INTERNATIONAL RESEARCH JOURNAL IN ADVANCED ENGINEERING AND TECHNOLOGY (IRJAET) E - ISSN: 2454-4752 P - ISSN : 2454-4744
VOL 3 ISSUE 4 (2017) PAGES 2287 - 2291
RECEIVED : 25.07.2017 PUBLISHED : 21.08.2017
Aug 21, 2017

SHARED MEMORY BASED PARALLEL PROGRAMMING MODELS

Ms.R.Kavitha[1], Mr.K.Jayasankar M.Sc,M.Phil,B.Ed[2] ,
[1]M.Phil, Research Scholar, K.M.G College Of Arts & Science, Gudiyattam,
[2] Assistant Professor, PG & Research Department Of Computer Science & Applications, K.M.G College Of Arts & Science, Gudiyattam.

ABSTRACT

Parallel programming models are quite challenging and emerging topic in the parallel computing era. These models allow a developer to port a sequential application on to a platform with more number of processors so that the problem or application can be solved easily. Adapting the applications in this manner using the Parallel programming models is often influenced by the type of the application, the type of the platform and many others. There are several parallel programming models developed and two main variants of parallel programming models classified are shared and distributed memory based parallel programming models. The recognition of the computing applications that entail immense computing requirements lead to the confrontation of the obstacle regarding the development of the efficient programming models that bridges the gap between the hardware ability to perform the computations and the software ability to support that performance for those applications.

Keywords: Parallel computing, Platform, Distributed memory.

1. INTRODUCTION

To answer this challenge this thesis confines and compares four different shared memory based parallel programming models with respect to the development time of the application under a shared memory based parallel programming model to the performance enacted by that application in the same parallel programming model. The programming models are evaluated in this thesis by considering the data parallel applications and to verify their ability to support data parallelism with respect to the development time of those applications. The data parallel applications are borrowed from the Dense Matrix dwarfs and the dwarfs used are Matrix-Matrix multiplication, Jacobi Iteration and Laplace Heat Distribution. The experimental method consists of the selection of three data parallel bench marks and developed under the four shared memory based parallel programming models considered for the evaluation. Also the performance of those applications under each programming model is noted and at last the results are used to analytically compare the parallel programming models.

Results for the study show that by sacrificing the development time a better performance is achieved for the chosen data parallel applications developed in Pthreads. On the other hand sacrificing a little performance data parallel applications are extremely easy to develop in task based parallel programming models. The directive models are moderate from both the perspectives and are rated in between the tasking models and threading models.

INTERNATIONAL RESEARCH JOURNAL IN ADVANCED ENGINEERING AND TECHNOLOGY (IRJAET) E - ISSN: 2454-4752 P - ISSN : 2454-4744
VOL 3 ISSUE 4 (2017) PAGES 2287 - 2291
RECEIVED : 25.07.2017 PUBLISHED : 21.08.2017
Aug 21, 2017

The OpenMP Application Program Interface (API) supports multi-platform shared-memory parallel programming in C/C++ and Fortran on all architectures, including Unix platforms and Windows NT platforms. Jointly defined by a group of major computer hardware and software vendors, OpenMP is a portable, scalable model that gives shared-memory parallel programmers a simple and flexible interface for developing parallel applications for platforms ranging from the desktop to the supercomputer. Our work creates a foundation and can be extended to address many other computationally intensive problems. Our experience can also help researchers in relevant areas in dealing with similar problems and in developing efficient parallel programs running on computer clusters.

2. RELATED WORK

In order to demonstrate the effectiveness of parallel processing for a problem on some platform, several concepts have been defined. These concepts will be used in later chapters to evaluate the effectiveness of parallel programs. These include speedup, which describes performance improvement in terms of time savings, efficiency, which considers both benefit and cost, and scalability, which represents how well an algorithm or piece of hardware performs as more processors are added. where S(n) is the speedup achieved with n processors, T(1) is the time required on a single processor, and T(n) is the time required on N processors.

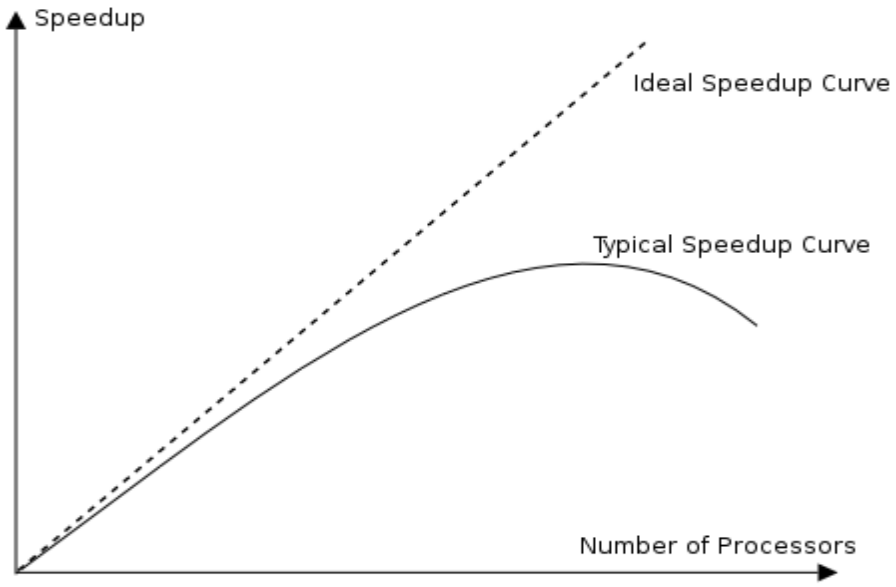

Fig.1. Typical Speed Curve

The discrepancies arise as to how the timings should be measured, and what algorithms to be used for different numbers of processors. A widely accepted method is to use optimal algorithms for any number of processors. However, in reality, optimal algorithm is hard to implement; even if it is implemented, the implementation may not perform optimally because of other machine-dependent and realistic factors, such as cache efficiency inside CPU. The concept of scalability cannot be computed but evaluated. A parallel system is said to be scalable when the algorithm and/or the hardware can easily incorporate and take advantage of more processors. This term is viewed as nebulous , since it depends on the target problem, algorithm applied, hardware, current system load, and numerous other factors. Generally, programs and hardware are said to be

scalable when they can take advantage of hundreds or even thousands of processors. But connection between any pair of nodes is not dedicated but shared: interconnection is implemented via a shared bus. This reduces the complexity significantly. In fact, its complexity is similar to that of line and ring topology. However, the dynamic characteristics, such as data transfer speed, are more inferior to those of fully-connected counterpart. Although collective communication is now very easy to implement, this single shared bus prevents more than one pair of nodes to carry out point-to-point communication. As a result, the system does not scale very well.

3. PROBLEM DEFINITION

The main intention of the research study is to compare the shared memory based parallel programming models. The comparison of the models is based on the development time of an application implemented under a specific programming model and the speedup achieved by an application implemented in a programming model. The purpose of the study is to answer the problems faced by the parallel programming community in the form of the research. This study compares the models based on the speedup and development time. The use of these two independent metrics for the comparison is because of the challenges faced by the organizations due to the conflicting requirements speedup and development time while developing the applications. Here are few key challenges faced by the HPC industry regarding the development time and speedup. All the metrics for the HPC are based on the estimations of certain factors, there is very less evidence and do not hold much in the HPC community. Development of parallel application software is constrained by the time and effort . It is necessary to empirically study the trade-offs associated with the development time of parallel applications. Requirements often need sophisticated empirical models to be defined for calculating the development metrics. The main intention of a better programming model is to support better performance on the other hand assist easy development of the reliable and scalable applications.

4. RESEARCH METHODOLOGY

The research approach used is a quantitative study and the process starts with the collection of data about the shared memory based parallel programming models that are considered for the evaluation of the thesis. For this research first a theoretical study is done on the programming models and the dwarfs selected for the thesis. Later an empirical study will be conducted for developing the dwarfs selected on each parallel programming model and calculate practically the factors development time and speedup for an application. The results obtained from the experiments are used to analyze and compare the parallel programming models. Answering the above research questions involves conducting quantitative research procedures in the concerned field of study. In order to understand how the application‘s development time and performance are affected by a specific parallel programming model a series of controlled human-subject experiments are conducted. The experiments are conducted for four different parallel programming models and for three different dwarfs. These experiments are used to collect data about the factors involved in this study and to later evaluate and compare those factors. This research study involves answering the research questions drawn from the problem statement. All the research questions are drawn with shared memory based parallel programming models into consideration and focuses on comparing the

INTERNATIONAL RESEARCH JOURNAL IN ADVANCED ENGINEERING AND TECHNOLOGY (IRJAET) E - ISSN: 2454-4752 P - ISSN : 2454-4744
VOL 3 ISSUE 4 (2017) PAGES 2287 - 2291
RECEIVED : 25.07.2017 PUBLISHED : 21.08.2017
Aug 21, 2017

directive based models, thread based models and task based models. For answering the research question.

5. ANALYSIS

A frame work is designed that describes a procedure to test the models empirically to validate the models against the research questions. Table 5 presents the frame work designed for this chapter. This frame work is used for the family of studies and in this case is used for a shared memory based parallel programming models. The models used in this study are Pthreads, OpenMP, TBB and Cilk++. The experimental methodology involves implementing the parallel computing dwarfs on these models and a data collection is done on these models for calculating the speedup and development time. The collected data is then analyzed and validated against the research questions. The results thus obtained are used to compare the parallel programming models.

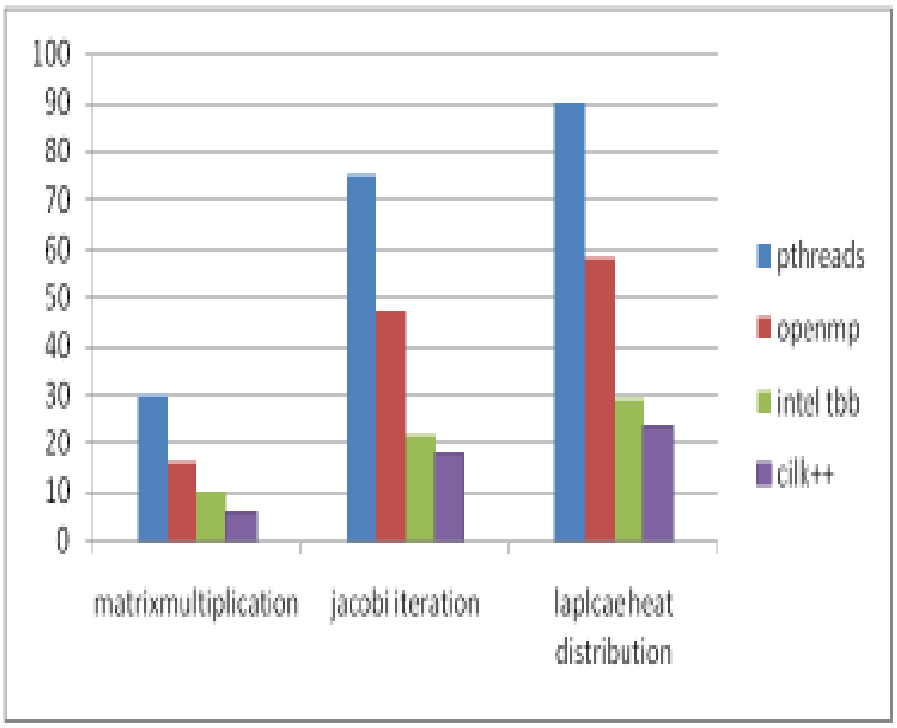

Fig.2. Development Model

For the implementation of the matrix multiplication under OpenMP the same type of the partitioning method is employed but the difference is that work is allocated among the threads done by the compiler. The developer only has to specify the size of the partition. The time taken for the Worker creation and management is approximately 37% of the total development time for the matrix multiplication algorithm and the time taken to develop the partition activity for allotting equal workload among the threads is approximately 63% of the total amount of time spent in developing the application.

CONCLUSION

It is clear that threading model Pthreads model is identified as a dominant programming model by supporting high speedups for two of the three different dwarfs but on the other hand the tasking models are dominant in the development time and reducing the number of errors by supporting high growth in speedup for the applications without any communication and less growth in self-relative speedup for the applications involving communications. The degrade of the performance by the tasking models for the problems based on communications is because task based models are designed and bounded to execute the tasks in parallel without out any interruptions or preemptions during their computations. Introducing the communications violates the purpose and there by resulting in less performance.

REFERENCES

[1] Alan H. Karp and Horace P. Flatt. ―Measuring Parallel Processor Performance‖, Communications of the ACM, vol. 33, Issue 5, pp. 539-543, ACM, New York, NY, USA, may, 1990.

[2] A. L.Wolf and D. S. Rosenblum.‖ A study in software process data capture and analysis‖, In Proceedings of the Second International Conference on Software Process, vol. 24 , Issue 8, pp. 115–124, February 1993.

[3] Ananth Grama, Anshul Gupta, George Karypis, and Vipin Kumar, ‖Introduction to Parallel Computing‖, Wesley Publishers Date: January 16, 2003.

[4] Andrew Funk, John R. Gilbert, David Mizell and Viral Shah. "Modeling Programmer Work flows with Timed Markov Models," CTWatch Quarterly, vol. 2, Number 4B, November, 2006.

[5] Angela C. Sodan. ‖ Message-Passing and Shared-Data Programming Models Wish vs. Reality‖, Proceedings of the 19th International Symposium on High Performance Computing Systems and Applications, pp.131-139, May 15-18, 2005.

[6] Barbara Chapman, Gabrielle Jost. ―Using OpenMP: Portable shared memory parallel programming‖, The MIT Press, Cambridge, Massachusetts, England, pp. 1-378, Oct- 2007.

[7] Barry Wilkinson and Michael Allen. ―Parallel programming", Pearson prentice-hall, 2005.