THE DESIGN OF A NEW EXPLICIT CONVECTION-DIFFUSION SOLVER ON A DISTRIBUTED PARALLEL SYSTEM

[1]Lim Lay Ngor, [2]Norhashidah Hj. Mohd. Ali, [3]Rosni Abdullah,
[4]Chiew Seen Kit, [5]Yong Low Yee Sheng
[1,2,4,5]School of Mathematical Sciences, Universiti Sains Malaysia, 11800 USM, Pulau Pinang
[3]School of Computer Sciences, Universiti Sains Malaysia, 11800 USM, Pulau Pinang

Abstract. *In this paper, we extend the group explici method pioneered by Yousif and Evans (1986) to the solution of parabolic partial differential equation (p.d.e.) specifically the two dimensional convection-diffusion equation. We shall also investigate the parallel design of this iterative scheme intended for a message passing environment. The computational implementation of the parallel strategy on a cluster of distributed computers at the Parallel Computing Lab, Dept. of Computer Science, USM, will be described and discussed.*

1 Introduction

Many physical phenomena in ecological and scientific problems can be modeled by equations that relate several partial derivatives of physical quantities such as forces, momentums, velocities, energy, temperature etc. Some of these problems involve a combination of *diffusion* and *convection* phenomena which are modeled by the *convection-diffusion* equation or commonly known as the *transport* equation:

$$\frac{\partial U}{\partial t} = \alpha_x \frac{\partial^2 U}{\partial x^2} + \alpha_y \frac{\partial^2 U}{\partial y^2} - \alpha_x' \frac{\partial U}{\partial x} - \alpha_y' \frac{\partial U}{\partial y} \tag{1.1}$$

in which α_x, α_y, α_x', and α_y' are positive constants. In solving this equation on a rectangular grid with spacings $\Delta x = \Delta y = h$ in both directions x and y, with $x_i = x_0 + ih$, $y_j = y_0 + jh$ $(i,j = 0,1,2,\ldots,n)$ one can use the implicit finite difference scheme based on the centred difference in time and space formulation about the point $(i,j,k+\frac{1}{2})$ commonly known as the Crank Nicolson scheme, which transforms (1.1) into

$$\begin{aligned}\frac{u_{i,j,k+1}-u_{i,j,k}}{\Delta t} &= \frac{1}{2}\alpha_x\left\{\frac{u_{i-1,j,k+1}-2u_{i,j,k+1}+u_{i+1,j,k+1}}{\Delta x^2}+\frac{u_{i-1,j,k}-2u_{i,j,k}+u_{i+1,j,k}}{\Delta x^2}\right\}\\ &+\frac{1}{2}\alpha_y\left\{\frac{u_{i,j-1,k+1}-2u_{i,j,k+1}+u_{i,j+1,k+1}}{\Delta y^2}+\frac{u_{i,j-1,k}-2u_{i,j,k}+u_{i,j+1,k}}{\Delta y^2}\right\}\\ &-\frac{1}{2}\alpha_x'\left\{\frac{u_{i+1,j,k+1}-u_{i-1,j,k+1}}{2\Delta x}+\frac{u_{i+1,j,k}-u_{i-1,j,k}}{2\Delta x}\right\}\\ &-\frac{1}{2}\alpha_y'\left\{\frac{u_{i,j+1,k+1}-u_{i,j-1,k+1}}{2\Delta y}+\frac{u_{i,j+1,k}-u_{i,j-1,k}}{2\Delta y}\right\}\end{aligned} \tag{1.2}$$

Let the Courant and diffusion numbers be

$$c_x = \alpha_x' \frac{\Delta t}{\Delta x}, \qquad s_x = \alpha_x \frac{\Delta t}{\Delta x^2}$$
$$c_y = \alpha_x' \frac{\Delta t}{\Delta x}, \qquad s_y = \alpha_x \frac{\Delta t}{\Delta x^2}$$

so that Equation (1.2) may be simplified to become

$$(1+s_x+s_y)u_{i,j,k+1}+(-\tfrac{1}{2}s_x-\tfrac{1}{4}c_x)u_{i-1,j,k+1}+(-\tfrac{1}{2}s_x+\tfrac{1}{4}c_x)u_{i+1,j,k+1}+(-\tfrac{1}{2}s_y-\tfrac{1}{4}c_y)u_{i,j-1,k+1}$$
$$+(-\tfrac{1}{2}s_y+\tfrac{1}{4}c_y)u_{i,j+1,k+1}=(1-s_x-s_y)u_{i,j,k}+(\tfrac{1}{2}s_x+\tfrac{1}{4}c_x)u_{i-1,j,k}+(\tfrac{1}{2}s_x-\tfrac{1}{4}c_x)u_{i+1,j,k}$$
$$+(\tfrac{1}{2}s_y+\tfrac{1}{4}c_y)u_{i,j-1,k}+(\tfrac{1}{2}s_y-\tfrac{1}{4}c_y)u_{i,j+1,k} \qquad (1.3)$$

The local truncation error for the difference approximation (1.3) can be established as of order $O(h^2+\Delta t^2)$ and is unconditionally stable for all Courant coefficients (c_x, c_y) and diffusion numbers s_x , $s_y \neq 0$ [2]. In this paper, we formulate a group iterative method based on this Crank Nicolson scheme in solving this second order *parabolic* equation. A brief description of the iterative method studied is presented in Section 2. In Section 3 and 4, we discuss the strategy used for parallelising the method and Section 5 presents the results of experiments performed.

2 Formulation of the Explicit Group Method

Assuming n to be odd, the mesh points are grouped in blocks of four points and the centred difference equation (1.3) is applied to each of these points resulting in the following (4x4) system : (here, $u_{ijk}=u(x_i,y_j,t_k)$)

$$\begin{bmatrix} 1+s_x+s_y & -\frac{1}{2}s_x+\frac{1}{4}c_x & 0 & -\frac{1}{2}s_y+\frac{1}{4}c_y \\ -\frac{1}{2}s_x-\frac{1}{4}c_x & 1+s_x+s_y & -\frac{1}{2}s_y+\frac{1}{4}c_y & 0 \\ 0 & -\frac{1}{2}s_y-\frac{1}{4}c_y & 1+s_x+s_y & -\frac{1}{2}s_x-\frac{1}{4}c_x \\ -\frac{1}{2}s_y-\frac{1}{4}c_y & 0 & -\frac{1}{2}s_x+\frac{1}{4}c_x & 1+s_x+s_y \end{bmatrix} \begin{bmatrix} u_{i,j,k+1} \\ u_{i+1,j,k+1} \\ u_{i+1,j+1,k+1} \\ u_{i,j+1,k+1} \end{bmatrix} = \begin{bmatrix} rhs_{i,j} \\ rhs_{i+1,j} \\ rhs_{i+1,j+1} \\ rhs_{i,j+1} \end{bmatrix} \qquad (2.1)$$

where

$$rhs_{i,j}=(\tfrac{1}{2}s_x+\tfrac{1}{4}c_x)u_{i-1,j,k+1}+(\tfrac{1}{2}s_y+\tfrac{1}{4}c_y)u_{i,j-1,k+1}+(1-s_x-s_y)u_{i,j,k}+(\tfrac{1}{2}s_x+\tfrac{1}{4}c_x)u_{i-1,j,k}$$
$$+(\tfrac{1}{2}s_x-\tfrac{1}{4}c_x)u_{i+1,j,k}+(\tfrac{1}{2}s_y-\tfrac{1}{4}c_y)u_{i,j+1,k}+(\tfrac{1}{2}s_y+\tfrac{1}{4}c_y)u_{i,j-1,k}$$

$$rhs_{i+1,j}=(\tfrac{1}{2}s_x-\tfrac{1}{4}c_x)u_{i+2,j,k+1}+(\tfrac{1}{2}s_y+\tfrac{1}{4}c_y)u_{i+1,j-1,k+1}+(1-s_x-s_y)u_{i+1,j,k}+(\tfrac{1}{2}s_x+\tfrac{1}{4}c_x)u_{i,j,k}+(\tfrac{1}{2}s_x-\tfrac{1}{4}c_x)u_{i+2,j,k}+(\cdot$$

$$rhs_{i+1,j+1}=(\tfrac{1}{2}s_x-\tfrac{1}{4}c_x)u_{i+2,j+1,k+1}+(\tfrac{1}{2}s_y-\tfrac{1}{4}c_y)u_{i+1,j+2,k+1}+(1-s_x-s_y)u_{i+1,j+1,k}$$
$$+(\tfrac{1}{2}s_x+\tfrac{1}{4}c_x)u_{i,j+1,k}+(\tfrac{1}{2}s_x-\tfrac{1}{4}c_x)u_{i+2,j+1,k}+(\tfrac{1}{2}s_y-\tfrac{1}{4}c_y)u_{i+1,j+2,k}+(\tfrac{1}{2}s_y+\tfrac{1}{4}c_y)u_{i+1,j,k}$$

$$rhs_{i,j+1}=(\tfrac{1}{2}s_x+\tfrac{1}{4}c_x)u_{i-1,j+1,k+1}+(\tfrac{1}{2}s_y-\tfrac{1}{4}c_y)u_{i,j+2,k+1}+(1-s_x-s_y)u_{i,j+1,k}$$
$$+(\tfrac{1}{2}s_x+\tfrac{1}{4}c_x)u_{i-1,j+1,k}+(\tfrac{1}{2}s_x-\tfrac{1}{4}c_x)u_{i+1,j+1,k}+(\tfrac{1}{2}s_y-\tfrac{1}{4}c_y)u_{i,j+2,k}+(\tfrac{1}{2}s_y+\tfrac{1}{4}c_y)u_{i,j,k} \qquad (2.2)$$

Using the *Mathematica* software, the inverse of the (4x4) coefficient matrix on the left hand side is evaluated. Thus, the explicit form of the system (2.1) can be written as

$$\begin{bmatrix} u_{i,j,k+1} \\ u_{i+1,j,k+1} \\ u_{i+1,j+1,k+1} \\ u_{i,j+1,k+1} \end{bmatrix} = \frac{1}{const} \begin{bmatrix} a_1 & a_2 & a_3 & a_4 \\ a_5 & a_1 & a_4 & a_6 \\ a_7 & a_8 & a_1 & a_5 \\ a_8 & a_9 & a_2 & a_1 \end{bmatrix} \begin{bmatrix} rhs_{ij} \\ rhs_{i+1,j} \\ rhs_{i+1,j+1} \\ rhs_{i,j+1} \end{bmatrix} \qquad (2.3)$$

where

$a_1 = (16 + 48a + 44a^2 + 12a^3 + 96ab + 44ab^2 + ac^2 + ad^2 + 48b + 44a^2b + 44b^2 + 12b^3 + bc^2 + bd^2 + c^2 + d^2)/16.0,$
$a_2 = (32a + 64a^2 + 24a^3 + 64ab + 40ab^2 - 32abc - 32ac + 2ac^2 - 2ad^2 + 64a^2b - 32bc - 16c - 12a^2c - 20b^2c - c^3 + cd^2)/64.0,$
$a_3 = (4ab + 4ab^2 - 2abc - 2abd + acd - 2ad + 4a^2b - 2bc + bcd - 2b^2c + cd - 2a^2d)/8.0,$
$a_4 = (64ab + 64ab^2 - 32abd - 32ad + 32b + 40a^2b + 64b^2 + 24b^3 - 2bc^2 - 32bd + 2bd^2 - 16d - 20a^2d - 12b^2d + c^2d - d^3)/64.0,$
$a_5 = (32a + 64a^2 + 24a^3 + 64ab + 40ab^2 + 32abc + 32ac + 2ac^2 - 2ad^2 + 64a^2b + 32bc + 16c + 12a^2c + 20b^2c + c^3 - cd^2)/64.0,$
$a_6 = (4ab + 4ab^2 + 2abc - 2abd - acd - 2ad + 4a^2b + 2bc - bcd + 2b^2c - cd - 2a^2d)/8.0,$
$a_7 = (4ab + 4ab^2 + 2abc + 2abd + acd + 2ad + 4a^2b + 2bc + bcd + 2b^2c + cd + 2a^2d)/8.0,$
$a_8 = (64ab + 64ab^2 + 32abd + 32ad + 32b + 40a^2b + 64b^2 + 24b^3 - 2bc^2 + 32bd + 2bd^2 + 16d + 20a^2d + 12b^2d - c^2d + d^3)/64.0,$
$a_9 = (4ab + 4ab^2 - 2abc + 2abd - acd + 2ad + 4a^2b - 2bc - bcd - 2b^2c - cd + 2a^2d)/8.0,$

and

$const = (256 + 1024a + 1408a^2 + 768a^3 + 144a^4 + 3072ab + 2816ab^2 + 768ab^3 + 64(abc^2 + abd^2 + ac^2 + ad^2) + 1024b + 2816a^2b + 768a^3b + 1408b^2 + 1248a^2b^2 + 768b^3 + 144b^4 + 64bc^2 + 64bd^2 + 32c^2 + 24a^2c^2 + 40b^2c^2 + c^4 + 32d^2 + 40a^2d^2 + 24b^2d^2 - 2c^2d^2 + d^4)/256.$

(2.4)

Here, $a = s_x$, $b = s_y$, $c = c_x$ and $d = c_y$. Note that at any time level, the computational molecule of the approximation (2.3) is similar to the one shown in Figure 1.

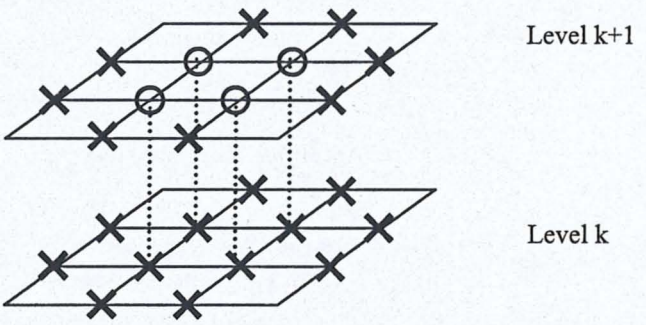

Figure 1. Computational molecule of the approximation (2.3)

The EG method proceeds with iterative evaluation of solutions in blocks of four points using these formulae throughout the whole solution domain until convergence is achieved. Also it may be observed that the formula (2.3) has the advantage of being explicit and thus suitable for parallelization.

3 Spatial Multicolour Strategy For Parallelism

It is imperative to note that every time step of the *parabolic* problem is an *elliptic* problem. Thus parallelism is exploited in the spatial dimension, in which at any one time level, the x-y plane at time level k+1 is decomposed into a number of horizontal strips consisting of two rows of four point groups arranged in the order shown in Fig. 2 for the case n = 9.

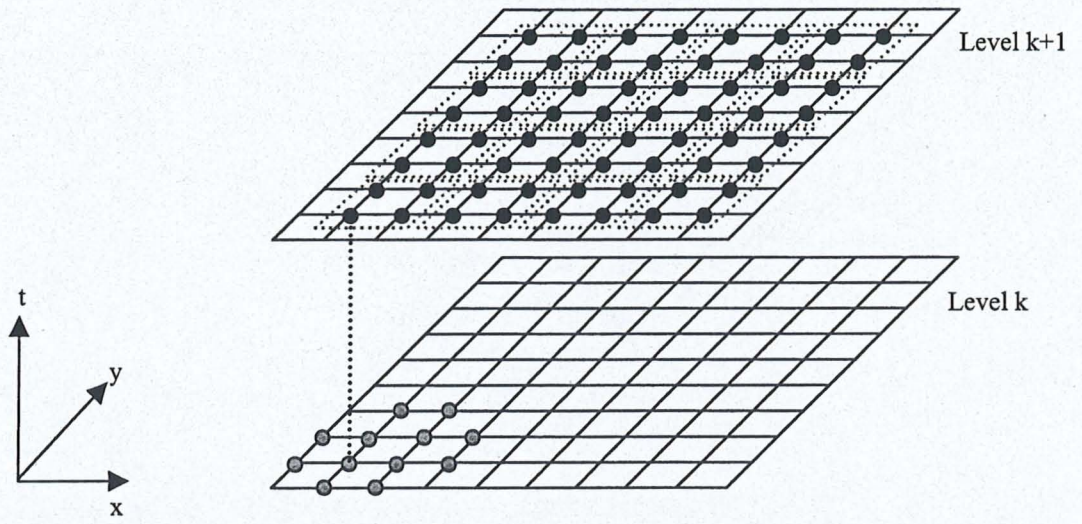

Figure 2. Points involved in updating the four point groups at time level (k+1) for n=9

At each time level, each iteration is split into 2 stages; the four points blocks coloured in white (W) are updated in the first stage, then the block of points in grey (G) are updated in the second stage as illustrated in Figure 3.

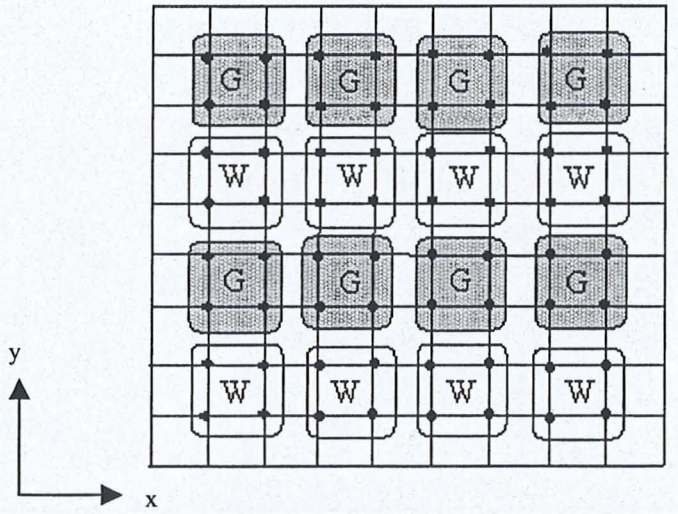

Figure 3. Decomposition of 2-coloured strips for parallelisation on x-y plane at each time level (k+1)

Updating the white group followed by the grey group in natural ordering using discretisation (2.3) will result in the coefficient matrix A of the linear system Au = B at each time level k+1 to have the following form

$$A = \begin{bmatrix} \underline{D} & C \\ F & \underline{D} \end{bmatrix} \begin{bmatrix} \underline{u}_W \\ \underline{u}_G \end{bmatrix} \tag{3.1}$$

with $\underline{D}$, C and F being block diagonal, lower and upper triangular matrices respectively. Hence, for each time level, the matrix A is π-consistently ordered and has property $A^{(\pi)}$ so that the theory of block Successive OverRelaxation (SOR) is valid and thus the scheme converges.

At each time level (k+1), the iterative evaluation of the system (3.1) may then be written as

$$\begin{aligned} \underline{u}_W^{(k+1)} &= \underline{D}^{-1}[B_W - C\underline{u}_G^{(k)}] \\ \underline{u}_G^{(k+1)} &= \underline{D}^{-1}[B_G - F\underline{u}_W^{(k+1)}]. \end{aligned} \tag{3.2}$$

It may be observed that the computations in each coloured group are independent of each other and therefore parallelisable. The strategy used in distributing the points at any particular time level is the same as in the *elliptic* case [1]. At each time level, a subset of equal number of consecutive strips are distributed to each processor available. Each processor iterates on its own group of points and checks for its own local convergence. After local convergence is achieved, a check for global convergence is made. The converged solutions will then be passed to the next time level as the initial guess for the next iteration process. The solution process continues until solutions at all the desired time levels have been obtained.

4 Numerical Experiments

In this section, we report on some preliminary results obtained for solving the convection-diffusion problem (1.1) on a Linux cluster located at the Department of Computer Science, USM. Communication between processors is performed using Parallel Virtual Machine (PVM) communication library. We consider the solution of problem (1.1) for $0 < x < X$, $0 < y < Y$, $t > 0$ with initial conditions

$$U(x,y,0) = \exp\left\{\frac{-(x-x_0)^2}{\alpha_x} - \frac{(y-y_0)^2}{\alpha_y}\right\}, \qquad 0 \le x \le X,\ 0 \le y \le Y \tag{4.1}$$

and boundary conditions

$$\begin{aligned}
U(0,y,t) &= \frac{1}{4t+1}\exp\left\{\frac{-(\alpha_x' t + x_0)^2}{\alpha_x(4t+1)} - \frac{(y-\alpha_y' t - y_0)^2}{\alpha_y(4t+1)}\right\} \\
U(X,y,t) &= \frac{1}{4t+1}\exp\left\{\frac{-(X-\alpha_x' t - x_0)^2}{\alpha_x(4t+1)} - \frac{(y-\alpha_y' t - y_0)^2}{\alpha_y(4t+1)}\right\} \\
U(x,Y,t) &= \frac{1}{4t+1}\exp\left\{\frac{-(x-\alpha_x' t - x_0)^2}{\alpha_x(4t+1)} - \frac{(Y-\alpha_y' t - y_0)^2}{\alpha_y(4t+1)}\right\} \\
U(x,0,t) &= \frac{1}{4t+1}\exp\left\{\frac{-(x-\alpha_x' t - x_0)^2}{\alpha_x(4t+1)} - \frac{(\alpha_y' t + y_0)^2}{\alpha_y(4t+1)}\right\}
\end{aligned} \qquad 0 \le y \le Y,\ 0 \le x \le X,\ t>0. \tag{4.2}$$

The exact solution on the region $0 \le x \le X$, $0 \le y \le Y$ is [4]:

$$U(x,y,t) = \frac{1}{4t+1}\exp\left\{\frac{-(x-\alpha_x' t - x_0)^2}{\alpha_x(4t+1)} - \frac{(y-\alpha_y' t - y_0)^2}{\alpha_y(4t+1)}\right\},\ t > 0. \tag{4.3}$$

The absolute test was used in the local and global convergence tests with tolerance $\varepsilon = 10^{-5}$. Experiments were performed on the model problem for various sizes of n with number of processors ranging from 1 to 6. Six mesh sizes have been used ranging from 121 to 721. A Single Program Multiple Data (SPMD) paradigm consisting of one parent process and one or more than one child processes is used in distributing the tasks. The main responsibility of the parent process is to spawn the child processes, distribute the data to the processes evenly, collect the computed data and print out the results. In addition to these, the parent process will also be involved in the computation processes and will not be left idle. Whilst the child processes will receive the spawned tasks, compute the data, and send the results to the parent. The timing results are presented in Table 1. The speedup values plotted against the number of processors for different n are shown in Figure 4.

Table 1. EXECUTION TIMES (IN SECS.) FOR THE PARALLEL ALGORITHM (t= 0.01, $\Delta t = 0.0005$, $\alpha_x = \alpha_y = 0.9$, $\alpha_x' = \alpha_y' = 0.1, x_0 = y_0 = 0.5$)

Convection Diffusion - EG	No. Of Processors	Iter	Error	Time	Speedup
N=121 W: 1.4536	1	180	0.000545	0.378054	1.000000
	2	180	0.000545	0.258850	1.460514
	3	180	0.000545	0.217236	1.740292
	4	180	0.000545	0.288120	1.312141
	5	180	0.000545	0.306627	1.232944
	6	180	0.000545	0.310484	1.217628
N=241 W:1.6186	1	260	0.000617	2.791991	1.000000
	2	260	0.000617	1.485046	1.88007
	3	260	0.000617	1.000897	2.789489
	4	260	0.000617	0.881286	3.168087
	5	260	0.000617	0.800575	3.487482
	6	260	0.000617	0.763433	3.657153
N=361 W:1.7200	1	360	0.000695	7.688926	1.000000
	2	360	0.000695	4.260272	1.804797
	3	360	0.000695	3.155303	2.436827
	4	360	0.000695	2.593486	2.964707
	5	360	0.000695	2.086896	3.684384
	6	360	0.000695	1.780085	4.319415
N=481 W:1.7252	1	520	0.001185	16.451758	1.000000
	2	520	0.001185	8.937741	1.840707
	3	520	0.001185	6.514518	2.525399
	4	520	0.001185	5.236424	3.141793
	5	520	0.001185	4.424787	3.71809
	6	520	0.001185	4.001257	4.111647
N=601 W:1.7275	1	704	0.001694	32.729571	1.000000
	2	704	0.001694	17.408637	1.880077
	3	704	0.001694	12.427545	2.633631
	4	704	0.001694	10.115257	3.235664
	5	704	0.001694	8.335114	3.926709
	6	704	0.001694	7.180388	4.55819
N=721 W:1.7353	1	866	0.002246	57.396197	1.000000
	2	866	0.002246	30.071938	1.90863
	3	866	0.002246	21.086880	2.721891
	4	866	0.002246	16.501501	3.478241
	5	866	0.002246	13.713812	4.185284
	6	866	0.002246	11.794802	4.866228

5. CONCLUSIONS

From the results in Table 1, it can be seen that the speedup curve for the proposed algorithm gets closer to the 'ideal' graph as n gets larger. It can be observed that a slightly greater speedup is obtained when the mesh size is larger indicating that the amount of computations carried out over the total overheads in this method is slightly greater compared to the smaller mesh size. From this work, we may conclude that the new parallel group iterative algorithm is able to benefit from parallelism when solving the *convection-diffusion* equation on a cluster of PC's.

Acknowledgment

The authors acknowledge the Fundamental Research Grant Scheme (304/PMATHS/670020) that has resulted in this article.

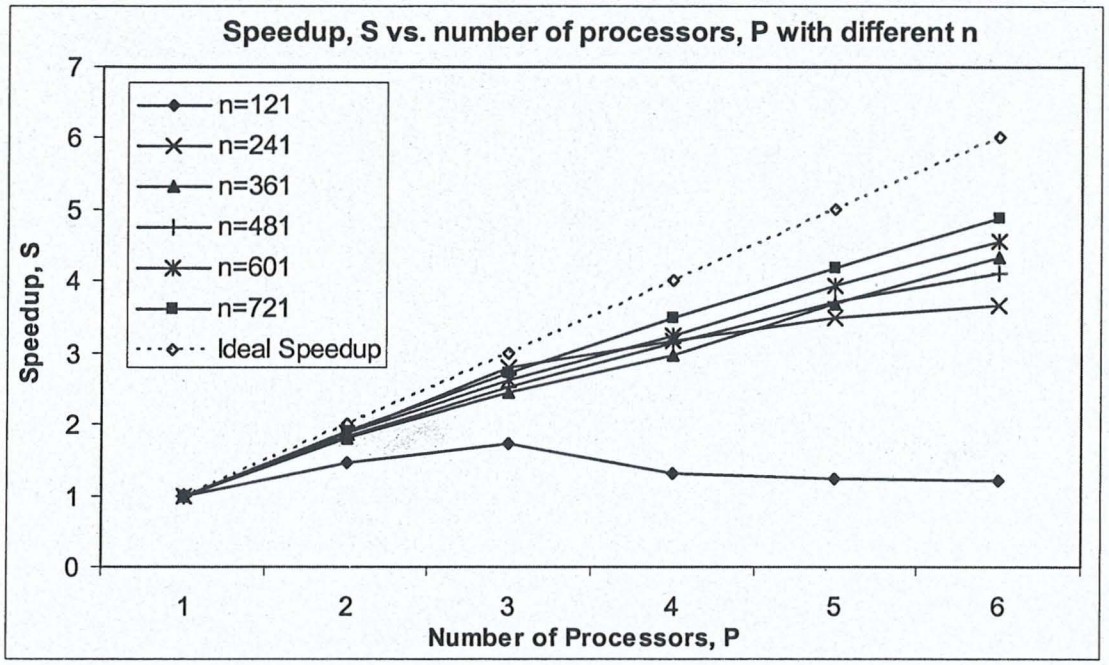

Figure 4. The speedup values plotted against the number of processors.

REFERENCES

[1] N. H. M.Ali, R. Abdullah and K.. J. Lee, 2003, *'Explicit Group Iterative Solver On A Message Passing Environment'*, Recent Advances in Parallel Virtual Machine and Message Passing Interface: Proceedings of the 10th European PVM/MPI 2003, Sept. 29-Oct. 2, Venice, Italy, Lecture Notes in Computer Science, Springer-Verlag Berlin, pp. 232-236, Editors: Jack Dongarra, Domenico Laforenza, Salvatore Orlando (ISBN 3-540-20149-1).

[2] I. Arsmah, *The Study of the Iterative Solution of the Boundary Value Problems by the Finite Difference Methods,* PhD Thesis, Universiti Kebangsaan Malaysia, 1992.

[3] D.J. Evans and W.S. Yousif, 1990, The Implementation of The Explicit Block Iterative Methods On the Balance 8000 Parallel Computer, *Parallel Computing*, **16**, 81-97.

[4] B.J. Noye, *Numerical Methods for Solving the Transport Equation, Numerical Modelling: Application to Marine Systems*, (ed. Noye), (Amsterdam: North Holland Publishing Company, 1987, 195-230).

[5] W.S. Yousif, and D.J. Evans, 1986, Explicit group over-relaxation methods for solving elliptic partial differential equations, *Math. Computer Simulation*, **28**, 453-466.

[6] J. Zhu, *Solving Partial Differential Equations On Parallel Computers* (Singapore: World Scientific Publishing Co., 1994)