Parallel Accelerated Group Iterative Algorithms in the Solution of Two-Space Dimensional Diffusion Equations

Norhashidah Hj Mohd Ali and Foo Kai Pin
Universiti Sains Malaysia
Malaysia

1. Introduction

Diffusion equations are mathematical models which explain how the concentration of one or more substances distributed in space is altered by a diffusion process which causes the substances to spread out over a surface in space. For a normal diffusion process, the flux of particles into one region must be the sum of particle flux flowing out of the surrounding regions. From Fick's first law, this can be represented mathematically by the following diffusion equation

$$\frac{\partial u}{\partial \text{ t}} = \nabla.(D\nabla u) \; . \tag{1}$$

If the diffusion coefficient is constant in space, then

$$\frac{\partial u}{\partial \text{ t}} = D\nabla^2 u \tag{2}$$

here ∇^2 is the normal Laplacian. The values u and D take on different meanings in different situations: in particle diffusion, u is interpreted as a concentration and D as a diffusion coefficient; while in heat diffusion, u is the temperature and D is the thermal conductivity.
The application of the finite difference methods for solving time-dependent Partial Differential Equations(PDEs) such as this, at any particular time level, yields a system of linear simultaneous equations of the form

$$\text{A}\underline{\text{u}} = \underline{\text{b}} \tag{3}$$

where iterative methods are normally more feasible in solving the system due to the sparsity of the matrix A. The applicability of explicit group methods, in which several unknowns are connected together in the iteration formula resulting in a sub-system that must be solved before any one of the them can be determined, have been investigated on solving these types of PDEs. In their early work, Evans and Abdullah (1983b) generated single-step, one-parameter families of finite difference approximations to the heat equation in one space dimension by coupling in groups of two the values of the approximations obtained by known asymmetric formulas at adjacent grid points at the advanced time level. The

resulting equations are implicit but they can be easily converted to explicit form. The method was shown to possess unconditional stability with good accuracies. Evans and Abdullah (1983a) also developed the group explicit method for the solution the two dimensional diffusion where a general two-level six point finite difference approximation was developed to solve the parabolic equation. The method was further developed as the Alternating Group Explicit (AGE) method (Evans & Sahimi, 1988), which is an analogue to the famous Alternating Direction Implicit(ADI) method but has the advantage of being explicit and thus very easy to parallelise. The emergence of newer explicit group methods on skewed or rotated grids with promising and improved results was greatly observed since the early 1990s. Among them are the works of Abdullah (1991) who developed the four-point Explicit Decoupled Group (EDG) by discretising the PDEs on skewed grids. This method was shown to require less computational time with the same order of accuracies than the Explicit Group (EG) method pioneered by Yousif and Evans (1986) in solving the Poisson model problem. A few years on, Othman and Abdullah (2000) modified the formulation of the EG method by deriving formulas based on the centred five points approximation formula with the grid spacing *h* and 2*h*, and the rotated five points approximation formula to come up with the improved modified four-point EG which was shown to exhibit lesser computational effort than the existing EG and EDG. Since then, active research has been conducted to investigate the capabilities of the variants of these group relaxation methods in improving the standard or traditional algorithms in solving several types of PDEs. This includes the work of Ali and Lee (2007) who derived the Accelerated OverRelaxation (AOR) variant of the EDG group scheme in the solution of elliptic equation where its performance results were compared with the EG (AOR) proposed by Martins et *al.* (2002). The new EDG (AOR) scheme requires less execution time than the existing EG (AOR) method where the gain in speed of EDG (AOR) method over the EG (AOR) method ranges from approximately 51% to 59%. The performance analysis of the parallel algorithms of these EG and EDG schemes were also established in Ng and Ali (2008) where the algorithms turn out to be efficient solvers for the steady-state elliptic equation on distributed memory multicomputer with high scalability.
In this chapter we shall present the formulation of new explicit group algorithms intended for solving the two dimensional time-dependent diffusion equation. A novel approach of using four points group strategy, implemented on different spacing stencils incorporated with the AOR technique, is used in the formulation. Explainations on how the methods need to be reconfigurated mathematically as to be successfully ported to run on a message-passing parallel computer system is also presented.

2. Formulations of group methods

We consider the finite difference discretization schemes for solving the two dimensional diffusion equation of the form

$$\frac{\partial u}{\partial \mathrm{t}} = \nabla^2 u + f(x,y,t) \tag{4}$$

with a specified initial and boundary conditions on a unit square with spacings $\Delta x = \Delta y = h = 1/n$ in both directions *x* and *y*, with $x_i = x_0 + ih$, $y_j = y_0 + jh$ $(i,j = 0,1,2,...,n)$, $t = k\Delta t$ $(k = 0, 1, 2, \ldots)$; here, *f* is a real continuous function. One commonly used implicit finite difference scheme based on the centred difference in time and space formulation about the point $(i,j,k+1/2)$ is the *Crank-Nicolson* scheme which transform (4) into

$$\begin{aligned}(1+2r)u_{i,j,k+1}-\frac{r}{2}u_{i-1,j,k+1}-\frac{r}{2}u_{i+1,j,k+1}-\frac{r}{2}u_{i,j-1,k+1}-\frac{r}{2}u_{i,j+1,k+1}=\\ (1-2r)u_{i,j,k}+\frac{r}{2}u_{i-1,j,k}+\frac{r}{2}u_{i+1,j,k}+\frac{r}{2}u_{i,j-1,k}+\frac{r}{2}u_{i,j+1,k}+\Delta tf_{i,j,k+\frac{1}{2}}\end{aligned} \tag{5}$$

where $r=\Delta t/h^2$. Based on this approximation, several group schemes have been constructed (Ali, 1998). The Explicit Group (EG) method, for example, was formulated by taking the iteration process in groups of four points. At each time level (k+1), the mesh points are grouped in blocks of four points (i,j), (i+1,j), (i+1,j+1) and (i,j+1) and equation (5) is applied to each of these points resulting in the following (4x4) system:

$$\begin{bmatrix} 1+2r & -\frac{r}{2} & 0 & -\frac{r}{2} \\ -\frac{r}{2} & 1+2r & -\frac{r}{2} & 0 \\ 0 & -\frac{r}{2} & 1+2r & -\frac{r}{2} \\ -\frac{r}{2} & 0 & -\frac{r}{2} & 1+2r \end{bmatrix} \begin{bmatrix} u_{i,j,k+1} \\ u_{i+1,j,k+1} \\ u_{i+1,j+1,k+1} \\ u_{i,j+1,k+1} \end{bmatrix} = \begin{bmatrix} rhs_{i,j} \\ rhs_{i+1,j} \\ rhs_{i+1,j+1} \\ rhs_{i,j+1} \end{bmatrix} \tag{6}$$

which may be solved explicitly in groups of four points as

$$\begin{bmatrix} u_{i,j,k+1} \\ u_{i+1,j,k+1} \\ u_{i+1,j+1,k+1} \\ u_{i,j+1,k+1} \end{bmatrix} = \\ \frac{1}{2(1+r)(1+2r)+(1+3r)} \begin{bmatrix} 7r^2+8r+2 & r(1+2r) & r^2 & r(1+2r) \\ r(1+2r) & 7r^2+8r+2 & r(1+2r) & r^2 \\ r^2 & r(1+2r) & 7r^2+8r+2 & r(1+2r) \\ r(1+2r) & r^2 & r(1+2r) & 7r^2+8r+2 \end{bmatrix} \begin{bmatrix} rhs_{i,j} \\ rhs_{i+1,j} \\ rhs_{i+1,j+1} \\ rhs_{i,j+1} \end{bmatrix}, \tag{7}$$

where

$$\begin{aligned} rhs_{i,j} &= \frac{r}{2}\left(u_{i-1,j,k+1}+u_{i,j-1,k+1}\right)+\frac{r}{2}\left(u_{i-1,j,k}+u_{i,j-1,k}+u_{i+1,j,k}+u_{i,j+1,k}\right) \\ &\quad +(1-2r)u_{i,j,k}+\Delta tf_{i,j,k+\frac{1}{2}} \\ rhs_{i+1,j} &= \frac{r}{2}\left(u_{i+2,j,k+1}+u_{i+1,j-1,k+1}\right)+\frac{r}{2}\left(u_{i,j,k}+u_{i+2,j,k}+u_{i+1,j-1,k}+u_{i+1,j+1,k}\right) \\ &\quad +(1-2r)u_{i+1,j,k}+\Delta tf_{i+1,j,k+\frac{1}{2}} \\ rhs_{i+1,j+1} &= \frac{r}{2}\left(u_{i-+2,j+1,k+1}+u_{i+1,j+2,k+1}\right)+\frac{r}{2}\left(u_{i,j+1,k}+u_{i+2,j+1,k}+u_{i+1,j,k}+u_{i+1,j+2,k}\right) \\ &\quad +(1-2r)u_{i+1,j+1,k}+\Delta tf_{i+1,j+1,k+\frac{1}{2}} \\ rhs_{i,j+1} &= \frac{r}{2}\left(u_{i-1,j+1,k+1}+u_{i,j+2,k+1}\right)+\frac{r}{2}\left(u_{i-1,j+1,k}+u_{i,j,k}+u_{i+1,j+1,k}+u_{i,j+2,k}\right) \\ &\quad +(1-2r)u_{i,j+1,k}+\Delta tf_{i,j+1,k+\frac{1}{2}} \end{aligned}$$

The method proceed with iterative evaluation of solutions in blocks of four points respectively using these formulas throughout the whole net region until convergence is achieved.
Using another type of discretization, which we called the *rotated* finite difference approximation:

$$\begin{aligned}\frac{u_{i,j,k+1}-u_{i,j,k}}{\Delta t} &= \frac{1}{2}\left[\frac{u_{i-1,j-1,k+1}-2u_{i,j,k+1}+u_{i+1,j+1,k+1}}{2\Delta x^2}+\frac{u_{i-1,j-1,k}-2u_{i,j,k}+u_{i+1,j+1,k}}{2\Delta x^2}\right]\\ &+\frac{1}{2}\left[\frac{u_{i-1,j+1,k+1}-2u_{i,j,k+1}+u_{i+1,j-1,k+1}}{2\Delta y^2}+\frac{u_{i-1,j+1,k}-2u_{i,j,k}+u_{i+1,j-1,k}}{2\Delta y^2}\right]+f_{i,j,k+\frac{1}{2}}\end{aligned} \tag{8}$$

another group scheme was formulated called the Explicit Decoupled Group (EDG) method. This approximation is obtained by using the Taylor series expansion of the solution u at appropriate grid points where the resulting computational stencil is 45^0 clockwise rotated from the stencil of the standard *Crank-Nicolson* (5). At each time level (k+1), the mesh points are grouped in blocks of four points, (i,j), (i+1,j+1), (i+1,j) and (i,j+1), and the rotated finite difference approximation (8) is applied to the u values at each of these points resulting in a (4x4) system which may be decoupled into two 2x2 matrices of the following form (Ali, 1998):

$$\cdot\begin{bmatrix}u_{i,j,k+1}\\ u_{i+1,j+1,k+1}\end{bmatrix}=\frac{16}{15r^2+32r+16}\begin{bmatrix}1+r & \frac{r}{4}\\ \frac{r}{4} & 1+r\end{bmatrix}\begin{bmatrix}rhs_{i,j}\\ rhs_{i+1,j+1}\end{bmatrix}, \tag{9}$$

and

$$\begin{bmatrix}u_{i+1,j,k+1}\\ u_{i,j+1,k+1}\end{bmatrix}=\frac{16}{15r^2+32r+16}\begin{bmatrix}1+r & \frac{r}{4}\\ \frac{r}{4} & 1+r\end{bmatrix}\begin{bmatrix}rhs_{i+1,j}\\ rhs_{i,j+1}\end{bmatrix}, \tag{10}$$

where

$$\begin{aligned}rhs_{i,j} &= \frac{r}{4}\left(u_{i-1,j-1,k+1}+u_{i+1,j-1,k+1}+u_{i-1,j+1,k+1}\right)+\frac{r}{4}\left(u_{i-1,j-1,k}+u_{i+1,j-1,k}+u_{i-1,j+1,k}+u_{i+1,j+1,k}\right)\\ &\quad +(1-r)u_{i,j,k}+\Delta t f_{i,j,k+\frac{1}{2}}\\ rhs_{i+1,j+1} &= \frac{r}{4}\left(u_{i+2,j+2,k+1}+u_{i+2,j,k+1}+u_{i,j+2,k+1}\right)+\frac{r}{4}\left(u_{i+2,j+2,k}+u_{i+2,j,k}+u_{i,j+2,k}+u_{i,j+2,k}\right)\\ &\quad +(1-r)u_{i+1,j+1,k}+\Delta t f_{i+1,j+1,k+\frac{1}{2}}\\ rhs_{i+1,j} &= \frac{r}{4}\left(u_{i,j-1,k+1}+u_{i+2,j-1,k+1}+u_{i+2,j+1,k+1}\right)+\frac{r}{4}\left(u_{i,j-1,k}+u_{i+2,j-1,k}+u_{i,j+1,k}+u_{i+2,j+1,k}\right)\\ &\quad +(1-r)u_{i+1,j,k}+\Delta t f_{i+1,j,k+\frac{1}{2}}\\ rhs_{i,j+1} &= \frac{r}{4}\left(u_{i-1,j,k+1}+u_{i+1,j+2,k+1}+u_{i-1,j+2,k+1}\right)+\frac{r}{4}\left(u_{i-1,j,k}+u_{i+1,j+2,k}+u_{i-1,j+2,k}+u_{i+1,j,k}\right)\\ &\quad +(1-r)u_{i,j+1,k}+\Delta t f_{i,j+1,k+\frac{1}{2}}.\end{aligned} \tag{11}$$

The EDG method requires less computing time whilst maintaining the same order of accuracies as the original EG method (Ali, 1998). Based on approximation formula (8) derived on different grid spacings, new modified group explicit methods will be formulated in combination of the Accelerated Over-Relaxation (AOR) technique. The next sub-sections will elaborate on the formulations of these methods.

2.1 Modified explicit group (MEG) AOR method

We consider the standard five-point formula for the diffusion equation on Ω_{2h} grid:

$$u_{i,j}^{(m+1)} = \frac{1}{1+2r}\left(\frac{r}{2}\left(u_{i-2,j}^{(m)} + u_{i+2,j}^{(m)} + u_{i,j-2}^{(m)} + u_{i,j+2}^{(m)} + v_{i-2,j} + v_{i+2,j} + v_{i,j-2} + v_{i,j+2}\right) + (1-2r)v_{i,j} + \Delta t F_{i,j}\right), \tag{12}$$

$$r = \frac{\Delta t}{4h^2}.$$

Here, u_{ij} is the value of u at the current time level (k+1), v_{ij} is the value of u from the previous time level (k), while m is the iteration level. F_{ij} is the value of f at the point (i,j) at time level (k+1). We begin by dividing the grid points in the solution domain into 3 types of points, indicated by □, △, ●, and arranged in a specific alternate ordering, as shown in Fig. 1. For the iterations, we consider the points indicated by ● in Fig. 2. Similar to the EG method described in the previous section, we may apply equation (12) to groups of four points of the iterative points to produce a 4x4 MEG formula. The convergence of this method may be improved by the introduction of the AOR technique (Hadjidimos, 1978) where drastic improvement in convergence can be obtained by choosing suitable relaxation parameters in its formula. The idea in AOR technique is to apply an extrapolation of a two-parameter Successive OverRelaxation (SOR)-type iterative procedure in the formula.

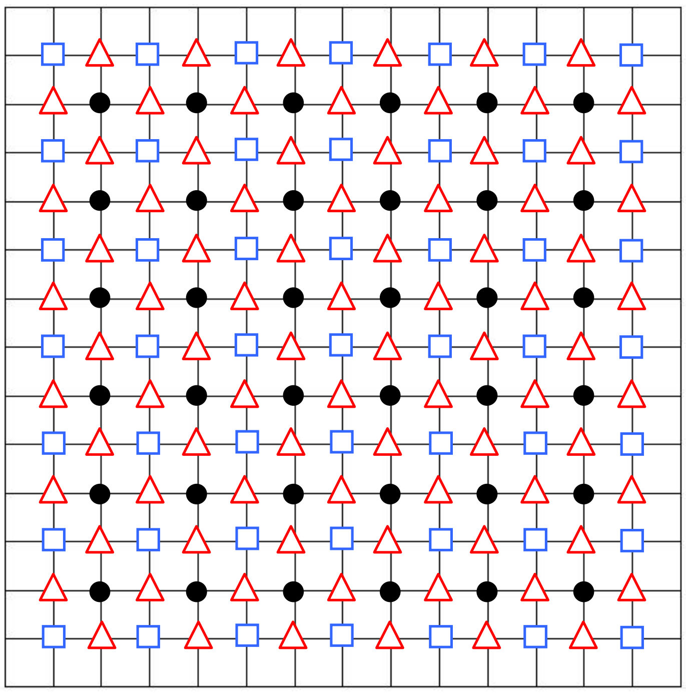


Fig. 1. Construction of different grid points in the spatial solution domain at time level k+1.

The two parameters may be exploited which result in methods which will converge faster than any other method of the same type. Using this idea, we can introduce the over-relaxation parameters ω and R into the 4x4 MEG formula as a way to further accelerate the convergence of the iterative method scheme as the following:

$$\begin{aligned}
u_{i,j}^{(m+1)} &= s_3[w(s_4b_5+s_5b_6+s_6b_7+s_5b_8)+R(s_4t_1+s_5\{t_2+t_3\})]+(1-w)u_{i,j}^{(m)}\\
u_{i+2,j}^{(m+1)} &= s_3[w(s_5b_5+s_4b_6+s_5b_7+s_6b_8)+R(s_5t_1+s_4t_2+s_6t_3)]+(1-w)u_{i+2,j}^{(m)}\\
u_{i+2,j+2}^{(m+1)} &= s_3[w(s_6b_5+s_5b_6+s_4b_7+s_5b_8)+R(s_6t_1+s_5\{t_2+t_3\})]+(1-w)u_{i+2,j+2}^{(m)}\\
u_{i,j+2}^{(m+1)} &= s_3[w(s_5b_5+s_6b_6+s_5b_7+s_4b_8)+R(s_5t_1+s_6t_2+s_4t_3)]+(1-w)u_{i,j+2}^{(m)}
\end{aligned} \tag{13}$$

where

$$\begin{aligned}
b_1 &= s_2v_{i,j}+\Delta tF_{i,j} & b_5 &= s_1(u_{i-2,j}^{(m)}+u_{i,j-2}^{(m)}+v_{i-2,j}+v_{i+2,j}+v_{i,j-2}+v_{i,j+2})+b_1\\
b_2 &= s_2v_{i+2,j}+\Delta tF_{i+2,j} & b_6 &= s_1(u_{i+4,j}^{(m)}+u_{i+2,j-2}^{(m)}+v_{i,j}+v_{i+4,j}+v_{i+2,j-2}+v_{i+2,j+2})+b_2\\
b_3 &= s_2v_{i+2,j+2}+\Delta tF_{i+2,j+2} & b_7 &= s_1(u_{i+4,j+2}^{(m)}+u_{i+2,j+4}^{(m)}+v_{i,j+2}+v_{i+4,j+2}+v_{i+2,j}+v_{i+2,j+4})+b_3\\
b_4 &= s_2v_{i,j+2}+\Delta tF_{i,j+2} & b_8 &= s_1(u_{i-2,j+2}^{(m)}+u_{i,j+4}^{(m)}+v_{i-2,j+2}+v_{i+2,j+2}+v_{i,j}+v_{i,j+4})+b_4
\end{aligned}$$

$$\begin{aligned}
s_1 &= 0.5r\\
s_2 &= 1-2r\\
s_3 &= \frac{1}{2(1+r)(1+2r)(1+3r)} & t_1 &= s_1(u_{i-2,j}^{(m+1)}-u_{i-2,j}^{(m)}+u_{i,j-2}^{(m+1)}-u_{i,j-2}^{(m)})\\
& & t_2 &= s_1(u_{i+2,j-2}^{(m+1)}-u_{i+2,j-2}^{(m)})\\
s_4 &= 7r^2+8r+2 & t_3 &= s_1(u_{i-2,j+2}^{(m+1)}-u_{i-2,j+2}^{(m)}).\\
s_5 &= r(1+2r)\\
s_6 &= r^2
\end{aligned} \tag{14}$$

Unlike SOR, there is no general formula to determine the parameters R and w. But according to Hadjidimos (1978), the parameter R is normally chosen to be close to the value ω obtained from the corresponding SOR technique which give the least number of iterations.
We can then define the four points MEG (AOR) method for diffusion equation as the following:

Algorithm 1

1. Divide the grid points into points of type ●, △ and □ at level k+1 as shown in Fig. 1.
2. Set the initial guess for the iterations.
3. Use Equations (13)-(14) to evaluate the solution of points of type ● iteratively at level k+1.
4. Check the convergence. If the iterations converge, go to step 5. Otherwise, repeat step 3 until convergence is achieved.
5. After the solution at points of type ● converge, the converged values are then adopted as the initial guess for the next time level.
6. Then, repeat steps 1 to 5 until the solutions at all the required time levels have been obtained.
7. For the solutions at the remaining points at level k+1 (Fig. 1), compute them directly once according to the following sequence:

a) For points of type □, use the *rotated* five points approximation formula on the $\Omega_{\sqrt{2}h}$ grid:

$$u_{i,j} = \frac{1}{1+2r_2}\Big(\frac{r_2}{2}(u_{i-1,j-1} + u_{i+1,j-1} + u_{i-1,j+1} + u_{i+1,j+1}$$

$$+v_{i-1,j-1} + v_{i+1,j-1} + v_{i+1,j-1} + v_{i+1,j+1}) + (1-2r_2)v_{i,j} + \Delta t F_{i,j}\Big) \text{ , } r_2 = \frac{\Delta t}{2h^2} \tag{15}$$

b)For points of type △, use the standard five points approximation formula on the Ω_h grid:

$$u_{i,j} = \frac{1}{1+2r_3} X$$
$$\left(\frac{r_3}{2}(u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1} + v_{i-1,j} + v_{i+1,j} + v_{i,j-1} + v_{i,j+1}) + (1-2r_3)v_{i,j} + \Delta t F_{i,j}\right) \tag{16}$$

Here,

$$r_3 = \frac{\Delta t}{h^2}.$$

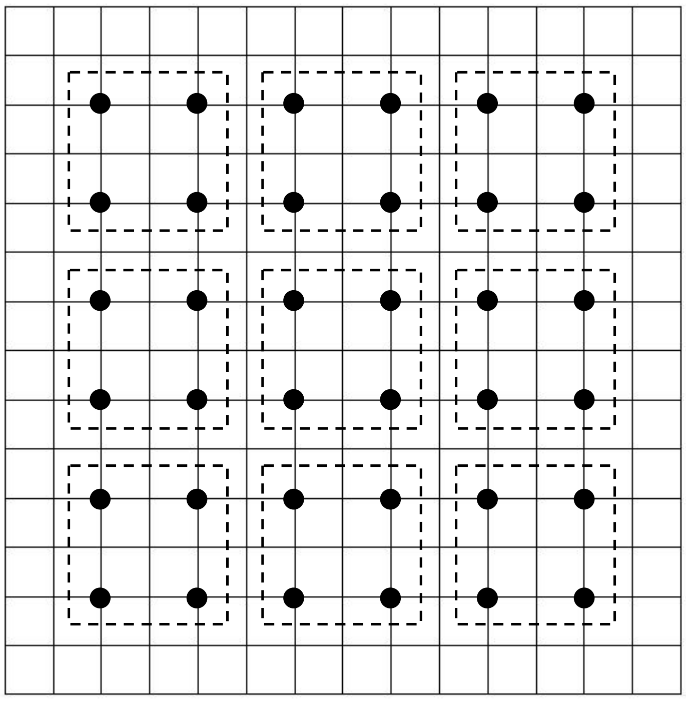


Fig. 2. Grid points which are involved in the iterative process at time level *k*+1.

2.2 Modified Explicit Decoupled Group (MEDG) AOR method

To formulate the MEDG (AOR) method for the diffusion equation, we consider the *rotated* five-point approximation formula for the diffusion equation with $2h$ spacing:

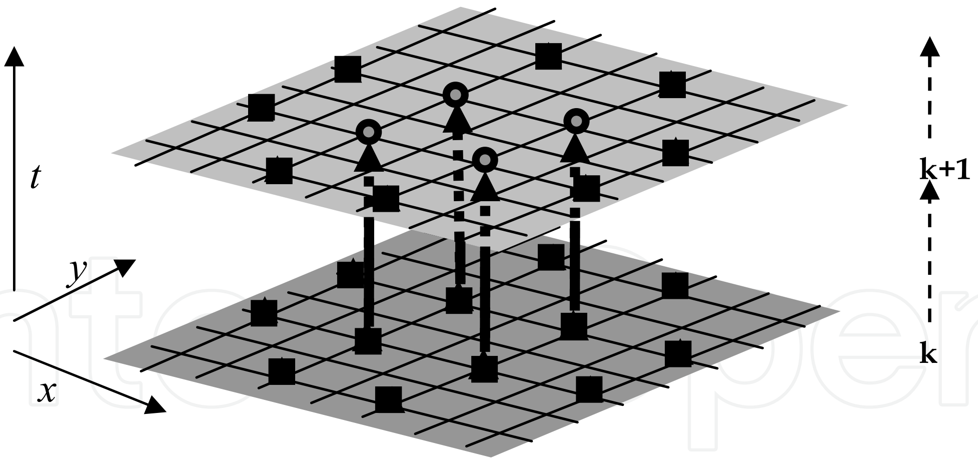


Fig. 3. Points involved in the updates of solutions in 4 points MEG (AOR).

$$u_{i,j}^{(m+1)} = \frac{1}{1+2r}\left(\frac{r}{2}(u_{i-2,j-2}^{(m)} + u_{i+2,j-2}^{(m)} + u_{i-2,j+2}^{(m)} + u_{i+2,j+2}^{(m)} \right.$$
$$\left. + v_{i-2,j-2} + v_{i+2,j-2} + v_{i-2,j+2} + v_{i+2,j+2}) + (1-2r)v_{i,j} + \Delta t F_{i,j}\right) \quad (17)$$

Here,

$$r = \frac{\Delta t}{4h^2}.$$

The method is constructed by firstly dividing the grid points into 4 types of points in a specific alternate ordering as shown in Fig. 5 in a unit square domain with n=14. The MEDG (AOR) formula for diffusion equation can then be obtained by applying equation (17) to groups of points of type ● in the solution domain. This application will produce a 4x4 system which can be inverted and rewritten in explicit decoupled form of two equations:

$$\begin{aligned} u_{i,j}^{(m+1)} &= s_1[w(s_3 b_3 + s_2 b_4) + R(s_3 t_1)] + (1-w)u_{i,j}^{(m)} \\ u_{i+2,j+2}^{(m+1)} &= s_1[w(s_2 b_3 + s_3 b_4) + R(s_2 t_1)] + (1-w)u_{i+2,j+2}^{(m)} \end{aligned} \quad (18)$$

where

$$\begin{aligned} b_1 &= s_4 v_{i,j} + \Delta t F_{i,j} \\ b_2 &= s_4 v_{i+2,j} + \Delta t F_{i+2,j+2} \\ b_3 &= s_2(u_{i-2,j-2}^{(m)} + u_{i+2,j-2}^{(m)} + u_{i-2,j+2}^{(m)} + v_{i-2,j-2} + v_{i+2,j-2} + v_{i-2,j+2} + v_{i+2,j+2}) + b_1 \\ b_4 &= s_2(u_{i+4,j+4}^{(m)} + u_{i+4,j}^{(m)} + u_{i,j+4}^{(m)} + v_{i+4,j+4} + v_{i+4,j} + v_{i,j+4} + v_{i,j}) + b_2 \\ t_1 &= u_{i-2,j-2}^{(m+1)} - u_{i-2,j-2}^{(m)} + u_{i+2,j-2}^{(m+1)} - u_{i+2,j-2}^{(m)} + u_{i-2,j+2}^{(m+1)} - u_{i-2,j+2}^{(m)} \end{aligned}$$

$$\begin{aligned} s_1 &= \frac{16}{15r^2 + 32r + 16} \\ s_2 &= 0.25r \\ s_3 &= 1 + r \\ s_4 &= 1 - r \end{aligned} \quad (19)$$

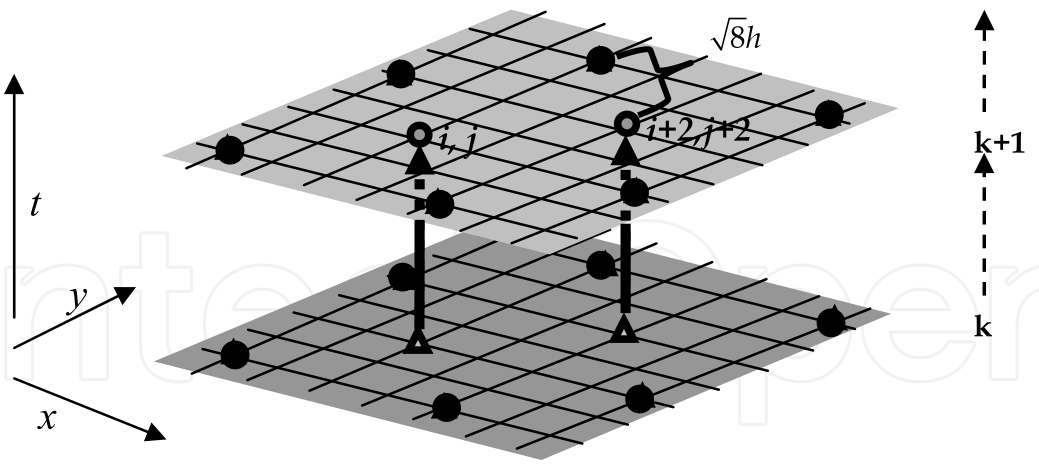


Fig. 4. Group of points involved in the iterative process with the spacing of 2*h* for MEDG (AOR) at time level *k*+1 and *k*.

The resulting grid or the computational molecule to update at the two points can be viewed in Fig. 4 with a mesh size 2*h*. From Fig. 5, it is obvious that the evaluation of equations (18) - (19) involves only points of type ●. This means that by using the approximation formulas (18)-(19), it is easy to see that the black filled points are linked only to the same type of points. Thus the iterative procedure involving these formulas can be performed independently of the other type of points. We can then formulate the four points MEDG (AOR) method as in **Algorithm 2**:

Algorithm 2

1. Divide the grid points at layer *k+1* into points of type ●, ○ ,△ and □ as shown in Fig. 5.
2. Set the initial guess for the iterations.
3. Evaluate the solution at the points of type ● iteratively at layer *k*+1 by using equations (18)-(19).
4. Check the convergence. If the iterations converge, go to step 5. Otherwise, repeat step 2 and step 3 until convergence is achieved.
5. After the solutions at points of type ● converge, the converged points are then adopted as initial guess for the next time level.
6. Then, repeat steps 1 to 5 until the solutions at all the required time levels have been obtained.
7. For the solutions at the remaining points at layer *k*+1 (Fig. 5), compute them directly once according to the following sequence:
   a. For points of type ○, use the standard five points approximation formula on the Ω_{2h} grid:

$$u_{i,j} = \frac{1}{1+2r_1}\left(\frac{r_1}{2}(u_{i-2,j}+u_{i+2,j}+u_{i,j-2}+u_{i,j+2}\right.$$

$$\left.+v_{i-2,j}+v_{i+2,j}+v_{i,j-2}+v_{i,j+2})+(1-2r_1)v_{i,j}+\Delta tF_{i,j}\right) \quad r_1=\frac{\Delta t}{4h^2} \tag{20}$$

b. For points of type □, use the *rotated* five points approximation formula on the $\Omega_{\sqrt{2}h}$ grid:

$$u_{i,j} = \frac{1}{1+2r_2}\left(\frac{r_2}{2}(u_{i-1,j-1} + u_{i+1,j-1} + u_{i-1,j+1} + u_{i+1,j+1}\right.$$

$$\left. + v_{i-1,j-1} + v_{i+1,j-1} + v_{i+1,j-1} + v_{i+1,j+1}) + (1-2r_2)v_{i,j} + \Delta t F_{i,j}\right),\ r_2 = \frac{\Delta t}{2h^2} \tag{21}$$

c. For points of type △, use the standard five points approximation formula on the Ω_h grid:

$$u_{i,j} = \frac{1}{1+2r_3}\left(\frac{r_3}{2}(u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1}\right.$$

$$\left. + v_{i-1,j} + v_{i+1,j} + v_{i,j-1} + v_{i,j+1}) + (1-2r_3)v_{i,j} + \Delta t F_{i,j}\right),\ r_3 = \frac{\Delta t}{h^2} \tag{22}$$

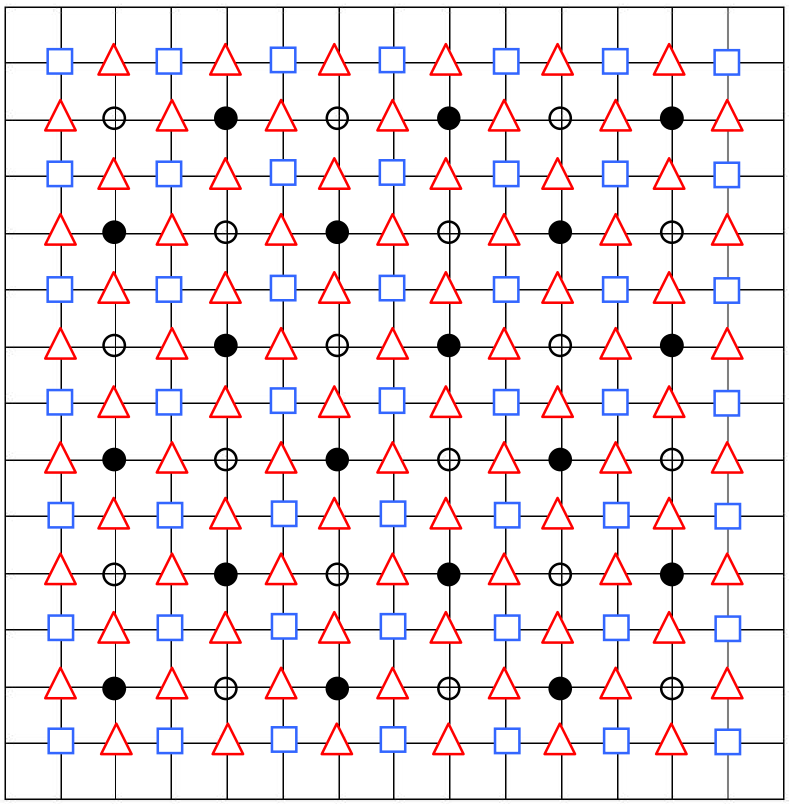


Fig. 5. The discretized domain of the four points MEDG (AOR) method at time level *k*+1.

3. Numerical experiments of the sequential group methods

In order to verify the performance of the proposed methods which were shown in previous sections, the algorithms were tested on the following model problem:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \sin x \sin y e^{-t} - 4 \tag{23}$$

with Dirichlet boundary conditions satisfying its exact solution

$$u(x,y,t) = \sin x \sin y e^{-t} + x^2 + y^2 ,$$

$(x,y) \in \partial\Omega$, $\partial\Omega$ is the boundary of the unit square Ω . The model equation (23) is of the form

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + g(x,y,t,u)$$

and is normally called a reaction-diffusion equation which models the movement of basic particles in sciences (for example, heat transfer, growth population, or dilution of chemical in water) in a region Ω . Here,

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$$

is the diffusion term which describes the movement of the particles and the remaining term at the right hand side of equation (23) (i.e. *g(x,y,t,u)*) is the reaction term which describes changes (due to birth, death, chemical reactions, etc.) occuring inside the region (or habitat). For this numerical experiment, we purposely find a model problem which has an exact solution to ensure that the proposed methods yield correct results. To terminate the iteration process, the relative error test, i.e. $Error = abs(u_{ij}^{(m+1)} - u_{ij}^{(m)}) / (1 + abs(u_{ij}^{(m+1)}))$, was used as the convergence test with tolerance $\varepsilon = 1.0 \times 10^{-6}$. As described in Section 2.1, the over-relaxation parameters, *R* and *w*, need to be found which give the best convergence rates for the proposed schemes. To achieve this, we obtained the values of *w* for the corresponding SOR scheme, and then the value of *R* was found by running the experiments using these specific values of *w* which gave the least number of iterations. Different grid sizes of *n* = 82, 102, 122, 142, 162, 182 and 202 were chosen to record the total iteration counts (*Iter*) at all time levels and computer timings (*t*) of the group AOR methods. The value of Δt = 0.0005 with 1000 time levels was used to run the programs.

The numerical results of the proposed MEG(AOR) and MEDG (AOR) methods together with the original explicit group methods EG(AOR), EDG(AOR) are tabulated in Tables 1 to 2. The point AOR method which uses the existing traditional *Crank-Nicolson* scheme (5) accelerated with the AOR technique is also shown in Table 3 for comparison purposes . The value of *R* was chosen experimentally to be close to the value of w as depicted in the tables. All of the methods tested are of second order accuracies so that the results they produce are of similar accuracies as seen in Table 4. From Tables 1 and 2, it can be seen that between EG(AOR) and EDG(AOR), the latter has better rates of convergence which is consistent with the results in Ali and Lee (2007) for the elliptic problem. The diffusion equation is a time dependent parabolic equation where each time level represents an elliptic problem. In these tables, it can also be observed that both of the proposed MEG (AOR) and MEDG (AOR) methods have better execution times than the original EG (AOR) and EDG (AOR) respectively which is due to the reduction in computing efforts of the proposed methods. In the proposed modified group schemes, lesser grid points are involved in the iterative processes than the original group schemes which result in lesser overall arithmetic operation counts

	EG(AOR)				**MEG(AOR)**			
n	*R*	*w*	*Iter*	*t*	*R*	*w*	*Iter*	*t*
82	1.308	1.3128-1.3134	13472	13.75	1.109	1.110-1.111	7421	6.766
102	1.3825	1.3843-1.3849	16613	21.875	1.1645	1.1646-1.1656	8703	9.469
122	1.44175	1.44561-1.4458	19841	34.375	1.2134	1.216-1.219	10410	13.453
142	1.49475	1.49903-1.4992	23523	51.172	1.26475	1.2672-1.2676	11672	19.015
162	1.5319	1.5374-1.5375	26931	72.11	1.308	1.3128-1.3134	13472	26.36
182	1.57195	1.5761-1.5762	30634	102.578	1.348	1.3495-1.3499	14914	33.344
202	1.60085	1.60397-1.60402	34307	133.375	1.3825	1.3839-1.385	16614	41.453

Table 1. The numerical performances of the EG(AOR) and MEG(AOR) methods.

	EDG(AOR)				**MEDG(AOR)**			
n	*R*	*w*	*Iter*	*t*	*R*	*w*	*Iter*	*t*
82	1.2305	1.238-1.239	10398	9.328	1.066	1.064-1.069	6208	5.89
102	1.302	1.3114-1.3117	12564	14.157	1.105	1.104-1.11	7239	8.89
122	1.367	1.3675-1.3678	14879	19.891	1.15	1.15-1.154	8286	12.391
142	1.4207	1.4213-1.4217	17448	29.406	1.19	1.191-1.195	9343	15.641
162	1.467	1.4694-1.4702	19853	41.235	1.23	1.237-1.24	10396	20.672
182	1.5075	1.50808-1.5082	22344	52.672	1.27	1.2773-1.2786	11480	26.266
202	1.54	1.5449-1.5453	24927	67.266	1.3017	1.311-1.312	12561	32.187

Table 2. The numerical performances of EDG(AOR) and MEDG(AOR) methods

Point C-N (AOR)				
n	*R*	*w*	*Iter*	*t*
82	1.34545	1.35832	15706	25.469
102	1.43985	1.4307	19127	43.906
122	1.4833	1.51066	23884	72.182
142	1.53835	1.5574	27683	101.11
162	1.58092	1.588	30430	148.813
182	1.6301	1.63048	34878	212.141
202	1.66	1.66004	38945	282.328

Table 3. The numerical performance of point *Crank-Nicolson* (C-N) AOR method.

Average Errors					
n	C-N AOR	EG AOR	EDG AOR	MEG AOR	MEDG AOR
82	5.76E-04	2.43E-04	4.13E-04	1.42E-04	5.25E-04
102	5.78E-04	2.31E-04	3.91E-04	1.41E-04	5.16E-04
122	6.35E-04	2.23E-04	3.74E-04	1.38E-04	5.06E-04
142	7.44E-04	2.24E-04	3.64E-04	1.35E-04	4.94E-04
162	8.96E-04	2.35E-04	3.66E-04	1.32E-04	4.81E-04
182	1.07E-03	2.58E-04	3.82E-04	1.29E-04	4.67E-04
202	1.26E-03	2.91E-04	4.11E-04	1.26E-04	4.55E-04

Table 4. Average errors of all the methods for different mesh sizes.

From Tables 1 and 2, it can also be concluded that MEDG (AOR) is able to show the most substantial reduction in execution times compared with the other group and point AOR schemes without having to jeorpadize the solution accuracies. MEDG (AOR) requires the least number of total iterations and computing timings to converge. The required number of iterations is reduced because the introduction of the over-relaxation parameters, *w* and *R*, into its formulas is able to reduce the most the number of iterations of the scheme compared with the other schemes tested. This combined with the fact that only about 1/8 of the total nodal points are involved in the iterative process at each time level results in the least computing times for this method. In summary, the proposed MEG (AOR) and MEDG (AOR) methods are viable alternative solvers to the diffusion equation with the latter being the more efficient one in terms of CPU times.

4. Parallel implementations for the group methods

This section will discuss the implementation of the proposed group methods on a message-passing environment.

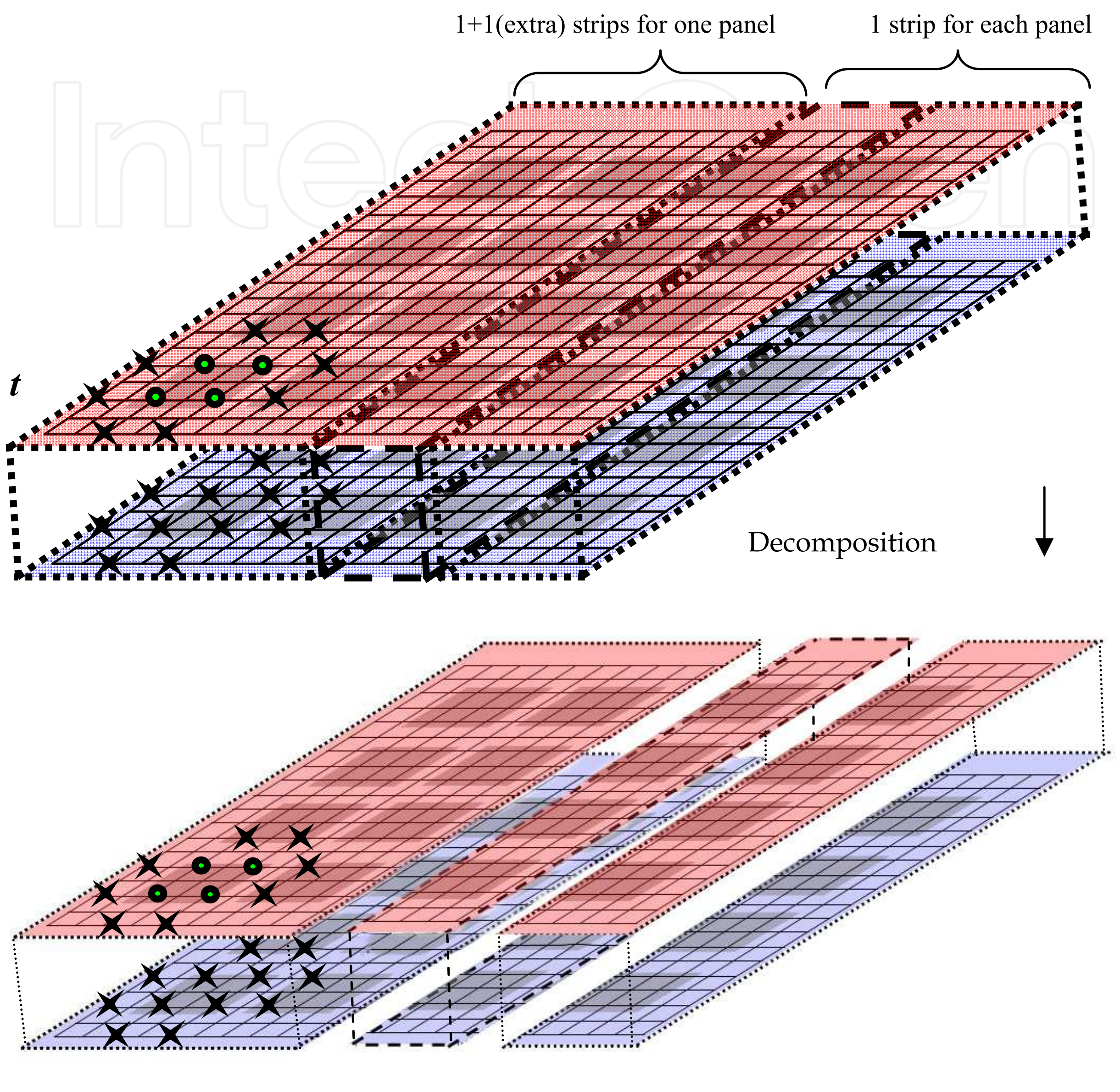


Fig. 6. Domain Decomposition For MEG (AOR) for the case n = 18 and p = 3

4.1 MEG (AOR) in parallel

For the MEG (AOR) method, we decompose the domain Ω into a number of vertical panels at layer k+1 based on the number of available processors, p. The idea is to allocate approximately equal number of strips to the processors. Each strip consists of four grid lines which form the four points blocks with the spacing of $2h$ between the points. The equal number of vertical strips in each panel can be approximated using a specific formula. The distribution of tasks (panels) to processors for the case n=18 and p= 3 is as shown in Fig. 6 where the configuration is as follows:

- Number of panels = Number of processors, p = 3.
- Number of strips in a panel = (n-2) / 4p = (18-2)/12 =1.
- Number of panels that have an extra one strip = ((n -2) % 4p)/2 = ((18-2) %12)/4 = 1.

As shown in Fig. 6, we distribute 1+1(extra strip) strips into panel 1. The other panels (panel 2 and 3) will be allocated with 1 strip of values to update.

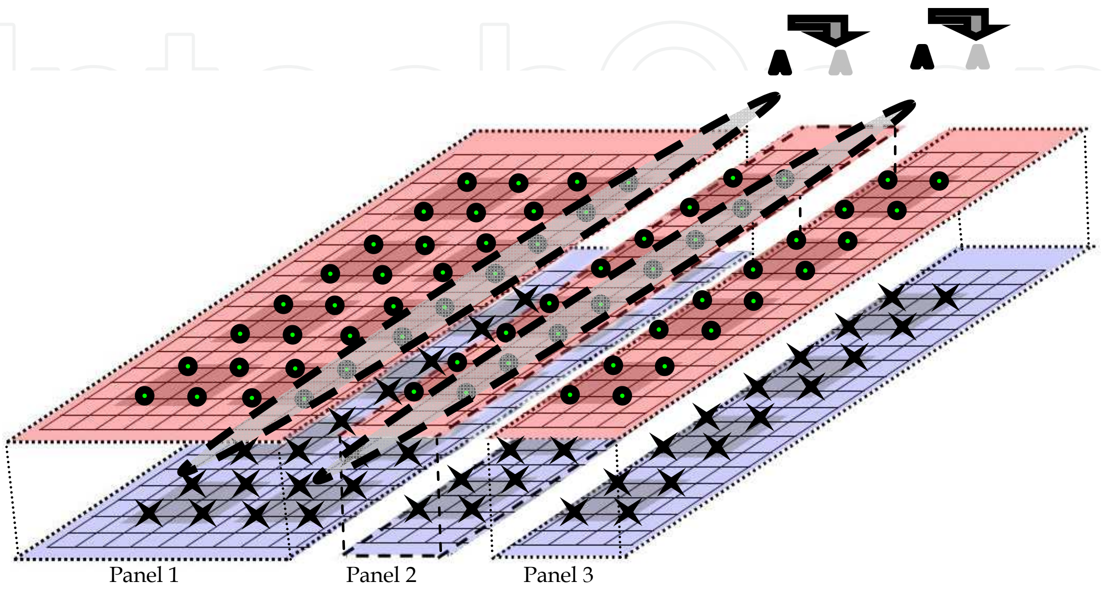


Fig. 7. Send right boundary cell values (grid A's) to left adjacent neighbouring panel.

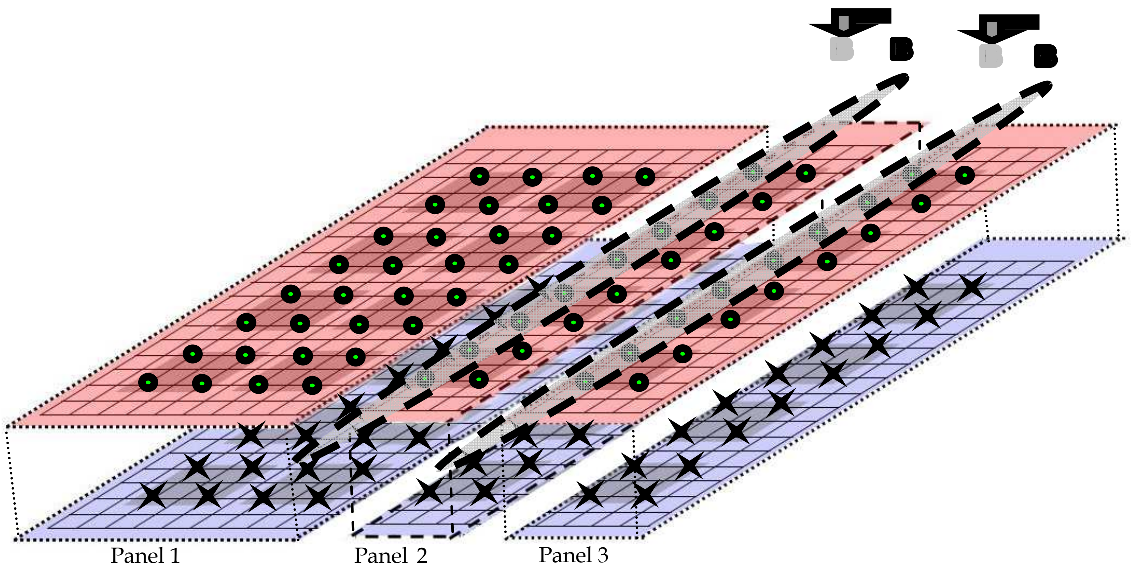


Fig. 8. Send left boundary cell values (grid B's) to right adjacent neighbouring panel.

After the domain Ω is decomposed into the individual panels, message passing needs to be done between the processors to send and receive data at the right and left boundaries of each panel. Based on equations (13)-(14), certain values from adjacent processors need to be communicated during the iterative cycle. The right boundary cell values, grid A's (panels 1 and 2) will be sent to the left adjacent neighbouring panels (panels 2 and 3) as shown in Fig. 7. The left boundary cell values, grid B's (panels 2 and 3) will be sent to the right adjacent neighbouring panel (panels 1 and 2) as shown in Fig. 8. These communications need to be executed correctly to ensure that each processor possesses the correct values needed for their respective independent calculations. After the message passing process is completed, the local error for each processor is calculated and is sent to the master processor for the global convergence check. The local convergence test used is the relative error test similar with their sequential counterparts. The global error is the sum of the local error from each processor. If the global error is greater than a certain tolerance ε, then the iteration is repeated.

4.2 MEDG (AOR) in parallel

Similar with the MEG (AOR) method, we decompose the spatial domain Ω into a number of vertical panels based on the number of available processors, p. For MEDG (AOR), we rotate the x-y axis clockwise 45^0 and forms the four points block with the spacing of $\sqrt{8}h$ between the points of the matrices.

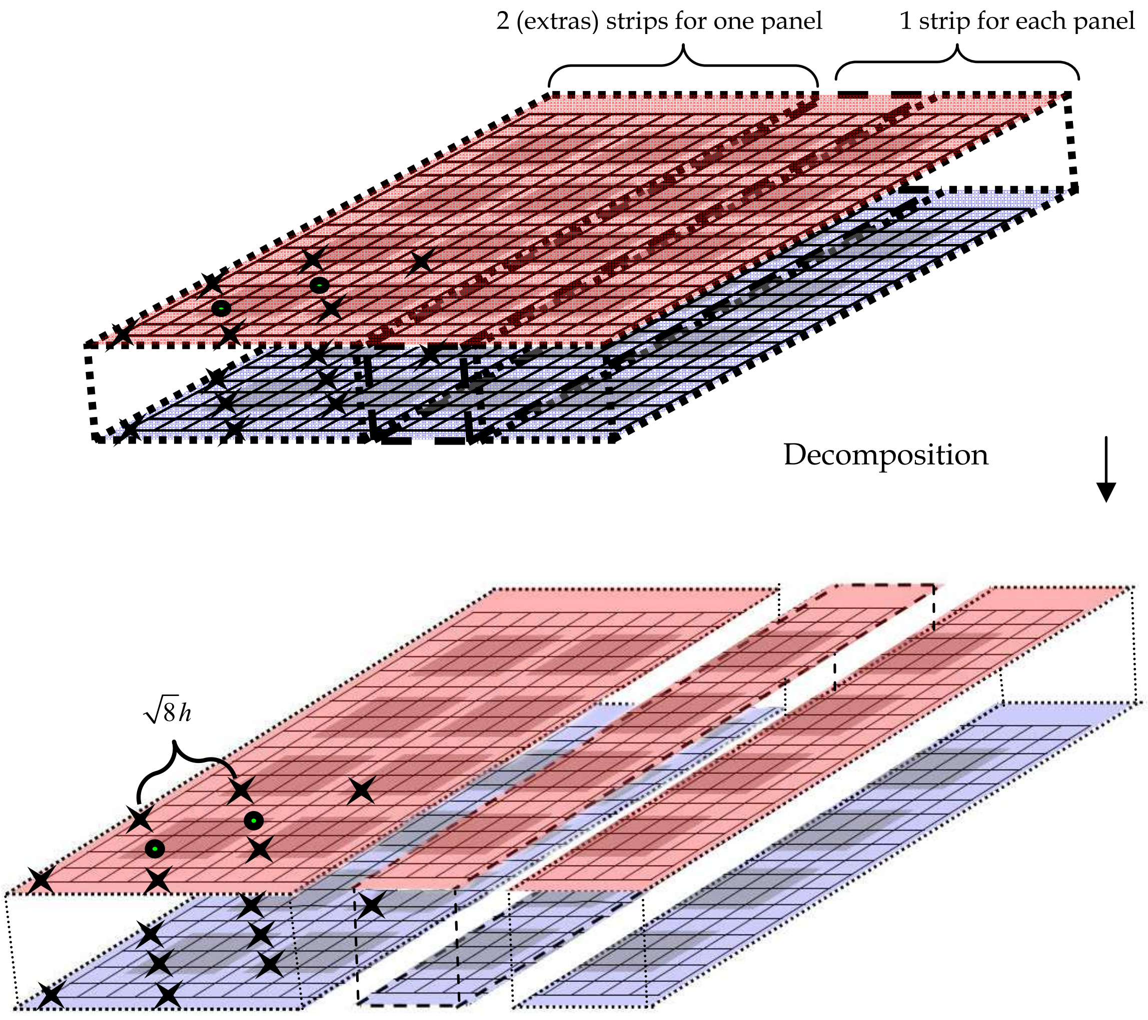


Fig. 9. Domain Decomposition For MEDG (AOR).

We again consider the ordering of the strips for the case n=18 and p= 3 as shown in Fig. 9. Each strip consists of four grid lines which form the four-point groups with spacing 2h. We will distribute 1+1(extra strip) strips into panel 1. The other panels (panels 2 and 3) will be allocated with 1 strip each to ensure that the tasks are distributed almost equally amongst the processors. The number of strips for each panel (processor) will be computed similar as the one in the previous method.

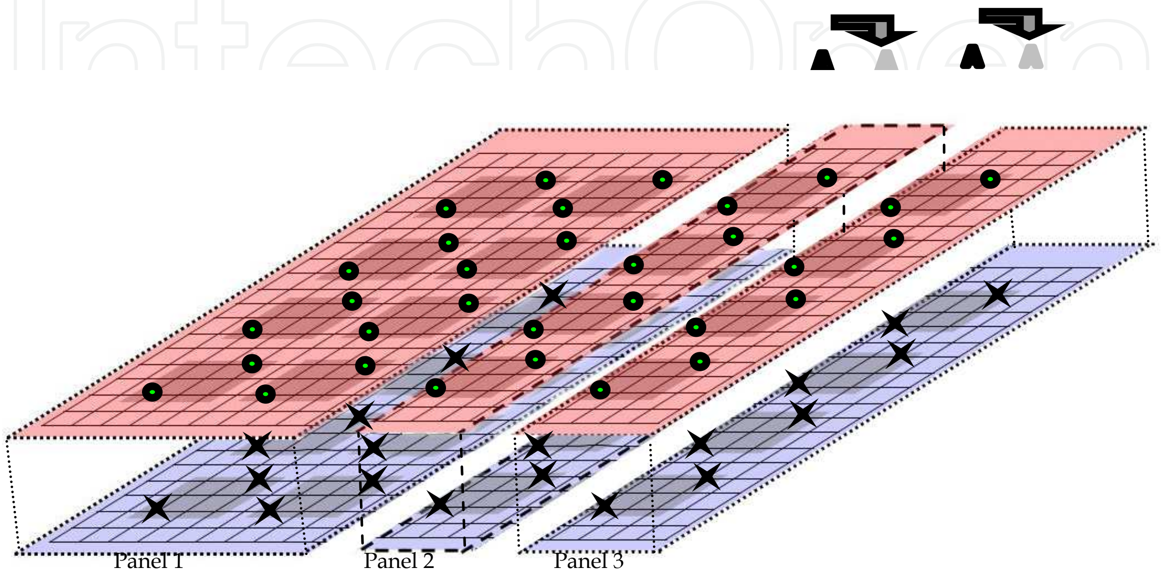


Fig. 10. Send right boundary cell values (grid A's) to the left adjacent neighbouring panel.

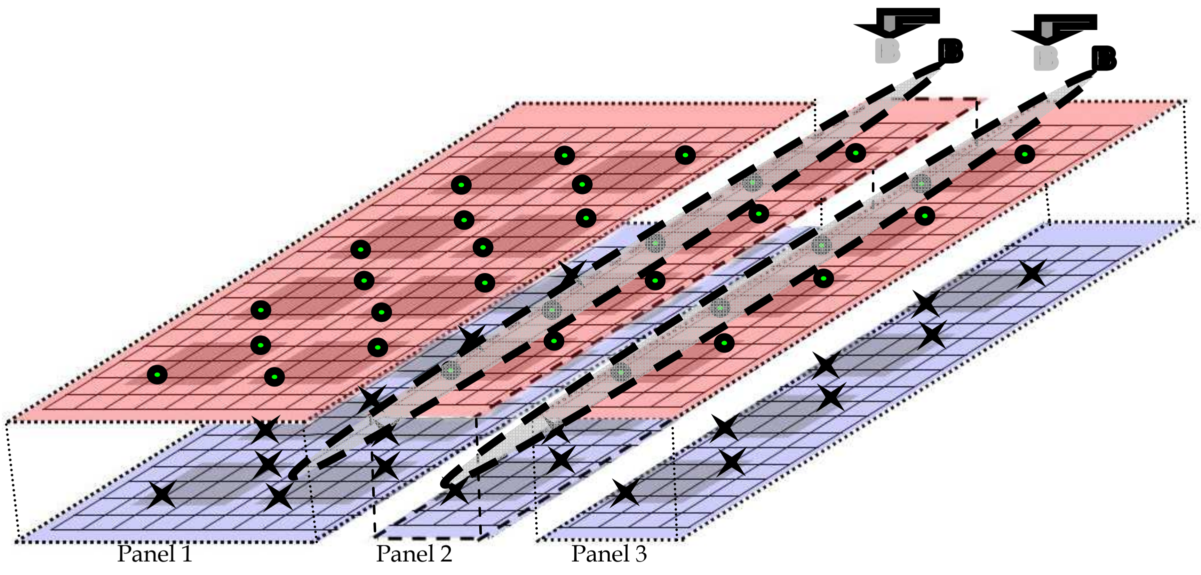


Fig. 11. Send left boundary cell values (grid B's) to the right adjacent neighbouring panel.

After the domain Ω is decomposed into the individual panels, message passing needs to be done between the processors to send and receive data at the right and left boundaries of each panel. The points involved in the iterative process are different from the ones in the previous method due to their different computational molecules. From equations (18)-(19), we can determine these specific values that need to be communicated between adjacent panels during the iterative cycle as shown in Fig. 10 and Fig. 11. The local and global convergences are checked the same way as in MEG.

5. Performance analysis of the parallel group methods

We assume that there are q^2 internal mesh points where q=n-1 and arithmetic operations estimates for each method are made per iteration. We assume that the values $r, \Delta t F_{i,j}$, 1-w, $s_1, s_2, s_3, s_4, s_5, s_6$ are stored beforehand. To update a single block in MEG (AOR) method, the computing cost is given by equation (24):

$$t_{\text{meg(aor)-update}} = 61t_a + 53t_m \tag{24}$$

with t_a = the cost of the addition for a double point and t_m = the cost of the multiplication for a double point. Here, we will consider the problem size, n, and number of processors, p, to have a complete iterative step for the computational cost of MEG (AOR) method which is given by

$$t_{\text{meg(aor)-comp}} = \frac{(q-1)^2}{16p} t_{\text{meg(aor)-update}} \tag{25}$$

The transition cost for message passing of double-type data in a distributed memory multicomputer is given by

$$t_{\text{send}} = t_s + q t_d \tag{26}$$

where t_s is the startup time, and t_d is the sending time for a double-type data. The computation of the MEG (AOR) formula requires that q points to be passed to the adjacent processor in an iteration. Therefore, the total communication cost of MEG (AOR) method in a single iterative step, consisting of two sequential point-to-point communications and one global collective communication, is given by

$$t_{\text{meg(aor)-comm}} = 4t_s + 4q t_d \tag{27}$$

After the message passing process is completed, the local error for each processor, p, is calculated and sent to the master processor for the global convergence check. Therefore,

$$t_{\text{meg(aor)-global}} = \frac{(q-1)^2}{4p}(2t_a + t_m) + p(t_s + t_d) \tag{28}$$

The total costs of iterations in parallel MEG (AOR) method is

$$t_{\text{meg(aor)}} = t_{\text{meg(aor)-comp}} + t_{\text{meg(aor)-comm}} + t_{\text{meg(aor)-global}} \tag{29}$$

After completing the iteration process, we need to compute at the remaining points by using the *rotated* five points formula for points of type□ and standard 5-points formula for points of type △. This process will be done directly once and the cost of these processes is

$$t_{\text{meg(aor)-once}} = \frac{(q+1)^2}{4} t_{\text{meg(aor)-rotated}} + \frac{(q^2\text{-}1)}{2} t_{\text{meg(aor)-standard}} \tag{30}$$

where

$$t_{\text{meg(aor)-rotated}} = 9t_a + 3\, t_m$$

and

$$t_{\text{meg(aor)-standard}} = 9t_a + 3\, t_m .$$

For the MEDG (AOR) method, we assume that the values *r*, $\Delta tF_{i,j}$, 1-*w*, s_1, s_2, s_3, s_4 are stored beforehand. To update a single block in this method, the cost is as follows:

$$t_{\text{medg(aor)-update}} = 27t_a + 18t_m \tag{31}$$

with t_a = the cost of the addition for a double point and t_m = the cost of the multiplication for a double point. We will also consider the problem size, *n*, and number of processors, *p*, to have a complete iterative step for the computational cost of MEDG (AOR) method which is given by

$$t_{\text{medg(aor)-comp}} = \frac{(q-1)^2}{16p} t_{\text{medg(aor)-update}} \tag{32}$$

The transition cost for message passing of double-type data in a distributed memory multicomputer is given by

$$t_{\text{send}} = t_s + qt_d \tag{33}$$

where t_s is the startup time, and t_d is the sending time for a double-type data. The execution of the MEDG (AOR) formula requires that *q* points to be passed to the adjacent processor in an iteration. Therefore, the total communication cost of MEDG (AOR) method in a single iterative step, consisting of two sequential point-to-point communications and one global collective communication, is given by

$$t_{\text{medg(aor)-comm}} = 4t_s + 4qt_d \tag{34}$$

After the message passing process is completed, the local error for each processor, *p* is calculated and is sent to the master processor for the global convergence check. Therefore,

$$t_{\text{medg(aor)-global}} = \frac{(q-1)^2}{8p}(2t_a + t_m) + p(t_s + t_d) \tag{35}$$

As such, the total costs of iteration in MEDG (AOR) method in parallel is

$$t_{\text{medg(aor)}} = t_{\text{medg(aor)-comp}} + t_{\text{medg(aor)-comm}} + t_{\text{medg(aor)-global}} \tag{36}$$

After completing the iteration process, we need to compute the solutions at the remaining points using the standard 5-points formula with the spacing of 2h for the points ○, rotated 5-points formula for □ and standard 5-points method for △. This process will be done only once directly and the cost of computing these values is

$$t_{\text{medg(aor)-once}} = \frac{(q-1)^2}{8} t_{\text{medg(aor)-standard\_2h}} \frac{(q+1)^2}{4} t_{\text{medg(aor)-rotated}} + \frac{(q^2-1)}{2} t_{\text{medg(aor)-standard}} \quad (37)$$

where

$$t_{\text{medg(aor)-standard\_2h}} = 9t_a + 3t_m \ , \ t_{\text{medg(aor)-rotated}} = 9t_a + 3\,t_m$$

and

$$t_{\text{medg(aor)-standard}} = 9t_a + 3t_m \ .$$

5.1 Benchmarking

Although it is difficult to obtain reliable estimates for various parameters in any performance models, we run several benchmarking tests on the computing cluster available at the School of Computer Science, Universiti Sains Malaysia (USM), in which the experiments of explicit group methods were carried out. This process is to ensure that we could get the best benchmark with more tests on different time. The specifications of the clusters are shown as below:

a. Stealth cluster consists of 1 unit of PC with two 900 MHz CPUs, 2GB RAM, and
b. 6 units of PCs each with two 1002 MHz CPUs, 2 GB RAM.
c. Solaris9 (SunOS 2.9) with Sun HPC ClusterTools 5 and Sun MPI 6.0.

Performance parameter	Benchmark in Stealth cluster
MEG (AOR) point update cost, $t_{\text{meg(aor)-update}}$	1.53 µs/block
MEDG (AOR)point update cost, $t_{\text{medg(aor)-update}}$	1.43 µs/block
sending startup time, t_s	2.3 µs
sending word cost, t_d	0.033 µs/point
global convergence check cost, $t_{\text{meg(aor)-global}}$ and $t_{\text{medg(aor)-global}}$	0.066 µs/point + 2.333 µs/proccessor

Table 5. Performance parameters benchmarking in Stealth Cluster, USM.

5.2 Scalability analysis

By referring to the parameter values in Table 5, we form the performance models for MEG (AOR) and MEDG (AOR) methods which are shown as the following:

a) MEG (AOR): $$t_{\text{method}}(n,p) = A\frac{(q-1)^2}{16p} + B + Cq + D\frac{(q-1)^2}{4p} + Ep, \quad (38)$$

b) MEDG (AOR): $$t_{\text{method}}(n,p) = A\frac{(q-1)^2}{16p} + B + Cq + D\frac{(q-1)^2}{8p} + Ep \quad (39)$$

where A, B, C, D and E are coefficients of the methods which shown in Table 6.

Method	A	B	C	D	E
MEG (AOR)	1.53	9.2	0.132	0.066 µs	2.333
MEDG (AOR)	1.43	9.2	0.132	0.066 µs	2.333

Table 6. Coefficients of the performance models in µs.

6. Numerical results of the parallel group accelerated methods

We implement the parallel algorithms on the Stealth cluster at USM. The experiments were carried out on 1 unit of PC with two 900 MHz CPUs, 2 GB RAM, and 6 units of PC, where each PC had two 1002 MHz CPUs and 2 GB RAM. The Operating System used was Solaris9 (SunOS 2.9) with Sun HPC ClusterTools 5 and Sun MPI 6.0. The parallel algorithms were tested on the same model problem that was used for the sequential version (23). For the MEG (AOR) and MEDG (AOR) methods, the sizes of n were chosen appropriately to make sure that all of the strips consisting of two grid lines can be decomposed approximately evenly to the 6 processors. The tolerance used was $\varepsilon = 1.0 \times 10^{-6}$ and the acceleration parameters, w and R, were chosen to give the least number of iterations.
From Table 6, we can see that the computation coefficient of MEDG (AOR) is slightly lesser than MEG (AOR). Therefore we expect that MEDG (AOR) should have better timings if compared to MEG (AOR). We further test the scalability analysis by comparing the experimental and predicted timings of these methods using n = 182 and 202 which are shown in Figs. 12 and 13 respectively.

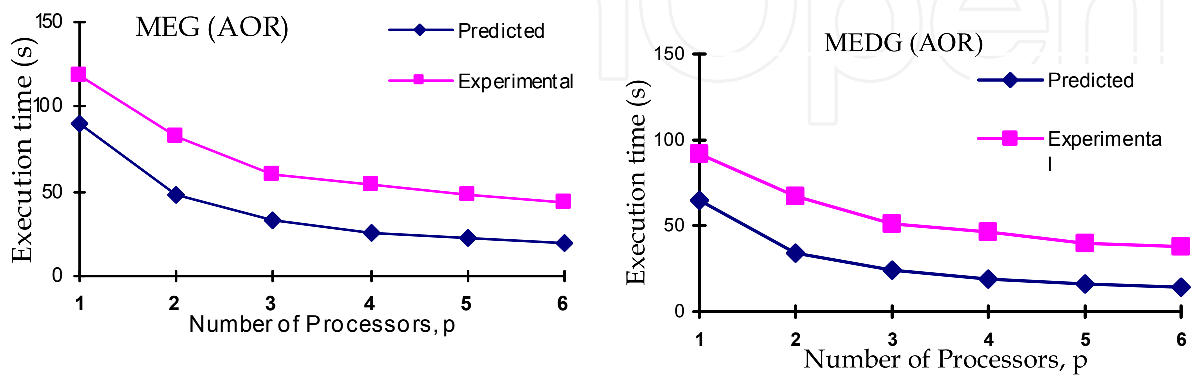
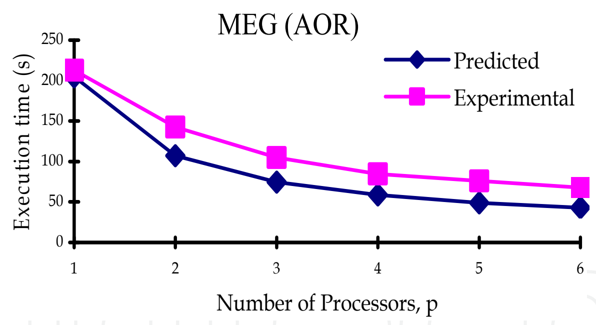


Fig. 12. Comparison of Predicted Timings and Experimental Timings of parallel MEG (AOR) and MEDG (AOR) methods for n = 182.



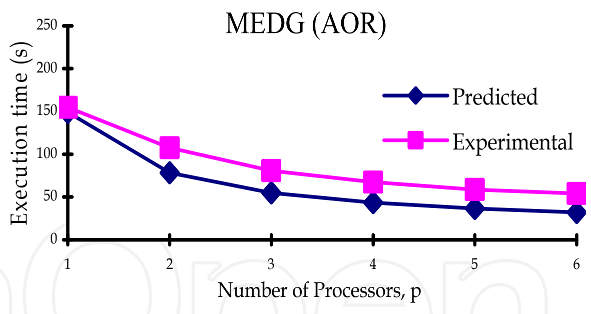
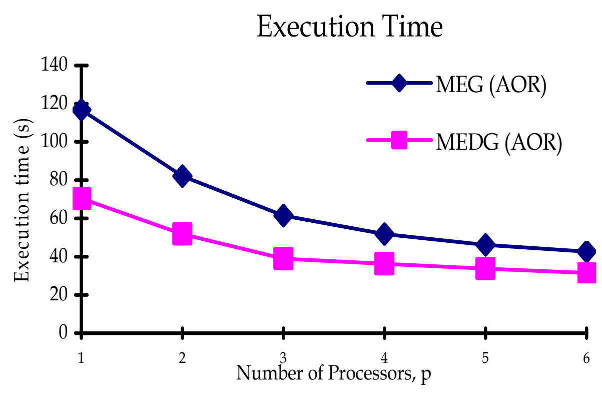


Fig. 13. Comparison of Predicted Timings and Experimental Timings of parallel MEG (AOR) and MEDG (AOR) methods for n = 202.

The figures show that the experimental and predicted timings are very close to one another especially when the grid size is larger. Comparing between the two grid sizes, it is found that the efficiency improves as the grid size increases. This improvement indicates that the performance models are more accurate as the grid sizes increase. Based on the parallel implementation which was described in Section 4, we used the size of *n* = 162, 182 and 202 to record the timings, speedups and efficiencies of both the MEG (AOR) and MEDG (AOR) methods. Several performance results of the MEG (AOR) and MEDG (AOR) methods are shown in Figs. 14-16.



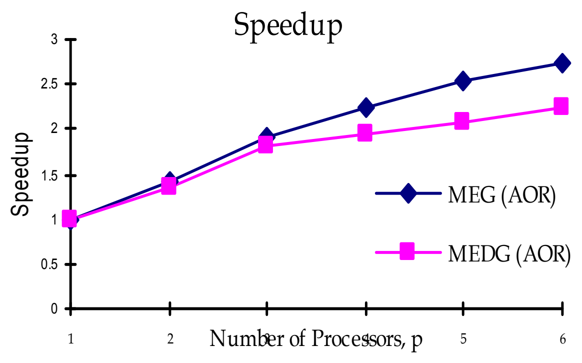
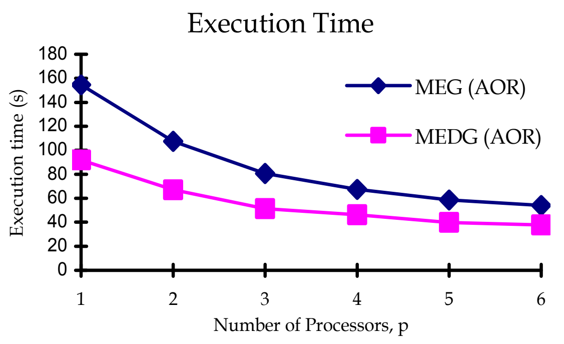


Fig. 14. Comparisons of execution time (Left) and speedup values (Right) between MEG(AOR) and MEDG(AOR) methods for *n* = 162.



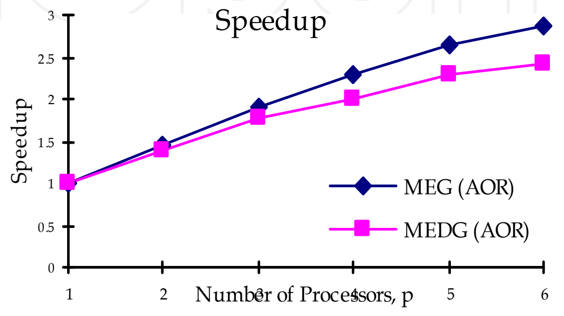
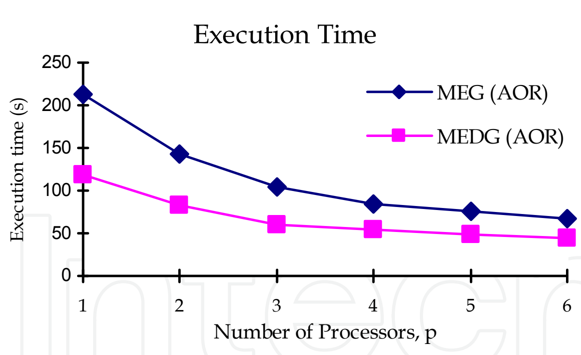


Fig. 15. Comparisons of execution timings (Left) and speedup values (Right) between parallel MEG(AOR) and MEDG(AOR) methods for *n* = 182.



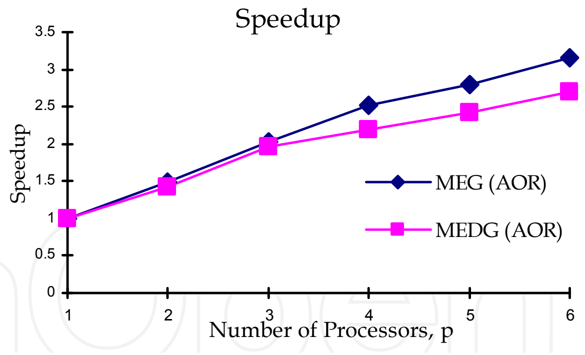


Fig. 16. Comparisons of execution timings (Left) and speedup values (Right) between parallel MEG(AOR) and MEDG(AOR) methods for n = 202.

From these figures, we can see that the parallel MEDG (AOR) is better in execution timings compared to the MEG (AOR). Generally we can see that with the enhancement of grid size, the speedup increases with nearly 70% efficiency. However, the speedup and efficiency values of MEG (AOR) are slightly better than MEDG (AOR). This difference in values indicates that the amount of computations carried out over the total communication overheads in MEG(AOR) is greater than the one in MEDG(AOR).

7. Conclusions

In this chapter, the formulation of new improved explicit group AOR methods in solving the two dimensional diffusion equation is presented. The improvement of the numerical result shows the potential of these methods in solving the parabolic equation. We further implement both of these methods on a cluster of distributed memory computer using Message-Passing Interface programming environment. The experimental results show that these two methods can be performed successfully in parallel on a cluster of distributed memory computer. Performance models to explain the parallel behaviour of these proposed methods were also developed. The experimental timings agreed with the predicted results especially when the grid size and processors increase. The MEDG (AOR) shows a faster rate of convergence with similar accuracies if compared with MEG (AOR), especially when the grid size increases. Both methods were shown to be suitable to be programmed on a distributed memory computer.

8. Acknowledgement

The authors acknowledge the Fundamental Research Grant Scheme (203/ PMATHS/ 6711188) for the completion of this work.

9. References

Abdullah, A. R. (1991). The Four Point Explicit Decoupled Group (EDG) Method: A Fast Poisson Solver. *International Journal of Computer Mathematics.* Vol. 38, pp. 61-70

Ali, N. H. M. & Lee, S. C. (2007). Group Accelerated Over Relaxation Methods On Rotated Grid. *Applied Mathematics and Computation*, Vol. 191, pp. 533-542

Ali, N,H.M. (1998). *The Design And Analysis of Some Parallel Algorithms For The Iterative Solution of Partial Differential Equations*, PhD Thesis, Universiti Kebangsaan Malaysia.

Evans, D.J. & Abdullah, A.R. (1983a). A New Explicit Method for the Solution of . *International Journal of Computer Mathematics*, Vol. 14, pp. 325-353

Evans, D.J. & Abdullah, A.R. (1983b). Group Explicit Methods for Parabolic Equations. *International Journal of Computer Mathematics.* Vol. 14, no. 1, pp. 73-105

Evans, D.J. & Sahimi, M. S. (1988). The Alternating Group Explicit(AGE) Iterative Method for Solving Parabolic Equations, 1-2 Dimensional Problems. *International Journal of Computer Mathematics*, Vol. 24, pp. 250-281

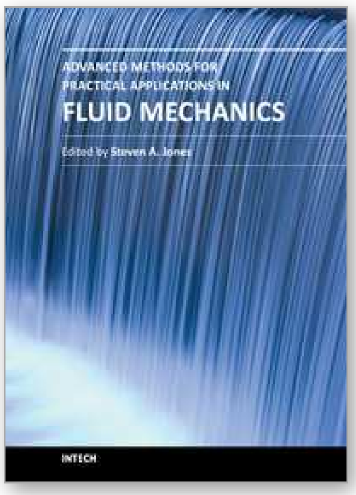
Hadjidimos, A. (1978). Accelerated OverRelaxation method. *Mathematics of Computation,* Vol. 32, pp. 149–157

Martins, M.M., Yousif, W.S. & Evans, D.J. (2002). Explicit group AOR method for solving elliptic partial differential equations. *Neural, Parallel and Science Computation,* Vol. 10, no. 4, pp. 411-422

Ng, K. F. & Ali, N. H. M. (2008). Performance Analysis of Explicit Group Parallel Algorithms for Distributed Memory Multicomputer. *Parallel Computing,* Vol. 34, no (6-8), pp. 427-440

Othman, M. & Abdullah, A.R. (2000). An Efficient Four Points Modified Explicit Group Poisson Solver. *International Journal of Computer Mathematics,* Vol. 76, pp. 203-217.

Yousif, W.S. & Evans, D.J. (1986). Explicit Group Over-relaxation Methods for Solving Elliptic Partial Differential Equations. *Mathematics & Computers In Simulation*, Vol. 28, pp. 453-466.



Advanced Methods for Practical Applications in Fluid Mechanics
Edited by Prof. Steven Jones

ISBN 978-953-51-0241-0
Hard cover, 230 pages
Publisher InTech
Published online 14, March, 2012
Published in print edition March, 2012

Whereas the field of Fluid Mechanics can be described as complicated, mathematically challenging, and esoteric, it is also imminently practical. It is central to a wide variety of issues that are important not only technologically, but also sociologically. This book highlights a cross-section of methods in Fluid Mechanics, each of which illustrates novel ideas of the researchers and relates to one or more issues of high interest during the early 21st century. The challenges include multiphase flows, compressibility, nonlinear dynamics, flow instability, changing solid-fluid boundaries, and fluids with solid-like properties. The applications relate problems such as weather and climate prediction, air quality, fuel efficiency, wind or wave energy harvesting, landslides, erosion, noise abatement, and health care.