A UNIFIED APPROACH FOR DESIGN OF LP POLYNOMIAL ALGORITHMS

Xiang-Sun Zhang

Institute of Applied Mathematics
Chinese Academy of Sciences
Beijing 100080, China

Abstract By summarizing Khachiyan's algorithm and Karmarkar's algorithm for linear program (LP) a unified methodology for the design of polynomial-time algorithms for LP is presented in this paper. A key concept is the so-called extended binary search (EBS) algorithm introduced by the author. It is used as a unified model to analyze the complexities of the existing modern LP algorithms and possibly, help designing new algorithms with polynomial-time iterations for problems in other areas.

Key Words Extended Binary Search, Khachiyan's Algorithm, Karmarkar's Algorithm, Unified Methodology

چکیده دراین مقاله با خلاصه کردن دو الگوریتم خاچیان و کارمارکار در برنامه سازی خطی (LP)، به یک روش یکسان شده برای طراحی الگوریتمهای چند جمله ای زمانی LP می رسیم. جستجوی دوتایی توسعه یافته، یک مفهوم کلیدی الگوریتم است که توسط نویسنده مقاله، معرفی می شود. این مفهوم به عنوان الگویی برای تحلیل پیچیدگیهای الگوریتمهای موجود روش LP و احتمالاً کمک در طراحی الگوریتمهای جدید با تکرار برحسب زمان چند جمله ایها برای مسایل دیگر، قابل استفاده خواهد بود.

INTRODUCTION

The presentation of Khachiyan's algorithm[1,2] and Karmarkar's algorithm[3] not only solves the long-standing open problem in the history of mathematical programming, i.e., the existence of polynomial-time algorithm of LP, but also establishes a new methodology for design of combinatorial optimization problem algorithms. Our interest is to investigate methodologies implied in these excellent pieces of research and discuss the possibilty to apply them to general class of optimization problems even with continuous variables.

A general optimization problem can be expressed as a pair $(F.C)$ where F is the set of feasible solution, and C is a cost function with F as it's domain. For a combinatorial problem, F is a discrete set, and $C(F)$ is a discrete set in R^1. The discreteness of a feasible set and a cost function value set is a combinatorial character of the problem. For a continuous optimization problem, such as a nonliner programming problem, the feasible set is generally a subset of R^n and the cost function C maps F to a subset in R^1. Although traditional algorithms of nonlinear programming are iteration-type algorithm and generate a sequecne (discrete!) of points which aproaches to a desired optimal solution, we can not make the cost function values of these points being embeded into a prepointed discrete set in R^1. Hence, from the view of traditional theory of nonlinear programming it is difficult to introduce combinatorial characters into the problem's feasible set or value set of the cost function.

LP is a class between combinatorial and continuous optimization problems. Traditional format to LP is to optimize a linear function on a polyhedron. If we restrict ourselves to search the optimal solution in the vertex set of the polyhedron, then we combinatorialize the feasible set of an LP. But the traditional LP algorithm-Simplex Method-dose not effectively use this combinatorial character to solve any instance of

LP in ploynomial time except some classes of problems which satisfy Hirsch conjecture (see [4], p 160). Hirsch defines the diameter of a polyhedron as the maximum, over all pairs of vertices, of the minimum length of an edge-path joining these vertices and claims that the diameter of a bounded polyhedron with dimension n and m facets does not exceed $(m-n)$. Because the lower bounds on maximum diameters provide bounds on the number of iterations required by the best simplex-type method applied to the worst problem of a given size, some classes of LP problems[5] which satisfy Hirsch's conjectures certainly make Simplex-method a polynomial-time method. The reason that the simplex-type method does not effectively solve all instances of LP is that the algorithms in nonlinear programming simplex method do not reduce the cost function value according to precombinatorialized ratios.

To devlop LP's polynomial algorithm Khachiyan and Karmarkar discarded the traditional LP format which, as metioned above, only posesses nomial combinatorial characters, and adopt respectively the linear strict inequalities (LSI) and a special linear inequalities (SLI) as follows:

LSI: Given an $m \times n$ integer matrix A and an m-vector b, there is an n-vector x such that $Ax<b$.

SLI: Given an n-integer vector c and an $m \times n$ integer matrix A satisfying $Ae = 0$, where e is an n-vector with every component equal to one, there is an n-vector x such that

$$Ax = 0,\ e^Tx = 1,\quad x \geq 0.\quad c^Tx \leq 0$$

To outward seeming problems LSI and SLI have no apparent combinatorial characters. They seemingly are classes of continuous problems. Therefore, it is interesting to investigate how Khachiyan or Karmarkar introduce combinatorial characters into these two types of problems. Such a research may help us to apply the principles of combinatories to the research of continuous optimization problems.

1. EXTENDED BINARY SEARCH

Binary search is a basic algorithm in combinatorics [6]. Supose we wish to determine an unknown integer x between 1 and B ($B>1$) by asking questions of the form *"Is $x>\eta$?"* for some η of our choice. We can do this by first asking whether x is in the upper or lower half of the interval [1, B], then by asking whether x is in the upper or lower half of the remaining smaller interval, and so on, until the interval in which x lies contains exactly one integer. This will happen after $\lceil \log B \rceil$ such questions (By $\lceil x \rceil$ we denote the smallest integer y such that $y \geq x$). Because the size of the problem is $\lceil \log B \rceil$, then binary search is a linear-time complexity algorithm.

We now extend this basic combinatorial method as follows: Let T be an integer such that $T \geq 2$.

```
Algorithm EBS
begin
     given $T$, $X := 1$, $Y := B$
  step    if  $Y - X \leq 1$ then stop
        else $\eta := X + (Y - X)/T$ or $\eta := X + (T-1)$
                  $(Y - X)/T$
        if $x > \eta$ then $X := \eta$ go to step
        else $Y := \eta$ go to step
end
```

For Algorithm EBS we have the following theorem:

***Theorem* 1.** Suppose integer $T \geq 2$, the Algorithm EBS determines an unknown integer x between 1 and B after $K = \lceil \log_{\frac{T}{T-1}} B \rceil$ questions of the form *"Is $x>\eta$?"*.

Proof. By B' we denote the length of the current remaining interval. Note that either $\eta = X + (Y-X)/T$ or $\eta = X + (T-1)(Y-X)/T$ the length of next remaining interval does not exceed $(T-1)B'/T$. Thus, by induction, the remaining length of the interval after K

questions is $(\frac{T-1}{T})^k B$. It is equivalent to applying a common binary search (BS) to interval $[1, 2^k(\frac{T-1}{T})^k B]$ with the same iterations. In other words, applying EBS to [1, B] and applying BS to $[1, 2^k(\frac{T-1}{T})^k B]$, one has the same remaining length after k iterations. This suggests that the number of iterations as one uses EBS to [1, B] can be estimated by the number of iterations of BS to $[1, 2^k(\frac{T-1}{T})^k B]$. Therefore,

$$K = \lceil \log 2^K(\frac{T-1}{T})^K B \rceil = K + \lceil \log (\frac{T-1}{T})^K B \rceil$$

$$\lceil \log (\frac{T-1}{T})^K B \rceil = 0$$

that is,

$$0 < (\frac{T-1}{T})^K B \le 1$$

then we take $K = [\log_{T/T-1} B \le 1]$ to prove the theorem.

***Corollary* 1.** Theorem 1 is valid for any $T \ge 2$.

***Corollary* 2.** Theorem 1 is valid for any T such that $2 > T > 1$. But in this case $K = \lceil \log_T B \rceil$.

***Proof*.** In this case one iteration of EBS reduces an interval of iength B' to length B'/T, then after K iterations the remaining length does not exceed B/T^k. It is equivalent to applying BS to $[1, (\frac{2}{T})^k B]$. By the similar calculation of Theorem 1, we have $K = \lceil \log B \rceil$.

***Remark* 1.** When T has the value $2 \ge T \ge 1$. T is the ratio of lengths of two successive remaining intervals.

Khachiyan and Karmarkar adopted respectively LSI and SLI to solve LP problems, but both LSI and SLI seem to be continuous-type problems. How they use potential combinatorial characters of these problems and develop polynomial time algorithms? We will gain an insight into this question from the view of above described algorithm-EBS.

A research mechanism that we suggest to answer the question consists of the following steps:

(i) For a given optimization problem, choose one element related to F or c in the pair (F,c) as the object to be combinatorialized. We call this element as a potential combinatorial character quantity (ccq in short). For example, as we will see, in Khachiyan's method the volume of feasible set in the ellipsoid is taken as the ccq; in Karmarkar's method the left side value of inequality $c^Tx \le 0$ is taken as the ccq. Note that it is necessary that there is a one-to-one correspondence between an approximate solution and a value of the ccq. Then finding the optimal solution is equivalent to finding the correspondent value of the ccq.

(ii) Design an iterative-type algorithm for the given problem such that the corresponding value of the ccq for the generated sequence of approximate solutions be covered by a pre-defined domain with upper and lower bounds. Without loss of the generality we suppose that the domain is an interval like [1, B].

(iii) Design the algorithm with one more requirement that each iteration makes the ccq change in the interval [1, B] according to the nature of an EBS with some T of our choice, so that in some sense finding the optimal solution for the primal problem is equivalent to finding a determined value of the ccq in [1, B] by using EBS. Therefore, by theorem 1 or corollary 2, the algorithm will terminate after at most $\lceil \log_{\frac{T}{T-1}} B \rceil$ or $\lceil \log_T B \rceil$ iterations.

The following theorem 2 states that if we can restrict the ccq in an interval [1, B] with $B = c.d^{\,p(L)}$ where $c>0$, $d>1$ and $p(L)$ is a polynomial function of the problem size L, then EBS with a proper parameter T will terminate in a finite number of iterations that are bounded by a polynomial of L.

***Theorem* 2.** Let $p(x)$ and $q(x)$ be polynomials of x with positive coefficients in the first term. $a, b, c, d,$ are constants larger than one. If the parameters B and T in theorem 1 are: $B = a.b^{p(L)}$, $T = c.d^{1/q(L)}$ and $2 \ge T > 1$, then the iteration number of EBS applied to [1, B] has a polynomial of L as the upper bound.

Proof. By corollary 2

$$K = \lceil \log_T B \rceil = \lceil \frac{q(L)\log_d a + p(L)\, q(L) \log_d b}{q(L) \log_d c + 1} \rceil$$

It is reasonable to assume that $q(L)\ \log_d c \geq 0$ for sufficiently large L, then we have

$$K \leq [p(L).\ q(L).\ \log_d ab]. \qquad Q.E.D.$$

2. COMPLEXITY ANALYSIS OF KHACHIYAN'S AND KARMARKAR'S METHOD BY USING THE ABOVE METIONED RESEARCH MACHANISM

Khachiyan discussed problem LSI, he implicitly defined the volume of the set $\{x|Ax<b\}$ as the ccq. With the definition of problem size $L = O(m.n + \lceil \log|W| \rceil)$ where $W = \prod A.\prod b$, (Here we use symbol $\prod x$ to write the product of the nonzero component of x(vector or matrix)), Khachiyan gave the upper and lower bound of the ccq.

***Lemma* 1** [2]. If an LSI system of size L has a solution, then the set of solutions within the sphere $\|x\| \leq n2^L$ has volume V at least $2^{-(n+2)L}$.

In other words, the set of solutions within the sphere $\|x\| \leq n.2^L$ has volume V in interval $[2^{-(n+2)L}, (2n.2^L)^n]$. If we let $2^{(n+2)L}V$ be the ccq of the problem, then it has domian in $[1, 2^{(2n+2)L+n \log 2n}] \equiv I$

From the initial ball $\|x\| \leq n.2^L$, the ellipsoid algorithm constructs a sequence of volume-decreasing ellipsoids which contain the solution set until it finds an ellipsoid, of which the center is a solution of the problem. When the algorithm iterates form one ellipsoid to the next ellipsoid, it is equivalent to applying EBS to the ccq in the interval I. By theorem 1 and 2 we need only to show that the ratio of two successive ellipsiods satisfies the parameter requirment in theorem 2. If so, the algorithm will terminate after polynomial (*in L*) iterations. Otherwise it will contradict to lemma 1.

By remark 1 and theorem 2 if $T = cd^{1/q(L)}$, then we will have the expected result. In fact, Khachiyan's ellipsoids have the described property that can be seen from the following rewritten lemma of [2].

***Lemma* 2** [2]. Denote the volumes of two successive ellipsoids E_j and E_{j+1} by Vol(E_j) and Vol (E_{j+1}), then

$$Vol(E_j)/\ Vol(E_{j+1}) > 2^{1/2(n+1)} \equiv 2^{1/q(L)} = T$$

there $q(L) = 2(n+1)$ is a low-order polynomial of L and apparently $2 > T > 1$.

This lemma explains that if the volume of current ellipsoid is B', then calculating the volume of next ellipsoid is equivalent to finding the remaining length of [1, B'] by using an EBS with $\eta = B'/T$ and $T = 2^{1/q(L)}$ where $q(L) = 2(n+1)$.

Now we turn to discuss Karmarkar's method. There are many variants of Karmarkar's method as summarized in [7]. We limit in this paper to the basic form of Karmarkar's method (BKM) as in [8]. The algorithm is as follows:

```
Algorithm BKM
begin
    given $x^0 = \frac{1}{n}e$, $r = \sqrt{n/(n-1)}$, $Ax^0 = 0$, $k := 0$
step    if $c^T x^k < 0$, then return $x^k$
            else if $k \geq N = \lceil (\frac{2}{1 - ln 2}) n^2 ln(nR) \rceil$ then
                answer "no"
                else set
                $D := diag(x_1^k, ... x_n^k)$
                $y^{k+1} := (I - DA^T(AD^2A^T)^{-1} AD - n^{-1}e.\ e^T)DC$
                $z^{k+1} := e - \frac{1}{2} rY^{k+1} / \| y^{k+1} \|$
                $x^{k+1} := Dz^{k+1} / \| Dz^{k+1} \|$
                $k := k+1$ go to step
end
```

Firstly Karmarkar implicitly uses the value $c^T x / (\prod x)^{1/n}$ as the ccq. The following lemma 3 and 4 describe an interval of the ccq. Now the size L of the problem is $O(m.n+\lceil m.n \log R \rceil)$ where $R=\max\{|a_{ij}|, |c_j|\}$ which is also appearing in Algorithm BKM.

***Lemma* 3.** (Theorem 15.1 of [8]) If SLI has solution, then the successive $x^k s$ generated by the BKM satisfy

$$\frac{(c^T x^{k+1})^n}{\prod x^{k+1}} < \frac{2}{e} \frac{(c^T x^k)^n}{\prod x^k}$$

that is

$$ccq^{k+1} / ccq^k < (\tfrac{2}{e})^{1/n} = e^{\frac{1}{n}(ln\,2\text{-}1)} = e^{1/q(L)}$$

where $q(L)= n/\,(ln2 - 1)$ is a low-order polynomial of L_2

***Lemma* 4.** Suppose the initial point is $x^0=(\frac{1}{n},\dots,\frac{1}{n})^T$, then the sequence $\{x^k\}$ generated by the algorithm BKM has values $c^Tx^k/(\prod x^k)^{1/n}$ in interval $I=[1/R^n n^{n-1}, nR]$.

Proof. Since $c^Tx^0 \le \frac{1}{n}(n.\max\{c_i\}) \le R$ and by lemma 3, ccq^k is a decreasing sequence, then $n.R$ is an upper bound of $c^Tx^k/(\prod x^k)^{1/n}$. On the other hand, consider problem $c^Tx^*=\min\{c^Tx \mid Ax=0, e^Tx=1, x\ge 0\}$. If the primal SLI has no solution, then $c^Tx^*>0$. In this case, for any feasible x such that $c^Tx>0$, it is from the basic theory of LP that there is an optimal basic feasible solution x^* such that $c^Tx \ge c^Tx^* > 0$. Therefore, by using Cramer rule, we know that the denominate of the basic feasible solution is of absolute value at most n^nR^n. By integrality, the value of c^Tx with x such that $c^Tx>0$ is not possibly less than $1/n^nR^n$. Hence the lower-bound of the interval is $1/n^{n-1}.\ R^n$.

From these lemmas we can see the algorithm BKM iterates the $ccq=n^{n-1}.\ R.\ c^Tx/(\prod x)^{1/n}$ in interval $[1, n^nR^{n+1}]$. Now we need only to show that this can be equivalent to an EBS that satisfies the condition of theorem 2 in order to prove the polynomial convergence. First we can see

$$n^nR^{n+1} = e^{n\ln n+(n+1)\ln R} = e^{P(L)}$$

where $P(L)$ is a low order polynomial of L. Secondly, lemma 3 shows that the algorithm BKM is exactly operating according to the pattern pointed by theorem 2.

CONCLUSION

The discussion in sections 1 and 2 reveals that Khachiyan's method and Karmarkar's method look very different in the form and mathematical techniques, but they have inherent coincidence. Both authors combinatorialize their problem that looks by appearance as a coutinuous problem. The inherent process of algorithm design consists of choosing a potential combinatorial character quantity to be combinatorialized, determining an interval, with upper bound of at most an exponential function of a polynomial of L, that covers the domain of the ccq, finally designing an algorithm operating parallel to an EBS on the interval.

We hope that this unified methodology can be used to handle other types of continuous problems, especially to study the iteration complexity of general algorithms for continuous problems.

REFERENCES

1. L. G. Khachiyan, "A Polynomial Algorithm for Linear Programming," *Doklady Akad Nauk USSR*, 244, No. 5(1979) 1093-96, Translated in *Soviet Math.* Doklady, 20, 191-94.
2. B. Aspvall and R. E. Stone, "Khachiyan's Linear Programming Algorithm," *Journal of Algorithms*, 1, No. 1(1980).
3. N. Karmarkar, "A New Polynomial-time Algorithm for Linear Programming," *Combinatorica* 4(1984), 373-395.
4. G. B. Dantzig, Linear Programming and Extensions,", Princeton University Press, Princeton, New Jersey, (1963).

5. V. Klee and P. Kleinschmidt, "The D-step Conjecture and its Relatives," *Mathematics of Operations Research* 12, (1987) 718-755.

6. C. H. Papadimitriou and K. Steiglitz, "Combinatorial Optimization: Algorithms and Complexity," Prentice-Hall, Inc., Englewood Cliffs, New Jersey, (1982).

7. M. J. Todd, "Recent Developments and New Directions in Linear Programming," Technical Report, School of Operations Research and Industrial Engineering, Cornel University, (1989).

8. A. Schrijver, "Theory of Linear and Integer Programming," John Wiley and Sons, New York, (1986).