Interior-Point Algorithms: First Year Progress Report on NSF Grant DDM-8922636

Yinyu Ye

October 19, 1995

1. Introduction

This document is the first year progress report on the optimization projects funded by NSF Grant DDM-8922636. The projects principally include the interior-point algorithms for linear programming (LP), quadratic programming (QP), linear complementarity problem (LCP), and nonlinear programming (NP). The anticipated discoveries and advances resulting from the project include the following.

1.1. Column Generation

While various interior point algorithms have been developed for LP (e.g., [12][15][17][23]), they all share a common drawback: they require the complete knowledge of the full constraint system. In contrast, the (revised) simplex and ellipsoid methods do not require the complete knowledge of the system in advance. These methods allow for column generation—the addition of constraints to the system only when needed. This technique permits a great deal of flexibility for solving optimization problems in which the number of constraints is either very large or not explicitly known. One main topic of the project is to develop interior-point algorithms that allow column generation and are implementable with practical efficiency. This will advance the practical solvability for a large class of optimization problems, such as semi-infinite programming, convex nonlinear programming, and combinatorial optimization.

1.2. Algorithm Termination

Unlike the simplex method for LP which terminates in *finite* time [5], interior-point algorithms are continuous optimization algorithms that generate a solution sequence converging to the optimal solution facet in *infinite* time. If the data for the LP is integral or rational, an argument is made that after a worst-case time bound the solution can be rounded from the latest approximate solution. However, several questions naturally arise. First, if the data is real, how do we achieve finite convergence (i.e., find a solution in finite time)? Second, regardless of the data's status, can we utilize a practical, cost-effective test to identify an solution so that the algorithm can be terminated before the worse-case time bound? Here, the solution may be given as a mathematical closed form using basic arithmetic operations, such as solving a consistent system of linear equations or a least-squares problem. The second goal of the project is to find such a test.

1.3. Convex Nonlinear Optimization

Most of the computational work on interior point methods has concentrated on LP problems. Recently several new insights have been found which make it possible to explore nonlinear programming problems using interior-point algorithms. The third topic of the project is to improve and implement interior point algorithms for convex NP. This will enhance the computational speed for solving many large-scale engineering and economic problems, some of which may have never been solved before.

1.4. Nonconvex Optimization

Many nonconvex optimization problems, such as nonconvex quadratic programming problems, are NP-Hard problems. These problems have wide applications in planning, manufacture, engineering, management, and computer science, and are among the most challenging problems in optimization. The fourth topic of the project is to investigate the potential of interior-point algorithms for solving some nonconvex optimization problems.

Some progress has been made in each of these areas. The project seems going well in general. Most of the first-year goals have been met. In fact, we have already entered the implementation and application phase of the project. Ten working papers have directly resulted from the project: four of them have been accepted for publication in *Mathematical Programming* or *SIAM Journal on Optimization*, and five have been invited for presentation in sponsored workshops or conferences. Below, I provide some partial results, analyze anticipated difficulties, and summarize ongoing and planned research activities.

2. Project Progress

2.1. Progress in Column Generation

Several approaches for column generation or decomposition were proposed and developed. One such approach is summarized in my report [Y1]. The first and second tasks (using approximate centers and multiple cuts) in my original proposal are now completed. My column generation algorithm reduces the previous "best" complexity bound by a factor $\sqrt{q}$, where q is the number of columns (constraint) generated in general LP decomposition algorithms. Moreover, the result has been extended to general convex programming problems, see [2]. We are now planning to implement and test the algorithm.

Another decomposition approach was proposed by Dantzig and myself [Y23] which has the same convergence properties as their well known dual affine scaling algorithms but may require less computational effort. The method differs from Dikin's algorithm of dual affine form in that the ellipsoid chosen to generate the improving direction in dual space is constructed from only a subset of the active constraints (or columns). Let us briefly explain as follows.

Consider the linear program whose dual form is

$$\max \quad b^T y, \quad s.t. \quad c - A^T y \geq 0 \tag{2.1}$$

We denote the dual slack variables by $u = c - A^T y \geq 0$ where $A \in R^{m \times n}$. Each iteration k of the Dikin algorithm starts with an interior dual y^k, and solves an ellipsoid subproblem centered at y^k:

$$\max \quad b^T y, \quad s.t. \quad y \in \mathcal{E} = \{y \in R^m : \|D^{-1} A^T (y - y^k)\| \leq 1\} \tag{2.2}$$

where the Euclidean norm of a vector v is denoted by $\|v\|$ and $D = \mathrm{Diag}(u^k)$ denotes the diagonal matrix with diagonal u^k. Let $\Delta = (AD^{-2}A^T)^{-1}b$. Then, the improved iterate $y^{k+1} = y^k + \bar{\Delta} = y^k + (\Delta b)^{-1/2}\Delta$ results from solving the ellipsoid constrained subproblem (2.2). The computation of $(AD^{-2}A^T)^{-1}b$ involves *full* columns of A even though in practical problems most columns either have little or adverse effects on the shape and size of the ellipsoid and hence on the location of the optimal point $y = y^{k+1}$. This suggests that a good part of the computational work could be bypassed if one knew (or had a good guess about) which columns to temporarily drop from (2.2). Hence in practice an ellipsoid $\mathcal{E}$ based on fewer *active* columns may accelerate the convergence of the method, especially for problems with many columns.

The basic idea of the method is the following. At the start of each major iteration k, we are given an interior iterate y^k. A selection of m dual constraints is made using an "column-ordering" rule which chooses those constraints which show "the most active" of being tight in the optimal dual solution. An ellipsoid centered at y^k is then inscribed in convex region defined by only these active constraints and an improving direction $\bar{\Delta}$ is computed. Minor cycling within the major iteration is then started.

During the minor cycle, the constraints selected to define the ellipsoid centered at y^k are built up to include the constraints (whenever there is one) that block feasible movement from y^k to $y^k + \bar{\Delta}$. If some blocks, they are used to augment the set of active constraints and the ellipsoid is revised; a new improving direction $\bar{\Delta}$ is recomputed by means of an update, and the minor cycle repeated until none blocks movement from y^k to $y^k + \bar{\Delta}$. After the minor cycling ends, $y^{k+1} = y^k + \bar{\Delta}$ initiates the next major iteration. The major iterations stop when an optimum solution is reached, which will occur in a finite number of iterations.

A variant of [Y23] for linear programming with column generation has been developed and implemented by Kaliski, one of my students, on an IBM 3090-200S machine. Our preliminary computational results indicate that approximately $2m$ columns suffice to complete a major iteration. Thus, computation work of a major iteration involves only $O(m)$ columns instead of $O(n)$ columns. This difference is very significant when $n >> m$. Table 1 illustrates some computational results for solving large-scale transportation problems, where $n = O(m^2)$, n is the number of variables, Itrs. is the number of major iterations, and CPU is the total CPU times in minutes. The last column of the table is the ratio of the CPU times using full vs active columns.

Figure 1 in the appendix graphically illustrates the ratio of CPU times using full vs active columns. The efficiency of using column generation seems getting better and better as the problem's size increases.

2.2. Progress in Algorithm Termination

Ever since the rebirth of interior-point algorithm in 1984 ([15]), some efforts have been made to find interior-point algorithm stopping criteria. These efforts resulted in four basic approaches. The first of these is a theoretical effort to identify the optimal basis. If the LP problem is nondegenerate, a related practical test was also developed to show that the unique optimal basis can be identified

	Using Full Columns		Using Active Columns		CPU Time Rate
n	Itrs.	CPU	Itrs.	CPU	Full vs Active
2500	14	0.03	10	0.02	1.85
10000	21	0.18	19	0.09	2.11
22500	21	0.47	17	0.20	2.36
40000	29	1.21	31	0.55	2.22
62500	24	2.03	21	0.62	3.27
90000	30	3.78	30	1.21	3.11
122500	25	4.16	34	1.77	2.35
160000	30	6.91	44	3.31	2.09
202500	27	8.74	41	3.76	2.32
250000	27	13.30	39	4.81	2.77
302500	30	15.01	52	7.61	1.97
360000	60	51.96	55	10.46	4.97
422500	32	30.91	47	9.63	3.21
490000	36	42.59	51	11.99	3.55
562500	37	58.47	50	13.74	4.25
640000	68	126.46	54	19.06	6.63
722500	30	57.79	47	17.10	3.38
810000	34	61.95	48	19.75	3.14
902500	30	85.33	38	18.43	4.63
1000000	29	87.26	41	22.06	3.95
1000000	45	157.94	46	28.10	5.62
1000000	30	111.70	35	24.52	4.55

Table 1: Solving Transportation Problems with or without Column Generation.

before the worst-case time bound. The test seemed to work fine for nondegenerate LP problems but failed for degenerate LP problems. Devastatingly, most of real LP problems are degenerate! The difficulty arises simply because any degenerate LP problem has multiple optimal bases. The second approach is to resort to the simplex method. Difficulties also arise with this approach because the simplex method may still require many pivot steps for some "bad" problems. The third approach is to slightly randomize the data such that the LP problem is nondegenerate and its optimal basis is one of the optimal bases of the original LP problem. The question remaining is how and when to randomize the data during the iterative process, which will significantly affect the success of the effort. The fourth approach is Gay's test [7] to identify the optimal facet and to find a feasible solution on the facet.

Recently, we established a theoretical base for Gay's test when it is used within some polynomial interior-point algorithms. We also simplified his test to find an optimal solution on the optimal facet. These developments are based on a recent theoretical result of Güler and myself [Y17] which characterizes the convergence behavior of several popular interior-point algorithms. We showed that these algorithms for LP generate a solution sequence whose every limit point is a strict complementarity solution, or a (relative) interior point in the optimal facet. This result is useful because the optimal facet is unique for both primal and dual, and in finite time we can "drop" (orthogonally project) the latest interior solution onto the optimal facet and it will be guaranteed to hit the interior of the target. The result indicates that these interior-point algorithms are finite-convergence algorithms for LP with general data, see [Y15] for details.

For a unique index set $\sigma^* \subset \{1, 2, ..., n\}$, the optimal facet for the primal LP is

$$\Omega_p = \{x : Ax = b,\ x \geq 0, \quad x_j = 0,\ \text{for } j \in \sigma^*\},$$

and the one for the dual LP is

$$\Omega_d = \{y : A^T y + s = c, \quad s_j = 0,\ \text{for } j \in \sigma^*\}.$$

Each of these facets has a nonempty (relative) interior, meaning a solution with strictly positive inequality slacks exists in either one of these two facets. Given an interior solution $(x^k, y^k, s^k$ generated by any of these interior-point algorithms, define

$$\sigma^k = \{j : x_j^k \geq s_j^k\}.$$

Then, we have proved that for some sufficiently large but finite K,

$$\sigma^k = \sigma^*, \quad \text{for all} \quad k \geq K.$$

Furthermore, if the LP problem has integral data, then $\sigma^k = \sigma^*$ before $(x^k)^T s^k < 2^{-O(L)}$, i.e., before the traditional worst-case terminating condition is satisfied, where L is the input size of the data.

In practice, we certainly do not wait too long to try our luck at σ^k. In the following we develop a simple procedure to test if σ^k is optimal and if an optimal solution can be reached. For simplicity, let those columns in A corresponding to σ^k form matrix B and the rest form matrix N, and denote those corresponding variables by x_B and x_N, respectively. Then we solve

$$\min \quad \|x_B - x_B^k\|, \quad s.t. \quad Bx_B = b \tag{2.3}$$

and

$$\min \quad \|y - y^k\|, \quad s.t. \quad B^T y = c_B \tag{2.4}$$

The first problem is equivalent to projecting x_B^k into the hyperplane $\{x_B : Bx_B = b\}$, and the second is to project y^k into the hyperplane $\{y : B^T y = c_B\}$. Note we are assuming no rank condition on B. If these problems are feasible, then the optimal solutions are unique for both of them. These two problems can be solved as two related least-squares. The amount of work to solve (2.3) and (2.4) is of the same order as one iteration of interior-point algorithms. Furthermore, if the resulting solutions x_B^* and y^* satisfy

$$x_B^* > 0 \quad \text{and} \quad c_N - N^T y^* > 0,$$

then, obviously $x^* = (x_B^*, 0)$ and y^* are exactly optimal solutions for the LP problem, and $\sigma^k = \sigma^*$. Thus, almost all polynomial-time interior-point algorithms, coupled with the test procedure, are finite convergence algorithms for linear programming. Furthermore, if the LP problem has integral data, an exactly optimal solution can be generated before $(x^k)^T s^k < 2^{-O(L)}$.

We have tried the approach of [Y21]. The test LP problems are those described by Todd in [25]. They are very degenerate and contain unbounded variables. Although they are small-sized, some difficulties in solving these problems were reported. For our experiments, all problems are stopped with an exact optimal solution generated (up to machine accuracy for solving linear equations).

The first table is from a set of problems with the degenerate primal and dual optimal solutions. They are the Model 1 problems of Todd. The second table is from a set of problems with the unbounded optimal variables, as well as the degenerate optimal solutions. They are the Model 2 problems of Todd. While unbounded optimal facets caused some difficulty for some interior-point algorithms, they actually have a positive effect on our approach.

In the following tables, m and n are the dimensions of A. The third column is the average number of iterations required by our approach. For each size we solve 10 problems. The fourth column is the range of iterations when these 10 problems are solved. The fifth column is the average number of tests. The sixth column is the average number of iterations reported by Todd when these type problems are solved there. His stopping criterion is the primal-dual gap being reduced below $1.e-7$ in Table 2 and $1.e-2$ in Table 3.

m	n	Average no. of Itrs.	Range of Itrs.	Average no. of Tests	Todd's no. of Itrs.
20	40	12.2	8—18	1	
50	100	20.4	15—29	1	24.5
100	200	26.2	20—43	1	30.6
150	300	29.2	20—43	1	29.2
200	400	31.7	21—45	1	32.3

Table 2: Solving Model 1 Problems with the Termination Test

m	n	Average no. of Itrs.	Range of Itrs.	Average no. of Tests	Todd's no. of Itrs.
20	40	9.8	6—15	1	
50	100	15.5	12—25	1	28.7
100	200	16.8	14—21	1.1	29.6
150	300	16.5	12—22	1	30.6
200	400	17.7	16—21	1.1	30.9

Table 3: Solving Model 2 Problems with the Termination Test

2.3. Progress in Convex Nonlinear Optimization

Several interior point algorithms currently exist for convex nonlinear optimization (e.g., [10][14][18][20][24]). We augmented those with a potential reduction algorithm (e.g., [16][22][Y5][Y10][Y14]) which has exhibited promising practical efficiency. Han, Pardalos and myself implemented and tested the algorithm using vectorization on an IBM 3090-600S computer with Vector Facilities using the VS Fortran compiler. All numerical results were obtained in double precision. ESSL (Engineering and Scientific Subroutine Library) was used for vector and matrix operations. All CPU times reported in the following tables are given in seconds.

2.3.1. Convex Quadratic Programming

Two engineering problems, the obstacle and elastic-plastic torsion problems are solved here (e.g., [4][6][8] [21]). They have the following form

$$\min \; g(w) = \frac{1}{2}\int_P \| \nabla w \|^2 dP - \int_P \sigma w dP \qquad \text{(QP)}$$
$$\text{s.t. } w \in S = \{w \in H_0^1(P) : l \le w \le u \text{ on } P\},$$

where σ is a force function, P is a bounded open set, and $H_0^1(P)$ is a space of all functions with compact support in P such that w and $\| \nabla w \|^2$ belong to $L^2(P)$ (see [21]). Using finite approximations, the problems are reduced to the form of quadratic programming problems with up to half a million variables.

Tables 4 and 5 show computational results with the obstacle and the elastic-plastic torsion problems with up to half a million variables. τ represents the relative optimality error.

	I			II		
n	Itrs.	CPU	τ	Itrs.	CPU	τ
10000	15	16.3	5.2E-5	15	25.4	6.2E-5
40000	17	131.1	1.0E-5	17	203.9	5.4E-6
90000	18	437.6	5.6E-6	18	699.9	2.3E-6
115600	19	700.3	2.9E-6	18	1018.7	4.0E-6
160000	19	1035.8	6.0E-6	18	1534.7	1.9E-4
250000	20	2110.5	5.1E-6	19	3141.9	6.7E-5
360000	22	4090.3	4.4E-6	19	5312.4	2.2E-6
490000	28	8977.8	3.4E-6	20	8966.0	8.1E-5

Table 4: Iterations Count and CPU Time for Solving Two Obstacle Problems.

Figures 2 through 12 in the appendix show the initial solution, the solution after 3 iterations, the solution after 5 iterations, the solution after 10 iterations, and the final solution of the obstacle problem (I) and the elastic-plastic torsion problem and final solution of the obstacle problem (II). All problems have the same problem size $n = 10000$. Note that the surface of the figure implies the shape of an elastic membrane. The figures show that the solutions are changing rapidly until the 5th iteration, after which the final solution's pattern shows. The technical detail is given in [Y6][Y16].

	I			II			III		
n	Itrs.	CPU	τ	Itrs.	CPU	τ	*Itr.*	CPU	τ
10000	17	12.4	7.5E-5	15	8.8	1.7E-5	12	5.4	3.2E-4
40000	21	104.9	1.2E-5	18	71.0	1.1E-5	15	43.0	3.0E-5
90000	23	354.5	2.1E-5	20	236.7	1.1E-5	17	146.1	1.0E-5
115600	25	551.5	9.4E-6	21	355.8	3.9E-6	18	222.2	2.1E-6
160000	26	911.5	1.3E-5	22	597.5	2.9E-6	19	381.1	2.2E-6
250000	28	1810.5	9.1E-6	23	1128.6	7.5E-6	20	722.5	1.2E-6
360000	30	3338.4	5.2E-6	25	2141.1	1.5E-6	21	1280.7	1.4E-6
490000	32	5528.3	2.5E-6	26	3405.8	1.9E-6	22	2052.4	1.1E-6

Table 5: Iterations Count and CPU Time for Solving Three Elastic-Plastic Torsion Problems.

2.3.2. Entropy Optimization

We consider the entropy optimization problem

$$\min \ f(x) = c^T x + \sum_{j=1}^{n} w_j x_j \ln(x_j) \tag{EP}$$

$$\text{s.t. } x \in \Omega_p = \{x \in R^n : Ax = b, x \geq 0\}$$

where $c, w \in R^n$ and $w \geq 0, A \in R^{m \times n}$ and $b \in R^m$, and superscript ^{T} denotes the transpose operation. We assume that Ω_p has a nonempty interior. The entropy optimization problem is one of the most popular convex nonlinear programs, including such problems as system equilibrium [5], image reconstruction (e.g., [3], [13], [29]), and transportation distribution (e.g., [28]) among many others.

A special potential reduction algorithm was developed and implemented on the IBM 3090-600S with Vector Facilities. We solved image reconstruction problems similar to those described by Zenios and Censor [29]. Figure 13 in the Appendix depicts sources of x-ray, detectors, an object to be reconstructed, and a Cartesian grid. We assume that the sources and detectors are points and the rays between them are lines. The cartesian grid has n pixels and the pixels are numbered from bottom-left to top-right. The variable x_j denotes the x-ray attenuation function at pixel j and it is assumed that it has a constant value through the pixel $j, j = 1, \ldots, n$. The element a_{ij} of the matrix A denotes the length of intersection of ray i and pixel $j, i = 1, \ldots, m, j = 1, \ldots, n$. Since we use the discretized model, sum of the unknown attenuation function along the ray i can be represented by

$$\sum_{j=1}^{n} a_{ij} x_j = b_i, i = 1, \ldots, m,$$

where b_i is the physical measurement at the detector of the ray i. The constraint $\sum_{j=1}^{n} x_j = 1$ is added to the linear constraints because it is inherent in entropy optimization problems. Hence we have $m+1$ constraints and n variables. For our example, we have 337 constraints and 3249 variables by adding one constraint $e^T x = 1$ to $Ax = b$.

We reconstructed 3 simple diagram images (circle, triangle, and rectangle) and the head section image similar to the one described in [29] using the potential reduction algorithm. The algorithm was terminated when the duality gap $\delta^k = (x^k)^T s^k$ is less than 10^{-11} (for the rectangle image reconstruction problem, 10^{-9} was used). Figure 15 in the Appendix shows the real images and reconstructed images obtained from the algorithm. The CPU times (in seconds) and the number of iterations needed to solve the problems are given in Table 6. It was observed that after a few iterations of the algorithm, the reconstructed images showed the pattern of the original images.

Diagram	CPU time	Itr.
Head	49.2	29
Circle	52.0	31
Triangle	53.7	32
Rectangle	35.7	21

Table 6: Iteration Count and CPU Time for Reconstructing Various Diagrams.

Potra and myself [Y18] further developed an algorithm for entropy optimization with globally polynomial convergence and locally quadratic convergence.

2.4. Progress in Nonconvex Optimization

Two nonconvex problems, QP and LCP, are considered.

2.4.1. Nonconvex Quadratic Programming

General quadratic programming plays an important role in optimization theory. In one sense it is a continuous optimization and a fundamental subroutine for general nonlinear programming, and it is considered one of the most challenging global optimization problems. Motivated by the success of interior point algorithms for convex QP, several researchers, including myself, extended interior algorithms to solving nonconvex QP. The approaches are all based on solving the nonconvex quadratic program with an ellipsoidal approximation of the feasible region. We now briefly describe the approach.

For simplicity consider

Problem 1. Is there an $x \in R^n$ satisfying $\|x\|_\infty \le 1$ and $q(x) := x^T Qx/2 + c^T x < 0$?

Problem 1 can be written as an optimization problem

$$\min \ q(x) \qquad\qquad \text{(BQP)}$$

$$\text{s.t. } x \in B(r) = \{x \in R^n : \|x\|_\infty \le r\},$$

which is a special case of QP with a box constraint. Although it looks simple, BQP is a "hard" problem—one of the NP-complete problems. Now, we turn our attention to a related question:

Problem 2. Is there an $x \in R^n$ satisfying $\|x\|_2 \le 1$, and $q(x) < 0$?

Here the box constraint in BQP is replaced by a sphere or ellipsoid. Let us solve this spherically constrained problem

$$\begin{aligned} &\min \ q(x) && \text{(EQP)} \\ &\text{s.t. } x \in E(r) = \{x \in R^n : \|x\| \leq r\}, \end{aligned}$$

Then, **Problem 2** becomes: Is the (global) minimal value of EQP less than 0 for $r = 1$?

Note that $E(1) \subset B(1) \subset E(\sqrt{n})$. Although Q may be indefinite or negative definite, EQP is an "easy" problem and is closely related to Rayleigh's quotient problem that is completely characterized by the least eigenvalue of Q. Using the binary line search technique, I developed an algorithm for finding an ϵ solution in $O(n^3 \ln(1/\epsilon))$ arithmetic operations. The parallel complexity of EQP is $O(\ln^2 n + \ln n \ln(n + 1/\epsilon))$ parallel steps using $O(n^4)$ processors. The polynomial bit-complexity of the underlying decision problem is proved by Vavasis and Zippel [27].

In order to solve nonconvex BQP, we repeatedly solve EQP, which generates a solution sequence. If the solution sequence is convergent, then the limit solution is feasible for BQP, and satisfies both the first and the second order necessary conditions for BQP. See [Y3] for details.

2.4.2. Nonconvex Linear Complementarity Problems

Kojima, Megiddo and myself [Y7] used the potential function of [Y9][Y11] to develop an algorithm for the LCP:

$$x^T s = 0, \quad s = Mx + q, \quad \text{and} \quad (x, s) \geq 0.$$

To achieve a potential reduction, we used the scaled gradient projection method. This method solves a broader class of LCP problems, including the P-matrix LCP and sufficient-matrix LCP that are nonconvex problems.

Also using the potential reduction algorithm, Pardalos and myself [Y8] further analyzed a condition number for the general LCP, which characterizes the degree of difficulty for searching its solution. The condition number builds a connection from easy LCPs to hard LCPs, i.e., there is a continuous shift in the degree of difficulty of the LCP based on the size of the condition number. Consequently, we developed a new class of LCPs solvable in polynomial time.

The potential reduction algorithm was developed and implemented for solving the P-matrix linear complementarity problem. In Table 7, all averages were obtained from 5 problems and CPU times are given in seconds [Y22].

n	Avg. Itr.	Avg. CPU time
100	34.6	1.03
200	85.8	15.49
300	133.0	77.17
400	167.0	228.03
500	197.6	506.38

Table 7: Iteration Count and CPU Time for Solving P-Matrix LCPs.

3. Ongoing and Planned Research

Our main objective is to further study and improve interior-point algorithms and program codes for the solution of large-scale linear and nonlinear optimization problems. We have already had positive theoretical developments and computational experience with solving quadratic problems, quadratic knapsack problems, entropy optimization problems, and some nonconvex QP and LCP problems. From this progress we see that interior-point algorithms have much potential for solving large-scale optimization problems.

3.1. Research in Column Generation

In addition to the approaches described above, another polynomial algorithm with column generation was developed by Viadya [26] which utilizes a new volumetric center. Although the practical value of his algorithm remains to be seen, his algorithm has a better theoretical complexity than ours. Since the analytic center we use is not a *geometric* center for polytopes, I expect we will have some difficulties in obtaining worst-case complexity similar to that of Vaidya's algorithm.

On the other hand, Goffin et al. [9] and Mitchell[19] have reported encouraging computational results in a decomposition algorithm similar to ours. Recently, Goldstein[11] also reported similar behavior of our algorithm in solving a min-max problem. The number of inequality constraints generated is virtually independent of the total number of inequality constraints in the original problem.

Let m be the number of the variable and n be the number of inequality constraints in a linear feasibility problem. Then, the following is a *nonrigorous* probabilistic argument why the above behavior may be anticipated. When a hyperplane cuts through the analytic center, the potential reduction in one iteration is

$$\ln(\psi(\alpha)) = \ln(\|\alpha - e\|) + \sum_{j=1}^{n} \ln \alpha_j$$

with $e^T\alpha = n$ and $\alpha > 0$ [Y1]. Let α_j be independently drawn from a probability distribution, say the uniform distribution in [0, 1], and let

$$\alpha := \frac{n\alpha}{\sum_{j=1}^{n} \alpha_j}.$$

Then, the expected value of $\ln(\psi(\alpha))$ is $-O(n)$ (as confirmed by many simulation runs). If this reduction holds for each iteration, then after k iterations the total potential reduction is

$$\sum_{i=1}^{k} O(2m + i - 1) = O(2mk + k(k-1)/2)$$

where the number of initial inequalities is $2m$. Thus, the terminating condition in Theorem 3 of [Y1] indicates that we need $O(2mk + k(k-1)/2) = (2m + k)L$ iterations to stop the iterative process. Here, we see that k depends on L and m only. This analysis is a subject for further research.

I also expect that the column dropping technique developed in [Y12]Y13] can be incorporated with the column generation method.

3.2. Research in Algorithm Termination

Since the exact solution is already on the optimal facet, it can be cornered to the basic solution in no more than $n - m$ pivot operations. For example, if $|\sigma(x_B^*)| \leq m$, then x^* is a basic solution; otherwise, we do the following. Find any nonzero direction Δx_B in the null space of B, i.e.,

$$B\Delta x_B = 0.$$

(If the secondary objective exists, we may set Δx_B as the projection to the null space of B from the secondary objective vector.) Note that the null space was already available when the test procedure was performed. Assume $\max(\Delta x_B) > 0$ (otherwise, let $\Delta x_B = -\Delta x_B$). Then assign

$$x_B^* := x_B^* - \alpha \Delta x_B,$$

where α is the largest possible step size such that $x_B^* \geq 0$. This will force at least one additional component of $x_B^* \geq 0$ to zero. Continue this process until $|\sigma(x_B^*)| = m$, i.e., a basic solution is obtained. Note the next null space can be updated from the previous one in $O(m^2)$ arithmetic operations. The total number of required pivots in the process is at most $|\sigma(x_B^*)| - m \leq n - m$.

We are planning to implement our test procedure for large-scale "real" LP problems.

3.3. Research in Convex Nonlinear Optimization

We also plan to continue to solve large-scale engineering problems. One important issue is parallel computation in solving the system of linear equations, which is the major computational work in each iteration of our algorithm. Most of these engineering problems have special structure (Q is block tridiagonal matrix). Thus, we have an opportunity to use parallel linear system solvers (e.g., [1]) in our implementation. We anticipate significant improvement in computational speed.

3.4. Research in Nonconvex Optimization

We propose to further study and develop algorithms for the general indefinite quadratic programming problem. Recently, we have found that the complexity of the sphere-constrained quadratic problem can be further improved [Y2]. I have used a global Newton method and shown that the complexity for the sphere-constrained QP is $O(n^3 \ln(\ln(1/\epsilon)))$ arithmetic operations, which is a significant improvement.

Another topic is the probabilistic or average complexity analysis for interior-point algorithms in global optimization. Recently, some attempts are made for linear programming to explain the excellent behavior of these algorithms [Y4][Y19]. Since some computational results are now available, this analysis may also be worthwhile for nonconvex QP and LCP.

The third topic is the complexity issue for generating a stationary point of some nonconvex optimization problems. One recent result is given in [Y20]. It presents a fully polynomial-time approximation scheme for computing a stationary point of the general LCP.

4. Related Publication and Working Papers

4.1. Publication

[Y1] "A potential reduction algorithm allowing column generation," manuscript (1989), to appear in *SIAM Journal on Optimization.*

[Y2] "A new complexity result on minimization of a quadratic function over a sphere constraint," Working Paper No. 90-23, College of Business Administration, The University of Iowa (1990), to appear in the Proceedings of the Princeton Conference on Global Optimization (Princeton University Press, 1991).

[Y3] "On affine scaling algorithms for nonconvex quadratic programming," manuscript (revised July 1989), to appear in *Mathematical Programming.*

[Y4] "Comparative analysis of affine scaling algorithms for linear programming," Working Paper No. 90-1, College of Business Administration, The University of Iowa (1990), to appear in *Mathematical Programming.*

[Y5] "On some efficient interior point methods for nonlinear convex programming," with K. Kortanek and F. Potra, Working Paper No. 89-24, College of Business Administration, The University of Iowa (1989), to appear in *Linear Algebra and its Applications* (1990).

[Y6] "Computational aspects of an interior point algorithm for quadratic programming problems with box constraints," with C. Han and P. Pardalos, in T. F. Coleman and Y. Li eds., *Large-Scale Numerical Optimization* (SIAM, Philadelphia, 1990).

[Y7] "An interior-point potential reduction algorithm for the linear complementarity problem," with M. Kojima and N. Megiddo, Research Report RJ6486, IBM Almaden Research Center (San Jose, CA, 1988), to appear in *Mathematical Programming.*

[Y8] "A class of LCPs solvable in polynomial time," with P. Pardalos, to appear in *Linear Algebra and its Applications* (1990).

[Y9] "A class of projective transformations for linear programming," *SIAM J. Computing* **19** (1990) 457-466.

[Y10] "An $O(n^3L)$ potential reduction algorithm for linear programming," to appear in *Mathematical Programming* (1990).

[Y11] "A centered projective algorithm for linear programming," *Mathematics of Operations Research* **15** (1990) 508-529.

[Y12] "Recovering optimal basic variables in Karmarkar's polynomial algorithm for linear programming," *Mathematics of Operations Research* **15** (1990) 564-571.

[Y13] "A 'build-down' scheme for linear programming," *Mathematical Programming* **46** (1990) 61-72.

[Y14] "An extension of Karmarkar's projective algorithm for convex quadratic programming," with E. Tse, *Mathematical Programming* **44** (1989) 157-179.

4.2. Working Papers

[Y15] "On the finite convergence of interior-point algorithms," Working Paper No. 91-5, College of Business Administration, The University of Iowa (1991).

[Y16] "Solving some engineering problems using an interior-point algorithm," with C. Han and P. Pardalos, CS-91-04, Department of Computer Science, The Pennsylvania State University (University Park, PA, 1991).

[Y17] "Convergence behavior of some interior-point algorithms," with O. Güler, Working Paper No. 91-4, College of Business Administration, The University of Iowa (1991).

[Y18] "An interior-point algorithm for solving entropy optimization problems with globally linear and locally quadratic convergence rate," Working Paper No. 90-22, College of Business Administration, The University of Iowa (1990).

[Y19] "On adaptive-step primal-dual interior-point algorithms for linear programming," with S. Mizuno and M. J. Todd, Technical Report No. 944, School of ORIE, Cornell University (Ithaca, NY, 1990).

[Y20] "A fully polynomial-time approximation algorithm for computing a stationary point the general LCP," Working Paper No. 90-10, College of Business Administration, The University of Iowa (1990).

[Y21] "Near-boundary behavior of the primal-dual potential reduction algorithm for linear programming," with K. Kortanek, J. Kaliski and S. Huang, Working Paper No. 90-8, College of Business Administration, The University of Iowa (1990).

[Y22] "Solutions of P-matrix linear complementarity problems," with C. Han, J. Kaliski and P. Pardalos, manuscript (1990).

[Y23] "A build-up interior method for linear programming: affine scaling form," with G. B. Dantzig, Technical Report SOL 90-4, Stanford University (Stanford, CA, 1990).

References

[1] M. W. Berry and A. Sameh. *Multiprocessor schemes for solving block tridiagonal linear systems*, The International Journal of Supercomputing Applications 2 (1988), pp. 37-57.

[2] J. V. Burke, A. A. Goldstein and Y. Ye. *Translation cuts for minimization*, manuscript, Department of Mathematics, University of Washington (Seattle, WA, 1991).

[3] Y. Censor, T. Elfving, and G.T. Herman. *Methods for entropy maximization with applications in image processing*, Proceeding of the Third Scandinavian Conference on Image Analysis (Eds: P. Johansen and P.W. Becker), pp. 296-300, Chartwell-Bratt, Lund, Sweden (1983).

[4] C. W. Cryer. *The method of Cristopherson for solving free boundary problems for infinite journal bearings by means of finite differences*, Mathematics of Computation 25 (1971), pp. 435-443.

[5] G. B. Dantzig. *Linear Programming and Extensions* (Princeton University Press, Princeton, 1960).

[6] R. S. Dembo and U. Tulowitzki. *On the minimization of quadratic functions subject to box constraints*, Technical Report YALEU/DCS/TR-302, Department of Computer Science, Yale University (CT, 1984).

[7] D. M. Gay. *Stopping tests that compute optimal solutions for interior-point linear programming algorithms*, Numerical Analysis Manuscript 89-11, AT&T Bell Laboratories (Murray Hill, NJ, 1989).

[8] R. Glowinski. *Numerical methods for nonlinear variational problems* (Heidelberg Berlin New York, Springer, 1984).

[9] J. L. Goffin, A. Haurie and J. P. Vial, *Decomposition and nondifferentiable optimization with the projective algorithm,* Technical Report G-89-25, GERAD, McGill University (Canada, 1989).

[10] D. Goldfarb and S. Liu. *An $O(n^3L)$ primal interior point algorithm for convex quadratic programming*, Technical Report, Department of IEOR, Columbia University (New York, NY, 1988).

[11] A. A. Goldstein, private communication (1990).

[12] C. C. Gonzaga, *An algorithm for solving linear programming problems in $O(n^3L)$ operations,* In N. Megiddo (ed.): Progress in Mathematical Programming (Springer-Verlag, New York, 1989).

[13] G.T. Herman. *A relaxation method for reconstructing objects from noisy X-rays*, Mathematical Programming 8 (1975), pp. 1-19.

[14] F. Jarre. *On the complexity of a numerical algorithm for solving smooth convex programs by following a central path*, Manuscript, University of Wurzburg (West Germany, 1988).

[15] N. Karmarkar, *A new polynomial-time algorithm for linear programming,* Combinatorica 4 (1984), pp. 373-395.

[16] M. Kojima, S. Mizuno and A. Yoshise. *An* $O(\sqrt{n}L)$ *iteration potential reduction algorithm for linear complementarity problems,* Research Report, Department of Information Science, Tokyo Institute of Technology (Tokyo, Japan, 1988).

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

[18] S. Mehrotra and J. Sun. *An algorithm for convex quadratic programming that requires* $O(n^{3.5}L)$ *arithmetic operations*, Manuscript, Department of IEMS, Northwestern University (Evanston, IL, 1988).

[19] J. E. Mitchell and M. J. Todd, *Solving combinatorial optimization problems using Karmarkar's algorithm. Part I: Theory*, Math Report No. 172, Rensselaer Polytechnic Institute (Troy, NY, 1989).

[20] R. C. Monteiro and I. Adler. *An extension of Karmarkar type algorithm to a class of convex separable programming problems with global rate of convergence*, Mathematics of Operations Research 15 (1990), pp. 408-422.

[21] J. Moré and G. Toraldo. *Algorithm for bound constrained quadratic programming problems*, Numerische Mathematik 55 (1989), pp. 377-400.

[22] Ju. E. Nesterov and A. S. Nemirovsky. *Self-concordant functions and polynomial-time methods in convex programming*, USSR Academy of Sciences, Central Economic and Mathematics Institute (Moscow, USSR, 1989).

[23] J. Renegar. *A polynomial-time algorithm, based on Newton's method, for linear programming,* Mathematical Programming 40 (1988), pp. 59-93.

[24] 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.

[25] M. J. Todd. *The effects of degeneracy and null and unbounded variables on variants of Karmarkar's linear programming algorithm*, in: T. F. Coleman and Y. Li, eds., Large-Scale Numerical Optimization (SIAM, Philadelphia, 1991), pp. 81-91.

[26] P. M. Vaidya. *A new algorithm for minimizing convex functions over convex sets,* manuscript, Bell Labs (Murray Hill, NJ, 1989).

[27] S. A. Vavasis and R. Zippel. *Proving polynomial-time for sphere-constrained quadratic programming*, TR 90-1182, Department of Computer Science, Cornell University (Ithaca, NY, 1990).

[28] D. S. Wong. *Maximum likelihood, entropy maximization, and geometric programming approaches to the calibration of trip distribution models*, Transportation Res., Part B, 15 (1981), pp. 329-343.

[29] S.A. Zenios and Y. Censor. *Parallel computing with block-iterative image reconstruction algorithms*, To appear in Applied Numerical Mathematics, IMACS.