Jurnal Teknologi

Full Paper

PARALLEL COMPUTING OF NUMERICAL SCHEMES AND BIG DATA ANALYTIC FOR SOLVING REAL LIFE APPLICATIONS

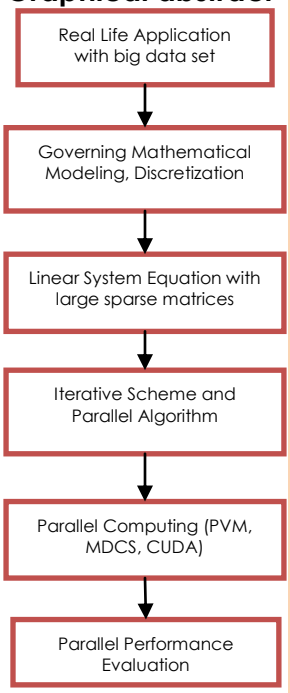
Norma Alias[a*], Nadia Nofri Yeni Suhari[a], Hafizah Farhah Saipan Saipol[a], Abdullah Aysh Dahawi[a], Masyitah Mohd Saidi[a], Hazidatul Akma Hamlan[a], Che Rahim Che Teh[b]

[a]CSNano, Ibnu Sina Institute for Scientific and Industrial Research, Universiti Teknologi Malaysia, 81310 UTM Johor Bahru, Johor, Malaysia
[b]Department of Mathematical Science, Faculty of Sciences, Universiti Teknologi Malaysia, 81310 UTM Johor Bahru, Johor, Malaysia

Article history
Received
13 December 2015
Received in revised form
20 March 2016
Accepted
28 February 2016

*Corresponding author
norma@ibnusina.utm.my

Graphical abstract



Abstract

This paper proposed the several real life applications for big data analytic using parallel computing software. Some parallel computing software under consideration are Parallel Virtual Machine, MATLAB Distributed Computing Server and Compute Unified Device Architecture to simulate the big data problems. The parallel computing is able to overcome the poor performance at the runtime, speedup and efficiency of programming in sequential computing. The mathematical models for the big data analytic are based on partial differential equations and obtained the large sparse matrices from discretization and development of the linear equation system. Iterative numerical schemes are used to solve the problems. Thus, the process of computational problems are summarized in parallel algorithm. Therefore, the parallel algorithm development is based on domain decomposition of problems and the architecture of difference parallel computing software. The parallel performance evaluations for distributed and shared memory architecture are investigated in terms of speedup, efficiency, effectiveness and temporal performance.

Keywords: Parallel Computing, Big Data, Parallel Algorithm, Domain Decomposition, PVM

Abstrak

Kajian ini mencadangkan beberapa aplikasi kehidupan sebenar untuk analisis data yang besar menggunakan perisian pengkomputeran selari. Beberapa perisian pengkomputeran selari yang dipertimbangan adalah Mesin Selari Maya, Perkomputeran Pengedaran Server MATLAB dan Pembinaan Pengiraan Peranti Unified untuk mensimulasikan masalah data yang besar. Pengkomputeran selari mampu mengatasi kelemahan prestasi pengkomputeran berjujukan dalam catatan masa, kecepatan dan kecekapan pengaturcaraan. Model matematik digunakan untuk menganalisis data yang besar berdasarkan kepada pendiskretan persamaan pembezaan separa dan matriks bersaiz besar diperolehi daripada pembangunan sistem persamaan linear. Skim berangka lelaran digunakan untuk menyelesaikan masalah-masalah tersebut. Kemudian, proses masalah pengiraan diringkaskan dalam algoritma selari. Oleh itu, pembangunan algoritma selari adalah berdasarkan masalah domain penguraian dan perbezaan seni bina perisian pengkomputeran selari. Penilaian prestasi perkomputeran selari untuk seni bina pengedaran memori dan perkongsian memori disiasat dari segi kecepatan, kecekapan, keberkesanan dan prestasi sementara.

Kata Kunci: Perkomputeran selari, Data besar, Algoritma selari, Domain penguraian, PVM

1.0 INTRODUCTION

The parallel computing is a computational type involved the simultaneous programming running programs and used multiple computer resource to investigate a big data analytic problem [1]. The power of computer depend on cost, performance, capability, and tackle intractable problems[2]. Thus, the parallel computing system is needed to support active simulation with a high-speed solution. It is done by transferring the sequential algorithm into the parallel algorithm in solving the big data systems. Parallel algorithm visualizes the mathematical model in real size problems with accurate prediction. Then, the computation of large sparse data size is supported by parallel computing systems with the number of processors for single CPU and huge size of memory[3].

The main memory for parallel computing either distributed memory or shared memory. Distributed memory consists of several numbers of processors for computing the sub-domain of a full grid solution and developed by connecting between processors to a main memory [4]. The main memory will accomplish instruction and data storage. Each processor runs their own operating system using local memory and linked with each other via a communication network [5]. The message passing functioned to distribute the data [2]. However, shared memory, shared the data between all processing elements in a single address space[6]. Shared memory is a proficient means of passing data between programs where the multiple programs are concurrently accessed to avoid overlapping data.

The organization of this paper started with introduction of parallel computing followed by the applications with big data using the parallel computing. Next, the explanation of parallel computing software and its architecture that are used in this paper which are Parallel Virtual Machine (PVM), MATLAB Distributed Computing Server (MDCS) and Compute Unified Device Architecture (CUDA), GPGPU. The explanation continued by the methodology to compute parallel computing and their algorithm based on the software architecture. The results of experiments for big data analytic problems are shown in next section in terms of numerical analysis and parallel performances. The last section is about conclusion of this paper.

2.0 BIG DATA ANALYTIC IN REAL LIFE APPLICATIONS

A big data analytic involve the process of predicting big data to make better prediction and visualization of the real life application. The real-life application is a complex system, involves big data problems and huge simulation. The function of mathematical modeling is to explain and recognize the system easily, as well as finding the optimum solutions to predict the pattern of the real-life application. Thus, the parallel computing is the best way to solve the mathematical model by numerical schemes with big data analytic and large sparse matrices. Several real-life applications in our research abilities use parallel computing to solve the mathematical modeling numerically with large sparse matrices and large data set.

2.1 Food Drying Process

Food drying is a process of removing moisture and water vapor from drying material at a certain temperature level into surrounding space[7].A chronology of food drying, start from sun drying, microwaves, cabinet trays, spray drier, freeze drier and the new technology of control pressure drop (DIC)[8, 9]. The drying process is one of the industrial preservation techniques that condensed the water content of food and reduces biochemical reactions of the degradation[10, 11]. The large sparse linear system matrices of the mathematical model of drying process obtain the effect of drying technique by improving the nutrient quality, cost effective and time consuming of the process.

2.2 Molding and Melting of Composite Material

Since the recycling of composites on a large scale is an unsolved problem, the treatment and simulations of thermosetting nanocomposite materials from electronic waste (e-waste) processes are important for a variety of e-waste composite recovery process. During the molding process, the numerical simulation for a complex geometry with large sparse matrices in automotive part is used to predict the temperature distribution and the curing behavior of the composite [12].

2.3 Ice Melting and Ocean Movement

Ice melting and ocean movement models have been developed to understand the response of them in affecting the Earth's climate change. In Antarctica, there has a huge ice sheet that covers the land mass of the area. A particular area in Antarctica is behaving differently for ice sheet, and it is complex. The ice covered the Antarctica continent and ocean movement have a main part of the global climate system of the Earth [13]. A theory of large-scale ocean circulation is applicable for ocean movement phenomena. The ocean movement models are involved the conservation of mass, momentum, salt, and thermodynamic energy and the state for seawater [14]. The melting of the Antarctica ice sheets and the ocean movement will lead to the increasing of the sea level, which is one of the effects of climate change.

3.0 PARALLEL COMPUTING SOFTWARE AND ITS ARCHITECTURE

The single computer may be impossibly practical for solving the big data and large sparse matrix of the model on account of poor performance in computational time, speedup and efficiency of programming. Thus, parallelization is desired to overcome those problems. Some parallel computing software are used to develop the parallel programming for achieving higher parallel performance evaluations (PPE). This paper proposed three difference parallel computing software for solving the grand challenge applications in section 2 and the analysis are based on their architectures, algorithms and performances.

3.1 Parallel Virtual Machine (PVM)

The heterogeneous computer systems of Parallel Virtual Machine (PVM) software are the unified structure of parallel programs with the straightforward manner using existing hardware [15]. The network of PVM architecture perceptibly handles the message routing, transferring data and task scheduling. A library of standard routines allow the initiation and termination of tasks across the network, global communication and barrier synchronization between tasks. The latest version of PVM in this paper is PVM3.4.6 embedded with Linux Fedora Core 21 operating system. The architecture overview of PVM in Figure 1:

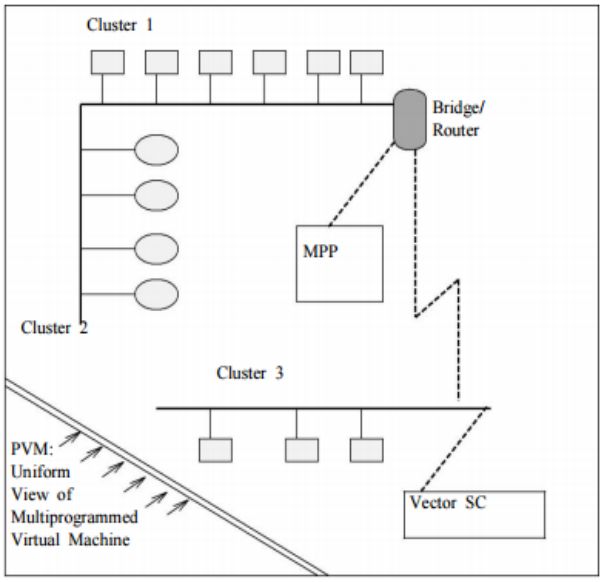


Figure 1 PVM architectural overview[15] where MPP is massively parallel processing and SC is system computing

3.2 MATLAB Distributed Computing Server (MDCS)

The parallel strategy in this paper using MATLAB Distributed Computing Server (MDCS) version R2011a on a cluster of workstation system. The MDCS consists of a heterogeneous computer cluster system contains eight workers with Intel®CoreTM 2 Duo processor architecture under Red Hat Linux 2.6 operation. The distributed memory architectures of MDCS are shown in Figure 2. Previous researchers have implemented the parallel algorithms using distributed memory architecture. Many researchers [4, 16, 17]used various techniques such as a domain decomposition method to solve the large sparse linear systems of multidimensional PDE.

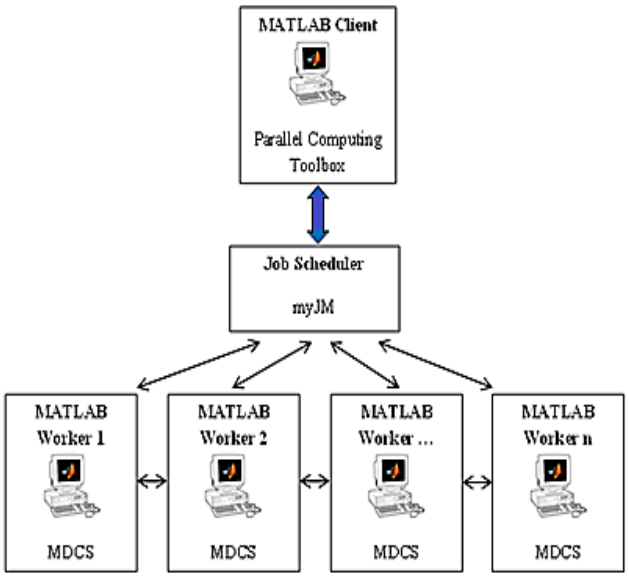


Figure 2 Parallel distributed architecture for MDCS

3.3 Compute Unified Device Architecture (CUDA), GPGPU

The latest parallel software in this paper is emphasized on Compute Unified Device Architecture (CUDA) as the parallel programming software in Windows 7 operations. A multiprocessor system or heterogeneous programming of CUDA software is embedded with standard C language to support the general-purpose computing, on graphics processing unit (GPGPU) and straightforward Application Programming Interface (APIs) for managing the device and memory. Since this software is a new technology, some researchers [18, 19] have implemented the parallel algorithms using shared memory architecture. The terminologies in CUDA programming are hosted and device. The data parallelism will be implemented on devices. Both of the host and the device have their own memory separated by PCI-Express bus, and it is called as CPU memory and GPU memory.

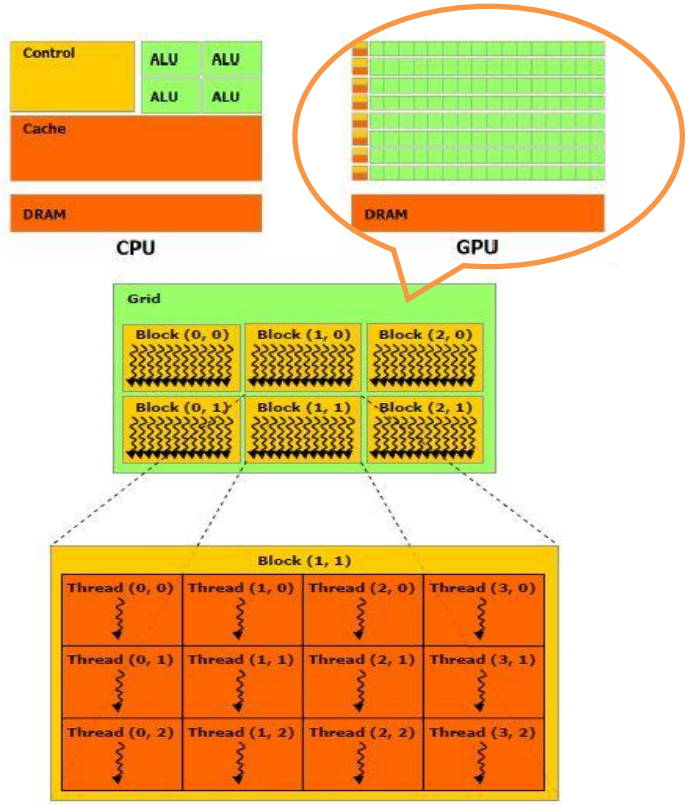


Figure 3 The Operate of GPU in conjunction with CPU. Reprinted from (NVIDIA, 2012).

4.0 METHODOLOGY AND PARALLEL ALGORITHMS

The numerical schemes are suitable for parallel computing in solving the grand challenge applications involving a big data problem. The governing equation of mathematical modeling for real-life application is based on partial differential equations (PDE) with difference equation approaches. In this process, the mathematical models are discretized using finite difference method (FDM) to generate a fine granularity of mesh structure. The discretization of the difference equations obtain the large system of linear equations in matrix form. Our intention on solving large sparse systems of equation is by achieving an effective convergence to the analytical solution of experiment results. The process achieves by implementing some iterative numerical schemes. The basic iterative numerical schemes like Jacobi (JB), Gauss Seidel (RBGS) and Successive Over Relaxation(SOR) schemes usually solve the problem in sequential[20]. However, the parallelization of these schemes by dividing the size of matrices depends on the tasks of processors. In addition, the advanced iterative numerical schemes such as Red-Black Gauss Seidel (RBGS), Alternating Group Explicit (AGE) and Iterative Alternating Decomposition Explicit (IADE) normally divide the task into parallel. The processes of computational are summarized in parallel algorithm. In short, the parallel algorithm for the iterative numerical scheme are based on domain decomposition[21].The domain decomposition technique is approaching due to the computational complexity and huge data. The data distribution and communication issues are the main part of domain decomposition. There are four steps to develop the parallel algorithm based on domain decomposition [22] in Figure 4:

a) Partitioning - Computation performing and data decompose into small tasks.
b) Communication – Coordination of task execution uses the applicable communication structures and algorithms.
c) Agglomeration - The evaluation of task and communication structures in the first two stages with respect to performance requirements and implementation costs.
d) Mapping - Each task is allocated to processor to satisfy the competing goals of maximizing processor utilization, minimizing communication costs and determining load-balancing algorithm.

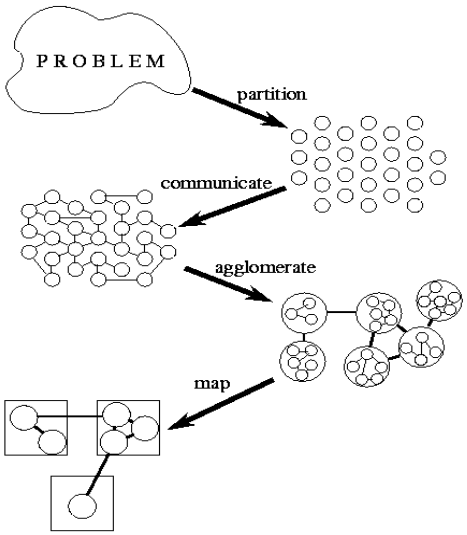


Figure 4 Domain Decomposition Design

4.1 Parallel Virtual Machine (PVM)

The summary of the parallel algorithm for PVMis represented in Figure 5. The connection between master and slaves using Local Area Network (LAN) through Ethernet. Master sends the input and process to all slaves. The computational task will run by the slave until the local stopping criteria, which declare by a master, is fulfilled by each slave. If the global stopping criterion that declares in the master already satisfied, the computation task will be stopped. The formula of local max error for all computation is:

$$|u^{k+1} - u^k| < \varepsilon$$

where the absolute of difference between new and old computation is less than epsilon.

Results obtained by the slave will be sent to the master. Master will process and store the results. The PVM
programming needs to pack and unpack data when transfer and receive the data.

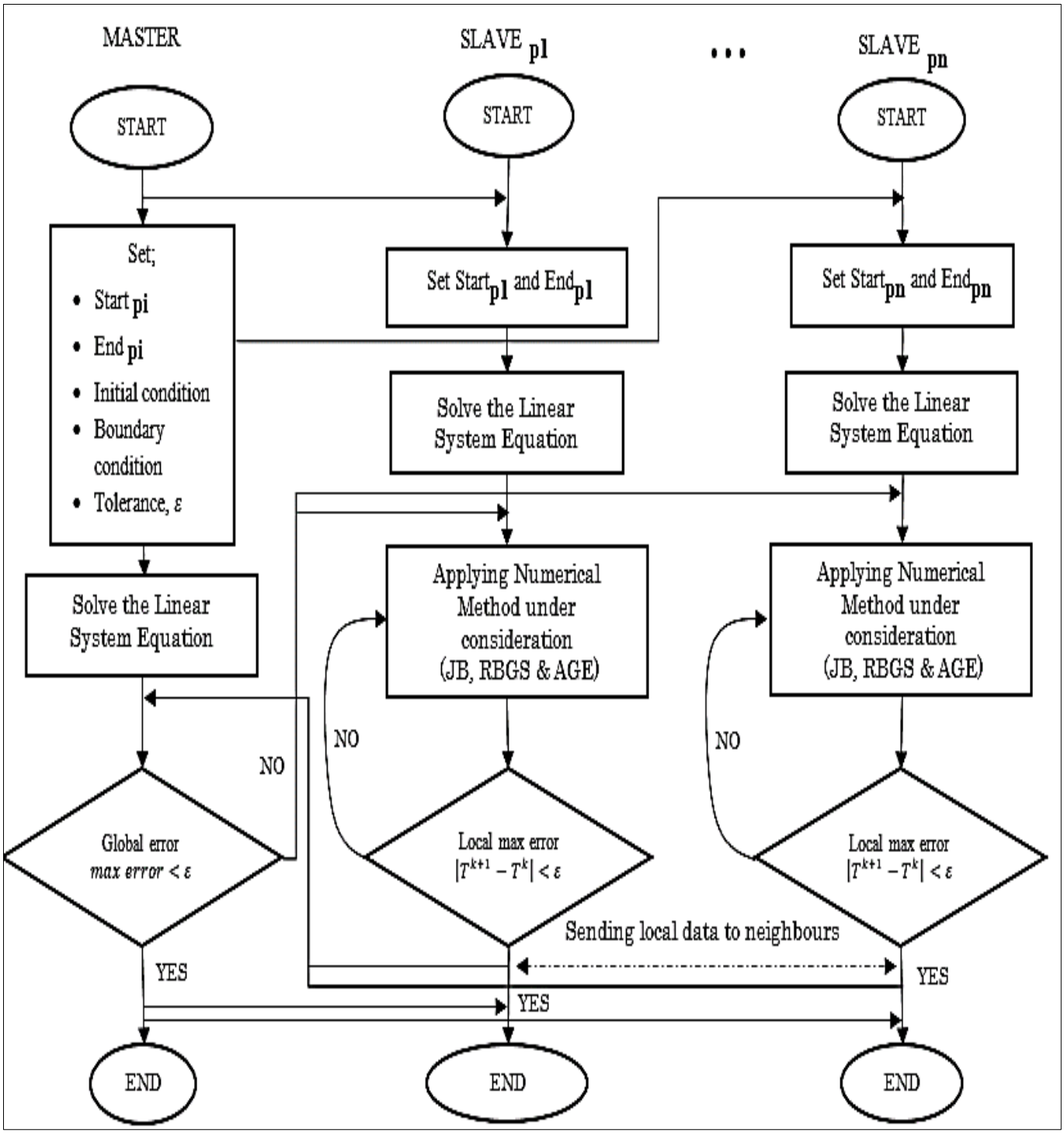


Figure 5 PVM Parallel Algorithm

4.2 Matlab Distributed Computing Server

The message passing activity among the workers is presented in Figure 6. MATLAB clients and workers connected with LAN through Ethernet. The server sends the required workload data to all clients. The computational tasks are assigned to specific clients who are closer to each other.

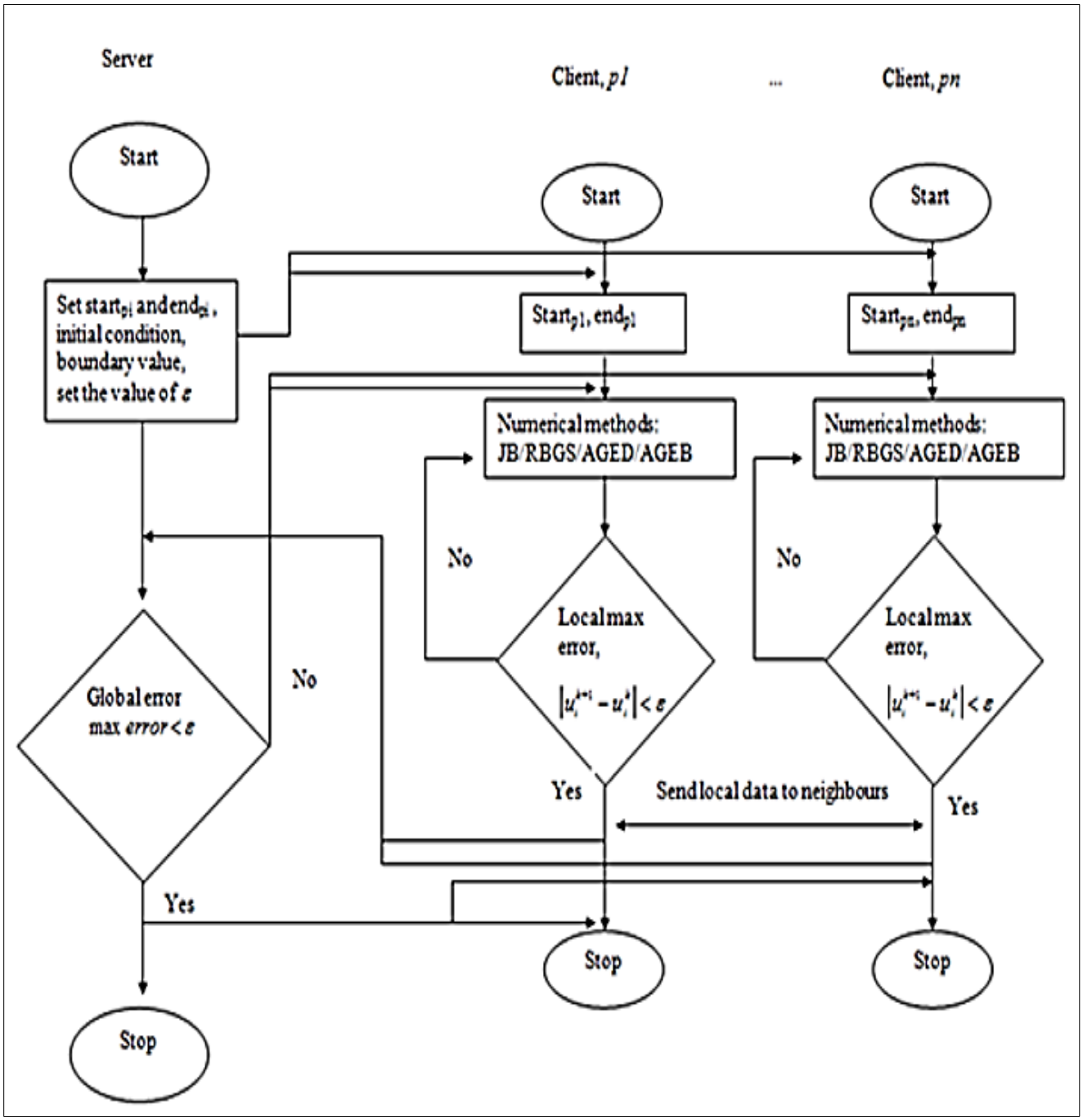


Figure 6 MDCS Parallel Algorithm

4.3 CUDA, GPGPU

Figure7 represents some steps of parallel algorithm obtain in CUDA. Host (CPU) will send the data to the GPU device after allocating thedata migrates in memory device. The data transferred from host to device by launching the kernel function involved the number of blocks and threads. In the device, the computation process remains until reach the given conditions. The distribution of the data for the computation depends on the number of block and threads. The computation results will send back from the device to the host.

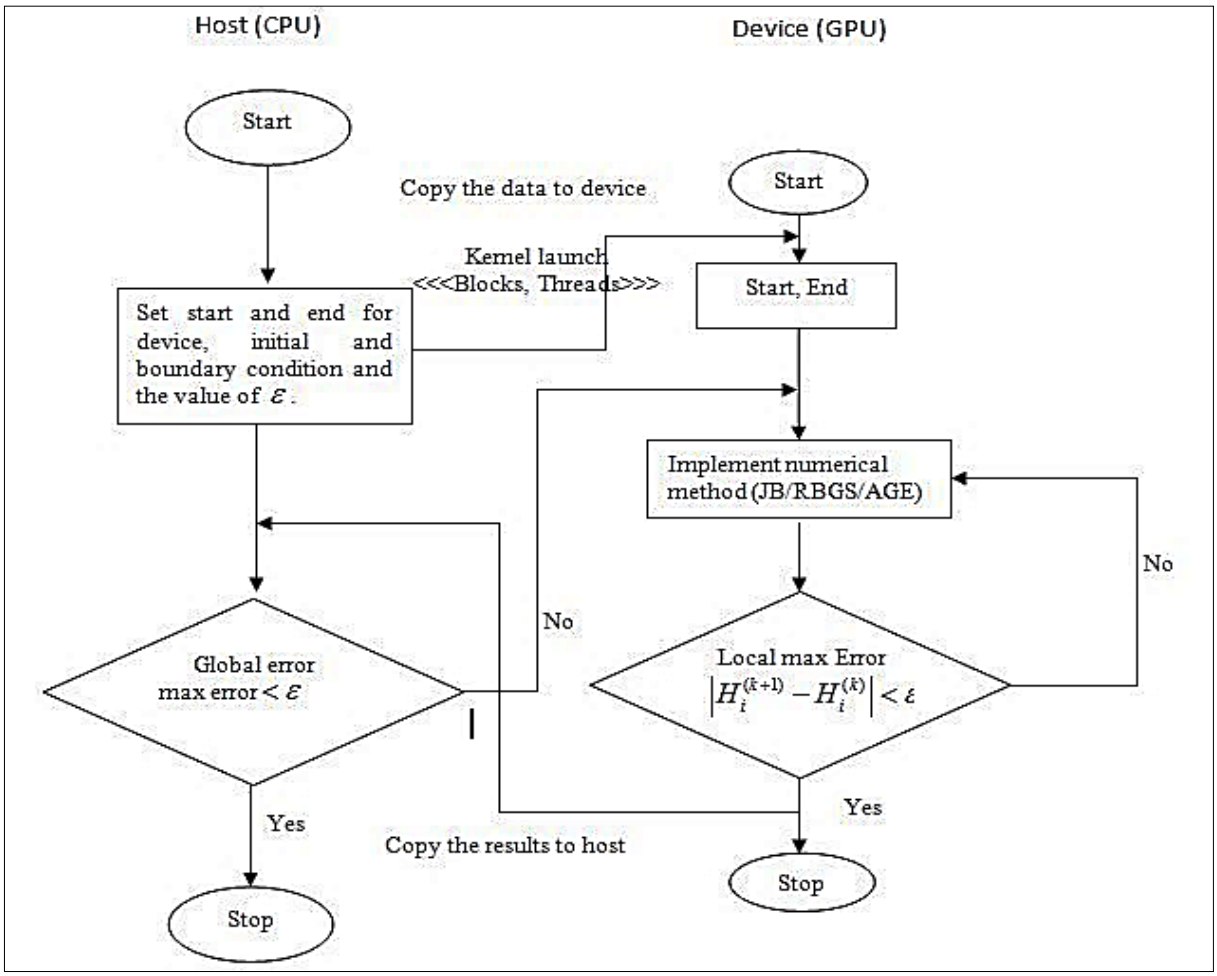


Figure 7 Parallel Algorithm for CUDA

5.0 NUMERICAL ANALYSIS

Numerical analysis is essential for validation and verification of the numerical method's algorithm. The analysis results are presented according to different applications that have been discussed in Section 2. The results are measured based on execution time, number of iterations, maximum error, and root mean square error (RMSE).

5.1 Molding and Melting of Composite Materials

Mathematical model representing the application for molding and melting of composite materials is employed from [23]. In visualizing the temperature profile of the composite materials during its molding and melting processes, the average weight parameter of FDM has been applied for discretization purposed. The linear system equation from the discretization is then solved using two iterative numerical schemes known as JB and RBGS schemes.

Table 1 Numerical Analysis of Melting and Molding Composite Materials

NUMERICAL ANALYSIS $tol = 1.0e^{-7}$		**Schemes**			
		JB SCHEME		**RBGS SCHEME**	
		Melt	Mold	Melt	Mold
Exec. time		200.79	201.65	153.93	154.95
Iteration no.		99	99	78	78
Comp	+/-	$10m$	$10m$	$10m$	$10m$
Comp	×/÷	$15m$	$18m$	$18m$	$15m$
RMSE		1.4366^{-3}	2.1257^{-6}	5.85^{-4}	2.9026^{-7}
Max. error		0.04019	0.00150	0.02432	0.0006

Table 1 above summarize the numerical analysis results for melting and molding of composite materials. Based on the Table 1, it assume that RBGS scheme has resulting the best output compared with JB schemes since the iteration number of RBGS is less than JB scheme for both melting and molding processes. The execution time for solving the problem using RBGS is shorter than JB scheme. The accuracy of RBGS is more accurate since its RMSE is lower than JB scheme. Figure 8 shows the visualization of melting and molding process for composite materials with different numerical schemes.

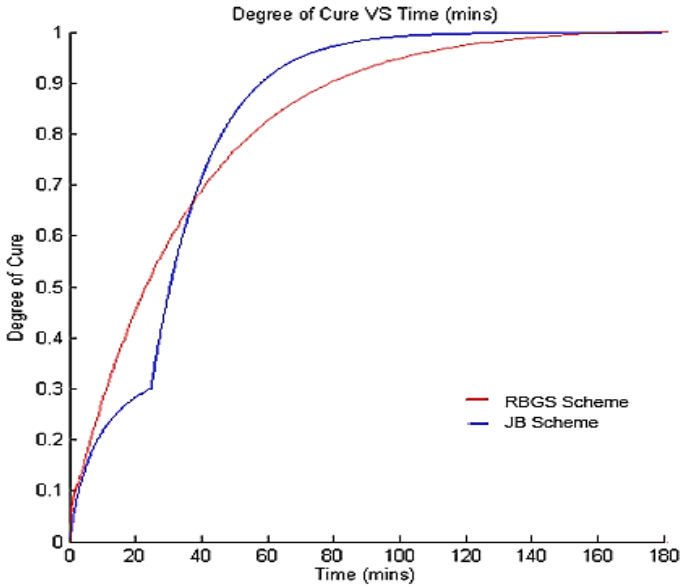


Figure 8 Visualization of melting and molding process for composite materials with different numerical scheme.

5.2 Food Drying Process

The mathematical model of heat and mass equation for drying process is solved using some iterative schemes such as JB, RBGS, AGE Douglas (AGED), and AGE Brian (AGEB). Table 2 shows the numerical analysis for every iterative numerical scheme. From the result, it is shown that AGEB is a superior method compared to AGED, RBGS, and JB scheme. The computation using AGEB involves faster time and high accuracy with the smallest value of maximum error.

Table 1 Numerical analysis of mass and heat equation

Equation	Schemes	Exec. Time (s)	No. of Iter,	Maximum Error	RMSE
Mass	**AGEB**	486.27	9	3.42229e-006	5.42294e-009
	AGED	653.18	14	3.64859e-006	9.76726e-009
	RBGS	3784.72	104	9.99879e-006	6.08226e-008
	JB	7452.19	207	9.99879e-006	4.39527e-008
Heat	**AGEB**	577.49	10	3.06284e-006	3.88697e-009
	AGED	585.62	12	5.00873e-006	1.22061e-008
	RBGS	5425.38	122	9.78178e-006	6.19446e-008
	JB	8655.12	243	9.78178e-006	4.47521e-008

5.3 Ice Melting and Ocean Movement

The mathematical modeling considered for ice melting is the ice thickness equation of 2-D ice flow thermodynamics coupled model. This equation is calculated to see the changes of the Antarctic ice thickness after several years. Then, the primitive equations are representing the mathematical modeling of ocean movement and also solved by using explicit method. The primitive equations are consisted of temperature equation and salinity equations. The numerical analysis for sequential in CUDA software are shown in Table 3.

Table 3: Numerical analysis for ice melting and ocean movement.

Equations	Time Execution	Maximum Error	RMSE
Ice Thickness	152.13	1.77e-4	1.360e-2
Temperature	145.57	8.37e-5	6.943e-5
Salinity	142.39	4.34e-5	2.1223-5

6.0 PARALLEL PERFORMANCE EVALUATION

A parallel performance evaluation (PPE) of PVM, MDCS and CUDA for solving PDE with FDM discretization method are analyzed based on the numerical results for different applications. The computational and simulation of numerical results are based on the parallel algorithms for each software. The indicators for PPE are investigating the time execution (*T)*, speedup, efficiency and temporal performances. Table 4 show the description of the PPE.

Table 4 Parallel Performance Evaluation Description

PPE	Description	Formula
Speedup	The differences of performance between sequential and parallel algorithms	$S_p = {T_1}/{T_p}$
Efficiency	The measurement of processor utilization.	$E_p = {S_p}/{p}$
Effectiveness	Parallel computing will become more effective when F_p value is high	$F_p = {S_p}/{pT_p}$
Temporal Performance	The maximum value of the temporal performance shows the best performance of parallel algorithms	$L_p = {1}/{T_p}$

6.1 Molding and Melting of Composite Materials

The PPE of distributed memory architecture of PVM software in this paper is shown in the Table 5. The computation involves one master and seven slaves for 10000 sizes of matrices of RBGS scheme.From the graph in Figure 9 shows that the speedup of PVM is approach to the optimum speedup. Besides, the efficiency of PVM are applicable to the optimum efficiency. The higher of effectiveness and maximum of temporal performances proof that the parallel computing of PVM able to solve the big data analytic problems.

Table 5 Parallel Performances Evaluation of PVM

No of Processor	S_p	E_p	F_p	L_p
1	1	1	0.0065	0.0065
2	1.9731	0.9865	0.0126	0.0127
3	2.5054	0.8351	0.0135	0.0162
4	3.4401	0.8600	0.0191	0.0222
5	4.3218	0.8644	0.0241	0.0279
6	4.6728	0.7788	0.0235	0.0302
7	6.3842	0.9120	0.0376	0.0412
8	6.4202	0.8025	0.0333	0.0414

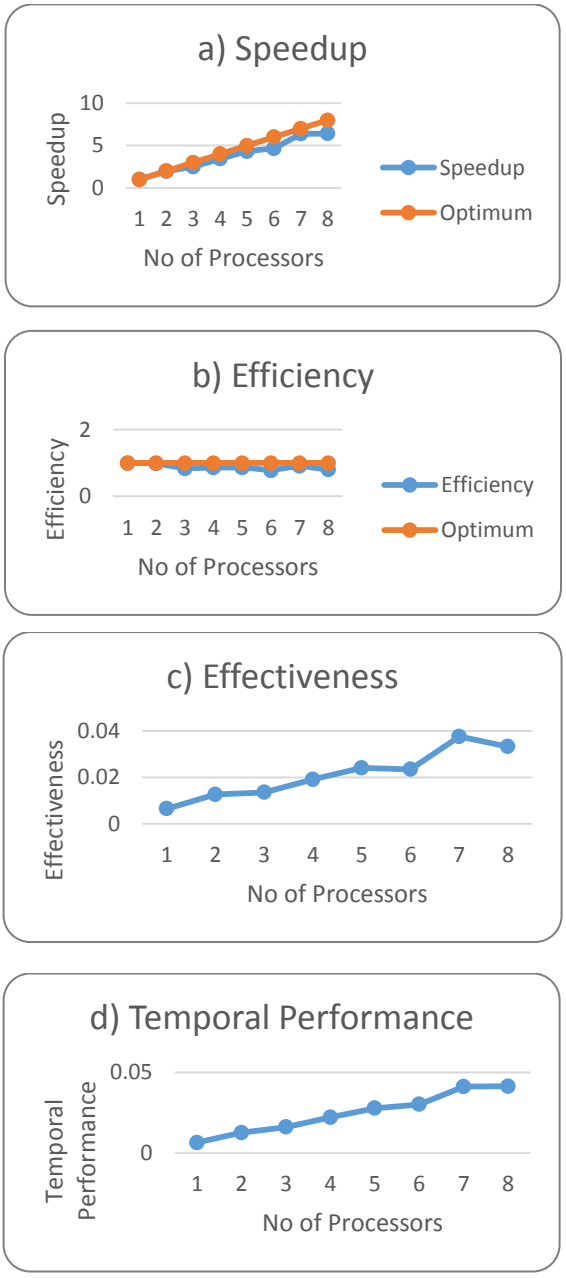


Figure 9 Parallel Performances Evaluation of PVM: a) Speedup, b) Efficiency, c) Effectiveness, d) Temporal Performance

6.2 Food Drying Process

The PPE of MDCS computing software for distributed memory architecture in this paper is shown in the Figure 10. The computation involves one server and seven clients for 900000 size of matrices for mass equations. Figure 10 shows that the AGEB are the superior scheme since the speedup are approach to optimum speedup rather than the other schemes. For efficiency, the AGEB are applicable to the optimum efficiency compared to the other schemes. The higher of effectiveness and temporal performances proof that the AGEB are the best schemes compared to AGE, RBGS and JB.

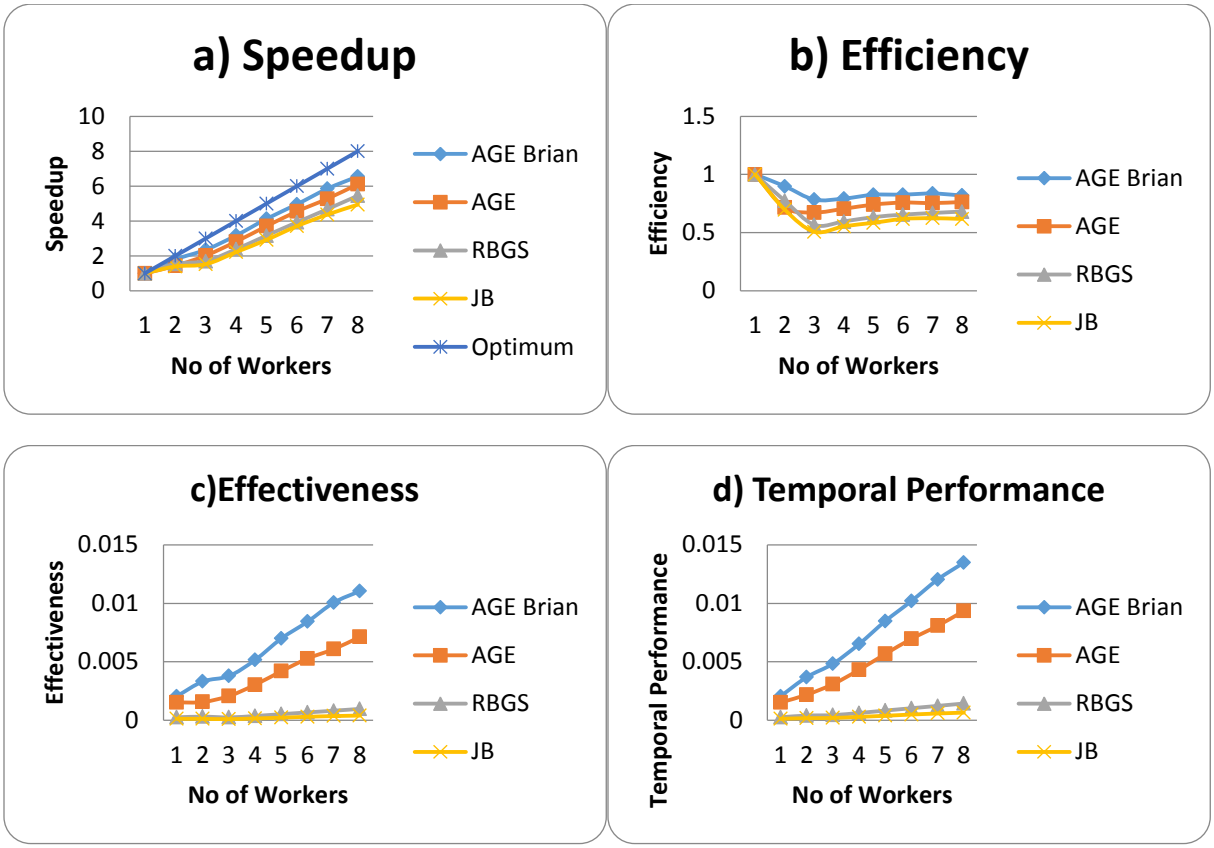


Figure 10 Parallel Performances Evaluation of MDCS: a) Speedup, b) Efficiency, c) Effectiveness, d) Temporal Performance

6.2 Ice Melting and Ocean Movement

The PPE of CUDA programming is different with PPE of PVM and MDCS. The parallelization of CUDA programming on shared memory architectures are based on the number of threads per block in GPU platform. Thus, the PPE are stated in Table 6. From the table, the speedup increases linearly effected by increasing in the number of threads. The graph in Figure 11 shows the optimum speedup by 64 threads. The greater number of threads per block, the execution times are increasing due to the overhead of the kernel launched.

Table 6 Parallel Performances Evaluation of CUDA

No. Threads Per Block	S_p	E_p	F_p	L_p
1	76.597	76.597	1.000	0.013
4	110.727	27.682	0.250	0.009
8	137.086	17.136	0.125	0.007
16	156.929	9.808	0.063	0.006
32	159.453	4.983	0.031	0.006
64	216.015	3.375	0.016	0.005
128	183.613	1.434	0.008	0.005
256	142.495	0.557	0.004	0.007
512	109.394	0.214	0.002	0.009

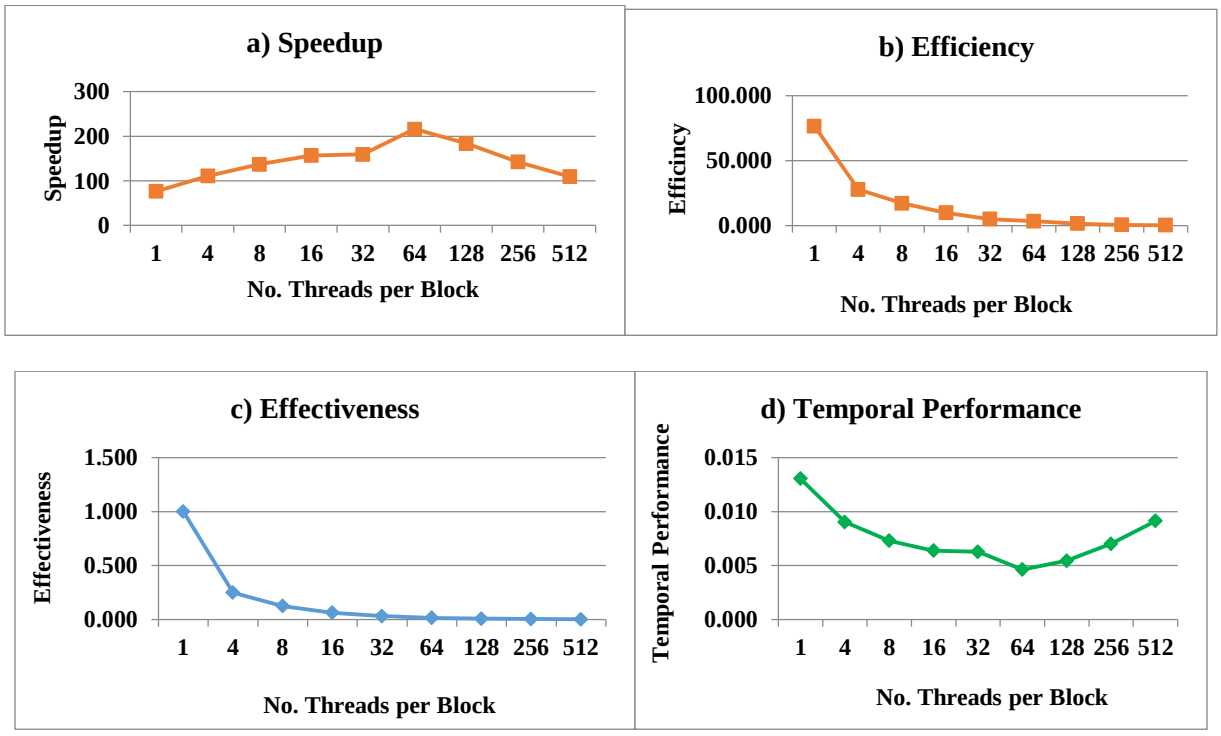


Figure11 Parallel Performances Evaluation of CUDA: a) Speedup, b) Efficiency, c) Effectiveness, d) Temporal Performance

As the number of threads per block is increasing, the effectiveness in CUDA is decreasing as shown in Figure 11. This is due to the asynchronous CPU-GPU memory transfers that deal with a bandwidth problem and the communication on the GPU and the threads is not balance in order to minimum the time. CUDA is more focusing on the reducing the computational runtime. The temporal performance in CUDA that indicate the superior number of threads in this problem is 64.

This paper presents the several grand challenge applications in a big data analytics, which are food drying process, moulding and melting of composite materials and ice melting and ocean movement applications. This paper proposed PVM, MDCS and CUDA to solve the big data analytics in real-life application. The summary of three parallel computing software are indicated in Table 7.

7.0 CONCLUSION

Table7 Summary of PVM, MDCS and CUDA Descriptions

Software Name	Parallel Virtual Machine (PVM)	MATLAB Distributed Computing Server (MDCS)	Compute Unified Device Architecture (CUDA)
Programming System	Heterogeneous	Heterogeneous	Heterogeneous
Memory	Distributed Memory	Distributed Memory	Shared Memory
Architecture	Single Instruction Multiple Data (SIMD)	Single Instruction Multiple Data (SIMD)	Single Instruction Multiple Thread (SIMT)
Operating System	Fedora Core 21 Linux	Red Hat 26 Linux	Windows 7
Terminology	Master and Slaves	Server and Clients	Host and Device

The numerical schemes for big data analyticsare generated from mathematical modeling based on partial differential equation. The numerical schemes used for solving mathematical model begin with the discretization of modeling, linear system development in large sparse matrices and iterative numerical schemes. The computation process is solved in parallel algorithm with domain decomposition strategy. The parallel algorithm is analysed on different architectures and difference parallel computing software. The numerical analysis of three grand challenge applications involving large

sparse matrices shown that the RGBS scheme is the superior iterative scheme followed by JB schemes.

From PPE for distributed memory architecture, proof that the PVM and MDCS software able to solve the parallel computing of numerical schemes and big data analytics in real life applications. For shared memory architecture, the PPE is analysed on speedup with the different number of threads. It concludes that, the higher of number of thread, the higher of speedup of parallel computing until it reached the optimum value for speedup. The numerical results and PPE prove that the parallel computing able to compute the mathematical model for big data applications. This paper shows that the PVM, MDCS and CUDA able to solve the parallel computing for numerical schemes and big data analytics in real life applications.

Acknowledgement

This research is supported in part by the Research Management Centre (RMC) and University Technology Malaysia (UTM) through Research University Grant (4F675) and the authors are grateful to the Ibnu Sina Institute, University Technology Malaysia for the excellent support to this research.

References

[1] Barney, B. 2012. *Introduction to Parallel Computing*. Lawrence Livermore National Laboratory.
[2] Boeriu, S., K. P. Wang, and J.C. Bruch Jr. *Lecture Notes on Parallel Computation.*
[3] Rauber, T. and G. Rünger.2013. *Parallel Programming: For Multicore And Cluster Systems*: Springer Science & Business Media.
[4] Alias, N., R. Anwar, C. R. C. Teh, N. Satam, N. Hamzah, Z. S. A. Ghaffar, R. Darwis and M.R. Islam. 2011. Performance Evaluation Of Multidimensional Parabolic Type Problems On Distributed Computing Systems. *2011 IEEE Symposium on Computers and Communications (ISCC)*. 103-110.
[5] Bader, D. A., W. E. Hart, and C. A. Phillips. 2005. *Parallel Algorithm Design For Branch And Bound*, In *Tutorials On Emerging Methodologies and Applications in Operations Research*. Springer. 5(1): 5-44.
[6] Moustafa, S., I. D. Malen, L. Plagne, A. Poncot, and P. Ramet. 2015. *Shared Memory Parallelism For 3D Cartesian Discrete Ordinates Solver.* Annals of Nuclear Energy. 82: 179-187.
[7] Chen, X. D. and A. S. Mujumdar. 2009. *Drying Technologies In Food Processing*. *John Wiley & Sons*.
[8] George, S., S. Cenkowski, and W. Muir. 2004. A Review Of Drying Technologies For The Preservation Of Nutritional Compounds In Waxy Skinned Fruit. *North Central ASAE/CSAE Conf, Winnipeg, Manitoba, Canada*.
[9] Alias, N., H. F. S. Saipol, and A. C. A. Ghani, 2014. Chronology of DIC Technique Based On The Fundamental Mathematical Modeling And Dehydration Impact. *Journal Of Food Science And Technology*. 51(12): 3647-3657.
[10] Alias, N., H. F. S. Saipol, and A. C. A. Ghani. 2012. Numerical Method For Solving Multipoints Elliptic-Parabolic Equation For Dehydration Process. in *Proceedings of 2nd Regional Conference On Applied And Engineering Mathematics (RCAEM-II)*.
[11] Mounir, S., P. Schuck, and K. Allaf. 2010. Structure And Attribute Modifications Of Spray-Dried Skim Milk Powder Treated By DIC (Instant Controlled Pressure Drop) Technology. *Dairy Science & Technology*. 90(2-3): 301-320.
[12] Behzad, T. and M. Sain. 2007. Finite Element Modeling Of Polymer Curing In Natural Fiber Reinforced Composites. *Composites Science and Technology*. 67(7): 1666-1673.
[13] Rutt, I. C., M. Hagdorn, N. R. J.. Hulton and A. J. Payne. 2009. The Glimmer Community Ice Sheet Model. *Journal of Geophysical Research: Earth Surface (2003–2012)*. 114(F2).
[14] Samelson, R. M. 2011. *The Theory Of Large-Scale Ocean Circulation*: Cambridge University Press.
[15] Geist, A. 1994. *PVM: Parallel Virtual Machine: A Users' Guide And Tutorial For Networked Parallel Computing*: MIT Press.
[16] Alias, N., A. C. A. Ghani, H. F. S. Saipol, N. Ramli and S. Q. M. Palil. 2012. Wave Equation For Early Detection Of Breast Cancer Growth Using MATLAB Distributed Computing. *2012 International Conference on Enabling Science and Nanotechnology, ESciNano 2012 - Proceedings*.
[17] Alias, N., R. Darwis, N. Satam and M. Othman. 2009. Parallelization of Temperature Distribution Simulations for Semiconductor and Polymer Composite Material on Distributed Memory Architecture. *Parallel Computing Technologies, Proceedings*. 5698: 392-398.
[18] Dziubak, T. and J. Matulewski. 2012. An Object-Oriented Implementation Of A Solver Of The Time-Dependent Schrödinger Equation Using The CUDA Technology. *Computer Physics Communications*. 183(3): 800-812.
[19] Sanderson, A. R., M. D. Miriah, R. M. Kirby and C. R. Johnson. 2009. A Framework For Exploring Numerical Solutions Of Advection–Reaction–Diffusion Equations Using A GPU-Based Approach. *Computing and Visualization in Science*. 12(4): 155-170.
[20] Alias, N., H. F. S. Saipol, and A.C.A. Ghani. 2012. Chronology of DIC Technique Based On The Fundamental Mathematical Modeling And Dehydration Impact. *Journal of Food Science and Technology*. 1-11.
[21] Alias, N. and M. R. Islam. 2010. A Review Of The Parallel Algorithms For Solving Multidimensional PDE Problems. *Journal of Applied Sciences*. 10(19): 2187-2197.
[22] Foster, I. 1996. Compositional Parallel Programming Languages. *ACM Transactions on Programming Languages and Systems (TOPLAS)*. 18(4): 454-476.
[23] Zulkifle, A. K. 1999. *Process Modelling Of Thermoset Composites*: University of Strathclyde.