OPERATIONS RESEARCH, Vol. 23, No. 1, January-February 1975

On the Set-Covering Problem: II. An Algorithm for Set Partitioning

Egon Balas

Carnegie-Mellon University, Pittsburgh, Pennsylvania

and

Manfred Padberg

International Institute of Management, Berlin, West Germany

(Received January 29, 1973)

In an earlier paper [*Opns. Res.* 20, 1153–1161 (1972)] we proved that any feasible integer solution to the linear program associated with the equality-constrained set-covering problem can be obtained from any other feasible integer solution by a sequence of less than *m* pivots (where *m* is the number of equations), such that each solution generated in the sequence is integer. However, degeneracy makes it difficult to find a sequence of pivots leading to an integer optimum. In this paper we give a constructive characterization of adjacency relations between integer vertices of the feasible set that enables us to generate edges (all, if necessary) connecting a given integer vertex to adjacent integer vertices. This helps overcome the difficulties caused by degeneracy and leads to a class of algorithms, of which we discuss two.

CONSIDER THE weighted set-partitioning (or equality-constrained set-covering) problem

$$(P):\ \min\{cx \mid Ax=e,\ x_j=0 \text{ or } 1,\ j\in N\},$$

where A is a $m\times n$ matrix of zeroes and ones, c is an arbitrary n-vector, $e=(1,\cdots,1)$ is an m-vector, and $N=\{1,\cdots,n\}$. Let (P') be the linear program obtained from (P) by replacing the conditions $x_j=0$ or 1 with $x_j\geqq 0$, $j\in N$.

It is well known that, if A is totally unimodular, then (P) can be solved by solving (P'), since the feasible set of (P') has only integer vertices. This property also holds if A, while not totally unimodular, is *balanced*,[3] and was recently shown to hold in the more general case where A is *perfect*.[9] One should also mention that, even when none of the above properties holds, an optimal solution to (P') will often be integer. In general, however, this need not be the case, and therefore solving (P) by traditional methods requires either cutting planes or enumeration.

In a previous paper,[1] we have established several useful properties of (P). The main result of that paper (Theorem 3.1) states that, if x^1 and x^2 are basic feasible integer solutions to the linear program (P'), x^2 better than x^1, then x^2 can be obtained from x^1 by a sequence of at most p pivots, such that each pivot generates a basic feasible integer solution not worse than its predecessor, p being the number of variables nonbasic in x^1, equal to 1 in x^2. This property of the set-partitioning problem [which, incidentally, is not shared by the inequality-constrained set-covering problem obtained from (P) by replacing $Ax=e$ with $Ax\geqq e$] implies

that the problem can be solved by pivoting, without using cutting planes or partitioning the feasible set by branch and bound, provided one can identify the correct sequence of pivots. To be more specific, this property means that, given a basic feasible integer solution x^1 to (P'), there is a better integer solution if and only if there is one that is adjacent to x^1 on the polytope of feasible solutions to (P'). The difficulty lies in identifying such adjacent vertices. Since set-partitioning problems tend to be highly degenerate, the feasible polytope usually contains an enormous number of vertices adjacent to a given vertex. Furthermore, lexicographic or similar techniques are of no avail in coping with degeneracy, since the sequence of pivots required to reach an adjacent vertex may include pivots on a negative entry in a degenerate row (i.e., a row corresponding to a basic index i such that $x_i^1=0$).

In the present paper we use the results of reference 1, and some new results to be stated below, in order to overcome this difficulty by generating new columns that produce adjacent integer vertices when pivoted into the basis. This leads to a class of algorithms (several different versions are possible) that solve the set-partitioning problem (P) by a finite sequence of primal simplex pivots, without the use of cutting planes, but with the use of new columns added to the simplex tableau at certain intervals.

Section 1 contains a constructive characterization of adjacency relations among the integer vertices of the feasible set of (P'), which enables us to generate all edges connecting a given integer vertex to adjacent integer vertices. One of the interesting by-products of this characterization is a rather tight bound on the diameter of the convex hull of 0-1 points satisfying $Ax=e$ (Corollary 3.4). Section 2 describes a column-generating procedure for obtaining all integer vertices adjacent to a given vertex, while Section 3 states two algorithms based on variants of this procedure. Finally, Section 4 gives a numerical example.

1. ADJACENT INTEGER VERTICES

LET $X=\{x\in R^n \mid Ax=e,\ x\geqq 0\}$ AND LET X_I be the convex hull of the integer points of X. Without loss of generality we can assume that A has no zero rows or columns. Then clearly,

$$X_I\subseteq X\subset K=\{x \mid 0\leqq x_j\leqq 1,\ j\in N\}. \qquad (1)$$

Two vertices of X (of X_I) are said to be *adjacent* if they are contained in an edge (one-dimensional face) of X (of X_I).

Any vertex of X_I is known to be a vertex of X. Hence, with any integer solution x^i to (P') one can associate a basis B_i. We will denote by I_i and J_i the basic and nonbasic index sets associated with B_i, and $Q_i=\{j\in N \mid x_j^i=1\}$, $\bar{Q}_i=N-Q_i$. To simplify notation, we shall assume that the variables of the tableau associated with B_i have been ordered so that $I_i=\{1,\cdots,m\}$. The columns of A will be denoted by a_j. Furthermore, we let $\bar{a}_j=B_i^{-1}a_j$ and denote by $\bar{a}^j$ the n-vector whose kth component is $\bar{a}_{kj}$ for $k\in I_i$, -1 for $k=j$, and 0 for all $k\in J_i-\{j\}$; i.e., $\bar{a}^j$ is the jth nonbasic column of the Tucker tableau.

We start by restating a result proved in reference 1 (Theorem 2.3).

Theorem 1. *Let x^1 and x^2 be basic feasible integer solutions to (P'), and let $\bar a_j = B_1^{-1} a_j$, $j \in J_1$. Then,*

$$\sum_{j\in J_1\cap Q_2} \bar a_{kj} = \begin{cases} 1, & k\in Q_1\cap \bar Q_2,\\ -1, & k\in Q_2\cap \bar Q_1\cap I_1,\\ 0, & k\in (Q_1\cap Q_2)\cup(\bar Q_1\cap \bar Q_2\cap I_1).\end{cases} \tag{2}$$

Next we establish the converse of Theorem 1.

Theorem 2. *Let x^1 be a basic feasible integer solution to (P'), let $\bar a_j = B_1^{-1} a_j$, $j\in J_1$, and let the index set $Q\subseteq J_1$ satisfy*

$$\sum_{j\in Q} \bar a_{kj} = \begin{cases} 0 \text{ or } 1, & k\in Q_1,\\ 0 \text{ or } -1, & k\in I_1\cap \bar Q_1.\end{cases} \tag{3}$$

Then

$$x^2 = x^1 - \sum_{j\in Q} \bar a^j \tag{4}$$

is a basic feasible solution to (P'), and

$$x_j^2 = \begin{cases} 1, & j\in Q_2 = Q\cup S,\\ 0, & \text{otherwise},\end{cases}$$

where $S = \{k\in Q_1 \mid \sum_{j\in Q} \bar a_{kj} = 0\} \cup \{k\in I_1\cap \bar Q_1 \mid \sum_{j\in Q} \bar a_{kj} = -1\}$.

Proof. Consider the problem $(\bar P')$ in $(n+1)$-space, obtained from (P') by augmenting A with the composite column $a_{j_*} = \sum_{j\in Q} a_j$. The transformed column $\bar a_{j_*} = B_1^{-1} a_{j_*}$ has an entry $\bar a_{kj_*} = 1$ for some $k\in Q_1$, for otherwise (3) implies $\bar a_{kj_*} \leqq 0$, $\forall k\in I_1$, which is impossible in view of the boundedness of the solution set. Pivoting on $\bar a_{kj_*} = 1$ yields a feasible solution $\tilde x^2$ to $(\bar P')$, defined by

$$\tilde x_j^{\,2} = \begin{cases} 1, & j\in\{j_*\}\cup S,\\ 0, & \text{otherwise}.\end{cases}$$

Since $\sum_{j\in S} a_j + a_{j_*} = \sum_{j\in S\cup Q} a_j = e$, it follows that x^2 as defined in the theorem is feasible for (P'). Since x^2 is integer, it is also basic. From Theorem 1, relation (4) follows with $Q = J_1 \cap Q_2$.

A set $Q\subset J_1$ for which (3) holds will be called *decomposable* if it can be partitioned into two subsets, Q^* and Q^{**}, such that (3) remains true when Q is replaced by Q^* and Q^{**} respectively.

We now give a necessary and sufficient condition for two integer vertices of X to be adjacent.

Theorem 3. *Let x^1 and x^2 be two integer solutions to (P'), with $Q = J_1\cap Q_2$. Then x^2 is adjacent to x^1 on X if and only if Q is not decomposable.*

Proof. (i) Suppose x^1 and x^2 are not adjacent on X. Let c be the row vector with n components defined by

$$c_j = \begin{cases} -2, & j\in Q_1\cap Q_2,\\ 2, & j\in I_1\cap \bar Q_1\cap \bar Q_2,\\ -1, & \text{for exactly one } j_0\in Q,\\ 0, & \text{otherwise}.\end{cases}$$

Consequently, $cx^1 > cx^2$ and x^2 is an optimal solution to problem (P) with c as defined above. Hence, Theorem 3.1 of reference 1 applies; i.e., there exists a sequence of $p = |Q|$ pivots, each in a column whose index is in Q, generating a sequence of basic feasible integer solutions $\xi^0 = x^1, \xi^1, \cdots, \xi^p = x^2$, with $c\xi^0 \geqq c\xi^1 \geqq \cdots \geqq c\xi^p$.

Define $r_i = |\{j\in I_1\cap\bar Q_1\cap\bar Q_2 \mid \xi_j^{\,i} = 1\}|$ and $s_i = |\{j\in Q_1\cap Q_2 \mid \xi_j^{\,i} = 0\}|$, $i = 0, 1, \cdots, p$.

Then, from the definition of c, and the monotonicity of $c\xi^i$, we have $-2|Q_1\cap Q_2| \geqq -2\{|Q_1\cap Q_2| - s_i\} + 2r_i - 1$, or, equivalently, $0 \leqq 2(r_i + s_i) \leqq 1$ for all $i\in\{0, 1, \cdots, p\}$. Consequently, $r_i = s_i = 0$ for $i = 0, 1, \cdots, p$; i.e., ξ^i satisfies

$$\xi_j^{\,i} = \begin{cases} 1, & j\in Q_1\cap Q_2,\\ 0, & j\in I_1\cap \bar Q_1\cap \bar Q_2.\end{cases} \qquad (i = 0, 1, \cdots, p) \tag{5}$$

Since by assumption x^1 is not adjacent to x^2, $p\geqq 2$ and there exists in the sequence $\xi^1, \cdots, \xi^p$ a solution $\xi^k = x^3$ such that $x^1 \neq x^3 \neq x^2$. Furthermore, since each solution in the above sequence is generated by pivoting into the basis a column a_j such that $j\in Q$, a solution ξ^i differs from $\xi^0 = x^1$ if and only if $\xi_j^{\,i} = 1$ for some $j\in Q$. Hence, from $x^3 \neq x^1$, $Q^* = \{j\in Q \mid x_j^3 = 1\} \neq \emptyset$. Also, $Q^{**} = \{j\in Q \mid x_j^3 = 0\} \neq \emptyset$, since otherwise $x^3 = x^2$. Also, $Q^*\cup Q^{**} = Q$, and $Q^*\cap Q^{**} = \emptyset$.

Further, since x^1 and x^3 are basic feasible integer solutions to (P'), from Theorem 1, relation (2) holds when $J_1\cap Q_2$ is replaced by Q^*. Therefore (3) also holds when Q is replaced by Q^*.

To prove that (3) also holds when Q is replaced by Q^{**}, we proceed as follows. From Theorem 2, $x^3 = x^1 - \sum_{j\in Q^*} \bar a^j$. Also from Theorem 2 and from $Q = Q^*\cup Q^{**}$, $x^2 = x^1 - \sum_{j\in Q} \bar a^j = x^3 - \sum_{j\in Q^{**}} \bar a^j$, and hence the components of the vector $\sum_{j\in Q^{**}} \bar a^j$ must all equal $0, +1$, or -1. Now define $x^4 = x^1 - \sum_{j\in Q^{**}} \bar a^j$.

We will show that $0 \leqq x_j^4 \leqq 1$ for all $j\in N$. Assume first that $x_k^4 = -1$ for some $k\in N$. Then we find that $x_k^1 = x_k^2 = 0$, and $x_k^3 = 1$. Consequently, $k\in \bar Q_1\cap \bar Q_2$. Since each pivot in the sequence was performed on a nonbasic column with index in the set Q, it follows that $k\in I_1$. By (5) this implies that $x_k^3 = 0$, a contradiction. This proves that $x_j^4 \geqq 0$ for all $j\in N$. Similarly, we find that $x_j^4 \leqq 1$ for all $j\in N$. From this and the integrality of $\sum_{j\in Q^{**}} \bar a^j$ it follows that x^4 is a basic feasible integer solution to (P'). Hence relation (3) holds when Q is replaced by Q^{**}.

Thus Q is decomposable.

(ii) Suppose now that Q is decomposable into Q^* and Q^{**}. Then the vectors

$$x^i = x^1 - \sum_{j\in S_i} \bar a^j, \qquad (i = 2, 3, 4) \tag{6}$$

where $S_2 = Q$, $S_3 = Q^*$, and $S_4 = Q^{**}$, are all feasible integer solutions to (P'), hence vertices of X_I. Let $\pi x = \pi_0$ be a supporting hyperplane for X_I such that $\pi x^i = \pi_0$ for $i = 1, 2$ and $\pi x \leqq \pi_0$, $\forall x\in X_I$. (If no such hyperplane exists, then x^1 and x^2 are not adjacent on X_I, hence on X, and the statement is proved.) Then, from (6), $\pi x^1 = \pi x^2 = \pi x^1 - \pi(\sum_{j\in Q} \bar a^j) = \pi_0$, or

$$\pi\Big(\sum_{j\in Q} \bar a^j\Big) = 0, \tag{7}$$

whereas

$$\pi x^3 = \pi x^1 - \pi\Big(\sum_{j\in Q^*} \bar a^j\Big) \leqq \pi_0 = \pi x^1, \qquad \pi x^4 = \pi x^1 - \pi\Big(\sum_{j\in Q^{**}} \bar a^j\Big) \leqq \pi_0 = \pi x^1,$$

or

$$\pi\Big(\sum_{j\in Q^*} \bar a^j\Big) \geqq 0, \qquad \pi\Big(\sum_{j\in Q^{**}} \bar a^j\Big) \geqq 0. \tag{8}$$

Then from (7) and (8) we have $\pi(\sum_{j\in Q^*} \bar a^j) = 0$, $\pi(\sum_{j\in Q^{**}} \bar a^j) = 0$, or $\pi x^3 = \pi x^4 = \pi_0$. Hence, any supporting hyperplane for X_I that contains x^1 and x^2 also contains x^3 and x^4; i.e., x^1 and x^2 cannot lie on an edge of, or be adjacent on, X_I. Hence a fortiori they cannot be adjacent on X.

Theorem 3 is stated in terms of the columns of the simplex tableau associated with a given solution x^1 and basis B_1. In reference 10 this result is restated in terms of the columns of the original matrix A, without reference to a specific basis.

An immediate consequence of Theorem 3 is the following interesting geometric property, first derived by TRUBIN.[12]

COROLLARY 3.1. *Two integer points are adjacent vertices of X_I if and only if they are adjacent vertices of X.*

Proof. Let x^1 and x^2 be adjacent vertices of X_I, and let B_1 be a basis associated with x^1 in (P'). From part (ii) of the proof of Theorem 3, this implies that $Q = J_1 \cap Q_2$ is not decomposable. Therefore, from Theorem 3, x^1 and x^2 are adjacent vertices of X. The converse is obvious, since $X_I \subseteq X$.

COROLLARY 3.2. *Two vertices of X_I, x^1 and x^2, are not adjacent if and only if*

$$x^2 = x^1 - \sum_{i=1}^{i=k} \sum_{j \in Q_{1i}} \bar{a}^j = x^1 + \sum_{i=1}^{i=k} (x^{1i} - x^1) \qquad (9)$$

with $k \geq 2$, where the points $x^{1i} = x^1 - \sum_{j \in Q_{1i}} \bar{a}^j$, $i = 1, \cdots, k$, are vertices of X_I adjacent to x^1, and $\bigcup_{i=1}^{i=k} Q_{1i} = J_1 \cap Q_2$.

Proof. If the condition holds, then (3) holds with Q replaced by each Q_{1i}, $i = 1, \cdots, k$, and also by $\bigcup_{i=1}^{i=k} Q_{1i}$. Therefore. $\sum_{j \in Q_{1i}} \bar{a}^j$ and $\sum_{j \in Q_{1h}} \bar{a}^j$ are orthogonal for all $i, h \in \{1, \cdots, k\}$, $i \neq h$, and hence (3) also holds with Q replaced by $\bigcup_{i=2}^{i=k} Q_{1i}$. Thus Q is decomposable into Q_{11} and $\bigcup_{i=2}^{i=k} Q_{1i}$, hence x^1 and x^2 are not adjacent. Conversely, if x^1 and x^2 are not adjacent, then Q can be decomposed into Q^* and Q^{**}. If $x^* = x^1 - \sum_{j \in Q^*} \bar{a}^j$ and $x^{**} = x^1 - \sum_{j \in Q^{**}} \bar{a}^j$ are both adjacent to x^1, the statement is proved; otherwise, the reasoning can be applied to Q^* and/or Q^{**}, and can be repeated as many times as needed to obtain sets Q_{1i} that are not decomposable.

COROLLARY 3.3. *If x^1 and x^2 are two nonadjacent vertices of X_I related to each other by (9), then for any subset H of $\{1, \cdots, k\}$, $x^* = x^1 - \sum_{i \in H} \sum_{j \in Q_{1i}} \bar{a}^j$ is a vertex of X_I.*

Proof. Follows from the fact that $\sum_{j \in Q_{1i}} \bar{a}^j$ and $\sum_{j \in Q_{1h}} \bar{a}^j$ are orthogonal for all $i, h \in \{1, \cdots, k\}$, $i \neq h$.

Corollary 3.2 has an interesting geometric interpretation. For an arbitrary polytope P, a *path* between two vertices x, y of P is a sequence of vertices $x^1, x^2, \cdots, x^k$, with $x^1 = x$, $x^k = y$, such that every pair of vertices x^i, x^{i+1}, $i = 1, \cdots, k-1$, is connected by an edge of P, the length of the path being $k-1$. The *edge-distance* $d(x, y)$ between x and y is then defined as the length of a shortest path on P between x and y. The *diameter* $\delta(P)$ of P is the longest edge-distance between any pair of vertices of P; i.e., $\delta(P) = \max_{x, y \in \text{vert} P} d(x, y)$.

For the next statement we shall require explicitly that the matrix A in the definition of X not contain identical columns.

COROLLARY 3.4. *$\delta(X_I) \leq [z^*/2] \leq [m/2q]$, where $\delta(X_I)$ is the diameter of X_I, $q = \min_{j \in N} \sum_{i=1}^{i=m} a_{ij}$, and $z^* = \max_{x \in X_I} \sum_{j=1}^{j=n} x_j$.*

Proof. (i) To prove the first inequality, let x^1 and x^2 be two vertices of X_I for which $d(x^1, x^2) = \delta(X_I)$. If x^1 and x^2 are not adjacent, from Corollaries 3.2 and 3.3, (9) holds with $k \geq \delta(X_I)$; if they are adjacent, (9) holds with $k = 1 = \delta(X_I)$. Now let K_1 be the set of indices i such that $\sum_{j \in Q_{1i}} \bar{a}^j$ has exactly one positive component. If $K_1 = \emptyset$, $k \leq [z^*/2]$ in (9), since $|Q_1| \leq z^*$. Suppose now that $K_1 \neq \emptyset$. Then for each $i \in K_1$, the composite column $\bar{a}^{1i} = \sum_{j \in Q_{1i}} \bar{a}^j$ has at least two negative components. For if not, then Q_{1i} is a singleton, say $Q_{1i} = \{h\}$, and the only negative component of $\bar{a}^{1i} = \bar{a}^h$ is $\bar{a}_{hh} = -1$; hence the components corresponding to the basic index set I_1 form a unit vector, which implies that the nonbasic column a_h of A is identical to a basic column, contrary to our assumption.

Now let

$$x^3 = x^1 - \sum_{i \in K_1} \sum_{j \in Q_{1i}} \bar{a}^j, \qquad x^4 = x^1 - \sum_{i \in K - K_1} \sum_{j \in Q_{1i}} \bar{a}^j,$$

where $K = \{1, \cdots, k\}$. From Corollary 3.3, both x^3 and x^4 are vertices of X_I. But then

$$x^4 = x^3 - \sum_{i \in K_1} \left(-\sum_{j \in Q_{1i}} \bar{a}^j\right) - \sum_{i \in K - K_1} \sum_{j \in Q_{1i}} \bar{a}^j,$$

which implies that $k \leq [|Q_3|/2]$, where $Q_3 = \{j \in N | x_j^3 = 1\}$. Consequently $k \leq [z^*/2]$, since $|Q_3| \leq z^*$.

(ii) To prove the second inequality, let $\bar{x}$ be such that

$$\sum_{i=1}^{i=n} \bar{x}_i = \max_{x \in X_I} \sum_{i=1}^{i=n} x_i = z^*.$$

Since the maximum over X_I is attained at a vertex, it is no restriction to assume that $\bar{x}$ is integer, and thus $z^* = |Q|$, where $Q = \{j \in N | \bar{x}_i = 1\}$. Since $\sum_{j \in Q} a_j = e$, it follows that $|Q| \leq m/q_0 \leq m/q$, where $q_0 = \min_{j \in Q} \sum_{i=1}^{i=m} a_{ij} \geq q$.

Since $|Q| = z^*$, it follows that $z^* \leq m/q$, and hence $[z^*/2] \leq [m/2q]$.

Remark. If $A = (A_G, I)$ in the definition of X, where A_G is the $m \times \binom{m}{2}$ incidence matrix of the complete graph with m vertices and I is the identity matrix of order m, then the upper bound $[z^*/2]$ on the diameter of X_I is actually attained. In this sense, the upper bound on $\delta(X_I)$ provided by Corollary 3.4 is a best possible one.

COROLLARY 3.5. *Two vertices x^1 and x^2 of X_I, which are at an edge-distance of k from each other, $2 \leq k \leq \delta(X_I)$, are connected by $k!$ paths of length k.*

Proof. From (9), $x^2 = x^1 - \sum_{i=1}^{i=k} \sum_{j \in Q_{1i}} \bar{a}^j$.

For any permutation $i_1, \cdots, i_k$ of the index set $\{1, \cdots, k\}$, it follows from Corollaries 3.2 and 3.3 that the points $x^1, x^{(i_1)}, x^{(i_2)}, \cdots, x^{(i_k)}$, where

$$x^{(i_1)} = x^1 - \sum_{j \in Q_{1i_1}} \bar{a}^j,$$
$$x^{(i_2)} = x^{(i_1)} - \sum_{j \in Q_{1i_2}} \bar{a}^j,$$
$$\cdots$$
$$x^{(i_k)} = x^2 = x^{(i_{k-1})} - \sum_{j \in Q_{1i_{k-1}}} \bar{a}^j$$

form a sequence of $k+1$ vertices of X_I, adjacent on X_I whenever adjacent in the sequence. Since there are $k!$ possible permutations of the index set $\{1, \cdots, k\}$, there are $k!$ such sequences of adjacent vertices, each one defining a path of length k from x^1 to x^2 on X_I.

THEOREM 4. *Let x^1 be a nonoptimal vertex of X_I, let x^{1i}, $i = 1, \cdots, k$, be the vertices of X_I adjacent to x^1, and such that $cx^{1i} < cx^1$, $i = 1, \cdots, k$. Then the convex polyhedral cone*

$$C = \{x | x = x^1 + \sum_{i=1}^{i=k} (x^{1i} - x^1)\lambda_i,\ \lambda_i \geq 0,\ i = 1, \cdots, n\}$$

contains an optimal vertex of X_I.

Proof. Let $\bar{x}$ be an optimal vertex of X_I. If $\bar{x}$ is adjacent to x^1, then $\bar{x} \in C$. Otherwise, $\bar{x}$ can be expressed (Corollary 3.2) as

$$\bar{x} = x^1 - \sum_{i=1}^{i=p} \sum_{j \in Q_{1i}} \bar{a}^j = x^1 + \sum_{i=1}^{i=p} (x^{1i} - x^1),$$

where x^{1i}, $i = 1, \cdots, p$, are vertices of X_I adjacent to x^1. Then

$$0 < cx^1 - c\bar{x} = \sum_{i=1}^{i=p} \sum_{j \in Q_{1i}} c\bar{a}^j = \sum_{i=1}^{i=p} \sum_{j \in Q_{1i}} (z_j - c_j),$$

where $z_j = c_B B_1^{-1} a_j$, c_B being the vector whose components are the components of c indexed by I_1. Let $K = \{1, \cdots, k\}$. Then $K \neq \emptyset$, since $c\bar{x} < cx^1$. From Corollary 3.3, the point

$$x^* = x^1 - \sum_{i \in K} \sum_{j \in Q_{1i}} \bar{a}^j = x^1 + \sum_{i \in K} (x^{1i} - x^1)$$

is a vertex of X_I, and from the definition of K,

$$cx^* = cx^1 - \sum_{i \in K} \sum_{j \in Q_{1i}} (z_j - c_j) \leqq cx^1 - \sum_{i=1}^{i=p} \sum_{j \in Q_{1i}} (z_j - c_j) = c\bar{x}.$$

Thus, since $\bar{x}$ is optimal, so is x^*; and since the vertices x^{1i}, $i \in K$ are among those that generate C, clearly $x^* \in C$.

The property stated in Theorem 4 is not true for arbitrary integer programs, as shown by the trivial counterexample of Fig. 1. Here $cx^1 > cx^2 > cx^3$, and the cone C (here just a halfline) clearly does not contain the (unique) optimal point x^3. It is, however, possible to generalize Theorem 4 (as well as Theorem 3) to arbitrary 0-1 programs (*see* reference 2).

The above results can be used to overcome the difficulties caused by degeneracy

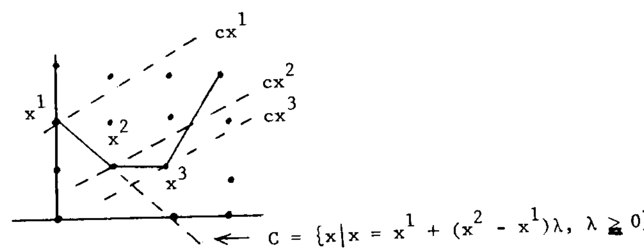

Figure 1

in finding integer vertices of X adjacent to a given integer vertex x^1. That is, by systematically generating composite columns of the form $\bar{a}_{j^*} = \sum_{j \in Q} \bar{a}_j$, where Q satisfies the requirements for $x^1 - \bar{a}^{j^*}$ to be a vertex of X_I adjacent to x^1, one can obtain all such vertices. While we are basically interested only in generating adjacent vertices better than a given vertex, we will first describe a more general procedure which produces all adjacent integer vertices. This seems useful, since the more specific goal of generating adjacent integer vertices better than a given vertex can be achieved in several ways by modifying the general procedure, whereas in this paper we confine ourselves to describing one such possible modification, and outlining a second one.

In the next section we discuss the general procedure.

2. COLUMN-GENERATING PROCEDURE

IN THIS SECTION we describe a procedure for generating all integer vertices of X adjacent to a given integer vertex. The procedure generates all *composite* columns of the form $\bar{a}^k = \sum_{j \in Q} \bar{a}^j = x^1 - x^2$, where x^2 is a vertex of X_I adjacent to x^1.

Let x^1 be a basic feasible integer solution to (P'), B_1 an associated basis, I_1 and J_1 the basic and nonbasic index sets. Further, let $A = (B_1, R_1)$, and let $\bar{a}^j$ be the jth column of $B_1^{-1} R_1$. Furthermore, let $I^0 = \{i \in I_1 | x_i^1 = 0\}$ denote the index set of 'degenerate' rows of I_1.

We will work with a submatrix T of $T_1 = B_1^{-1} R_1$, namely, the one consisting of the rows of T_1 specified by the index set I^0. The columns of the submatrix (tableau) T will be denoted by $\hat{a}_j$. We shall continue to denote the components of $\hat{a}_j$ by $\bar{a}_{ij}$. The column-generation procedure (CGP) generates a sequence of tableaux T^1, T^2, $\cdots$, T^f with associated column index sets $J^1, J^2, \cdots, J^f$ by adding new columns and/or deleting old ones. Initially, we set $T^1 = T$ and $J^1 = J$. The columns of any tableau T^k for $2 \leqq k \leqq f$ are *composites* of the columns of T^1, so that the index j of each new column corresponds to some subset $Q_j \subset J^1$ of the original column index set; i.e., $\hat{a}_j = \sum_{h \in Q_j} \hat{a}_h$. For the original columns, $Q_j = \{j\}$.

The information required to generate the columns of T^{k+1} is obtained from the rows of the tableau T^k, which are processed one at a time. When all entries of a row of T^k become equal to 0 or -1, the row is *marked*; and another, unmarked row is chosen for processing.

For simplicity, we will let T and J, respectively, denote any 'current' tableau T^k and its associated column index set J^k, respectively, and let T' and J' denote the 'next' tableau T^{k+1} and its associated column index set J^{k+1} respectively. The rules of CGP then are as follows:

1. If all rows have been marked, go to 4. Otherwise, go to 2.
2. Choose any unmarked row r. Define

$$J^+ = \{j \in J | \bar{a}_{rj} > 0\}, \qquad J^- = \{j \in J | \bar{a}_{rj} < 0\}.$$

a. If $J^- = \emptyset$, mark row r and remove from T all columns with $j \in J^+$. Call the resulting tableau T' and set $J' = J - J^+$. Go to 3.

b. If $J^- \neq \emptyset$, but $J^+ = \emptyset$, mark row r, and remove from T all columns $\hat{a}_j$ such that $j \in \hat{J} = \{j \in J^- | \bar{a}_{rj} < -1\}$. Call the resulting tableau T' and set $J' = J - \hat{J}$. Go to 3.

c. If $J^- \neq \emptyset$ and $J^+ \neq \emptyset$, choose any $t \in J^+$ and proceed as follows:

(i) Define $S_t = S_t^1 \cap S_t^2 \cap S_t^3$, where

$$S_t^1 = \{j \in J^- | a_t a_j = 0\},$$

$$S_t^2 = \{j \in J^- | \bar{a}_{hj} + \bar{a}_{ht} \geqq -1 \quad \text{for all marked rows} \quad h\},$$

$S_t^3 = \{j \in J^- | Q_j \cup Q_t$ cannot be partitioned into Q_h, $h = j_1, \cdots, j_p$, with $j_i \in J$, $j_i \neq t$, for $i = 1, \cdots, p\}$.

(ii) For each $j \in S_t$ add a new column $\hat{a}_j + \hat{a}_t$ to T and a new index k to J, where $Q_k = Q_j \cup Q_t$; then remove $\hat{a}_t$ from T and t from J. Call the resulting tableau T' and the resulting column index set J'. Go to 3.

3. Designate T' and J' to be the current tableau T and index set J, respectively, and return to 1.

4. Construct the full tableau $\bar{T}$ by computing for each column of T the entries in the nondegenerate rows (indexed by $I_1 - I^0$). Denote by $\bar{a}_j$ the column of $\bar{T}$ corresponding to $\hat{a}_j$. Then create the final tableau T_f by removing from $\bar{T}$ all columns $\bar{a}_j$ that violate the condition $\bar{a}_{ij} = 0$ or 1, $\forall i \in I_1 - I_0$.

Stop: T_f yields all integer vertices adjacent to x^1, each in one pivot.

THEOREM 5. *In a finite number of steps* CGP *terminates with a tableau* T_f *such that* (i) *pivoting into the basis any column of* T_f *yields an integer vertex of* X *adjacent to* x^1, *and* (ii) *all integer vertices of* X *adjacent to* x^1 *can be obtained by such a pivot.*

Proof. The procedure generates composite columns $\bar{a}_k = \sum_{j\epsilon Q_k} \bar{a}_j$. We will show that (i) each $\bar{a}_k$ satisfies (3), (ii) for each $\bar{a}_k$, Q_k is not decomposable, and (iii) all composite columns satisfying (i) and (ii) are generated and present in T_f.

Since all integer vertices $\bar{x}$ of X adjacent to x^1 are of the form $\bar{x} = x^1 - \bar{a}^k$, where $\bar{a}^k$ is a composite column (possibly a singleton) whose associated $\bar{a}_k$ satisfies (i) and (ii), and since the variable associated with such a composite column can be pivoted into the basis with value 1 [in view of (i)], this will prove the theorem.

(i) Each $\bar{a}_k$ satisfies (3): Composite columns violating (3) are eliminated either as soon as they are generated (steps 2a, 2b of CGP), or at the end (step 4 of CGP).

(ii) For each $\bar{a}_k$, Q_k is not decomposable: This is guaranteed by the fact that in combining a column $\bar{a}_t$ with other columns $\bar{a}_j$ to generate composites, we restrict ourselves to $j\epsilon S_t^3$ (step 2c of CGP).

(iii) To show that all composite columns with the required properties are indeed generated and present in T_f, we point to the fact that the original tableau $T_1 = B_1^{-1}R_1$ contains all the columns whose composites yield integer solutions corresponding to vertices of X adjacent to x^1. Suppose that at each iteration we construct the full tableau $\bar{T}$ (obtained from T by completing the columns of T with the entries in the nondegenerate rows), and that at the kth iteration the current tableau $\bar{T}$ still has the above property of T_1. Then, after one iteration of CGP, the new full tableau $\bar{T}'$ also has the property. Indeed, $\bar{T}'$ is obtained from $\bar{T}$ in one of the following ways:

(α) By removing from $\bar{T}$ all columns having positive entries in a degenerate row that has no negative entries, or by removing from $\bar{T}$ all columns having entries strictly less than -1 in a degenerate row that has no positive entries (steps 2a, 2b of CGP). In both cases, none of the removed columns can yield, in conjunction with any other column, a composite column satisfying (3).

(β) By removing from $\bar{T}$ a column $\bar{a}_t$ having a positive entry in a degenerate row that has positive *and* negative entries, while adjoining to $\bar{T}$ all composite columns of the form $\bar{a}_j + \bar{a}_t$ that satisfy $j\epsilon S_t$, where S_t is defined in step 2c(i) of CGP. This set S_t is constructed so that no composite column $\bar{a}_k = \sum_{j\epsilon Q_k} \bar{a}_j$ containing $\bar{a}_t$ is excluded if it satisfies $\bar{x} = x^1 - \bar{a}^k$ for some vertex $\bar{x}$ adjacent to x^1; i.e., $\bar{a}_k$ satisfies (3), and Q_k is not decomposable. In fact, in step 2c(ii) only such composites $\bar{a}_j + \bar{a}_t$ are excluded that satisfy at least one of the following relations:

(a) $\bar{a}_{ij} \geq 0$ and $\bar{a}_{it} > 0$ for some (degenerate) row $i\epsilon I^0$.

(b) $a_j a_t \neq 0$.

(c) $\bar{a}_{ij} + \bar{a}_{it} < -1$ in a marked row i.

(d) $Q_j \cup Q_t$ can be partitioned into Q_h, $h = j_1, \cdots, j_p$, where $j_i \neq l$, $j_i \epsilon J$, for $i = 1, \cdots, p$.

In case (a), the restriction is justified by the fact that, for any composite column $\bar{a}_k = \sum_{j\epsilon Q_k} \bar{a}_j$, if $t\epsilon Q_k$ and $\bar{a}_{it} > 0$ for some $i\epsilon I^0$, then (3) requires Q_k to contain some index j such that $\bar{a}_{ij} < 0$. Case (b) is obvious, whereas (c) eliminates columns $\bar{a}_j + \bar{a}_t$ for which $\bar{a}_{ij} + \bar{a}_{it} < -1$ in a row $i\epsilon I^0$ which has no positive entry, i.e., columns that cannot yield, in conjunction with any other set of columns, a composite column satisfying (3) for the given row. Finally, case (d) eliminates columns that are composites of other columns present in the tableau; should such a column be needed, it will be generated again from those other columns.

Thus, if $\bar{T}$ contains all the columns whose composites (possibly including singletons) yield integer vertices of X adjacent to x^1, then so does $\bar{T}'$ and therefore T_f, the final tableau.

The procedure is clearly finite, since each iteration either removes from the current tableau some columns, or adds to it some new columns that are composites of the original columns. Since no composite column is added twice and there are only n columns in the original tableau, all legitimate composites are generated in a finite number of iterations.

Corollary 5.1. CGP *remains finite if in step 2c, new columns $\hat{a}_j + \hat{a}_k$ are added to T only for some (rather than all) $j\epsilon S$, but $\hat{a}_t$ is not removed from T; provided that at finite intervals (say, at each pth iteration for some fixed $p > 0$), step 2 is applied in its original form.*

It should be mentioned that the successive tableaux T need not be explicitly generated and stored. It is sufficient to store (besides A and c), B_1^{-1} and the index sets Q_j corresponding to each composite column j of T. Since the rows of T are processed one at a time, they can be generated from A and B_1^{-1} when needed. The only instance (before the final step 4) when one has to do something outside the current row i is the construction of S_t^2; and in this context one may wish to define S_t as $S_t = S_t^1 \cap S_t^3$ and perform the test used in the definition of S_t^2 once the row under consideration has no positive entries. This way one can restrict all calculations of step 2 to the current row.

Example. Table I gives an illustration of CGP. The first five columns of A form a basis B_1, with B_1^{-1}, $T_1 = B_1^{-1}R_1$, and $B_1^{-1}e$ as shown in Table I. Thus $J_1 = \{6, 7, 8, 9, 10, 11\}$ $I^0 = \{2, 4, 5\}$, and the first tableau is T_1 of Table II. Starting with $i = 2$, we obtain the new Tableau T_2. The next iteration, in which we choose $i = 4$, reduces the tableau to T_3. Finally, choosing $i = 5$, we obtain T_4. The corresponding rows 1 and 3 of the transformed simplex tableau are now easily constructed, yielding $\hat{T}_5$. Thus, the integer feasible solution $x_1 = x_3 = 1$, $x_j = 0, j \neq 1, 3$, has three adjacent integer feasible solutions, defined by $x_8 = x_{10} = 1$, $x_7 = x_{11} = 1$ and $x_4 = x_6 = x_{11} = 1$, respectively (with $x_j = 0$ unless otherwise specified). We notice that the order in which the rows of the tableau are processed may have a considerable impact on the size of the intermediate tableaux. If each time one chooses a row with a minimum number of positive entries, one tends to generate fewer tableaux.

Our column-generating procedure is kindred in spirit to an algorithm of Chernikova,[5,6] based on theoretical results by Burger,[4] for generating all edges of a polyhedral cone. A useful discussion and restatement of Chernikova's algorithm in the context of the simplex method is to be found in a recent paper by Rubin.[11] It was actually Rubin's paper that started our thinking along the lines that led to the procedure of this section. Our method, like Chernikova's, generates edges of the cone associated with a given vertex, by combining columns and using information from the degenerate rows in order to restrict the combinations to those potentially useful in identifying an edge that contains an adjacent vertex.

However, Chernikova's algorithm takes linear combinations with coefficients chosen so as to yield a zero component in a certain degenerate row, which in general implies the use of coefficients different from 1, whereas our composite columns are linear combinations with all coefficients equal to 1. Also, we use the degenerate rows in a different way, and our strongest elimination devices are the orthogonality requirement and the conditions (3), which are peculiar to our problem and have no counterpart in Chernikova's algorithm.

One could of course, in view of Corollary 3.1, generate all vertices of X, whether integer or not, adjacent to a given vertex, by using Chernikova's algorithm, and then remove the noninteger ones. However, since the number of all adjacent vertices can be vastly superior to that of the integer ones, this does not seem reasonable. In the case of the above numerical example, for instance, Chernikova's procedure generates 10 noninteger vertices adjacent to $x_1=x_3=1$, $x_j=0$, $j\neq 1, 3$, in addition to the three integer vertices that were also generated by our procedure.

3. SET PARTITIONING WITHOUT CUTTING PLANES

IN THIS SECTION we describe two algorithms for solving the equality-constrained set covering problem (P). Both algorithms share the feature that they apply the primal simplex method to problem (P') without recourse to cutting planes, and use a column-generating procedure to overcome the difficulties caused by degeneracy. In Algorithm I, the column-generating procedure is geared to producing *one* 'improving edge' as soon as possible, i.e., a composite column that can be pivoted into the basis so as to yield an integer solution adjacent to, and better than, the current one. In Algorithm II, CGP is used to generate *all* composite columns that yield an integer solution adjacent to and better than the current one; then, in view of Theorem 4, all remaining columns of the current tableau can be removed. Itis not clear at this stage which of the two procedures is preferable, and hybrid algorithms are also feasible.

TABLE I

AN ILLUSTRATION OF THE COLUMN-GENERATING PROCEDURE

$$A=\begin{bmatrix}1&0&0&0&0&1&1&1&1&0&0\\1&1&0&0&0&0&0&0&0&1&1\\0&1&1&0&0&1&1&1&0&0&0\\0&1&1&1&0&0&0&1&1&0&1\\0&0&1&1&1&0&1&0&0&1&0\end{bmatrix},$$

$$B_1^{-1}=\begin{bmatrix}1&0&0&0&0\\-1&1&0&0&0\\1&-1&1&0&0\\0&0&-1&1&0\\-1&1&0&-1&1\end{bmatrix},$$

$$T_1=B_1^{-1}R_1=\begin{bmatrix}1&1&1&1&0&0\\-1&-1&-1&-1&1&1\\2&2&2&1&-1&-1\\-1&-1&0&1&0&1\\-1&0&-2&-2&2&0\end{bmatrix},\qquad B_1^{-1}e=\begin{bmatrix}1\\0\\1\\0\\0\end{bmatrix}.$$

Both algorithms start by applying the primal simplex method to (P'), and of course one has only to gain if one can simplify (P') before starting. Thus, the various 'reduction rules' proposed in the literature (*see*, for instance, references and 8) should first be applied to (P'). Also, a good starting solution may be of great help, and any efficient heuristics for finding one can help a lot.

ALGORITHM I

PRIMAL. Apply the primal simplex method to (P') as long as you can pivot on $+1$ in a nondegenerate row. Whenever this becomes impossible, let $\bar{x}$ be the current (integer) solution, B the associated basis, I and J the basic and nonbasic index sets, $A=(B, R)$, and $T=B^{-1}R$ the current (all-integer) simplex tableau. Let $\bar{a}_j$, $j\epsilon J$, be the columns of $\tilde{T}$, and $\bar{c}_j$, $j\epsilon J$, the reduced costs, with $\bar{c}_j\geq 0$ required for optimality. If $\bar{c}_j\geq 0$, $\forall j\epsilon J$, stop: $\bar{x}$ is optimal for (P).

TABLE II

TABLEAUX FOR THE ILLUSTRATION OF TABLE I

T_1	6	7	8	9	10	11
2	−1	−1	−1	−1	1	1
4	−1	−1	0	1	0	1
5	−1	0	−2	−2	2	0

T_2	6	7	8	9	(6,10)	(8,10)	(9,10)	(6,11)	(7,11)
2	−1	−1	−1	−1	0	0	0	0	0
4	−1	−1	0	1	−1	0	1	0	0
5	−1	0	−2	−2	1	0	0	−1	0

T_3	6	7	8	(6,10)	(8,10)	(6,11)	(7,11)
2	−1	−1	−1	0	0	0	0
4	−1	−1	0	−1	0	0	0
5	−1	0	−2	1	0	−1	0

T_4	6	7	(8,10)	(6,11)	(7,11)
2	−1	−1	0	0	0
4	−1	−1	0	0	0
5	−1	0	0	−1	0

$\tilde{T}_5$	6	7	(8,10)	(6,11)	(7,11)
1	1	1	1	1	1
3	2	2	1	1	1

Otherwise, let T be the tableau consisting of the rows of $\bar{T}$ indexed by $I^0=\{i\epsilon I|\bar{x}_i=0\}$. Let $\hat{a}_j$, $j\epsilon J$, be the columns of T. Go to MCGP.

MCGP (*modified column-generating procedure*). At the kth iteration of MCGP, let T be the current tableau, with column index set J.

1. If any of the following three situations holds: (a) all rows have been marked, (b) $c_j\geqq 0$ for all $j\epsilon J$, or (c) $\bar{a}_{ij}\leqq 0$ for all $j\epsilon J$ and all unmarked rows i, then stop: the solution $\bar{x}$ is optimal for (P). Otherwise, let

$$\bar{c}_t=\min_{j\epsilon J}\{\bar{c}_j|\bar{a}_{ij}>0 \text{ for at least one unmarked row } i\}.$$

Choose an unmarked row r such that $\bar{a}_{rt}>0$ and go to 2.

2. This is step 2 of the general procedure CGP (*see* section 2) with step 2c(ii) replaced by:

(ii) Order S_t so that $\bar{c}_j<\bar{c}_k\Rightarrow j<k$. Define $J^0\subset S_t$ to be the subset of the $j\epsilon S_t$ such that $\bar{c}_j+\bar{c}_t<0$ and

$$\bar{a}_{hj}+\bar{a}_{ht}=\begin{cases}0 \text{ or } 1, & h\epsilon I-I^0,\\ 0 \text{ or } -1, & h\epsilon I^0.\end{cases} \quad (3')$$

(iii) If $J^0\neq\emptyset$, let j be the smallest index in J^0, define $Q_k=Q_j\cup Q_t$, and go to Block Pivot.

(iv) If $J^0=\emptyset$, for each $j\epsilon S_t$ add a new column $\hat{a}_j+\hat{a}_t$ to T and a new index k to J, where $Q_k=Q_j\cup Q_t$; then remove $\hat{a}_t$ from T and t from J. Call the resulting tableau T' and the resulting column index set J'. Go to 3.

3. Designate T' and J' to be the current tableau T and index set J, respectively, and return to 1.

Block Pivot. In the simplex tableau $\bar{T}$ associated with $\bar{x}$, pivot into the basis each column $j\epsilon Q_k$. The resulting solution is integer and better than $\bar{x}$; the associated tableau is integer. Go to Primal.

Algorithm II

Primal. This is like Algorithm I, with the following differences:

(a) Everything applies to the *current* problem $(\bar{P}')$ rather than to (P'). At the start $(\bar{P}')=(P')$, later $(\bar{P}')$ contains new columns and does not contain some (or all) of the original columns. If $\bar{c}_j\geqq 0$, $\forall j\epsilon J$, then the n-vector associated in the obvious way with $\bar{x}$ (rather than $\bar{x}$ itself) is an optimal solution to (P).

(b) The last sentence should read: Go to CGP.

CGP (*column-generating procedure*). This is the general CGP of Section 2, with the following amendment to step 4:

Also remove from $\bar{T}$ all columns $\bar{a}_j$ such that $\bar{c}_j\geqq 0$. If no columns are left in the tableau, the n-vector associated in the obvious way with $\bar{x}$ is an optimal solution. Otherwise, go to the problem associated with the new tableau $(\bar{P}')$ and go to Primal.

Both algorithms find an optimal solution to (P) in a finite number of steps. For Algorithm I, this follows from Theorem 5: each time MCGP is used, either the current solution is found to be optimal (because $\bar{c}_j\geqq 0$, $\forall j\epsilon J$ or because no further composite columns can be generated), or a composite column is generated which defines a vertex of X_I adjacent to and better than the current one. For Algorithm II, it follows from Theorems 4 and 5: the latter one guarantees that CGP generates all vertices of X_I adjacent to the current vertex $\bar{x}$, hence in particular all vertices adjacent to and better than $\bar{x}$, whereas the former guarantees that the tableau consisting solely of composite columns defining such vertices contains all the columns needed to produce an optimal solution.

4. NUMERICAL EXAMPLE

In this section we solve an example by Algorithm I.

Table III gives the vector c and the matrix A for the example.

TABLE III
The Numerical Example

$$c=(5,\ 4,\ 3,\ 2,\ 2,\ 2,\ 3,\ 1,\ 2,\ 2,\ 1,\ 1,\ 1,\ 0,\ 0)$$

$$A=\begin{pmatrix}1&0&0&0&0&1&1&1&1&0&0&0&1&1&0\\1&1&0&0&0&0&0&0&0&1&1&0&0&0&1\\0&1&1&0&0&1&1&1&0&0&0&1&0&0&0\\0&1&1&1&0&0&0&1&1&0&1&0&0&0&0\\0&0&1&1&1&0&1&0&0&1&0&1&1&0&0\end{pmatrix}$$

The simplex tableau $\bar{T}$:

	1	$-x_1$	$-x_6$	$-x_7$	$-x_8$	$-x_9$	$-x_{10}$	$-x_{11}$	$-x_{12}$	$-x_{14}$	$-x_{15}$
z	−5	3	2	1	1	3	−3	−2	−2	1	−3
x_{13}	1	1	1	1	1	1	0	0	0	1	0
x_2	1	1	0	0	0	0	1	1	0	0	1
x_3	0	−1	1	1	1	0	−1	−1	1	0	−1
x_4	0	0	−1	−1	0	1	0	1	−1	0	0
x_5	0	0	−1	0	−2	−2	2	0	1	−1	1

The simplex tableau $\bar{T}'$:

	1	$-x_1$	$-x_3$	$-x_6$	$-x_7$	$-x_9$	$-x_{11}$	$-x_{12}$	$-x_{13}$	$-x_{14}$	$-x_{15}$
z	−3	8	−3	1	0	5	1	−5	2	3	0
x_{10}	1	2	−1	0	0	1	1	−1	1	1	1
x_2	0	−1	1	0	0	−1	0	1	−1	−1	0
x_8	1	1	0	1	1	1	0	0	1	1	0
x_4	0	0	0	−1	−1	1	1	−1	0	0	0
x_5	0	−2	2	1	2	−2	−2	3	0	−1	−1

Primal produces the simplex tableau $\bar{T}$ of Table III, in which no more pivots are possible on an element $\bar{a}_{ij}=1$ (such that $\bar{c}_j<0$).

MCGP. $I^0=\{3,4,5\}$; $J=\{1,6,\cdots,12,14,15\}$.

First iteration: No rows marked. $\bar{c}_t=-3$, $t=10$; choose $i=5$. $S_{10}^1=\{6,8,9,14\}$; $S_{10}=S_{10}^1 = S^1_{\cdot\cdot}$; $S_{10}=\{8,14,6,9\}$. $\bar{c}_8+\bar{c}_{10}=1-3=-2<0$, and $\bar{a}_8+\bar{a}_{10}$ satisfies (3′).

Block Pivot introduces x_8 and x_{10} into the basis, replacing $\bar{T}$ by the simplex tableau $\bar{T}'$ of Table III.

Primal sends to the next step.

MCGP. $I^0=\{2,4,5\}$, $J=\{1,3,6,7,9,11,\cdots,15\}$.

First iteration: No rows marked. $\bar{c}_t=-5$, $t=12$; choose $i=2$. $S_{12}=\{14, 9, 1\}$. $\bar{c}_{14}+\bar{c}_{12}=3-5=-2<0$, but $\bar{a}_{5,12}+\bar{a}_{5,14}=3-1=2$, violates (3′). $\bar{c}_9+\bar{c}_{12}=5-5\geq 0$.

Remove column 12 and add columns (14, 12), (9, 12), and (1, 12) to T^1 (not shown in Table III) to obtain the tableau T^2 of Table IV.

Second iteration: No rows marked. $\bar{c}_t=-3$, $t=3$; choose $i=2$. $S_3=\{14, 1\}$.

Remove column 3 and add columns (14, 3), (1, 3) to T^2 to obtain T^3 as shown in Table IV.

Third iteration: The row with index $i=2$ is marked. $\bar{c}_t=-2$, call $\{14, 12\}=Q_{16}$, so $t=16$. Choose $i=5$. $S_{16}=\{15, 11\}$. $\bar{c}_{11}+\bar{c}_{16}=1-2<0$, and $\bar{a}_{11}+\bar{a}_{16}$ satisfies (3′).

TABLE IV

Tableaux for the Numerical Example

		1	3	6	7	9	11	13	14	15	(14,12)	(9,12)	(1,12)
	$\bar{c}_j$	8	−3	1	0	5	1	2	3	0	−2	0	3
T^2:	2	−1	1	0	0	−1	0	−1	−1	0	0	0	0
	4	0	0	−1	−1	1	1	0	0	0	−1	0	−1
	5	−2	2	1	2	−2	−2	0	−1	−1	2	1	1

		1	6	7	9	11	13	14	15	(14,12)	(9,12)	(1,12)	(14,3)	(1,3)
	$\bar{c}_j$	8	1	0	5	1	2	3	0	−2	0	3	0	5
T^3:	2	−1	0	0	−1	0	−1	−1	0	0	0	0	0	0
	4	0	−1	−1	1	1	0	0	0	−1	0	−1	0	0
	5	−2	1	2	−2	−2	0	−1	−1	2	1	1	1	0

The simplex tableau $\tilde{T}''$:

	1	$-x_1$	$-x_2$	$-x_3$	$-x_4$	$-x_6$	$-x_7$	$-x_8$	$-x_9$	$-x_{13}$	$-x_{15}$
z	−2	5	4	1	−1	3	2	1	1	−1	0
x_{10}	0	1	0	−1	−1	0	0	−1	−1	0	1
x_{12}	1	0	1	1	0	1	1	1	0	0	0
x_{14}	1	1	0	0	0	1	1	1	1	1	0
x_{11}	1	0	1	1	1	0	0	1	1	0	0
x_5	0	−1	−1	1	2	−1	0	0	1	1	−1

Block Pivot introduces x_{12}, x_{11}, and x_{14} into the basis and produces the simplex tableau $\tilde{T}''$ of Table IV.

Primal sends to the next step.

MCGP. $I^0=\{10, 5\}$, $J=\{1, \cdots, 4, 6, \cdots, 9, 13, 15\}$.

First iteration: No rows marked. $\bar{c}_t=-1$, $t=13$; choose $i=5$. $S_{13}=\{15, 2\}$. The next tableau is T^2 of Table V.

Second interation: No rows marked. $\bar{c}_t=-1$, call $\{15, 13\}=Q_{16}$, so $t=16$. Choose $i=10$. $S_{16}=\emptyset$. Remove column 16 to obtain T^3 of Table V.

Third iteration: No rows marked. $\bar{c}_t=-1$, $t=4$; choose $i=5$. $S_4=\{15, 6, 1\}$. Remove column 4 and add columns (15, 4), (6, 4), and (1, 4) to T^3 to obtain T^4 of Table V.

Fourth iteration: No rows marked. $\bar{c}_t=-1$, call $\{15, 4\}=Q_{17}$, so $t=17$. Choose $i=5$. $S_{17}=\{6\}$. Remove column 17 and add column (6, 15, 4) to obtain T^5 with $\bar{c}_j\geq 0$, $\forall j\epsilon J$.

Hence the solution $x_{11}=x_{12}=x_{14}=1$, $x_j=0$ for $j\neq 11$, 12, 14 is optimal.

ACKNOWLEDGMENTS

We wish to thank David S. Rubin for his helpful comments, and in particular for supplying the basic idea for the present version of part (i) of the proof of Theorem 3, which is both shorter and more appealing than the previous version.

TABLE V

Additional Tableaux for the Numerical Example

		1	2	3	4	6	7	8	9	15	(15,13)	(2,13)
	$\bar{c}_j$	5	4	1	−1	3	2	1	1	0	−1	3
T^2:	10	1	0	−1	−1	0	0	−1	−1	1	1	0
	5	−1	−1	1	2	−1	0	0	1	−1	0	0

		1	2	3	4	6	7	8	9	15	(2,13)
	$\bar{c}_j$	5	4	1	−1	3	2	1	1	0	3
T^3:	10	1	0	−1	−1	0	0	−1	−1	1	0
	5	−1	−1	1	2	−1	0	0	1	−1	0

		1	2	3	6	7	8	9	15	(2,13)	(15,4)	(6,4)	(1,4)
	$\bar{c}_j$	5	4	1	3	2	1	1	0	3	−1	2	4
T^4:	10	1	0	−1	0	0	−1	−1	1	1	0	−1	0
	5	−1	−1	1	−1	0	0	1	−1	0	1	1	1

We also wish to thank the National Science Foundation and the US Office of Naval Research for their support of the first author's research, as well as the Deutsche Forschungsgemeinschaft for its support of the second author's work.

REFERENCES

1. E. Balas and M. Padberg, "On the Set-Covering Problem," *Opns. Res.* **20**, 1153–1161 (1972).

2. ——— AND ———, "Adjacent Vertices of the Convex Hull of Feasible 0-1 Points," Management Sciences Research Report #298, November 1972–April 1973, Carnegie-Mellon University (to appear in *SIAM J. Appl. Math.*).
3. C. BERGE, "Balanced Matrices," *Mathematical Programming* **2**, 19–31 (1972).
4. E. BURGER, "Über homogene lineare Ungleichungs-Systeme," *Zeitschrift für Angewandte Mathematik und Mechanik* **36**, 135–195 (1956).
5. N. V. CHERNIKOVA, "Algorithm for Finding a General Formula for the Nonnegative Solutions of a System of Linear Equations," *USSR Computational Mathematics and Mathematical Physics* **4**, 157–158 (1964).
6. ———, "Algorithm for Finding a General Formula for the Nonnegative Solutions of a System of Linear Inequalities," *USSR Computational Mathematics and Mathematical Physics* **5**, 228–233 (1965).
7. R. GARFINKEL AND G. NEMHAUSER, "The Set-Partitioning Problem: Set Covering with Equality Constraints," *Opns. Res.* **17**, 848–856 (1969).
8. M. PADBERG, "On the Facial Structure of Set Packing Polyhedra," *Mathematical Programming*, **5**, 199–215 (1973).
9. ———, "Perfect Zero-One Matrices," *Mathematical Programming*, **6**, 180–196 (1974).
10. ——— AND M. R. RAO, "The Travelling Salesman Problem and A Class of Polyhedra of Diameter Two," IIM Preprint No. I/73-5, International Institute of Management, Berlin, Germany.
11. D. S. RUBIN, "Neighboring Vertices on Convex Polytopes," Graduate School of Business Administration, University of North Carolina at Chapel Hill, March 1972.
12. V. A. TRUBIN, "On a Method of Solution of Integer Programming Problems of a Special Kind," *Soviet Math. Dokl.* **10**, 1544–1546 (1969).

OPERATIONS RESEARCH, Vol. 23, No. 1, January-February 1975

Solving Constrained Transportation Problems

D. Klingman

University of Texas, Austin, Texas

and

R. Russell

University of Tulsa, Tulsa, Oklahoma

(Received March 6, 1973)

This paper presents a specialized method for solving transportation problems with several additional linear constraints. The method is basically the primal simplex method, specialized to exploit fully the topological structure embedded in the problem. It couples the poly-ω technique of CHARNES AND COOPER with the row-column sum method to yield an 'inverse compactification' that minimizes the basis information to be stored between successive iterations, and in addition minimizes the arithmetic calculations required in pivoting. In particular, the solution procedure only requires the storage of a spanning tree and a $(q+1)\times q$ matrix (where q is the number of additional constraints) for each basis. The steps of updating costs and finding representations reduce to a sequence of simpler operations that utilize fully the triangularity of the spanning tree. Procedures for obtaining basic primal 'feasible' starts are also presented.

THIS PAPER presents a specialized method for solving transportation problems with several extra linear constraints. Such linear models occur frequently in transportation applications. The warehouse-funds-flow model and the gas-blending model developed by CHARNES AND COOPER[3] are specific applications that are transportation models with additional constraints. Some scheduling models, such as a constrained version of WAGNER's employment-scheduling problem,[25] also fall into this class of problems.

Operations-research literature contains a number of ingenious techniques for transforming a transportation problem with an extra constraint(s) into a larger equivalent transportation problem; for instance, the early works by MANNE [reference 6, pp. 382–383], HADLEY,[16] SIMMONARD,[23] Charnes and Cooper,[3] and the current work by Wagner,[26] GLOVER, KLINGMAN, AND ROSS,[15] CHARNES, GLOVER, AND KLINGMAN,[4] and Charnes and Klingman[5] are indicative of the interest in this problem. However, these transformations are not possible with an arbitrary extra constraint.

This paper develops a solution procedure that exploits fully the topological structure embedded in this problem; it is basically the primal simplex method specialized to take full advantage of the computational schemes and list structures[12,17,24] used in codifying the row-column sum method[6] and the dual method.[21] These specialized primal computer codes have typically solved pure transportation problems 150 times faster than state-of-the-art linear-programming codes.[14]