SIMPLEX TABLEAU BASED APPROXIMATE PROJECTION IN KARMARKAR'S ALGORITHM

A THESIS
SUBMITTED TO THE DEPARTMENT OF INDUSTRIAL ENGINEERING AND THE INSTITUTE OF ENGINEERING AND SCIENCES OF BİLKENT UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

By
Yavuz GÜNALAY
September, 1990

Yavuz Günalay

SIMPLEX TABLEAU BASED APPROXIMATE PROJECTION IN KARMARKAR'S ALGORITHM

A THESIS

SUBMITTED TO THE DEPARTMENT OF INDUSTRIAL ENGINEERING

AND THE INSTITUTE OF ENGINEERING AND SCIENCES

OF BİLKENT UNIVERSITY

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF SCIENCE

By

Yavuz Günalay

September, 1990

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Mustafa Akgül (Principal Advisor)

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Prof. Halim Doğrusöz

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Osman Oğuz

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Béla Vizvári

Approved for the Institute of Engineering and Sciences:

Prof. Mehmet Baray
Director of Institute of Engineering and Sciences

ABSTRACT

SIMPLEX TABLEAU BASED APPROXIMATE PROJECTION IN KARMARKAR'S ALGORITHM

Yavuz Günalay
M.S. in Industrial Engineering
Supervisor: Assoc. Prof. Mustafa Akgül
September, 1990

In this thesis, our main concern is to develop a new implementation of Karmarkar's LP Algorithm and compare it with the standard version. In the implementation, the "Simplex Tableau" information is used in the basic step of the algorithm, the projection. Instead of constructing the whole projection matrix, some of the orthogonal feasible directions are obtained by using the Simplex Tableau and to give an idea of its effectiveness, this approximation scheme is compared with the standard implementation of Karmarkar's Algorithm, by D. Gay. The Simplex Tableau is also used to calculate a basic feasible solution at any iteration with a very modest cost.

Keywords: Karmarkar's LP Algorithm, Simplex Tableau.

ÖZET

KARMARKAR'IN ALGORİTHMASINDA SİMPLEX TABLOYA BAĞLI YAKLAŞIK İZ DÜŞÜM UYGULAMASI

Yavuz Günalay
Endüstri Mühendisliği Bölümü Yüksek Lisans
Tez Yöneticisi: Doç. Mustafa Akgül
Eylül, 1990

Bu çalışmada, Karmarkar'ın Doğrusal Programlama Algoritmasının yeni bir uygulaması geliştirilmiş ve bu uygulama standart algoritma ile karşılaştırılmıştır. Uygulamadaki yenilik "Simplex Tablo" bilgisinden yararlanılmasıdır. Her iterasyonda projeksiyon matriksinin hesaplanması yerine "feasible" yönler Simplex Tablodan yararlanılarak bulunmuş ve bu yönlerin bir bölümü kullanılarak değer vektörünün yaklaşık iz düşümü hesaplanmıştır. Ayrıca, herhangi bir iterasyonda Simplex Tablo kullanılarak bir köşe noktasının ziyaret edilmesi çok az bir extra çaba gerektirmektedir.

Anahtar kelimeler: Karmarkar'ın Algoritması, Simplex Tablo.

To my parents,

ACKNOWLEDGEMENT

I would like to thank to Assoc. Prof. Mustafa Akgül for his supervision, guidance, suggestions, and patience throughout the development of this thesis. I am grateful to Prof. Halim Doğrusöz, Assoc. Prof. Osman Oğuz and Asst. Prof. Béla Vizvári for their valuable comments.

My sincere thanks are due to all of my close friends, both from Bilkent and outside, even from US, for their valuable remarks, comments and encouragement.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Linear Programming is an optimization problem of a linear cost function over a convex polyhedra defined by a finite number of linear inequalities. The well known algorithm to solve the LP problems is the Simplex Method. Although Simplex Method is very efficient for the most real world problems, the worst case behavior is exponential, which makes it theoretically unsatisfactory. The recent polynomial time LP algorithm is demonstrated by N. Karmarkar [16] in 1984.

In this thesis, "Simplex Tableau Based Approximate Projection (STBAP) Algorithm", an implementation of Karmarkar's Algorithm, will be presented. Karmarkar's algorithm is an interior point algorithm, and its most costly step is the calculation of a feasible decent direction in every iteration. In the STBAP Algorithm this direction vector is constructed approximately by projecting the cost vector over *some* of the orthogonal feasible directions. In most of the variants of the Karmarkar Algorithm approximate projection is utilized and the calculated vector may be an infeasible direction, because of the used approximate projection technique. But in the STBAP Algorithm the constructed direction is always feasible because it is the convex linear combination of *some* of the orthogonal feasible direction vectors. In order to calculate the orthogonal feasible directions, a Simplex Tableau of the problem is formed. But, in contrast to the Simplex Method it is not necessary to change it in every iteration. The tableau is changed in some iterations to keep the basis matrix well-conditioned. And these basis changes may cause an early termination of the algorithm with the optimum basis, instead of terminating due to the original stopping criteria which is very loose in most problems.

This thesis consists of 6 chapters. The next chapter contains the notation and literature review. In chapter 3, the Karmarkar Algorithm is discussed briefly. The suggested STBAP Algorithm is stated in chapter 4, and it is compared with the standard implementation of Karmarkar's algorithm by D. Gay [10] in chapter 5. Last chapter is reserved for conclusion and further research suggestions.

2. NOTATION and LITERATURE REVIEW

In this thesis the following notation will be used.

•) Unless otherwise specified, the upper case letters (e.g. A) define matrices, and lower case letters (e.g. x) define one dimensional vectors of appropriate sizes.

•) $N(A)$ represents the null space of matrix A.

•) Superscript "T" is used to represent the transpose.

Since 1947, the most commonly used LP algorithm was the Simplex Method, which had been introduced by G.B. Dantzig [7]. Let **P** be a Linear Programming problem in canonical form;

$$\begin{aligned} \mathbf{P)} \qquad & \min\ c^T x \\ & \text{s.t. } A\,x \leq b \end{aligned}$$

where A is a $m \times n$ matrix, $b \in R^m$, and $c, x \in R^n$. The Simplex algorithm starts at a vertex of $\mathcal{P} = \{x \in R^n : \quad Ax \leq b\}$, the feasible region of the problem **P** and pivots to a neighbor vertex of $\mathcal{P}$ until the optimal solution is reached or it is proven that the problem is unbounded. Any vertex of $\mathcal{P}$ can be defined by the intersection of n linearly independent hyperplanes. The number of vertices of $\mathcal{P}$, i.e. the basic feasible solutions, is bounded above with number of such intersection points, which is equal to $\begin{pmatrix} m \\ n \end{pmatrix}$. Since the maximum number of vertices is finite and pivoting requires $O(n^2)$ elementary arithmetic operations, Simplex Method is a finite algorithm if visiting a vertex twice (i.e. cycling) is avoided. But, it is not a polynomial time algorithm, which is also shown by Klee and Minty [19], Edmonds [8], Jeroslow [15] and many others, by generating special type of LP problems. Many scientists have worked on the variants of the Simplex algorithm to overcome the deficiency of this exponential running time complexity of the algorithm by applying different pivoting rules [4, 15, 12], using duality theory; Dual Simplex Method [23] or Primal-Dual Simplex Method [9]. Up to now, none of those studies have been succesfull.

In contrast to this exponential worst case behavior, the Simplex Method works very fast for most real world problems. A probabilistic analysis on Simplex Method was presented by K.H. Borgwardt [6] in his book. He proved that the expected number of the pivots is polynomial and this makes the Simplex Algorithm efficient in most real world problems. But, the idea of finding a polynomial algorithm to LP always attracted researchers. In 1979 L.G. Khachian [18], a Russian mathematician, announced that the system of Linear Inequalities can be solved in polynomial time. His algorithm, which is named as "Ellipsoid Algorithm", was designed to solve the feasibility problems, but it was easily modified to solve LP problems in polynomial time. The algorithm depends on the idea of shrinking the ellipsoids. The problem is to search a feasible vector, $x \in \varphi = \{x \in R^n : Ax \leq b \,\&\, -2^L \leq x_i \leq 2^L \;\; i = 1,2,..,n\}$, where $b \in R^m$ and L is the bit size required to input the data. Initially, a large ellipsoid E^0 enough to contain the polyhedra φ is built. In each iteration, E^k is shrinked such that the new ellipsoid, E^{k+1} still contains φ and the volume of the ellipsoid is reduced at least a constant amount which does not depend on the problem data. If the center of E^k is in φ, then the algorithm terminates with the solution vector. Otherwise, the iteration are continued until the volume of the ellipsoid becomes less than a certain value with which one can claim that φ is empty.

In 1984, a new polynomial time LP Algorithm was presented by N. Karmakar [16] at the Symposium on Theory of Computing, in Washington D.C. The algorithm was called the Projective Algorithm. He worked on the problem **KP**;

$$\begin{aligned} \textbf{KP)} \qquad & \min \; c^T x \\ & \text{s.t.} \; Ax = 0 \\ & \qquad e^T x = 1 \\ & \qquad x \geq 0 \end{aligned}$$

where A is a $m \times n$ matrix, $b \in R^m$, $c, x \in R^n$, and $e \in R^n$ is the vector of ones.

With the assumptions:

i) $c^T x^* = 0$, where x^* is the optimum solution vector of KP.

ii) KP has a nonempty and bounded feasible region, $\mathcal{KP}$.

iii) A is full row rank.

iv) The center of the unit simplex S, $\frac{1}{n}e$, is feasible.

The algorithm starts at an interior point, $x^0 \in \mathcal{KP}$, and generates a sequence of new interior points, x^1, x^2, ...x^k. It stops when a stopping criterion is satisfied, $c^T x^k \leq 2^{-L}$,

where L is the length of input data. In each iteration, the algorithm utilizes a projective transformation, $T(.)$, from simplex S onto itself, such that the current interior point, x^i, is mapped to $\frac{1}{n}e$, the center of the simplex, S. Simplex is the polytope defined by $n+1$ *affinely independent* vectors in R^n and the unit simplex, ($S = \{x \in R^n : e^T x = 1\,,\ x \geq 0\}$) One can construct two spheres centered at $\frac{1}{n}e$ with radii r and R, such that the small one is contained in the simplex S and the large one contains S ($r = \frac{1}{\sqrt{n(n-1)}}$ & $R = \sqrt{\frac{n-1}{n}}$) see fig. 2.1.

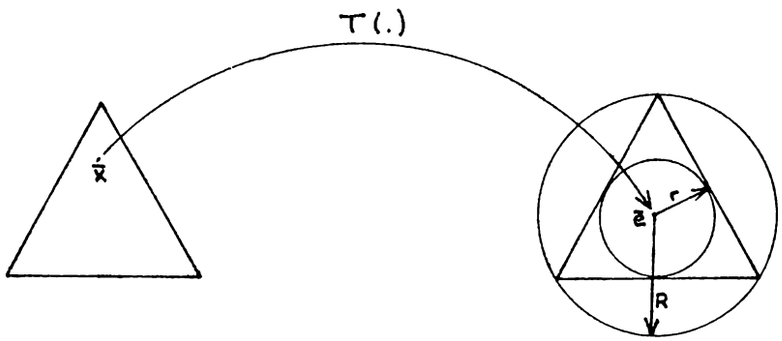

Figure 2.1: The projective transformation from the unit simplex S in R^n onto itself.

If the current point is the center of S, then feasibility is always preserved by moving at most r units along a feasible unit direction p, i.e. p satisfies $Ap = 0$ $e^T p = 0$ $\| p \| = 1$. Then the algorithm is given as follows:

0) Initialization. $k = 0$, $x^0 = \frac{1}{n}e$.

1) Optimality check. If $c^T x^k \leq 2^{-L}$ STOP, otherwise continue.

2) Basic iteration.

   i) $B = \begin{bmatrix} AD \\ e^T \end{bmatrix}$

   ii) $P_{N(B)} = I - B^T(B^T B)^{-1}B$

   iii) $p = P_{N(B)}Dc$

   iv) $y = \frac{1}{n}e + \hat{\alpha}\frac{p}{\|p\|}$, where $\hat{\alpha} < r$.

   v) $x^{k+1} = T^{-1}(y, x^k)$, where $T(x, x^k) = \frac{D^{-1}x}{e^T D^{-1}x}$ and D is an $n \times n$ diagonal matrix with diagonal entries are; $D_{ii} = x_i^k$.

   vi) $k = k + 1$ and return to step (1).

Since our original cost function, $c^T x$ is not invariant under projective transformations, a somehow different cost function, named as "Potential Function", $f(c,x) = \sum_{j=1}^{n} \log(\frac{c^T x}{x_j})$, which is invariant under such transformations is used. It is shown that, in every iteration at least a constant amount reduction, δ which is independent from the problem data, in potential function is guaranteed. Using the stopping criterion which will be discussed in the next chapter, it is concluded that the number of iterations is $O(nL)$. In each iteration, $O(n^3)$ arithmetic operations is required to perform the matrix of projection onto the $N(A)$. Those make the total time complexity of Karmarkar's Algorithm $o(n^4 L)$. In contrast to both, the Ellipsoid Algorithm and the Simplex Method, the round-off errors in each iteration is not accumulated in the Karmarkar algorithm, and this is a very good feature for the computer implementations.

After the presentation of Karmarkar, a large number of studies on the subject of the projective algorithm were published. Those studies can be classified in two groups :
i) those taking the advantage of the sparseness of the large scale LP problems [1, 24],
ii) those using approximate projection [3, 14, 21].

A well known variant of Karmarkar's algorithm was reported by M.J. Todd & B.P. Burrell [26], in 1985. They combined the duality theory with the interior point approach and used the dual variables to get an estimate for the optimum value, as a lower bound. This was a relaxation on the first assumption, and as a result of it, the variant could solve the problems with unknown objective values. Another practical improvement that they introduced was to generate the basic point solutions (*not* necessarily feasible) at any desirable iteration with a very modest extra cost.

A variant of the Karmarkar's Algorithm for problems in standard LP form was introduced by D.M. Gay [10], in 1987. The standard form LP is defined as;

$$\begin{aligned} \textbf{SP)} \qquad & \min\ c^T x \\ & \text{s.t. } A\,x = b \\ & \qquad x \geq 0 \end{aligned}$$

where A, b, c, and x have the same sizes as in P. The variant requires no apriori knowledge of the optimal objective function value. It uses a similar approach with Todd and Burrell [26] to compute the more strict lower bounds for the optimal value. Besides the elimination of the first assumption, the variant works exactly as the original algorithm. It utilizes the exact projection, and it stops when the "gap" between the current objective value and the lower bound is reduced to $\varepsilon = 2^{-L}$.

In 1988, D. Goldfarb & S. Mehrotra [13] reported a relaxed variant of Karmarkar's Algorithm. They used the similar LP family $\mathbf{LP(z)}$ with Anstriecher [3], for the transferred problem formulations.

$$\mathbf{LP(z)} \qquad \begin{aligned} \min\ & (c - zd)^T x \\ \text{s.t.}\ & A\,x = 0 \\ & e^T x = 1 \\ & x \geq 0 \end{aligned}$$

where $d = x^k$ the current iterate solution. The relaxed problem $\mathbf{LP(z)}$ is solved using approximate projection. The new iterate point is $x^{k+1} = T^{-1}(y, d)$ where $y = e + \alpha r \frac{p}{\|p\|}$, $0 < \alpha < 1$ and $p \in R^n$ is a decent direction. They called (p, z) as an admissible pair if the following conditions are satisfied; i) $A'p = 0$, $e^T p = 0$ and $p \neq 0$,
ii) $(c' - zd)(e + R\frac{p}{\|p\|}) \leq 0$, where $A' = AD$, $c' = Dc$ and D is a $n \times n$ diagonal matrix with $D(i,i) = x_i^k \quad i = 1, 2, ..., n$. Using a Congugate Gradient (CG) technique $p = Z'w(c")$, the decent vector, is generated approximately from the least square problem $\|Z'w(c") - c"\|$ where $c" = (Dc - zd)$, $w(c") \in R^n$ and Z' is the basis matrix of $N(A')$. Basic linear algebra knowledge gives us a simple representation of the basis of $N(A)$ as $Z = \begin{bmatrix} -B^{-1}N \\ I \end{bmatrix}$, where B and N are the partitions of the matrix A. Therefore, $Z' = \begin{bmatrix} -D_B^{-1}B^{-1}ND_N \\ I \end{bmatrix}$ where D_B and D_N are the partitions of the diagonal matrix D as $D = \begin{bmatrix} D_B & 0 \\ 0 & D_N \end{bmatrix}$. If the generated (p, z) pair is not admissible, then either a more accurate projection vector p' is required, or a more strict lower bound z', where $z < z' \leq z^*$ is needed. If a CG technique is used, the more accurate projection vector p' can be easily obtained by continuing on iterations on least square problem from where it was temporarily stopped, instead of recalculating it from scratch. Also, the updated lower bound $z' = z + \Delta$, $\Delta > 0$ can be calculated using the information of approximate CGLS solution.

Performance of the Karmarkar's algorithm and its variants are highly dependent on the stopping criteria. One way of improving the stopping criteria is to identify the optimal basis as early as possible. Recent studies are mostly related with this early identification schemes. Gay [11], Kovacevic [22], Tapia [25] and Ye [28] use very similar techniques to form the optimal basis in their studies. The simplest one is due to Tapia and Zhang [25]. They define an indicator vector q, which is the diagonal entries of the matrix of projection to the row space of the constraint matrix, $R(A)$. And it is stated that this indicator vector convergence to a $0 - 1$ vector, which indicates whether the variable is nonbasic or basic at the optimum solution quadraticly faster than the convergence of the interior point solution to the optimum solution (i.e. $\|q^k - q^*\| \leq O(\|x^k - x^*\|^2)$). These stopping rule implementations will be discussed in the Appendix B briefly. Also, in the late seventies M.C. Cheng presented an early optimum basis identification scheme for the Simplex Method [29].

In 1986, M. Kojima [20] reported that it is possible to determine the basis of the optimal solution before the algorithm satisfies the stopping criterion. This reduced the number of iterations in some problems, but he could not manage to put a limit to the number of iterations. In the same year, Kojima & Tone published a modified version of the Karmarkar's Algorithm [21]. Besides the optimum basic variable test, they presented a test for nonbasic variables at the optimum solution, as well. They used approximate projection in this variant and obtained some good results.

3. KARMARKAR'S ALGORITHM

The algorithm was announced at the Symposium on Theory of Computing by N. Karmarkar, in April 1984. In the worst case the algorithm requires $o\,(n^4L)$ arithmetic operations, where n is the dimension of the problem and L is the input data length. The algorithm works on the special type of LP problems over rational data, call it **KP** (defined in Chap. II). After the projective transformation $T(x,d) = \frac{D^{-1}x}{e^T D^{-1}x}$ in the image space an equivalent problem $\tilde{\mathbf{K}}\tilde{\mathbf{P}}$ is defined as;

$$\begin{array}{lll} \bar{\mathbf{K}}\bar{\mathbf{P}}) & \min & \bar{c}^T y \\ & \text{s.t.} & \bar{A}y = 0 \\ & & e^T y = 1 \\ & & y \geq 0 \end{array}$$

where $\bar{c} = Dc$, $\bar{A} = AD$, and D is diagonal matrix with $D_{ii} = d_i$ and $d \in int(\mathcal{KP})$, i.e. an interior point of the problem KP. The potential function which is invariant under projective transformation is defined

$$\Phi(c,x) = n \log c^T x - \sum_{j=1}^{n} \log(x_j)$$

and in the image space $\Phi(\bar{c},y) = \Phi(c,x) + \log[Det(D)]$, where $y = T(x,d)$. This invariance will lead the number of iterations in the algorithm be polynomial, if at least a constant amount reduction in the potential function is guaranteed in the image space.

Theorem 1: For $\alpha \in (0,1)$, there exist a $\delta(\alpha) > 0$ such that,

$$\Phi(\bar{c},y) \; - \; \Phi(\bar{c},\bar{e}) \; \leq \; -\delta(\alpha)\,,$$

where $\bar{e} = \frac{1}{n}e$ and $y = \bar{e} - \alpha r \frac{P\bar{c}}{\|P\bar{c}\|}$. □.

The proof of theorem 1 is given at the Appendix A in details. Using this main result, the following theorem concludes that the algorithm has a polynomial time complexity.

Theorem 2: Let n be the dimension of the problem **KP** and L be the input data length. Under the given assumptions in Chap. II, Karmarkar's Algorithm will obtain an interior vector $\bar{x}$ satisfying

$$c^T \bar{x} \leq 2^{-L} \tag{1}$$

in at most $O(nL)$ iterations.

Proof: Let us start with the interior vector, $x^0 = \bar{e}$. After k iterations we get another interior point x^k, and a $k\,\delta(\alpha)$ reduction in potential function. So,

$$\Phi(c, x^k) - \Phi(c, \bar{e}) \leq -k\delta(\alpha) \tag{2}$$

Using the definition of potential function

$$n \log \frac{c^T x^k}{c^T \bar{e}} \leq \sum_{j=1}^{n} \log x_j^k + n \log n - k\delta(\alpha) \tag{3}$$

Or,

$$c^T x^k \leq K_0 e^{-(\frac{\delta(\alpha)}{n})k} \tag{4}$$

where $K_0 = nc^T \bar{e} = \sum_{j=1}^{n} c_j$ and for $k = o(nL)$ Equation 4 becomes,

$$c^T x^k \leq e^{-L} \leq 2^{-L}$$

□.

Moreover the stopping criterion guarantees us that any basic feasible x vector satisfies $c^T x < c^T \bar{x}$ will be an optimal solution to the problem **KP**. It is a very well known fact that calculation of the projection matrix P in each iteration is of $O(n^3)$. This concludes that the Karmarkar's Algorithm is a polynomial algorithm, and its worst case bound is of the $O(n^4 L)$. But in most variants this is reduced to $O(n^{3.5} L)$ using approximation techniques in the step (2 ii). Depending on the implementation the algorithm could run even 50 times faster than the Simplex Method for large scale problem instances [1, 17, 24].

4. SIMPLEX TABLEAU BASED APPROXIMATE PROJECTION ALGORITHM

In this chapter, a new variant of Karmarkar's Algorithm, Simplex Tableau Based Approximate Projection (STBAP) Algorithm will be discussed. As its name implies, this variant uses approximation in calculating the projection matrix like many other variants of Karmarkar Algorithm. In STBAP Algorithm, the Simplex Tableau (ST) information is embedded into the approximation rule. Also, an early termination with an optimum ST is another advantageous feature of the algorithm.

Our algorithm works on the canonical form LP problems

$$\begin{aligned} \mathbf{LP} \qquad \min \quad & c^T x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0 \end{aligned}$$

with the following assumptions:

i) LP has an optimum value of zero.

ii) Constraint matrix, A is of full row rank.

iii) The center of the unit simplex in R^n is feasible.

In order to convert the problem **LP** into **KP** and be sure that it satisfies all the assumptions, the A and c matrices change, and the new problem $\tilde{\mathbf{LP}}$ is:

$$\begin{aligned} \tilde{\mathbf{LP}} \qquad \min \quad & \tilde{c}^T \tilde{x} \\ \text{s.t.} \quad & \tilde{A}\tilde{x} = 0 \\ & \tilde{x} \geq 0 \end{aligned}$$

where $\tilde{c}^T = [c^T, -z^*]$, z^* is the optimum objective value of **LP**, $\tilde{A} = [A, -b]$ and $\tilde{x} \in R^{n+1}$ with $\tilde{x}_{1:n} = x$ and $\tilde{x}_{n+1} = 1$ (constant). Since $\tilde{x}_{n+1}$ has to be fixed to one, the inverse transformation is defined as; $T^{-1}(y, d) = \frac{1}{y_{n+1}} Dy$, where d and D are defined in the previous

chapter. In $\tilde{\mathbf{LP}}$, there is no constraint as $e^T x = 1$, because after the transformation $T(.)$, the equivalent problem $\bar{\mathbf{KP}}$ in Y-space (image space of the transformation $T(.)$) always satisfies the constraint, $e^T y = e^T \frac{D^{-1}x}{e^T D^{-1}x} = 1$, and the optimum value of the problem $\tilde{\mathbf{LP}}$ is zero ($\tilde{c}^T \tilde{x}^* = c^T x^* - z^* = 0$). After the formation of the problem $\tilde{\mathbf{LP}}$, the ST of the problem is constructed by using a partition of constraint matrix, $\tilde{A} = [B, N]$ where B is a nonsingular square submatrix of $\tilde{A}$.

4.1 Simplex Tableau

Since **KP** has a special structure, ST looks different than the ones in the Simplex Method. There is no need to carry the "rhs" vector and to calculate the objective value, since "rhs" is a vector of zeros and the basis is not necessarily feasible. Thus, the "rhs" column (i.e. the last column) of the original tableau can be eliminated. Therefore, the built ST occupies a place of $(m+1) \times (n+1)$ instead of $(m+1) \times (n+2)$ in the memory. Due to the construction of **KP** from **LP**, the first element of the last column of the ST gives the gap between the optimum value and current basic solution's objective value, and the remaining elements in the last column represent the current basic solution in opposite sign.

ST)

$c_N - c_B B^{-1} N$	$-z^* + c_B B^{-1} b$
$B^{-1} N$	$-B^{-1} b$

where c_B and c_N are the appropriate partition of the cost vector, c with respect to the partition of A.
Results:

1. The current basis is feasible if and only if the last column of the ST is non-positive.
2. The current basis is optimal if and only if both:
   i) the last column of the ST is non-positive,
   ii) the first row of the ST is non-negative.

   which means, the last element of the first row is equal to zero. This makes sense, because if the basis is optimal then $x_B^* = B^{-1} b$, $c_B x_B^* = c_B B^{-1} b = z^*$.

In order to transform ST into the image space, Y-space, replace B by BD_B , N by ND_N , c_B by $c_B D_B$ and c_N by $c_N D_N$. Since $x_{n+1} = 1$, there is no need to modify b. Then

STy looks somehow different but it gives exactly the same information, because both D_B and D_N are positive diagonal matrices.

STy)

$(c_N - c_B B^{-1} N) D_N$	$-z^* + c_B B^{-1} b$
$D_B^{-1} B^{-1} N D_N$	$-D_B^{-1} B^{-1} b$

Again the same results are valid for the STy. STy has the same space requirement and time complexity in build up as the original ST used in the Simplex Method; space(STy) is $O(mn)$ and time(STy) is $O(m^2 n)$. STy has to be updated in each iteration due to the changes in matrices D_B and D_N, and this updating is $O(mn)$ simple multiplications. Besides these regular updates, in some iterations the basis is changed to ensure the basis matrix, BD_B be well-conditioned.

Since approximate projection is utilized, in some iterations the constructed direction is not admissible (i.e. it could not guarantee the minimum reduction δ in potential function), and exact projection is required. In order to prevent the bad effects of the ill-conditioned basis matrix over the constructed decent direction, the basis representation of $N(A)$ is changed (i.e. the basis of the ST), before the exact projection. This ST change means that a new basic feasible solution is reached. First, it is checked whether it is the optimum solution or not. If the optimum basic feasible solution is not reached, a new interior vector is found via line search over the potential function between the current solution and the new basic solution. This type of tableau update is $O(n^2 m)$. For assuring the numerical stability the pivot rule used in finding the basic solution is different than the one in the Simplex Method.

The pivot rule is:

Since the current solution is strictly positive, while entering a nonbasic variable into the basis it is possible to decrease it as well. Therefore, every nonbasic variable enters the basis ones, by increasing or decreasing its value. For the leaving variable, among the variables with non-zero entries in ST at the required column (i.e. the column of the entering variable) choose the one which has the smallest value in the current interior solution, so that the basis matrix will be kept well-conditioned.

4.2 Projection onto $N(A)$

In each iteration of the Karmarkar's algorithm, the transformed cost vector in Y-space have to be projected onto the null space of constraint matrices in Y-space to get an admissible moving direction, $p = P_{N(\frac{AD}{eT})} Dc$. If the exact projection is applied to the

vector Dc the direction is:

$$p = \left[I - DA^T(AD^2A^T)^{-1}AD - \frac{ee^T}{n} \right] Dc$$

Let $M = DA^T(AD^2A^T)^{-1}AD$, then MDc is the costly term of the calculation of the direction vector, p. It requires $O\,(n^3)$ arithmetic operations whereas the other terms require $O\,(n)$.

Since in each iteration the D matrix changes, M has to be recalculated, and its time complexity directly effects the overall performance of the algorithm. Therefore several methods, like Least Square, Chelosky Factorization, etc... are used to calculate the MDc approximately, and reduce the time complexity of the algorithm.

The STBAP Algorithm also utilizes approximate projection, but contrary to most approximation techniques the scheme used causes only very small round-off errors, like the implementation of Goldfarb & Mehrotra [13]. Because the idea is not to approximate the space, $N(A)$, but to use a proper subsection of it, $\eta(A) \subset N(A)$.

Let $\hat{p} = (I - M)Dc$, then another representation of the direction vector is ; $\hat{p} = ZZ^TDc$, where Z is n orthonormal basis of $N(A)$. The simplex tableau basis of N(AD) is $\bar{Z} = \begin{bmatrix} -D_B^{-1}B^{-1}ND_N \\ I \end{bmatrix}$ and it is orthonormalized by Gram-Schmidt technique to form Z. One can calculate the exact projection over Dc using the whole nullspace basis, Z. In the case of approximate projection a subset of the basis vectors is used. Basis vectors are the columns of Z i.e. z^i. Then the approximate direction is; $\tilde{p} = Z_k Z_k^T Dc$, where $Z_k = \{z^i : i \in I_k\}$, and $0 < k < 1$ is the approximation coefficient, and $I_k \subset \{1, 2, .., n-m\}$. In the implementation I_k is chosen such that ; $I_k = \{i : i = j_1, .., j_l \;,\; |\bar{c}_{j1}| \geq |\bar{c}_{j2}| \geq ... \geq |\bar{c}_{jl}| \geq ... \geq |\bar{c}_{j(n-m)}| \;,\; l = \lfloor k(n-m) \rfloor\}$, where $\bar{c}_i$ denotes the reduced cost of variable i. Therefore, $\tilde{p} = \sum_{i \in I_k} z^i z^{iT} Dc$ is the direction calculated in a *k-approximate* projection iteration which has a time complexity of $O\,(l^2 n)$.

4.3 Stopping Criteria

The STBAP algorithm terminates due to the "Simplex Tableau Test" (STT) or the original stopping criteria which is also utilized in the standard Karmarkar Algorithm implementation as well.

The algorithm is stopped by STT if the current ST basis is the optimum basis. STT gives a positive answer if i) the first row of the ST is nonnegative, ii) the last column of the ST is nonpositive.

4.4 The Algorithm

In the previous section the termination of the algorithm is discussed. But in order to commence the iteration a strictly positive solution vector, x^0 is required. In the algorithm this difficulty is solved by using the "Big-M" method. First, the dimension of the problem, n is increased by one. Secondly, the constraint matrix and cost vector are modified as; $\hat{A} = [A, b - Ae]$, $\hat{c} = [c, M]$ where $M \in R$ is very large number. Then $x^0 = e \in R^{n+1}$ is a strictly positive solution to the new problem, and it releases the assumption (iii). Since M is very large at the optimum solution, $x^*_{n+1} = 0$.

After the conversion of the problem into $\tilde{\text{LP}}$, the algorithm starts with the initial interior point, $x^0 = e$ and a partition of the constraint matrix, $P\tilde{A} = [B, N]$, where B is any nonsingular $m \times m$ submatrix of $\tilde{A}$ and P is a permutation matrix, which will be ignored throughout the section. Using the partition initial ST is constructed and its optimality is checked by STT. Unless the optimum basis is attained the iterations are commenced. STy is built using the current interior solution, x^k. Then, approximate projection scheme is applied to the Dc vector and a feasible direction, p in Y-space is calculated. The feasible direction is first tested if it is a decent direction by calculating the directional derivative of the potential function at the center of the simplex. If it is a decent direction, the step size, α along the direction via line search is found, and the reduction at the potential function due to the decent direction with the found step size is calculated. If the reduction is above a preset value, in the implementation it is taken $\delta_{min} = 0.1$, the new iterate point in Y-space, $y = \bar{e} - \alpha rp \quad \alpha \in (0, 1)$ is back transformed to get the next interior solution, x^{k+1}. Then the next iteration starts.

But, in the case where at least one of the two tests fail; i) the basis changes by means of the pivoting rule defined in the "Simplex Tableau" section and the new basis is checked for optimality by STT, ii) if the optimal basis is not achieved an exact projection scheme is done in the next iteration. Then the algorithm continues on iterations using approximate projection again.

The itemized structure of the STBAP Algorithm is given as follows:

0) Initialization. Choose a nonsingular basis of the constraint matrix, A. If it is the optimal basis, STOP
   else, set $k = 0$, $flag = 0$ and start iterations.

1) Calculate the STy.
   If $flag = 0$ goto (2),
   else goto (3).

2) Approximate Projection. Using the reduced cost information, select a subset of feasible directions, and project the cost vector onto the subspace defined by the chosen directions, $p = P_{app}\, Dc$.
Goto (4).

3) Exact Projection. Using the ST calculate the orthogonal basis of the $N(AD)$ and the projection matrix $P_{N(AD)}$. Project the cost vector onto $N(AD)$, $p = P_{N(AD)}\, Dc$

4) Project p onto $N(e^T)$, $\hat{p} = P_{N(e^T)}\, p$ and calculate the directional derivative due to $\hat{p}$ at point $\frac{1}{n}e$.
If the directional derivative is negative, goto (6).

5) Using ST find a basic solution (bfs) which has a better objective value than the current interior point, x^k.
Apply STT; if it is the optimal basic feasible solution (bfs), STOP
else i) find a better interior point via line search between the current point and the found bfs; ii) set $flag = 1$ goto (1).

6) Calculate the step size, α and find the new interior point,
$x^{k+1} = T^{-1}(\frac{1}{n}e - \alpha r\hat{p}\,,\, x^k)$. Set $k = k + 1$ and goto (1).

5. COMPARISON and RESULTS

In order to compare STBAP Algorithm with the standard version of Karmarkar's Algorithm, two computer programs were coded in C language. The implementation by D. Gay was chosen as the standard Karmarkar Algorithm. In both codes the same data structures were used (i.e. neither of them utilize the sparsity or structure of the problem and no preprocessing was done in both of them). Test problems are randomly generated in laboratory. The generated problems are in two different densities, 10% and 80%. First, A matrix is built via the given density by random numbers generated between -10 and 10. Then the rhs vector is calculated as $b = Ae$, so that problem is always feasible and satisfies the assumption (iii). The cost coefficients are calculated between the limits -99 and 99 with a density of 40%. In order to construct the $\tilde{\mathbf{L}}\tilde{\mathbf{P}}$ problem, the value of the generated problems re calculated by using MINOS 5.0 on Data General MV 2000 mainframe. And for the computations the Sun Workstations were used.

The results of both algorithms are stated in the tables 1-2 and figures 1-4 at Appendix C. In the tables the following abbreviations are used. "App. Percent" : Percentage of approximation, $100k$ (k is the approximation coefficient). "# of **ST** change" : Number of basis changes during the algorithm. "# of Exact Proj." : Number of iterations that exact projection scheme is done. "Total iter." : Total number of iterations. "Termination" : In termination the stopping criteria; **CS** means optimum ST cause the termination, **G** means the gap between the optimum and current objective value is less than $\epsilon = 1 \times 10^{-6}$, **IL1** & **IL2** are used for termination due to iteration limit with a success or a fail to get the optimum, solution respectively. Where the iteration limit is taken as 51. The results due to the 100% approximation form the class of problems solved by the standard Karmarkar Algorithm.

If the completion times are compared (see the figures 1 & 2), STBAP algorithm seemed to be advantageous for all the approximation schemes. This big difference is due to the time spend on calculation of the whole projection matrix in the standard implementation, whereas it is rarely calculated in the STBAP algorithm. In the last two figures, the total iterations and the number of iterations with exact projections (EP) are given. In terms

of the iterations, the low (10%) app. and the high (75%) app. schemes have closer total iterations to the standard implementation. As expected, the number of EP iterations decreased as the approximation coefficient k, increase. Because the approximated projected vector is closer to the exact projected vector as the approximation coefficient increases, of course the time per iteration increases as well. The best approximation scheme is seemed to use a low approximation coefficient (less than 20%, like 10%).

6. CONCLUSION

In this study, it is tried to use the Simplex Tableau information in the implementation of Karmarkar Algorithm. Two computer programs are coded in order to compare this implementation with the standard implementation by D. Gay. The results of the comparison between the **STBAP** Algorithm and the standard Karmarkar Algorithm implementation, shows that the idea of embedding the ST information into the interior point algorithms is promising. Therefore, a better computer code of the algorithm is needed to carry on the studies over the large scale problems. Also, the first assumption that the optimum value of the problem is zero has to be relaxed.

In the future, the possible research topics are:

- Investigate the best frequency of the basis changes in ST.
- Cooperate the stopping criteria discussed in App. B with the STBAP Algorithm and find the most compatible one with the algorithm.
- Is it applicable to use a size reduction scheme in the STBAP Algorithm?

A. PROOF OF POLYNOMIALITY OF THE KARMARKAR ALGORITHM

It is the bound on the number of iterations which makes the algorithm polynomial. In this section we will try to investigate why the algorithm stops in at most $o(nL)$ iterations [2]. The stopping criterion is:

$$c^T x^k \leq 2^{-L}$$

For simplicity I will use c , $\bar{e}$, P instead of $\hat{c}$, $\frac{1}{n}e$, and projection onto the null space of B , $P_{N(B)}$ matrices, through out this section.

Lemma 1: Let $X = \bar{e} + N(B) = \{x \in R^n : Ax = 0 , e^T x = 1\}$. Then $\exists u \in N(B)$ s.t. $\bar{e} + u \in X$, $c^T \bar{e} \leq c^T u$.

Proof:

$$\forall x \in X \qquad x = \bar{e} + x_o \quad x_o \in N(B) \tag{1}$$

$$\forall x_o \in N(B) \qquad c^T x = c^T(P x_o) = (c^T P^T) x_o = (Pc)^T x_o \tag{2}$$

Therefore,

$$c^T x = c^T \bar{e} + (Pc)^T x^o \quad \forall x \in X . \tag{3}$$

Since the feasible region in the image space is contained by the ball $B(\bar{e}, R)$ in X, and $-Pc$ is the minimizing direction in $B(\bar{e}, R)$ projected onto X, for the function $f(x) = c^T x$, $f(\bar{e} - Ru) \leq f(x^*)$, where $u = \dfrac{Pc}{\|Pc\|}$.

Or, $c^T \bar{e} - R(Pc)^T u \leq c^T x^* = 0$, by eqn. 3 .

Since $R = \sqrt{\frac{n-1}{n}} < 1$ and $(Pc)^T u = c^T u \geq 0$,

$$c^T\bar{e} - c^T u \leq c^T\bar{e} - Rc^T u \leq 0 \ .$$

$$c^T\bar{e} \leq c^T u \tag{4}$$

□.

Lemma 2: For $\|x\| \leq \beta < 1$, we have the inequalities,

$$x - \frac{x^2}{2(1-\beta)^2} \leq \log(1+x) \leq x \ . \tag{5}$$

Proof: Let $F(t) = \log(1+t)$. By the Taylor's theorem,

$$F(t) = F(0) + F'(0)t + \frac{1}{2}F''(\theta)t^2 \quad for\ some|\theta| \in (-\beta, \beta)$$

Then

$$\log(1+t) = t - \frac{t^2}{2(1+\theta)^2}$$

Since $1+\theta \geq 1-\beta$ and $\frac{t^2}{2(1+\theta)^2} \geq 0$, from the equality

$$t - \frac{t^2}{2(1+\theta)^2} \leq \log(1+t) \leq t$$

□.

Theorem 1: For $\alpha \in (0,1)$, there exist a $\delta(\alpha)$, depending only on α such that,

$$\Phi(\hat{c}, y) - \Phi(\hat{c}, \bar{e}) \leq \delta(\alpha) \tag{6}$$

Proof: Putting $y = \bar{e} - \hat{\alpha}u$, $u = \dfrac{Pc}{\|Pc\|}$, $\bar{e} = \frac{1}{n}e$ into the LHS,
where $\hat{\alpha} < r \cong \frac{1}{n}$ for $n \gg 1$.

$$\begin{aligned}
\Phi(c,y) - \Phi(c,\bar{e}) &= n\log(c^T\bar{e} - \hat{\alpha}c^T u) - \sum_{j=1}^{n}\log(\bar{e} - \hat{\alpha}u)_j \\
&\qquad -n\log c^T\bar{e} + \sum_{j=1}^{n} log\,\frac{1}{n} \qquad (7) \\
&= n\log\left(\frac{c^T\bar{e} - \hat{\alpha}c^T u}{c^T\bar{e}}\right) - \sum_{j=1}^{n}\log(n\bar{e} - n\hat{\alpha}u)_j \qquad (8)
\end{aligned}$$

By lemma 1,

$$n\log\left(\frac{c^T\bar{e} - \hat{\alpha}c^T u}{c^T\bar{e}}\right) \leq n\log(1-\hat{\alpha}) \tag{9}$$

$$\sum_{j=1}^{n} \log(n\bar{e} - n\hat{\alpha}u)_j = \sum_{j=1}^{n} \log(1 - \hat{\alpha}u_j) \tag{10}$$

Using lemma 2 and changing $\hat{\alpha}$ by $\frac{\alpha}{n}$, Equations 9 and 10 turn out to be;

$$n \log\left(\frac{c^T\bar{e} - \hat{\alpha}c^Tu}{c^T\bar{e}}\right) \leq -\alpha \tag{11}$$

$$-\sum_{j=1}^{n} \log(n\bar{e} - n\hat{\alpha}u)_j \leq \sum_{j=1}^{n} -\alpha u_j + \frac{\alpha^2 u_j^2}{2(1-\beta_j)^2} \tag{12}$$

Since $\alpha u_j < \beta_j$ and $u \in N(e^T)$ is an unitary vector Equation 12 becomes,

$$\sum_{j=1}^{n} -\alpha u_j + \frac{\alpha^2 u_j^2}{2(1-\beta_j)^2} \leq \sum_{j=1}^{n} -\alpha u_j + \frac{\alpha^2 u_j^2}{2(1-\alpha)^2} \tag{13}$$

$$\leq \frac{\alpha^2}{2(1-\alpha)^2} \tag{14}$$

Put Equations 11 and 14 into Equation 8,

$$\Phi(c, y) - \Phi(c, \bar{e}) \leq -\delta(\alpha)$$

where $\delta(\alpha) = \alpha - \frac{\alpha^2}{2(1-\alpha)^2} > 0$.

□.

B. STOPPING CRITERIA

In this section, some of the stopping criteria rules by D.Gay [11], V.V.Kovacevic-Vujcic [22], Y.Ye [28] and Kojima [20] will be discussed. The rule by R.A. Tapia [25] was stated in chapter 4.

M. Kojima [20] stated a sufficient condition for a variable of a LP to be a basic variable at all optimum solution. He worked on the canonical LP for Karmarkar's algorithm, **KP**. The proposed conditions are:

- $rank \begin{pmatrix} A \\ e^T \end{pmatrix} = m+1$, for $A \in R^{m \times n}$.
- $x^k \in int(\mathcal{KP})$, where $int(\mathcal{KP})$ is used to indicate the set of interior points of problem KP.
- $c^T x^k > 0$.
- $c^T x^* \leq 0$.

where (x^k) is a sequence of vectors in R^n and x^* is the optimum solution vector.

Indeed, these conditions are satisfied by the sequence of interior points (x^k), generated by any proper variant of the Karmarkar Algorithm. Then, the test is:
For $i = 1, 2, .., n$ define the piecewise linear function $g^i : R \rightarrow R$
s.t.

$$g^i(\mu) = min \left\{ c_j^p + \frac{c^T x^k}{n} + \left(P_j^i + \frac{1}{n} \right) \mu \ : \ j \neq i \right\}$$

where P^i denotes the i^{th} column of the projection matrix, P and $c^p = PDc$.
Suppose that

$$g^i(\mu) > 0 \quad for \ some \ \ \mu \geq 0 \ ,$$

then $x_i > 0$ *for any optimum solution.*
The proof is given in the related paper by a contradiction. It is stated that the test is not effective for the first κ iterations and the value of the κ can be estimated by intuition.

V.V. Kovacevic-Vujcic [22] proposed a sequence of vectors $(\bar{x}^k)$ related to the sequence (x^k) generated by an interior point method have a higher degree of convergence to the optimum solution x^* than the original interior point solution (x^k). It is stated that different interior point methods share the same property that the normalized sequence of the search directions $\frac{x^{k+1}-x^k}{c^T x^{k+1}-c^T x^k}$ converges to the same direction.

Let us consider the Karmarkar's canonical form problem, KP with his assumptions[1] and a sequence of interior points, (x^k). If (x^k) satisfies the following conditions:

- $x^k \in int(\mathcal{KP}) \quad k = 1, 2, ...$
- $x^{k+1} \neq x^k \quad k = 1, 2, ...$
- $\lim_{k\to\infty} x^k = \bar{x}$, where $\bar{x} \in KP$.
- $\lim_{k\to\infty} \frac{x^{k+1}-x^k}{\|x^{k+1}-x^k\|} = s$
- $\exists\, T > 0 \;\; s.t. \;\; \bar{x} - ts > 0\,, \quad 0 < t \leq T$

then the procedure of generating the *auxiliary* sequence $(\bar{x}^k)$ is:

$$\bar{x}^{k+1} = x^k + \alpha_k(x^{k+1} - x^k) \quad k = 0, 1, ...$$

where $\alpha_k = min\left\{\frac{-x_i^k}{x_i^{k+1}-x_i^k} \;:\; x_i^{k+1} - x_i^k < 0 \;\; i = 1, 2, .., n\right\}$.

D. Gay presented a method based on the "complementary slackness" idea for finding the optimum basis. Let P and D defines the primal and dual problems respectively.

$$P = min\{c^T x \;:\; Ax = b\,, \;\; x \geq 0\}$$

$$D = \{b^T y \;:\; A^T y \leq c\}$$

and the dual slacks are;

$$s = c - A^T y \geq 0 \;\; .$$

Then the complementary slackness conditions imply for the primal and dual optimal solutions, x^* and s^*;

$$x^{*T} s^* = 0 \;\; .$$

The triple (x, y, s) is called "strict complementary triple" if $x + s > 0$, besides the third equation.

Using the above definitions, the stopping test for a primal-dual problem pair (P, D) which have a strict complementary triple is to search for a basis;

[1] In the original paper both the problem and the vector sequence are in general form

$B = \{1 \leq i \leq n \ : \ \lim_{k\to\infty} \frac{x_i^k}{s_i^k} = \infty\} \ = \ \{1 \leq i \leq n \ : \ \lim_{k\to\infty} \frac{s_i^k}{x_i^k} = 0\}$. This test is very suitable for the primal-dual algorithms.

In the implementation a small enough threshold value, $\tau > 0$ has to be decided so that , $B \equiv \{i \ : \ \frac{s_i^k}{x_i^k} \leq \tau\}$ will give the optimum basis at iteration k. The crucial point is the problem dependence of the the threshold value, τ. This difficulty could be solved by rescaling the constraint matrix, A.

Y. Ye proposed a built-down scheme for both the Karmarkar Algorithm and the Simplex Method. This is a size reduction procedure applied to the problem until the size reduced to $rank(A) = m$ (in case of nondegeneracy), i.e. the optimum basis is reached. The scheme commences with the "optimum basis candidate" set, which is all the columns of the constraint matrix, A at the beginning. Then the columns are monotonically eliminated via a special pricing rule. The pricing rule is based on the ellipsoid theory. The dual ellipsoid that contains all the optimal dual slacks is searched, and using the complementary slackness conditions:
$s_i^* x_i^* = 0 \ \ \forall i \leq n$, one can identify the nonbasic variables at any optimal solution.

For the primal-dual pair $(P \ - \ D)$, with the assumptions of Karmarkar, let's define the transformed primal problem, $\bar{P}$;

$$\bar{P} = min\{\bar{c}^T x \ : \ \bar{A}x = 0 \ \ x \geq 0\}$$

where $\bar{A} = [A, -b]$ and $\bar{c}^T = [c^T, -z^*]$ with z^* is the optimum value of the problem P. Thus, the optimum dual slacks set is defined as;

$$S^* = \{s^* \in R_+^{n+1} \ : \ s^* = \bar{c} - \bar{A}^T y \ , \ \ y \in R^m\}$$

M.J. Todd [27] derived that the ellipsoid that contains all the optimum dual slacks is;

$$\|Ds^*\|^2 \ \leq \ (e^T D s^*)^2 \ = \ (c^T x^k - z^*)^2$$

where D is the diagonal matrix defined by

$$D \ = \ \begin{bmatrix} diag(x^k) & 0 \\ 0 & 1 \end{bmatrix}$$

We know that in any optimum dual solution the slack variable $s_i^* > 0$ implies that x_i^* is nonbasic in any primal solution. Therefore, the minimum value of every slack variable has to be calculated. Indeed, one has to solve the following optimization problems for $i = 1, 2, .., n$.

$$\begin{aligned} \textbf{EOP)} \qquad & \text{min.} \ \ s_i \\ & \text{s.t.} \ \ s = \bar{c} - \bar{A}^T y \\ & \qquad \|Ds\| \leq \varepsilon^k \end{aligned}$$

where $\varepsilon^k = (\bar{c}^T x^k - z^*)$.

If the minimum value of the problem **EOP** is positive for some i, then the corresponding variable in primal problem, x_i is nonbasic in any optimum solution. Thus the i^{th} column of the constraint matrix can be eliminated from the "candidate" set.

C. RESULTS OF THE TEST PROBLEMS

In this section the results of the both algorithms and the graphs demonstrating the performance of different approximation schemes are given. In the tables the following abbreviations are used. "App. Percent" : Percentage of approximation, $100k$ (k is the approximation coefficient). "# of **ST** change" : Number of basis changes during the algorithm. "# of Exact Proj." : Number of iterations that exact projection scheme is done. "Total iter." : Total number of iterations. "Termination" : In termination the stopping criteria; **CS** means optimum ST cause the termination, **G** means the gap between the optimum and current objective value is less than $\epsilon = 1 \times 10^{-6}$, **IL1** & **IL2** are used for termination due to iteration limit with a success or a fail to get the optimum, solution respectively. Where the iteration limit is taken as 51. The results due to the 100% approximation form the class of problems solved by the standard Karmarkar Algorithm.

Problem	App. Percent	# of ST change	# of Exact Proj.	Total Iter.	Time(min.)	Termination
Prob. 1	10	5	4	27	9:51	CS
Prob. 2	10	8	7	31	15:22	CS
Prob. 3	10	5	4	18	10:53	CS
Prob. 4	10	6	5	28	11:14	CS
Prob. 5	10	5	4	21	9:16	CS
Prob. 6	10	4	3	23	7:40	CS
Prob. 7	10	6	6	21	12:29	CS
Prob. 8	10	7	7	27	14:31	CS
Prob. 9	10	4	4	30	9:48	CS
Prob.10	10	6	5	31	11:40	CS
Prob. 1	25	4	3	35	12:56	CS
Prob. 2	25	3	3	51	14:58	IL1
Prob. 3	25	5	4	48	16:30	CS
Prob. 4	25	2	2	51	12:46	IL1
Prob. 5	25	4	4	25	12:43	CS
Prob. 6	25	5	4	32	13:41	CS
Prob. 7	25	5	4	30	13:51	CS
Prob. 8	25	3	3	35	12:30	CS
Prob. 9	25	2	2	51	12:42	IL1
Prob.10	25	5	4	42	15:25	CS
Prob. 1	50	3	3	36	12:22	CS
Prob. 2	50	3	2	36	10:46	CS
Prob. 3	50	3	2	35	8:59	CS
Prob. 4	50	3	2	38	11:16	CS
Prob. 5	50	3	2	35	10:42	CS
Prob. 6	50	4	3	48	14:29	CS
Prob. 7	50	3	2	32	10:21	CS
Prob. 8	50	3	2	35	10:53	CS
Prob. 9	50	2	1	22	6:54	CS
Prob.10	50	2	1	39	11:46	CS
Prob. 1	75	2	1	27	14:30	CS
Prob. 2	75	2	1	22	12:34	CS
Prob. 3	75	3	2	23	14:47	CS
Prob. 4	75	3	2	24	15:10	CS
Prob. 5	75	3	2	33	18:37	CS
Prob. 6	75	2	1	17	10:21	CS
Prob. 7	75	2	2	23	14:45	CS
Prob. 8	75	2	1	22	12:39	CS
Prob. 9	75	2	1	18	11:01	CS
Prob.10	75	3	2	25	15:29	CS
Prob. 1	100		20	20	54:01	G
Prob. 2	100		18	18	43:04	G
Prob. 3	100		18	18	41:57	G
Prob. 4	100		22	22	52:42	G
Prob. 5	100		15	15	33:17	G
Prob. 6	100		17	17	40:01	G
Prob. 7	100		18	18	41:20	G
Prob. 8	100		17	17	38:41	G
Prob. 9	100		17	17	41:16	G
Prob.10	100		15	15	30:54	G

Table C.1: Results of the problems of small size (50 × 100) with density 10%.

Problem	App. Percent	# of ST change	# of Exact Proj.	Total Iter.	Time(min.)	Termination
Prob. 1	10	6	5	30	15:43	CS
Prob. 2	10	6	5	27	15:38	CS
Prob. 3	10	4	4	14	12:01	CS
Prob. 4	10	7	6	35	18:14	CS
Prob. 5	10	6	5	22	14:35	CS
Prob. 6	10	6	6	27	16:50	CS
Prob. 7	10	6	5	25	17:12	CS
Prob. 8	10	6	6	20	17:05	CS
Prob. 9	10	6	6	27	17:06	CS
Prob.10	10	6	6	27	17:20	CS
Prob. 1	25	4	4	51	17:40	IL1
Prob. 2	25	3	4	41	17:48	IL1
Prob. 3	25	6	5	48	17:15	CS
Prob. 4	25	4	5	51	22:21	IL1
Prob. 5	25	6	5	25	20:19	CS
Prob. 6	25	5	5	32	19:25	CS
Prob. 7	25	5	4	30	18:45	CS
Prob. 8	25	4	5	51	20:10	IL1
Prob. 9	25	4	4	34	15:06	CS
Prob.10	25	4	3	50	16:01	CS
Prob. 1	50	3	2	43	16:48	CS
Prob. 2	50	3	2	39	16:16	CS
Prob. 3	50	3	3	46	19:06	CS
Prob. 4	50	3	3	51	20:14	IL1
Prob. 5	50	3	3	45	18:42	CS
Prob. 6	50	3	2	42	16:33	CS
Prob. 7	50	3	2	23	13:55	CS
Prob. 8	50	3	3	51	20:25	IL1
Prob. 9	50	2	2	36	15:01	CS
Prob.10	50	2	3	51	21:05	IL1
Prob. 1	75	2	1	19	14:25	CS
Prob. 2	75	2	2	21	17:08	CS
Prob. 3	75	3	2	32	22:20	CS
Prob. 4	75	3	3	31	23:04	CS
Prob. 5	75	3	3	30	20:41	CS
Prob. 6	75	3	2	26	17:00	CS
Prob. 7	75	3	2	31	21:33	CS
Prob. 8	75	4	3	38	27:56	CS
Prob. 9	75	2	1	35	21:29	CS
Prob.10	75	4	3	36	24:58	CS
Prob. 1	100		17	17	43:04	G
Prob. 2	100		14	14	36:14	G
Prob. 3	100		22	22	56:24	G
Prob. 4	100		16	16	37:41	G
Prob. 5	100		20	20	50:30	G
Prob. 6	100		18	18	46:44	G
Prob. 7	100		19	19	48:10	G
Prob. 8	100		18	18	38:22	G
Prob. 9	100		18	18	42:24	G
Prob.10	100		16	16	39:54	G

Table C.2: Results of the problems of small size (50×100) with density 80%.

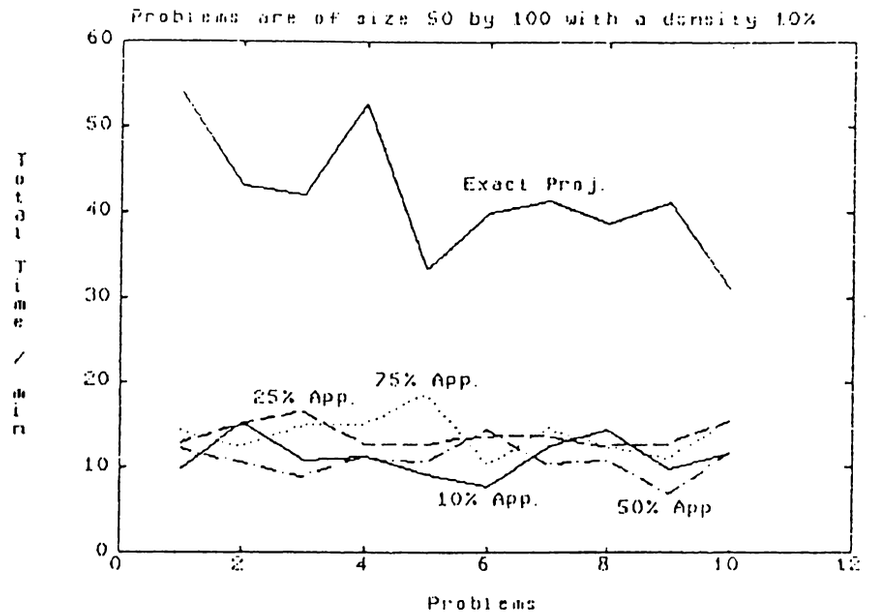

Figure C.1:

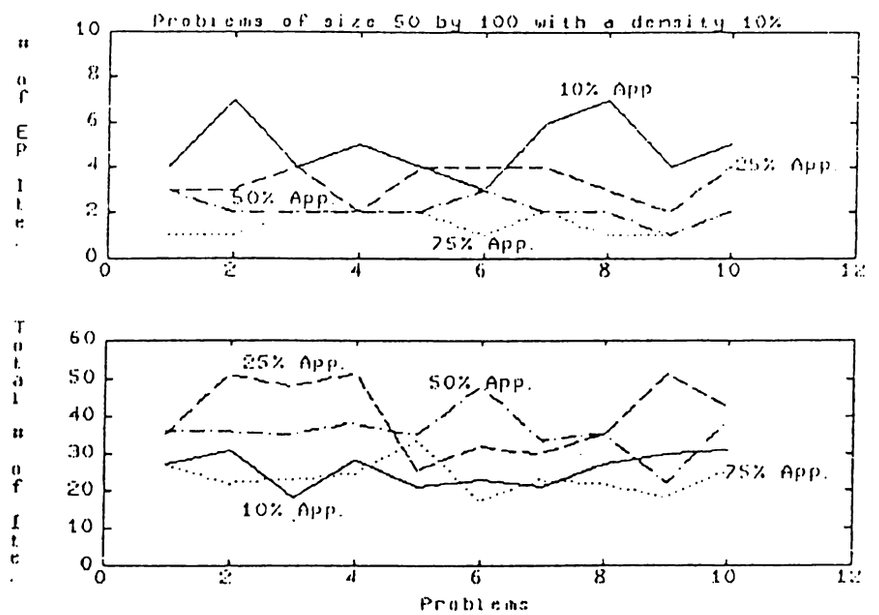

Figure C.2:

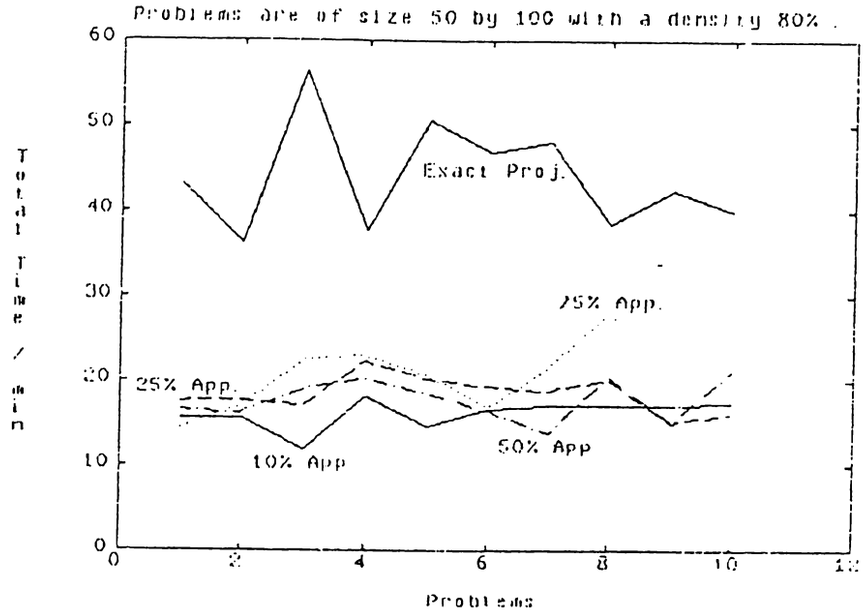

Figure C.3:

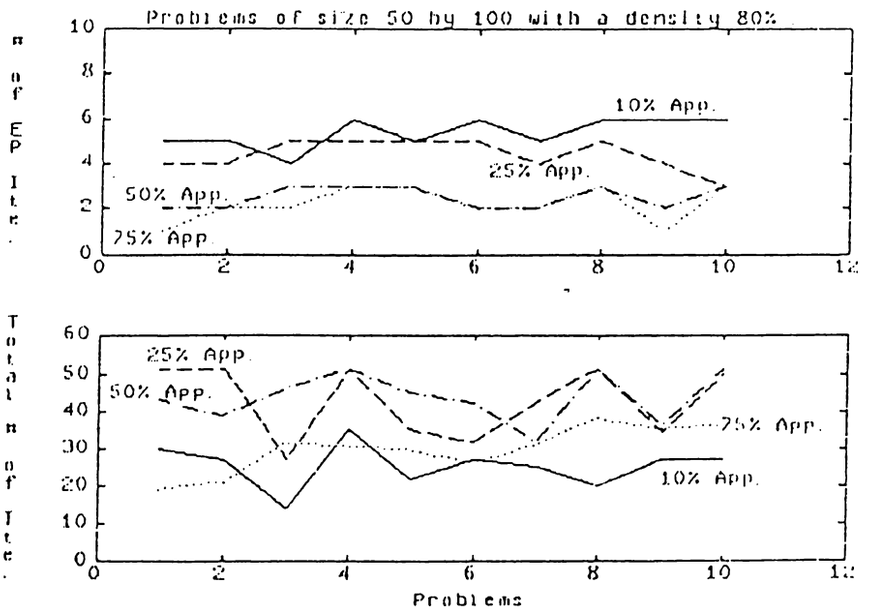

Figure C.4:

REFERENCES

[1] Adler I., Karmarkar N., Resende M.G.C. and Veiga G., Data Structures and Programming Techniques For the Implementation of Karmarkar's Algorithm, Mathematical Programming 44, 297-335, 1989.

[2] Akgül M., A Short Proof of Karmarkar's Main Result, Doğa - Turkish Journal of Mathematics 14, 48-55, 1990.

[3] Anstriecher K.M., A Monotonic Projective Algorithm For Fractional Linear Programming, Algoritmica 1, 483-498, 1986.

[4] Bland R.G., New Finite Pivoting Rules For the Simplex Method, Mathematics of Operation Research 2, 103-107, May 1977.

[5] Bland R.G., Goldfarb D., and Todd M.J., The Ellipsoid Method: A Survey, Operations Research 2, 1039-1091, 1981.

[6] Borgwardt K.H., The Simplex Method - A Probabilistic Analysis, Springer-Verlag, 1987.

[7] Dantzig G.B., Maximization of a Linear Function of Variables Subject to Linear Inequalities, Chap. XXI of Activity Analysis of Production and Allocation Codes Commission Monograph 13, T.C. Koopmer, John Wiley, New york, 1951.

[8] Edmonds J., Exponential Growth of the Simplex Method for Shortest Path Problems, University of Waterloo, 1970.

[9] Ford L.R.Jr. and Fulkerson D.R., Constructing Maximal Dynamic Flows From Static Flows, Operations Research Vol.6, 419-433, 1958.

[10] Gay D.M., A Variant of Karmarkar's LP Algorithm For Problems in Standart Form, Mathematical Programming 37, 81-90, 1987.

[11] Gay D.M., Stopping Tests That Compute Optimal Solutions For Interior Point LP Algorithms, Numerical Analysis Manuscript 89-11, **AT & T** Bell Laboratory, Dec. 1989.

[12] Goldfarb D. and Reid J.K., A Practical Steepest-Edge Simplex Algorithm, Mathematical Programming 12, 361-371, 1977.

[13] Goldfarb D. and Mehrotra S., Relaxed Variants of Karmarkar's Algorithm for Linear Programs With Unknown Optimal Objective Value, Mathematical Programming 40, 183-195, 1988.

[14] Goldfarb D. and Mehrotra S., A Relaxed Revision of Karmarkar's Method, Mathematical Programming 40, 289-315, 1988.

[15] Jeroslow R., The Simplex Algorithm With The Pivot Rule of Maximizing Criterion Improvement, Discrete Mathematics 4, 367-377, 1973.

[16] Karmarkar N., A New Polynomial Time Algorithm For LP, Combinatorica Vol.4, 375-395, 1984.

[17] Karmarkar N.K. and Ramakrishnan K.G., Implementation and Computational Results of the Karmarkar Algorithm for Linear Programming, Using an Iterative Method for Computing Projections, AT & T Bell Laboratories, Research Report, 1988.

[18] Khachian L.G., A Polynomial Algorithm in LP, Soviet Mathematics Doklady 20, 191-194, 1979.

[19] Klee V. and Minty G.J., How Good is the Simplex Algorithm?, Mathematical Note no.643, Mathematics Research Laboratory, Boeing Scientific Research Laabs, Feb. 1970.

[20] Kojima M., Determinig Basic and Nonbasic Variables of Optimal Solutions in Karmarkar's New LP Algorithm, Algorithmica Vol.1, 499-515, 1986.

[21] Kojima M. and Tone K., An Efficient Implementation of Karmarkar's New LP Algorithm, Technical Report No. B-180, Department of Information Sciences, Tokyo Institute of Technology, Japan, April 1986.

[22] Kovacevic V.V. and Vujcic , Improving the Rate of Convergence of the Interior Point Methods For LP, Faculty of Organizational Sciences, Belgrade University, 1989.

[23] Lemke C.E., The Dual Method of Solving The Linear Programming Problem, Naval Research Logistics Quarterly 1, 36-47, 1954.

[24] Oohori T. and Ohuchi A., An Efficient Implementation of Karmarkar's Algorithm For Large Scale Linear Programs, Tech. Report, Hokkaido Istitute of Technology, Japan, Sept. 1988.

[25] Tapia R.A. and Zhang Y., A Fast Optimal Basis Identification Technique For Interior Point LP Methods, Technical Report No. 89-1, Department of Mathematical Sciences, Rice University, Oct. 1989.

[26] Todd M.J. and Burrell B.P., An Extension of Karmarkar's Algorithm For Linear Programming Using Dual Variables, ALgoritmica 1, 409-424, 1986.

[27] Todd M.J., Improved Bounds & Containing Ellipsoids in Karmarkar 's Linear Programming Algorithm, Mathematics of Operations Research 13, 409-424, 1988.

[28] Ye Y., A Built-down scheme for Linear Programming, Mathematical Programming Vol. 46, 61-73, 1990.

[29] Cheng M.C., New Criteria For The Simplex Algorithm, Mathematical Programming Vol. 19, 230-236, 1980.