SYNCHRONOUS AND ASYNCHRONOUS MULTI-COORDINATION METHODS FOR THE SOLUTION OF BLOCK-ANGULAR PROGRAMS

R.R. MEYER[†] AND G. ZAKERI[‡]

Abstract. Several types of multi-coordination methods for block-angular programs are considered. We present a computational comparison of synchronous multi-coordination methods. The most efficient of these approaches is shown to involve an intermediate number of blocks in the coordination phase. We also develop a new stabilization algorithm and present asynchronous multi-coordination schemes, which are particularly useful when the number of blocks exceeds the number of available processors or when the block sizes vary significantly.

1. Introduction. In this paper we present multi-coordinator synchronous and asynchronous solution methods for the block-angular problem BAP:

$$\begin{array}{llllll}
\min_x & c(x) & & & & \\
\text{s/t} & A_{[1]}x_{[1]} & & & = & b_{[1]} \\
 & & A_{[2]}x_{[2]} & & = & b_{[2]} \\
 & & & \ddots & & \\
 & & & A_{[K]}x_{[K]} & = & b_{[K]} \\
 & & & D(x) & \leq & d
\end{array}$$

$$\mathbf{0} \leq x \leq u$$

We assume that functions c and D are convex and at least once continuously differentiable. The $x_{[i]}$s are blocks of variables and $A_{[i]}$s are matrices which in the case of multicommodity network flow problems will be node-arc incidence matrices. We assume the upper bounds $u_{[i]}$ are finite. The blocks $x_{[i]}$ are only coupled together through the J mutual constraints $D(x) \leq d$. Previously in [4] we presented multi-coordination schemes for the solution of the BAP using barrier decomposition. Here we remind the reader of those methods and present new results as well as a new stabilization algorithm. We also present asynchronous multi-coordination methods for the BAP.

2. Review of Synchronous Multi-Coordination Methods. Schultz and Meyer (in [10]) developed specialized barrier methods for the solution of the BAP. In order to solve the BAP we solve barrier problems (denoted BP) of the form:

$$\begin{array}{lll}
\underset{x}{\text{minimize}} f(x) = c(x) + \tau\rho(d - D(x)) & & \\
\text{subject to} \quad A_{[k]}x_{[k]} = b_{[k]} & & k = 1, \ldots, K \\
\mathbf{0} \leq x \leq u & &
\end{array} \tag{2.1}$$

*This research was partially funded by National Science Foundation grants CDA–9024618 and CCR–9306807 and Air Force Office of Scientific Research grant F 49620–94–1–0036.

[†]Center for Parallel Optimization, Computer Sciences Department, University of Wisconsin–Madison, 1210 West Dayton Street, Madison, WI 53706

[‡]Operations Research Group, Department of Engineering Science, Auckland University, Private Bag 92019, Auckland, New Zealand

where ρ is a convex barrier function (see [10]). To obtain an interior feasible point for BP we (approximately) solve a sequence of shifted barrier problems (denoted SBP) of the form:

$$\begin{aligned} \text{(2.2)} \qquad & \underset{x}{\text{minimize}} c(x) + \tau^i \rho(\theta^i - D(x)) \\ & \text{subject to} \quad A_{[k]} x_{[k]} = b_{[k]} \qquad k = 1, \ldots, K \\ & \mathbf{0} \leq x \leq u \end{aligned}$$

The value of θ^1 is determined by:

$$\text{(2.3)} \qquad \theta_j^1 = \begin{cases} d_j & \text{if } D_{j\cdot}(x^0) < d_j \\ D_{j\cdot}(x^0) + \Theta & \text{if } D_{j\cdot}(x^0) \geq d_j \end{cases}$$

where $\Theta > 0$ is a constant. In later iterations we vary θ^i according to:

$$\text{(2.4)} \qquad \theta_j^{i+1} = \begin{cases} d_j & \text{if } D_{j\cdot}(x^i) < d_j \\ \lambda_\theta D_{j\cdot}(x^i) + (1 - \lambda_\theta)\theta_j^i & \text{if } D_{j\cdot}(x^0) \geq d_j \end{cases}$$

We (approximately) solve the BP and the SBP using an iterative fork-join scheme. At iteration t we first find search directions by constructing $\overline{\mathcal{R}}(x^t)$, a type of trust region around x^t, and solving the problem:

$$\begin{aligned} \text{(2.5)} \qquad & \text{minimize} \nabla f(x^t)(y - x^t) \\ & \text{subject to} \quad Ay = b \\ & \mathbf{0} \leq y \leq \overline{\mathcal{R}}(x^t) \end{aligned}$$

Let $\mathcal{B} := \{x | Ax = b \text{ and } \mathbf{0} \leq x \leq u\}$. We require $\overline{\mathcal{R}}(x^t)$ to be a continuous function of x^t satisfying that for any $x \in \mathcal{B}$, $\mathbf{0} \leq x \leq \overline{\mathcal{R}}(x^t) \leq u$. Define a decoupled resource allocation for $\mathcal{B}$ by

$$\mathcal{R}(x^T) := \{z \in R^n | \mathbf{0} \leq z \leq \overline{\mathcal{R}}(x^t)\}$$

$\overline{\mathcal{R}}(x^t)$ is constructed so that:

- For any bounded sequence $\{x^t\} \subseteq \mathcal{B}$, $\bigcup_{t=0}^{\infty} \mathcal{R}(x^t)$ is bounded. This condition is referred to as the boundedness of resource allocation.
- For any $z \in \mathcal{B}$ and any bounded sequence $\{x^t\} \subseteq \mathcal{B}$ with

  $$\alpha^t := \max\{\alpha | 0 \leq \alpha \leq 1 \text{ and } x^t + \alpha(z - x^t) \in \mathcal{R}(x^t)\}$$

  we have $\liminf_{t \to \infty} \alpha^t > 0$. This condition ensures that we can always take a step in any feasible search direction.

For more details on the decoupled resource allocation see [10]. Problem (2.5) can be decomposed into K independent subproblems of the form:

$$\begin{aligned} \text{(2.6)} \qquad & \text{minimize} \nabla f(x^t)_{[k]}(y_{[k]} - x^t_{[k]}) \\ & \text{subject to} \quad A_{[k]} y_{[k]} = b_{[k]} \\ & \mathbf{0} \leq y_{[k]} \leq \overline{\mathcal{R}}(x^t)_{[k]}, \end{aligned}$$

one for each block k (which is assigned to processor k).

Once the search directions $y^t_{[k]}$ are determined we need to assign stepsizes to be taken in each direction. This was originally done by Schultz and Meyer (in [10]) using a complex coordinator. We, however, perform this step using multiple simpler coordinators hence taking advantage of parallelism. The single-variable, group and block-plus-group multi-coordination methods are discussed extensively in [4]. We will briefly review them now.

2.1. Single-Variable Multi-Coordination (SVMC). Here, once each processor has obtained the search direction for its corresponding block from (2.5), it solves the following single-variable coordinator problem:

$$\begin{array}{lll} (2.7) & \underset{w_k \in \Re}{\text{minimize}} f(x_{[1]}^t, \ldots, x_{[k-1]}^t, x_{[k]}^t + (y_{[k]}^t - x_{[k]}^t) w_k, x_{[k+1]}^t, \ldots, x_{[K]}^t) & \\ & \text{subject to} \quad \mathbf{0} \leq x_{[k]}^t + (y_{[k]}^t - x_{[k]}^t) w_k \leq u_{[k]} & \end{array}$$

Let w_k^* be the optimal solution to the above, then define

$$x_{[k]}^{t+\frac{1}{2}} = x_{[k]}^t + (y_{[k]}^t - x_{[k]}^t) w_k^*.$$

We then find the coordinator with the least objective at time t (we will refer to the index of such coordinator as $c(t)$). This amounts to a simple pass through objective values of the coordinators. Now the new iterate is determined by:

$$x_{[k]}^{t+1} = \begin{cases} x_{[k]}^{t+\frac{1}{2}} & \text{if } k = c(t) \\ x_{[k]}^t & \text{otherwise} \end{cases}$$

The above coordination scheme is simple and fast and highly parallelizable, however at each iteration only one block of the variables get updated.

2.2. Group Multi-Coordination. In group multi-coordination each processor, rather than being responsible for only one block of variables, is responsible for several blocks. Suppose $\Gamma_p \subset \{1, \ldots, K\}$ is the set of blocks coordinator p is responsible for, then the group coordinator problem for processor p is given by:

$$\begin{array}{lll} (2.8) & \underset{w \in \Re^K}{\text{minimize}} f(x_{[1]}^t + (y_{[1]}^t - x_{[1]}^t) w_1, \ldots, x_{[K]}^t + (y_{[K]}^t - x_{[K]}^t) w_K) & \\ & \text{subject to} \quad \mathbf{0} \leq x_{[k]}^t + (y_{[k]}^t - x_{[k]}^t) w_k \leq u_{[k]} \quad k \in \Gamma_p & \\ (2.9) & \qquad w_j = 0, \quad j \notin \Gamma_p. & \end{array}$$

Here again we look for the coordinator with the least objective (coordinator $c(t)$). The new iterate is obtained by updates in the blocks that are members of $\Gamma_{c(t)}$. The limiting cases are $|\Gamma_p| = 1$, corresponding to single-variable case and $|\Gamma_p| = K$, corresponding to the original Schultz-Meyer method. The computational experience below indicates that intermediate values lead to the most efficient implementation. The convergence proofs for the multi-coordination methods given above (as well as a third method called block-plus-group multi-coordination) can be found in [4].

3. Computational Results. In this section we present computational results from the implementation of single-variable and group multi-coordination schemes for linear BAP's. Two sets of test problems are discussed: 1) the well-known PDS problems that arise from a logistic application, and 2) some randomly generated problems. In the last part of this section we present our computational results for each of the above mentioned schemes and compare them to those of De Leone [4], Zenios and Pinar [8], McBride and Mamer [5], Schultz and Meyer [10] and Grigoriadis and Khachiyan [2].

3.1. Parallel Implementation. Our algorithm follows the basic three-phase method of Schultz and Meyer. Figure (3.1) presents a sketch of the three-phase method. Although we use the same structure as the three-phase method, we generate approximate solutions of the shifted barrier problem using single-variable or group multi-coordination schemes. The algorithm for approximately solving (SBP) using single-variable multi-coordination is presented in (3.2) and the group multi-coordination scheme is presented in figure (3.3).

We implemented our code on the Thinking Machines Corporation's **Connection Machine CM-5**. The machine runs the **CMOST 7.3** operating system and we used the **CMMD** message passing library for inter-processor communication. In the single-variable coordination scheme very little inter-processor communication is necessary since each processor uses only the search direction it produced to generate the sv-coordinator problem. In fact the only inter-processor communication in this algorithm takes place when we determine the candidate with the least coordinator objective (see step 3 of (3.2)). This is done using the `reduce` command which very efficiently searches all processors for a minimum such value.

The group coordination scheme calls for more communication. Each processor, needs to know the search directions for the blocks in its group. We could have used the `scan` command available from the **CMMD** library so that each node would only get the information regarding the group assigned to it. However we found the `concat` command more flexible and efficient. The `scan` command would require many more passes through the processors than the `concat` command. Had we used the `scan` command, there would be $|\gamma| - 1$ passes (where γ is the group denoted Γ_p in Group Multi-Coordination section) through the processors to distribute the necessary information. We used `concat` to give every processor information about the current search direction on every node (the entire vector y rather than $y_{[k]}$ for some blocks k).

3.2. Parameter Values. The algorithm terminates if the maximum subproblem objective norm was less than the parameter spobj-tol or if the number of iterations reaches 100 for the PDS problems or 300 for the MNETGEN problems. The code achieved at least six digits of accuracy in the optimal objective (compared with the previous results obtained by Schultz and Meyer). We scaled the cost coefficients so that $||c||_\infty = 1$. The remaining parameter values were as follows:

- $\lambda_\theta = 0.95$ is the parameter used in (2.4) to produce a sequence of shifted barriers which converge to the original barrier.
- $\tau^0 = 100$ is the initial value of the penalty parameter.
- $\tau_{\text{inf}} = 10^{-6}$ is the minimum possible value for the penalty parameter.
- $\tau_{\text{fact}} \in (0.25, 0.4)$ is the factor by which we reduced the penalty parameter (for the larger problems we used a greater value of τ_{fact} which corresponds to a more gradual decrease of the penalty parameter).
- spobjtol $= 10^{-6}$ is the bound on the subproblems' optimality gap.

3.3. The PDS Problems. We chose the Patient Distribution System (PDS) problems because these are real-world problems and quite a few recently developed methods (including Schultz-Meyer) have used the PDS problems as bench marks. The PDS model is a logistics model designed to help make decisions about patient evacuation. PDS-x denotes the problem that models a scenario lasting x days. Table (3.1) lists the sizes of the problems we considered and the reader can observe that the size of PDS-x is essentially a linear function of x. PDS problems are linear multicommodity network flow problems with 11 commodities. The column labeled *max node* gives the maximum number of nodes for any commodity and the column labeled *max arc* presents the maximum number of arcs for any commodity. The last two columns in the table present the size of the problem when considered as an LP. The column labeled *total constr.* contains the total number of node constraints plus

Assume that the parameters

$$\Theta > 0 \lambda_\theta \in (0,1)$$
$$\tau^1 > 0 \ \ \tau_{\text{inf}} > 0 \ \ \lambda_\tau \in (0,1)$$

are given. Also define:

$$\mathcal{B} = \{x| \quad Ax = b, \qquad 0 \le x \le u\}$$

$$\mathcal{C} = \{x| \quad Dx \le d\}$$

RELAXED PHASE
$i = 0$
Compute x^0 as the solution of the "relaxed" problem; this is BAP without the coupling constraints
If we determine that $\mathcal{B} = \emptyset$ then quit
Set θ^1 as in (2.3)
If $x^0 \in \mathcal{C}^\circ$
Then go to the refine phase
Otherwise go to the feasibility phase
FEASIBILITY PHASE
$i = i + 1$
Generate x^i as an approximate solution of the ith SBP
Set θ^{i+1} as in (2.4)
$\tau^{i+1} = \tau^i$
If $x^i \in \mathcal{B} \cap \mathcal{C}^\circ$
Then go to the refine phase
Otherwise repeat the feasibility phase
REFINE PHASE $i = i + 1$
Set $\theta^i = d$
Generate x^i as an approximate solution of the BP with penalty parameter τ^i
$\tau^{i+1} = \max(\tau_{\text{inf}}, \lambda_\tau \tau^i)$
Repeat the refine phase until x^i is "suitable"

FIG. 3.1. *The three-phase method*

the number of mutual constraints. The column labeled *total var.* contains the total number of variables.

The block constraint matrices for these block-angular problems are node-arc incidence matrices. We take advantage of this fact in our code and use a very efficient network flow solver NSM [11] to solve the subproblems. NSM uses the network simplex method to solve the subproblems. Since the upper bounds on the subproblems are changed from each iteration to the next (we adjust the decoupled resource allocation) we can not use "hot starting". That is, we need to start with an all-artificial basis at every iteration.

We used the optimization package MINOS [7] in the form of a subroutine (MINOS 5.4) in order to solve the coordinator problems for both single-variable and group coordination. MINOS solves the above using a reduced-gradient algorithm in

To generate an approximate solution of the SBP using **single-variable** multi-coordination:

1. Solve the K linear subproblems:

$$\begin{aligned} &\min_{y_{[k]}} \nabla f(x^t)_{[k]}(y_{[k]} - x^t_{[k]}) \\ \text{such that} \quad & A_{[k]} y_{[k]} = b_{[k]} \\ & \mathbf{0} \le y_{[k]} \le \overline{\mathcal{R}}(x^t)_{[k]} \end{aligned}$$

(where $\overline{\mathcal{R}}(x^t)_{[k]}$ is the decoupled resource allocation at x^t) for each block k and let $y^t_{[k]}$ be the optimal solution of the k^{th} subproblem. Define:

$$(y^t_k)_{[i]} = \begin{cases} y^t_{[k]} & \text{if } i = k \\ x^t_{[k]} & \text{otherwise} \end{cases}$$

2. Solve the K single-variable coordinator problems:

$$\min_{w_k \in \Re} f(x^t + (y^t_k - x^t)w_k) \text{ subject to } \mathbf{0} \le x^t + (y^t_k - x^t)w_k \le u$$

and set $x^{t,k} = x^t + (y^t_k - x^t)w^*_k$ where w^*_k is the optimal solution of the coordinator.
3. Choose $x^{t+1} = x^{t,c(t)}$ where $c(t)$ is the index of the block that produces the least objective for the barrier problem.

FIG. 3.2. *Inner iteration for single-variable multi-coordination*

conjunction with a quasi-Newton algorithm. MINOS requires any domain constraint for the objective function to be specified explicitly. Therefore we had to put in linear constraints that imposed lower bounds on the arguments of each log term involved in the barrier function. This procedure amounted to imposing upper and lower bounds on w when using single-variable multi-coordination. In the group multi-coordination case however we required J (recall J is the number of coupling constraints) linear constraints for each coordinator problem which is quite significant for large problems. We used the default parameters except for the feasibility and optimality tolerance which were set to 10^{-11} in the single-variable case and to 10^{-6} in the group case.

3.4. Analysis of the Results.

3.4.1. Single-variable multi-coordination. In table (3.2) we present the solution results for the PDS problems we tested using single-variable multi-coordination method. In table (3.2) the first column, labeled *feas*, contains the number of outer iterations for the feasibility phase (there were 2 inner iterations per outer iteration). During the feasibility phase, from one outer iteration to the next the variable θ was changed as in (2.4). The *opt* column contains the number of additional outer iterations to optimality. We changed the value of the penalty parameter τ by the relation $\tau^{i+1} = \max(\tau_{\text{fact}} * \tau^i, \tau_{\text{inf}})$. This column is followed by the *inner* column which contains the number of inner iterations per outer iteration in the optimality phase.

To generate an approximate solution of the SBP using **group** multi-coordination:

1. Solve the linear subproblem:

$$\begin{aligned} &\min_{y_{[k]}} \nabla f(x^t)_{[k]}(y_{[k]} - x^t_{[k]}) \\ \text{such that} \quad &A_{[k]} y_{[k]} = b_{[k]} \\ &\mathbf{0} \leq y_{[k]} \leq \overline{\mathcal{R}}(x^t)_{[k]} \end{aligned}$$

(where $\overline{\mathcal{R}}(x^t)_{[k]}$ is the decoupled resource allocation at x^t) for each block k

2. Let γ_k be the group of blocks assigned to node k. We will drop the k subscript for ease of notation. Each group coordinator problem is of the form:

$$\min_{w} f(x^t + Y^t_\gamma w) \quad \text{subject to} \quad 0 \leq x^t + Y^t_\gamma w \leq u \ .$$

where Y^t_γ is the matrix of search directions for blocks in γ.
Set $x^{t,k} = x^t + Y^t_\gamma w^*$ where w^* is the optimal solution of above.

3. Choose $x^{t+1} = x^{t,c(t)}$ where $c(t)$ is the index of the group that produces the least objective for the barrier problem.

FIG. 3.3. *Inner iteration for group multi-coordination*

problem	*max node*	*max arc*	*coupling*	*total constr.*	*total var.*
pds.1	126	339	87	1,473	3,729
pds.2	252	685	181	2,953	7,535
pds.3	390	1117	303	4,593	12,287
pds.5	686	2,149	553	8,099	23,639
pds.10	1,399	4,433	1,169	16,558	48,763
pds.20	2,857	10,116	2,447	33,874	105,728
pds.30	4,223	15,126	3,491	49,944	154,998
pds.40	5,652	20,698	4,672	66,844	212,859

TABLE 3.1
Sizes of the PDS problems tested

Next is the column labeled *rlx time*. This is the time it took to solve the most time consuming subproblem in the relaxed phase. Column *coor time* contains the sum of the times it took to solve the coordinators that took longest to be solved. In column *sub time* we summed the time it took to solve the subproblems that took longest to solve in each iteration. The column labeled *total* contains the total time taken to solve the corresponding PDS problem and *com* is the percentage of time spent on communication. The communication overhead is calculated by subtracting the time it took to solve the relaxed problem and the subproblem and coordination time from the total time and then dividing this by the total time. We used asterisks to indicate the times were too small to extract a meaningful communications percentage.

problem	*feas*	*opt*	*inner*	*rlx time*	*coor time*	*sub time*	*total*	*com*
pds.1	5	15	2	0.02	0.77	0.63	2.4	*
pds.2	5	15	2	0.04	0.90	1.49	3.7	*
pds.3	5	18	2	0.07	1.68	3.15	7.1	*
pds.5	5	23	2	0.23	3.19	9.14	16.9	*
pds.10	5	23	2	0.65	6.13	28.61	45.11	20
pds.20	5	23	2	4.6	12.56	191.91	252.84	17
pds.30	5	20	4	13.09	33.65	754.36	901.72	11
pds.40	5	20	4	36.57	45.75	1404.14	1806.44	17

TABLE 3.2
Solution using single-variable multi-coordination

problem	*sub time*	*coor time*	*total*	*group size*	*% communication*
pds.10	29	6	45	1	20
pds.10	20	43	102	3	37
pds.20	192	13	253	1	17
pds.20	141	135	355	3	21
pds.30	754	34	902	1	11
pds.30	447	182	800	3	20
pds.30	308	191	654	5	22
pds.30	484	647	1273	7	11
pds.30	569	647	1359	11	12
pds.40	1404	46	1806	1	17
pds.40	1124	189	1704	3	21
pds.40	897	199	1434	5	21
pds.40	1009	666	1980	7	14
pds.40	1254	926	2370	11	7

TABLE 3.3
Comparison of single-variable and group multi-coordination

3.4.2. Group multi-coordination. Table (3.3) contains information about the solution of large PDS problems using the group multi-coordination scheme. We also tried the group multi-coordination method on small PDS problems. The results for the small problems were not competitive with those found in the single- variable implementation. In table (3.3) the columns are labeled much the same as columns in table (3.2). The only new column is *group size*. This column contains the number of variables that we picked as the group size for each problem for the optimality phase.

In the group multi-coordination, the feasibility phase was still implemented using single-variable coordinators. As shown in table (3.2) a feasible point was obtained very efficiently using single-variable coordinators. Table (3.3) shows that for problems smaller than pds-30, we don't gain anything from group multi-coordination. However for the larger PDS problems (e.g. PDS-30 and PDS-40) we can save time (26-27% speed up) if we choose the appropriate group size. The best size in these cases is 5, smaller than the full size coordinator of 11 used in Schultz-Meyer. Figure (3.4) plots the solution time for the large PDS problems using single-variable multi-coordination vs using group multi-coordination.

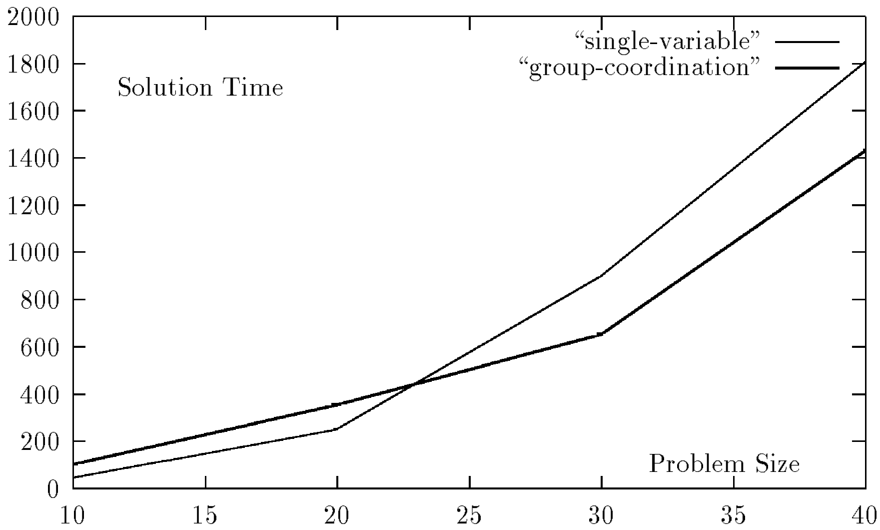

FIG. 3.4. *Single-variable multi-coordination vs. group multi-coordination*

problem	*SM*	*ZP*	*GK*	*MM*	*MC*
pds.10	1711	408	123	40	45
pds.20	7920	1947	372	427	253
pds.30	19380	7504	756	838	654
pds.40	37620	–	1448	4517	1434

TABLE 3.4
Time comparison with other solution methods

3.4.3. Comparison. In table (3.4) we compare the best of our results (column labeled *MC*) with other reported results on the same set of problems. The first column gives the timing results obtained by Shultz and Meyer. They implemented their method on the Sequent Symmetry machine on which each node is 5-10 times slower than the nodes of the connection machine CM5. Results obtained by Zenios and Pinar (in column labeled *ZP*) have been implemented on the Cray-YMP with 8 processors using the vector units hence the processors are 2-4 times faster than the nodes of CM5. Grigoriadis and Khachiyan [2] implemented their algorithm on the IBM RS 6000-550 which is 3 times faster than a node on the CM-5. McBride and Mamer [5] implemented their algorithm on the HP-730 work station which is also three times as fast as a node of the CM5.

3.4.4. The MNETGEN Problems. Another set of problems we considered were those produced by MNETGEN [1], a multicommodity network flow generator which is a derivative of NETGEN [3]. We discovered that the problems produced by this generator contain some mutual constraints that held as equations for any feasible point. Therefore there is no interior for the mutual constraints. Hence we perturbed the right hand side of the mutual constraints by 1 (this perturbation is 0.06-0.25 % of the original right hand side). We anticipated difficulties with these problems due to their random nature and lack of interior. We calculated the objectives to four

Given $0 < \gamma_1 < \gamma_2 < 1$

if $\nabla f^t_{[k]}.(y^t_{[k]} - x^t_{[k]}) \geq 0$
    set $\alpha^t_k = 0$
else
    set $\alpha = 1$
    repeat until explicitly stopped :
(4.1)   if $(f(x^t + \alpha(y^t_k - x^t)) \leq f(x^t) + \gamma_1 \alpha \nabla f^t_{[k]}.(y^t_{[k]} - x^t_{[k]})$
        then $\alpha^t_k = \alpha$ and stop.
        else $\alpha = \gamma_2 \alpha$ and continue.

FIG. 4.1. *Stabilization Algorithm for Single-Variable Multi-Coordination*

digits of accuracy. The largest problem that we solved in this test set was the 200.8 problem with 8 blocks, 200 nodes per block and 449 arcs per block as well as 277 mutual constraints. The most efficient method was group multi-coordination with group size 7, which required about 140 seconds to solve this problem. The overall size of this problem is comparable to the smaller PDS problems, so the performance of the method is not as good on this randomly-generated set as it is on the real-world PDS problems.

4. Stabilization Algorithm for SVMC. We may use the stabilization algorithm outlined in figure 4.1 for approximately solving the single variable coordinator k. Let $x^{t+1,k} = x^t + \alpha^t_k(x^t + y^t_k)$ where x^t and y^t_k are as before and α^t_k is a step-size similar to w_k in section 3.3. Define $j(t)$ to be the index of the block with least subproblem objective. Also define $c(t)$ to be the index of the coordinator with least objective at time t. Then we choose $x^{t+1} = x^{t+1,c(t)}$, so we have

$$f(x^{t+1}) \leq f(x^t) + \gamma_1 \alpha^t_k \nabla f^t_{[k]}.(y^t_{[k]} - x^t_{[k]}) \qquad (\forall k).$$

Recall that $f(x) = c(x) + \tau\rho(d - D(x))$, where ρ is a convex barrier function. Therefore, it is clear that f is a proper convex function for a feasible BAP. In addition, we require f to be essentially smooth. A proper, convex function is said to be essentially smooth (see [9]) if it satisfies the following three conditions for $S = \mathrm{int}(\mathrm{dom} f)$:

- S is non-empty.
- f is differentiable throughout S.
- $\lim_{i\to\infty} |\nabla f(x_i)| = +\infty$ if $\{x_i\}$ is a sequence in S converging to the boundary of S.

We now proceed to prove the convergence of the stabilization algorithm:

THEOREM 1 (convergence of stabilization for SVMC). *Suppose*

- *f is essentially smooth and $\mathcal{B}$ is closed and convex,*
- *$x^0 \in \mathcal{B}$ is given,*
- *the iterates x^t are bounded (one way of insuring this is to require the upper bounds $u < \infty$),*

- y_k^t *be defined blockwise as:*

$$(y_k^t)_{[i]} = \begin{cases} y_{[k]}^t & \textit{if } i = k \\ x_{[i]}^t & \textit{otherwise} \end{cases}$$

- x^{t+1} *and* α^t *chosen according to the stabilization algorithm.*

Then

- *The stabilization algorithm terminates in a finite number of steps,*
- *the svmc algorithm produces a sequence* $\{x^t\}$ *of feasible points,*
- *if* $\tilde{x}$ *is a limit point of* $\{x^t\}$ *then* $\tilde{x}$ *is a KKT point for the barrier problem.*

Proof. We thin the sequence so that:

$$x^t \rightarrow \tilde{x}$$

$$y_{j(t)}^t \rightarrow \tilde{y}$$

1. For each coordinator k, the stabilization algorithm terminates in a finite number of steps with $\alpha_k^t \in [0..1]$:
We have

$$f(x^t + \alpha(y_k^t - x^t)) - f(x^t) - \alpha \nabla f_{[k]}^t.(y_{[k]}^t - x_{[k]}^t) \in O(\alpha^2)$$

therefore for α small enough, the stabilization condition (4.1) is satisfied. We know $\gamma_1 < 1$ hence α can get as small as desired.
2. We start with $x^0 \in \mathcal{B}$ and $\alpha \in [0..1]$ and y_k^t feasible for the coordinator, hence we maintain feasibility from x^t to x^{t+1}.
3. Here we want to prove that using stabilization algorithm, if $\tilde{x}$ is a limit point of the sequence x^t, then $\tilde{x}$ is a KKT point for the barrier problem:
Our algorithm ensures $f(x^{t+1}) \leq f(x^t)$, so either $f(x^t) \rightarrow -\infty$ in which case the problem is unbounded or $\lim_{t\rightarrow\infty} f(x^{t+1}) - f(x^t) = 0$. In the latter case (4.1) shows:

$$\liminf_{t\rightarrow\infty} \alpha_{j(t)}^t \nabla f_{[j(t)]}^t (y_{[j(t)]}^t - x_{[j(t)]}^t) \geq 0$$

but

$$\nabla f_{[j(t)]}^t.(y_{[j(t)]}^t - x_{[j(t)]}^t) = \nabla f^t.(y_{j(t)}^t - x^t) \quad \text{ by def of } y_{j(t)}^t.$$

Hence $\liminf_{t\rightarrow\infty} \alpha_{j(t)}^t \nabla f^t.(y_{j(t)}^t - x^t) \geq 0$.
Now if $\alpha_{j(t)}^t$ is bounded away from zero we have:

$$\liminf_{t\rightarrow\infty} \nabla f^t.(y_{j(t)}^t - x^t) \geq 0$$

and the theorem is proved using the sufficiency of search directions lemma. So consider $\alpha_{j(t)}^t$ for which a subsequence $\alpha_{j(\sigma(t))}^{\sigma(t)} \rightarrow 0$ (we'll drop the subscript σ from here on). Let $f(x^{t+1,k})$ be the objective of the kth coordinator at time $t+1$, then from the stabilization algorithm and the fact that $\alpha_{j(t)}^t \rightarrow 0$ we have:

$$f(x^t + \frac{\alpha_{j(t)}^t}{\gamma_2}(y_{j(t)}^t - x^t)) > f(x^t) + \frac{\gamma_1}{\gamma_2}\alpha_{j(t)}^t \nabla f^t (y_{j(t)}^t - x^t)$$

(otherwise we would have terminated with $\alpha = \frac{\alpha^t_{j(t)}}{\gamma_2}$.) Hence

$$\frac{f(x^t + \frac{\alpha^t_{j(t)}}{\gamma_2}(y^t_{j(t)} - x^t)) - f(x^t)}{\alpha^t_{j(t)}} > \frac{\gamma_1}{\gamma_2}\nabla f^t.(y^t_{j(t)} - x^t)$$

Since $\alpha^t_{j(t)} \to 0$ we get:

$$\nabla f(\tilde{x}).(\tilde{y} - \tilde{x}) \geq \gamma_1 \nabla f(\tilde{x}).(\tilde{y} - \tilde{x})$$

So we have $\nabla f(\tilde{x}).(\tilde{y} - \tilde{x}) \geq 0$ and hence the claim is proved. □

5. Asynchronous Decomposition Methods. This section is devoted to asynchronous solution methods, which are useful when $K > P$ where K is the number of blocks and P is the number of available processors.

5.1. Description of the Method. In the previous sections we described iterative methods that found search directions for each block k, then used these directions to produce a new iterate (single variable or group multi-coordination). If $K > P$, then it is not possible to assign one block per processor to compute the search directions. Therefore, either every processor will have to solve several subproblems or we must use a limited number of search directions in the coordinator at each step. Since the first alternative will introduce a serial bottleneck, we have developed an algorithm for the second alternative. Let $n = \lceil K/P \rceil$, then we propose the following algorithm for the parallel solution of the multi-commodity network flow problem when $K > P$:

Initialize Divide the blocks into $n-1$ batches $(0..n-2)$ each of size P and one batch of size $n.P - K$ (i.e. the sets $\{qP, qP+1, \cdots, (q+1)P - 1\}$ for $q \in (0..n-2)$ and the set $\{qP, qP+1, \cdots, K\}$ for $q = n-1$). Set $t = 0$.

Repeat until convergence: **1.** $q := t \bmod n$

2. Solve (in parallel) the subproblems for blocks in batch q.

3. Apply the stabilization algorithm (see figure (5.1))to approximately solve the multi-coordinators (in parallel).

4. $t := t + 1$.

It is evident from the above algorithm that new information is calculated for each batch of blocks every n iterations. The function $\eta(k,t)$ given below represents (when $t \geq n$) the last time, on or before t, when block k was processed:

$$\eta(k,t) = \begin{cases} -1 & \text{if } t < \lfloor k/P \rfloor \\ \lfloor \frac{k}{P} \rfloor + n.\lfloor \frac{t - \lfloor \frac{k}{P} \rfloor}{n} \rfloor & \text{o.w.} \end{cases} \tag{5.1}$$

When $t < \lfloor k/P \rfloor$ then block k has not been processed for the first time yet and this is indicated by setting $\eta(k,t) = -1$. Notice that:

$$\frac{t - \lfloor k/P \rfloor}{n} - \lfloor \frac{t - \lfloor \frac{k}{P} \rfloor}{n} \rfloor \leq 1$$

therefore

$$t - \lfloor \frac{k}{P} \rfloor - n.\lfloor \frac{t - \lfloor \frac{k}{P} \rfloor}{n} \rfloor \leq n$$

hence

$$t - \eta(k,t) \leq n \quad \text{if } t \geq \lfloor k/P \rfloor.$$

$\eta \backslash t$	0	1	2	3	4	5	6	7	8	9	10
$\eta(0-3,t)$	0	0	0	3	3	3	6	6	6	9	9
$\eta(4-7,t)$	-1	1	1	1	4	4	4	7	7	7	10
$\eta(8-11,t)$	-1	-1	2	2	2	5	5	5	8	8	8

TABLE 5.1
Table of values for the culling function for $P = 4$ and $K = 12$

$\Gamma_{p,q} \backslash t$	0	1	2	3	4	...
$\Gamma(0,q)$	{0,1}	{4,5}	{8,9}	{0,1}	{4,5}	...
$\Gamma(1,q)$	{0,1,2}	{4,5,6}	{8,9,10}	{0,1,2}	{4,5,6}	...
$\Gamma(2,q)$	{1,2,3}	{5,6,7}	{9,10,11}	{1,2,3}	{5,6,7}	...
$\Gamma(3,q)$	{2,3}	{6,7}	{10,11}	{2,3}	{6,7}	...

TABLE 5.2
Table of $\Gamma_{p,q}$ for group size 3.

On the other hand if $t < \lfloor k/P \rfloor$ then

$$t - \eta(k,t) = t + 1 < \lfloor k/P \rfloor + 1 \leq \lfloor K/P \rfloor \leq n$$

Therefore $t - \eta(k,t) \leq n$ for all t. Table (5.1) is a table of $\eta(k,t)$ for 11 iterations ($t = 0, \ldots, 10$), for the case when $P = 4$, and $K = 12$. As table (5.1) indicates, the most recent search direction considered for block 5 at time 9 is the same information as that obtained at iteration 7. Note that at any given time t, there is one set of blocks which has up-to-date information.

We use the notation $\Gamma_{p,q}$ for the group of blocks used in the coordination step (see figure (5.1)), on processor p at time t where $t \bmod n = q$. Let $s \geq 1$, be the chosen maximal group size and for simplicity, we assume that s is odd. Then we define $\Gamma_{p,q}$ by:

$$\Gamma_{p,q} = \{k | k = q.P + p \pm i \quad \text{where } 0 \leq i \leq \frac{s-1}{2}, i \in Z, \text{ and } qP \leq k \leq (q+1)P - 1\}$$

Tables (5.2) and (5.3) present $\Gamma_{p,q}$ for $K = 12, P = 4$ and group sizes 3 and 1. Note that

$$\bigcup_p \Gamma_{p,q} = \{qP, qP+1, \cdots, (q+1)P - 1\}$$

which is the same as the batch of blocks processed at time t where $t \bmod n = q$.

Define Y^t as:

$$Y^t = \begin{pmatrix} y^t_{[1]} - x^t_{[1]} & \cdots & \mathbf{0} \\ \vdots & \ddots & \mathbf{0} \\ \mathbf{0} & \cdots & y^t_{[K]} - x^t_{[K]} \end{pmatrix} \tag{5.2}$$

Let p be the index of our processor and $\hat{w}^t_p$ be the solution to the following:

$$\begin{aligned} & \min \nabla f(x^t) Y^t w + w' H^t w \\ & \text{subject to } w_{[k]} = \mathbf{0} \quad \text{if } k \notin \Gamma_{p,q} \\ & \mathbf{0} \leq x^t + Y^t w \leq u \end{aligned} \tag{5.3}$$

$\Gamma_{p,q} \backslash t$	0	1	2	3	4	...
$\Gamma(0,q)$	$\{0\}$	$\{4\}$	$\{8\}$	$\{0\}$	$\{4\}$	...
$\Gamma(1,q)$	$\{1\}$	$\{5\}$	$\{9\}$	$\{1\}$	$\{5\}$	...
$\Gamma(2,q)$	$\{2\}$	$\{6\}$	$\{10\}$	$\{2\}$	$\{6\}$	...
$\Gamma(3,q)$	$\{3\}$	$\{7\}$	$\{11\}$	$\{3\}$	$\{7\}$	...

TABLE 5.3
Table of $\Gamma_{p,q}$ for group size 1.

Given $0 < \gamma_1 < \gamma_2 < 1$, if $\nabla f(x^t) Y^t \hat{w}_p^t \geq 0$ then set $\alpha_p^t = 0$, else:

Set $\alpha = 1$

Repeat

if $f(x^t + \alpha Y^t \hat{w}_p^t) - f(x^t)$

$\leq \gamma_1 [\alpha \nabla f(x^t) Y^t \hat{w}_p^t - ||\alpha Y^t \hat{w}_p^t||^2]$

then choose $\alpha_p^t = \alpha$ and stop

else reset $\alpha = \gamma_2.\alpha$ and continue.

On processor p, the new iterate $x^{t+1,p} = x^t + \alpha_p^t Y^t \hat{w}_p^t$.

FIG. 5.1. *Stabilization Algorithm on processor p for Asynchronous group multi- coordination*

Here we require H^t to be symmetric positive definite and assume:

$$(\exists \beta) \quad \text{such that} \quad (\forall w) \quad |w' H^t w| \leq \beta ||w||^2.$$

Figure (5.1) contains the algorithm we propose for solving the group coordinator on processor p. This algorithm is a two step coordination scheme.

Let p_t be the index of the processor which produced the best (least) solution to (5.3) at time t and $c(t)$ the index of the processor with best (least) coordinator objective $(f(x^{t+1,p}))$. We choose the next iterate $x^{t+1} = x^{t+1,c(t)}$ and $\alpha^t = \alpha^t_{c(t)}$.

Note that for all k such that $k \notin \Gamma_{p,q}$ we have $w_{[k]} = 0$ therefore at each iteration t we only consider search directions for the blocks in $\Gamma_{p,q}$ which are precisely the search directions calculated this iteration (i.e. those k such that $\eta(k,t) = t$).

Also note that if $\nabla f(x^t) Y^t \hat{w}_{p_t}^t \geq 0$ then $\nabla f(x^t) Y^t \hat{w}_p^t \geq 0$ as well, for any other p. [Observe that $\nabla f(x^t) Y^t \hat{w}_{p_t}^t \geq 0$ implies that the solution to (5.3) is nonnegative for all p. Now if there exists a p such that $\nabla f(x^t) Y^t \hat{w}_p^t = -\xi < 0$ then choose $0 < \lambda < 1$ such that $\lambda < \frac{\xi}{(\hat{w}_p^t)' H^t (\hat{w}_p^t)}$. $\lambda \hat{w}_p^t$ then will be feasible for (5.3) with a lower (negative) objective value than that produced by $\hat{w}_p^t$ which is the optimal solution.]

5.2. Lemmas. LEMMA 5.1. $\alpha^t = 0$ *if and only if* $\alpha_p^t = 0 \quad (\forall p)$.

Proof. Let p be an arbitrary but fixed processor index. If $\alpha^t = 0$ then

$$f(x^{t+1}) = f(x^t) \leq f(x^{t+1,p}) \leq f(x^t) + \gamma_1 [\alpha_p^t \nabla f(x^t) Y^t \hat{w}_p^t - ||\alpha_p^t Y^t \hat{w}_p^t||^2].$$

So if $\alpha_p^t > 0$ then the right hand side is less than $f(x^t)$ and this yields a contradiction. Proof for the other direction is trivial since $\alpha^t = \alpha_p^t$ for some p. ☐

LEMMA 5.2. $\alpha^t = 0$ *iff* $\alpha^t_{p_t} = 0$.

Proof. If $\alpha^t = 0$ then $\alpha^t_{p_t} = 0$ by lemma 5.1. Suppose $\alpha^t_{p_t} = 0$, therefore we must have had $\nabla f(x^t) Y^t \hat{w}^t_{p_t} \geq 0$. This would imply $\nabla f(x^t) Y^t \hat{w}^t_p \geq 0$ for all p, (as noted above) therefore $\alpha^t = 0$ and the lemma is proved. ◻

LEMMA 5.3. *Suppose* $(\forall k \in \Gamma) \quad z^t_k \leq 0$ *but let* Γ *be a finite set of integers and* $\liminf_{t\to\infty} \sum_{k\in\Gamma} z^t_k \geq 0$, *then* $(\forall k \in \Gamma) \liminf_{t\to\infty} z^t_k = 0$.

Proof. Suppose that there exists r such that $\liminf_{t\to\infty} z^t_r < 0$. Now we know:

$$\sum_{k\in\Gamma} z^t_l \leq z^t_r$$

(since all the terms in the sum are nonpositive). So taking lim inf of both sides we obtain:

$$\liminf_{t\to\infty} \sum_{k\in\Gamma} z^t_l \leq \liminf_{t\to\infty} z^t_r < 0$$

which is a contradiction. ◻

LEMMA 5.4. *Suppose we have*

- $\lim_{t\to\infty} x^{t+1} - x^t = \mathbf{0}$, *and*
- $\liminf_{t\to\infty} \nabla f(x^t) Y^t \hat{w}^t_p \geq 0$ *for a given* p.

Then for any q, *if* $k \in \Gamma_{p,q}$, *then* $\liminf_{t\to\infty} \nabla f(x^t)_{[k]} . (y^t_{[k]} - x^t_{[k]}) \geq 0$.

Proof. Let q be arbitrary but fixed. We will first prove that

$$(\exists T \text{ such that } \forall t > T, \text{ if } t \bmod n = q) \quad \nabla f(x^t)_{[k]} . (y^t_{[k]} - x^t_{[k]}) > -\varepsilon$$

for any $\epsilon > 0$ and for any $k \in \Gamma_{p,q}$. This is equivalent to proving the lemma for the subsequence given by $t \bmod n = q$. We then use this result to prove the lemma in the general case.

First we will show that

$$\liminf_{t\to\infty} \sum_{k\in\Gamma_{p,q}} \nabla f(x^t)_{[k]} (y^t_{[k]} - x^t_{[k]}) \geq 0 \tag{5.4}$$

for the sequence $(t \bmod n = q)$.

Assume on the contrary. Then there exists $s(t)$, a subsequence of the sequence given by t such that $t \bmod n = q$ for which:

$$\lim_{t\to\infty} \sum_{k\in\Gamma_{p,q}} \nabla f(x^{s(t)})_{[k]} . (y^{s(t)}_{[k]} - x^{s(t)}_{[k]}) = -\xi < 0$$

for some $\xi > 0$ (note that such $s(t)$ must exist since the sequence is bounded and since the lim inf is assumed to be negative.) Then thin this sequence (we will drop the new subscript) to get:

$x^{s(t)} \to \tilde{x}$	
$y^{s(t)} \to \tilde{y}$	by the boundedness of resource allocation.
$Y^{s(t)} \to \tilde{Y}$	by definition of Y^t .
$H^{s(t)} \to \tilde{H}$	by boundedness of eigenvalues requirement on $H^{s(t)}$.

Now choose $\lambda_{p,q} \in (0..1]$ so that:

$$\sum_{k\in\Gamma_{p,q}} \lambda_{p,q} \nabla f(\tilde{x})_{[k]} . (\tilde{y}_{[k]} - \tilde{x}_{[k]}) + \lambda^2_{p,q} i_{p,q}{}' \tilde{H} i_{p,q} < \frac{-\lambda_{p,q}\xi}{2}$$

where

$$(i_{p,q})_{[k]} = \begin{cases} \mathbf{1} & \text{if } k \in \Gamma_{p,q} \\ \mathbf{0} & \text{otherwise.} \end{cases}$$

(Note that essentially we need $\lambda_{p,q}(i_{p,q}{}'\tilde{H}i_{p,q}) < \xi/2$.) Since $\lambda_{p,q} \in (0..1]$, $\lambda_{p,q}i_{p,q}$ is feasible for the coordinator on node p, so from definition of $\hat{w}_p^{s(t)}$, for $s(t)$ sufficiently large and in the thinned subsequence, we'll have:

$$\nabla f(x^{s(t)})Y^{s(t)}\hat{w}_p^{s(t)} \leq \nabla f(x^{s(t)})Y^{s(t)}\lambda_{p,q}i_{p,q} + (\lambda_{p,q}i_{p,q})'H^{s(t)}(\lambda_{p,q}i_{p,q}) < \frac{-\lambda_{p,q}\xi}{4} < 0.$$

And this contradicts the second hypothesis so we obtain that for the subsequence given by $t \bmod n = q$:

$$\liminf_{t\to\infty} \sum_{k\in\Gamma_{p,q}} \nabla f(x^t)_{[k]}.(y^t_{[k]} - x^t_{[k]}) \geq 0$$

Also, from the definition of $y^t_{[k]}$, we have that

$$\nabla f(x^t)_{[k]}.(y^t_{[k]} - x^t_{[k]}) \leq 0.$$

Therefore using lemma 5.3 we have that for any $k \in \Gamma_{p,q}$ and for any $\varepsilon > 0$:

$$(\exists T \text{ such that } \forall t > T, \text{ if } t \bmod n = q) \quad \nabla f(x^t)_{[k]}.(y^t_{[k]} - x^t_{[k]}) > -\varepsilon \tag{5.5}$$

Next, we prove (5.5) for all t large enough regardless of their remainder modulo n (q is still fixed however we no longer assume $t \bmod n = q$). Let $k \in \Gamma_{p,q}$ be an arbitrary but fixed block index, and assume on the contrary that there exists a subsequence $\sigma(t)$ such that:

$$\lim_{t\to\infty} \nabla f(x^{\sigma(t)})_{[k]}.(y^{\sigma(t)}_{[k]} - x^{\sigma(t)}_{[k]}) = -\xi < 0$$

for some $\xi > 0$. Let $g(t) = \eta(k, \sigma(t))$, then we will have:

$$g(t) \bmod n = q, \tag{5.6}$$

and

$$|g(t) - \sigma(t)| < n + 1. \tag{5.7}$$

From (5.6) and (5.5) we have that

$$\exists T_g \text{ such that } \forall t > T_g, \quad \nabla f(x^{g(t)})_{[k]}.(y^{g(t)}_{[k]} - x^{g(t)}_{[k]}) \geq -\xi/4$$

and from the assumption on $\sigma(t)$ we have that:

$$\exists T_\sigma \text{ such that } \forall t > T_\sigma, \quad \nabla f(x^{\sigma(t)})_{[k]}.(y^{\sigma(t)}_{[k]} - x^{\sigma(t)}_{[k]}) < -\xi/2.$$

Therefore for $t > \max(T_g, T_\sigma)$,

$$|\nabla f(x^{g(t)})_{[k]}.(y^{g(t)}_{[k]} - x^{g(t)}_{[k]}) - \nabla f(x^{\sigma(t)})_{[k]}.(y^{\sigma(t)}_{[k]} - x^{\sigma(t)}_{[k]})| > \xi/4. \tag{5.8}$$

On the other hand from the hypothesis we have that:

$$\lim_{t\to\infty} x^{t+1} - x^t = \mathbf{0}$$

hence by (5.7)

$$\lim_{t\to\infty} ||x^{g(t)} - x^{\sigma(t)}|| = 0. \tag{5.9}$$

Also recall that $\nabla f(x)$ and $\overline{\mathcal{R}}(x^t)$ are continuous functions of x and for every t we have that any subproblem:

$$\begin{aligned} &\min_{y_{[k]}} \nabla f(x^t)_{[k]}(y_{[k]} - x^t_{[k]}) \\ \text{such that} \quad & A_{[k]}y_{[k]} = b_{[k]} \\ & \mathbf{0} \le y_{[k]} \le \overline{\mathcal{R}}(x^t)_{[k]} \end{aligned}$$

has an optimal solution. Therefore we can invoke theorem (1) of [6] which states that the optimal value of the above LP is a continuous function of x^t, therefore:

$$\lim_{t\to\infty} |\nabla f(x^{g(t)})_{[k]}.(y^{g(t)}_{[k]} - x^{g(t)}_{[k]}) - \nabla f(x^{\sigma(t)})_{[k]}.(y^{\sigma(t)}_{[k]} - x^{\sigma(t)}_{[k]})| = 0$$

which contradicts (5.8), therefore

$$\liminf_{t\to\infty} \nabla f(x^t)_{[k]}.(y^t_{[k]} - x^t_{[k]}) \ge 0$$

for all $q \in \Gamma_{p,q}$, for any q. □

Lemma 5.5. *Suppose we have*

- $\lim_{t\to\infty} x^{t+1} - x^t = \mathbf{0}$, *and*
- $\liminf_{t\to\infty} \nabla f(x^t) Y^t \hat{w}^t_{p_t} \ge 0$.

Then $\liminf_{t\to\infty} \nabla f(x^t)_{[k]}.(y^t_{[k]} - x^t_{[k]}) \ge 0$ *for every block* k.

Proof. As noted in the introduction, the hypothesis $\liminf_{t\to\infty} \nabla f(x^t) Y^t \hat{w}^t_{p_t} \ge 0$ implies that

$$\liminf_{t\to\infty} \nabla f(x^t) Y^t \hat{w}^t_p \ge 0$$

for every processor p. So now apply lemma 5.4 to obtain the result. □

5.3. Convergence Theorem.

Theorem 2. *Suppose that*

1. *f is essentially smooth and $\mathcal{B}$ is closed, bounded and convex.*
2. *An initial feasible point $x^{(0)} \in \mathcal{B}$ is given.*
3. *y^t_k is the optimal solution of the subproblem for block k and the search direction for block k at time t.*
4. *Y^t is defined as in (5.2).*
5. *We use the stabilization algorithm in this section to obtain new x_p iterates from old ones on each processor.*
6. *The new iterate x^t is the x^t_p corresponding to the least coordinator objective amongst all processors used in iteration t.*

Then

1. *the stabilization algorithm terminates in a finite number of steps with $\alpha^t_p \in [0..1]$.*

2. *For the sequence $\{x^t\}$ either $f(x^t) \to -\infty$ or each $\tilde{x}$ that is a limit point of $\{x^t\}$, is also a KKT point of the barrier problem we are minimizing.*

Proof. Since the feasible region is bounded we must have a limit point for the $\{x^t\}$. To prove that for each coordinator p the stabilization algorithm terminates in a finite number of steps with $\alpha_p^t \in [0..1]$ observe that for any given processor p at time t if $\nabla f(x^t) Y^t \hat{w}_p^t \geq 0$ then we terminate with $\alpha_p^t = 0$ and we clearly have finite termination. If on the other hand $\nabla f(x^t) Y^t \hat{w}_p^t < 0$ we have:

$$f(x^{t+1,p}) - f(x^t) - \alpha \nabla f(x^t) Y^t \hat{w}_p^t + ||\alpha Y^t \hat{w}_p^t||^2 \in O(\alpha^2).$$

Now since $0 < \gamma_1 < 1$ for α small enough the stabilization condition will be satisfied hence we exit the algorithm in finite number of steps and with $\alpha_p^t \in [0..1]$.

We maintain feasibility throughout the iterations since $x^{(0)}$ is feasible and each new iterate is a convex combination of the previous iterate and search directions, all of which are feasible for the network constraints and weights are chosen so as not to violate any bound constraint. The algorithm insures $f(x^{t+1}) \leq f(x^t)$ so either $f(x^t) \to -\infty$ in which case the problem is unbounded or $\lim_{t\to\infty} f(x^{t+1}) - f(x^t) = 0$. Since the feasible region is assumed to be bounded we need only consider the latter case.

We shall establish that $\nabla f(x^t) Y^t \hat{w}_{p_t}^t \geq 0$ and that $\lim_{t\to\infty} x^{t+1} - x^t = 0$. Once these results are established, we can apply lemma 5.5 to get $\liminf_{t\to\infty} \nabla f(x^t)_{[k]}(y_{[k]}^t - x_{[k]}^t) \geq 0$. This result along with the sufficiency of search directions lemma will then prove the theorem.

We will partition the sequence $\alpha_{p_t}^t$ into two subsequences:

$$\alpha_{p_t}^{s(t)} = 0 \quad \text{and} \quad \alpha_{p_t}^{v(t)} > 0.$$

For the subsequence given by $s(t)$ we have that $\alpha_{p_t}^{s(t)} = 0$ and hence by lemma 5.2 $\alpha^{s(t)} = 0$ which gives

$$\lim_{t\to\infty} x^{s(t)+1} - x^{s(t)} = 0. \tag{5.10}$$

Also since $\alpha_{p_t}^{s(t)} = 0$ then

$$\lim_{t\to\infty} \nabla f(x^{s(t)})_{[k]}.Y^{s(t)} \hat{w}_{p_t}^{s(t)} \geq 0 \quad (\forall p) \tag{5.11}$$

For the subsequence $v(t)$ and from the stabilization algorithm we have

$$\begin{aligned} f(x^{v(t)+1}) - f(x^{v(t)}) &\leq f(x^{v(t)+1,p_t}) - f(x^{v(t)}) \\ &\leq \gamma_1 \left[\alpha_{p_t}^{v(t)} \nabla f(x^{v(t)}) Y^{v(t)} \hat{w}_{p_t}^{v(t)} - ||\alpha_{p_t}^{v(t)} Y^{v(t)} \hat{w}_{p_t}^{v(t)}||^2\right] \end{aligned}$$

which implies:

$$0 < \gamma_1 \leq \frac{f(x^{v(t)+1}) - f(x^{v(t)})}{\alpha_{p_t}^{v(t)} \nabla f(x^{v(t)}) Y^{v(t)} \hat{w}_{p_t}^{v(t)} - ||\alpha_{p_t}^{v(t)} Y^{v(t)} \hat{w}_{p_t}^{v(t)}||^2}.$$

Now the numerator approaches 0 hence we must have the denominator approaching 0 as well. Since the denominator consists of two negative terms we get:

$$\alpha_{p_t}^{v(t)} \nabla f(x^{v(t)}) Y^{v(t)} \hat{w}_{p_t}^{v(t)} \to 0,$$

and

$$||\alpha_{p_t}^{v(t)} Y^{v(t)} \hat{w}_{p_t}^{v(t)}||^2 \to 0. \tag{5.12}$$

And if we repeat the same argument with $\alpha^{v(t)}$ instead of $\alpha_{p_t}^{v(t)}$ then:

$$||\alpha^{v(t)} Y^{v(t)} \hat{w}_{c(t)}^{v(t)}||^2 \to 0 \tag{5.13}$$

which means

$$x^{v(t)+1} - x^{v(t)} \to 0. \tag{5.14}$$

So using (5.12) and taking lim inf of both sides of the required condition of convergence in the stabilization algorithm we get:

$$\liminf_{t\to\infty} \alpha_{p_t}^{v(t)} \nabla f(x^{v(t)}) Y^{v(t)} \hat{w}_{p_t}^{v(t)} \geq 0.$$

Now for any subsequence $v_1(t)$ of $v(t)$ for which $\liminf_{t\to\infty} \alpha_{p_t}^{v_1(t)} > 0$ then we have

$$\liminf_{t\to\infty} \nabla f(x^{v_1(t)}) Y^{v_1(t)} \hat{w}_{p_t}^{v_1(t)} \geq 0 \tag{5.15}$$

On the other hand, for any subsequence $v_2(t)$ with $\liminf_{t\to\infty} \alpha_{p_t}^{v_2(t)} = 0$ we want to prove the analog of (5.15). So assume on the contrary that:

$$\liminf_{t\to\infty} \nabla f(x^{v_2(t)}) Y^{v_2(t)} \hat{w}_{p_t}^{v_2(t)} < 0$$

Then there must exist a subsequence $\sigma(t)$ of $v_2(t)$ for which

$$\lim_{t\to\infty} \nabla f(x^{\sigma(t)}) Y^{\sigma(t)} \hat{w}_{p_t}^{\sigma(t)} = -\xi < 0$$

Now, from the stabilization algorithm we have that:

$$\frac{f(x^{\sigma(t)} + \frac{\alpha_{p_t}^{\sigma(t)}}{\gamma_2} Y^{\sigma(t)} \hat{w}_{p_t}^{\sigma(t)}) - f(x^{\sigma(t)})}{\alpha_{p_t}^{\sigma(t)}} >$$

$$\frac{\gamma_1}{\gamma_2} \left[\nabla f(x^{\sigma(t)}) Y^{\sigma(t)} \hat{w}_{p_t}^{\sigma(t)} - \frac{\alpha_{p_t}^{\sigma(t)}}{\gamma_2} ||Y^{\sigma(t)} \hat{w}_{p_t}^{\sigma(t)}||^2 \right].$$

Now we thin the sequence $\sigma(t)$ to get:

$$x^{\sigma(t)} \to \tilde{x},$$

$$Y^{\sigma(t)} \to \tilde{Y},$$

and

$$\hat{w}_{p_t}^{\sigma(t)} \to \tilde{w}_j.$$

Taking limits from both sides of the inequality will give:

$$\frac{1}{\gamma_2} \nabla \tilde{f}.(\tilde{Y} \tilde{w}_j) \geq \frac{\gamma_1}{\gamma_2} \nabla \tilde{f}.(\tilde{Y} \tilde{w}_j).$$

Hence:

$$\liminf_{t\to\infty} \nabla f(x^{\sigma(t)}) Y^{\sigma(t)} \hat{w}_{p_t}^{\sigma(t)} \geq 0$$

which is a contradiction. Therefore we must have:

$$\liminf_{t\to\infty} \nabla f(x^{v_2(t)}) Y^{v_2(t)} \hat{w}_{p_t}^{v_2(t)} \geq 0 \tag{5.16}$$

From (5.15) and (5.16) we get:

$$\liminf_{t\to\infty} \nabla f(x^{v(t)}) Y^{v(t)} \hat{w}_{p_t}^{v(t)} \geq 0 \tag{5.17}$$

Now we can put together (5.10), (5.11), (5.14), and (5.17) to conclude:

$$\liminf_{t\to\infty} x^{t+1} - x^t = 0$$

and

$$\liminf_{t\to\infty} \nabla f(x^t) Y^t \hat{w}_{p_t}^t \geq 0$$

which together with lemma 5.5 imply:

$$\liminf_{t\to\infty} \nabla f(x^t)_{[k]}(y_{[k]}^t - x_{[k]}^t) \geq 0.$$

Therefore we can apply the sufficiency of search directions which delivers the conclusion. $\square$

6. Conclusion. We have developed synchronous and asynchronous multi-coordination schemes for the solution of the block-angular program. These multi-coordination schemes are highly paralellizeable. We presented numerical results which showed the efficiency of the synchronous single-variable and group multi-coordination schemes. The results demonstrate significant improvement over the Schultz-Meyer predecessor and are at least comparable with the best of other solution methods.

REFERENCES

[1] A. Ali and J. Kennington, *MNETGEN program documentation*, Tech. Report IEOR 77003, Department of Industrial Engineering and Operations Research, Southern Methodist University, Dallas, Texas 75275, 1977.

[2] M. Grigoriadis and L. Khachiyan, *An exponential-function reduction method for block-angular convex programs*, Tech. Report 211, Laboratory for Computer Sciences Research, Rutgers, Computer Sciences Dept., May 1993.

[3] D. Klingman, A. Napier, and J. Stutz, *NETGEN—A program for generation of large-scale (un)capacitated assignment, transportation and minimum cost network problems*, Management Science, 20 (1974), pp. 814–822.

[4] R. D. Leone, R. Meyer, S. Kontogiorgis, A. Zakarian, and G. Zakeri, *Coordination in coarse grained decomposition*, SIAM Journal on Optimization, 4 (1994), pp. 777–793.

[5] R. D. McBride and J. W. Mamer, *Solving multicommodity flow problems with a primal embedded network simplex algorithm.* submitted.

[6] R. Meyer, Tech. Report 373, Computer Sciences Department, University of Wisconsin.

[7] B. Murtagh and M. Saunders, *MINOS 5.4 release notes, appendix to MINOS 5.1 user's guide*, technical report, Stanford University, 1992.

[8] M. Pinar and S. Zenios, *Parallel decomposition of multicommodity network flows using a linear-quadratic penalty algorithm*, ORSA Journal on Computing, 4 (1992), pp. 235–249.

[9] R. Rockafellar, *Convex Analysis*, Princeton University Press, Princeton, NJ, 1970.

[10] G. Schultz, *Barrier Decomposition for the parallel optimization of block-angular programs*, PhD thesis, University of Wisconsin-Madison, 1991.

[11] A. Zakarian. Private communication.