CENTER FOR
PARALLEL OPTIMIZATION

A THREE-PHASE ALGORITHM FOR BLOCK-STRUCTURED OPTIMIZATION

by

Gary L. Schultz and Robert R. Meyer

Computer Sciences Technical Report #932

May 1990

A THREE-PHASE ALGORITHM FOR BLOCK-STRUCTURED OPTIMIZATION

GARY L. SCHULTZ AND ROBERT R. MEYER

Work supported in part by NSF grant CCR-8709952 and AFOSR grant 89-0410.

Abstract. We develop a decomposition method, based on barrier functions, for solving block angular linear programs. The convergence properties of the method are briefly described. We then present promising computational results for one of the largest classes of linear programming models occurring in the literature, a set of multicommodity flow problems with up to 100,000 constraints and 300,000 variables.

1. Block Angular Linear Programs. The block angular linear programming problem is: Find x^* which minimizes $c^\mathsf{T} x$ subject to the constraints

$$\left(\begin{array}{cccc} \boxed{A_1} & & & \\ & \boxed{A_2} & & \\ & & \ddots & \\ & & & \boxed{A_K} \\ \multicolumn{4}{c}{\boxed{\qquad\qquad D \qquad\qquad}} \end{array}\right) \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_K \end{pmatrix} \begin{matrix} = \\ = \\ \vdots \\ = \\ \leq \end{matrix} \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_K \\ d \end{pmatrix} \tag{1}$$

and the bounds $\mathbf{0} \leq x \leq u$. The dimension of the kth block of A is $M(k) \times N(k)$ and we define $M := \sum_k M(k)$ and $N := \sum_k N(k)$. The dimension of D is $J \times N$.

The algorithm begins by finding x^0 as the solution of the relaxed problem

$$\operatorname*{minimize}_{x} c^\mathsf{T} x \text{ subject to } Ax = b \text{ and } 0 \leq x \leq u. \tag{2}$$

A two-phase barrier method, described in the next two sections, starts with x^0 and produces a solution that is optimal within a specified tolerance.

We make the assumption that solving a set of subproblems of the form

$$(3) \qquad \underset{x_k}{\text{minimize}}\, \tilde{c}_k^\top x_k \text{ subject to } A_k x_k = \tilde{b}_k \text{ and } \tilde{\ell}_k \leq x_k \leq \tilde{u}_k \qquad \text{for } k = 1, \ldots, K$$

is easy relative to solving the entire original problem (1). (Note that the relaxed problem (2) decomposes into subproblems of this form.) This is quite reasonable for many reasons. First of all, the block structure of A allows us to solve a set of problems of the form (3) as K independent subproblems. If K is the order of 10 to 100, then this is an ideal task for today's MIMD machines. Secondly, it may be possible to utilize special structure in the A_k that would be more difficult to exploit if the coupling constraints ($Dx \leq d$) were kept. One important example is the case in which (1) is a multicommodity flow problem and each A_k is a node-arc-incidence matrix. And third, the difficulty of solving most linear programs in practice increases as a quadratic or cubic function with the size of the problem.

2. The Shifted Barrier Scheme. In this section we outline a scheme that allows us to deal with the block constraints explicitly and consider the effect of the coupling constraints implicitly. Define the feasible set of the block constraints as

$$\mathcal{B} := \{x | Ax = b \text{ and } \mathbf{0} \leq x \leq u\},$$

and the feasible set of the coupling constraints as

$$\mathcal{C} := \{x | Dx \leq d\}.$$

Suppose $x^i \in \mathcal{B}$ is given. At each iteration, we approximate the original problem by the *barrier problem*

$$(4) \qquad \underset{x}{\text{minimize}}\, f(x, \tau, \theta) := c^\top x - \tau \sum_{j=1}^{J} \ln(\theta_j - D_j x) \text{ subject to } x \in \mathcal{B}$$

where $0 < \tau \in \mathbb{R}$ and $0 < \theta \in \mathbb{R}^J$ are parameters. Note that $f(x, \tau, \theta)$ is convex in x with domain $\operatorname{dom} f(\cdot, \tau, \theta) = \{x | Dx < \theta\}$. Allowing $\theta \neq d$ has the property of "shifting" the barrier. For this reason we call the nonlinear portion of f a *shifted logarithmic barrier function.*

After finding x^0 as the solution of (2) the parameters θ^0 and τ^0 are specified by taking $\tau^0 > 0$ and

$$(5) \qquad \theta_j^0 = \begin{cases} d_j & \text{if } D_j x^0 < d_j \\ D_j x^0 + \Theta & \text{if } D_j x^0 \geq d_j \end{cases}$$

where $\Theta > 0$. This will have the effect of making $x_0 \in \mathcal{B}$ an interior point of $\operatorname{dom} f(\cdot, \tau^0, \theta^0)$. We then compute x^{i+1} by doing one step of a multi-dimensional search on the barrier problem (4). If $Dx^{i+1} < d$, we have produced a feasible point. If not, then we modify θ by taking

$$(6) \qquad \theta_j^{i+1} = \begin{cases} d_j & \text{if } D_j x^{i+1} < d_j \\ \lambda_\theta D_j x^{i+1} + (1-\lambda_\theta)\theta_j^i & \text{if } D_j x^{i+1} \geq d_j \end{cases}$$

where $\lambda_\theta \in (0,1)$ is a constant, while maintaining $\tau^{i+1} = \tau^i$. Then set $i \leftarrow i+1$ and do the process again.

We have shown [SM90] that if a point $x \in \mathcal{B} \cap \operatorname{int} \mathcal{C}$ exists, then *such a point may be generated in a finite number of iterations.* In order to do this, however, the barrier problems (4) must be solved accurately enough, e.g., the objective function value is within some constant of being optimal ($f(x^{i+1}, \tau^i, \theta^i) \leq f^*(\tau^i, \theta^i) + \beta$ where $f^*(\tau^i, \theta^i)$ is the optimal value function of (4), and β is constant).

Once the method has $\theta^{\hat{\imath}} = d$ so that $x^{\hat{\imath}} \in \mathcal{B} \cap \operatorname{int} \mathcal{C}$, we maintain $\theta^i = d$ and begin to relax the effect of the barrier terms in f. This is done by letting $\tau^i \downarrow 0$. Questions of convergence become important at this point. Classical convergence results for barrier functions [FM68] state that minimizers of (4) converge to minimizers of (1) as $\tau \downarrow 0$. We use a sequence $\{\tau^i\}$ generated by the recurrence

$$\tau^{i+1} \leftarrow \max\{\lambda_\tau \tau^i, \tau_{\inf}\}$$

where $\tau^0, \tau_{\inf} > 0$ and $\lambda_\tau \in [0,1)$. The following result (see page 102 of [FM68] or page 341 of [McC83], for example) is then used to choose an appropriate $\tau_{\inf}$:

THEOREM 2.1. *Let x^* be an optimal solution of (1), and say $\bar{x}$ is a minimizer of (4) with $\tau > 0$ and $\theta = d$. Then $c^\top \bar{x} \geq c^\top x^* \geq c^\top \bar{x} - \tau J$. (Recall that J is the number of rows of D.)*

So if the user chooses $\tau_{\inf} = \varepsilon / J$ then solving the barrier problem (4) in the limit produces limit points $\bar{x} \in \mathcal{B} \cap \operatorname{int} \mathcal{C}$ such that $|c^\top \bar{x} - c^\top x^*| \leq \varepsilon$.

In summary, this method is a three phase process. The first phase may be called the *relaxed phase.* It consists of solving (2). If this problem is infeasible, so is the original problem. The second phase may be called the *feasibility phase.* During this phase we are seeking $x^{\hat{\imath}} \in \mathcal{B} \cap \operatorname{int} \mathcal{C}$ by trying to force $\theta = d$. If the feasibility phase succeeds, we move on to the third and *final phase.* This phase seeks to approximate a minimizer of (1) by reducing the effects of the barrier term, i.e. by reducing τ.

3. Solving Barrier Problems. In the previous section we described a method that is based on the barrier problem (4). Suppose we are given a current point $x^i \in \mathcal{B}$. We use a two-stage generalization of the Frank-Wolfe method to obtain the next iterate x^{i+1}. In the first stage, we linearize (4) and add a trust region $R(x^i)$ giving the following problem:

$$(7) \qquad \underset{\delta x}{\operatorname{minimize}} \nabla_x f(x^i, \tau, \theta) \delta x \text{ subject to } x^i + \delta x \in \mathcal{B} \cap R(x^i).$$

This is of the form (3) with $\tilde{c} = \nabla_x f(x^i, \tau, \theta)$, $\tilde{b} = 0$ and $R(x^i)$ consisting of simple bounds. Denote a solution of (7) by δx^i.

The purpose of the trust region is to prevent the linearization from ignoring the poles of the barrier function. We suggest the following choice for the trust region:

$$R(x^i) := \left\{ x = x^i + \delta x | D_{jkn}\delta x_{kn} \leq \max\left\{\gamma_{\text{inf}}, \gamma(\theta_j - D_j x^i)\right\} \ \forall j, k, n \right\}$$

where $\gamma_{\text{inf}}, \gamma > 0$ are constants. Since the boundary of the trust region is always a distance of $\min_{jkn} |\gamma_{\text{inf}}/D_{jkn}| > 0$ from the current point, the standard convergence proof of the Frank-Wolfe method may be easily accommodated.

In the second stage, we solve the *coordination problem*, which is a multi-dimensional analog of the familiar line search technique. Let Y^i be the $N \times K$ matrix

$$Y^i := \begin{pmatrix} \delta x_1^i & 0 & \cdots & 0 \\ 0 & \delta x_2^i & \cdots & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & \cdots & \delta x_K^i \end{pmatrix}$$

so that the kth column of Y^i corresponds to the kth block of the update δx^i. The coordination problem is that of finding w^i that *approximately* solves

$$\text{(8)} \qquad \underset{w}{\text{minimize}} f(x^i + Y^i w, \tau, \theta) \text{ subject to } \mathbf{0} \leq x^i + Y^i w \leq u.$$

We will then choose $x^{i+1} = x^i + Y^i w^i$ for our next iterate. One may see that the block structure of Y^i means that (8) is a K dimensional optimization problem with simple bound constraints. Note that $x^{i+1} \in \mathcal{B}$ if $x^i \in \mathcal{B}$. Also, since a search procedure is used in the coordination (8), it is easy to enforce $x^{i+1} \in \text{int}\,\mathcal{C}$ if $x^i \in \text{int}\,\mathcal{C}$. Details of how approximate solutions of (8) are computed may be found in [SM90]. For a theoretical discussion of load balancing alternatives for this type of method, see [SM89].

4. Numerical Results for the PDS Problems. This section presents some preliminary computational experience for a set of large multicommodity network flow problems. The test set is the set of Patient Distribution System problems developed at the Military Airlift Command at Scott Air Force Base. Some of the smaller test problems in this set have been seen in the literature before. For example, in [CHK+89] the KORBX system was used to solve problems as large as PDS-20 in approximately 18 hours. The sizes of PDS problems in this report are given in table 1.

Table 2 shows the cumulative time at the end of each of the three phases of the method (starting after the problem data was read in). We do a maximum of 50 iterations before stopping the code. Comparison with a colleague [De 90] shows that the objective functions match the optimal objective functions in at least five digits for PDS-5, PDS-10 and PDS-20. We do not have accurate lower bounds for the remaining problems. The sequence $\{\theta^i\}$ was specified by setting $\Theta = 1$ and $\lambda_\theta = 0.9$. Smaller values of λ_θ typically led to doing more iterations in the feasibility phase. The sequence $\{\tau^i\}$ was specified by setting $\tau^0 = 10$, $\lambda_\tau = 0.5$ and $\tau_{\text{inf}} = 10^{-8}/J$. In our computations, the costs are normalized so that $\|c\|_\infty = 1$. Thus $\tau^0 = 10$ is ten times the maximum cost coefficient, and at the beginning the barrier is much more important than the objective function. With this setting of τ_{inf} we had $\tau^{50} > \tau_{\text{inf}}$. The trust region was specified by

Problem	K	$\max_k M(k)$	$\max_k N(k)$	J	$M+J$ (rows)	N (cols)
PDS-05	11	686	2,149	553	8,099	23,639
PDS-10	11	1,399	4,433	1,169	16,558	48,763
PDS-20	11	2,857	10,116	2,447	33,874	105,728
PDS-40	11	5,652	20,698	4,672	66,844	212,859
PDS-60	11	8,423	31,474	6,778	99,431	329,643

TABLE 1
Sizes of the PDS problems in this report.

using $\gamma = 0.7$ and $\gamma_{\inf} = 10^{-8}$. Interestingly, the smaller $\gamma_{\inf} > 0$ was, the better the empirical convergence rate was.

The runs were done on two machines: a DECstation 3100 and a 20 processor Sequent Symmetry machine. All computations were done in double precision. The portion of the code that solves the subproblems is a network simplex code written in FORTRAN (a modification of the RNET code of [GH79]). The balance of the code was written in C. More than 80% of the computation time is spent solving subproblems. All times are wall clock time.

We parallelized the subproblem solution. The speedup $\mathcal{S}(\mathcal{P})$ on $\mathcal{P}$ processors of the Sequent Symmetry is measured as a ratio of wall clock times. Actual speedups observed for PDS-05 on the Sequent Symmetry were $\mathcal{S}(2) = 1.67$, $\mathcal{S}(3) = 2.25$, $\mathcal{S}(4) = 2.69$, $\mathcal{S}(6) = 3.42$ and $\mathcal{S}(11) = 4.61$. Where we have done profiles, the parallelized portions of the code account for between 80% and 90% of the total computational work. Assuming 80% of the work is done in parallel, it is reasonable to expect $\mathcal{S}(11)$ to be bounded by $11/3 \approx 3.67$. If 90% of the work is done in parallel, the same estimate gives the a bound of $11/2 = 5.5$. In light of all this, the speedup results obtained look reasonable. Possibilities of overlapping subproblem solution with coordination and using additional combinations of updates Y^i are discussed in [SM89].

REFERENCES

[CHK+89] Major William J. Carolan, Major James E. Hill, Jeffery L. Kennington, Second Lieutenant Sandra Niemi, and Captain Stephen J. Wichmann. An empirical evaluation of the KORBX algorithms for military airlift applications. Technical Report 89-OR-06, Southern Methodist University, May 1989.

[De 90] Renato De Leone. Personal Communication, 1990.

[FM68] Anthony V. Fiacco and Garth P. McCormick. *Nonlinear Programming: Sequential Unconstrained Minimization Techniques.* John Wiley and Sons, 1968.

[GH79] Michael D. Grigoriadis and Tau Hsu. *RNET, The Rutgers Minimum Cost Network Flow Subroutines, Users Documentation.* Department of Computer Science, Hill Center for the Mathematical Sciences, Rutgers University—Busch Campus, New Brunswick, New Jersey 08903, 3.6 edition, October 1979.

[McC83] Garth P. McCormick. *Nonlinear Programming, Theory, Algorithms, and Applications.* John Wiley and Sons, 1983.

[SM89] Gary L. Schultz and Robert R. Meyer. A flexible parallel algorithm for block-constrained optimization problems. In Ramesh Sharda, *et al*, editors, *Impacts of Recent Computer Advances on Operations Research*, New York, 1989. North Holland.

[SM90] Gary L. Schultz and Robert R. Meyer. Forthcoming technical report, 1990.

PDS-05			
phase	relaxed	feasible	final
total iterations	0	15	50
objective $\times 10^{-10}$	2.78244	2.81280	2.805406
DECstation	14sec	4min	14min
Sequent(1)	31sec	9min26sec	36min
Sequent(2)	16sec	5min24sec	22min
Sequent(3)	11sec	3min49sec	16min
Sequent(4)	8sec	3min5sec	13min32sec
Sequent(6)	6sec	2min18sec	10min38sec
Sequent(8)	6sec	2min12sec	10min15sec
Sequent(11)	4sec	1min32sec	8min

PDS-10			
phase	relaxed	feasible	final
total iterations	0	16	50
objective $\times 10^{-10}$	2.63334	2.68523	2.672724
DECstation	32sec	9min	32min
Sequent(11)	9sec	4min	17min30sec

PDS-20			
phase	relaxed	feasible	final
total iterations	0	15	50
objective $\times 10^{-10}$	2.33420	2.40617	2.382209
DECstation	2min11sec	35min	2hr17min
Sequent(11)	48sec	11min23sec	1hr

PDS-40			
phase	relaxed	feasible	final
total iterations	0	12	50
objective $\times 10^{-10}$	1.71878	1.94745	1.887724
DECstation	9min	2hr12min	10hr36min
Sequent(11)	4min30sec	35min	3hr50min

PDS-60			
phase	relaxed	feasible	final
total iterations	0	12	50
objective $\times 10^{-10}$	1.21595	1.52171	1.436928
DECstation	19min	5hr	25hr
Sequent(11)	7min	1hr29min	9hr

TABLE 2
Timing results for solution of selected Patient Distribution System problems.