

COWLES FOUNDATION FOR RESEARCH IN ECONOMICS

AT YALE UNIVERSITY

Box 2125 Yale Station
New Haven, Connecticut 06520

COWLES FOUNDATION DISCUSSION PAPER NO. 649

Note: Cowles Foundation Discussion Papers are preliminary materials circulated to stimulate discussion and critical comment. Requests for single copies of a paper will be filled by the Cowles Foundation within the limits of supply. References in publications to discussion papers (other than mere acknowledgement by a writer that he has access to such unpublished material) should be cleared with the author to protect the tentative character of these papers.

SCARF'S PROCEDURE FOR INTEGER PROGRAMMING AND
A DUAL SIMPLEX ALGORITHM

by

Philip M. White

Andrew S. Caplin

Ludo Van der Heyden

November 1982

SCARF'S PROCEDURE FOR INTEGER PROGRAMMING
AND A DUAL SIMPLEX ALGORITHM¹

by

Philip M. White^{2,4}

Andrew S. Caplin²

Ludo Van der Heyden^{3,5}

¹We would like to thank Herbert Scarf of Yale University for his encouragement and inspiration. We would also like to thank John Geanakoplos of Yale University and Michael Todd of Cornell University for their comments and suggestions.

²Department of Economics, Yale University, New Haven, CT 06520

³School of Organization and Management, Yale University, New Haven, CT 06520

⁴Support from the Social Sciences and Humanities Research Council of Canada is gratefully acknowledged.

⁵Support from the Cowles Foundation and from the U.S. Office of Naval Research (Contract Number N00014-77C-0518) are gratefully acknowledged.

Abstract

Herbert Scarf has recently introduced an algorithm for integer programs based on the concept of primitive sets. We show that as the choice variables become continuous, this algorithm converges to a dual simplex algorithm. This result is robust in the sense that even before the limit is reached, the simplex path is contained in the primitive sets which define Scarf's path to the solution of the integer program.

Key words: dual simplex algorithm

integer programming

linear programming

primitive sets

Scarf's algorithm

I. Introduction

In a recent paper, Herbert Scarf [1981a] has developed an algorithm for solving discrete programming problems. The fundamental objects in this approach are primitive sets which were first introduced in the computation of competitive equilibria [Scarf, 1973]. The algorithm traverses a sequence of such sets. The terminal primitive set on this path is the first to contain a point satisfying all the constraints of the programming problem. Properties of the primitive set then ensure that this point is optimal.

Important examples of discrete programming problems are linear integer programs for which the production set is generated by an activity analysis matrix with the activity levels restricted to be integral. These programs have the form:

$$\begin{aligned} \max \quad & a_0 x \\ \text{s.t.} \quad & a_j x \geq b_j, \quad j = 1, 2, \dots, m, \\ & x \text{ integer} \end{aligned}$$

The difficulty in applying Scarf's algorithm to this problem lies in describing the movement between primitive sets. Primitive sets have so far been characterized only for problems of low dimension [Scarf, 1981b and 1982] so that for larger problems the algorithm is still conceptual.

When the requirement that the activity vector x be integral is replaced by the requirement that x be rational with fixed denominator h , the set of feasible points is enlarged. For each grid size h , there is a sequence of primitive sets which leads to the solution of the corresponding problem. In the limit as h becomes large, the integer program reduces to

the corresponding linear program i.e., the programming problem with the integral restriction omitted.

In this paper we examine the properties of Scarf's integer programming algorithm in the limit. We show that as the grid is refined the algorithm becomes a particular dual simplex algorithm. In general, dual simplex procedures monotonically reduce the objective function while at any point violating a number of constraints, $j_1 < j_2 < \dots < j_p$. The special property of our dual simplex algorithm is the monotonicity in its resolution of the constraints. The index of the last violated constraint, j_p , is monotonically nonincreasing and while j_p is constant the infeasibility in constraint j_p decreases.

Our results demonstrate a strong relationship between the path of Scarf's integer programming algorithm and the path for the limit linear programming problem. We show that the limit path is covered by the path of primitive sets for the integer program regardless of grid size. Furthermore each of the primitive sets in the sequence which solves the integer program intersects the limit path. Thus even before going to the limit the integer programming algorithm never strays far from this variant of the dual simplex algorithm.

Our results support Scarf's approach in several ways. They attribute robustness to the integer programming algorithm by showing that it leads to an efficient algorithm for the limit linear programming problem. This raises the hope that the integer programming algorithm may inherit some of the success of its limit analogue.

The results also provide a natural initialization of the integer programming procedure. After solving the linear program, the integer programming

procedure could be started from a primitive set containing the linear programming solution. Our results indicate that such a primitive set exists.

Furthermore the simplex method holds interest for economists due to its interpretation as a price-guided mechanism for identifying an optimal production plan. In the presence of indivisibilities, prices no longer suffice. A neighborhood structure defined by primitive sets provides an alternative mechanism to verify optimality. The fact that primitive sets converge in the limit to dual feasible prices suggests that this structure may prove valuable in analyzing economic problems involving indivisibilities.

The paper is organized as follows. Section 2 contains a brief description of Scarf's integer programming algorithm. Section 3 introduces the simplex-type procedure to which Scarf's algorithm converges and the path generated by it. Section 4 contains our main results relating the limit path to the path of almost completely labeled primitive sets. Finally, section 5 shows that the monotonicity properties of the limit path are inherited from the path of almost completely labeled primitive sets. Throughout this paper the geometry of both algorithms will be emphasized with the hope that such exposition more clearly exhibits their relationship.

II. Primitive sets, labeling, and Scarf's algorithm

We now turn to a brief description of Scarf's algorithm¹ for solving the integer program

¹We refer to Scarf [1981a] for a detailed exposition.

$$\begin{aligned} \max \quad & a_0 x \\ \text{s.t.} \quad & a_j x \geq b_j, \quad j = 1, 2, \dots, m \\ \text{and} \quad & x \in Z^n \end{aligned}$$

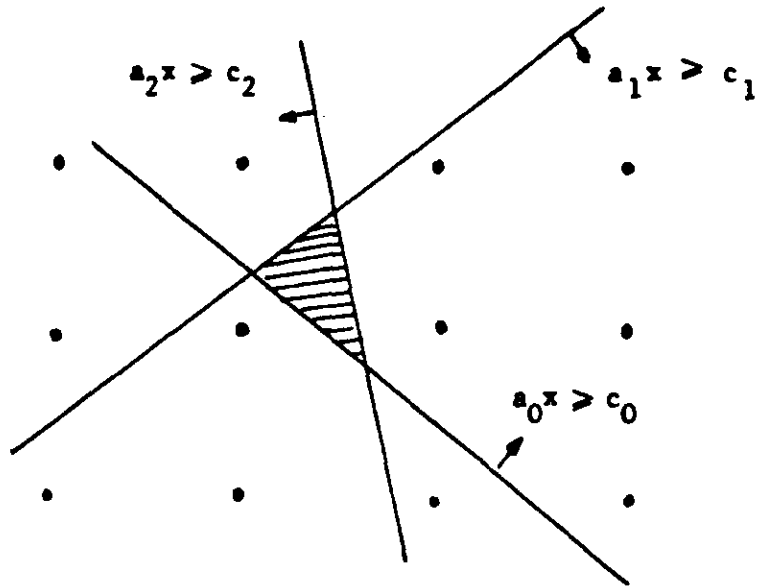
where Z^n is the n-dimensional integer lattice.

In what follows, we make a boundedness assumption which ensures that the problem either has a finite maximum or is infeasible.

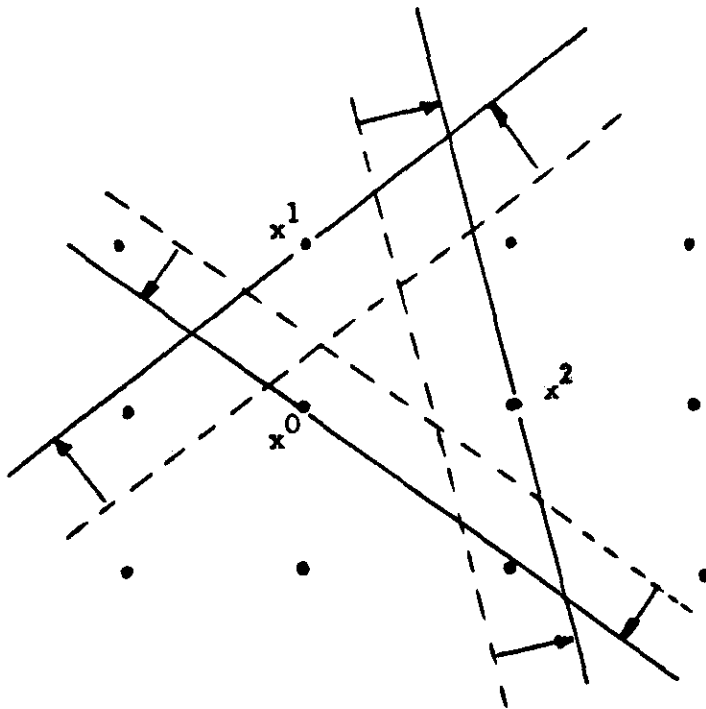
2.1. Boundedness Assumption: For any placement c_j , $j = 0, 1, \dots, m$, of the inequalities, the set of points enclosed by the inequalities $a_j x \geq c_j$, $j = 0, 1, \dots, m$, is bounded.

The three components of the algorithm are: (i) the definition of a primitive set, (ii) a replacement operation which defines movement between adjacent primitive sets, and (iii) a labeling rule for the lattice points which specifies a particular sequence of primitive sets followed by the algorithm to find the optimum.

The definition of a primitive set is derived from the following constructive procedure. Place the inequalities $a_j x \geq c_j$, $j = 0, 1, \dots, m$, so that the region they enclose is free of integer lattice points. Assumption (2.1) ensures that such a region exists. Relax back one of the inequalities by decreasing the value of its right hand side until it hits a lattice point which is accepted by the other inequalities. Leave this inequality in its new position. The new larger region still has no lattice points in its interior. Relax another inequality until it hits a lattice point which is accepted by the other inequalities. Continue the process by successively relaxing each of the inequalities to its own lattice point. This procedure, which is illustrated in diagram 1, identifies a particular primitive set of lattice points.



a. A lattice free region defined by the inequalities $a_i x \geq c_i$, $i = 0, 1, 2$.



b. The primitive set $\{x^0, x^1, x^2\}$ arising from a relaxation of these inequalities.

Diagram 1

In the course of this construction, two difficulties may arise. First, it may be possible to remove an inequality entirely without introducing any new lattice point. To deal with this case we introduce a set of ideal lattice points called slacks which allow us to say that every inequality relaxes to its own lattice point.

2.2. Definition: Slack Lattice Points: The lattice points in z^n are augmented by $m + 1$ slacks s^0, s^1, \dots, s^m such that $a_i s^i = -\infty$ and $a_j s^i = +\infty$ for $j \neq i$.

The second difficulty arises if, as an inequality relaxes, it simultaneously hits several lattice points which are accepted by the other inequalities. It is readily verified that the following non-degeneracy assumption ensures that the relaxed inequality and the other inequalities admit simultaneously at most a finite number of lattice points.

2.3. Nondegeneracy Assumption: Every $n \times n$ submatrix of the matrix

$$A = \begin{bmatrix} a_0 \\ a_1 \\ \vdots \\ a_m \end{bmatrix}$$

has full rank.

With this assumption, we can ensure that the lattice point first encountered by an inequality is well-defined by adopting a lexicographic tie-breaking rule over the lattice points in the hyperplane $a_i x = c_i$.

2.4. Definition: Lexicographic Tie-breaking Rule: For any two points y and z , $a_i y$ is lexicographically larger than $a_i z$, denoted $a_i y >_{\text{lex}} a_i z$, if

(i) $a_i y > a_i z$

or (ii) $a_i y = a_i z$

and for some k , $0 \leq k \leq m$,

$$a_j y = a_j z, \quad j = 0, 1, \dots, k-1,$$

$$a_k y > a_k z.$$

To summarize the above discussion, we formally define a primitive set.

2.5. Definition: Primitive Set: A set $\{x^0, x^1, \dots, x^m\}$ of lattice points and slacks is a primitive set if there is no lattice point which satisfies all the inequalities $a_i x >_{\text{lex}} \min_{j=0,1,\dots,m} (a_i x^j)$ for $i = 0, 1, \dots, m$.

In a primitive set each inequality is placed at the lattice point where it attains its minimum, ties being broken lexicographically. Observe that each lattice point is thereby associated with precisely one inequality. Unless it is otherwise apparent, we shall adopt the notational convenience that the i th inequality attains its minimum on the primitive set at x^i .

A variety of different primitive sets may be obtained by commencing the construction with different lattice free regions or by performing the relaxations in a different order. The reader may convince himself that assumptions (2.1) and (2.3) guarantee that the constructive procedure as described is well-defined and will generate all possible primitive sets associated with the given matrix A .

The replacement operation defining movement between adjacent primitive sets is carried out as follows. An element, say x^i , of a primitive set $\{x^0, x^1, \dots, x^m\}$ is removed from the set by pressing in the i th inequality from x^i until it hits some other element, say x^k , of the primitive set. The replacement for x^i is determined by then relaxing the k th inequality until it hits a lattice point, y , which is strictly accepted by the other inequalities, including the i th in its new position at x^k . A new primitive set has been obtained from the old one by replacing x^i by y . It is useful to note that if y were now removed from the new primitive set, its replacement would be x^i . Thus the operation is reversible. The replacement operation is illustrated in diagram 2.

The final element necessary for the full description of the algorithm is a labeling of each lattice point with an integer from the set $\{0, 1, \dots, m\}$. Notice that definition (2.6) gives slack s^i label i .

2.6. Definition: Labeling: Each lattice point x is given an integer label $\ell(x)$ equal to the index of the last violated constraint at x .

That is $\ell(x) = i$ if $a_i x < b_i$ and $a_j x \geq b_j$ for $j > i$. If $a_j x \geq b_j$, $j = 1, 2, \dots, m$, then $\ell(x) = 0$.

The labeling associates with a primitive set $\{x^0, x^1, \dots, x^m\}$ a set of labels $\{\ell(x^0), \ell(x^1), \dots, \ell(x^m)\}$. The primitive set is said to be completely labeled if its set of labels is $\{0, 1, \dots, m\}$. We now briefly argue that in a completely labeled primitive set the lattice point with label 0 is the solution to the integer programming problem. By the nature of the labeling this lattice point is certainly feasible. The definition of a primitive set requires that there be no lattice point x satisfying all

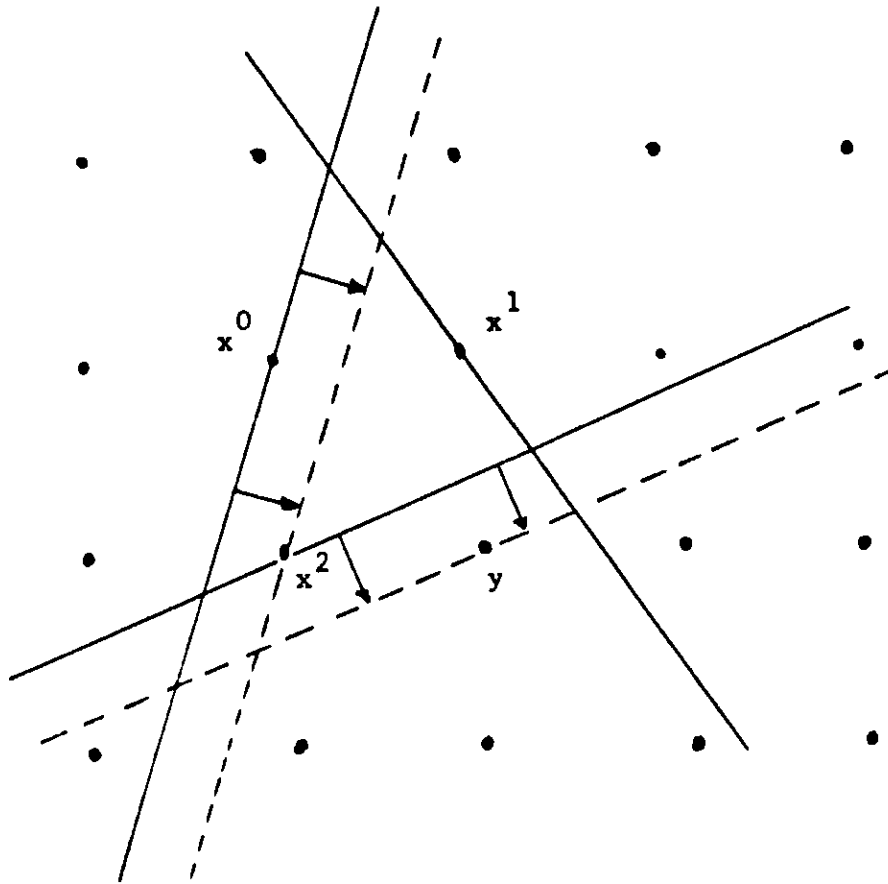


Diagram 2. The replacement for x^0 in the primitive set $\{x^0, x^1, x^2\}$ is y .

the inequalities $a_1 x >_{\text{lex}} a_1 x^1$. Since one of the points in the primitive set bears label i , $a_1 x^1 < b_i$. Hence no lattice point x satisfies $a_1 x \geq b_i$, $i = 1, 2, \dots, m$, and $a_0 x >_{\text{lex}} a_0 x^0$. This shows that $\ell(x^0) = 0$ and that x^0 is the solution to the integer programming problem.

In Scarf's algorithm the search for a completely labeled primitive set proceeds through a sequence of primitive sets, all of which miss only label 0 . Such primitive sets are said to be almost completely labeled. In such a set two lattice points share a label. At a typical position the algorithm has just introduced one of these lattice points and now removes the other according to the replacement operation described above. The algorithm terminates if the incoming lattice point bears label 0 . If not, a new almost completely labeled primitive set is reached and the algorithm continues by removing from the primitive set the lattice point which shares a label with the incoming point.

As demonstrated in Scarf [1981a], the family of all almost completely primitive sets is ordered by this procedure into a single sequence leading to the unique completely labeled primitive set.

III. The Limit Procedure

To motivate the limit algorithm, we provide a heuristic discussion of primitive sets and labeling for the linear programming problem. Recall from section II that for a placement of a subset of the inequalities to define a primitive set it is necessary that no feasible activity vector be strictly accepted by all inequalities. In the limit all points are feasible and satisfaction of this condition forces the placement to define a region

with nonempty interior. Thus under the nondegeneracy assumption the geometric picture of the limit analogue is a subset of the inequalities placed through a given point and accepting only that point. The inequalities not involved in this picture are at their respective slacks.

Under the nondegeneracy assumption the condition that a group of inequalities placed at a single point accept no other point is equivalent to the existence of a positive linear combination of the normals of these inequalities which equals zero. As a special case, when the subset of inequalities includes the objective function and exactly n others, this corresponds to a dual feasible basis familiar from the simplex algorithm.

We shall extend the standard definition of dual feasible basis to include any $n + 1$ rows of the matrix A which when placed through a point form a limit primitive set.

3.1. Definition: Dual feasible basis: $n + 1$ rows j_0, j_1, \dots, j_n of the matrix A form a dual feasible basis if there exists a positive vector $p = (p_{j_0}, p_{j_1}, \dots, p_{j_n})$ such that

$$p_{j_0} a_{j_0} + p_{j_1} a_{j_1} + \dots + p_{j_n} a_{j_n} = 0 .$$

We later make use of the fact that when some additional inequality is introduced into a dual feasible basis, the nondegeneracy assumption ensures that one and only one of the other inequalities can be dropped while maintaining the remaining inequalities as a dual feasible basis. This operation, which is the limit form of the replacement of one slack by another in a primitive set, corresponds to a standard dual simplex pivot step.

As the grid size is refined the nonslack lattice points in a primitive set for the discrete problem become a single point with several inequalities passing through it. For this single point to maintain the label properties of primitive sets, the point must inherit all labels found in primitive sets converging to it. A limit point is therefore associated not only with its own label, but also with the labels of points arbitrarily close to it.

Now consider a limit point, y , and let $0 < j_1 < j_2 < \dots < j_m$ be the indices of the inequalities passing through y . The label set of y must contain all labels but 0. Any neighborhood of y therefore contains points with labels j_1, j_2, \dots, j_m . The presence in y 's neighborhood of points with label j_1 implies that y satisfies all constraints $j > j_1$ and either violates constraint j_1 or satisfies it with equality. The requirement that y 's label set also contain the indices j_2, j_3, \dots, j_m then forces the corresponding constraints to be met at y with equality.

This discussion motivates the definition of the limit path which in later sections we demonstrate to have an intimate relationship with the discrete algorithm.

3.2. Definition: Limit Path: The limit path consists of all points y for which there exist n constraints $j_1 < j_2 < \dots < j_n$ such that

(i) y satisfies all constraints with index greater than j_1 :

$$a_j y \geq b_j \text{ for all } j > j_1 ;$$

(ii) y exactly satisfies constraints j_2, j_3, \dots, j_n and either violates or exactly satisfies constraint j_1 :

$$a_{j_1} y = b_{j_1} \text{ for } i = 2, 3, \dots, n ,$$

$$a_{j_1} y \leq b_{j_1} ;$$

(iii) Inequalities $0, j_1, j_2, \dots, j_n$ form a dual feasible basis.

We now demonstrate that the limit path consists of a connected series of line segments leading to the solution of the linear program. Consider tracing the path by moving from a representative point, y , toward the solution. By definition y lies on the intersection of $a_{j_1} x = b_{j_1}$ for $i = 2, 3, \dots, n$ and satisfies the constraint $a_j x \geq b_j$ for $j > j_1$. When inequalities $0, j_1, j_2, \dots, j_n$ are placed through y there are no other points accepted. We now perturb this system of inequalities by increasing the value of inequality j_1 and decreasing the objective function. The solutions for these inequalities lie on the line along which the equality constraints $a_{j_1} x = b_{j_1}$ for $i = 2, 3, \dots, n$ intersect. This line is followed until one of two situations arises:

(i) The path moves all the way to the intersection of the hyperplanes $a_{j_1} x = b_{j_1}$ for $i = 1, 2, \dots, n$ without violating any constraint j greater than j_1 . At this stage, we have solved the linear program obtained by deleting constraints $1, 2, \dots, j_1-1$. If in addition none of these constraints is violated then the intersection is the solution to the entire problem and the path ends. On the other hand, if some constraint with index less than j_1 is violated, then the violated constraint with highest index, j_0 , is introduced into the dual feasible basis. One of the other inequalities drops out. If it is the objective function the problem is easily seen to be infeasible and the path terminates.

Otherwise some inequality, j_k , drops out and the path continues along the intersection of $a_{j_1} x = b_{j_1}$ for $i = 1, \dots, k-1, k+1, \dots, n$ increasing the value of inequality j_0 and decreasing the value of the objective function. For the path to continue this movement must satisfy constraint j_k . To verify this, note that the replacement of inequality j_k with j_0 in the dual feasible basis implies that inequalities j_k and j_0 increase in the same direction along the line $a_{j_1} x = b_{j_1}$ for $i = 1, \dots, k-1, k+1, \dots, n$. Since the current position is left by increasing the value of inequality j_0 , the value of inequality j_k must also increase, so that constraint j_k remains satisfied.

(ii) The second possibility is that as the point moves along the line $a_{j_1} x = b_{j_1}$, $i = 2, 3, \dots, n$, some inequality, say j_0 with $j_0 > j_1$, becomes exactly satisfied. The path cannot move further along the line. Instead inequality j_0 is brought into the dual feasible basis. If the objective function drops out of the dual feasible basis, then the problem is infeasible and the path terminates. Otherwise some other inequality drops out. This inequality cannot be inequality j_1 for along the line $a_{j_1} x = b_{j_1}$ for $i = 2, 3, \dots, n$, inequalities j_0 and j_1 increase in opposite directions. Thus j_0 is not a candidate to replace j_1 in the dual feasible basis $0, j_1, j_2, \dots, j_n$. Therefore a constraint j_k with $k \geq 2$ drops out and the limit path follows the line $a_{j_1} x = b_{j_1}$, $i = 0, 2, 3, \dots, k-1, k+1, \dots, n$ increasing the value of inequality j_1 and decreasing the value of the objective function. An argument similar to that above demonstrates that moving in this direction keeps constraint j_k satisfied.

The limit procedure corresponds to a variant of the ordinary dual simplex method and can be carried out with a sequence of dual simplex tableaux containing $m + 1$ columns and $n + 1$ rows. At a typical position in the ordinary dual simplex method, there is a dual feasible basis including the objective function. The algorithm rests at the intersection of the constraints in the basis. An iteration involves introducing a violated constraint into the dual feasible basis and moving to the new intersection. In the course of this step, no attention is paid to the violation of constraints outside the basis. In contrast, the limit procedure in increasing the value of the last violated constraint changes direction when it is about to violate a constraint with higher index.

The procedure may also be viewed as an application of Van der Heyden's algorithm for the linear complementarity problem [Van der Heyden, 1980]. Ravindran's numerical experiments suggest that from a computational viewpoint complementarity algorithms for linear programming compare well with ordinary simplex methods [Ravindran, 1973]. The limit procedure differs from the complementarity algorithms in Ravindran's paper only in the order in which it treats the constraints.

IV. The Limit Path and the Almost Completely Labeled Path

We now turn to our main results which illuminate the links between the limit path and the sequence of almost completely labeled primitive sets.

4.1. Definition: Let $\{x^0, x^1, x^2, \dots, x^m\}$ be a primitive set. A point y is said to be contained in this primitive set if $a_i y \geq a_i x^i$, $i = 0, 1, 2, \dots, m$.

4.2. Theorem: Every point on the limit path is contained inside a completely labeled or an almost completely labeled primitive set.

Proof: Let y be on the limit path. Then there are inequalities

$j_1 < j_2 < \dots < j_n$ such that $a_j y \geq b_j$ for $j > j_1$ and y is the only point satisfying $a_j x \geq a_j y$ for $j = 0, j_1, j_2, \dots, j_n$ where $a_{j_i} y = b_{j_i}$ for $i = 2, 3, \dots, n$.

Consider the placement of the inequalities

$$\begin{aligned} a_0 x &\geq a_0 y + \epsilon & , & \quad \epsilon > 0 & , \\ a_j x &\geq b_j & , & \quad j \neq 0, j_1 & , \\ a_{j_1} x &\geq a_{j_1} y & . & & \end{aligned}$$

The region defined by these inequalities is empty. Construct a primitive set by relaxing the inequalities in the order $1, 2, 3, \dots, j_1-1, 0, j_1, j_1+1, \dots, m$. Inequalities 1 through j_1-1 relax back to their own slacks since the region enclosed by the remaining inequalities is empty. Thus the slacks s^i , $i = 1, 2, \dots, j_1-1$, are members of the primitive set carrying labels 1 through j_1-1 .

The zeroth inequality relaxes to a lattice point or slack, x^0 , which satisfies constraints j_1+1 through m . Thus, $0 \leq \ell(x^0) \leq j_1$. We also know that $a_0 y \geq a_0 x^0$ since the alternative would leave the region empty at this stage. Thus y is contained in the closed region defined by the inequalities in their position at this stage in the construction. This ensures that y is contained in the primitive set which is obtained by further relaxation of inequalities j_1 through m .

It remains to demonstrate that labels j_1 through m are elements of the label set. Inequality j_1 relaxes to x^{j_1} which satisfies

constraints j_1+1 through m but violates constraint j_1 since $a_{j_1} x^{j_1} < a_{j_1} y \leq b_{j_1}$. Thus $\ell(x^{j_1}) = j_1$. Similarly when inequality j is relaxed for any $j > j_1$ it relaxes to a lattice point or slack, x^j , which satisfies all constraints with index greater than j but violates constraint j . Thus $\ell(x^j) = j$.

The constructed primitive set is $\{x^0, s^1, s^2, \dots, s^{j_1-1}, x^{j_1}, x^{j_1+1}, \dots, x^m\}$. The $(m+1)$ tuple of labels of this primitive set is $\{\ell(x^0), \ell(s^1), \dots, \ell(s^{j_1-1}), \ell(x^{j_1}), \ell(x^{j_1+1}), \dots, \ell(x^m)\} = \{\ell(x^0), 1, 2, \dots, m\}$. The primitive set is completely labeled if $\ell(x^0) = 0$ and almost completely labeled if $\ell(x^0) > 0$. This completes the demonstration of Theorem (4.2).

It is interesting to note that the limit path is covered by a subset of the almost completely labeled primitive sets. In particular it is covered by those primitive sets for which x^0 takes the doubled label and the doubled label is less than or equal to the index of the first non-slack element aside from x^0 .

These comments suggests that there might be almost completely labeled primitive sets which do not contain an element of the limit path. The following theorem demonstrates that this suggestion is false.

4.3. Theorem: Every completely labeled or almost completely labeled primitive set contains a point of the limit path.

Proof: Let $\{x^0, s^1, s^2, \dots, s^{j-1}, x^j, x^{j+1}, \dots, x^m\}$ be an almost completely labeled or completely labeled primitive set for which apart from x^0 , which might be s^0 , x^j is the non-slack element with lowest index j .

Since any point x satisfies $a_1 x \geq a_1 s^1 = -\infty$, it suffices to show that there is a point of the limit path in the region defined by

$$\begin{aligned} \underline{4.4.} \quad & a_0 x \geq a_0 x^0 \quad , \\ & a_k x \geq a_k x^k \quad \text{for } k \geq j \quad . \end{aligned}$$

Since some element in the primitive set bears label k , we know that $a_k x^k < b_k$ for $k \geq j$. Thus $\ell(x^k) \geq k$. Since slacks take their own label, label j is taken by either x^j or x^0 .

Since $a_k x^k < b_k$, inequalities $j+1$ through m in (4.4) can be pressed in to their constraint position to obtain the smaller region

$$\begin{aligned} & a_0 x \geq a_0 x^0 \quad , \\ & a_j x \geq a_j x^j \quad , \\ & a_k x \geq b_k \quad \text{for } k > j \quad . \end{aligned}$$

This region is non-empty since either x^0 or x^j satisfies constraints $j+1$ through m , and by the definition of a primitive set $a_0 x^j \geq a_0 x^0$ and $a_j x^0 \geq a_j x^j$. Now the zeroth inequality can be pressed in until a single point y is left in the region. The point y is the solution to the linear program

$$\begin{aligned} & \max \quad a_0 x \\ & \text{s.t.} \quad a_j x \geq a_j x^j \quad , \\ & \quad \quad a_k x \geq b_k \quad \text{for } k > j \quad . \end{aligned}$$

This solution is associated with a dual feasible basis made up of the objective function and a subset $j_1 < j_2 < \dots < j_n$ of the inequalities j through m . $a_k y \geq b_k$ for $k > j$ and $a_k y = b_k$ for $k = j_2, j_3, \dots, j_n$. If $j_1 \neq j$ then $a_{j_1} y = b_{j_1}$. If $j_1 = j$ then $a_j y = a_j x^j < b_j$. In either case, y lies on the limit path and by construction y is contained in the primitive set, proving Theorem (4.3).

Theorems (4.2) and (4.3) demonstrate the relationship between the limit path and the integer programming algorithm. The theorems hold regardless of grid size. For any fixed grid size the union of all points contained in the sequence of almost completely labeled primitive sets covers the limit path and each of these primitive sets contains a point of the limit path. Thus as the grid becomes finer the set of points contained in the almost completely labeled primitive sets converges to the limit path.

V. Monotonicity in the Almost Completely Labeled Path

Our purpose in this section is to demonstrate that certain monotonicity features of the limit path may also be discovered in the almost completely labeled path of the discrete problem. These properties of the discrete path are surprising since the Lemke and Howson argument [Lemke and Howson, 1964] which establishes its convergence is purely combinatorial and does not involve monotonicity.

Recall that in the limit procedure, the violated constraint with highest index, j_1 plays a distinguished role. Along the path, this index is monotonically nonincreasing. The algorithm moves to solve the linear program obtained by discarding constraints 1 through j_1-1 , increasing the value of inequality j_1 while decreasing the value of the objective function. Having solved this subproblem, a new constraint with index less than j_1 replaces j_1 as the distinguished constraint.

In order to show that the above features are present even before passing to the limit, we first define the value of an inequality at a primitive set. It is natural to assign to each inequality the value it assumes in its primitive set placement. The following definition continues to adopt the convention that

over a primitive set $\{x^0, x^1, \dots, x^m\}$, $a_1 x$ attains its (lexicographic) minimum at x^1 .

5.1 Definition: Value of an inequality at a primitive set: The value of the i th inequality at the primitive set $\{x^0, x^1, \dots, x^m\}$ is $a_1 x^i$.

Scarf [1981a] has already established that the value of the objective function is nonincreasing along the almost completely labeled path. His demonstration of this result is based on the ability to orient the graph of almost completely labeled primitive sets by examination of the labels. Since one of the lattice points bears label k for $k \geq 1$, $a_k x^k < b_k$ and $\ell(x^k) \geq k$. This permits the definition of the increasing sequence of indices $0 = i_0 < i_1 < \dots < i_p$ such that $\ell(x^{i_h}) = i_{h+1}$, $h = 0, 1, \dots, p-1$, and $\ell(x^j) = j$ for $j = i_0, i_1, \dots, i_{p-1}$. $x^{i_{p-1}}$ and x^{i_p} share label i_p . At a typical position of the algorithm one of these two lattice points is about to be removed from the primitive set. To move in the direction of the completely labeled primitive set, $x^{i_{p-1}}$ is removed if p is even and x^{i_p} is removed if p is odd.

Given this orientation it becomes quite simple to demonstrate the monotonicity of the objective function in the discrete procedure. Recall that the value of the objective function at the primitive set $\{x^0, x^1, \dots, x^m\}$ is $a_0 x^0$. The value of the objective function increases only if x^0 is replaced in the primitive set. This requires that x^0 and x^{i_1} share a label. But in this situation the orientation of the almost completely labeled path requires the replacement of x^{i_1} and not x^0 . Therefore the value of the objective function does not increase in the discrete procedure.

To discuss the order in which constraints are resolved, we need to define the distinguished constraint at an almost completely labeled primitive set.

5.2. Definition: Distinguished Constraint: The distinguished constraint at an almost completely labeled primitive set is the constraint whose index is equal to the smallest of the labels of the nonslack lattice points in the primitive set.

Let $\{x^0, s^1, s^2, \dots, s^{k-1}, x^k, x^{k+1}, \dots, x^m\}$ be an almost completely labeled primitive set for which x^k is the non-slack vector with lowest index aside from x^0 . Since $\ell(x^j) \geq j$, and slacks take their own labels, x^0 or x^k must take label k . The index of the distinguished constraint equals k if $\ell(x^0) \geq k$ and equals $\ell(x^0)$ if $\ell(x^0) < k$.

To demonstrate that the index of the distinguished constraint never increases, it suffices to show that if only one nonslack point bears the label of the distinguished constraint, it is not removed from the primitive set. But if only one nonslack point bears this label and it is the doubled label, then $\ell(x^0) < k$ and x^0 shares the doubled label with a slack vector. In this case, the orientation requires the removal of the slack from the primitive set.

This establishes the first similarity in the constraint resolution between the discrete and limit procedures. We now demonstrate that the value of the distinguished constraint does not decrease. Furthermore, the index of the distinguished constraint decreases only when the subproblem obtained by deleting constraints 1 to $k-1$ has been solved.

Notice that if $\ell(x^0) < k$ then, as above, x^0 shares the doubled label with the $\ell(x^0)$ -slack, this slack is removed and the value of the distinguished constraint clearly increases.

If $\ell(x^0) \geq k$ then the distinguished constraint is k . A replacement step removes x^{i_h} for $h = p-1$ if p is even and $h = p$ if p is odd. In the course of the replacement, inequality i_h is pressed in until it hits another element, say x^j of the primitive set. The labeling of the elements in an almost completely labeled primitive set implies that either $j = i_{h-1}$ or $j > i_h$. The replacement of x^{i_h} then proceeds by relaxing inequality j . Since $1 \leq k \leq i_1$, inequality k could be relaxed during a replacement step only if $k = i_1$. But this requires that $h = 0$ or 2 , which is excluded by the rules governing the replacement step. Thus the value of the distinguished constraint is non-decreasing.

Let us consider the first time a replacement step introduces a lattice point, say x^* , satisfying constraints k through m . Thus $\ell(x^*) < k$ and the index of the distinguished constraint decreases. To introduce such a point x^{i_1} must have been replaced and in the course of the replacement step as inequality i_1 was pressed in, x^0 was the first point of the primitive set to be hit. Inequality 0 was subsequently relaxed until it hit x^* .

By the definition of a primitive set no lattice point satisfies all the inequalities:

$$\begin{aligned} a_0 x &>_{\text{lex}} a_0 x^* , \\ a_{i_1} x &>_{\text{lex}} a_{i_1} x^0 , \\ a_i x &>_{\text{lex}} a_i x^i , \quad i \neq 0, i_1 . \end{aligned}$$

Inequalities 1 to $k-1$ are at their own slacks. Since $a_i x^i < b_i$ for $i \geq k$ and $a_{i_1} x^0 < b_{i_1}$, no lattice point satisfies

$$\begin{aligned} a_0 x &>_{\text{lex}} a_0 x^* , \\ a_i x &\geq b_i , \quad i \geq k . \end{aligned}$$

Thus x^* solves the subproblem obtained by deleting the first $k-1$ constraints.

5.3. Theorem: Along the path of almost completely labeled primitive sets the index of the distinguished constraint, k , never increases. Furthermore, the value of the distinguished constraint never decreases and the index of the distinguished constraint decreases only when the subproblem obtained by discarding constraints 1 through $k-1$ has been solved.

REFERENCES

- Dantzig, G. B., [1963], Linear Programming and Extensions, Princeton University Press, Princeton, New Jersey.
- Ravindran, A., [1973], "A comparison of the primal-simplex and complementary pivot methods for linear programming," Naval Research Logistics Quarterly 20, pp. 95-100.
- Scarf, H. E., [1973], with the collaboration of T. Hansen, The Computation of Competitive Equilibria, Yale University Press, New Haven, CT.
- _____, [1981a], "Production Sets with Indivisibilities: Part I, Generalities," Econometrica 49, pp. 1-32.
- _____, [1981b], "Production Sets with Indivisibilities: Part II, The Case of Two Variables," Econometrica 49, pp. 395-423.
- _____, [1982], "Integral polyhedra in three space," Cowles Foundation Discussion Paper No. 632, Yale University, New Haven, CT.
- Van der Heyden, L., [1980], "A variable dimension algorithm for the linear complementarity problem," Mathematical Programming 19, pp. 328-346.