行政院國家科學委員會專題研究計畫成果報告

ADI 方法的平行算法之探討
On Investigation of Parallel Methods for Alternating Direction Implicit Scheme

計畫編號：NSC 88-2115-M-018-005-
執行期限：87 年 8 月 1 日至 88 年 7 月 31 日
主持人：周忠強　　國立彰化師範大學數學系

一、中文摘要

本計劃主要是在多處理器上探討二或三維度拋物線型偏微分方程式的數值方法。有限差分法為其主軸，其中包括了交錯方向隱性法及交錯方向顯性法。在多處理器上用交錯方向隱性法求解，必定會遭遇到要使用高斯消去法、空間分割、及資料交換等問題，這些都是我們要去一一克服的問題。目前我們為週期邊界問題設計一個平行算法。如果能善用區域資料和適當的資料傳遞，可以提升交錯方向隱性法的效率。

關鍵詞：拋物線型偏微分方程式、交錯方向隱性法、高斯消去法、多處理器、空間分割

Abstract

We investigate the implementation of several classical methods for solving parabolic equations for higher (2 or 3) dimensions on multiprocessors. The methods considered are the Alternating Direction Implicit (ADI) algorithms and the Alternating Direction Explicit (ADE) methods. We focus on parallel ADI methods which contains Gaussian elimination, domain decomposition, and data communication. Currently, we develop a algorithm for periodic problems. The high efficiency of the method is achieved by carefully using local data and suitable data passing.

Keywords: Parabolic Equations, Alternating Direction Implicit, Gaussian Elimination, Multiprocessors, Domain decomposition.

二、緣由與目的

Today's supercomputers which are designed with multiprocessors and extra large memory make it possible to tackle real world problems. To avoid sequential bottlenecks which is caused by excessive communication requirements, scalable and portable parallel algorithms are required for these powerful machines. In this report, we develop parallel algorithms for the alternating direction implicit methods (ADI) which is devoted for solving parabolic or elliptic partial differential equations with higher dimensions. The ADI methods have to solve many independent relatively small tridiagonal systems of linear equations whose coefficients are distributed over each required processor on the supercomputer. Here we focus our methods for parabolic equations. For convenience, a simple two dimensional problem is discussed in the following.

The initial-boundary value problem is given by:

$$u_t = a_1(x,y)u_{xx} + a_2(x,y)u_{yy}$$

on a square, and initial and boundary value are specified. Note that there is no term with a mixed derivative. If we apply a scheme similar to the Crank-Nicolson for the equation, with

discretization of both spatial derivatives, the scheme is unconditionally stable, but the inverted matrix at each time step is much more difficult to carry out than the tridiagonal matrices encountered in one dimensional problem. The ADI method, which we now present, is a way of reducing two dimensional problem to a succession of one dimensional problem.

Let A_1 and A_2 be linear operators such that

$$A_1 u = a_1(x,y)u_{xx},$$
$$A_2 u = a_2(x,y)u_{yy}.$$

Under careful consideration, it lead to ADI [2,3] methods, the Peaceman-Rachford algorithm [10] and the Douglas-Rachford method [4] are two of the most basic and original schemes.

The Peaceman-Rachford algorithm is formulaed as

$$(I - \frac{k}{2}A_{1h})\tilde{v}^{n+1/2} = (I + \frac{k}{2}A_{2h})v^n$$
$$(I - \frac{k}{2}A_{2h})v^{n+1} = (I + \frac{k}{2}A_{1h})\tilde{v}^{n+1/2}.$$

And the Douglas-Rachford method is constructed as

$$(I - kA_{1h})\tilde{v}^{n+1/2} = (I + kA_{2h})v^n$$
$$(I - kA_{2h})v^{n+1} = \tilde{v}^{n+1/2} - kA_{1h}v^n.$$

In above two methods, apparently, the two steps alternate which direction is implicit and which is explicit. The term ADI applies to any method that involves the reduction of the problem to one dimensional implicit problems by solving the tridiagonal systems. The two methods are unconditionally stable, as is easily seen by von Neumann analysis for two dimensions.

For large size problems, we have to solve the tridiagonal systems on parallel computer. There are many parallel algorithms [1,5,6,9,11] for the solution of tridiagonal linear systems. However, they all are designed for two points boundary problems. Now we have to develop a parallel scheme on periodic boundary problems.

三、結果與討論

ADI methods, the Peaceman-Rachford algorithm or the Douglas-Rachford method, together with periodic conditions, the finite difference scheme at each time step (including the intermediate time steps) produces n's linear systems of equations like

$$AX = B,$$

where X and B are N column vectors and A is a tridiagonal-like $N \times N$ matrix described as

$$\begin{pmatrix} b_1 & c_1 & & & & & & a_1 \\ a_2 & b_2 & c_2 & & & & & \\ & a_3 & b_3 & c_3 & & & & \\ & & \ddots & \ddots & \ddots & & & \\ & & & \ddots & \ddots & \ddots & & \\ & & & & a_{N-2} & b_{N-2} & c_{N-2} & \\ & & & & & a_{N-1} & b_{N-1} & c_{N-1} \\ c_N & & & & & & a_N & b_N \end{pmatrix}$$

Solving this system consists of three steps:

Step1: Forward elimination of the sub-diagonal elements, $a_2,\ldots,a_N$.
Step2: Backward elimination to the super-diagonal elements, $c_{N-1},\ldots,c_1$. At the end of Step 2, solve for the unknowns, $x_2,x_3,\ldots,x_{N-2}$.
Step3: Solve for x_1 and x_N from

$$b_1 x_1 + a_1 x_N = d_1',$$
$$c_N x_1 + b_N x_N = d_N'.$$

The above non-pivoting elimination method for solving the tridiagonal-like system is stable if A is column diagonally dominant.

To solve the tridiagonal-like system in parallel is more complicated. Our parallel solver adapted the ideas in the solver for the standard tridiagonal system. Assume that the system is to be solved by P processor. For simplification, suppose that N is divisible by P.

The parallel solution is obtained from the following steps:

Step1: The first step is evenly distribution of equations to each processor. There are N/P consecutive equations which is assigned to each processor. It means that each processors i take equations numbered form $(i-1)N/P+1$ to iN/P.

Step2: This step is the simultaneous forward eliminations by all processors. On each processor i, the sub-diagonal elements of row $(i-1)N/P+2$ through row iN/P are eliminated. These eliminations on a processor require no information from other processors.

Step3: It is a process of the backward elimination simultaneously. Processor i, $i=2,...,P-1$, eliminates the super-diagonal elements of equations numbered $iN/P-2$ to $(i-1)N/P$. Processor 1 eliminates super-diagonal elements for equations $N/P-2$ to 1, and processor P eliminates super-diagonal elements for equations $N-1$ to $(P-1)N/P$. Since the super-diagonal element in the last equation of processor i, $i=1,2,...,P-1$, must be eliminated by using the equation in processor $i+1$, interprocessor communication is needed for processor i to get the updated row $iN/P+1$.

Step4: Solve that the unknowns $x_1, x_{iN/P}$, $i=1,2,...,P$, in the system satisfy another set of $P+1$ equations like the original tridiagonal-like system. The $P+1$ tridiagonal-like equations are distributed to the P processors. The solution process of these equations by the P processors is mainly sequential. (It may also be processed by one only processor by passing those equations to the working processor.) However, since each processor has only one equation, the solution complexity is independent of the problem size. The major work of this step is to solve the $P+1$ equations by the sequential algorithm on all processors (or one processor).

Step5: Solve that the other unknowns after the unknowns $x_1, x_{iN/P}$, $i=1,2,...,P-1$, have been solved. It is clear that each processor can solve for the other unknowns by subtracting multiples of the fill-in columns from the equations it holds. The computations on different processors can be done in parallel. The only communication step is that the unknown $x_{iN/P}$ calculated at step 4 by processor i, $i=1,2,...,P-1$, has to be passed to processor $i+1$ before the solution process for the rest of unknowns can start.

四、計畫成果自評

Under current designed methods for periodic boundary problems, together with original parallel solver for tridiagonal linear system, we can obtain parallel solutions of higher dimensional parabolic differential equations. However, Mattor *et. Al.* [9] reported that the parallel solver can not reach an efficient speedup on tridiagonal systems. Now we have been trying to develop a new algorithm which uses local data in stead of communicating date. Therefore, the local data generated by explicit methods are required. It has to be much more efficient, but it may losses unconditionally stable property. We will report in near future.

五、參考文獻

[1] Juan C. Agui and Javier Jimenez. A Binary Tree Implementation of a Parallel Distributed Tridiagonal Solver. *Parallel Computing*, 21:233-241, 1995.

[2] J. Douglas Jr. The Solution of the Diffusion Equation by a High Order Correct Difference Equation. *J. Math. Phys.*, 35:145--, 1956.

[3] J. Douglas Jr. ADI Methods for Three Space Variables. *Numer. Math.*, 4:41--, 1962.

[4] J. Douglas Jr and H. H. Rachford Jr. On the numerical solution of heat conduction problems in two or three space variables. *Trans. Amer. Math. Soc.*, 82: 421-439, 1956.

[5] S. L. Johnsson, Y. Saad, and M. H. Schultz. *The Alternating Direction Algorithm on Multiprocessors.* Research Report 382, department of Computer Science, Yale University,1985.

[6] Hong Jiang and Yau Shu Wang. A Parallel Alternating Direction Implicit Preconditioning Method. *J. Comput. Appl. Math.*, 36 : 209-226, 1991.

[7] Arno Krechel, Hans-Joachim Plum, and Klaus Stuben. Parallelization and Vectorization Aspects of the Solution of Tridiagonal Linear Systems. *Parallel Computing*, 14:3-49, 1990.

[8] Y. Lin and C. W. Cryer. An Alternating

Direction Implicit Algorithm for the Solution of Linear Complementarity Problems Arising from Free Boundary Problems. *Appl. Math. Optim.*, 13: 1-17, 1985.

[9] Nathan Mattor, Timothy J. Williams, and Dennis W. Hewett. Algorithm for Solving Tridiagonal Matrix Problems in Parallel. *Parallel Computing*, 21: 1769-1782, 1995.

[10] D. W. Peaceman and H. H. Rachford Jr. The numerical solution of parabolic and elliptic differential equations. *H, Sic. Indust. Appl. Math.*, 3: 27-38, 1955.

[11] U. W. Rathe, P. Sander, and P. L. Knight. A Case Study in Scalability: An ADI Method for the Two-dimensional Time-dependent Dirac Equation. *Parallel Computing*, 25: 525-533, 1999.