International Journal of Engineering, Mathematical and Physical Sciences
ISSN: 2517-9934
Vol:5, No:12, 2011

Parallel Explicit Group Domain Decomposition Methods for the Telegraph Equation

Kew Lee Ming and Norhashidah Hj. Mohd. Ali

***Abstract*—**In a previous work, we presented the numerical solution of the two dimensional second order telegraph partial differential equation discretized by the centred and rotated five-point finite difference discretizations, namely the explicit group (EG) and explicit decoupled group (EDG) iterative methods, respectively. In this paper, we utilize a domain decomposition algorithm on these group schemes to divide the tasks involved in solving the same equation. The objective of this study is to describe the development of the parallel group iterative schemes under OpenMP programming environment as a way to reduce the computational costs of the solution processes using multicore technologies. A detailed performance analysis of the parallel implementations of points and group iterative schemes will be reported and discussed.

***Keywords*—**Telegraph equation, explicit group iterative scheme, domain decomposition algorithm, parallelization.

I. Introduction

CONSIDER the telegraph equation which is a hyperbolic, second order partial differential equation (PDE) defined in the rectangular region $\Omega = \{(x, y, t) \mid 0 < x, y < 1, t > 0\}$ of the following form:

$$\frac{\partial^2 U}{\partial t^2}(x,y,t) + 2\alpha \frac{\partial U}{\partial t}(x,y,t) + \beta^2 U(x,y,t) = A\frac{\partial^2 U}{\partial x^2}(x,y,t) + B\frac{\partial^2 U}{\partial y^2}(x,y,t) + F(x,y,t) \quad (1)$$

where $\alpha(x,y,t) > 0, \beta(x,y,t) \geq 0, A(x,y,t) > 0, B(x,y,t) > 0$, with the initial and Dirichlet boundary conditions are given by

$$U(x,y,0) = f_1(x,y); U_t(x,y,0) = f_2(x,y)$$
$$U(0,y,t) = g_1(y,t); U(1,y,t) = g_2(y,t);$$
$$U(x,0,t) = g_3(x,t); U(x,1,t) = g_4(x,t).$$

Let $k > 0$ and $h > 0$ be the time step and space step, respectively. We discretized the Ω uniformly in both x and y directions with a mesh size $h = 1/n$ where n is an arbitrary positive integer. The grid points are given by $(x_i, y_j, t_m) \equiv (ih, jh, mk)$ where $m = 1, 2, 3, \ldots$. Let $U_{i,j}^m$ be the exact solution of the differential equation and $u_{i,j}^m$ be the computed solution of the approximation method at the grid point (x_i, y_j, t_m).

L. M. Kew is a doctoral candidate at the School of Mathematical Sciences, Universiti Sains Malaysia, 11800 Penang, Malaysia (corresponding author e-mail: leeming_kew@hotmail.com).

Norhashidah Hj. M. Ali is a permanent staff of School of Mathematical Sciences, Universiti Sains Malaysia, 11800 Penang, Malaysia. (e-mail: shidah@cs.usm.my).

There are various ways to discretize Equation (1), where this equation is commonly encountered in physics and engineering mathematics. In [1], two new explicit group relaxation methods derived from the standard and rotated five-point difference approximation are used to solve the two dimensional second order hyperbolic telegraph equation (1). These explicit group methods were developed using small fixed size group strategy which required lesser execution time than the classic point iterative methods. The aim of this paper is to develop a more economical computational solution for Equation (1) by utilizing a domain decomposition strategy [2] to divide the discretized solution domain, and parallelize it using OpenMP programming environment. In the next section, a brief explanation will be given on the standard and rotated five-point difference formulas and explicit group relaxation methods for the two dimensional telegraph equations. Domain decomposition techniques and ordering strategies for each discretized method will be discussed in Section III. Section IV describes the parallelization under OpenMP programming environment. The numerical experiments and the results are presented in Section V. Finally, concluding remarks are given in Section VI.

II. The Group Iterative Methods

As described in the previous section, Equation (1) can be approximated by various finite difference formulas. One commonly used formula is the standard five-point difference scheme:

$$\begin{aligned}&(1 + a + 2r^2 + b/2)u_{i,j,m+1} - (r^2/2)u_{i-1,j,m+1} \\ &-(r^2/2)u_{i+1,j,m+1} - (r^2/2)u_{i,j-1,m+1} - (r^2/2)u_{i,j+1,m+1} \\ &= (2 - 2r^2 - b/2)u_{i,j,m} + (r^2/2)u_{i-1,j,m} + (r^2/2)u_{i+1,j,m} \\ &+(r^2/2)u_{i,j-1,m} + (r^2/2)u_{i,j+1,m} + (a-1)u_{i,j,m-1} + \Delta t^2 F_{i,j,m+1/2}\end{aligned} \quad (2)$$

where $r = \Delta t/h, a = \alpha\Delta t, b = \beta^2\Delta t^2$ and $h = \Delta x = \Delta y = 1/n$.

Another formula is the rotated five-point difference scheme:

$$\begin{aligned}&(1 + a + r^2 + b/2)u_{i,j,m+1} - (r^2/4)u_{i-1,j-1,m+1} \\ &-(r^2/4)u_{i+1,j+1,m+1} - (r^2/4)u_{i-1,j+1,m+1} - (r^2/4)u_{i+1,j-1,m+1} \\ &= (2 - r^2 - b/2)u_{i,j,m} + (r^2/4)u_{i-1,j-1,m} + (r^2/4)u_{i+1,j+1,m} \\ &+(r^2/4)u_{i-1,j+1,m} + (r^2/4)u_{i+1,j-1,m} + (a-1)u_{i,j,m-1} + \Delta t^2 F_{i,j,m+1/2}\end{aligned} \quad (3)$$

Equation (3) is obtained by rotating the x-y axis clockwise 45 degrees [3]. Rotated five-point difference scheme can be constructed by dividing the grid points into 2 types of points on the x-y plane of the solution domain. Iterations can be generated involving one type of points only and when

convergence is achieved, the solution at the remaining points will be evaluated directly using equation (2).

The application of Equation (2) and (3) to the grid points in the solution domain at each time level will result in a large sparse linear system of form

$$Au = b \tag{4}$$

where the matrix A and the column vector b are both known, and the column vector u is unknown. We will next describe the construction of the Explicit Group (EG) and Explicit Decoupled Group (EDG) methods.

A. Explicit Group (EG) Method

Applying equation (2) to any group of four points on a solution domain will result in a (4x4) system of equations

$$\begin{pmatrix} k1 & k2 & k3 & k2 \\ k2 & k1 & k2 & k3 \\ k3 & k2 & k1 & k2 \\ k2 & k3 & k2 & k1 \end{pmatrix} \begin{pmatrix} u_{i,j,m+1} \\ u_{i+1,j,m+1} \\ u_{i+1,j+1,m+1} \\ u_{i,j+1,m+1} \end{pmatrix} = \begin{pmatrix} rhs_{i,j} \\ rhs_{i+,j} \\ rhs_{i+1,j+1} \\ rhs_{i,j+1} \end{pmatrix} \tag{5}$$

where

$$k1 = 1 + a + 2r^2 + b/2 \,;\, k2 = -r^2/2 \text{ and } k3 = 0$$

$$rhs_{i,j} = (r^2/2)[u_{i-1,j,m+1} + u_{i,j-1,m+1}]$$
$$+(r^2/2)[u_{i-1,j,m} + u_{i,j-1,m} + u_{i+1,j,m} + u_{i,j+1,m}]$$
$$+(2 - 2r^2 - b/2)u_{i,j,m} + (a-1)u_{i,j,m-1} + \Delta t^2 F_{i,j,m+1/2}$$

$$rhs_{i+1,j+1} = (r^2/2)[u_{i+2,j+1,m+1} + u_{i+1,j+2,m+1}]$$
$$+(r^2/2)[u_{i,j+1,m} + u_{i+1,j,m} + u_{i+2,j+1,m} + u_{i+1,j+2,m}]$$
$$+(2 - 2r^2 - b/2)u_{i+1,j+1,m} + (a-1)u_{i+1,j+1,m-1} + \Delta t^2 F_{i+1,j+1,m+1/2}$$

$$rhs_{i,j+1} = (r^2/2)[u_{i-1,j+1,m+1} + u_{i,j+2,m+1}]$$
$$+(r^2/2)[u_{i-1,j+1,m} + u_{i,j,m} + u_{i+1,j+1,m} + u_{i,j+2,m}]$$
$$+(2 - 2r^2 - b/2)u_{i,j+1,m} + (a-1)u_{i,j+1,m-1} + \Delta t^2 F_{i,j+1,m+1/2}$$

$$rhs_{i+1,j} = (r^2/2)[u_{i+1,j-1,m+1} + u_{i+2,j,m+1}]$$
$$+(r^2/2)[u_{i,j,m} + u_{i+1,j-1,m} + u_{i+2,j,m} + u_{i+1,j+1,m}]$$
$$+(2 - 2r^2 - b/2)u_{i+1,j,m} + (a-1)u_{i+1,j,m-1} + \Delta t^2 F_{i+1,j,m+1/2}$$

The (4x4) system in (4) can be inverted to produce a four-point EG formula:

$$\begin{pmatrix} u_{i,j,m+1} \\ u_{i+1,j,m+1} \\ u_{i+1,j+1,m+1} \\ u_{i,j+1,m+1} \end{pmatrix} = \begin{pmatrix} m1 & m2 & m3 & m2 \\ m2 & m1 & m2 & m3 \\ m3 & m2 & m1 & m2 \\ m2 & m3 & m2 & m1 \end{pmatrix} \begin{pmatrix} rhs_{i,j} \\ rhs_{i+1,j} \\ rhs_{i+1,j+1} \\ rhs_{i,j+1} \end{pmatrix} \tag{6}$$

where

$$m1 = 2(4a^2 + 4ab + 16ar^2 + 8a + 16r^2 + 4 + 8r^2b + 4b + 14r^4 + b^2)$$
$$/(8a^3 + 12a^2b + 48a^2r^2 + 24a^2 + 24ab + 88ar^4 + 96ar^2 + 48ar^2b$$
$$+24a + 8 + 6ab^2 + 12b + 12r^2b^2 + 48r^2 + b^3 + 88r^4 + 44r^4b$$
$$+48r^6 + 48r^2b + 6b^2);$$

$$m2 = 2r^2 / (4a^2 + 4ab + 16ar^2 + 8a + 4b$$
$$+12r^4 + 16r^2 + 8r^2b + 4 + b^2);$$

$$m3 = 4r^4 / (8a^3 + 12a^2b + 48a^2r^2 + 24a^2 + 24ab + 88ar^4$$
$$+96ar^2 + 48ar^2b + 24a + 6ab^2 + 12r^2b^2 + 12b + 8$$
$$+48r^2 + b^3 + 88r^4 + 44r^4b + 48r^6 + 48r^2b + 6b^2).$$

B. Explicit Decoupled Group (EDG) Method

Similarly, applying equation (3) to any group of four points on a solution domain at each time level will result in a (4x4) system

$$\begin{pmatrix} k1 & k2 & k3 & k3 \\ k2 & k1 & k3 & k3 \\ k3 & k3 & k1 & k2 \\ k3 & k3 & k2 & k1 \end{pmatrix} \begin{pmatrix} u_{i,j,m+1} \\ u_{i+1,j+1,m+1} \\ u_{i+1,j,m+1} \\ u_{i,j+1,m+1} \end{pmatrix} = \begin{pmatrix} rhs_{i,j} \\ rhs_{i+1,j+1} \\ rhs_{i+1,j} \\ rhs_{i,j+1} \end{pmatrix} \tag{7}$$

where

$$k1 = 1 + a + r^2 + b/2 \,;\, k2 = -r^2/4 \text{ and } k3 = 0$$

$$rhs_{i,j} = (r^2/4)[u_{i-1,j-1,m+1} + u_{i+1,j-1,m+1} + u_{i-1,j+1,m+1}]$$
$$+(r^2/4)[u_{i-1,j-1,m} + u_{i+1,j-1,m} + u_{i+1,j+1,m} + u_{i-1,j+1,m}]$$
$$+(2 - r^2 - b/2)u_{i,j,m} + (a-1)u_{i,j,m-1} + \Delta t^2 F_{i,j,m+1/2}$$

$$rhs_{i+1,j+1} = (r^2/4)[u_{i+2,j,m+1} + u_{i+2,j+2,m+1} + u_{i,j+2,m+1}]$$
$$+(r^2/4)[u_{i,j,m} + u_{i+2,j,m} + u_{i+2,j+2,m} + u_{i,j+2,m}]$$
$$+(2 - r^2 - b/2)u_{i+1,j+1,m} + (a-1)u_{i+1,j+1,m-1} + \Delta t^2 F_{i+1,j+1,m+1/2}$$

$$rhs_{i+1,j} = (r^2/4)[u_{i,j-1,m+1} + u_{i+2,j-1,m+1} + u_{i+2,j+1,m+1}]$$
$$+(r^2/4)[u_{i,j-1,m} + u_{i+2,j-1,m} + u_{i+2,j+1,m} + u_{i,j+1,m}]$$
$$+(2 - r^2 - b/2)u_{i+1,j,m} + (a-1)u_{i+1,j,m-1} + \Delta t^2 F_{i+1,j,m+1/2}$$

$$rhs_{i,j+1} = (r^2/4)[u_{i-1,j,m+1} + u_{i+1,j+2,m+1} + u_{i-1,j+2,m+1}]$$
$$+(r^2/4)[u_{i-1,j,m} + u_{i+1,j,m} + u_{i+1,j+2,m} + u_{i-1,j+2,m}]$$
$$+(2 - r^2 - b/2)u_{i,j+1,m} + (a-1)u_{i,j+1,m-1} + \Delta t^2 F_{i,j+1,m+1/2}$$

which can be written in decoupled system of (2x2) equations in explicit form

$$\begin{pmatrix} u_{i,j,m+1} \\ u_{i+1,j+1,m+1} \end{pmatrix} = \frac{1}{A} \begin{pmatrix} m1 & m2 \\ m2 & m1 \end{pmatrix} \begin{pmatrix} rhs_{i,j} \\ rhs_{i+1,j+1} \end{pmatrix} \tag{8}$$

and

$$\begin{pmatrix} u_{i+1,j,m+1} \\ u_{i,j+1,m+1} \end{pmatrix} = \frac{1}{A} \begin{pmatrix} m1 & m2 \\ m2 & m1 \end{pmatrix} \begin{pmatrix} rhs_{i+1,j} \\ rhs_{i,j+1} \end{pmatrix} \tag{9}$$

where

$A = 16 + 32a + 32r^2 + 16b + 16a^2 + 32ar^2 + 16ab + 15r^4 + 16r^2b + 4b^2$;
$m1 = 8(2 + 2a + 2r^2 + b); m2 = 4r^2$.

Similar to the rotated five-point formula, the EDG scheme is also constructed by dividing the grid points into 2 types of points on the x-y plane of the solution domain at each time level. It corresponds to generation of iterations on one type of points by using either equation (8) or (9) until a certain convergence criteria are met. After convergence is achieved, the solution at the other remaining half of the points are evaluated directly once using the standard five-point difference formula of equation (2). With this, the execution time of this EDG scheme may be reduced by half of the execution time of the EG scheme. Further details regarding these explicit group iterative methods on elliptic and parabolic equations can be obtained in [1, 3-5].

III. Domain Decomposition Method

Most domain decomposition methods (DDM) are developed for solving elliptic problems [6, 7] and parabolic problems [8, 9]. Domain decomposition methods have been considered as very efficient methods for solving partial differential equations on parallel computers [9]. It can be classified to two classes; overlapping and non-overlapping methods with respect to the decomposition of the domain. The main difference between overlapping and non-overlapping schemes is in the choice of the shared boundaries or areas. In this paper we focus on the use of overlapping domain decomposition method in the spatial domain to solve the two-dimensional hyperbolic problem (1). In order to solve this DDM with overlapping sub-domain, Schwarz alternating procedure (SAP) is used. This Schwarz alternating procedure operates between two overlapping sub-domains; solving the Dirichlet problem on one sub-domain in each iteration by taking the boundary conditions based on the most recent solution obtained from the other sub-domain. The details of this SAP can be obtained in [2].

In order to implement this domain decomposition algorithm, different ordering strategies need to be considered for each finite discretization scheme due to the shared boundaries between sub-domains. In this case, the standard point iterative scheme uses natural ordering strategies, the rotated point iterative scheme uses zebra ordering strategies while the explicit group iterative method uses red black group ordering strategies and explicit decoupled group method uses combination of red black and zebra ordering strategies. We describe the ordering strategies used for the explicit group iterative method and its algorithm. In Fig.(1), we decompose the solution domain into four sub-domains, $\Omega_1, \Omega_2, \Omega_3$ and Ω_4. In order to solve at the points $\mathbf{O}$ in Ω_1, we need the points 1, 2, 3, 4 where the points 1 and 2 are from Ω_2 and Ω_3 respectively, while the points 3 and 4 are from Ω_1. In the case of parallelization, the sub-domains $\Omega_1, \Omega_2, \Omega_3$ and Ω_4 are computed concurrently. There is a possibility that the solutions at the points 1 and 2 are updated on the respective sub-domains when the points $\mathbf{O}$ are computed. This may cause inaccuracy in the numerical results. Thus, we need to organize the ordering strategies to prevent any conflict on the usage of points among sub-domains. With this in mind, a red black group ordering strategy is introduced to this scheme. The black group points are computed concurrently, followed by red group points. The algorithm of this scheme is presented in Table 1.

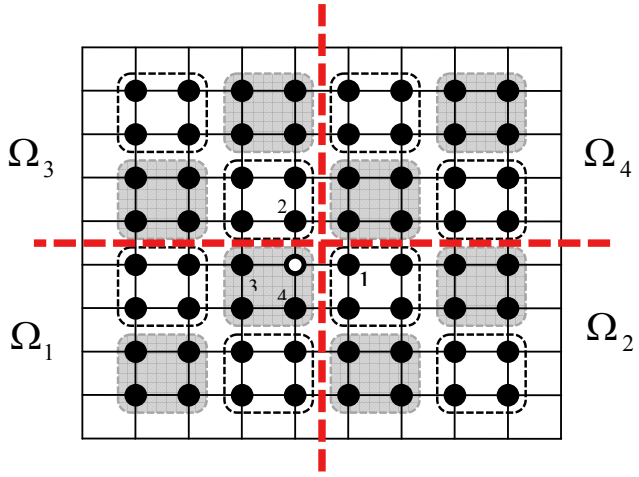

Fig. 1 Explicit group scheme with domain decomposition method

This domain decomposition strategy can also be implemented for the EDG scheme. But the iterations will only involve two out of 4 points in a group using either equation (8) or (9). After the global iteration converges, the solutions at the remaining points are obtained directly using the standard equation (2) before proceeding to the next time level

TABLE I
Algorithm for Explicit Group Scheme Using Red Black Group Ordering Strategy

```
1.  Choose an initial guess u to the solution
2.    For each time step:
3.      Set Boundary Condition
4.      Until convergence, Do (Global):
5.          Identify the subdomain boundaries values
6.          Until Convergence, Do (Local):
7.              For each subdomain:
8.                  Solve at the black group points
9.              EndDo
10.             For each subdomain:
11.                 Solve at the red group points
12.             EndDo
13.             Check the local convergence test
14.         EndDo
15.         Check the global convergence test
16.     EndDo
17. EndDo
```

IV. Parallel Implementation on Multi-core Processors

A multiple-core (multi-core in short) processor is an integrated circuit (IC) which combines two or more independent cores in a single PC. As of now, some PC may have dual-core or quad-core processors. Hence for the purpose of this paper we will focus on quad-core processors. However a multi-core PC will operate with a single core, unless

specially instructed to run on multi-core. To run on multi-core computers, the programs must be modified appropriately. The aims are to enhance the processing performance, to reduce power consumption, and to process multiple tasks simultaneously with better efficiency. In order to take advantage of these multi-core architectures, programs need be tailored to enable it to be executed in parallel mode [10]. Table 2 presented the syntax of implementing the program on multi-core processor. Considering parallelism, it is observed that Steps 7 – 12 in Table 1 is the most expensive part of the algorithm and this part should have the most advantage from the parallelization. For convenience, the solution domain is decomposed into four horizontal strips. Each strip will send to each core or thread for execution which runs concurrently.

TABLE II
SYNTAX FOR IMPLEMENTING THE PROGRAM ON MULTI-CORE PROCESSOR

```
#include <omp.h>
void main()
{
    int num_threads;
    omp_set_num_threads(omp_num_procs());
    #pragma omp parallel for
    {
        Compute the points in each sub-domain
    }
}
```

V. EXPERIMENTAL RESULTS AND DISCUSSIONS

In order to demonstrate the viability of the proposed methods in solving the two dimensional second order hyperbolic equation (1), experiments were carried out on a quad core i7 CPU 2.0 GHz, 4GB of RAM with Window 7 operating system using Microsoft Visual Studio 2010. This experiment is to solve the hyperbolic problem (1) with the exact solution $u(x,y,t) = x^2 + y^2 + t$ and $f(x,y,t) = -2 + x^2 + y^2 + t$ over the region $\Omega = [0,1]\times[0,1]$ and $0 \le t \le 1$ [11]. The boundary and initial conditions are given by

$u(0,y,t) = y^2 + t;\ u(1,y,t) = 1 + y^2 + t;$

$u(x,0,t) = x^2 + t;\ u(x,1,t) = x^2 + 1 + t.$

$u(x,y,0) = x^2 + y^2;\ u_t(x,y,0) = x^2 + y^2 + 1.$

The relaxation factor is set equal to 1.0 (Gauss Seidel relaxation scheme). The local and global epsilons were set equal to $10E-9$ and $10E-10$, respectively. The time step used is $\Delta t = 0.001$ and the total time is $T = 100$. The experiment result in Table 3 shows the comparison execution times obtained for the sequential algorithm (1 thread) and parallel algorithm (4 threads) for the standard point, rotated point, explicit group and explicit decoupled group methods. The speedup is used to measure the performance of the parallel algorithms compared to the corresponding sequential algorithms. The speedup formula used is in the form of

$$\text{Speedup} = \frac{\text{Execution time for a single thread}}{\text{Execution time using 4 threads}};\ S_p = \frac{T_1}{T_2 / 4}$$

It can be observed that the computational results obtained from the EG and EDG methods maintained the same degree of accuracies with the point methods. The EDG method requires the least computing times compared to other methods due to its lower computational complexity. It is observed that the speedups for the rotated point, EG and EDG are not as good as the standard point method due to its ordering strategies. For the EG method for example, the black group points need to finish its computation before the updates on red group points start to be computed. This may incur more overheads and delay the execution timings. The same thing also occurred in the rotated point and the EDG method. However, as shown in Table 3, the execution times of the parallel EDG can be saved up to about 50% compared to the sequential EDG, 41% for the EG method, 52% for the rotated point method and 47% for the standard point method for the mesh size 321.

TABLE III
EXPERIMENT RESULTS

h^{-1}	Non Parallel (1 Thread)			Parallel (4 Threads)			
	Standard Point Iterative Method						
	Iter	Max Error	Elapsed Time	Iter	Max Error	Elapsed Time	Speed-up
81	2	6.978E-5	77.835	2	6.978E-5	37.307	8.345
161	2	6.978E-5	396.618	2	6.978E-5	194.343	8.163
201	2	6.978E-5	731.795	2	6.978E-5	362.542	8.074
321	2	6.940E-5	2554.604	2	6.942E-5	1351.804	7.565
h^{-1}	Rotated Point Iterative Method						
	Iter	Max Error	Elapsed Time	Iter	Max Error	Elapsed Time	Speed-up
81	2	6.981E-5	25.573	2	6.981E-5	22.947	4.458
161	2	6.981E-5	170.089	2	6.981E-5	117.238	5.803
201	2	6.981E-5	280.781	2	6.981E-5	196.040	5.729
321	2	6.981E-5	1862.908	2	6.981E-5	889.472	8.378
h^{-1}	Explicit Group Iterative Method						
	Iter	Max Error	Elapsed Time	Iter	Max Error	Elapsed Time	Speed-up
81	2	6.978E-5	30.301	2	6.978E-5	27.071	4.477
161	2	6.980E-5	192.598	2	6.980E-5	141.821	5.432
201	2	6.980E-5	366.113	2	6.980E-5	268.105	5.462
321	2	6.980E-5	1698.085	2	6.980E-5	1008.308	6.736
h^{-1}	Explicit Decoupled Group Iterative Method						
	Iter	Max Error	Elapsed Time	Iter	Max Error	Elapsed Time	Speed-up
81	2	6.981E-5	24.358	2	6.981E-5	22.261	4.377
161	2	6.981E-5	162.271	2	6.981E-5	111.641	5.814
201	2	6.981E-5	273.845	2	6.981E-5	190.635	5.745
321	2	6.981E-5	1559.025	2	6.981E-5	790.265	7.891

VI. CONCLUSION

In this paper, we have presented the utilization of domain decomposition techniques on some newly developed finite difference schemes in solving two dimensional telegraph equations. The finite difference schemes used in this paper were the explicit group (EG) and explicit decoupled group (EDG) methods. We have described the development of these parallel group iterative schemes under OpenMP programming environment. For comparison purposes, we also include the results of the pointwise schemes; the traditional standard and the rotated point methods. The experimental results show that the parallel algorithms managed to successfully save up to 50% of the computational costs compared to their sequential algorithms. Research on how these algorithms need to be re-constructed to be implemented on parallel technology using

graphics processing unit (GPU) is still under investigation and will be reported soon.

ACKNOWLEDGMENT

Financial support provided by Grant #203/PMATHS/6711188 and the School of Maths KPI Grant #1002/PMATHS/ARSP13000 for the completion of this article are gratefully acknowledged.

REFERENCES

[1] Kew, L.M. and Ali, N.H.M. "Explicit Group Iterative Methods for the Solution of Telegraph Equations", Lecture Notes In Engineering and Computer Science, World Congress On Engineering 2010, The 2010 International Conference of Applied and Engineering Mathematics, 30th June – 2nd July 2010, Imperial College, London, 2010, pp.1770- 1775.
[2] Saad, Y. "Iterative Methods for Sparse Linear Systems". 2nd Edition. pp. 382-421.
[3] Abdullah, A.R. "The Four Point Explicit Decoupled Group EDG Method: A Fast Poisson Solver". International Journal of Computer Mathematics, 38, 1991, pp.61-70.
[4] Yousif, W.S. and Evans, D.J. "Explicit De-Coupled Group Iterative Methods and Their Parallel Implementations". Parallel Algorithms and Applications, 7, 1995, pp.53-71.
[5] Evans, D.J. "Group Explicit Methods for the Numerical Solution of Partial Differential Equations". Loughborough University of Technology, UK, Gordon and Breach Science Publisher, The Netherlands, 1997, pp.106-162.
[6] Dryja, M. and Widlund, O.B. "Some Domain Decomposition Algorithms for Elliptic Problems", in: L. Hayes, D. Kincaid (Eds.), Iterative Methods for Large Linear Systems, Academic Press, San Diego, CA. 1989.
[7] Cai, X.-C. and Widlund, O.B. "Domain Decomposition Algorithms for Indefinite Elliptic Problems", SIAM J. Sci. Statist. Comput., 13, 1992, pp. 243–258.
[8] Dawson, C.N., Du, Q., and Dupont, T.F. "A Finite Difference Domain Decomposition Algorithm for Numerical Solution of the Heat Equation". Mathematics of Computation, 57(195), 1991, pp. 63-71.
[9] Jun, Y. and Mai, T.Z. (2006). "IPIC Domain Decomposition Algorithm for Parabolic Problems". Applied Mathematics and Computation, 177, 2006, pp. 352-364.
[10] Ali, A. "Introduction to Hybrid MPI/OpenMP Programming". Presented at ICTP Advanced School in High Performance and Grid Computing, 11-22 April 2011, Abdus Salam ICTP – Trieste.
[11] Dehghan, M. and Shokri, A., "A Meshless Method for Numerical Solution of a Linear Hyperbolic Equation with Variable Coefficients in Two Space Dimensions." Numerical Methods for Partial Differential Equations, 2008, pp.494-506.