VERIFICATION OF INCREASED EFFICIENCY OF PROGRAMS WHEN EXECUTED USING PARALLEL COMPUTING FEATURES

Arshiya A. Sheikh
Master of Engineering (M.E.) Scholar
Department of Computer Science & Engineering,
Sant Gadge Baba Amravati University, Amravati, India

Abstract : The past few decades have seen large fluctuations in the perceived value of parallel computing. At times, parallel computation has optimistically been viewed as the solution to all of our computational limitations. Increasingly, parallel processing is being seen as the only cost-effective method for the fast solution of computationally large and data-intensive problems. The emergence of inexpensive parallel computers such as commodity desktop multiprocessors and clusters of workstations or PCs has made such parallel methods generally applicable, as have software standards for portable parallel programming. This sets the stage for substantial growth in parallel software. This research work is to verify the efficiency of Program when executed using parallel computing features. The Program will execute on single processor and same program is executed on Parallel Computing environment. The efficiency of the program in terms of time and space will be calculated and tabulated. The results are intended to verify the increased efficiency of programs when executed using parallel computing environment by plotting different features with an understanding of the approximate performance of the language implementations. The purpose of this research is to reduce the time required for processing and execution of a program.

Keywords **– Parallel Computing, Program Slicing and Verifying efficiency**

1. Introduction:

The Computer Technology has experienced four generation of vast development in past four decades. Physically starting from relays to very large scale integrated devices. Increase in Computer speed from decades has improved the computer performance. However; still the computer system is not the sole factor contributing to high performance. A Computer System is really a composite of such items as Processors, Memories, Functional Units, Interconnection Networks, Compilers, Operating Systems, Peripheral Devices, Communication Channels and Database Bank. The good understanding of the underlying hardware, software system structure and Computer Algorithms to be implemented on computer system can design a powerful and cost effective computer system architecture which can be solved a large computational problem in an efficient manner. These disciplines constitute the technical scope of computer architecture.

Computer Architecture is a system concept integrating hardware, software, algorithms, and language to perform large computations. A good computer architect should master all this disciplines. Modern computers are equipped with powerful hardware facilities driven by extensive software packages. According to Sidney Fernbach: “Today’s Large Computers (Mainframes) would have been considered ‘Super-Computers’ 10 to 20 years ago. By the same token, today’s Super-Computers will be considered ‘State of the Art’ standard equipment 10 to 20 years from now.”

As the hardware Component of the Computer System changes from decades to decades, Operating System of the Computer has been improved in four phases:

- Batch Processing
- Multiprogramming
- Time Sharing
- Multiprocessing

In four Phases of operating system the level of Parallelism has increases gradually.

2. Parallel Processing:

Parallel processing is an important part of any high performance computing model. Parallel Processing involves the use of computing resources such as CPU and Memory to complete the task. Parallel processing involves the division of a task into several sub tasks and making the system work on each of these smaller tasks in parallel. If multiple nodes or processor are engage in doing a computational task it execution time is faster than the single processor. Parallel processing improves the response time and throughput by utilizing all the computing resources in the network.

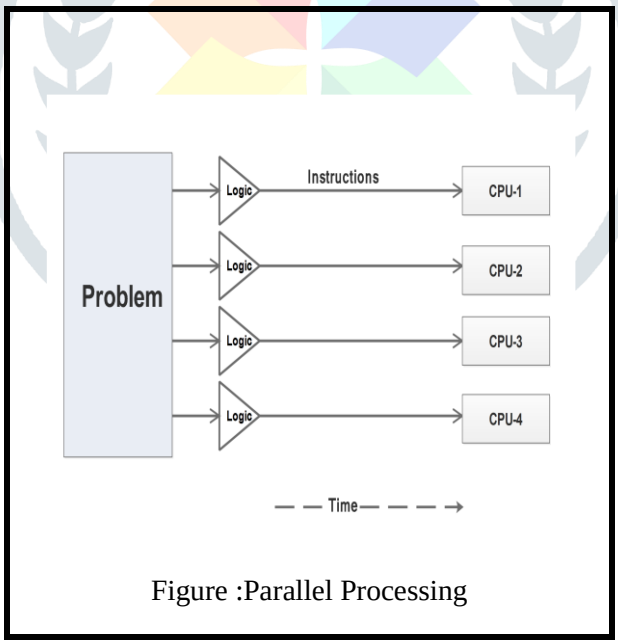

Figure :Parallel Processing

3. Task Parallelism:

The task of taking an existing sequential program and parallelizing it is extremely hard. It is not easy to catch all the dependences in the program, especially if it is not a program with which we are familiar. If the program is design for task parallelism then it will automatically translates ordinary program into efficient Parallel Program.Task Parallelism involves the breaking of a task into sub-tasks and then allocating each sub-task to different node or processor for execution. Task Parallelism does not usually scale with the size of a problem [5].

The pseudocode below illustrates task parallelism

```
if      Node= "1" then
                do task "X"
else   if    Node = "2" then
                do task  "Y"
end if
```

4. Flowchart:

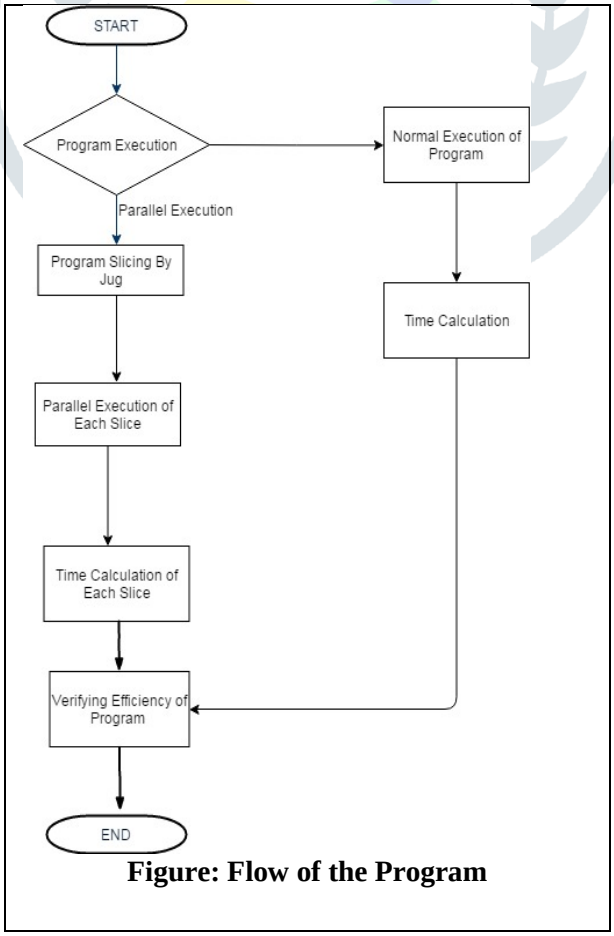

Figure: Flow of the Program

5. Results:

The System Developed is checked with the different sets of program. The Comparisons of Programs on Single system and Parallel execution of the same program is done and plot in the graph. The System is tested on number of Program with different data values. The three nodes with different configuration are used to test the functioning of System. Out of three one node acts as a Server and remaining two nodes as a client node. The Table shows Program Execution time on single system (column no.2) and program slices execution time on client node. (Column no.3 & column no.4). The system is tested on different programs. The result of different program with different data elements are shown in the table.

Table: Program Execution Time on Single Processor and Execution time of Program slices on Client 1 & Client 2.

Sr.No	Program Execution on Single Processor	Program Slice Execution time on Client 1	Program Slice Execution time on Client 2
1	0.001000165939331	0.03496	0.02813
2	0.0010001659393331	0.7532	0.6521
3	0.0009999275207519	0.6321	0.7845
4	0.0020003318786621	0.214	0.2356
5	0.0010001659393310	0.364	0.316
6	0.0009999275207519	0.4316	0.356
7	0.0010001659393310	0.764	0.645
8	0.0009999275207519	0.9454	0.8247
9	0.0009999275207519	0.2383	0.2356
10	0.0009999275207519	0.4867	0.4533

According to the Table we get the following chart of Program execution time on single system and client nodes.

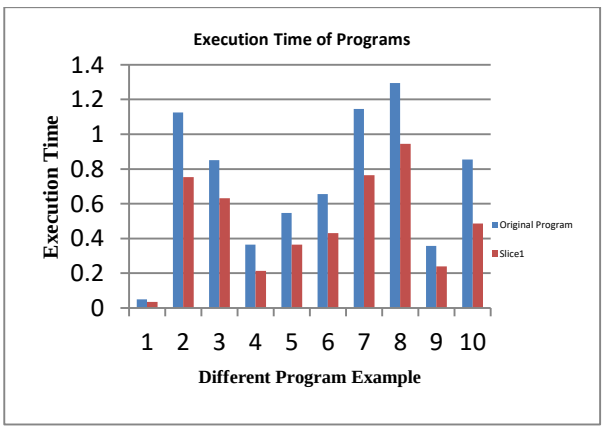

Figure: Execution Time of Program with different example

The Figure shows the execution time of ten different programs in distributed parallel computing tool, using the program slicing concept and parallel computing. From the Table the time required for execution of the Program slice is always less than the original program and the time required to execute the original program using the Distributed Parallel Computing Tool is quite less than the conventional method.

6. Conclusion:

While working on the Parallel Computing Architecture we came to know that, parallelism presents new challenges in writing correct and efficient code. As the trend of increasing parallelism at the hardware level will continue for the foreseeable future, parallel computation become more and more important in application programming. There is a very active body of research on making parallelism easier and less error-prone for programmers. The research presents in the thesis; shows parallel computing increase the efficiency of the program when used with parallel computing features. As the number of processing unit increase the time consumed by the program decreases. The Research work present Novel Model of Task Parallelism using Python framework which can be extended to Machine Learning in future.

7. References:

1. Michael Flynn “Very High-Speed Computing Systems”, IEEE, January 1967, pp 1901-1909
2. Xing Cai, Hans Langtangen “On the Performance of the Python Programming Language for Serial and Parallel Scientific Computation”, ResearchGate, January 2005
3. Muhammad Rashid, Damien Picard, B. Pottier “Application Analysis of Parallel Processing”, ResearchGate, September 2008
4. Lisandro Dalcin, Rodrigo Paz, Pablo Kler, Alejandro Cosimo “Parallel Distributed Computing using Python”, ResearchGate, September 2011
5. Inderpal Singh “Review on Parallel and Distributed Computing”, SJET, 2013, pp 218-225
6. Cristobal Navarro, Naincy Hitschfeld, Luis Mateu “A Survey on Parallel Computing and its Application in Data – Parallel Problems Using GPU Architecture”, ResearchGate, September 2013
7. Waide B. Tristram, Karen L. Bradshaw “Hydra:APythonFrameworkforParallel Computing”, Department of Computer Science, Rhodes University, Grahamstown, South Africa, March 2015
8. Luis Coelho “Jug: Software for Parallel Reproducible Computation in Python”, JORS, 2017, pp 1-10