Operations Research Letters 12 (1992) 181–186
North-Holland

September 1992

A build-up variant of the logarithmic barrier method for LP *

D. den Hertog, C. Roos and T. Terlaky **

Faculty of Technical Mathematics and Computer Science, Delft University of Technology, Mekelweg 4, PO Box 5031, 2600 GA Delft, Netherlands

Received July 1991
Revised March 1992

We propose a strategy for building up the linear program while using a logarithmic barrier method. The method starts with a (small) subset of the dual constraints, and follows the corresponding central path until the iterate is close to (or violates) one of the constraints, which is in turn added to the current system. This process is repeated until an optimal solution is reached. If a constraint is added to the current system, the central path will, of course, change. We analyze the effect on the barrier function value if a constraint is added. More importantly, we give an upper bound for the number of iterations needed to return to the new path. We prove that in the worst case the complexity is the same as that of the standard logarithmic barrier method. In practice this build-up scheme is likely to save a great deal of computation.

interior point method; linear programming; logarithmic barrier function; polynomial algorithm; build-up variant

1. Introduction

Karmarkar [4] pioneered the rapidly developing field of interior point methods for linear programming. These methods not only have nice theoretical properties, but are very efficient from a practical point of view, especially for large problems. One drawback to all interior point methods is the great computational effort required for each iteration. In each iteration the search direction p is obtained by solving a linear system with normal matrix $AD^{-2}A^{\mathrm{T}}$, where A is the constraint matrix $(m \times n)$ and D a positive diagonal matrix depending on the current iterate. Therefore, working with a subset of the dual constraints rather than the full system, would save a great deal of computation.

The first attempt to save computations is the so-called 'build-down' process, proposed by Ye [12], [14]. In his approach, a criterion for detecting (non)binding constraints is derived on the basis of a circumscribed ellipsoid. If a constraint is detected to be non-binding in the optimal set, it is removed from the system. Consequently, the system becomes increasingly smaller, which reduces the computational effort for computing the normal matrix $AD^{-2}A^{T}$. However, the speed of the detection process is crucial. If the nonbinding constraints are only detected during the last stage of the algorithm, the reduction in computation is neglegible. To the best of our knowledge, there are no computational results to be found in literature concerning this build-down process.

The second attempt to save computations is the 'build-up' or 'column generation' method. Papers on column generation techniques within interior point methods were first written by Mitchell [5] and Goffin and Vial [3] for the pro-

* This work is completed with the support of a research grant from SHELL.

** On leave from the Eötvös University, Budapest, and partially supported by OTKA No. 2116.
Correspondence to: C. Roos, Faculty of Technical Mathematics and Computer Science, Delft University of Technology, Mekelweg 4, PO Box 5031, 2600 GA Delft, Netherlands.

jective method. However, these papers provide no theoretical analysis of the effect on the potential function and/or the number of iterations after the addition of a column/row.

Ye [13] proposed a (non-interior) potential reduction method for the linear feasibility problem which allows column generation. In each iteration an inequality violated at the current center is added to the system (in a shifted position), until a feasible point has been found. He proved that such a point can be found in $O(\sqrt{n}L)$ iterations. Although each linear programming problem can be formulated as a linear feasibility problem, this is an inefficient way of solving linear programming problems.

Dantzig and Ye [1] proposed a build-up scheme for the dual affine scaling algorithm. This method differs from the 'standard' affine scaling method in that the ellipsoid chosen to generate the search direction p is constructed from a set of m 'promising' dual constraints. If the next iterate $y+p$ violates one of the other constraints, this constraint is added to the current system and a new ellipsoid and search direction (using the new set of constraints) are calculated. After making the step, a new set of m promising dual constraints is selected.

Tone [10] proposed an active-set strategy for Ye's [11] dual potential reduction method. In this strategy the search direction is also constructed from a subset of constraints which have small dual slacks in the current iterate. More constraints are added if no sufficient potential reduction is obtained. After making the step a new set of dual constraints, with small slack values, is selected. This algorithm converges to an optimal solution in $O(\sqrt{n}L)$ iterations.

Elaborating the above ideas, we propose in this note a build-up strategy for the long-step logarithmic barrier method. Comparing with the usual long-step logarithmic barrier method, this approach has two advantages: it enables us to work with a smaller subset in each iteration, and it gives a better (theoretical) complexity (the same, at worst) than that of the standard logarithmic barrier method. The build-up method starts with a (small) subset of the dual constraints, and follows the corresponding central path until the iterate is close to (or violates) one of the other constraints, at which point the constraint is added to the current system. This process is repeated until an optimal solution with prescribed precision is obtained. Note that this approach differs from that of Dantzig and Ye [1], in that we do not select a new subset of m dual constraints at each iteration.

If a constraint is added to the current system, the central path will, of course, change. We analyze the effect on the barrier function value after adding a constraint. More importantly, an upper bound is obtained for the additional number of iterations needed to return to the new path. The analysis is an extension of the analysis for the 'standard' logarithmic barrier method given in [8], [9] and [2].

Section 2 contains some of the results obtained for the 'standard' logarithmic barrier method in [8], [9] and [2], which we need there. In Section 3 we state the build-up strategy and analyze its complexity.

The following notations are used. We denote by e the vector of all ones, and I the identity matrix. Given an n-dimensional vector s, we denote by S the $n \times n$ diagonal matrix whose diagonal entries are the coordinates s_j of s; s^T is the transpose of the vector s and the same notation holds for matrices. Finally $\|s\|$ denotes the l_2 norm of s.

2. Properties near the central path

In this section we present the problem, its central path and a measure for the distance to the central path. Some well-known results for logarithmic barrier methods, which are needed in the sequel, are recalled from [8], [9], and [2].

Consider the well-known standard linear programming problem:

(P)

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

Here A is an $m \times n$ matrix, b and c are m- and n-dimensional vectors, respectively. In this note we deal with the dual formulation for (P):

(D)

$$\max\{b^T y:\ A^T y + s = c,\ s \geq 0\}.$$

Without loss of generality, we assume that all the coefficients are integer. L denotes the length of the input data of (D). We make the standard assumptions that the interior of the feasible region of (D) is non-empty and that the optimal set is bounded. In order to simplify the analysis, we also assume that A has full rank, though this assumption is not essential.

The dual logarithmic barrier function is given by

$$f(y, \mu) := \frac{b^{\mathrm{T}} y}{\mu} + \sum_{j=1}^{n} \ln s_j, \tag{1}$$

where μ is a positive parameter. It achieves a maximum value at a unique interior point. The necessary and sufficient first order optimality conditions for this point are:

$$A^{\mathrm{T}} y + s = c, \quad s \geq 0,$$
$$Ax = b, \quad x \geq 0,$$
$$Sx = \mu e.$$

The unique solution to this system is denoted by $(x(\mu), y(\mu), s(\mu))$. The primal and dual central path is defined as the solution set $x(\mu)$ and $y(\mu)$, $\mu > 0$, respectively.

Roos and Vial [8] introduced the following measure for the distance of an interior feasible point y to the central point $y(\mu)$:

$$\delta(y, \mu) := \min_x \left\{ \left\| \frac{Sx}{\mu} - e \right\| : Ax = b \right\}.$$

The unique solution to the minimization problem in the definition of $\delta(y, \mu)$ is denoted by $x(y, \mu)$. If the Newton direction for (1) is denoted by p, then it can be verified that $\delta(y, \mu) = \| S^{-1} A^{\mathrm{T}} p \|$. A closed formula for p is:

$$p = (AS^{-2}A^{\mathrm{T}})^{-1} \left(\frac{b}{\mu} - AS^{-1}e \right). \tag{2}$$

It can also easily be verified that

$$y = y(\mu) \Leftrightarrow \delta(y, \mu) = 0.$$

The following lemmas and theorem are recalled from [8], [9] and [2].

Lemma 1. *If $\delta := x(y, \mu)$ is primal feasible. Moreover,*

$$\mu(n - \delta\sqrt{n}) \leq c^{\mathrm{T}} x - b^{\mathrm{T}} y \leq \mu(n + \delta\sqrt{n}).$$

Lemma 2. *If $\delta(y, \mu) < 1$ then $y^* = y + p$ is a strictly feasible point for* (D), *and*

$$\delta(y^*, \mu) \leq \delta(y, \mu)^2.$$

Lemma 3. *If $\delta := \delta(y, \mu) < 1$, then*

$$f(y(\mu), \mu) - f(y, \mu) \leq \frac{\delta^2}{1 - \delta^2}.$$

Lemma 4. *Let $\tilde{\alpha} := (1 + \delta)^{-1}$. Then $y + \tilde{\alpha} p$ is strictly feasible and*

$$\Delta f := f(y + \tilde{\alpha} p, \mu) - f(y, \mu) \geq \delta - \ln(1 + \delta).$$

Lemma 5. *If $\bar{\mu} := (1 - \theta)\mu$ and $\delta := \delta(y, \mu) \leq \frac{1}{2}$ then*

$$f(y(\bar{\mu}), \bar{\mu}) - f(y, \bar{\mu}) \leq \frac{\theta}{1 - \theta}(\theta n + 3\sqrt{n}) + \tfrac{1}{3}.$$

In the standard long-step logarithmic barrier method line searches along Newton directions are carried out until the iterate is in the vicinity of the current center. At this point the barrier parameter is reduced, followed by Newton iterations for the purpose of returning into the vicinity of the new center. Suppose the logarithmic barrier method, as defined in [2], starts with barrier parameter μ_0 and $0 < \theta < 1$, independent of n (say $\theta = \frac{1}{2}$), then the following theorem gives an upper bound for the total number of iterations (see [2]).

Theorem 1. *After at most $K = \mathrm{O}(L + \ln n\mu_0)$ reductions of the barrier parameter, the long-step algorithm ends up with a primal and a dual solution such that $x^{\mathrm{T}} s \leq 2^{-2L}$. Each reduction of the barrier parameter requires at most $\mathrm{O}(n)$ Newton iterations.*

3. The build-up strategy

In the build-up logarithmic barrier method we start with a certain (small) subset Q of the index set $\{1, \ldots, n\}$. Then we start the logarithmic barrier method, with respect to the dual constraints in Q. This means that we work with the subproblem

(D_Q)

$$\max\{b^T y : a_i^T y \leq c_i, \ i \in Q\}$$

of the full problem (D). At each iteration we check for each index $i \notin Q$ if

$$s_i < 2^{-t}, \tag{3}$$

where t is some 'proximity' parameter. If there is such an index, we add it to Q, go back to the previous iterate (for which $s_i \geq 2^{-t}$) and continue the process. In other words, a constraint is added to the subproblem if the candidate for the next iterate violates or is close to this constraint.

Let $\delta_Q(y, \mu)$, $f_Q(y, \mu)$ and p_Q be the δ-measure, the barrier function value and the Newton direction, respectively, with regard to the subsystem Q instead of the full system. Note that computing p_Q requires much less computation time than computing p, if Q is small compared with the full system (see equation (2)). Observe that $\delta_Q(y, \mu)$ can easily be computed, since $\delta_Q(y, \mu) = \| S_Q^{-1} A_Q^{\mathrm{T}} p_Q \|$, where the subscript Q again denotes the restriction to Q. The algorithm goes as follows:

Build-Up Algorithm
Input:
$\mu = \mu_0$ is the barrier parameter value,
t is a proximity parameter;
θ is the reduction parameter, $0 < \theta < 1$;
Q is the initial subset of dual constraints;
y is a given interior feasible point such that $\delta_Q(y, \mu) \leq \frac{1}{2}$;
begin
  while $s^{\mathrm{T}} x(y, \mu) > 2^{-2L}$ **do**
    begin
      $\mu := (1-\theta)\mu$;
      while $\delta_Q(y, \mu) > \frac{1}{2}$ **do**
        begin
          $\tilde{y} := y$;
          $\tilde{\alpha} := \arg\max_{\alpha > 0} \{ f_Q(y + \alpha p_Q, \mu)$:
            $s_i - \alpha a_i^{\mathrm{T}} p_Q > 0,\ \forall i \in Q \}$;
          $y := y + \tilde{\alpha} p_Q$;
          if $\exists i \notin Q$: $s_i < 2^{-t}$ **then**
            begin
              $y := \tilde{y}$;
              $Q := Q \cup \{ i: s_i < 2^{-t}, i \notin Q \}$
            end
        end
    end
end.

In the Appendix it is shown how to obtain a starting point y and index set Q such that y is feasible for the whole problem and $\delta_Q(y, \mu) \leq \frac{1}{2}$.

Let q denote the cardinality of the current subset Q. If θ is independent of q (say $\frac{1}{2}$), then we know from Lemma 4 and 5 that the standard logarithmic barrier method requires at most $O(q)$ Newton iterations between two reductions of the barrier parameter. Lemma 6 gives this number if constraints are added between the two reductions.

Lemma 6. *Between two successive reductions of the barrier parameter μ, the Build-Up Algorithm requires $O(q + r(t+L))$ Newton iterations, where r denotes the number of constraints added between these reductions of μ.*

Proof. Let $\bar{\mu}$ be the current value and μ the previous value of the barrier parameter (i.e. $\bar{\mu} := (1-\theta)\mu$). Let $i_1, \ldots, i_r$ denote the indices of the r constraints added to Q while the barrier parameter has value $\bar{\mu}$. And let y^k, for $k = 1, \ldots, r$, denote the iterate just after the i_k-th constraint has then added, while $y^k(\bar{\mu})$ and $f^k(y, \bar{\mu})$ denote the corresponding center and barrier function. Finally, let y^0 denote the iterate at the moment that μ is reduced to $\bar{\mu}$ (i.e. y^0 is in the vicinity of $y^0(\mu)$), and $f^0(y, \mu)$ the corresponding barrier function.

Since in each iteration the barrier function value decreases by a constant (Lemma 4), the number of Newton iterations needed to go from y^0 to the vicinity of $y^r(\bar{\mu})$ is, at most, proportional to

$$P := f^0(y^1, \bar{\mu}) - f^0(y^0, \bar{\mu}) + \sum_{k=1}^{r-1} \left(f^k(y^{k+1}, \bar{\mu}) - f^k(y^k, \bar{\mu}) \right) + f^r(y^r(\bar{\mu}), \bar{\mu}) - f^r(y^r, \bar{\mu}). \quad (4)$$

Since $f^k(y^k, \bar{\mu}) = f^{k-1}(y^k, \bar{\mu}) + \ln s_{i_k}^k$,

$$\sum_{k=1}^{r-1} \left(f^k(y^{k+1}, \bar{\mu}) - f^k(y^k, \bar{\mu}) \right) = \sum_{k=1}^{r-1} \left(f^k(y^{k+1}, \bar{\mu}) - f^{k-1}(y^k, \bar{\mu}) - \ln s_{i_k}^k \right) = f^{r-1}(y^r, \bar{\mu}) - f^0(y^1, \bar{\mu}) - \sum_{k=1}^{r-1} \ln s_{i_k}^k.$$

Substituting this into (4), while using $f^r(y^r, \bar{\mu}) = f^{r-1}(y^r, \bar{\mu}) + \ln s_{i_r}^r$, we obtain

$$P = f^r(y^r(\bar{\mu}), \bar{\mu}) - f^0(y^0, \bar{\mu}) - \sum_{k=1}^{r} \ln s_{i_k}^k = f^r(y^r(\bar{\mu}), \bar{\mu}) - f^0(y^0(\bar{\mu}), \bar{\mu}) + f^0(y^0(\bar{\mu}), \bar{\mu}) - f^0(y^0, \bar{\mu}) - \sum_{k=1}^{r} \ln s_{i_k}^k. \quad (5)$$

Moreover,

$$f^r(y^r(\bar{\mu}), \bar{\mu}) - f^0(y^0(\bar{\mu}), \bar{\mu})$$
$$= \sum_{k=1}^{r} \left(f^k(y^k(\bar{\mu}), \bar{\mu}) - f^{k-1}(y^{k-1}(\bar{\mu}), \bar{\mu})\right)$$
$$\leq \sum_{k=1}^{r} \left(f^k(y^k(\bar{\mu}), \bar{\mu}) - f^{k-1}(y^k(\bar{\mu}), \bar{\mu})\right)$$
$$= \sum_{k=1}^{r} \ln s_{i_k}^k(\bar{\mu}), \tag{6}$$

where the inequality follows because $y^{k-1}(\bar{\mu})$ maximizes $f^{k-1}(y, \bar{\mu})$ and, therefore,

$$f^{k-1}(y^{k-1}(\bar{\mu}), \bar{\mu}) \geq f^{k-1}(y^k(\bar{\mu}), \bar{\mu}).$$

Since y^0 is in the vicinity of $y^0(\mu)$ we have, according to Lemma 5:

$$f^0(y^0(\bar{\mu}), \bar{\mu}) - f^0(y^0, \bar{\mu})$$
$$\leq \frac{\theta}{1-\theta}\left(\theta q + 3\sqrt{q}\right) + \tfrac{1}{3}. \tag{7}$$

Now substituting (6) and (7) into (5) gives

$$P \leq \frac{\theta}{1-\theta}\left(\theta q + 3\sqrt{q}\right) + \tfrac{1}{3} - \sum_{k=1}^{r} \ln s_{i_k}^k + \sum_{k=1}^{r} \ln s_{i_k}^k(\bar{\mu})$$
$$\leq \frac{\theta}{1-\theta}\left(\theta q + 3\sqrt{q}\right) + \tfrac{1}{3} + r(t+L) \ln 2.$$

The last inequality follows because criterion (3) gives us $s_{i_k}^k \geq 2^{-t}$ and because $s_{i_k}^k(\bar{\mu}) \leq 2^L$. Taking $\theta = \frac{1}{2}$, the lemma follows. □

Now let Q^*, with cardinality q^*, be the final subset of dual constraints used by the algorithm. The following theorem gives an upper bound for the total number of iterations.

Theorem 2. *After at most* $O(q^*(t + L + \ln q^*\mu_0))$ *Newton iterations the Build-Up Algorithm ends up with a* 2^{-2L}*-optimal solution.*

Proof. According to Theorem 1 after $O(L + \ln q^*\mu_0)$ reductions of the barrier parameter, the algorithm ends up with a primal and dual solution y for the subproblem (D_{Q^*}) corresponding to Q^*, such that the duality gap is less than 2^{-2L}. Since y is also feasible for (D) and the subsystem Q^* forms a relaxation of (D), we also have that $z^* - b^Ty \leq 2^{-2L}$, where z^* is the optimal value of (D).

If r constraints are added between two reductions of the barrier parameter then, according to Lemma 6, at most $O(q^* + r(t+L))$ Newton iterations are needed. This means that adding a constraint requires, on the average, at most $O(t + L)$ additional Newton iterations. During the process at most q^* constraints were added. So, as far as the work due to adding constraints goes, at most $O(q^*(t+L))$ Newton iterations are needed. Consequently, the overall complexity is $O(q^*(L + t + \ln q^*\mu_0))$. □

We note that in the algorithm all constraints which violate criterion (3) are added to the current subsystem of constraints. It is easy to see that the analysis also permits us to add one (or some) of these constraints. Intuitively, this will lead to smaller values of q^*. The following lemma even shows that if we take $t \geq L$, then only supporting hyperplanes of the feasible region are added.

Theorem 3. *Suppose, in the Build-Up Algorithm we only add the constraint with the smallest slack value of all constraints which violate the criterion. Then if we set* $t \geq L$*, only supporting hyperplanes of the feasible region will be added.*

Proof. Suppose the k-th constraint is not supporting. The minimal value for the slack variable s_k over the feasible region is assumed in a vertex of the feasible region. For a vertex it is well known that either $s_i = 0$ or $s_i \geq 2^{-L}$. Since the k-th constraint is not supporting, we obtain that $s_k \geq 2^{-L}$ for all feasible points. Consequently, the k-th constraint will never be added by the Build-Up Algorithm. □

Appendix: Initialization of the Algorithm

Suppose the dual constraints are divided in the sets Q and $\bar{Q}$ such that $Q \cup \bar{Q} = \{1, \ldots, n\}$. Now assume that y and $(x_Q, x_{\bar{Q}})$ are the (weighted) μ_0-centers for the problem:

$$\max \frac{b^Ty}{\mu_0} + \sum_{i \in Q} \ln s_i + \sum_{i \in \bar{Q}} \frac{1}{\sqrt{\bar{q}}} \ln s_i,$$

that is they satisfy

$$A_Q x_Q + A_{\bar{Q}} x_{\bar{Q}} = b,$$

$$A_Q^{\mathrm{T}} y + s_Q = c_Q,$$

$$A_{\bar{Q}}^{\mathrm{T}} = s_{\bar{Q}} = c_{\bar{Q}},$$

$$X_Q s_Q = \mu_0 e,$$

$$X_{\bar{Q}} s_{\bar{Q}} = \frac{\mu_0}{\bar{q}} e.$$

It is well-known that such points can be obtained by transforming the original problem. See e.g. Monteiro and Adler [6], Renegar [7]. If we denote $a := A_{\bar{Q}} x_{\bar{Q}}$ then $(x_Q, 1)$ and y satisfy

$$A_Q x_Q + \xi a = b,$$

$$A_Q^{\mathrm{T}} y + s_Q = c_Q,$$

$$a^T y + \eta = x_{\bar{Q}}^{\mathrm{T}} c_{\bar{Q}},$$

$$X_Q s_Q = \mu_0 e,$$

$$\xi \eta = \mu_0.$$

This means that y is feasible for the whole problem (D) and is the center for the subproblem consisting of the q constraints and an additional constraint $a^{\mathrm{T}} y \leq x_{\bar{Q}}^{\mathrm{T}} c_{\bar{Q}}$, which is also a valid inequality for (D). Note that the constraints which are not in Q are condensed into this additional constraint.

References

[1] G.B. Dantzig and Y. Ye, "A build-up interior method for linear programming: Affine scaling form", Technical Report SOL 90-4, Systems Optimization Laboratory, Department of Operations Research, Stanford University, CA, 1990.

[2] D. Den Hertog, C. Roos and J.-Ph. Vial, "A complexity reduction for the long-step path-following algorithm for linear programming", *SIAM J. Optim.* 2/1, 71–87 (1992).

[3] J.-L. Goffin and J.-Ph. Vial, "Cutting planes and column generation techniques with the projective algorithm", *SIAM J. Optim.* **65**, 409–429 (1990).

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

[5] J.E. Mitchell, "Karmarkar's algorithm and combinatorial optimization problems", Dissertation, Cornell University, Ithaca, NY, 1988.

[6] R.D.C. Monteiro and I. Adler, "Interior path following primal-dual algorithms, Part I: linear programming", *Math. Programming* **44**, 27–41 (1989).

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

[8] C. Roos and J.-Ph. Vial, "A polynomial method of approximate centers for linear programming", *Math. Programming* **54**/3, 295–305 (1992).

[9] C. Roos and J.-Ph. Vial, "Long steps with the logarithmic penalty barrier function in linear programming", in: J. Gabszevwicz, J.-F. Richard and L. Wolsey (eds.), *Economic Decision-Making: Games, Economics and Optimization*, dedicated to Jacques H. Drèze, Elsevier Science Publisher, Amsterdam, 1989, 433–441.

[10] K. Tone, "An active-set strategy in interior point method for linear programming", Working Paper, Graduate School of Policy Science, Saitama University, Urawa, Saitama 338, Japan, 1991.

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

[12] Y. Ye, "Eliminating columns and rows in potential reduction and path-following algorithms for linear programming", Working Paper Series No. 89-7, Department of Management Sciences, The University of Iowa, Iowa City, Iowa.

[13] Y. Ye, "A potential reduction algorithm allowing column generation", *SIAM J. Optim.* **2**, 7–20 (1992).

[14] Y. Ye, "The "build-down" scheme for linear programming", *Math. Programming* **46**, 61–72 (1990).