = 0$.) *Thus, unlike* $\{x_{pq}\}$ *which refer to a single tableau,* $\{\hat{x}_{pq}\}$ *correspond to different tableaus for different* $(p, q) \in A$. (Note that $\Sigma_{j \in J | (i,j) \in A} \hat{x}_{ij}$ may exceed a'_i. In Example 1 considered earlier, $\hat{x}_{23} = 30$, $\hat{x}_{33} = 35$ and $\hat{x}_{34} = 55$ so that $\hat{x}_{33} + \hat{x}_{34} = 90 > a_3 = 55$.)

Lemma 1. *With the notation just described*

$$\sum_{(i,j) \in A} \hat{x}_{ij} \leq 2T' - R'. \tag{13}$$

Proof. Considering the initial basis B carry out the crossing out routine of Remark 2 but never crossing out the row or column containing the largest rim entry R' (if there is more than one row or column containing the maximum rim entry select any one). The correspondence of Remark 3 between rims and cells in B now gives for each (p, q) in B the constraint $\hat{x}_{pq} \leq a'_p$ or $\hat{x}_{pq} \leq b'_q$ depending on whether row p or column q corresponds to cell (p, q). From this it follows that

$$\sum_{(p,q) \in A} \hat{x}_{pq} \leq \sum_{(p,q) \in B} \hat{x}_{pq} \leq \sum_{p \in I} a'_p + \sum_{q \in J} b'_q - R' = 2T' - R'$$

since $A \subseteq B$.

Denoting as a *step* the operations (6)–(12) of the cell cost operator algorithm (Al) we now have the following result.

Lemma 2. *The number of steps of the cell cost operator algorithm* (A1) *is at most* $(2T' - R')/s$ *where* s *is given by* (1).

Proof. At each step of (A1) when (p, q) is the cell on which the cost operator is applied, the amount x'_{pq} shipped by cell (p, q) is reduced by at least s, see (6). Hence there can be at most $\hat{x}_{pq}/s$ steps before either (p, q) leaves A or its cost is restored to its true value. Consequently from Lemma 1 the maximum number of steps required for the algorithm is

$$\sum_{(p,q) \in A} \hat{x}_{pq}/s \leq (2T' - R')/s.$$

Theorem 3. *A nondegenerate transportation problem can be solved by algorithm* (A1) *in at most* $2T-1$ *steps. A degenerate problem can be solved in at most* $m(2T+1)-1$ *steps.*

Proof. Since $s \geq 1$ and $R' \geq 1$ for a nondegenerate problem the first result follows directly from Lemma 2 and eq. (11). By Remark 1, after perturbation a degenerate problem has $s \geq 1/m$ and $R' \geq 1+(1/m)$ so that from Lemma 2 the bound is $[2T'-1-(1/m)]m = m(2T'-1)-1$. Substituting for T' from eq. (12) the result follows.

Corollary. *Algorithm* (A1) *will solve an* $n \times n$ *assignment problem in at most* $n(2n+1)-1$ *steps.*

It is interesting to compare the bounds provided by Theorem 3 to the bounds on the number of labellings of the primal-dual method of Ford–Fulkerson [5, p. 100] which, using our notation, is given by $T+(T-1)(m+1)$ labellings for both degenerate and nondegenerate transportation problems. The first term in this expression bounds the number of labellings resulting in 'breakthrough'; the second, the number of labellings resulting in 'non-breakthrough'. The non-breakthrough labellings are roughly comparable to the steps of the cell cost operator algorithm (A1). (Both involve the search of a subset of cells $\{(i, j)\}$ (labelled rows $\times$ unlabelled columns in [5] which is similar to $I_R \times J_C$) to find the minimum reduced cost $(c_{ij}-u_i-v_j)$. Our algorithm requires the determination of a cycle which by using the method provided in [9] is roughly comparable to the flow-augmentation labelling required by the Ford–Fulkerson method.) Consequently if we disregard the first term in the Ford–Fulkerson bound, their method requires $(T-1)(m+1)$ comparable steps. Similarly the Balinski–Gomory [1] primal method requires $T \cdot m$ comparable steps. Thus the bound for the cell cost operator algorithm is much stronger (by a factor of approximately $m/2$) than the Ford–Fulkerson and Balinski–Gomory algorithms for nondegenerate transportation problems. It is slightly weaker (by a factor of approximately 2) than these other two methods for degenerate transportation problems. For most real transportation problems the assumption of primal nondegeneracy appears to be a reasonable one so that in most problems we would expect the cell cost operator bound to be better. It is to be pointed out, however, that while the assumption of primal nondegeneracy reduces the upper bound of the cell cost operator algorithm by a factor of m, it does not change the upper bound for the other two methods. We tried to see whether other realistic assumptions such as dual nondegeneracy would reduce the Ford–Fulkerson and Balinski–Gomory bounds. We have been unable to find any such assumption.

Edmonds and Karp [4] have provided a transportation algorithm which involves T flow augmentations. (Their "scaling" algorithm requires $n[2+\log_2(T/n)]$ flow augmentations, assuming $m \leq n$.) But each flow augmentation requires computing $(c_{ij}-u_i-v_j)$ for *all* the $(m \times n)$ cells and then solving a shortest path problem (from a common source to all other nodes) in a network for which all arc lengths are nonnegative. Thus their typical flow augmentation step is much more involved compared to a typical step of the cell cost operator

algorithm. However, for square $(n \times n)$ problems, the maximum number of elementary operations for their flow augmentation step is $O(n^2)$, the same as that for a step of the cell cost operator algorithm. Thus in terms of elementary operations, the Edmonds–Karp bound is much stronger (by an order of n) compared to the cell cost operator algorithm for degenerate problems but both algorithms involve the same order of computations (at the maximum) for the more realistic primal nondegenerate problems.

It is also interesting to compare the theoretical bounds obtained to the number of steps in the computational tests for the 100×100 transportation problems reported in Table 1. From this table we find the number of steps (pivots) tends to increase with the probability of infeasibility p. Considering $p = 0.8$, the average number of steps was 406 and the *maximum* over the 21 randomly generated problems for $p = 0.8$ was 521 (this last number is not reported in Table 1). Since the a_i and b_j were drawn randomly from $1, 2, \ldots, 100$, $T \simeq 100 \times (100/2) = 5{,}000$ so that assuming near nondegeneracy (i.e., x_{ij} for the basic cells is mostly nonzero), the bound on the number of steps $= 2T - 1 \simeq 10{,}000$ which is much greater than the maximum value of 521 obtained in the computations. One possible reason may be the good starting basis used in the computations (the computer program used the "modified row minimum starting Rule" [10] whereas the computation of the bound assumed nothing about how good the initial basis is – the set A could in the limit be the entire initial basis B). Consequently, the same 21 transportation problems of size 100×100 with $p = 0.8$ were solved again with a random primal feasible initial basis. The average number of steps increased to 477 while the maximum over the 21 problems increased to 662. Thus there is clearly a small effect of a good choice of an initial basis, but the theoretical bound is still much larger than the realized bounds. This is, however, to be expected since the computation of the bound corresponds to the worst case, which is most unlikely to happen in any actual problem.

7. Area cost operator algorithm

Here we modify the cell cost operator algorithm of Section 3 to become the area cost operator algorithm. It directly follows from the algorithms described in pp. 209–211, pp. 215–221 and pp. 240–248 of [8]. We will need to make use of the quantities u_i^*, v_j^* defined on p. 220 of [8].

(A2) *Area cost operator algorithm*

(0)–(4) Identical to corresponding steps in Algorithm (A1).

(5) If $A = \emptyset$ the optimum solution has been found; stop. Otherwise set $\gamma_{ij} = 1$ for $(i, j) \in A$ and 0 otherwise and go to (6).

(6) Use the current basis B to solve the equations $u_i^* + v_j^* = \gamma_{ij}$ for $(i, j) \in B$.

(7) Determine the maximum extent μ^A for the basis preserving area cost operator from the formula

$$\mu^A = \underset{(i,j)\in N}{\text{Minimum}} \frac{c_{ij} - u_i - v_j}{u_i^* + v_j^* - \gamma_{ij}}$$

where $N = \{(i, j) \mid (\gamma_{ij} - u_i^* - v_j^*) < 0\}$. If the set $N = \emptyset$ set $\mu^A = \infty$. Record the *entering cell* (e, f) at which this minimum is taken on. If $\mu^A = 0$ go to (11). Else go to (8).

(8) Let $\mu' = \text{Minimum}_{(i,j)\in A}(c_{ij}^* - c_{ij})$. Let $\delta = \text{Min}(\mu^A, \mu')$.

(9) Change the dual variables as follows: Replace u_i by $u_i + \delta u_i^*$ for $i \in I$. Replace v_j by $v_j + \delta v_j^*$ for $j \in J$.

(10) Replace c_{pq} by $c_{pq} + \delta$ for $(p, q) \in A$ and Z by $Z + \delta \Sigma_{(p,q)\in A} x_{pq}$. If $c_{pq} = c_{pq}^*$ for some $(p, q) \in A$ remove each such (p, q) from A and go to (5). Else go to (11).

(11) Bring (e, f) into the basis determining the *leaving cell* (r, s). Whenever possible select (r, s) in A. Change shipping amounts x_{ij} on the cycle $B + \{(e, f)\}$. Replace B by $B - \{(r, s)\} + \{(e, f)\}$.

(12) If $(r, s) \in A$, replace c_{rs} by c_{rs}^* and remove (r, s) from A. If A is changed, go to (5). Else go to (6).

Example 2. We now resolve the problem in Fig. 1 using the area cost operator method starting from the modified problem in Fig. 2 with $A = \{(2, 3), (3, 3), (3, 4)\}$.

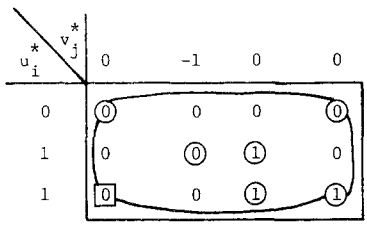
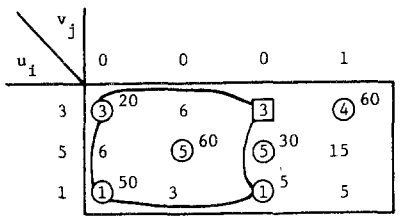


Figure 8



A = {(2,3), (3,3)}

Figure 9

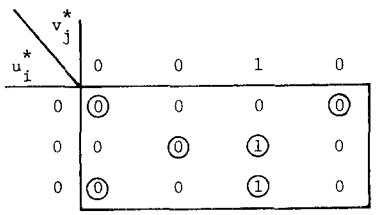
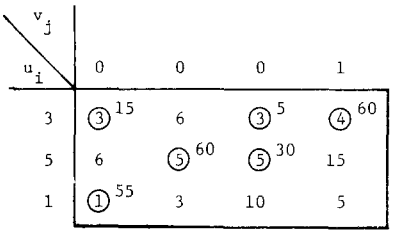


Figure 10



A = {(2,3)}

Figure 11

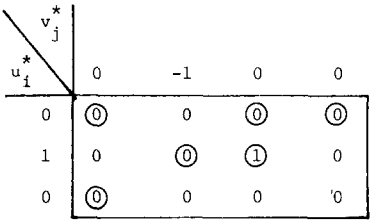
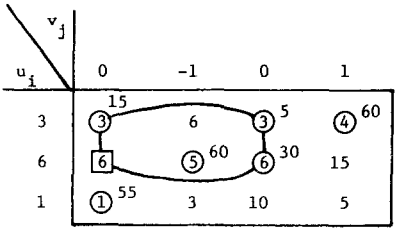


Figure 12



Z = 805+(1x30) = 835

Figure 13

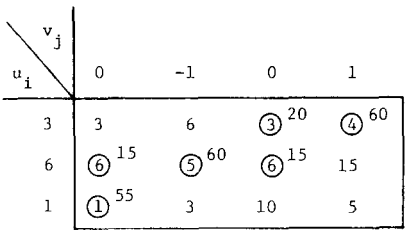


Figure 14

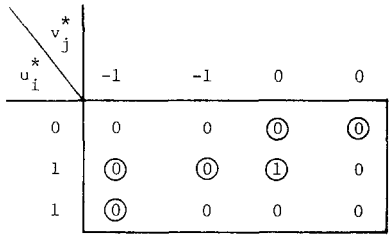
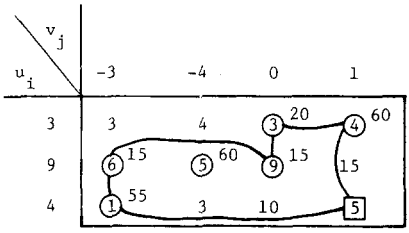


Figure 15



Z = 835+(3x15) = 880

Figure 16

Operation number	Result
(5), (6)	The γ_{ij}, u_i^*, v_i^* are shown in Fig. 8.
(7)	Since $N = \{(2,1), (2,4), (3,1)\}$ we see that $\mu^A = 0$ and $(e,f) = (3,1)$.
(11)	Adding $(3,1)$ to B we see that $(r,s) = (3,4)$. New shipping amounts are given in Fig. 9.
(12)	$c_{34} = 5$, $A = \{(2,3), (3,3)\}$.
(5), (6)	The new γ_{ij}, u_i^*, v_i^* are shown in Fig. 10.
(7)	Now $N = \{(1,3)\}$ so that $\mu^A = 0$ and $(e,f) = (1,3)$.
(11)	Adding $(1,3)$ to B gives $(r,s) = (3,3)$. New shipping amounts are given in Fig. 11.
(12)	$c_{33} = 10$, $A = \{(2,3)\}$.
(5), (6)	The new γ_{ij}, u_i^*, v_j^* are shown in Fig. 12.
(7)	We have $N = \{(2,1), (2,4)\}$ so that $\mu^A = 1$ and $(e,f) = (2,1)$.
(8)	$\mu' = 6$, $\delta = 1$.
(9)	$u_2 = 6$, $v_2 = -1$.
(10)	$c_{23} = 6$. Result in Fig. 13. $Z = 805 + (1 \times 30) = 835$.
(11)	Adding $(2,1)$ to B we see that $(r,s) = (1,1)$. Result in Fig. 14.
(6)	The u_i^*, v_j^* appear in Fig. 15.
(7)	$N = \{(2,4), (3,3), (3,4)\}$ so that $\mu^A = 3$ and $(e,f) = (3,4)$.
(8)	$\mu' = 5$, $\delta = 3$.
(9)	$u_2 = 9$, $u_3 = 4$, $v_1 = -3$, $v_2 = -4$.
(10)	$c_{23} = 9$, see Fig. 16. $Z = 835 + [3 \times 15] = 880$.
(11)	Adding $(3,4)$ to B gives $(r,s) = (2,3)$. Shipping amounts are as shown in Fig. 7.
(12)	$c_{23} = 11$, $A = \emptyset$.
(5)	Stop. Optimum found.

8. Bounds for the area cost operator algorithm

The convergence of the area cost operator algorithm for nondegenerate problems is insured by Theorem 14 on p. 247 of [8]. (We have not been able to prove convergence for the degenerate case; in fact the algorithm may cycle in this case.) We can use the proof technique of this theorem to derive bounds on the number of steps needed by the algorithm to solve the perturbed problem P'. (Recall that for primal nondegenerate problems P' = P.)

Theorem 4. (A) *If problem* P *is nondegenerate, then the area cost operator algorithm converges in at most* T *steps.*

(B) *For a degenerate transportation problem the area cost operator algorithm converges in at most* $m(T+1)$ *steps.*

Proof. In the proof of Lemma 6(c) on p. 243 of [8] it can be shown that Δ, which measures the amount given by the minimum giver, satisfies $\Delta \geq s$. Hence in the proof of Lemma 6(c) the quantity $\Sigma_{i\in I}\Sigma_{j\in J}\gamma_{ij}x'_{ij} = \Sigma_{(p,q)\in A}x'_{pq}$ decreases by at least s each time. But

$$\sum_{(p,q)\in A} x'_{pq} \leq \sum_{(p,q)\in B} x'_{ij} = T' \tag{14}$$

since $A \subseteq B$. Consequently the algorithm would take at most T'/s steps.

(A) For primal nondegenerate problems, $s \geq 1$ and from eq. (11) $T' = T$ so that the bound is $T'/s \leq T$ steps.

(B) For degenerate problems after perturbation, $s \geq 1/m$ and from eq. (12) $T' = T + 1$ so that the bound is $T'/s \leq mT' = m(T+1)$ steps.

Acknowledgment

The authors wish to thank Professor M.L. Balinski and the referees for many valuable comments on an earlier version of this paper.

This report was prepared as part of the activities of the Management Sciences Research Group, Carnegie-Mellon University, and supported by the U.S. Office of Naval Research.

References

[1] M.L. Balinski and R.E. Gomory, "A primal method for the assignment and transportation problems", *Management Science* 10 (1964) 578–593.

[2] A. Charnes and W.W. Cooper, *Management models and industrial applications of linear programming*, Vols. I and II (Wiley, New York, 1961).

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

[4] J. Edmonds and R.M. Karp, "Theoretical improvements in algorithmic efficiency for network flow problems", *Journal of the Association for Computing Machinery* 19 (1972) 248–264.

[5] L.R. Ford and D.R. Fulkerson, *Flows in networks* (Princeton University Press, Princeton, NJ, 1962).
[6] A. Orden, "The transshipment problem", *Management Science* 2 (1956), 276–285.
[7] M. Simmonard, *Linear programming* (Prentice-Hall, Englewood Cliffs, N.J., 1966).
[8] V. Srinivasan and G.L. Thompson, "An operator theory of parametric programming for the transportation problem – I and II", *Naval Research Logistics Quarterly* 19 (1972) 205–252.
[9] V. Srinivasan and G.L. Thompson, "Accelerated algorithms for labelling and relabelling of trees, with applications to distribution problems", *Journal of the Association for Computing Machinery* 19 (1972) 712–726.
[10] V. Srinivasan and G.L. Thompson, "Benefit-cost analysis of coding techniques for the primal transportation algorithm", *Journal of the Association for Computing Machinery* 20 (1973) 194–213.
[11] W. Szwarc, "Some remarks on the time transportation problem", *Naval Research Logistics Quarterly* 18 (1971) 473–485.