ISSN: 2277-9655
Impact Factor: 3.785

IJESRT
INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

# REVIEW OF PARALLEL COMPUTING, PARALLEL PROGRAMMING AND ITS APPLICATIONS

**M.Narayana Moorthi.*, Dr.R.Manjula**
School of computing science and Engineering,VIT University, Vellore-14., TN, India

**DOI**: 10.5281/zenodo.51543

## ABSTRACT

The next generation computing systems are focusing on parallel computing to solve the problem in fast and high speed using parallel programming concepts. The running time of any complex algorithm or applications can be improved by running more than one task at the same time with multiple processors concurrently. Here the performance improvement is measured in terms of increase in the number of cores per machine and is analyzed for better Energy Efficient optimal work load balance. In this paper we investigate the review and literature survey of parallel computers for future improvements. We aim to present the theoretical and technical terminologies for parallel computing. The focus here is the process of comparative analysis of single core and multicore systems to run an application program for faster execution time and optimize the scheduler for better performance.

**KEYWORDS**: Multicore programming, Parallel Computer, Parallel Program, High speed Computing

## INTRODUCTION

The parallel programming [1] [2] [3] helps in identification and assignment of independent tasks or works to multiple processor cores. Now days everyone is having desktop and laptop with multiple processor [4] [5] [6] [7] [8] cores. So it is necessary for us to understand the various concepts and methods involved in parallel programming and parallel computing to make use of next generation computers more effectively. The processing capability of Microprocessor is increasing in terms of clock speed and inclusion of more and more execution units with pipeline support in a chip to support parallelism. According to Moore's law the numbers of transistors are doubled in a CPU every 18 months. The Moore's law is restated that the numbers of processor cores are doubling in every 18 months with respect to parallel computers. The parallel computer is defined as the collection of more than one processing element or cores to solve a large size problem in fastest rate. The GPGPU, Multicore and Many core processors are replacing the current world of computing systems. Processing the task in parallel [9] is the computing challenge. The way we measure the performance is the running time or execution time of the program or application. The method to reduce the running time of a program is through high speed computing which can be done through parallel programming and parallel computing machines. In case of parallelization, Amdahl's Law states that if Px is the proportion of a program that can be made parallel and (1-Px) is the proportion that cannot be parallelized, then the maximum speedup that can be achieved by using Nx Processors is 1/ (1-Px) +Px/Nx. Balancing the workload among threads or cores or processors is critical to application performance. The key objective for load balancing is to minimize idle time on threads and share the workload equally among all cores and threads with minimal work sharing overheads.

### Multi-core Architecture Overview

Computer [10] [11] [12] [13] [14] [15] [16] is an electronic digital machine which takes instructions to solve the given problem or application. Any computer machine consists of Input devices like keyboard, mouse and output device like monitor and printer. It has important device to compute which is known as CPU (Central Processor Unit) or Microprocessor. It also has many types and sizes of memory for instruction and data storage. The fundamental computer machine is identified as Von-Neumann Architecture. It has only one processor or single core to solve the

given problem. The program is executed sequentially. To improve the run time of program parallelism is introduced with different levels. The Multicore Processor Architectures [17] [18] [19] [20] [21] [22] consist of one or more processor cores to provide higher levels of parallelism. The parallel programming approach is practiced to get high performance within the system. A multi-core processor is composed of two or more independent cores. Manufacturers typically integrate the cores onto a single integrated circuit die (known as a chip multiprocessor or CMP), or onto multiple dies in a single chip package.

**Speedup**: The measure of speedup is as follows. For a problem of size *n*, the expression for speedup is: *Sp= Ts* (*n*, 1) / *T* (*n*, *p*), Where *Ts* (*n*, 1) is the time of the best sequential algorithm and *T* (*n*, *p*) is the time of the parallel algorithm with *p* processors, both solving the same problem. The speedup increases with increase in number of processor cores.

The quality of an algorithm [23] either sequential or parallel is measured using various metrics like the running time, space, speedup, efficiency , memory access time, hit ratio, Floating point operations per second, etc.
Complexity Analysis: An algorithm is a finite sequence of steps well defined to solve a given problem [19]. In many case there will be more than one method or algorithm to solve the given problem. Here the choice or selection of an algorithm depends on efficiency or complexity of an algorithm. The complexity of an algorithm measures the running time and or storage space required for the algorithm. The following notations are used to measure the complexity of an algorithm. Given M1 is an algorithm, n1 is the input data size, f1(x) is the output

Time Complexity of sequential algorithm (vector operations) =O (n)
Time Complexity of parallel algorithm (vector operations) = O (n/p);
P is the number of processor cores

The sequential algorithm runs on only one core or processor. The parallel algorithm runs on many cores or processors. The two measures of complexity of an algorithm are as follows. Worst case (Maximum time or space) and Average case (Expected Value) plays an important role to select the algorithm.

**Parallel Programming Languages - Overview of Open MP**

Open MP [10] [11] [12] [13] [14] [15] [16] [17] Open Multiprocessing is a shared memory parallel programming Language. It has the collection of compiler directives, library functions and environment variables. Open MP follows the Fork-Join Model for the execution of parallel program. This Open MP can be used for shared memory parallelism in C /C++ [8] [9] Programs. When the program starts the single thread named as master thread is active. This active thread executes the sequential portion of the program. When parallel operations ae required the master thread creates additional threads which are known as fork. In the parallel region of the code the master and additional threads executes. At the end of parallel region the additional threads are destroyed and only master thread continues this is known as join. A program written in Open MP executes as a master or single thread as same as sequential program. When the parallel construct is encountered, the master thread creates the number of threads and each thread computes the task simultaneously or concurrently. A compiler directive in C or C++ is called a pragma. The following tools and software is necessary for Open MP programming. The visual studio 2010, Intel C++ Compiler, Open MP Language.
The following steps illustrate the proposed work for Performance Study, Evaluation and analysis of single core and Multicore Machines.
Step1: Identify and define the large size problem to be solved
Step2: Identify the sequential algorithm to solve the above problem
Step3: Run this sequential algorithm code in a single core machine
Step4. Measure the various performance parameters (Like Total Execution time or Run time, Cache memory size, hit ratio, CPU utilization, etc)
Step5: Identify the concurrency in the above application code (Independent sub tasks)
Step6: Write the parallel code using parallel programming languages (Open MP)
Step7: Run the above code using multicore machines
Step8: Measure the various performance parameters (Like Total Execution time or Run time, Cache memory size, hit ratio, CPU utilization, etc)
Step9: Rerun the above code using various thread size or processors cores (2, 4, 8, 16, and 32, n) and make a comparative analysis table.

**Experimental Analysis:**

Sorting Techniques: The most important application algorithm in computer science is the sorting of 'n' elements. There are various algorithms exist to sort the array of 'n' elements. The following bubble sort, selection sort, insertion sort and quick sort algorithms were implemented for sequential and parallel codes. Few observations on the experiment is as follows. The sorting algorithm's performed take almost same time to sort the data on sequential and parallel running with minor differences. It is observed that when the data is large the performance gain of parallel computing plays important role. As observed in all the algorithms the smaller the data sent the less efficient the parallel computing so parallel computing is more suitable for larger data elements.

***Table 1 : Selection sort***

| Number of elememts | Sequential time | Parallel time |
|---|---|---|
| 10 | 0.007813 | 0.015625 |
| 30 | 0.039063 | 0.031280 |
| 60 | 0.078125 | 0.054688 |

***Table 2: Insertion sort***

| Number of elements | Sequential time | Parallel time |
|---|---|---|
| 20 | 0.015625 | 0.05688 |
| 80 | 0.0625 | 0.054688 |
| 100 | 0.08593 | 0.054688583 |

***Table 3: Bubble sort***

| Number of elements | Sequential time | Parallel time |
|---|---|---|
| 30 | 0.039063 | 0.023438 |
| 80 | 0.062500 | 0.062500 |
| 100 | 0.125000 | 0.070313 |

***Table 4: Quick sort***

| Number of elements | Sequential time | Parallel time |
|---|---|---|
| 50 | 0.031250 | 0.039063 |
| 100 | 0.117188 | 0.078125 |

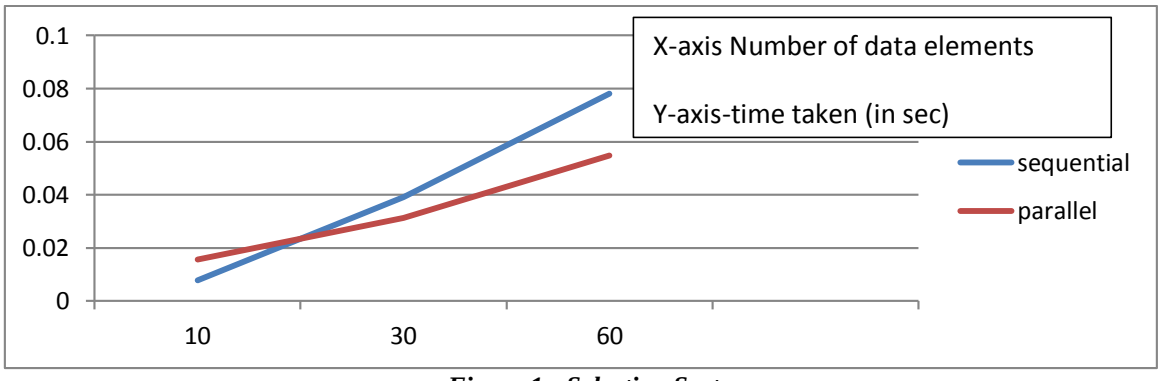


***Figure 1 : Selection Sort***

ISSN: 2277-9655
Impact Factor: 3.785

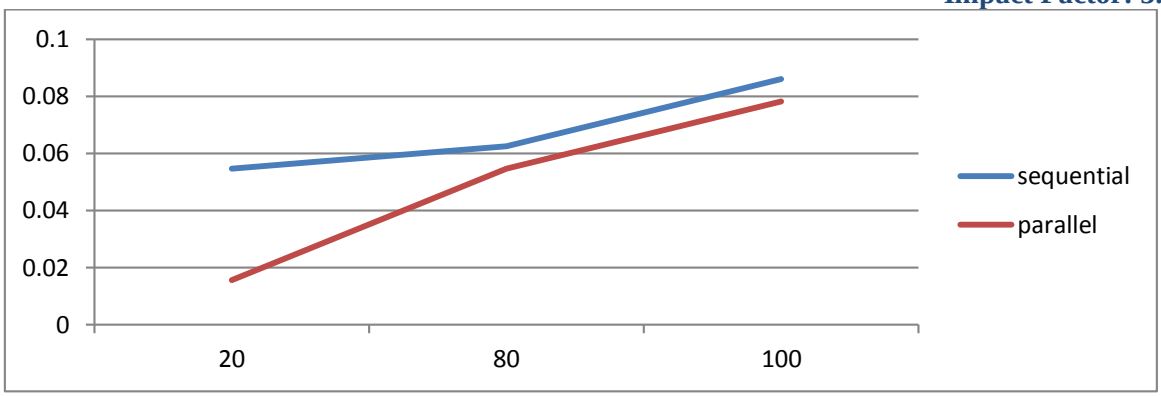


*Figure 2: Insertion sort*

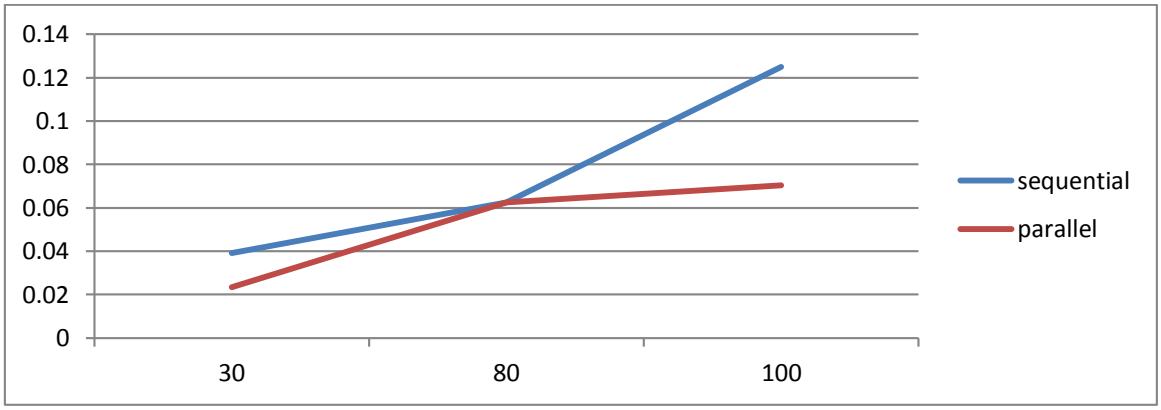


*Figure 3: Bubble Sort*

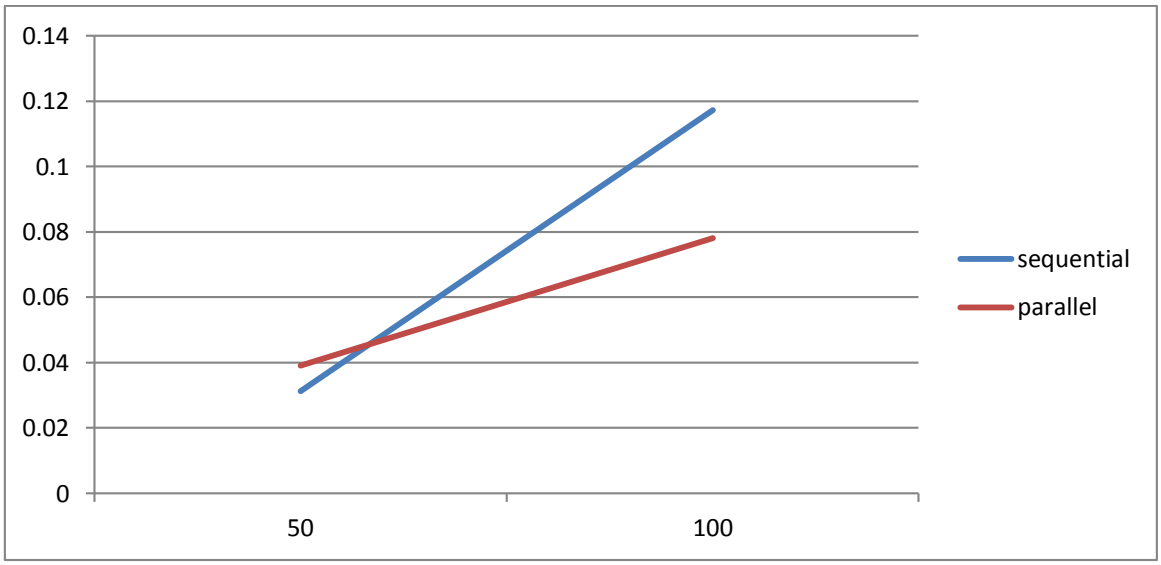


*Figure 4 : Quick Sort*

## CONCLUSION AND FUTURE WORK

Here in this paper we have studied the single core and multicore machine operations. We also analyzed a sample application code sorting algorithms operation on single core and multicore machines. It is observed that for small "n" value we do not see any difference in serial and parallel computations. If the data size is more we get the benefit of parallel computing by running multiple tasks continuously and concurrently with multiple cores. The future and or next generation computers are equipped with many core and GPGPU Systems. It has more benefits in terms of processing cores but it is also giving open challenge for the software developers to make use of multiple cores simultaneously for parallel computing. So it is time for us to learn the parallel programming model to address the increasing processor cores in the next generation computers.

## REFERENCES

1. J. L. Hennessy, D. A. Patterson, "Computer Architecture: A Quantitative Approach", 2nd Edition, Morgan Kaufmann Publishing Co. 1996.
2. Kai Hwang, "Advanced Computer Architecture parallelism, scalability, programmability", Tata McGraw hill edition 2006
3. David E.Culler, Jaswinder Pal Singh , Anoop Gupta, "Parallel Computer Architecture a hardware / software approach" , Elsevier .
4. Frank Schirrmeister, "Multi-core Processors: Fundamentals, Trends, and Challenges', *Embedded Systems Conference* 2007 ESC351, Imp eras, Inc.
5. Muti-Core Processors, The Next Evolution in Computing
6. http://multicore.amd.com/Resources/33211A_Multi-Core_WP_en.pdf
7. Intel Multi-core technology, http://www.intel.com/multicore/
8. Blair Guy,"An Analysis of Multicore Microprocessor capabilities and their suitability for current day application Requirements", Bowie University, Maryland in Europe, Nov 2007.
9. Herb Sutter, "The free lunch is over: A fundamental turn toward concurrency in software", Dr.Dobb's Journal, 30(3), March 2005
10. Herb Sutter," The concurrency revolution" in C /C++ users Journal, 23(2), February 2005.
11. Michael J.Quinn,"Parallel programming in C with Open MP", Mc Graw Hill Edition
12. https://software.intel.com/en-us/blogs/2008/12/31/top-10-challenges-in-parallel-computing
13. Overview of Performance Measurement and Analytical Modeling Techniq... http://www1.cse.wustl.edu/~jain/cse567-11/ftp/multcore/index.html Garrison Prinslow, gprinslow@gmail.com (A paper written under the guidance of Prof. Raj Jain)
14. www.openmp.org
15. www.intel.com
16. M.Narayana Moorthi, P.Mohan Kumar, Dr.J.Vaideeswaran.,"Overview of application performance on Multicore environment".,IJARCS, Volume 1,No.4,Nov-Dec 2010
17. Rajnish Dashora, Harsh P. Bajaj, Akshat Dube, Narayanamoorthy M, "ParaRMS Algorithm:A Parallel Implementation of Rate Monotonic Scheduling Algorithm Using OpenMP", School of Computing Sciences and Engineering,VIT University,Vellore, India,dashora.rajnish@gmail.com, School of Computing Sciences and Engineering,VIT University,Vellore, India,harshpbajaj@yahoo.co.in, School of Electrical Engineering,VIT University,Vellore, India,akshatdube.28@gmail.com , School of Computing Sciences and Engineering,VIT University,Vellore, India,mnarayanamoorthy@vit.ac.in, IEEE Explore
18. Pijush Samui,"Handbook of Research on Computational Techniques for Simulation based Engineering" IGI Global Publications – 2015.
19. V.Rajaraman, C.Sivaram Murthy, "Parallel computers Architecture and Programming ", prentice hall of india(p) Ltd 2003.
20. Seymour Lipschutz, Marc Lars Lipson, " Theory and Problems of Discrete Mathematics", second edition, Tata Mcgraw Hill Publishing company Ltd
21. Vijay Anand Korthikarti,Gul Agra, "Analysis of parallel Algorithms for Energy Conservation in Scalable Multicore Architectures" , International conference on parallel processing 2009.
22. Euiseong Seo, Jinkyu Jeong, Seonyeong park and Juonwon Lee, "Energy Efficient scheduling of Real time tasks om Multicore Processors", IEEE Transactions on parallel and distributed systems,Vol-19,No.11,November 2008.

ISSN: 2277-9655
Impact Factor: 3.785

23. Wan yeen Lee, "Energy Efficient scheduling of periodic Real time tasks on lightly loaded Multicore Processors", IEEE Transactions on parallel and distributed systems, vol 23,No3,March 2012.
24. M.Narayana Moorthi, R.Manjula,"Performance Evaluation and Analysis of Parallel Computers Workload International Journal of Grid and Distributed Computing", Vol. 9, No. 1 (2016), pp.127-134 http://dx.doi.org/10.14257/ijgdc.2016.9.1.13