Progress in Linear Programming: Interior-Point Algorithms

Yinyu YE*

March 1994 (Revised August 1994)

Abstract: According to current estimates, more than $100 million in human and computer time is invested yearly in the formulation and solution of linear programming problems. Businesses, large and small, use linear programming models to optimize communication systems, to schedule transportation networks, to control inventories, to plan investments, and to maximize production.... In this article we describe some recent developments in linear programming. We highlight progress in interior-point algorithms during the last ten years.

*Department of Management Sciences, The University of Iowa, Iowa City, Iowa 52242, USA. Research supported in part by NSF grant DDM-9207347, an Iowa Business School Summer Grant, and K.C. WONG Education Foundation, Hong Kong through Academia Sinica, Beijing, China.

1 Introduction

Linear programming is a fundamental optimization problem. The essential ingredients of an optimization problem are a set of **decision variables**, an **objective function** of the variables to be maximized or minimized, and a set of **constraints** that characterize acceptable values of the variables. For example, an inequality constraint specifies that a certain function of the variables must be greater than or equal to a constant.

In a linear programming problem, or simply linear program, the objective function and all constraint functions are linear. For a linear program with n variables, represented by an n-dimensional (column) vector x, the objective function is conventionally denoted by $c^T x$, where c is an n-dimensional vector of coefficients; it is usually, without losing generality, assumed that $c^T x$ is to be minimized. Here, the superscript ^{T} denotes the transpose operation.

The constraints can be stated in a variety of forms. Fortunately, these forms are mathematically equivalent, although certain choices may have algorithmic implications. In particular, the well-known **standard form** of a linear programming problem involves

$$\begin{array}{lll} (LP) & \text{minimize} & c^T x \\ & \text{subject to} & Ax = b, \; x \geq 0, \end{array}$$

where constraint matrix A is $m \times n$ and constraint vector b has dimension m. With this form, all general constraints are equalities, and the only inequality constraints are simple bounds on all variables. In words, this problem is to minimize $c^T x$ subject to $Ax = b$ and $x \geq 0$.

The linear program is said to be **feasible** if and only if its constraints are consistent, i.e., there is a vector x which satisfies all equality and inequality constraints; otherwise, it is called **infeasible**. The problem is called **unbounded** if there is a sequence $\{x^k\}$ such that x^k is feasible for all k but the objective function values $c^T x^k$ tend to $-\infty$.

Given any linear program, designated as the **primal**, there is always a closely related linear program called the **dual**. When the primal is in standard form, the dual problem is

$$\begin{array}{lll} (LD) & \text{maximize} & b^T y \\ & \text{subject to} & A^T y \leq c, \end{array}$$

where the vector of decision variables y has dimension m. In words, this problem is to maximize $b^T y$ subject to $A^T y \leq c$. We usually use $s = c - A^T y$ to represent the difference of c and $A^T y$, which is called the slack vector. Thus, $c - A^T y \geq 0$ is equivalent to $s \geq 0$ in the dual. Denote by $\mathcal{F}$ the set of all (x, y, s) such that x and (y, s) are feasible for the primal and dual, respectively, and denote by $\mathcal{F}^0$ the set of interior points in $\mathcal{F}$ with $(x, s) > 0$.

If x and y are feasible for the primal and dual, respectively, it is easy to show that $b^T y \leq c^T x$, so that there is always a **duality gap** between the primal and the dual objective function values. However, these values are equal when x and y are optimal primal and dual solutions. This relation is the fundamental **duality theorem** of linear programming, which is a cornerstone for all algorithms designed for solving linear programs. It can be easily verified that zero duality gap is equivalent to $x^T s = 0$, which is called the **complementary slackness condition**; it implies that in each nonnegative pair (x_j, s_j), $j = 1, 2, ..., n$, at least one of the two is zero.

The aim of this article is to describe some of the recent developments in linear programming and to suggest some directions in which future progress might be made. We will concentrate on the research in the last ten years on interior-point algorithms for linear programming, stemming from the fundamental paper of Karmarkar [19].

2 A brief history of linear programming

Linear inequalities define a polyhedron, whose properties have been studied by mathematicians for centuries. Ancient Chinese and Greeks studied calculating volumes of simple polyhedra in three-dimensional space. Fourier's fundamental research connecting optimization and inequalities dates back to the early 1800s. At the end of 19th century, Farkas and Minkowski began basic work on algebraic aspects of linear inequalities. In 1910 De La Vallée Poussin developed an algebraic technique for minimizing the infinity-norm of $b - Ax$ that can be viewed as a precursor of the simplex method.

Beginning in the 1930s, such notable mathematicians as von Neumann, Kantorovich, and Koopmans were studying mathematical economics based on linear inequalities. During World War II, it was observed that decisions involving the best movement of personnel and optimal allocation of resources could be posed and solved as linear programs. Then, linear programming began to assume its current popularity.

A solution of a linear program always lies at a vertex of the feasible region, which itself is a polyhedron. Unfortunately, the number of vertices associated with a set of n inequalities in m variables can be exponential in the dimensions—in this case, up to $n!/m!(n-m)!$. Except for small values of m and n, this number is so large as to prevent examining all possible vertices for searching an optimal vertex.

The simplex method, invented in the mid-1940s by George Dantzig, is a procedure for examining optimal candidate vertices in an intelligent fashion. It constructs a sequence of adjacent vertices with improving values of the objective function. Thus, the method travels along edges of the polyhedron until it hits an optimal vertex. Improved in various way in the intervening four decades, the simplex method continues to be the workhorse algorithm for solving linear programming problems. On average, the number of vertices or iterations visited by the simplex method seems to be roughly linear in m and perhaps logarithmic n (see Chvátal [8]).

Although it performs well on average, the simplex method will indeed examine every vertex when applied to certain linear programs. Klee and Minty in 1972 gave such an example. These examples confirm that in the worst case, the simplex method needs an exponential number of iterations to find the optimal solution. As interest in complexity theory (a computer science theory for measuring effectiveness of algorithms) grew, many researchers believed that a good algorithm should be polynomial—i.e., broadly speaking, the amount of work required to computing the solution should be bounded above by a polynomial in the "size" of the problem, say in m, n and L, where L is the number of bits required to represent integer data A, b, and c. Thus, the simplex method is not a polynomial algorithm.

In 1979, a new approach to linear programming—Khachiyan's ellipsoid method—received dramatic and widespread coverage in the international press. Khachiyan [18] has proved that the ellipsoid method, developed during the 1970s by other mathematicians Shor, Yudin and Nemirovskii, is a polynomial algorithm for linear programming. It constructs a sequence of shrinking ellipsoids with two properties: the current ellipsoid always contains the optimal solution set, and each member of the sequence undergoes a guaranteed reduction in volume, so that the solution set is squeezed more tightly at each iteration.

The ellipsoid method was intensively studied by practitioners as well as theoreticians, see Bland et al. [7]. Based on the expectation that a polynomial linear programming algorithm would in some sense be faster than the simplex method, it was a great disappointment that the best implementations of the ellipsoid method were not even close to competitive. In contrast to the situation for the simplex method, the number of steps required for the ellipsoid method to terminate was almost always close to the worst case bound—whose value, although defined by a polynomial, is typically very large. Thus, after the dust eventually settled, the prevalent view among linear programming researchers was that Khachiyan had proved a genuinely polynomial linear programming algorithm, but the simplex method remained the clear winner in practice.

This contradiction, the fact that an algorithm with the desirable theoretical property of polynomiality might nonetheless compare unfavorably with the (worst-case exponential) simplex method, set the stage for the exciting new developments. It was no wonder, then, that the announcement of Karmarkar in 1984 [19] of a new polynomial interior-point algorithm with the potential of dramatic improvement in practical effectiveness over the simplex method made front-page news in major newspapers and magazines throughout the world. The rest of our article describes the research leading from Karmarkar's algorithm.

3 Karmarkar's Algorithm

Karmarkar's algorithm is of interest to researchers because, like the ellipsoid method, the method is provably a polynomial algorithm. Karmarkar has proved that the total number of iterations required to solve the problem by his method is at most proportional to nL, and each iteration solves a system of n linear equations and requires arithmetic operations at most proportional to n^3. Thus, the iteration complexity of Karmarkar's algorithm is "of order nL" in the worst case, written as $O(nL)$; the arithmetic operation complexity of Karmarkar's algorithm is $O(n^4L)$.

Interestingly, this bound is identical to that of the ellipsoid method. However, Karmarkar has further shown that, using a partial rank-one updating scheme, one can reduce the number of arithmetic operations per iteration (average over all iterations) to $O(n^{2.5})$. Thus, the overall operation complexity of Karmarkar's algorithm is $O(n^{3.5}L)$, although the partial updating scheme may have no practical value for solving large-scale sparse linear programs.

In addition to its polynomial complexity bound, Karmarkar's algorithm is of interest because of its novel approach to linear programming. In contrast to the simplex method traveling along an edge of the feasible region, Karmarkar's algorithm moves in the interior of the region, and generates a sequence of interior points leading to an optimal solution.

Given an positive (interior) feasible point x^k for (LP), Karmarkar's method uses a projective transformation from the n-dimensional Euclidean space R^n to the $(n+1)$-dimensional R^{n+1}:

$$\hat{x} = T(x) := (\frac{(n+1)x^T(X^k)^{-1}}{e^T(X^k)^{-1}x+1}, \frac{(n+1)}{e^T(X^k)^{-1}x+1})^T$$

to embed the feasible region of (LP) onto the intersection of a regular simplex and a linear subspace:

$$\begin{array}{lll} (SLP) & \text{minimize} & \hat{c}^T\hat{x} \\ & \text{subject to} & \hat{A}\hat{x} = 0, \\ & & e^T\hat{x} = n+1, \\ & & \hat{x} \geq 0, \end{array}$$

where

$$\hat{c}^T = (c^TX^k, -z^*) \in R^{n+1}$$

$$\hat{A} = (AX^k, -b),$$

$X^k = \text{diag}(x^k)$ denotes the diagonal matrix with diagonal entries equal to the elements of x^k, e is the vector of all ones of appropriate dimension, and z^* denotes the optimal value of the linear program. (Here, we assume that z^* is known in advance for a moment.) Moreover, we see that $T(x^k) = e$, the center of the simplex $\mathcal{S} = \{x : e^Tx = n+1,\ x \geq 0\}$, is a feasible point for the problem, i.e., $\hat{A}e = 0$.

The simplex is the generalization of the equilateral triangle, and it has a nice geometric property: it contains the inscribed ball $B(1)$ and is contained in the circumscribed ball $B(\sqrt{n(n+1)})$, where

$$B(r) = \{\hat{x} :\ \hat{A}\hat{x} = 0,\ e^T\hat{x} = n+1,\ \|\hat{x} - e\| \leq r\}.$$

The minimizer, $\hat{x}^*$, over $B(\alpha)$ is easily calculated by projecting $\hat{c}$ onto the null space $\{x :\ \hat{A}x = 0,\ e^Tx = 0\}$. Since the optimal value of (SLP) is zero, the minimum of $\hat{c}^T\hat{x}$ over the circumscribed ball is nonpositive, and hence its minimum over the contained ball $B(\alpha)$, $0 < \alpha < 1$, is at most $(1 - \alpha/\sqrt{n(n+1)})$. In other words,

$$\hat{c}^T\hat{x}^* \leq (1 - \alpha/\sqrt{n(n+1)})\hat{c}^Te = (1 - \alpha/\sqrt{n(n+1)})(c^Tx - z^*).$$

Furthermore, the $(k+1)$th iterate for (LP) can be obtained by using the inverse projective transformation $R^{n+1} \to R^n$:

$$x^{k+1} = T^{-1}(\hat{x}^*) := X^k \hat{x}^*_{1:n} / \hat{x}^*_{n+1},$$

where $\hat{x}^*_{1:n}$ is the vector of the first n components of $\hat{x}^*$, and $\hat{x}^*_{n+1}$ is the $(n+1)$th component of $\hat{x}^*$ which we call the homogenizing variable.

Unfortunately, we may not achieve a corresponding decrease in the true objective function $c^T x - z^*$ of (LP), since

$$c^T x^{k+1} - z^* \neq \hat{c}^T \hat{x}^*.$$

The reason is that linear functions are not invariant under the projective transformation. Hence, instead of measuring the improvement of $c^T x$, Karmarkar used a potential function

$$\phi(x, z^*) := (n+1)\ln(c^T x - z^*) - \sum_{j=1}^{n} \ln x_j = \hat{\phi}(\hat{x}, z^*) := (n+1)\ln \hat{c}^T \hat{x} - \sum_{j=1}^{n+1} \ln \hat{x}_j.$$

defined on each strictly positive feasible point x of (LP). This potential function combines (a nonlinear transformation of) the objective function with a barrier term repelling x from the boundary of the feasible region.

Moreover, $\hat{\phi}$ is seen to be homogeneous of degree zero and invariant under the projective transformation in its second equation. For a suitable choice of α, one can show that

$$\hat{\phi}(\hat{x}^*, z^*) \leq \hat{\phi}(e, z^*) - .3,$$

and, hence,

$$\phi(x^{k+1}, z^*) \leq \phi(x^k, z^*) - .3$$

Suppose that the feasible region of (LP) is bounded. Then, starting from any initial strictly feasible point not too close to the boundary of the nonnegative orthant, after $O(nL)$ iterations we will have

$$\frac{c^T x^k - z^*}{c^T x^0 - z^*} \leq 2^{-L},$$

where x^k can be rounded to an optimal solution for (LP).

We now return to the assumption that the optimal value z^* of (LP) is known. It has been shown that z^* can be replaced by a lower bound z^0 to z^*. Then, at the kth iteration a possible new lower bound z^{k+1} may be generated as a by-product, and the potential function $\phi(x^{k+1}, z^{k+1})$ is reduced from $\phi(x^k, z^k)$ by 0.3. Thus, after $O(nL)$ iterations, we still have

$$\frac{c^T x^k - z^*}{c^T x^0 - z^0} \leq \frac{c^T x^k - z^k}{c^T x^0 - z^0} \leq 2^{-L}.$$

This lower bound technique was first proposed by Todd and Burrell [43].

An important feature of Karmarkar's algorithm is that the potential function can be used to monitor progress. As a result, on each step a line search of the potential function, along the direction generated by the algorithm, can be used to accelerate the algorithm's speed. The use of such a line search makes the algorithm naturally adaptive to the particular problem being solved, meaning its required number of iterations may not necessarily be close to the worst-case bound in practice, while retaining all theoretical convergence properties.

Although Karmarkar's algorithm converges quickly in practice and has many nice theoretical properties, it has some less desirable features as well. For example, the need for the knowledge of z^*. One can use a lower bound z to replace z^* as we discussed earlier, but the initial z^0 needs to set at -2^L in the worst case. Again this is hard to implement in practice. One can also combine (LP) and (LD) and minimize the duality gap, in which case $z^* = 0$. However, the resulting primal-dual problem has a doubled dimension, and its solution may therefore require an excessive amount of work. Another feature making the algorithm unattractive to some researchers is the use of the projective transformation, which is uncommon in optimization. Consequently, few workers in the field have much intuition regarding its value. Many people felt that similar theoretical results could be obtained for simpler algorithms, which apply more directly to the original problem.

4 The affine scaling algorithm

Given an positive (interior) point x^k for (LP), the affine-scaling algorithm uses a simple affine scaling transformation $R^n \rightarrow R^n$:

$$\hat{x}^T = T(x) := (X^k)^{-1}x$$

to reformulate (LP) as

$$\begin{array}{ll} \text{minimize} & \hat{c}^T\hat{x} \\ \text{subject to} & \hat{A}\hat{x} = b, \\ & \hat{x} \geq 0, \end{array}$$

where

$$\hat{c} = X^k c \in R^n$$

$$\hat{A} = AX^k.$$

Again, we see that e is a feasible point for the problem. Unlike the projective transformation, e may not be the "center" in the scaled feasible region, and consequently we don't have the nice geometric property possessed by the simplex: the scaled feasible region contains the inscribed ball $B(1)$ and but may not be contained in the circumscribed ball $B(\sqrt{n(n+1)})$, where

$$B(r) = \{\hat{x} : \ \hat{A}\hat{x} = b, \ \|\hat{x} - e\| \leq r\}.$$

Nevertheless, we can still minimizer $\hat{c}^T\hat{x}$ over $B(\alpha)$. Its minimizer $\hat{x}^*$ is again easily calculated by projecting $\hat{c}$ onto the null space $\{x : \hat{A}x = 0\}$. The $k+1$th iterate for (LP) can be obtained by using the inverse affine scaling

$$x^{k+1} = T^{-1}(\hat{x}^*) := X^k\hat{x}^*.$$

One can easily verified that

$$c^Tx^{k+1} = \hat{c}^T\hat{x}^* < \hat{c}^Te = c^Tx^k,$$

i.e., the objective function value is monotonically decreasing. However, we don't know how much reduction would be assured, and there is no known polynomial iteration bound for this algorithm.

Surprisingly, the algorithm appears to have been first described and analyzed by Dikin in 1967 [10] (it was independently rediscovered by several researchers in 1985). As far as the choice of α goes, it seems reasonable to choose a large value to decrease the objective function more as long as $\hat{x}^*$ remains positive. In fact, Dikin chose $\alpha = 1$ and showed that, if $\hat{x}^*$ has a zero component, it must be optimal. Other authors allow α to be greater than 1, as long as all components of $\hat{x}^*$ remain greater than γ where γ is chosen between $.1 - .05$. This corresponds to moving a proportion of the way $1-\gamma$ to the boundary of the feasible region.

The convergence of the affine scaling algorithm was proved by Dikin for $0 < \alpha \leq 1$, under the assumption of primal non-degeneracy. Recently, Tsuchiya and Muramatsu proved that for $\gamma = 1/3$, the algorithm converges for any linear program. For detailed results on the convergence of affine scaling algorithms, see [45] and the references therein.

Compared with Karmarkar's algorithm, the affine scaling algorithm has an appealing simplicity, despite the lack of a polynomial complexity bound. In the early stage, many implementations of interior-point algorithms were based on the dual affine scaling algorithm, which applies the same idea, as described above for the primal, to the dual (LD) (see Adler et al. [1]).

5 The path-following algorithm

Although at its introduction Karmarkar's algorithm represented a novel approach to linear programming, some researchers suspected immediately that the algorithm was related to a certain classical method in nonlinear optimization: the barrier method. The method adds a penalty term for feasible points approaching the boundary of the feasible region, that is, the boundary of the non-negative orthant in (LP). For a given $\mu > 0$, which is called the barrier or penalty parameter, the method solves

$$\begin{array}{lll} (BLP(\mu)) & \text{minimize} & c^Tx - \mu\sum_{j=1}^n \ln x_j \\ & \text{subject to} & Ax = b, \end{array}$$

where the non-negative constraints are unnecessary since its logarithmic barrier objective function tends to infinity as one of the components of x tends to 0. In general, the barrier method solves the problem to obtain an approximate point of the minimizer $x^*(\mu)$ of $BLP(\mu)$. Gradually reducing μ, one can generate a sequence of approximate minimizers, leading to an solution of (LP). Note that the minimizer $x^*(\mu)$ satisfies the following relation: there exist (y, s) such that

$$\begin{pmatrix} Ax & = & b \\ A^T y + s & = & c \\ XSe & = & \mu e \\ x \geq 0 & & s \geq 0 \end{pmatrix}, \tag{1}$$

where $S = \mathrm{diag}(s)$. Also note that if we apply the barrier function method for the dual, i.e., solve

$$\begin{array}{lll} (BLD(\mu)) & \text{maximize} & b^T y + \mu \sum_{j=1}^{n} \ln s_j \\ & \text{subject to} & A^T y + s = c \\ & & s \geq 0, \end{array}$$

then, the minimizer $(y^*(\mu), s^*(\mu))$ of $BLD(\mu)$ also satisfies the relation (1). Thus, the solution (x, y, s) of (1) is a minimizer pair for both $BLP(\mu)$ and $BLD(\mu)$.

The continuous path of the minimizers $x^*(\mu)$, $0 < \mu < \infty$, is called the **central path**. This path has been studied by Bayer and Lagarias [6], Megiddo [27], and Sonnevend [39]. Surprisingly again, this path was analyzed by McLinden in 1980 [26]. Interestingly, the continuous version of Karmarkar's algorithm exactly follows this central path for the problem being solved. This fact suggested that a homotopy, or path-following algorithm, could be designed with the explicit intent of moving in a neighborhood of the central path.

The first complexity result for such an algorithm was given by Renegar in 1986 [37]. There are many variations on the path-following theme, but the essential idea can be illustrated as follows. The algorithm is initiated with a value μ^0 and an interior point x^0 that is close to the minimizer $x^*(\mu^0)$. The value of μ^0 then is reduced to μ^1, and a single Newton step, similar to the step of Karmarkar's algorithm, is taken to generate x^1. The complexity analysis is based on showing that if μ is not reduced too quickly, then x^1 is close to the minimizer $x^*(\mu^1)$, so that the process can be continued. The fact is that μ can be reduced at a rate of $(1 - constant/\sqrt{n})$, which results in an $O(\sqrt{n}L)$-iteration algorithm. Not surprisingly, it was soon shown that the partial rank-one updating could be applied to the path-following framework, leading to the development of $O(n^3 L)$ algorithms for linear programming in early 1987 (e.g., see Gonzaga [14]).

6 The primal-dual algorithm

All of the original path-following algorithms work either in the primal form or in the dual form. It is natural to develop an algorithm to simultaneously work in both primal and dual, since the optimizers of the $BLP(\mu)$ and $BLD(\mu)$ for a given μ share the same system of equations. Such an algorithm first was developed by Kojima et al. [20] and later by Kojima et al. [21] and Monteiro and Adler [32].

Let an interior point x^k for the primal and an interior point (y^k, s^k) for the dual be in a neighborhood of the central path, defined as

$$\mathcal{N}(\beta) = \left\{ (x, s) \in \mathcal{F}^0 : \|Xs - \mu e\| \leq (1-\beta)\mu \quad \text{where} \quad \mu = \frac{x^T s}{n} \right\}$$

for a constant $\beta \in (0,1)$, where $\|\cdot\|$ without subscript designates the Euclidian norm, e denotes the vector of ones and $X = \mathrm{diag}(x)$ denotes the diagonal matrix with diagonal entries equal to the elements of x.

Kojima et al. generate directions by solving a system of linear equations:

$$\begin{array}{rrrcl} & S^k d_x & +X^k d_s & = & \gamma \mu^k e - X^k s^k \\ & A d_x & & = & 0 \\ -A^T d_y & & -d_s & = & 0. \end{array} \tag{2}$$

where

$$\gamma = 1 - constant/\sqrt{n}.$$

Then, they update

$$x^{k+1} = x^k + d_x, \quad y^{k+1} = y^k + d_y, \quad s^{k+1} = s^k + d_s = c - A^T y^{k+1}.$$

They have shown that $(x^{k+1}, s^{k+1}) \in \mathcal{N}(\beta)$ and

$$(x^{k+1})^T s^{k+1} \leq \gamma (x^k)^T s^k = (1 - constant/\sqrt{n})(x^k)^T s^k.$$

Thus, the algorithm needs at most $O(\sqrt{n}L)$ iterations to reach

$$\frac{(x^k)^T s^k}{(x^0)^T s^0} \leq 2^{-L}.$$

In general, each iteration of a path-following method solves a system of linear equations where the coefficient matrix has the normal form ADA^T and D is an n-dimensional diagonal scaling matrix. It can be seen that

$$D = (X^k)^2, \quad \text{for the primal algorithm,}$$

$$D = (S^k)^{-2}, \quad \text{for the dual algorithm,}$$

and

$$D = (X^k)(S^k)^{-1}, \quad \text{for the primal-dual algorithm.}$$

It seems the scaling matrix in the primal-dual algorithm is the geometric mean of those in the primal and dual algorithms.

In addition to its complexity advantage, the path-following framework is well suited for extension to convex quadratic programming (QP) and monotone linear complementarity problems (LCP). Unfortunately, however, it is quite difficult to design an adaptive path-following algorithm that can take long steps without jeopardizing the $O(\sqrt{n}L)$-iteration complexity bound. In particular, convergence of the algorithm is established by arguing that the iterates remain close to the central path, which will force the algorithm to take short steps. Such algorithms are not practical. (There are several path-following algorithm that can take long steps, but their theoretical iteration complexity is $O(nL)$ or higher, see Den Hertog and Roos [9] and Gonzaga [16], and the references therein.)

7 The affine-scaling potential reduction algorithm

By mid-1987, the development of an adaptive (like Karmarkar's algorithm based on potential reduction), simple (like the affine scaling algorithm to avoid projective transformation), and $O(n^3L)$-operation or $O(\sqrt{n}L)$-iteration (as possessed by the path-following framework) algorithm was a major issue among some researchers concerned with interior-point algorithms for linear programming.

A number of results contributed to the solution of the problem. For example, it was shown that all interior-point algorithms were using combination of two directions—the affine scaling direction and a centering direction that moved iterates toward the central path. In fact, the affine scaling algorithm could be turned into a polynomial algorithm, with the same complexity as Karmarkar's algorithm, by adding the centering direction, see Gonzaga [15].

Tanabe [40] and Todd and Ye [44] introduced a primal-dual potential function for linear programming:

$$\psi(x, s) := \rho \ln(x^T s) - \sum_{j=1}^{n} \ln(x_j s_j).$$

where $\rho > n$. Todd et al. described a centered projective algorithm that operated in primal-dual space using the average scaling $(X^k)(S^k)^{-1}$. Using $\rho = n + \sqrt{n}$, they proved that each iteration of the algorithm reduced the potential function by a constant, which led to an $O(\sqrt{n}L)$ algorithm. Unfortunately, however, a constant reduction in ψ could only be guaranteed when iterates (x^k, s^k) lie in $\mathcal{N}(\beta)$.

This restriction was removed later. In the summer of 1988, Ye [51] presented an affine scaling potential reduction algorithm with $O(\sqrt{n}L)$-iteration complexity. The algorithm works directly in the original form (LP) and uses a simple rescaling as in the affine scaling algorithm; it is based on reducing the primal-dual potential function similar to Karmarkar's algorithm; and it possesses $O(\sqrt{n}L)$-iteration complexity as does the path-following algorithm. The algorithm does not need the iterates to remain in $\mathcal{N}(\beta)$, i.e., close to the central path, and, therefore, a line search can be used without jeopardizing the theoretical complexity bound (see Anstreicher and Bosch [3] and Freund [11]).

Finally, Kojima et al. [22] proposed an elegant primal-dual potential reduction algorithm that updates the primal and dual simultaneously as in their primal-dual path-following algorithm. The iterate direction is identical to that generated in the primal-dual path-following algorithm, but simply ignoring the use of $\mathcal{N}(\beta)$. Again, each iteration of their algorithm decreases ψ by a constant for $\rho = n + \sqrt{n}$. The primal-dual potential reduction algorithm is very close to what is implemented in practice with one exception: in practice the cost of line search seems much higher than that of a simple ratio test to determine step sizes for linear programming.

8 The infeasible-interior-point algorithm

At the beginning of the 1990s, there were several remaining issues concerning interior-point algorithms for linear programming. First, almost all interior-point algorithms solve the linear program under the regularity assumption that $\mathcal{F}^0 \neq \emptyset$ or equivalently that the optimal solution set for both (LP) and (LD) is bounded. Since no prior knowledge is usually available on the status of $\mathcal{F}^0$, one has to explicitly bound the feasible region of the linear program. This bound has to be set at 2^L in the worst case, where L is the data length of the linear program, assuming A, b, and c are integer. Obviously, this bounding is impossible to implement in practical computation. (If A, b and c are real number data, then no polynomially computable bound for M is known.)

Second, most interior-point algorithms have to start at an interior feasible point. The complexity of obtaining such an initial point is the same as that of solving the linear program itself. One approach to resolve this difficulty is the classical big-M method: add an artificial variable for the residual (error) in the equality system and set a huge number M as its coefficient in the objective function, hoping this variable has zero value at every optimal solution. In theory, M should equal 2^L, which is also too big to implement in practice.

Finally, a complete linear programming algorithm should accomplish two tasks: 1) affirmatively detect the infeasibility or unboundedness status of the linear program, then 2) generate an optimal solution if the problem is neither infeasible nor unbounded.

At the beginning of 1990, there were several encouraging primal-dual implementations that performed very well in practice (Lustig et al. [24][25] and Mehrotra [28]). They were called primal-dual infeasible-interior-point algorithms, which relaxed the theoretical condition that x^k and (y^k, s^k) satisfy the equality constraints in the primal and dual, respectively. They generated direction from

$$\begin{array}{rrrcl} & S^k d_x & +X^k d_s & = & \gamma\mu^k e - X^k s^k \\ & A d_x & & = & b - Ax^k \\ -A^T d_y & & -d_s & = & -c + (A^T y^k + s^k). \end{array} \tag{3}$$

However, no theoretical convergence result was established for this approach until Kojima et al. [23] proved that the algorithm converges if the initial positive (infeasible) point is big enough and the linear program is feasible and bounded. Later, an $O(n^2 L)$ complexity could be established for this approach if the linear program possesses an optimal solution and if the initial point is set to $(x^0, s^0) = M(e, e)$, where M is 2^L in the worst case (see Zhang [55]). (Subsequently, several researchers developed an $O(nL)$ complexity bound under the same conditions, see Mizuno et al. [30], Potra [36], and Wright [49].) Thus, the big M difficulty remained in these polynomial infeasible interior-point algorithms.

In the summer of 1992, Ye et al. [54] presented an $O(\sqrt{n}L)$-iteration homogeneous and self-dual algorithm to resolve these issues. The algorithm possesses the following features:

- It solves the linear programming problem without any regularity assumption concerning the existence of optimal, feasible, or interior feasible solutions.
- It can start at any positive primal-dual pair, feasible or infeasible, near the central ray of the positive orthant (cone), and it does not use any big M penalty parameter or lower bound.
- Each iteration solves a system of linear equations whose dimension is almost the same as that solved in the standard (primal-dual) interior-point algorithms.
- If the linear program has a solution, the algorithm generates a sequence that approaches feasibility and optimality simultaneously; if the problem is infeasible or unbounded, the algorithm will correctly detect infeasibility for at least one of the primal and dual problems.

Later, Xu et al. [50] have further simplified the algorithm. Their work deals with the homogeneous and self-dual linear feasibility model of Goldman and Tucker [13]:

$$\begin{array}{lrrrl} (HLF) & & Ax & -b\tau & = 0, \\ & -A^T y & & +c\tau & \geq 0, \\ & b^T y & -c^T x & & \geq 0, \\ & y \text{ free} & x \geq 0 & \tau \geq 0. & \end{array} \tag{4}$$

Denote by s the slack vector for the second (inequality) constraint and by κ the slack scalar for the third (inequality) constraint. Then, the problem is to find a strictly complementary point such that

$$x^T s = 0, \;\; \tau\kappa = 0, \quad \text{and} \quad x + s > 0, \;\; \tau + \kappa > 0.$$

Given $(y^k, \; x^k > 0, \tau^k > 0, s^k > 0, \kappa^k > 0)$, an (infeasible) interior point for (HLF), the kth iteration of the HSD algorithm solves the following system of linear equations for the direction $(d_y, d_x, d_\tau, d_s, d_\kappa)$

$$\begin{array}{rrrrrcr} & A\ d_x & -b\ d_\tau & & & = & \eta\ r_P^k, \\ -A^T\ d_y & & +c\ d_\tau & -d_s & & = & -\eta\ r_D^k, \\ b^T\ d_y & -c^T\ d_x & & & -d_\kappa & = & \eta\ r_G^k, \end{array} \tag{5}$$

and

$$\begin{array}{rcl} X^k d_s + S^k d_x & = & \gamma\mu^k e - X^k s^k, \\ \tau^k d_\kappa + \kappa^k d_\tau & = & \gamma\mu^k - \tau^k\kappa^k, \end{array} \tag{6}$$

where non-negative η and γ are two parameters, and

$$\mu^k = ((x^k)^T s^k + \tau^k \kappa^k)/(n+1), \tag{7}$$

$$r_P^k = b\tau^k - Ax^k, \quad r_D^k = c\tau^k - A^T y^k - s^k, \quad r_G^k = c^T x^k - b^T y^k + \kappa^k. \tag{8}$$

Here r_P^k, r_D^k, and r_G^k represent the infeasibility residuals of the kth iteration for primal, dual, and primal-dual "gap", respectively. Comparing to systems (2) and (3), we see that the dimension of system (5) is increased only by 1.

9 Further results

Besides the algorithmic progress of linear programming described in the previous sections, many other significant results have been obtained during the last few years. We briefly describe some of them below.

9.1 Local convergence

Interior-point codes suggest that the number of iterations necessary to find an "exact" optimal solution grows much more slowly with the dimension n. There is thus a gap between observed performance and theoretical bounds. One attempt to narrow this gap is to analyze the local or asymptotic convergence behavior of interior-point algorithms.

Local convergence speed of iterates is a popular criterion for measuring efficiency of algorithms. Consider the sequence of feasible points $\{(x^k, s^k)\}$ generated by an interior-point algorithm such that the (complementarity or duality) gap $(x^k)^T s^k \to 0$. Then we say that this gap sequence converges superlinearly to zero if

$$\lim_{k\to\infty} \frac{(x^{k+1})^T s^{k+1}}{(x^k)^T s^k} = 0,$$

and quadratically to zero if

$$\limsup_{k\to\infty} \frac{(x^{k+1})^T s^{k+1}}{((x^k)^T s^k)^2} < +\infty.$$

Recently several results were developed for the local convergence of polynomial interior-point algorithms. It was shown that several interior-point algorithms for linear programming (LP) exhibited superlinear convergence of the primal-dual gap to zero under some combination of various assumptions.

The assumption-free quadratic convergence result for linear programming was first established by Mehrotra [29] and by Ye et al. [53]. Their analysis is based on the predictor-corrector algorithm of Mizuno et al. [31]. Each iteration of the algorithm solves two systems of linear equations or two least squares problems—one in the predictor step and one in the corrector step. Given the iterate $(x^k, s^k) \in \mathcal{N}(\beta/2)$, the predictor step generates directions by solving system (2) with $\gamma = 0$, and creates the iterate $(x^{k+1}, s^{k+1}) \in \mathcal{N}(\beta)$. The next iteration is the corrector step, which generates directions by solving system (2) with $\gamma = 1$ and obtains the next iterate $(x^{k+2}, s^{k+2}) \in \mathcal{N}(\beta/2)$. There is no duality gap reduction in corrector steps, but each predictor step yields quadratic reduction in the duality gap asymptotically.

9.2 Probabilistic analysis

There is a series of recent attempts regarding the average complexity of interior point algorithms for linear programming. The idea of average case analysis is to obtain rigorous probabilistic bounds on the number of iterations required by an algorithm to reach some termination criterion, when the algorithm is applied to a random instance of a problem drawn from some probability distribution. In the case of the simplex method for linear programming, average case analysis has provided some theoretical justification for the observed practical efficiency of the method, despite its exponential worst case bound. As we mentioned earlier, although many interior point algorithms devised in the last several years are polynomial time methods, in practice they too generally perform much better than worst case bounds would indicate.

Some of these average case analyses of interior-point algorithms are not rigorous as in the case of the simplex method; instead of assuming a random problem held fixed throughout the iterations, they make a probabilistic assumption about the data at a particular iteration, analyze the performance at that iteration, and hence make heuristic estimates of the "typical" behaviour of interior-point algorithms. Nevertheless, these studies indicate behaviour closer to what is observed in practice; see Mizuno et al. [31] and the references therein.

The first rigorous average case analysis for an interior point method was provided by Ye [52]. He proved a "high probability" bound of $O(\sqrt{n} \ln n)$ iterations for termination, using a variety of algorithms applied to several different probabilistic models. Here n is the number of variables in a standard form problem, and "high probability" means that the probability that the bound holds goes to one, as $n \to \infty$.

Recently, Anstreicher et al. [4] considered the self–dual algorithm of Ye et al. [54], with a modified finite termination scheme, applied to random linear programs generated according to a model of Todd. This model has degenerate optimal solutions, and possesses no feasible starting point. Their main result is that the expected number of iterations before termination with an exact optimal solution is bounded above by $O(\sqrt{n} \ln n)$. In terms of generality of the probabilistic model, computational effort per iteration, and overall bound on the expected number of iterations, their analysis provides the best probabilistic result to date for an interior point algorithm.

9.3 Complexity lower bounds

There have also been efforts to look for **lower** bounds on the number of iterations required for various interior-point algorithms; see Todd [42] and the references therein. One important result is due to Todd who obtains a bound of at least $n^{1/3}$ iterations to achieve a constant factor decrease in the duality gap. The algorithm he studies is the primal-dual affine-scaling algorithm, which is close to methods used in practical implementations. He allows almost any reasonable step size rule, such as going 99.5% of the way to the boundary of the feasible region, again as used in practical codes; such step size rules definitely do not lead to iterates lying close to the central path. Furthermore, he has showed that this lower bound holds for many polynomial variants of primal-dual interior-point methods, which restrict iterates to certain neighborhoods of the central path.

9.4 Strongly polynomial algorithm

Many network flow problems can be formulated as linear programs, where the size of matrix A is polynomial but the size of vectors c and b may be exponential in m and n. Thus, the development of linear programming algorithms whose complexity depends only on A is of great interest.

The first such an algorithm was developed by Tardos [41], under the assumption that the data of A, b and c are rational numbers. Her algorithm constructs a sequence of up to n linear programs, then it uses any polynomial algorithm to solve each of these linear programs. The question is whether there exists a direct algorithm whose running time depends only on A where the data of A, b and c can be real numbers.

This question has been answered by the recent work of Vavasis and Ye [48]. They have proposed a "layered-step" interior-point algorithm. The algorithm follows the same central path, either with the classical iteration step or with a new type of step called a "layered least squares" step. The algorithm returns the exact optimal solution, under the real number complexity model, after a finite number of iterations—in particular, after $O(n^{3.5}c(A))$ iterations, where $c(A)$ is a function of A. For network flow problems $c(A)$ is a polynomial in m and n. Thus, the result establishes a direct strongly polynomial algorithm for solving linear programs whose A contains small-integer entries.

9.5 Column generation

As various interior point algorithms have been developed for linear programming, they have a common drawback: they need the complete knowledge of the linear system, and their computational complexity depends on all of the inequalities (in the dual), although some of them are useless for defining an optimal solution.

On the other hand, the decomposition algorithm of Dantzig and Wolfe and the ellipsoid method [18] do not need knowledge of the full system in advance—they allow column generation: if a constraint (or column) is needed during their course, it is then called and added to the process. This column generation technique permits a great deal of flexibility for solving linear programs, such as semi-infinite programming, convex nonlinear programming and combinatorial optimization problems, in which the number of inequalities is very large or some constraints are not explicitly known.

Based on the theory of "centers" of polyhedra, several interior-point column-generation or cutting-plane algorithm have been developed for solving linear inequality system with an infinite number of constraints. These algorithms start with a simple containing polytope (which contains the solution set) and its known "center." In each iteration, they select an inequality violated at the "center," use it to cut the polytope, and then find the new (approximate) "center" for the shrunken and containing polytope. The "volume" of the polytope is reduced at a constant ratio in each iteration.

Several centers, based on different volumes in measuring a polytope, have been considered. One algorithm uses an analytic center. It has recently been proved by Nesterov [33] and Goffin et al. [12] that the algorithm is a polynomial approximation algorithm, whose complexity depends only on m and the size of the solution set. Furthermore, using an additional technique to eliminating existing inequalities of the containing polytope, Atkinson [5] has proved that the algorithm is actually a polynomial algorithm.

Another algorithm uses a volumetric center, due to Vaidya [46]. His algorithm possesses the same arithmetic operation complexity as the ellipsoid algorithm, where it also depends only on m and the size of the solution set. If a modification of the algorithm is applied to finite-dimensional (LP), the iteration complexity is $O(n^{1/4}m^{1/2}L)$, which is better than $O(n^{1/2}L)$ if $m \leq O(n^{1/2})$. In a further effort, this iteration bound is reduced to $O(n^{1/4}m^{1/4}L)$ by Vaidya and Atkinson [47].

9.6 Nonlinear convex optimization

Although almost all results developed for LP have been extended to convex QP, a new theory is required for solving more general convex programs beyond LP and QP. Such a breakthrough in this direction was first made by Nesterov and Nemirovskii. The details of this development can be found in their recent book [34].

Their theory applies to solving a general optimization problem

$$\begin{array}{lll} (GLP) & \text{minimize} & < c, x > \\ & \text{subject to} & Ax = b, \ x \in \mathcal{K}, \end{array}$$

where $<, >$ is an inner product and $\mathcal{K}$ is a general convex cone in a finite dimensional space. For example, $\mathcal{K}$ could be the set of all $n \times n$ positive semidefinite matrices. There is also a dual problem related to (GLP):

$$\begin{array}{lll} (GLD) & \text{minimize} & < b, y > \\ & \text{subject to} & A^*y + s = c, \ s \in \mathcal{K}^*, \end{array}$$

where $\mathcal{K}^*$ is the dual cone of $\mathcal{K}$,

$$\mathcal{K}^* = \{s : < x, s > \geq 0, \text{ for all } x \in \mathcal{K}\}.$$

Note that the duality relations between (LP) and (LD) may not hold here.

Nesterov and Nemirovskii described a universal barrier function as the volume of the polar set of $\mathcal{K}$. Thus, many complexity and convergence results for linear programming can be extended to solving the nonlinear convex problems. In particular, Alizadeh analyzed in detail the problem in which $\mathcal{K}$ is the set of $n \times n$ positive definite matrices [2].

Renegar [38] developed a perturbation theory for solving this problem, which he called the generalized linear programming problem. He examined a few relations between solution characteristics of the problem and the amount by which the problem must be perturbed to obtain either primal or dual infeasibility. He related the computational complexity and the sensitivity of the problem to data perturbations.

More recently, Nesterov and Todd [35] developed a symmetric primal-dual algorithm, similar to the one described in Section 6, to solve (GLP) when the cone is "self-scaled". Güler [17] gave a new method to construct barrier functions for homogeneous convex cones. He proved that the universal barrier function is the logarithmic function of the characteristic function of cones studied in the 1950s, a theory in several established branches of mathematics, such as Lie groups, differential geometry, symmetric spaces, Jordan algebras, etc.

10 Conclusion

During the last ten years, implementations based on the simplex method have been also improved dramatically. Today's computer codes of both the simplex and interior-point methods are solving linear programs with more than a million variables. Such problems were thought intractable just a few years ago. Moreover, problems which used to need hours to solve are now solved in minutes. It appears that the development of interior-point methods has created a healthy competition among linear programming algorithms. It is reasonable to believe that a brighter future lies ahead in the research and application area of linear programming.

References

[1] I. Adler, N. Karmarkar, M.G.C Resende, and G. Veiga, "An implementation of Karmarkar's algorithm for linear programming," *Mathematical Programming* 44 (1989) 297-335.

[2] F. Alizadeh, "Combinatorial Optimization with Interior-Point Methods and Semi-Definite Matrices," Ph.D. Thesis, Computer Science Department, University of Minnesota, 1991.

[3] K. M. Anstreicher and R. A. Bosch, "Long steps in an $O(n^3L)$ algorithm for linear programming," *Mathematical Programming* 54 (1992) 251-267.

[4] K. M. Anstreicher, J. Ji, F. Potra, and Y. Ye, "Average performance of a self-dual interior-point algorithm for linear programming," in P. Pardalos eds., *Complexity in Numerical Optimization* (World Scientific, New Jersey, 1993) pp. 1-15.

[5] D. S. Atkinson, "A cutting plane algorithm that uses analytic centers," Ph. D. Thesis, Coordinated Science Lab., University of Illinois, 1993.

[6] D. A. Bayer and J. C. Lagarias, "The nonlinear geometry of linear programming: I. Affine and projective scaling trajectories, II. Legendre transform coordinates and central trajectories," *Transactions of the American Mathematical Society* 314 (1989) 499-581.

[7] R. G. Bland, D. Goldfarb, and M. J. Todd, "The ellipsoid method: a survey," *Operations Research* 29 (1981) 1039-1091.

[8] V. Chvátal, *Linear Programming* (Freeman, New York, 1983).

[9] D. Den Hertog and C. Roos, "A survey of search directions in interior-point methods for linear programming," *Mathematical Programming* 52 (1991) 481-510.

[10] I. I. Dikin, "Iterative solution of problems of linear and quadratic programming," *Soviet Mathematics Doklady* 8 (1967) 674-675.

[11] R. M Freund, "Polynomial-time algorithms for linear programming based only on primal scaling and projected gradients of a potential function," *Mathematical Programming* 51 (1991) 203-222.

[12] J.-L. Goffin, Z.-Q. Luo, and Y. Ye, "Further complexity analysis of a primal-dual column generation algorithm for convex or quasiconvex feasibility problems," manuscript, Faculty of Management, McGill University (Montreal, Canada, 1993), to appear in *SIAM J. Optimization*

[13] A. J. Goldman and A. W. Tucker, "Polyhedral convex cones," in: H. W. Kuhn and A. W. Tucker eds., *Linear Inequalities and Related Systems* (Princeton University Press, Princeton, NJ, 1956) pp. 19-40.

[14] C. C. Gonzaga, "An algorithm for solving linear programming programs in $O(n^3 L)$ operations," in: N. Megiddo, ed., *Progress in Mathematical Programming, Interior Point and Related Methods* (Springer-Verlag, New York, 1988) pp. 1-28.

[15] C. C. Gonzaga, "Polynomial affine algorithms for linear programming," *Mathematical Programming* 49 (1991) 7-21.

[16] C. C. Gonzaga, "Path-following methods for linear programming," *SIAM Review* 34 (1992) 167-224.

[17] O. Güler, "Barrier functions in interior-point methods," manuscript, Department of Mathematics and Statistics, University of Maryland at Baltimore County (Baltimore, MA, 1994).

[18] L. G. Khachiyan, "A polynomial algorithm in linear programming," *Soviet Mathematics Doklady* 20 (1979) 191-194.

[19] N. Karmarkar, "A new polynomial-time algorithm for linear programming," *Combinatoria* 4 (1984) 373-395.

[20] M. Kojima, S. Mizuno, and A. Yoshise, "A primal-dual interior point algorithm for linear programming," in: N. Megiddo, ed., *Progress in Mathematical Programming, Interior Point and Related Methods* (Springer-Verlag, New York, 1989) pp. 29-47.

[21] M. Kojima, S. Mizuno, and A. Yoshise, "A polynomial-time algorithm for a class of linear complementarity problems," *Mathematical Programming* 44 (1989) 1-26.

[22] M. Kojima, S. Mizuno, and A. Yoshise, "An $O(\sqrt{n}L)$ iteration potential reduction algorithm for linear complementarity problems," *Mathematical Programming* 50 (1991) 331-342.

[23] M. Kojima, N. Megiddo, and S. Mizuno, "A primal-dual infeasible-interior-point algorithm for linear programming," *Mathematical Programming* 61 (1993) 263-280.

[24] I. J. Lustig, "Feasibility issues in a primal-dual interior-point method for linear programming," *Mathematical Programming* 49 (1990/91) 145-162.

[25] I. J. Lustig, R. E. Marsten, and D. F. Shanno, "Computational experience with a primal-dual interior point method for linear programming" *Linear Algebra and Its Applications* 152 (1991) 191-222.

[26] L. McLinden, "An analogue of Moreau's proximation theorem, with applications to the nonlinear complementarity problem," *Pacific Journal of Mathematics* 88 (1980) 101-161.

[27] N. Megiddo, "Pathways to the optimal set in linear programming," in: N. Megiddo, ed., *Progress in Mathematical Programming, Interior Point and Related Methods* (Springer-Verlag, New York, 1988) 131-158.

[28] S. Mehrotra "On the implementation of a (primal-dual) interior point method," *SIAM Journal on Optimization* 2 (1992) 575-601.

[29] S. Mehrotra, "Quadratic convergence in a primal-dual method," *Mathematics of Operations Research* 18 (1993) 741-751.

[30] S. Mizuno, F. Jarre, and J. Stoer, "A unified approach to infeasible-interior-point algorithms via geometrical linear complementarity problems," Preprint 213, Department of Mathematics, Wurzburg University (Wurzburg, Germany, 1994).

[31] S. Mizuno, M. J. Todd, and Y. Ye, "On adaptive-step primal-dual interior-point algorithms for linear programming," *Mathematics of Operations Research* 18 (1993) 964-981.

[32] R. C. Monteiro and I. Adler, "Interior path following primal-dual algorithms. Part I: Linear programming," *Mathematical Programming* 44 (1989) 27-42.

[33] Yu. Nesterov, "Cutting plane algorithm from analytic centers: efficiency estimates," manuscript, Central Economic and Mathematical Institute (Moscow, 1992).

[34] Yu. Nesterov and A. Nemirovskii, "Interior-Point Polynomial Algorithms in Convex Programming," SIAM (Philadelphia, PA, 1993).

[35] Yu. Nesterov and M. J. Todd, "Self-scaled cones and interior-point methods in nonlinear programming," Technical Report 1091, School of ORIE, Cornell University (Ithaca, NY, 1994).

[36] F. A. Potra, "On a predictor and corrector method for solving linear programs from infeasible starting points," TR 34, Department of Mathematics, The University of Iowa (Iowa City, IA, 1992).

[37] J. Renegar, "A polynomial-time algorithm based on Newton's method for linear programming," *Mathematical Programming* 40 (1988) 59-94.

[38] J. Renegar, "Some perturbation theory for linear programming," manuscript, School of ORIE, Cornell University (Ithaca, NY, 1992).

[39] G. Sonnevend, "An analytical center for polyhedrons and new classes of global algorithms for linear (smooth, convex) programming," in: *Lecture Notes in Control and Information Sciences* 84 (Springer, New York, 1985) pp. 866-876.

[40] K. Tanabe, "Centered Newton method for mathematical programming," in M. Iri and K. Yajima, eds., *System Modeling and Optimization* (Springer-Verlag, New York, 1988) pp. 197-206.

[41] É. Tardos, "A strongly polynomial algorithm to solve combinatorial linear programs," *Operations Research* 34 (1986) 250-256.

[42] M. J. Todd. "A lower bound on the number of iterations of primal–dual interior–point methods for linear programming," In G.A. Watson and D.F. Griffiths, editors, *Numerical Analysis 1993*, Volume 303 of *Pitman Research Notes in Mathematics* 303 (Longman Press, Burnt Hill, UK, 1994) pp. 237-259.

[43] M. J. Todd and B. P. Burrell, "An extension of Karmarkar's algorithm for linear programming using dual variables," *Algorithmica* 1 (1986) 409-424.

[44] M. J. Todd and Y. Ye, "A centered projective algorithm for linear programming," *Mathematics of Operations Research* 15 (1990) 508-529.

[45] T. Tsuchiya and M. Muramatsu, "Global convergence of a long-step affine scaling algorithm for degenerate linear programming problems," Research Memo 423, The Institute of Statistical Mathematics (Tokyo, Japan, 1992).

[46] P. M Vaidya, "A new algorithm for minimizing convex functions over convex sets," manuscript (1989), to appear in *Mathematical Programming*

[47] P. M Vaidya and D. S. Atkinson, "A technique for bounding the number of iterations in path-following algorithms," in P. Pardalos eds., *Complexity in Numerical Optimization* (World Scientific, New Jersey, 1993) pp. 462-489.

[48] S. A. Vavasis and Y. Ye, "An accelerated interior-point method whose running time depends only on A," TR 93-1391, Department of Computer Science; Cornell University (Ithaca, New York, 1993).

[49] S. Wright, "A path-following infeasible-interior-point algorithm for linear complementarity problem," MCS-P334-1192, Mathematics and Computer Science Division, Argonne National Laboratory (Chicago, IL, 1992).

[50] X. Xu, P. F. Hung, and Y. Ye, "A simplified homogeneous and self-dual linear programming algorithm and its implementation," Working Paper, College of Business Administration, The University of Iowa (Iowa City, IA, 1993).

[51] Y. Ye, "An $O(n^3L)$ potential reduction algorithm for linear programming," *Mathematical Programming* 50 (1991) 239-258.

[52] Y. Ye, "Toward probabilistic analysis of interior-point algorithms for linear programming," *Mathematics of Operations Research* 19 (1994) 38-52.

[53] Y. Ye, O. Güler, R. A. Tapia, and Y. Zhang, "A quadratically convergent $O(\sqrt{n}L)$-iteration algorithm for linear programming," *Mathematical Programming* 59 (1993) 151-162.

[54] Y. Ye, M. J. Todd, and S. Mizuno, "An $O(\sqrt{n}L)$-iteration homogeneous and self-dual linear programming algorithm," *Mathematics of Operations Research* 19 (1994) 53-67.

[55] Y. Zhang, "On the convergence of a class of infeasible interior-point methods for the horizontal linear complementarity problem," *SIAM J. on Optimization* 4 (1994) 208-227.