HG CP 94–0820–CICT–5–2

Parallel implementation of DD-BSR-MILU preconditioner for 3-D linear elasticity problems on symmetric multiprocessor computers

Krassimir GEORGIEV
Bulgarian Academy of Sciences, Sofia, Bulgaria

The symmetric multiprocessor (SMP) computers are computers with shared memory but it is possible to simulate distributed memory architecture and to use the standard Message Passing Interface (MPI) in its version for distributed memory computers such as clusters of workstations. The only difference is that the network parameters β (start-up time) and α (time for transferring of single word) of the clusters are quite larger. The numerical experiments reported bellow are obtained using two SMP: SUN Enterprise 3000, located now in the Department of Mathematics of the University of Nijmegen, an workstation with 8×167 MHz Ultra SPARC processors and 1 Gb main memory and SUN SPARC 20 located now in the CICT-Sofia, an workstation with 4 $\times$ 99 MHz processors and 128 Mb main memory.

After discretizing the system of differential equations of the 3D linear elasticity problem by threelinear parallelepipedal finite elements the large system of linear equations (1) is obtained which has to be solved in order to find the displacements $\mathbf{U} = [U_i]^T$.

$$K\,\mathbf{U} \;=\; \mathbf{F}, \tag{1}$$

The system (1) is solved by using the standard Preconditioned Conjugate Gradient (PCG) procedure via an additive Domain Decomposition-Block Size Reduction-Modified Incomplete (DD-BSR-MILU) preconditioner. The computational domain Ω is first divided into non-overlapping substructures $\Omega_i, i = 1, 2, ..., N$ with diameters of the order H. All the substructures Ω_i are further divided into parallelepipeds with characteristic diameters of order h. We extend each subdomain Ω_i to a larger region $\widehat{\Omega}_i$ [4].

Let $\widehat{K_i}$ be a local stiffness matrix according to the subdomain $\widehat{\Omega}_i$. We define restriction maps R_i and extension maps R_i^T, as follows. Let the global stiffness matrix

K is of order n and denote with $\hat{n}_i$ the number of the interior with respect to $\widehat{\Omega}_i$, nodes. For each subdomain $\widehat{\Omega}_i$ the matrix R_i is $(n \times \hat{n}_i)$ restriction matrix with entries 1's and 0's , taking into account the indexes of the nodes belong to $\widehat{\Omega}_i$. This matrix restricts a vector $\mathbf{U}$ of length n to $R_i\mathbf{U}$ of length $\hat{n}_i$. Thus, the local stiffness matrix $\widehat{K_i}$ corresponding to the subdomain $\widehat{\Omega}_i$ is

$$\widehat{K_i} \;=\; R_i\, K\, R_i^T.$$

In the case of many subdomains, the additive Schwartz algorithm is a generalization of the well known two-subdomain case with preconditioner

$$M_{add}^{-1} = \sum_{i=1}^{N} R_i^T\; \widehat{K_i}^{-1}\; R_i.$$

As $R_i^T\; \widehat{K_i}^{-1}\; R_i\; \mathbf{y}$ can be computed in parallel for the different subdomains, this leads to a coarse grain parallelism.

The algorithm for solving the system $M_{add}\mathbf{z}_k \;=\; \mathbf{r}_k$ into the kth CG step is as follows.

- $\widehat{\mathbf{r}}_k^i$ – vector of a length $\hat{n}_i$; entries are equal to those of $\mathbf{r}_k$ for the components corresponding to the meshpoints in $\widehat{\Omega_i}$ and 0 elsewhere;
- $\widehat{\mathbf{z}}_k^i$: $\widehat{K_i}\widehat{\mathbf{z}}_k^i \;=\; \widehat{\mathbf{r}}_k^i$;
- $\mathbf{z}_k^i$ – vector of a length n; an extension of $\widehat{\mathbf{z}}_k^i$ with 0 entries outside $\widehat{\Omega}_i$;
- $\mathbf{z}_k \;=\; \sum_{i=1}^N \mathbf{z}_k^i$

The coefficient matrices of the systems of linear algebraic equations

$$\widehat{K_i}\widehat{\mathbf{z}}_k^i \;=\; \widehat{\mathbf{r}}_k^i \tag{2}$$

are symmetric positive definite block matrices and the preconditioned conjugate gradient method with block–size reduction block–ILU preconditioner proposed by Chan and Vassilevski [2] (see also: [1] and [3]) is used to solve the systems in each subdomain.

The whole computational domain is the parallelepiped

$$\Omega \;=\; [0, x_{max}] \times [0, y_{max}] \times [0, z_{max}].$$

Single pile in a homogeneous sandy clay soil layer is considered. An uniformly load distribution on the cross section of the pile is assumed (Benchmark 1).

When one uses a distributed-memory parallel computer, he maps as a rule the different subdomains to individual processors. The problem which has to be solved is

Table 1: SUN Enterprise 3000

No. proc.	Iter.	Memory	Elapsed time	Speed up (S_p)	Efficiency (E)
1	49	10082 K words	4386		
2	4	6793 K words	5985	0.73	0.37
4	4	5379 K words	1393	3.15	0.79
8	4	3352 K words	685	6.40	0.80

Table 2: SUN SPARC 20

No. proc.	Iter.	Memory	Elapsed time	Speed up (S_p)	Efficiency (E)
1	49	10082 K words	2106		
2	4	6793 K words	2816	0.75	0.37
4	4	5379 K words	832	2.53	0.63

the communications between the different processors. The standard Message Passing Interface (MPI) [5, 6] is used.

At this version of the codes realizing our algorithm we have used:

- **blocking send** (MPI_SEND) and **blocking receive** (MPI_RECV) operations in a **standard mode** from the *Point-to-Point communications*;

- MPI_BARRIER for **barrier synchronization**, MPI_BCAST for **broadcasting** data, MPI_GATHER and MPI_GATHERV (a vector variant of MPI_GATHER) for **gathering** data, MPI_ALLREDUCE for **reduction** operations from the *Collective communications*;

- MPI_WTIME for **timing** (*elapsed time*) the programs or sections of the programs.

An option $-O4$ was used during the compilation stage for the numerical experiments on SUN SPARC 20 while due to some technical reason this option was not used for SUN Enterprise 3000. This fact explains the quite larger times when eight processors SUN Enterprise 3000 was used for runs of the code.

It is well seen that with increasing the number of the subdomains in the domain decomposition algorithm the computational time decreases. Unfortunately, this is not the case when there are only two subdomains. The reaso of that is that when the whole computational domain is devided in two subdomains with an overlapping, the subdomain with the pile is quite large and the number of inner iterations (PCG-BSR-MILU) are only approximately two times less than in the algorithm without subdividing. Taking in account that one needs to performe some domain decomposition iterations (outer iterations) it may be explained the bad time results for this case (row 2 in the two tables). At the same time, if one compares only the results obtained for four and eight processors (see rows 3 and 4 in Table 2) it well seen that the ratio between the elapsed time in the two runs is approximately two, the same as the ratio between the number of the processor using in the two runs.

References

[1] T. Chan, S. Margenov, P. Vassilevski, *Performance of Block-ILU factorization preconditioning techniques for linear elasticity problems*, SIAM J. Sci.Comput. (to appear)

[2] T. Chan, P. Vassilevski, *A framework for block-ILU factorizations using block–size reduction*, Math. Comp. (to appear)

[3] T. Chan, P. Vassilevski, *Convergence analysis of block–ILU factorization algorithms based on block–size reduction*, UCLA CAM Report, Math. Comp. (to appear)

[4] K. Georgiev , *Implementation of the additive overlapping domain decomposition method to 3-D elasticity problems*, **Mathematica Balkanica**, **4**, (1996), (to appear)

[5] W. Gropp, E. Lusk, A. Skjellum, *Using MPI: Portable Parallel Programming with the Message-Passing Interface*, The MIT Press, Cambridge, Massachusetts, Third printing (1996)

[6] M. Snir, St. Otto, St. Huss-Lederman, D. Walker, J. Dongara, *MPI: The Complete Reference*, The MIT Press, Cambridge, Massachusetts, (1996)