PROCEEDINGS OF
THE 2009 INTERNATIONAL CONFERENCE ON
PARALLEL AND DISTRIBUTED PROCESSING TECHNIQUES AND
APPLICATIONS

PDPTA 2009

Volume II

Editor

Hamid R. Arabnia

Associate Editors

Jesus Carretero, Jose Daniel Garcia,
George A. Gravvanis, Minoru Ito, Kazuki Joe
Hiroaki Nishikawa, Ashu M. G. Solo

WORLDCOMP'09
July 13-16, 2009
Las Vegas Nevada, USA
www.world-academy-of-science.org

A Red-Black Gauss Seidel Parallel Algorithm for Solving Neumann Boundary Elliptic PDEs

Huiqing Helen Yang and Zhifu Xie
Mathematics & Computer Science
Virginia State University
Petersburg, VA, USA

ABSTRACT

Partial differential equations play an important role in the physical sciences and engineering. The problem addressed in this paper is to solve Elliptic PDEs with Neumann boundary. The main effort of the paper is placed on investigating a Red-Black Gauss Seidel algorithm which can solve Neumann boundary Elliptic PDEs in parallel. The parallel algorithm is implemented on the SGI Origin 2000. The preliminary numerical results and efficiency analysis are provided.

Keywords: Parallel algorithm, numerical method, Elliptic PDE, Neumann Boundary, Gauss Seidel algorithm

1. Introduction

Partial differential equations (PDEs) provide mathematical models in the modeling physical phenomena. Many physics problem can be described by models which require the solution of partial differential equations. As one type of the PDEs, the elliptic PDEs always occur purely as boundary value problems. There are many different approaches to solve the elliptic PDEs. The three most widely used numerical methods to solve PDEs are the finite difference methods (FDM), finite element methods (FEM) and finite volume methods (FVM). Other methods are including spectral methods, collocation methods, etc. However, numerically solving an elliptic equation with Neumann boundary condition is still a challenging research field. The solutions for the elliptic PDEs with Neumann boundary condition are usually not unique because Neumann boundary condition only gives the derivative conditions along the boundary of the domain. A change in conditions at any point on the boundary will result in the solution change throughout the domain [1].

The applications of the finite difference methods (FDM) to the elliptic differential equations often lead to large system of algebraic equations. For practical interest, the iteration methods, such as Jacobi iteration method, Gauss-Seidel Relaxation (GSR), SOR and so on, are generally used to solve the large system of linear equations. The emergence of parallel computers and their potential for the numerical solution of grand challenge problems has lead to large amount of research in parallel solving PDEs. In this research, we investigated an approach which can parallel solve a Neumann boundary elliptic equation. By means of finite differences, a large system of algebraic equation can be derived.

The paper is organized as follows: the problem is described in section 2. Section 3 shows a finite difference method for a Neumann boundary Elliptic equation. A red-black Gauss Seidel parallel algorithm is provided in section 4. Numerical results and performance analysis are shown in section 5.

2. Problem Statement

PDE problem specification consists of the system of partial differential equations, the domains of definition, boundary conditions, and initial conditions. Elliptic Partial differential equations model time independent problems and often arise when determining the steady-state solutions to time-dependent problems. Elliptic partial differential equations can be specified over nonrectangular, or irregular, domains. However, finite difference method generally does not work well on nonrectangular domains. In this research, we consider an elliptic partial differential equation with Neumann boundary condition in a square domain.

Given

$$-(au_{xx}+bu_{yy})=f(x,y) \tag{1}$$

on the square domain $\Omega=[\alpha_1,\alpha_2]\times[\beta_1,\beta_2]$ subject to the Neumann boundary condition:

$$\begin{cases} u_x = g_1(\alpha_1, y) & when \quad x=\alpha_1 \\ u_x = g_2(\alpha_2, y) & when \quad x=\alpha_2 \\ u_y = g_3(x, \beta_1) & when \quad y=\beta_1 \\ u_y = g_4(x, \beta_2) & when \quad y=\beta_2 \end{cases} \tag{2}$$

where ab>0, and f(x, y) is a given function of x and y.

3. A Finite Difference Method for a Neumann Boundary Elliptic Equation

In this research, a parallel solver decomposes the solution of elliptic PDE into two steps, that is, discretizing the domain and then solving the resulting linear system of equations.

Let us superimpose on the domain $\Omega=[\alpha_1, \alpha_2]\times[\beta_1, \beta_2]$, a mesh of grid points in both the horizontal and vertical directions is illustrated in Figure 1.

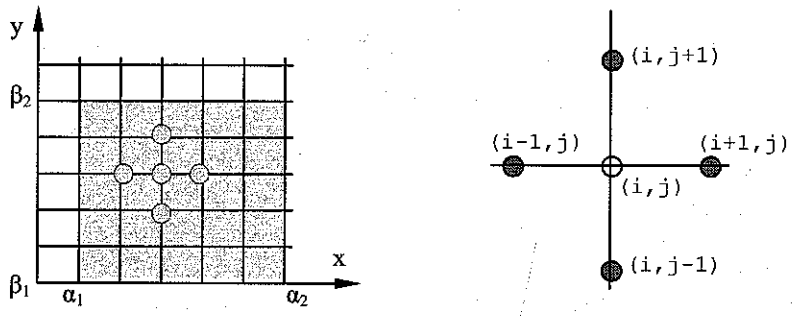

Figure 1

Let $\begin{cases} h_x = \dfrac{\alpha_2-\alpha_1}{N+1} \\ h_y = \dfrac{\beta_2-\beta_1}{N+1} \end{cases}$. Because the domain is a square, $h_x = h_y = h$.

Then the node points can be given by $\begin{cases} x_i = \alpha_1 + ih & where \quad i=0,1,2,\ldots\ldots,N,N+1 \\ y_j = \beta_1 + jh & where \quad j=0,1,2,\ldots\ldots,N,N+1 \end{cases}$

The interior grid points are given by $(x_i, y_j)=(ih, jh)$, simply denoted by

$$(i,j) \quad where \quad i,j=1,2,\ldots\ldots,N.$$

Now, consider a typical grid point (x_j, y_i). We approximate u_{xx} and u_{yy} at this point by the centered difference approximations.

$$u_{xx} = \frac{1}{h^2}[u(x_{j-1}, y_i) - 2u(x_j, y_i) + u(x_{j+1}, y_i)] = \frac{1}{h^2}(u_{j-1,i} - 2u_{j,i} + u_{j+1,i})$$
$$u_{yy} = \frac{1}{h^2}[u(x_j, y_{i-1}) - 2u(x_j, y_i) + u(x_j, y_{i+1})] = \frac{1}{h^2}(u_{j,i-1} - 2u_{j,i} + u_{j,i+1}) \quad (3)$$

Thus, substituting (3) into (1), we have:

$$\frac{1}{h^2}(2(a+b)u_{j,i} - au_{j-1,i} - au_{j+1,i} - bu_{j,i-1} - bu_{j,i+1}) = f(x_j, y_i) \quad (4)$$
$$2(a+b)u_{j,i} - au_{j-1,i} - au_{j+1,i} - bu_{j,i-1} - bu_{j,i+1} = h^2 f_{ji} \quad (5)$$

From the equation (5), we can obtain:

$$\begin{cases} j=1,\ i=1 & 2(a+b)u_{1,1} - au_{0,1} - au_{2,1} - bu_{1,0} - bu_{1,2} = h^2 f_{1,1} \\ j=1,\ i=2 & 2(a+b)u_{1,2} - au_{0,2} - au_{2,2} - bu_{1,1} - bu_{1,3} = h^2 f_{1,2} \\ \quad\vdots & \quad\vdots \\ j=1,\ i=N & 2(a+b)u_{1,N} - au_{0,N} - au_{2,N} - bu_{1,N-1} - bu_{1,N+1} = h^2 f_{1,N} \end{cases}$$

$$\begin{cases} j=2,\ i=1 & 2(a+b)u_{2,1} - au_{1,1} - au_{3,1} - bu_{2,0} - bu_{2,2} = h^2 f_{2,1} \\ j=2,\ i=2 & 2(a+b)u_{2,2} - au_{1,2} - au_{3,2} - bu_{2,1} - bu_{2,3} = h^2 f_{2,2} \\ \quad\vdots & \quad\vdots \\ j=2,\ i=N & 2(a+b)u_{2,N} - au_{1,N} - au_{3,N} - bu_{2,N-1} - bu_{2,N+1} = h^2 f_{2,N} \end{cases}$$
$$\vdots \qquad\qquad \vdots$$

$$\begin{cases} j=N-1,\ i=1 & 2(a+b)u_{N-1,1} - au_{N-2,1} - au_{N1} - bu_{N-1,0} - bu_{N-1,2} = h^2 f_{N-1,1} \\ j=N-1,\ i=2 & 2(a+b)u_{N-1,2} - au_{N-2,2} - au_{N2} - bu_{N-1,1} - bu_{N-1,3} = h^2 f_{N-1,2} \\ \quad\vdots & \quad\vdots \\ j=N-1,\ i=N & 2(a+b)u_{N-1,N} - au_{N-2,N} - au_{NN} - bu_{N-1,N-1} - bu_{N-1,N+1} = h^2 f_{N-1,N} \end{cases}$$

$$\begin{cases} j=N,\ i=1 & 2(a+b)u_{N,1} - au_{N-1,1} - au_{N+1,1} - bu_{N,0} - bu_{N,2} = h^2 f_{N,1} \\ j=N,\ i=2 & 2(a+b)u_{N,2} - au_{N-1,2} - au_{N+1,2} - bu_{N,1} - bu_{N,3} = h^2 f_{N,2} \\ \quad\vdots & \quad\vdots \\ j=N,\ i=N & 2(a+b)u_{N,N} - au_{N-1,N} - au_{N+1,N} - bu_{N,N-1} - bu_{N,N+1} = h^2 f_{N,N} \end{cases}$$

Specifying the derivative values on the boundary, we know that

$u_{0,i}$ where i=1, 2, …, N
$u_{j,0}$ where j=1, 2, …, N
$u_{N+1,i}$ where i=1, 2, …, N
$u_{j,N+1}$ where j=1, 2, …, N

These values are no longer known from the boundary condition at $\partial\Omega$. Instead these values will be the additional unknown; we need other equations that can be derived as follows.

We approximate $u_{xx}(\alpha_1,y)$, $u_{xx}(\alpha_2,y)$ $u_{yy}(x, \beta_1)$ and $u_{xx}(x, \beta_2)$ by:

$u_{xx}(\alpha_1, y) \approx (u_{-1,i} - 2\,u_{0,i} - u_{1,i}) / h^2$ at $x=\alpha_1$ where i=0,1, 2, ……,N+1
$u_{xx}(\alpha_2, y) \approx (u_{N,i} - 2\,u_{N+1,i} - u_{N+2,i}) / h^2$ at $x=\alpha_2$ where i=0,1, 2, ……,N+1
$u_{yy}(x, \beta_1) \approx (u_{j,-1} - 2\,u_{j,0} - u_{j,1}) / h^2$ at $x=\beta_1$ where j=0,1, 2, ……,N+1
$u_{yy}(x, \beta_2) \approx (u_{j,N} - 2\,u_{j,N+1} - u_{j,N+2}) / h^2$ at $x=\beta_1$ where j=0,1, 2, ……,N+1

We use two grid points ±h for outside the domain. Then by the boundary condition we have:

$g_1(\alpha_1, y_i) = u_x(\alpha_1, y_i) \approx (u_{1,i} - u_{-1,i}) / 2h$ at $x=\alpha_1$ where $i=0,1,2,\ldots\ldots,N+1$
$\Rightarrow \quad u_{-1,i} = u_{1,i} - 2h\, g_1(\alpha_1, y_i)$ where $y_i = \beta_1 + ih$; $i=0, 1, 2, \ldots\ldots,N+1$

$g_2(\alpha_2, y_i) = u_x(\alpha_2, y_i) \approx (u_{N+2,i} - u_{N,i}) / 2h$ at $x=\alpha_2$ where $i=0,1,2,\ldots\ldots,N+1$
$\Rightarrow \quad u_{N+2,i} = u_{N,i} + 2h\, g_2(\alpha_2, y_i)$ where $y_i = \beta_1 + ih$; $i=0, 1, 2, \ldots\ldots,N+1$

$g_3(x_j, \beta_1) = u_y(x_j, \alpha_1) \approx (u_{j,1} - u_{j,-1}) / 2h$ at $y=\beta_1$ where $j=0,1,2,\ldots\ldots,N+1$
$\Rightarrow \quad u_{j,-1} = u_{j,1} - 2h\, g_3(x_j, \alpha_1)$ where $x_j = \alpha_1 + jh$; $j=0, 1, 2, \ldots\ldots,N+1$

$g_4(x_j, \beta_2) = u_y(x_j, \alpha_2) \approx (u_{j,N+2} - u_{j,N}) / 2h$ at $y=\beta_2$ where $j=0,1,2,\ldots\ldots,N+1$
$\Rightarrow \quad u_{j,N+2} = u_{j,N} + 2h\, g_4(x_j, \alpha_2)$ where $x_j = \alpha_1 + jh$; $j=0, 1, 2, \ldots\ldots,N+1$

Therefore, $u_{-1,i}$, $u_{N+1,i}$, $u_{j,-1}$ and $u_{j,N+1}$ can be eliminated by the boundary conditions. Finally, (2) can be approximated by:

$$\begin{cases} u_{-1,i}^{(k)} = u_{1,i}^{(k)} - 2h\, g_1(\alpha_1, y_i) \\ u_{N+2,i}^{(k)} = u_{N,i}^{(k)} + 2h\, g_2(\alpha_2, y_i) \\ u_{j,-1}^{(k)} = u_{j,1}^{(k)} - 2h\, g_3(x_j, \alpha_1) \\ u_{j,N+2}^{(k)} = u_{j,N}^{(k)} + 2h\, g_4(x_j, \alpha_2) \end{cases} \tag{6}$$

where i, j =0, 1, 2, ……., N+1.

The equation (1) can be approximated by:

$$u_{j,i}^{(k+1)} = \frac{1}{2(a+b)}\left(a u_{j-1,i}^{(k)} + a u_{j+1,i}^{(k)} + b u_{j,j-1}^{(k)} + b u_{j,i+1}^{(k)} + h^2 f_{ji}\right) \tag{7}$$

where i, j=0, 1, 2, ……., N+1.

Equation (7) will be (N+2)×(N+2) linear system. Use the standard 5-stencil central finite difference formula, i.e.

$$\begin{array}{ccc} & -b & \\ -a & 2(a+b) & -a \\ & -b & \end{array}$$

This will lead to a system involving a large positive definite sparse matrix with at most 5 nonzero elements per row. The sparse system can be solved using an iterative method such as Jacobi, Gauss-Seidel, successive over-relaxation (SOR) and so on. In this research, Gauss Seidel algorithm is applied.

4. A Red-Black Gauss Seidel Algorithm

The natural ordering is the fastest ordering. Assume N=4, the scheme of a nature ordering on a two-dimensional domain is shown below:

	31	32	33	34	35	36
	25	26	27	28	29	30
	19	20	21	22	23	24
	13	14	15	16	17	18
↑	7	8	9	10	11	12
	1	2	3	4	5	6

j i →

Let

Red points (i, j) if i+j is even
Black points (i, j) if i+j is odd

$\Omega_{h,\,red} = \{ (i, j) \mid i+j \text{ is even} \}$
$\Omega_{h,\,black} = \{ (i, j) \mid i+j \text{ is odd} \}$

If Ω_h is collection of all grid points, then $\Omega_h = \Omega_{h,\,red} + \Omega_{h,\,black}$

We can apply red-black ordering to solve the equation (7) in parallel. The following figure shows the scheme of a red-black ordering on a two-dimensional domain.

B	R	B	R	B	R
R	B	R	B	R	B
B	R	B	R	B	R
R	B	R	B	R	B
B	R	B	R	B	R
R	B	R	B	R	B

An iteratio7 method can be given as follows:

Step 1: given an initial guess for all interior grid points $u_{ji}^{(k)} \approx u_{ji}^{*}$ (where i, j=0,1, 2, ……., N, N+1)
Step 2: calculate or modify the boundary condition by using (6)
Step 3: apply Gauss-Shield to $\Omega_{h,\,red}$
Step 4: Exchange data on the ghost boundary
Step 5: apply Gauss-Shield to $\Omega_{h,\,black}$
Step 6: Exchange data on the ghost boundary
Step 6: k+1 → k go back to Step2 until tolerate <0.000001

5. Numerical Results and Performance Analysis

As a case study, let

$$-(au_{xx} + bu_{yy}) = f(x, y)$$

Let $a = b = 1$ and $f(x, y) = -4$ (Poisson's equation). We assume that the domain $\Omega = [0,1] \times [0,1]$.

The Neumann boundary condition:

$$\begin{cases} u_x = 0 & when \quad x = 0 \\ u_x = -2 & when \quad x = 1 \\ u_y = 0 & when \quad y = 0 \\ u_y = -2 & when \quad y = 1 \end{cases}$$

Because the elliptic equation is no real characteristics, there are no curves emanating from the boundary along which disturbances propagate. For a Neumann boundary elliptic equation, a unique solution can not be obtained without specifying the dependent variable at least one point on the boundary. In the case study, a point (1,-4) on the boundary is given.

To ensure a unique solution, a point on the boundary is specified. The algorithm has been implemented. Given problem size N=22 and number of processors, the numerical results are shown in the following table.

Table. RB-GS scheme

# of processors	Running time (secs)
1	12.634
4	5.028
9	4.128

Performance Analysis

Speedup (S) is a measure that captures the relative benefit of solving a problem in parallel. It is defined as the ratio of the best time taken to solve a problem on a single processor (T_1) to the best time required to solve the same problem on a parallel computer with p identical processors (T_p).

$$S = \frac{T_1}{T_p} \tag{8}$$

Efficiency (E) is defined as the average speedup per processor, that is:

$$E = \frac{S}{p} \tag{9}$$

The performance of parallel computing is shown in the following figures.

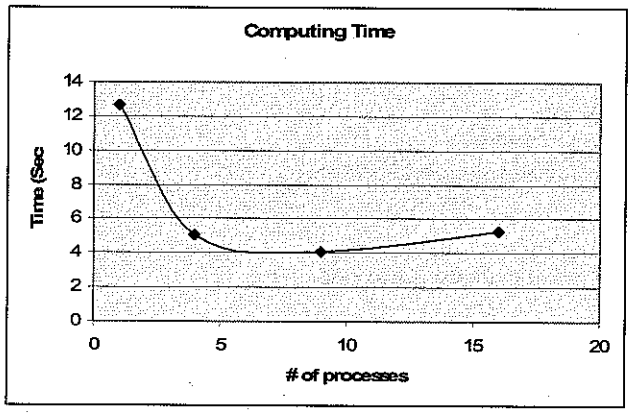

Figure 2. Running time vs the number of processes

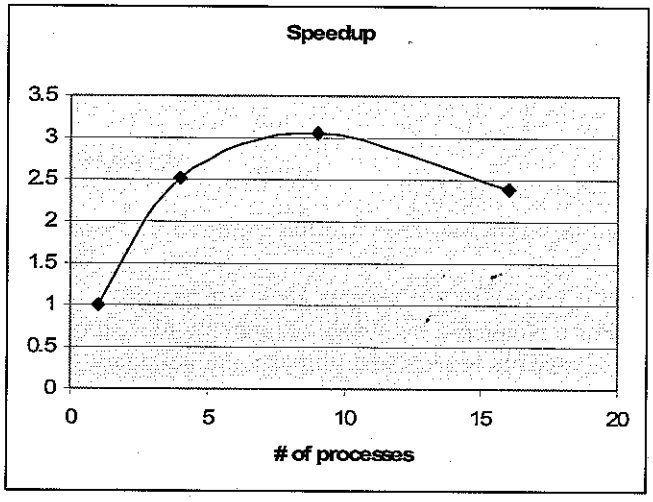

Figure 3. Speedup Performance

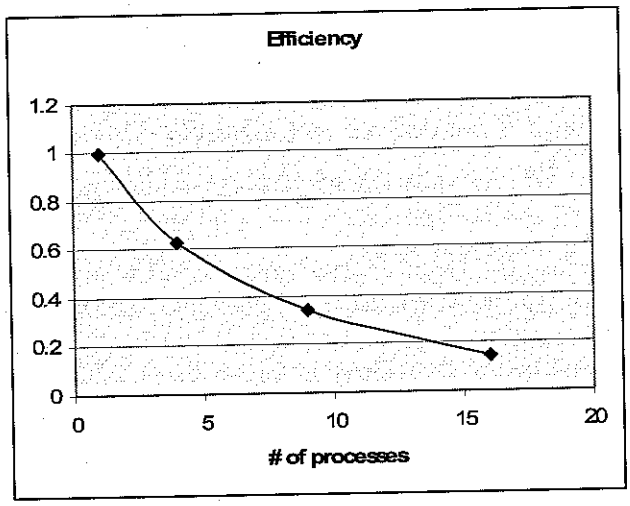

Figure 4. Efficiency

6. Conclusion

Parallel processing methods are a means to achieve significant speedup of computationally expensive algorithms. The investigation shows that

- the performances of parallel algorithms is better the sequential algorithm;
- Red-Black GS scheme improves the performance of Jacobi scheme by using two colors.
- Red-Black GS has the same convergence rate as the GS with the natural ordering for the 5-point stencil.

Sine the Red-Black GS method requires data-exchange on the ghost boundary twice in each iteration, communication is expense on distributed cluster parallel machine. For more general cases, for example, $-(u_{xx}+u_{yy}+u_{xy})=f(x,y)$, with Neumann boundary condition, more colors are needed. This leads to the multi-color Gauss Shield method for our future work.

7. References

[1] Michael A. Celia and William G. Gray, Numerical Methods for Differential Equations – Fundamental Concepts for Scientific and Engineering Application, Prentice-Hall, Inc. A Simon & Schuster Company Englewood Cliffs, New Jersey.

[2] Vipin Kumar, Ananth Grama, Anshul Gupta and George Karypis, "Introduction to Parallel Computing: Design and Analysis of Algorithms", The Benjamin/Cummings Publishing Company, Inc, 1994.

[3] Feras A. Mahmoud and Mohammad H. AI-Towaiq, "Parallel Algorithm for the solutions of PDEs in Linux clustered workstations", Journal of Applied Mathematics and Computation, Vol. 200, Issue 1, 2008.

[4] Michael Renardy and Robert C. Rogers, "An Introduction to Partial Differential Equations", Springer-Verlag, 1993.

[5] Richard E. Crandall, "Topics in Advanced Scientific Computation", Springer-Verlag, 1996.