Polska
Akademia
Nauk
Instytut
Badań
Systemowych

Methodology and applications of decision support systems

Proceedings of the 3-rd
Polish-Finnish Symposium
Gdańsk-Sobieszewo, September 26-29, 1988

edited by
Roman Kulikowski

Polska Akademia Nauk Instytut Badań Systemowych

Methodology and applications of decision support systems

Proceedings of the 3-rd
Polish-Finnish Symposium
Gdańsk-Sobieszewo, September 26-29, 1988

edited by
Roman Kulikowski

Warszawa 1989

Secretary of the Conference
dr. Andrzej Stachurski

Wykonano z gotowych oryginałów tekstowych
dostarczonych przez autorów

ISBN 83-00-02543-X

A VARIANT OF KARMARKAR ALGORITHM FOR LP PROBLEMS MAKING USE OF SENSITIVITY ANALYSIS

ANDRZEJ STACHURSKI[1]

Systems Research Institute, Polish Academy of Sciences
Newelska 6, 01-447 Warszawa, Poland

ABSTRACT

The aim of this paper is to present a modified version of Karmarkar algorithm for solving linear programming problems. The recently developed Karmarkar algorithm possesses an interesting property of polynomial dependence of the number of arithmetic operations on the size n of the problem.

It is commonly known that the solution cost in Karmarkar method is dominated by calculation of the projection of the vector $-D^Tc$ onto the linear subspace $ADy=0$, where D is a scaling matrix (with the off-diagonal entries equal to 0) and A is the rectangular matrix of some subset of linear constraints (having the form $Ax=0$).

Our main idea is to obtain an approximate solution to the projection problem by taking the first order Taylor expansion formula and applying the sensitivity analysis results to the QP problem, which is equivalent to finding the projection to get the first order derivatives.

Key Words: linear programming, Karmarkar method, sensitivity analysis.

1. FORMULATION OF THE PROBLEM

Since the development of the polynomial time algorithm by N. Karmarkar (1984) there is considerable interest in its applications and in developing strategies for modifying its rules in order to achieve greater computational efficiency. It is an interior point barrier algorithm. The problem considered by Karmarkar has the following form:

$$\min \ c^Tx, \tag{1}$$

subject to

$$Ax=0, \tag{2}$$

[1]Sponsored by the Research Program RPBR/RP.I.02.2.5

$$e^T x = n, \qquad (3)$$

$$x \geq 0, \qquad (4)$$

where A is an $m \times n$ matrix, x and c column n-vectors, e – the n-vector of all ones, and where it is assumed that at x^*, the solution to (1), $c^T x^* = 0$. We follow the suggestion of Shanno and Marsten (1985) in using $e^T x = n$ instead of $e^T x = 1$ as in Karmarkar's original (1984) formulation. We will omit here the discussion of the transformation of a standard linear programming problem to the form (1-4) defined above. This is not essential to our presentation and can be found, for instance, in the papers of Shanno (1987) and Tomlin (1987). We shall also assume that we have an initial feasible point $x^0 > 0$.

2. THE KARMARKAR ALGORITHM

Since some familiarity with the Karmarkar's paper (1984) is important, we will now briefly summarize the projective algorithm. Under the above assumptions let $D = \mathrm{diag}(x_1^0, x_2^0, \ldots, x_n^0)$ and employ a projective transformation and its inverse defined by:

$$y = n \cdot \frac{D^{-1} x}{e^T D^{-1} x} \; ; \qquad x = n \cdot \frac{Dy}{e^T Dy} . \qquad (5)$$

The projective transformation transforms problem (1-4) into a fractional program on a simplex in the space of y variables:

$$\min \frac{c^T Dy}{e^T Dy} \qquad (6)$$

subject to:

$$ADy = 0, \qquad (7)$$

$$e^T y = n, \qquad (8)$$

$$y \geq 0. \tag{9}$$

Let us observe that from the definition of D, the point x^0 in x-space is mapped onto the point $e^T=(1,1,...,1)$. The general idea now is to neglect the denominator in the objective function. It can be shown that problem (6-9) is equivalent to the following linear programming problem

$$\min \ v(y)=\tilde{c}^T y, \tag{10}$$

subject to:

$$\tilde{A}y=0, \tag{11}$$

$$e^T y=n, \tag{12}$$

$$y \geq 0, \tag{13}$$

where $\tilde{c}=Dc$, $\tilde{A}=AD$. Consequently $v^*=v(y^*)=0$. A "large" improving step for the problem (9-12) is taken away from the center of the simplex $S=\{y \in R^n;\ e^T y=n,\ y \geq 0\}$ to the new point in y-space. This new point is transformed back to the x-space and the result is evaluated.

More specifically, the algorithm generates a sequence of points $x^0, x^1, ..., x^k$, where $x_j^0>0$ $(j=1,2,...,n)$ as follows:

1. Define $D=\mathrm{diag}(x_0^k, x_1^k, ..., x_n^k)$ and

$$B = \begin{bmatrix} AD \\ e^T \end{bmatrix}. \tag{14}$$

2. Compute

$$c_p = \left[I-B^T(BB^T)^{-1}B\right]Dc, \tag{15}$$

i.e., project the steepest descent direction for problem (9-12) into the null space of its constraint matrix B.

3. Normalize c_p and scale it by the radius of the largest sphere, which can be inscribed into the simplex S to generate the direction vector

$$p = \frac{c_p}{\|c_p\|\sqrt{n(n-1)}} \,. \qquad (16)$$

4. Take a descent step of length α in the direction p to get a new feasible point for (10-13):

$$y' = e - \alpha p. \qquad (17)$$

5. Calculate x^{k+1} as the image of the inverse projective transformation

$$x^{k+1} = \frac{Dy'}{e^T Dy'} \,. \qquad (18)$$

If the new point satisfies the termination criterion then STOP. Otherwise, set $k:=k+1$ and go to Step 1.

It is stressed by many authors that the majority of the work at each iteration is carried out in Step 2. This step ensures the feasibility of the new point.

3. CALCULATE THE PROJECTIONS CHEAPLY

It is commonly known that the solution cost is dominated by calculation of the projection in Step 2. The matrix $\hat{B}$ changes from one iteration to another in a very unpleasant way. Therefore it is difficult to update at a low cost any factorization of the matrix BB^T. From the very beginning of the development of the Karmarkar method, many authors have tried to overcome this difficulty (see, for instance Dennis, Morshedi and Turner (1987), Goldfarb and Mehrotha (1988a, 1988b), Shanno (1988), Todd (1988)).

In this paper we present a new, different approach. The key observation for our considerations is that if the subsequent approximating points x^k and x^l (with $l>k$) do not differ substantially, then one can use the Taylor expansion formula at point x^k to calculate an approximate projection at point x^l instead of the exact one defined by formula (15).

Let us denote by z^0 the projection calculated at point x^k with $D^0=\mathrm{diag}(x_1^k,x_2^k,\dots,x_n^k)$, i.e.

$$z^0=\left[I-B^T(BB^T)^{-1}B\right]D^0c. \tag{19}$$

and respectively by z the projection calculated at point x^l with $D'=\mathrm{diag}(x_1^l,x_2^l,\dots,x_n^l)$, i.e.,

$$z=\left[I-B^T(BB^T)^{-1}B\right]D'c. \tag{20}$$

Let us recall that B also depends on D (see formula (14)).

Using the Taylor formula one may write z as

$$z=z^0+z'\Delta D+\dots , \tag{21}$$

where z' is the first order derivative of the projection (19) with respect to D at $D=D^0$ and

$$\Delta D=D'-D^0=\mathrm{diag}(x_1^l,x_2^l,\dots,x_n^l)-\mathrm{diag}(x_1^k,x_2^k,\dots,x_n^k).$$

In formula (21) only the first order expansion is indicated. The natural question arises immediately: is it worth considering the higher order terms in formula (21)? The answer is negative, since the total number of arithmetic operations necessary to calculate the second order approximations (21) is similar to the number of operations required to find the projection by directly applying the original formula (15). For the higher order approximations it is even larger.

The second important question is: how to calculate z' in formula (21)? It is easily seen that calculation of the projection z in (15) is equivalent to solving the following quadratic programming problem:

$$\min_{z} \frac{1}{2} \|z-\tilde{c}\|^2 \tag{22a}$$

subject to:

$$Bz=0. \tag{22b}$$

The answer to the second question follows from the sensitivity analysis methodology applied to the QP problem (22). Let us start from the formulation of the Kuhn-Tucker optimality conditions for (22). In our case they are necessary and sufficient.

$$y-\tilde{c}+B^T\lambda=0, \tag{23a}$$

$$By=0. \tag{23b}$$

Matrix B and vector $\tilde{c}$ depend on the parameters placed in the main diagonal of matrix D. Let's recall that $\tilde{c}=Dc$,

$$D=\mathrm{diag}(x_1^k,x_2^k,\ldots,x_n^k)=\mathrm{diag}(u_1,u_2,\ldots,u_n)$$

and

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

where we denoted x^k by u.

The differentiation of (23) with respect to the parameters u leads towards the following formulae

$$\frac{\partial y}{\partial u} - \frac{\partial \tilde{c}}{\partial u} + \left[\frac{\partial B}{\partial u} \right]^T \lambda + B^T \frac{\partial \lambda}{\partial u} = 0, \tag{24a}$$

$$\frac{\partial B}{\partial u} y + B \frac{\partial y}{\partial u} = 0. \qquad (24b)$$

As a result, simple recalculations show that $\partial\lambda/\partial u$ may be found as the solution of the following m sets of linear equations

$$- \frac{\partial B}{\partial u} y - B\frac{\partial \tilde{c}}{\partial u} + B\left[\frac{\partial B}{\partial u} \right]^T \lambda + BB^T \frac{\partial \lambda}{\partial u} = 0 \qquad (25a)$$

with the same matrix BB^T.

Accordingly $\partial y/\partial u$ may be obtained by the following formula

$$\frac{\partial y}{\partial u} = \frac{\partial \tilde{c}}{\partial u} - \left[\frac{\partial B}{\partial u} \right]^T \lambda - B^T \frac{\partial \lambda}{\partial u}. \qquad (25b)$$

If any factorization of B is known then in order to find $\partial\lambda/\partial u_i$ one has to solve (for each $i=1,2,\ldots,n$) one equation with an upper triangular matrix and one equation with a lower triangular matrix. The number of arithmetic operations (multiplications and additions) required for that is of the order $O(m^2)$. Calculation of the i-th term $\left[- \frac{\partial B}{\partial u} y - B\frac{\partial \tilde{c}}{\partial u} + B\left[\frac{\partial B}{\partial u}\right]^T \lambda \right]_i$ requires $O(m^2n)$ arithmetic operations for each i. Therefore the total number of computational operations necessary to find $\partial\lambda/\partial u$ is $O(m^2n^2)$. In order to find $\partial y/\partial u$ by (25b) one also needs $O(m^2n)$ arithmetic operations. This leads us toward the conclusion that the total number of arithmetic operations necessary to solve equations (25) may be reduced to $O(m^2n^2)$ in the subsequent iterations following that with $D=D^0$.

Let us notice furthermore, that $B\left[\frac{\partial B}{\partial u}\right]^T$ and $B\frac{\partial \tilde{c}}{\partial u}$ do not depend on u and may be calculated once at the point u^0 and stored, and that

$$\frac{\partial B}{\partial u_i} = \begin{bmatrix} 0 & \dots & 0 & a_{1i} & 0 & \dots & 0 \\ 0 & \dots & 0 & a_{2i} & 0 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & 0 & a_{ni} & 0 & \dots & 0 \\ 0 & \dots & 0 & 0 & 0 & \dots & 0 \end{bmatrix}. \qquad (26)$$

These observations suggest the way of further reduction of computational cost.

5. CONCLUDING REMARKS

In this paper we briefly presented a new approach to calculating Karmarkar's projections cheaply. Our main idea stems from the sensitivity analysis results being applied to the quadratic programming problem, the solution of which is identical with the desired projection. In our approach to finding an approximate solution only the knowledge of a factorization of B from some past iteration is required. This seems to be advantageous compared with the direct way of solving the projection problem with high accuracy at each iteration. The eventual numerical realization will require of course an answer to many additional questions. We hope however, that it is worth investigating it.

Let us stress furthermore that it is very easy to propose a parallel variant of the method for solving equations (25). It may be that on a parallel computer with many parallel processors even an approximation by the second order Taylor expansion formula could be realized in reasonable computational time. We rejected this possibility in our considerations as too expensive from the computational point of view.

REFERENCES

Dennis J.E. Jr., Morshedi A.M., Turner K. (1987), A variable metric variant of the Karmarkar algorithm for linear programming, *Mathematical Programming*, 39, 1-20.

Goldfarb D., Mehrotha S. (1988a), Relaxed variants of Karmarkar's method, *Mathematical Programming*, 40, 183-195.

Goldfarb D., Mehrotha S. (1988b), A relaxed version of Karmarkar's method, *Mathematical Programming*, 40, 289-315.

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

Shanno D.F. (1988), Computing Karmarkar's projection quickly, *Mathematical Programming*, 41, 61-71.

Shanno D.F., Marsten R.E. (1985), On implementing Karmarkar's method, *Working Paper* 85-01, Graduate School of Administration, U.C. Davis (Davis, CA).

Todd M.J. (1988), Exploiting special structure in Karmarkar's linear programming algorithm, *Mathematical Programming*, 41, 97-113.

Tomlin J.A. (1987), An experimental approach to Karmarkar's projective method for linear programming, *Mathematical Programming Study*, 31, 175-191.