Moscow DD Conference

May 1990

Numerical Experiments with a Domain Decomposition Method for Parabolic Problems on Parallel Computers

Gérard Meurant

C E A

Centre d'Etudes de Limeil-Valenton

France

Outline :

1) Problems to be solved

2) Sketch of the method

3) Numerical experiments on parallel machines

Problems to be solved

We are interested into solving linear systems arising from the implicit discretization of linear partial differential parabolic equations like,

$$\frac{\partial u}{\partial t} - \frac{\partial}{\partial x}(a(x,y)\frac{\partial u}{\partial x}) - \frac{\partial}{\partial y}(b(x,y)\frac{\partial u}{\partial y}) = f$$
$$\text{in } \Omega \subset R^d, \quad d = 2$$

$$u\mid_{\partial\Omega} = 0, \quad u(x,0) = u_0(x) = 0$$

The **model problem** will be

$$\frac{\partial u}{\partial t} - \Delta u = f$$

Ω being the unit square

For efficiency and stability, we use an implicit scheme

Crank–Nicolson scheme

$$\frac{\partial u}{\partial t} + Lu = f$$

Finite difference discretization with $n+1$ points on each side of the square

$$h = \frac{1}{n+1}$$

$$L \Longrightarrow \frac{1}{h^2} A$$

A is a block tridiagonal matrix (of the steady state problem)

Time is discretized with a centered scheme : $t \in [0, T]$, step size $k = \Delta t$

$$2\frac{h^2}{k}(u^{m+1} - u^m) + Au^{m+1} + Au^m = h^2(f^{m+1} + f^m)$$

For every time step we have to solve the system

$$(2\frac{h^2}{k}I + A)u^{m+1} = 2\frac{h^2}{k}u^m - Au^m + h^2(f^{m+1} + f^m)$$

The advantages of the implicit scheme are :

– second order accuracy

– unconditional stability

Let $\theta = 2\frac{h^2}{k}$, the matrix of the system is

$$A_t = \theta I + A$$

A_t is a symmetric strictly diagonally dominant M–matrix.

We actually solve

$$A_t x = h^2(f^{m+1} + f^m) - 2A_t u^m,$$

then,

$$u^{m+1} = x + u^m.$$

For the model problem :

$$A_t = \begin{pmatrix} D_t & -I & & & \\ -I & D_t & -I & & \\ & \ddots & \ddots & \ddots & \\ & & -I & D_t & -I \\ & & & -I & D_t \end{pmatrix},$$

$$D_t = \begin{pmatrix} b & -1 & & & \\ -1 & b & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & b & -1 \\ & & & -1 & b \end{pmatrix},$$

$$b = 4 + \theta$$

To solve the system, we use the **Conjugate Gradient method**

$$r^0 = b - A_t x^0,$$

for $k = 0, 1, \ldots$ until convergence,

$$Mz^k = r^k,$$

$$\beta_k = \frac{(r^k, z^k)}{(r^{k-1}, z^{k-1})},\ \beta_0 = 0,$$

$$p^k = z^k + \beta_k p^{k-1},$$

$$\alpha_k = \frac{(r^k, z^k)}{(A_t p^k, p^k)},$$

$$x^{k+1} = x^k + \alpha_k p^k,$$

$$r^{k+1} = r^k - \alpha_k A_t p^k.$$

We are looking for a parallel preconditioner M

Sketch of the method

We use a method which was proposed in a different setting by Y. Kuznetsov :
New algorithms for approximate realization of implicit difference schemes
Sov. J. Numer. Anal. Math. Modelling v 3 n^0 2 (1988)

Main tools :

–1) Superposition of solutions
–2) Decay of elements of inverses

Domain Decomposition method.

Let us first summarize the method with "exact" subdomain solves.

We use boxes (square subdomains). The subproblems are solved with the Linpack band solver.

We implement this algorithm on two parallel machines, Sequent S81 with 8 processors and the CRAY Y–MP with 8 processors.

Numerical experiments

$$\frac{\partial u}{\partial t} - \Delta u = f$$

$$u\,|_{\partial\Omega} = 0, \quad u(x,0) = u_0(x) = 0$$

Ω is the unit square

$$h = \frac{1}{34}, \qquad \Delta t = \frac{1}{60} \qquad T = 0.5$$

$$\theta = 0.104$$

$$u(x,t) = \sin(xt)xy(1-x)(1-y)\exp(xy)$$

$$\text{max error} = 0.11\ 10^{-4}$$

Reduction of the residual in CG : 10^{-2}

Suggested overlapping with $\epsilon = 10^{-2}$ and $a = 4 + \theta : 4$

Upper bound : 12

The Linpack solves are too expensive

Use instead an **Incomplete Cholesky decomposition MIC(0)**

$$M = LDL^T$$

L has the same sparsity pattern as the lower triangular part of A

Sequent 34×34

MIC(0) as a preconditioner

116 iterations 10.5 seconds on 1 processor

diagonal preconditioner

523 iterations 21.6 seconds on 1 processor, 3.7 on 8 proc (speed up 5.8)

MIC(0) with an overlapping of 4

120 iterations 17.6 seconds on 1 processor, 2.4 on 8 proc (speed up 7.3)