Parallel Computing - A Case Study on Multicore Computing

Afiqah Ahmad [#1], Mohamed Faidz Mohamed Said [#2]

[#] *Faculty of Computer & Mathematical Sciences, Universiti Teknologi MARA*
70300 Seremban, Negeri Sembilan, MALAYSIA

[1] afiqah.aa95@gmail.com

[2] faidzms@ieee.org

***Abstract*—There are generally technical problems in integrating components in parallel computing. In could be overcome by applying certain mathematical approaches that would scientifically solve specific issues relating to parallelism. The improvement in computer technology by using numerical method will be mainly discussed in this paper. One of the issues is the installation of multi-core processor to achieve better performance during the completion of programs. The processing and completing one task commonly take too much time. The majority improvements of technology in computers involve mathematical methods. The prominent numerical techniques used are matrix computation, time reduction as well as linear system for computational area. Ordinarily, matrix unit computation is used to maximize on-chip resource utilization and to leverage the advantages of the current multi-core revolution to improve the performance of data-parallel applications. Besides that, branch and bound (B&B) algorithm, heterogeneous architecture and pipeline algorithm are also used in science and chemical area in order to confirm chemical reaction into living organism. In this paper, a further discussion of different approaches will be given with distinct algorithms and techniques that are used from previous researches. Based on those processes and methods for each area involved, some of it give good results and excellence performances.**

Keywords: **multi-core, branch and bound, parallel computing**

I. INTRODUCTION

Multi-core computing is a processing system composed of two or more independent cores or central processing units (CPUs). The cores are normally integrated onto a chip multiprocessor in a single chip package. Cores in a multi-core device may be combined together tightly or loosely. The first multiprocessor was in 1970's which was the Intel 4004 chip. The emergence of multi-core processor in 2000's is due to the limit of improvement in the single core manufacturing, and this turnover of idea also expands further the advancement of the microprocessor technology.

The basis of this case study is the improvement of multi-core by using different techniques. Apparently, multi-core is a pioneering type of processor that was implemented into the computer or smart phone back then. This special kind of multiprocessor has multiple instruction multiple data (MIMD) modes that include different cores executing different codes also known as threads which are operating on different data. In multi-core computer, processors have a shared memory where all cores share the same memory through some cache system.

Besides that, multi-processors are widely suitable beyond many new applications that are multithread including network, digital signal processing, and graphics. In computer architecture, multi-core could have better performance with much faster in cache coherency circuits in a single chip and smaller size compared to traditional symmetric multiprocessing (SMP). The advantages of multi-core systems are their source codes will often be unavailable and this could prevent compilation against the specific hardware configuration.

Nowadays, multi cores are used in optimization and the utilization has shown its improvements. The methods and techniques used are mathematical and technological approach such as superscalar, vector processing, multithreading and linear algebra. The improvement of performance depends on the software algorithms and its implementation. These multi-core products are manufactured by computer technologies companies that include AMD, ARM, Broadcom, Intel and VIA.

The paper is organized as follows. Section II will show the literature review for completing this case study. Section III describes the methodology that is related with multi-core computing. Finally, Section IV concludes this paper.

II. LITERATURE REVIEW

The purpose of this project [1] is to describe the techniques and methods used in certain situation. At first, B&B algorithm parallelizing technique is considered suitable for solving optimization problems in large number of computational fields such transportation, logistics and others. In [1] it states that the search space in B&B represents as tree whose root node represents the original unsolved problem while the internal nodes are partially solved sub-problems. In [2] they use the same method of B&B algorithm for solving large combinatorial problems on GPU-enhanced multi-core machines with two leads scenario such as concurrent (RLL-GB&B) and cooperative (PLL-GB&B). Furthermore, there are some techniques used in order to improve multi-core processor performance. Common unit matrix is proposed in order to maximize on-chip resource utilization and performance of DLP application [3]. This technique is extended with common unit matrix on single core exploited on ILP and DLP only. The process of exploiting on ILP and

DLP is used by SystemC where it stimulates and evaluates its performance. There are huge differences in performance levels between multi-core and extended single-core when matrix unit is used. Additionally, matrix computations are applied in implementing matrix operations at huge and intermediate scales by using OpenMP and SWARM models that generate better result in running parallel [4]. Fig. 1 below shows the features of multi-core programming models while Fig. 2 displays some matrix computations and related methods. The result of matrix unit demonstrates that multi-core extended has better performance compared to single-core extended.

Features of multi-core programming models.

Feature – model	Pthreads	OpenMP	Cilk Plus	TBB	SWARM	FastFlow
Implementation	Library	Compiler	Language	Library	Library	Library
Data partitioning	Explicit	Implicit	Implicit	Implicit	Implicit	Explicit
Synchronization	Explicit	Implicit	Implicit	Implicit	Implicit	Explicit
Parallel patterns	fork-join	fork-join map reduction	fork-join map reduction	fork-join map reduction scan pipe	map reduction scan	farm pipe

Fig. 1. Features of multi-core programming models

Some matrix computations and related methods.

Matrix computations			
Computational kernel	Operation	Computation	Memory
Vector addition	$c_i = a_i + b_i,\ 1 \le i \le n$	n	$3n$
Inner or dot product	$z = x^T \cdot y = \sum_{i=1}^{n} x_i \cdot y_i$	$2n$	$2n+1$
Matrix addition	$c_{ij} = a_{ij} + b_{ij},\ 1 \le i, j \le n$	n^2	$3n^2$
Outer product	$z_{ij} = x \cdot y^T = x_i \cdot y_j,\ 1 \le i, j \le n$	n^2	n^2+2n
Matrix – vector product	$y_i = \sum_{j=1}^{n} a_{ij} x_j,\ 1 \le i \le n$	$2n^2$	n^2+3n
Matrix transpose	$a_{ji}^T = a_{ij},\ 1 \le i, j \le n$	n^2	$2n^2$
Matrix product	$c_{ij} = \sum_{k=1}^{n} a_{ik} \cdot b_{kj},\ 1 \le i, j \le n$	$2n^3$	$3n^2$
Solving linear systems			
Computational kernel	Operation	Computation	Memory
Gaussian elimination	division: $l_{ik} = a_{ik}^k / a_{kk}^k,\ k+1 \le i \le n$ elimination: $a_{ij}^{k+1} = a_{ij}^k - l_{ik} a_{kj}^k$ $b_i^{k+1} = b_i^k - l_{ik} b_k^k,\ k \le i, j \le n$	$\frac{2}{3}n^3$	n^3+n^2
Jacobi method	$x_i^{(k)} = \frac{1}{c_{i,i}} (b_i - \Sigma_{j \ne i} a_{i,j} x_j^{(k-1)}),\ 1 \le i \le n$	$4n^2$	$k(n^2+3n)$

Fig. 2. Some matrix computations and related methods

On the other hand, the advantages of multi-core parallelism include using Stencil computations which are the base of solving problems. In parallelism, there are major issues and challenges due to critical parameters such current architectural features, impacts on mechanisms of cache memory, vectorization and compilation [5, 6]. For example, limited parallelism in current design makes the process on multi-core platform difficult or impossible to apply [7]. This importance of numerical kernel used is shown in the multi-level optimization strategy and Machine Learning (ML). These strategies are applied in order to combine manual vectorization, space tiling, stencil composition and to predict stencil kernel performance on multi-core architecture. Both techniques present a comparison to succeed the objectives. For multi-level strategy, results are compared to Pochoir frameworks with different set of three compilers that are Intel, Clang and GCC used on multi-core platforms [5]. There are some other programming frameworks used in parallel scientific computations with several parameters such as Pthread, OpenMP, Intel Cilk Plus, Intel TBB, SWARM and FastFlow. Among the frameworks, OpenMP and SWARM models are tested to produce the best results running in parallel with compiler optimisation [4]. Meanwhile the ML demonstrates two different kernels which are the 7-point Jacobi and seismic wave modelling. These kernels are utilized in order to achieve the effectiveness of its approach. Numerical kernel also uses auto-tuning method to achieve better performances on multi-core processor. Instead of Stencil computations techniques mentioned, certain problem is solved by using numerical equation which is Partial Differential Equations (PDEs). This model is used for weather forecast which gathers the existences of PDEs whereas the calculations need data from several neighbouring mesh nodes to decompose computational domain [8].

III. METHODOLOGY

Parallelism is also presented by applying pipeline parallel techniques. There are adaptive pipeline parallel scheme (AD-PIPE), power efficient version (AE-PIPE) and pipeline multithreading (PMT). These schemes exhibit wide applicability in parallelizing general sequential programs on multi-core processor [7, 9]. Usually, AD-PIPE is used to adjust the thread number in different levels according to its imbalances workloads dynamically that achieve a stable partition for constant input workloads. Then, AE-PIPE is added via scheduling threads depending on variable input workloads. While, PMT performance is limited in remarkable inter-core communication, the clustered pipelined multithreading (CPMT) presented has enhanced performance due to accelerated sequential programs on commodity multi-core processor. CPMT provides very low average by eliminating false sharing as well as reducing communication operation and transit delays in the software-only approach. In addition, chemical area also uses metabolomics pipeline techniques for identification of chemical reactions on living organisms. Particular chemical elements of atoms and chemical compounds of molecules exist in samples of cells, body fluids and others [10].

Next, heterogeneous multi-core architectures are developed to upgrade multi-core efficiency [11-13], which means data storage and data sharing between multiple heterogeneous cores have new opportunity on memory system architecture. Heterogeneous architecture for hardware use Pattern Aware Memory Systems (PAMS) which supports static and dynamic data structure and also has comparison with Baseline system to prove the hardware mechanism [12]. Improved design of heterogeneous architecture for Internet of Things (IoTs) is achieved by applying a multi-core Task-Efficient Sink Node (TESN). Moreover, the test is done to maximize its computing efficiency by applying the designed Weighted-Least Connection (WLC) task schedule strategy. In IoTs, there are two types of cores in the sink node which are master and slave cores [11] that deal with tasks allocation and data processing for each sink node. The comparison between TESN and WLC is made to confirm which strategy is the best. Then, experimental results are shown that TESN is the better strategy because its well efficiency, load balance, lower congestion and it speeds up the process time of sensor nodes clearly.

Besides that, maximization on computer efficiency also involves the time aspect such as confirming the execution time on processing the system. Thus, multiprocessor-system-

on-chip (MPSoCs) has achieved its goal for reducing schedule time by using holistic approach to resource partitioning and multiple embedded applications of task scheduling under memory awareness [14]. When it involves time, usually it is also concerning the accelerators. Nowadays, there are so many application developers that use combinations of programming models in order to utilize the performance of the system [13]. Successively, dOpenCL approach is used in implementing the existing OpenCL programming model for distributed systems, and the same goes to its extension for running multiple applications concurrently.

IV. Conclusion

In this paper, the algorithms and methods being discussed have different objectives in order to make changes from single-core into multi-core processors. There are several algorithms required to enhance the performances of single core processor in term of run-time system. In addition, the existing single processor performances could be improved by developing another technique in order to come out with multi-core processor. Therefore, specifically by using proper suitable algorithms and techniques, highly-effective executions can be achieved to solve issues and problems in multi-core architecture.

REFERENCES

[1] T.-T. Vu and B. Derbel, "Parallel Branch-and-Bound in Multi-Core Multi-CPU Multi-GPU Heterogeneous Environments," *Future Generation Computer Systems*, vol. 56, no. Supplement C, pp. 95-109, 2016/03/01/ 2016, doi: https://doi.org/10.1016/j.future.2015.10.009.

[2] I. Chakroun, N. Melab, M. Mezmaz, and D. Tuyttens, "Combining Multi-Core and GPU Computing for Solving Combinatorial Optimization Problems," *Journal of Parallel and Distributed Computing*, vol. 73, no. 12, pp. 1563-1577, 2013/12/01/ 2013, doi: https://doi.org/10.1016/j.jpdc.2013.07.023.

[3] M. I. Soliman and A. F. Al-Junaid, "A Shared Matrix Unit for a Chip Multi-Core Processor," *Journal of Parallel and Distributed Computing*, vol. 73, no. 8, pp. 1146-1156, 2013/08/01/ 2013, doi: https://doi.org/10.1016/j.jpdc.2013.03.004.

[4] P. D. Michailidis and K. G. Margaritis, "Scientific Computations on Multi-Core Systems Using Different Programming Frameworks," *Applied Numerical Mathematics*, vol. 104, no. Supplement C, pp. 62-80, 2016/06/01/ 2016, doi: https://doi.org/10.1016/j.apnum.2014.12.008.

[5] G. Sornet, F. Dupros, and S. Jubertie, "A Multi-level Optimization Strategy to Improve The Performance of Stencil Computation," *Procedia Computer Science*, vol. 108, no. Supplement C, pp. 1083-1092, 2017/01/01/ 2017, doi: https://doi.org/10.1016/j.procs.2017.05.217.

[6] V. Martínez, F. Dupros, M. Castro, and P. Navaux, "Performance Improvement of Stencil Computations for Multi-core Architectures Based on Machine Learning," *Procedia Computer Science*, vol. 108, no. Supplement C, pp. 305-314, 2017/01/01/ 2017, doi: https://doi.org/10.1016/j.procs.2017.05.164.

[7] Y. Lu, Y. Li, B. Song, W. Zhang, H. Chen, and L. Peng, "Parallelizing Image Feature Extraction Algorithms on Multi-Core Platforms," *Journal of Parallel and Distributed Computing*, vol. 92, no. Supplement C, pp. 1-14, 2016/05/01/ 2016, doi: https://doi.org/10.1016/j.jpdc.2016.03.001.

[8] C. V. P. Mohan and P. Talukdar, "Three dimensional numerical modeling of simultaneous heat and moisture transfer in a moist object subjected to convective drying," *International Journal of Heat and Mass Transfer*, vol. 53, no. 21-22, pp. 4638-4650, 2010.

[9] Y. Zhang, G. Xiao, and T. Baba, "Accelerating Sequential Programs on Commodity Multi-Core Processors," *Journal of Parallel and Distributed Computing*, vol. 74, no. 4, pp. 2257-2265, 2014/04/01/ 2014, doi: https://doi.org/10.1016/j.jpdc.2013.12.009.

[10] M. M. Jaghoori *et al.*, "PMG: Multi-Core Metabolite Identification," *Electronic Notes in Theoretical Computer Science*, vol. 299, no. Supplement C, pp. 53-60, 2013/12/25/ 2013, doi: https://doi.org/10.1016/j.entcs.2013.11.005.

[11] T. Qiu, A. Zhao, R. Ma, V. Chang, F. Liu, and Z. Fu, "A Task-Efficient Sink Node Based on Embedded Multi-Core soC for Internet of Things," *Future Generation Computer Systems*, 2016/12/23/ 2016, doi: https://doi.org/10.1016/j.future.2016.12.024.

[12] T. Hussain, "A Novel Hardware Support for Heterogeneous Multi-Core Memory System," *Journal of Parallel and Distributed Computing*, vol. 106, no. Supplement C, pp. 31-49, 2017/08/01/ 2017, doi: https://doi.org/10.1016/j.jpdc.2017.02.008.

[13] P. Kegel, M. Steuwer, and S. Gorlatch, "dOpenCL: Towards Uniform Programming of Distributed Heterogeneous Multi-/Many-Core Systems," *Journal of Parallel and Distributed Computing*, vol. 73, no. 12, pp. 1639-1648, 2013/12/01/ 2013, doi: https://doi.org/10.1016/j.jpdc.2013.07.021.

[14] H. Salamy, "An Effective Approach to Schedule Time Reduction on Multi-Core Embedded Systems," *Computers & Electrical Engineering*, vol. 64, no. Supplement C, pp. 15-33, 2017/11/01/ 2017, doi: https://doi.org/10.1016/j.compeleceng.2016.07.001.

[15] A. Ahmad. (2017). Parallel Computing – A Case Study on Multi-core Computing. Available: https://www.youtube.com/watch?v=fdMn-3owkxg&rel=0 [Accessed: 28-Nov-2017]