METU Studies in Development 12 (1-2) 1985, 131-155

# A PRIMER ON KARMARKAR'S ALGORITHM FOR LINEAR PROGRAMMING

Quirino PARIS*

(Received: September 1985)

A new algorithm for solving linear programs has been proposed recently by N. Karmarkar. Instead of hopping from extreme point to extreme point along the boundary of the solution set, as the simplex method does, Karmarkar's algorithm cuts across the set of feasible solutions. As a solution procedure for solving very large-scale linear programming problems, the new algorithm shows great potential. In this paper, Karmarkar's algorithm is described. Also, although Karmarkar's paper does not provide any explicit discussion of duality, it is shown that in this algorithm dual variables and dual relations are alive and well. Finally the degeneracy and multiple optimal solution issues are discussed with respect to the new algorithm.

## 1) INTRODUCTION

A new algorithm for solving linear programs has been proposed recently by N. Karmarkar. Its description is available in an undated, mimeographed paper. The algorithm promises to be a remarkable one for several reasons. First of all, it is a worst-case polynomial-time algorithm[1]. In principle, this property constrasts favorably with the simplex method which is a worst-case exponential-time algorithm. A word of caution, however, must be given before rushing to the conclusion

* Proffesor of Agricultural Economics at the University of California, Davis, USA.
The author wishes to acknowledge valuable discussions with Art Havenner, Gordon King, and Margherita D'Aprile, visiting professor of mathematics from the University of Calabria, Cosenza, Italy.

1 Worst-case computational complexity is defined as the upper bound on the time required to solve any LP problem of a given size. It is possible to construct LP problems for which the worst-case computational complexity of the simplex method is exponential in the number of variables, n, that is $(2^n-1)\,L^2$, where L is the size of the problem measured as the binary encoding of all coefficients. The worst-case computational complexity of Karmarkar's algorithm is $n^{3.5}L^2$, a polynomial expression.

that the era of the simplex method may be over. Although it is a worst-case exponential time algorithm, the simplex method has been shown to be a polynomial-time algorithm for average problems in the sense of Borgwardt. It is unlikely that, for these "average" problems, any new algorithm will dramatically improve the performance of the simplex method. Yet, Karmarkar has claimed that his algorithm has outperformed the simplex method by 50 times when tested on problems of several thousand variables.

The interest in the new algorithm is two-fold: as a radically new conception for approaching mathematical programming problems and as a routine for generating efficient numerical solutions to very large-scale problems.

As a new conception, the algorithm is as exciting as the simplex method itself. Instead of hopping from extreme point to extreme point along the boundary of the solution set, as the simplex method does, Karmarkar's algorithm cuts across the set of feasible solutions. This idea was pursued by researchers for over 30 years. In 1956, Ragnar Frish presented an algorithm for solving convex programming that, characteristically, he named "multiplex." His algorithm cuts across the set of feasible solutions and uses potential functions[2] as does Karmarkar's, but was not a success.

As a solution procedure for solving very large-scale linear programming problems, Karmarkar's algorithm shows great potential. The present unavailability of an efficient computer code compells us to postpone any assessment of this conjecture.

## 2) THE PRINCIPAL IDEAS

Karmarkar's main ideas are several. There is a fundamental projective transformation of the solution space into an affine space,[3] whose role is that of providing the recursive process of the algorithm.

There is a (repeated) "least-squares" projection of the objective function onto the transformed affine space whose role is to provide the direction of optimization.

There is an optimization over the intersection of the transformed solution space and a properly constructed sphere whose role is to

2 The potential function is the logarithm of the ratio of two linear functions.

3 An affine space is the result of a transformation that maps parallel lines into parallel lines.

determine the next feasible interior point. This sphere has a remarkable role. It essentially replaces the nonnegativity conditions on the primal variables and keeps all the successive feasible points away from the boundaries of the solution set. Another main idea of the algorithm, therefore, is to find a sequence of strictly interior points converging to an optimal solution of the transformed problem which, in turn, (via the inverse projective transformation) corresponds to an optimal solution of the original problem.

The overall result of these ideas is to provide a procedure which is not based upon basis changes, as required by the simplex method. The opposite view is, in fact, taken. all activities are active at the beginning and, in nondegenerate problems, (n-m) of them (n = number of activities, m = number of constraints) become gradually inactive as the algorithm converges to a solution.

What about duality? In principle, if the sequence of solution points describes a strictly interior path in the feasible solution set, it would seem that dual variables would not play any significant role. Indeed, Karmarkar's paper does not provide any explicit discussion of duality. It is possible to show, however, that also in this algorithm dual variables and dual relations are alive and well. The discussion of duality, in fact, will help in understanding the working of the algorithm.

### The Projective Transformation

Two types of projective transformations are employed in the algorithm. The first type is used only once, at the outset, to transform an inhomogeneous system of constraints into a homogeneous one.

Suppose we are given the following system of constraints

(1) $Q = \{x \mid Ax = b\}$

$P = \{x \mid x \geq 0\}$

with A, an (mxn) matrix of coefficients, of rank (A) = m. Suppose also that we know a strictly interior vector $x_o = (x_{o1}, \ldots, x_{on})$, that is $x_{oi} > 0$ $i = 1, \ldots, n$ and $Ax_o = b$.

Let us define a diagonal matrix $D_o = \text{diag}\,(x_o)$ and the vector e as the sum vector. Then, a projective transformation is defined as

(2) $x' = \dfrac{D_0^{-1} x}{e^T D_0^{-1} x + 1} > 0$

$x'_{n+1} = 1 - e^T x' > 0$

The properties of this projective transformation are as follows:

(a) It maps the positive orthant $P_n$ into the unit simplex,

$$S' = \{x' \mid e^T x' = 1, x' \geq 0\} \subset R^{n+1}.$$

(b) Furthermore,

$$e^T x' = \frac{e^T D_0^{-1} x}{e^T D_0^{-1} x + 1} = 1 - x'_{n+1}$$

$$e^T D_0^{-1} x = e^T D_0^{-1} x - x'_{n+1} e^T D_0^{-1} x + 1 - x'_{n+1}$$

$$0 = -x'_{n+1} (e^T D_0^{-1} x + 1) + 1$$

$$x'_{n+1} = \frac{1}{e^T D_0^{-1} x + 1}.$$

(c) The inverse projective transformation is given by

$$x = \frac{D_0 x'}{x'_{n+1}}.$$

In fact from (b),

$$e^T x' (e^T D_0^{-1} x + 1) = e^T D_0^{-1} x$$

$$e^T x' = e^T D_0^{-1} x (1 - e^T x')$$

$$\frac{e^T x'}{(1 - e^T x')} = e^T D_0^{-1} x$$

$$e^T \left[ \frac{x'}{1 - e^T x'} - D_0^{-1} x \right] = 0$$

and finally,

$$x = \frac{D_0 x'}{1 - e^T x'} = \frac{D_0 x'}{x'_{n+1}}$$

because the relation must hold for any x.

(d) The image of the point $x_o \in P$ is the center of the unit simplex,

$$a_o = \frac{D_o^{-1} x_o}{c^T D_o^{-1} x_o} = \frac{c}{c^T c} = \frac{1}{n+1} c.$$

This projective transformation is used once, to change the above inhomogeneous system of constraints in (1) into a homogeneous one as follows

$$(3)\quad Ax = b$$
$$AD_o x' = bx'_{n+1}$$
$$AD_o x' - bx'_{n+1} = 0$$

and, of course,

$$c^T x' + x'_{n+1} = 1$$
$$x' \geq 0,\ x'_{n+1} \geq 0$$

By defining the matrix $A' = [AD_o, -b]$, the transformed system can be written as

$$(4)\quad \Omega' = \{x' \mid A'x' = 0\}$$
$$S' = \{x' \mid c^T x' = 1,\ x' \geq 0\}$$

where, now, $A'$ is a $[m \times (n+1)]$ matrix and $x'$ is a $(n+1)$ vector.

Any linear programming set of constraints can be written in the form of system (4), which is the initial form required by Karmarkar's algorithm.

The second type of projective transformation is similar and is applied repeatedly to a linear programming system in the form of (4). With the assumption of an initial strictly interior point for system (1) as formulated above, a strictly interior point is readily available also for system (4) in the form of the center of the simplex

$$x'_o = a_o = \frac{1}{n+1} e.$$

In fact, $A'x'_o = \frac{AD_o e}{n+1} - \frac{b}{n+1} = \frac{1}{n+1}(Ax_o - b) = 0.$

The second projective transformation is now defined. Let D be a $(n+1) \times (n+1)$ diagonal matrix with a new expanded interior point, $x'^*$ on the diagonal $D = \text{diag}(x'^*)$. Then, the manifold (4) in the affine

space $\Omega'$ can be transformed into another manifold in the affine space $\Omega''$, and vice versa, by the following projective transformation

$$x = \frac{Dx''}{e^T Dx''}$$

$$(5)\quad \Omega'' = \{x'' \mid ADx'' = 0\}$$
$$S'' = \{x'' \mid e^T x'' = 1,\ x'' \geq 0\}.$$

The inverse projective transformation is

$$\frac{D^{-1}x'}{e^T D^{-1} x'} = x'$$

which takes $x'^*$ into the center of the simplex.

**The Primal and Dual Formulations of the Transformed LP Problems**

Consider the following dual pair of LP problems

| | **Primal** | **Dual** |
|---|---|---|
| (6) | min $CMIN = c^T x$ | max $CMAX = b^T y$ |
| | subject to $Ax = b$ | subject to $y$ untrestricted |
| | $x \geq 0$ | $A^T y \leq c$ |

where $A$ is an $(m \times n)$ matrix of coefficients with rank $(A) = m$. Any LP problem can be brought into this form by adding slack variables and eliminating unrestricted primal variables and redundant constraints.

Assuming we know a strictly interior point, $x_o > 0$, for the primal problem of (6) we can apply the first projective transformation to it in order to bring it into the initial specification required by Karmarkar's algorithm. Using the $D_o$ matrix defined above,

| | **Primal** | **Dual** |
|---|---|---|
| (7) | min $\{c^T D_o x' - CMIN x'_{n+1}\}$ | max $y_{m+1}$ |
| | subject to $AD_o x' - bx'_{n+1} = 0$ | subject to $y$ unrestricted |
| | $e^T x' + x'_{n+1} = 1$ | $y_{m+1}$ unrestricted |
| | $x' \geq 0$ | $D_o A^T y + e y_{m+1} \leq D_o c$ |
| | $x'_{n+1} \geq 0$ | $-b^T y + y_{m+1} \leq -CMIN$ |

where $y$ is a $(m \times 1)$ vector of dual variables while $y_{m+1}$ is a scalar. Notice that the optimal value of the primal objective function is zero and

that to implement Karmarkar's algorithm the value of CMIN must be known. A workable suggestion which circumvents this unusual requirement will be discussed later. The proof of convergence of Karmarkar's algorithm rests upon the use of logarithmic potential functions involving the ratio of the objective function in (7) to the primal variables. For empirical purposes, however, the linear objective function in (7) is a good approximation of the potential criterion and the one used in the numerical implementation of the algorithm.

Similarly, the optimal value of the dual objective function is equal to zero, corresponding to the vanishing of the $y_{m+1}$ dual variable. Let us define the augmented matrix $A' = [AD_o, -b]$ and the augmented vectors $c'^T = (c^T D_o, -CMIN)$ and $x'^T = (x'^T, x'_{n+1})$. Then, problem (7) can be rewritten more compactly as

| | **Primal** | **Dual** |
|---|---|---|
| (8) | min $c'^T x'$ | max $y_{m+1}$ |
| | subject to $A'x' = 0$ | subject to $y$ unrestricted |
| | $e^t x' = 1$ | $y_{m+1}$ unrestricted |
| | $x' \geq 0$ | $A'^T y + e\, y_{m+1} \leq c'$. |

This specification of the problem is the initial reference form of Karmarkar's algorithm. By simplifying the structure of the manifold $A'x' = 0$ to be a single plane, we can represent diagrammatically the primal of system (8) as in Figure 1.

The second type of projective transformation discussed in a previous section transforms the manifold $A'x' = 0$ and the objective function $c'^T x'$ in another affine space and objective function which constitute the reference working system for the algorithm. Return to the above system (8) is always possible (and, indeed, necessary) by means of the inverse projective transformation. Let us define, again, the $[(n+1) \times (n+1)]$ diagonal matrix $D = \text{diag}(x')$, where $x' > 0$. Then, system (8) can be transformed as follows:

| | **Primal** | **Dual** |
|---|---|---|
| (9) | min $c''^T Dx''$ | max $y_{m+1}$ |
| | subject to $A'Dx'' = 0$ | subject to $y$ unrestricted |
| | $e^T x'' = 1$ | $y_{m+1}$ unrestricted |
| | $x'' \geq 0$ | $DA'^T y + e\, y_{m+1} \leq Dc'$ |

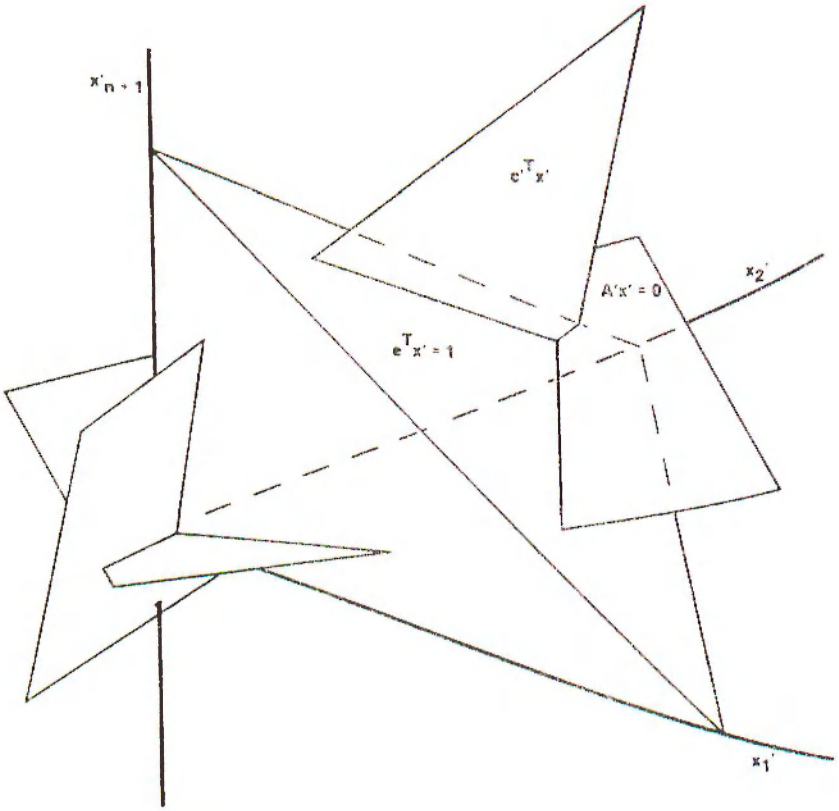


Fig. 1 The primal structure of problem (8)

For a more compact and convenient representation, let us define the partitioned matrix B as $B = \begin{bmatrix} A'D \\ c'^T \end{bmatrix}$. Then problem (9) can be rewritten as

| | **Primal** | **Dual** |
|---|---|---|
| (10) | min $c'^T D x''$ | max $u^T_{m+1}\, y = y_{m+1}$ |
| | subject to $Bx'' = u_{m+1}$ | subject to $y$ unrestricted |
| | $x'' \geq 0$ | $B^T y \geq Dc'$ |

where $u_{m+1}$ is a vector with unity in the $(m+1)$ location and zeros elsewhere, and $y$ is now redefined to be a $(m+1)$ vector of dual variables $y^T = (y^T, y_{m+1})$.

The idea that a "least-squares" approach to the solution of linear programming problems is incorporated into Karmarkar's algorithm can be easily seen from the dual of (10). Consider the constraints. The dual variables y constitute a vector of unrestricted components (as in the least-squares method). By introducing a vector of dual slacks $Dy_s$ the constraints can be rewritten as the usual linear system

(11) $B^Ty + Dy_s = Dc'$.

By minimizing the sum of squared dual slacks, the familiar least-squares formula for the y vector (the unrestricted dual variables) is obtained

(12) $y = (BB^T)^{-1}BDc'$.

Hence, at the optimum, the dual variables of linear programming problems can be computed using relation (12). During the process of reaching the optimum (say, at the kth iteration) the dual variables are not feasible and have the following structure

(13) $y(k) = (BB^T)^{-1}BDc' - (BB^T)^{-1}BDy_s$.

The $(BB^T)$ matrix can be stated somewhat more explicitly as

$$BB^T = \begin{bmatrix} A'D^2A'^T & : & A'De \\ e^TDA' & : & e^Te \end{bmatrix} = \begin{bmatrix} A'D^2A'^T & : & 0 \\ \cdots & \cdots & \cdots \\ 0 & : & e^Te \end{bmatrix}$$

Hence, the dual variables can be restated as

(14) $y(k) = (A'D^2A'^T)^{-1}A'D^2c' - (A'D^2A'^T)^{-1}A'D^2y_s$

$$y_{m+1}(k) = \frac{1}{n+1}e^TDc' - \frac{1}{n+1}e^TDy_s$$

As k becomes large, $y_{m+1} \to 0$ as does the vector $Dy_s$ of dual slacks.

The dual slacks play a fundamental role in Karmarkar's algorithm, as will be shown later on. From equation (11) it can easily be seen that they are analogous to least-squares residuals if the space generated by the vectors of $B^T$ is considered. Equivalently, they correspond to the orthogonal projection of the objective function vector, Dc', onto the null space of B.

## 3) AN OVERVIEW OF THE ALGORITHM

Any algorithm for solving mathematical programming problems require a three-step procedure to

1. determine the direction of movement,
2. determine the length of the step to be taken in that direction,
3. determine whether an optimum was achieved.

Karmarkar's algorithm is no exception. Karmarkar abides to the above requirements by first replacing the positive orthant by a sphere inscribed in the simplex S''and sharing the center with it. By staying within (or on the boundary of) this sphere, the algorithm generates a sequence of points, $\{x''\}$, whose coordinates are strictly positive and thus far away from their zero lower limit. These events occur in the transformed space of problem (10). When inversely projected back into the space of problem (8), some coordinates of the solution vector x can and do tend to zero. The idea of the sphere and its use is possibly the most original contribution of Karmarkar to the algorithmic methodology.

The sequence of feasible points $\{x''\}$ generated by the algorithm lies in the intersection of three subspaces: the space $\Omega'' = \{x'' \mid A'Dx'' = 0\}$, the simplex $S'' = \{x'' \mid e^T x'' = 1,\ x'' > 0\}$ and the sphere $B''(a_o, \alpha r)$, a ball of radius $\alpha r$, sharing its center $a_o$ with the simplex.

The diagrammatic representation of problem (9) is obviously similar to that of problem (8). The coordinates of x'', however, are different from those of x', because the diagonal matrix D is now present. The sphere $B''(a_o, \alpha r)$ inscribed in the simplex S'' appears as a circle in Figure 2. Four dimensions would be required to represent a three dimensional sphere associated with this problem. The intersection of the three spaces $B''(a_o, \alpha r) \cap \Omega'' \cap S''$ in Figure 2 is represented by the dotted segment through center the $a_o$.

At each iteration, the starting point is the center of both the simplex and of the sphere, $a_o$. The direction of optimization is determined by projecting the $Dc'$ vector orthogonally onto the null space of $B = \begin{bmatrix} A'D \\ e^T \end{bmatrix}$. The negative projection $c_p$, conveniently normalized, constitutes the direction of optimization. The next point x'' is determined by moving from $a_o$ in the direction of $-\hat{c} = -c_p / |cp|$ by a step

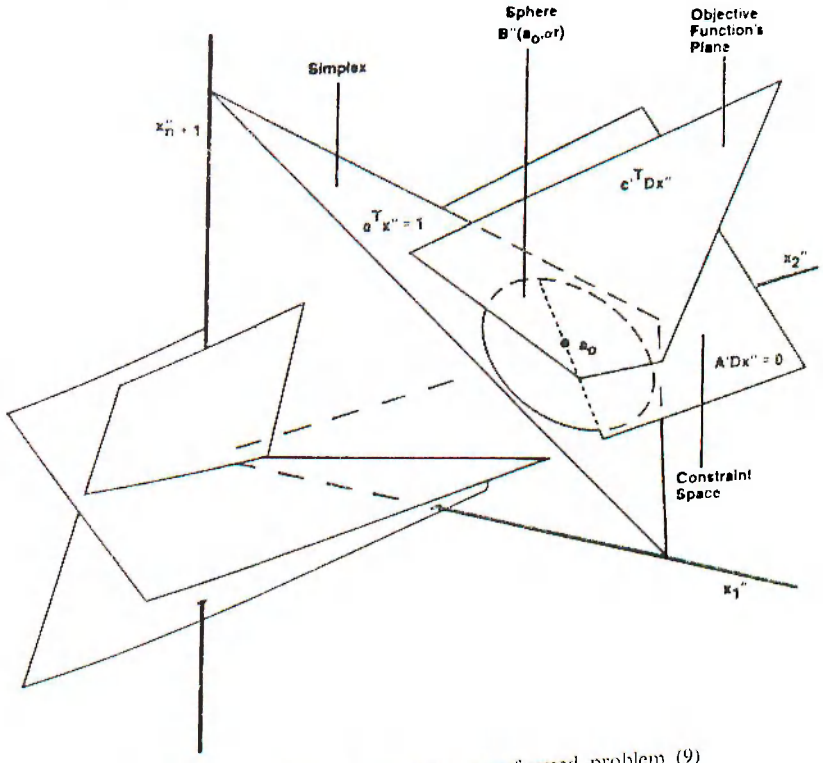


Fig. 2 The primal structure of the transformed problem (9)

of length αr, where r is the radius of the largest sphere inscribed in the simplex S'' and α is a number between 0 and 1. Karmarkar demonstrates in a very elegant and simple way that a point x'' so generated, minimizes the objective function $c''^{T}Dx''$ over the intersection of the manifold, $ADx'' = 0$, of the simplex S'' and of the sphere $B''(a_o, \alpha r)$ (Karmarkar, thm 5). Intuitively and considering Figure 2, the result of Karmarkar's algorithm can be likened to the Japanese art of paper folding, origami, as Karmarkar himself has suggested.

For any practical purposes, the convergence criterion is taken to be the following ratio

$$(15)\quad \frac{c''^{T}x''}{c''^{T}a_o} - \frac{c''^{T}Dx''}{c''^{T}Da_o} < 2^{-q}$$

where q is the precision of the computing machine used in the calculations. This criterion is an approximation of the theoretical one ba-

sed on logarithmic potential functions. Since the optimal objective function of both problems in (8) and (9) is zero, suitable tolerance limits can be established to determine its occurrence.

If the tolerance threshold has not been achieved, the "recursive" inverse transformation is applied to the newly found point x'' to obtain a new point x' in problem (8) such that it can be used to redefine the diagonal matrix D and restart the process over again.

Karmarkar's algorithm belongs to the class of steepest-descent methods. From Leunberger (p. 215): "The method of steepest descent is defined by the iterative algorithm

$$x_{k+1} = x_k - z_k g_k$$

where $z_k$ is a nonnegative scalar minimizing $f(x_k - zg_k)$. In words, from the point $x_k$ we search along the direction of the negative gradient $g_k = \nabla f(x_k)$ to a minimum point on this line; the minimum point is taken to be $x_{k+1}$."

This description is very similar to the process introduced above. There are important variants, however. First of all, in Karmarkar's algorithm, the point $x_k$ is is represented by its double projective transformation which takes it into the center of the simplex S'', $a_o$. Secondly, a transformation of the objective function gradient is used as the direction of optimization, namely the normalized orthogonal projection of the gradient onto the $\Omega''$ space. Thus, Karmarkar's recursive equation can be written as

(16) $x''(k+1) = a_o - z\hat{rc}(k)$

where $\hat{c} = c_p / |c_p|$, and $c_p = [I - B^T (BB^T)^{-1} B]Dc'$, is the orthogonal projection of $Dc'$ onto the manifold $A'Dx'' = 0$ and $e^T x'' = 1$. The center of the simplex and of the sphere, $a_o$, must be thought of as being obtained at each iteration as the result of the following projective transformations:

$x'(k) = Dx''(k) / e^T Dx''(k)$ and its inverse $a_o = D^{-1}x'(k) / e^T D^{-1}x'(k)$.

### The Algorithm

A. **Initializing setup.** Matrix A, vectors c and b, and a strictly interior point $x_o$ of problem (6) must be available. The procedure of finding an initial, strictly interior feasible point will be discussed in a subsequent section.

Still, at this stage of the discussion, the value of the objective function, CMIN, is assumed known. This assumption will be relaxed later.

A.1 Define the (nxn) diagonal matrix $D_o = \text{diag}(x_o)$, the augmented matrix $A' = [AD_o, -b]$, and augmented vector $c' = (c^T D_o, -CMIN)$. These computations bring the original LP problem (6) in the form of problem (7).

A.2 Define the center of the simplex $x^0 = a_o = e/(n+1)$.

B. **Iterative Loop**

B.1 Define the $[(n+1) \times (n+1)]$ diagonal matrix $D = \text{diag}(x^k)$.

B.2 Define the matrix $B = \left[\frac{A'D}{e^T}\right]$; that is, augment the matrix $A'D$ with a row of ones. This operation will set up the LP problem in the form of problem (10).

B.3 Compute the orthogonal projection of the vector $Dc'$ onto the null space of B as follows:

$$c_p = [I - B^T(BB^T)^{-1}B]\,Dc'.$$

This projection $c_p$ is identically equal to the slack variables of the dual problem in (10). It also contains the computation of the dual variables.

B.4 Normalize the projection $c_p$ to obtain $\hat{c} = c_p/(c_p'c_p)^{1/2}$. As the algorithm progresses, the coordinates of $c_p$ tend to vanish, but those of $\hat{c}$ will not.

B.5 Compute the next feasible point of problem (10) as $x''(k+1) = a_o - \alpha r\hat{c}(k)$, where $r = 1/\sqrt{n(n-1)}$ and $\alpha \in (0,1)$. Karmarkar suggests an $\alpha = 1/4$, but it seems plausible that also $\alpha$ should be chosen optimally as in the traditional descent algorithms.

B.6 Compute the inverse projective transformation $x^{k+1} = Dx''/e^TDx''$ useful for defining the next diagonal matrix D and restart the process.

B.7 Verify the convergence criterion with either $x''$ or $x'$ such as

$$\frac{c''^{T}Dx''}{c''^{T}Da_0} \qquad \frac{c''^{T}x'}{c''^{T}a_0} \geq q$$

where q is the precision of the available computer. If the convergence relation is satisfied, stop iterating and provide a printout of the results. If the convergence criterion is not satisfied, return to B.1.

C. **Report**

C.1 Compute the vector of original primal variables as $x = x' / [nx'_{n+1}]$.

C.2 Compute the dual variables as $y = (BB^T)^{-1} BDc'$.

The first m components of the y vector are the original dual variables. The last component, $y_{m+1}$, is the dual objective function of problem (10) and is equal to zero.

In spite of the above discussion, the doubt may linger in the reader's mind as to why the algorithm works so remarkably well. We may attempt a further explanation by pointing out again the crucial role played by the normalized orthogonal projection of the objective function. With reference to Figure 2, a step from $a_0$ in the direction of $-\hat{c}$ of length $\alpha r$ will modify the coordinates of $a_0$ into those of $x''$ by small amounts at each iteration. This is so because, for relatively large n, $r = 1 / \sqrt{n(n-1)}$ is a rather small number, further reduced by $\alpha < 1$. Accordingly, some of the $x''$ coordinates will increase slightly and some will decrease slightly (from $a_0$) in direct relationship to the costliness and / or profitability of the corresponding activities. These decreases and increases in $x''$ coordinates are transferred to the $x'$ vector via the inverse projective transformation. And now, the crucial point: while the small displacements from $a_0$, the center of the simplex, are not cumulative in $x''$, they are so when transferred to the $x'$ vector, which in turn, will redefine the diagonal matrix D for the next iteration. Hence, as the number of iterations, k, becomes large, it is possible to notice that some (indeed, (n–m)) coordinates of the $x'$ vector tend to zero. In this way, repeated orthogonal projections of $Dc'$ will reinforce the direction of optimization, the matrix D will tend toward a structure like $D = \begin{bmatrix} D_m & 0 \\ 0 & 0 \end{bmatrix}$ which corresponds to dropping

the (n − m) uneconomical activities of the A′ matrix. In the nondegenerate case, this process essentially reduces the nonbasis initial procedure to a basis one, as one would expect for the successful conclusion of the algorithm. Again, with reference to Figure 2, the algorithm stops when the orthogonal projection of the objective function vector Dc′ onto the constraint space has vanished. This is accomplished by repeated adjustments of both the original c′ and A′ by means of the diagonal matrix D. At the end, the transformed objective function plane will cut through the center of the simplex and of the inscribed sphere and will be parallel to the constraint manifold.

Finally, the representation of the dual of problem (9), as in Figure 3, exhibits the "least-squares" role of the dual variables as the coefficients of a linear combination of the vectors in $DA'^{T}$ corresponding to the orthogonal projection of the objective function gradient

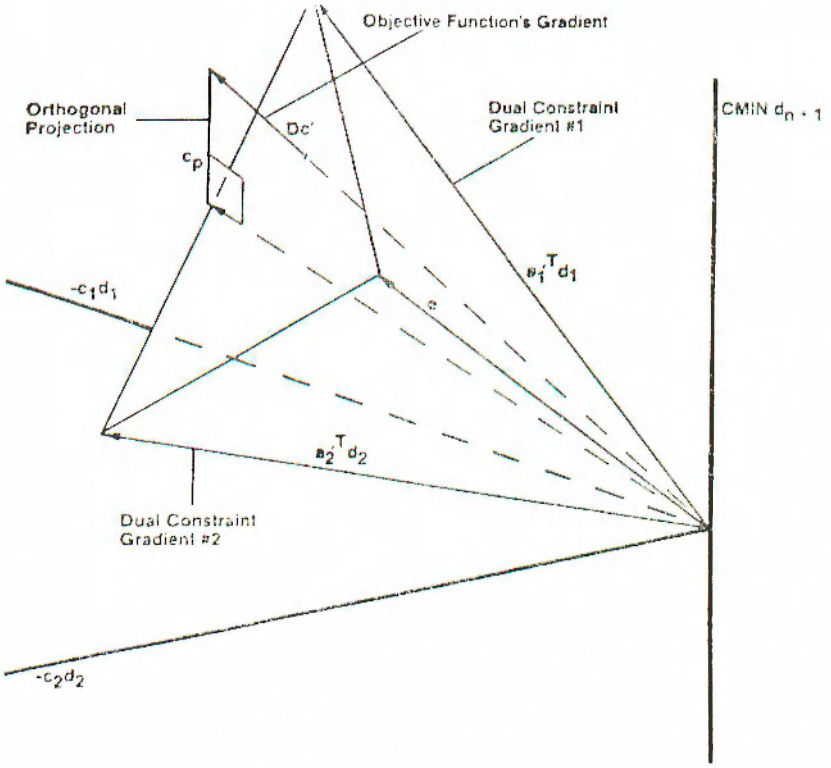


Fig. 3 The dual structure of problem (9)

$Dc'$ onto the space generated by the same vectors. At the optimum, the vector e vanishes from the picture because the associated dual variable $y_{m+1}$ is driven to zero. The orthogonal projection $c_p$ will also vanish and the vector $D_{c'}$ will lie in the plane generated by $a'_1 d_1$ and $a'_1 d_1$.

The flow-chart of Figure 4 stylizes the relationships among the three specifications required by Karmarkar's algorithm. The two projective transformations and their inverses allow, at any instance, for the representation of a given point in each of the three spaces. The iterative loop is fed by the diagonal matrix D which is regenerated at each iteration by the new vector x'. The D matrix redefines the objective function and the constraints at each iteration and enters into the determination of x'' via the orthonormal projection $\hat{c}$.

Intuitively, the combined use of repeated orthonormal projections and projective transformations employed by the algorithm may be interpreted as a process of folding the constraints' and the objective

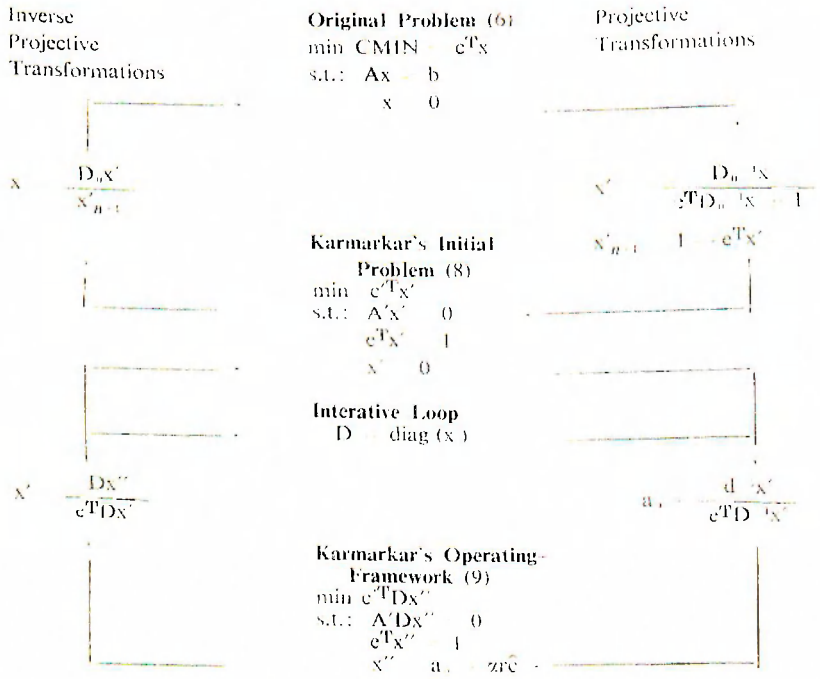


Fig. 4 A stylized flow-chart of the algorithm

functions' spaces until the optimum solution (in nondegenerate LP problems) is achieved when the shrinking intersection of those spaces becomes a single point.

### Removing The Requirement of Knowing CMIN

The procedure discussed so far requires the knowledge of CMIN, the optimal value of the objective function. Karmarkar presents a method for relaxing this assumption which he called the "sliding objective function" approach. It is a search for the optimal value CMIN which uses sliding upper and lower bounds. At first sight, it would seem that this method would lessen the efficiency of the algorithm as it requires repeated phases for resetting the upper and lower bounds.

An alternative and convenient approach which seems to work is based upon the realization that, from the dual of problem (7), the quantity $b^T y$ must be equal to CMIN at the optimum. Hence, since it is possible to obtain a least-squares estimate of the dual variables y at each iteration, they can be used to estimate CMIN, and replace the previous value of it in the vector c'. Several numerical examples were successfully solved using this approach. Depending upon whether the original primal problem is a max or a min problem, it may be convenient to experiment also with an estimate of CMIN as provided by $c'^T x'$ (k). With this approach, the initial value of CMIN is taken to be zero.

### Computational Efficiency and Numerical Stability

The most complex step in Karmarkar's algorithm consists in the computation of the orthogonal projection, $c_p$. These calculations require the inverse of the matrix $(BB^T)$ which is modified at each iteration. Efficient computer codes of the algorithm would exploit the possibility of updating the previous inverse matrix. Karmarkar has suggested one such approach.

Usually, from a numerical stability point of view, the introduction of the capacity constraint defining the unit simplex has damaging effects. Furthermore, in conventional algorithms, the rounding-off errors accumulate as the number of iterations increases. As a consequence, the condition number of the matrix to invert (that is, the ratio of the largest to the smallest characteristic roots) worsens throughout the computational process.

With respect to rounding-off errors, Karmarkar asserts that his algorithm is "self-correcting" or that "errors do not accumulate." This is due to the fact that, **at each iteration,** the next point x'' (k + 1) is computed as a deviation from the center $a_0$ rather than from the previous x(k) point, as in all other algorithms developed to date. Therefore, it would appear that, although the capacity constraint is to be feared in a conventional setting, in Karmarkar's approach it poses no special problems. The algorithm is remarkable also from a numerical stability viewpoint, in the sense that it "improves" the condition number of a matrix as the number of iterations increases.

The speed of convergence is governed by the required precision and Karmarkar's algorithm is capable of achieving any arbitrary accuracy compatible with the computer at hand. Of course, the progress toward the exact solution slows down as the number of iterations increases. It may be of interest, therefore, to exploit a characteristic of the algorithm which (after a suitable number of iterations) indicates those coordinates of the primal solution that will monotonically go to zero.

When (n-m) such coordinates have been detected, it is possible to break away from Karmarkar's algorithm to set up a square matrix of optimal activities and to invert it with one of the efficient procedures such as the LU or the Cholesky decomposition. This procedure could save many iterations in the final phase of the algorithm when it becomes more and more difficult (and tedious) to approximate with accuracy the true optimal values of the solution.

### The Initial, Strictly Interior Point

Karmarkar's algorithm is remarkable also with respect to the problem of determining an initial, strictly interior feasible point. Consider the problem of finding a strictly interior point of the following system

(17) $Ax = b,\ x > 0.$

Let us choose $x_0 > 0$ to be an arbitrary point, strictly interior to the positive orthant. This is, obviously, not a stringent requirement. Substituting for $x_0$ in (17) will produce, in general, a discrepancy, defined as

(18) $d = Ax_0 - b.$

Define an artificial variable $\lambda$ and the associated artificial problem as follows:

(19) min $\lambda$

subject to $Ax + \lambda d = b$

$x \geq 0, \lambda \geq 0.$

It is easy to verify that $x = x_0 > 0$ and $\lambda = 1$ constitute a feasible solution for problem (19). Given the arbitrary choice of $x_0$ within the positive orthant, the selection of $x_0 = e/n$, with e the sum vector, seems convenient. By applying Karmarkar's algorithm as discussed above (with $\lambda MIN = 0$) to problem (19) it is possible to find a strictly interior solution to problem (17), if it exists. In this case, the optimal value of the artificial variable, $\lambda$, is $\lambda = 0$. If it is not possible to drive $\lambda$ to zero after a given (theoretically determined) number of iterations, problem (17) does not possess a feasible solution.

This phase I algorithm completes the relaxations of the assumptions maintained throughout the discussion of the phase II of Karmarkar's method.

### Degeneracy and Multiple Optimal Solutions

Two related issues of great relevance must be regularly faced in mathematical programming. They are the notions of degeneracy and of multiple optimal solutions. Karmarkar (or, better, Karmarkar's paper) says nothing on either issue. Degeneracy, as it soon will be clear, does not induce the well known computational traps in Karmarkar's algorithm for the simple reason that it never surfaces. Since the algorithm generates a sequence of strictly interior points, the vector $x'$ has always positive (no matter how small) components. (Here, the precision of the computer comes directly into play.) From an empirical viewpoint, however, it is possible to recognize a degenerate solution when a desirable and suitable tolerance limit is chosen for detecting "practical" zero coordinates of the solution. The algorithm seems to combine the best of two worlds: of theoretical nondegeneracy, thus eliminating all the conceptual difficulties associated with it; and of degeneracy for practical purposes which allows for the possibility of detecting multiple optimal solutions.

The interesting question, however, is how to use Karmarkar's algorithm to compute the alternate optimal solutions, when they

exist. In studying this issue we gain another valuable insight into how radically different Karmarkar's algorithm is compared to the simplex method or any other algorithm. Since the algorithm proceeds from interior points, it is more likely that it will converge to a linear combination of the existing alternate optimal solutions rather than to an extreme point optimal solution. Hence, the optimal solution computed by Karmarkar's algorithm would have more positive coordinates than the number of constraints, contrary to any algorithm which utilizes a change of basis technique. From an empirical viewpoint, the solution returned by Karmarkar's algorithm, therefore, has the potential of being more interesting, because more diversified, than any single extreme point optimal solution returned by the simplex method. Of course, since there is an infinite number of optimal solutions which are linear combinations of basic solutions, care should be exercised to see that such a linear combination is itself optimal in some sense. With some further insights the algorithm may become also an efficient procedure for finding all extreme point optimal solutions.

## 4) TEMPORARY CONCLUSIONS

With the algorithm not yet in print, any offer of conclusions may appear audacious. We are in the presence, however, of a discovery that can be compared with the simplex method itself. Indeed, Karmarkar's algorithm seems to be at its best there where the simplex method must give up, as in the case of very, very large-scale problems. It is already clear that its discovery has ushered in a new era in mathematical programming, one that promises to be as exciting as that opened up by the glorious simplex method.

## Appendix A: The Simplex and the Spheres

The unit simplex S, the largest inscribed sphere, $B(a_o, r)$, and the smallest circumscribing sphere, $B(a_o, R)$, play crucial roles in Karmarkar's algorithm both at the theoretical and computational levels. The ratio of the two sphere's radiuses is a determinant of the convergence speed of the algorithm.

**The center of the unit simplex:** Consider the unit simplex $S \subset R^n$. Then, each coordinate of its center $a_o$ is equal to $1/n$. To verify this proposition we wish to compute the minimum distance between the center, $a_o$, and the origin, 0, of the unit simplex: that is, we want to find coordinates of $a_o^T \quad (t_1, \ldots, t_n)$ such that

(A1) min dist $(a_o, 0)$ subject to $e^T t = 1$.

This problem is equivalent to the following one:

(A2) $\min_t t^T t$ subject to $e^T t = 1$

where e is a (1xn) vector of ones. By Lagrangean techniques

(A3) $\min L = t^T t - 2\mu (1 - e^T t)$.

The relevant first order conditions are

(A4i) $\partial L / \partial t = 2t - 2\mu e = 0$

(A4ii) $\partial L / \partial \mu = 2(1 - e^T t) = 0$.

Substituting (A4i) into (A4ii), $1 - \mu e^T e = \mu n$, and $\mu = 1/n$. Therefore, the center of the unit simplex is $a_o = e/n$.

**The radius, r, of the largest inscribed sphere:** Each face (say the first one, $f_1$) of the unit simplex is tangent to the largest sphere with center in $a_o$ inscribed in it at the point $p_1$ with coordinates $p_1 = \left(0, \frac{1}{n-1}, \ldots, \frac{1}{n-1}\right)$, because, in order to be on the unit simplex, $p_1$ is subject to the condition $p_1^T e = 1$.

Hence, the radius, r, of the largest inscribed sphere is the distance between $a_o$ and $p_1$ or

(A5) $r = \text{SQRT}\,[(a_o - p_1)^T (a_o - p_1)]$

$$= \text{SQRT}\left\{\left[\left(\frac{1}{n}, \frac{e_{n-1}}{n}\right) - \left(0, \frac{e_{n-1}}{n-1}\right)\right]^T \left[\left(\frac{1}{n}, \frac{e_{n-1}}{n}\right) - \left(0, \frac{e_{n-1}}{n-1}\right)\right]\right\}$$

where SQRT stands for square root and $e_{n-1}$ is a $(n-1)$ vector of unit elements.

Thus,

$$r = \text{SQRT}\left[a_o^T a_o - 2\frac{e_{n-1}^T e_{n-1}}{n(n-1)} + \frac{e_{n-1}^T e_{n-1}}{(n-1)^2}\right]$$

$$= \text{SQRT}\left[\frac{1}{n} - \frac{2}{n} + \frac{1}{n-1}\right] = \text{SQRT}\left[\frac{1}{n(n-1)}\right]$$

**The radius, R, of the smallest circumscribing sphere:** Similarly, the radius, R, of the smallest circumscribing sphere is the distance from $a_o$ to any vertex of the unit simplex, say the first, $v_1$, with coordinates $v_1 = (1, 0, \ldots, 0) = (1, 0_{n-1})$. Hence,

$$(A6)\quad R = \text{dist}(a_o, v_1) = \text{SQRT}\left\{\left[\left(\frac{1}{n}, \frac{e_{n-1}}{n}\right) - (1, 0_{n-1})\right]^T \left[\left(\frac{1}{n}, \frac{e_{n-1}}{n}\right) - (1, 0_{n-1})\right]\right\}$$

where $0_{n-1}$ is a $[1x(n-1)]$ vector of zeros. Thus,

$$R = \text{SQRT}\left\{\left[\left(\frac{1}{n} - 1\right), \frac{e_{n-1}}{n}\right]^T \left[\left(\frac{1}{n} - 1\right), \frac{e_{n-1}}{n}\right]\right\}$$

$$= \text{SQRT}\left[\frac{(1-n)^2}{n^2} + \frac{n-1}{n^2}\right] = \text{SQRT}\left[\frac{n-1}{r}\right].$$

Finally, the ratio of the smallest circumscribing to the largest inscribed spheres is

$$\frac{R}{r} = \text{SQRT}\left(\frac{n-1}{n}\right) \text{SQRT}\,[n(n-1)] = (n-1).$$

**Appendix B: Properties of Vectors $x''$, $a_o$, and $\hat{c}$**

An explicit list of results implied by the Karmarkar's specification will undoubtedly facilitate the understanding of the algorithm. These results are the properties of the recursive equation $x'' = a_o - zrc$. They will be stated with little comment. All vectors are of dimension $(n-1)$

$$(B1)\quad Bx'' = Ba_o - zrB\hat{c}$$

$$= u_{m+1} - zr(0) = u_{m+1}$$

where $u_{m+1}$ is a unit vector with unity in the $(m+1)$ location and zeros everywhere else $Ba_o = u_{m+1}$ by feasibility, whereas

$B\hat{c} = B[I - B^T(BB^T)^{-1}B]Dc'/|c_p| = 0_{(n-1)x(n-1)}Dc'/|c_p|$ where $0_{(n-1)x(n-1)}$ is a null matrix.

$$(B2)\quad x''^T\hat{c} = a_o^T\hat{c} - zr\hat{c}^T\hat{c}$$

$$= zr.$$

$a_o^T c = 0$ because, from (B1), $B\tilde{c} = 0$. Hence, $\tilde{c}$ lies in the space generated by $B^T$, while $a_o$ lies in the space generated by B. $\tilde{c}^T\tilde{c} = 1$, because $\tilde{c}$ is a vector of unit length.

(B3) $x''^T x'' = x''^T a_o - \alpha r x''^T \tilde{c}$

$= x''^T e/n - 1 + \alpha r^2$

$= \frac{1}{n+1} + \frac{\alpha^2}{(n+1)n} - \left(1 - \frac{\alpha^2}{n}\right)/(n+1)$

In this section, $r = \frac{1}{(n-1)n}$ because the dimensions of the relevant space are $(n-1)$. If $\alpha$ is taken as $\alpha = 1$, $x''^T x'' = 1/n$.

## Appendix C: The Big M Method

The procedure used in the paper to bring a linear program in its conventional specification into the homogeneous structure required by Karmarkar's algorithm was based on a projective transformation.

There exists another (perhaps more intuitive but ultimately not convenient) procedure for this purpose. Consider the following LP problem

(C1) min CMIN $c^T x$

subject to $Ax = b$

$x \geq 0$

where A is a (mxn) matrix.

Select a sufficiently large number M, which bounds from above the sum of the primal variables, that is:

$c^T x \leq M$

or

(C2) $c^T x + x_{n+1} = M$

where $x_{n+1} \geq 0$ is considered the slack variable of this additional constraint.

By multiplying both the objective function and the constraints by M and carrying out the corresponding substitution we obtain

(C3) M CMIN $Mc^Tx$

$MAx \quad Mb$

or

$(c^Tx \quad x_{n+1})$ CMIN $Mc^Tx$

$MAx \quad b(c^Tx \quad x_{n+1})$

and finally,

$x_{n+1}$ CMIN $(Mc^T \quad c^T CMIN)x$

$(MA \quad bc^T)x \quad b \; x_{n+1}$.

Therefore, the LP specification of any LP problem required by Karmarkar's algorithm using the big M method is

(C4) min $(Mc^T \quad c^T CMIN)x' \quad CMINx'_{n+1}$

subject to $(MA \quad bc^T)x' \quad bx'_{n+1} \quad 0$

$c^Tx' \quad x'_{n+1} \quad !$

$x' \quad 0 \; x'_{n+1} \quad 0$

where $x' \quad x / M$, $x'_{n+1} \quad x_{n+1} / M$.

This method does not seem to have any advantage over the projection method. First of all, the selection of a sensible upper bound M on the sum of the primal variables may be difficult and its knowledge does not avoid the requirement of knowing also the optimal value of the objective function. Secondly, to apply Karmarkar's algorithm it is necessary to know a strictly interior point. Hence, there is no advantage in postponing the computation of it. The projection method, which only requires knowledge of an interior point seems preferable.

## REFERENCES

BORGWARDT, K.H. The Average Number of Pivot Steps Required by the Simplex Method is Polynomial, **Zeitschrift für Operations Research**, 26 (1982): 157-177.

FRISH, R. The Multiplex Method for Linear Programming, **Sankhya**, 18 (1957): 329-362.

KARMARKAR, N. **A New Polynomial-Time Algorithm for Linear Programming**, Undated, mimeograph paper (presumably, Summer 1984), AT&T Bell Laboratories, Murray Hill, NJ.

LUENBERGER, D.G. **Linear and Nonlinear Programming**, Second Edition, Addison-Wesley, Reading, MA, 1984.

## ÖZET

### KARMARKAR'IN DOĞRUSAL PROGRAMLAMA ALGORİTMASI ÜZERİNE

Doğrusal Programlama problemlerini çözmek üzere, N. Karmarkar tarafından geçtigimiz günlerde yeni bir yöntem önerildi. Simplex yönteminde olduğu gibi çözüm setinin sınırlarındaki, uç noktaların birişinden diğerine atlamak yerine, bu algoritma, erişilebilir çözüm setini keserek, ve çözüm setinin içinden çalışarak çözüm aramaktadır. Bu yöntem, özellikle çok büyük doğrusal programlama problemlerinin çözümü için ümit vermektedir. Bu çalışmada, Karmarkar'ın önerdiği çözüm yöntemi tartışılmaktadır. Aynı zamanda, Karmarkar'ın çalışmasında açık olarak tartışılmayan duality'nin de önerilen yöntem için geçerli olduğu gösterilmektedir. Son olarak, çoklu çözüm (multiple optimum) ve dejenere çözüm (degeneracy) sorunları söz konusu yöntem çerçevesinde irdelenmektedir.