Parameter Trade-off And Performance Analysis of Multi-core Architecture

Surendra Kumar Shukla, CNS Murthy, and P. K. Chande

1 Introduction

Today advancement in hardware and software are going ahead hand in hand. The need of speed, in heterogeneous integrated system, is growing at a very high rate. Which has lead to the use of multiple core of CPUs on single chip. Now mobile and other applications are depending upon these multi-core architectures. But flip-side is that, applications are not capable in taking full advantage of this. This is because parallel computing has many challenges like parallelism, scalability, inter-core communication delay etc. [1, 2]. However in a multi-core system, these are certain aspects, which could be utilized/acted upon the further honers the capabilities of multi-cores. These aspects could be identified/ recognized as parameters to model the system and study their effects towards performance enhancement. There are some parameters which can play a key role for the performance improvement of multi-core architecture. Parameters like parallelism(hardware/software), scalability, need of process communication (API') etc. are some known parameters. Many researchers have provided some techniques for improving the multi-core system performance [3] [4]. But their techniques focus on individual parameter like parallelism in isolation [5]. It may improve the performance of the system in terms of throughput based on a single parameter. In this research work we intent to identify and use multiple parameters in an optimal way. We would also intent to see if the parameters could be adjusted(effecting function on the device) to suit the application nature, and analyze the performance.

S.K. Shukla (✉) • C. Murthy
Department of CSE, CDGI, Indore, M.P., India
e-mail: surendr.shukla@gmail.com; cnsmurthy@gmail.com

P.K. Chande
School of Computer Science & IT DAVV Univercity, Indore, M.P., India
e-mail: pkchandein@yahoo.co.in

2 Literature Review

Bryan schauer [3] has mentioned his work entitled "Multi core Processors- A Necessity", that one important aspect in increasing the performance of the multi-core architecture is the speed up. Speed up can be achieved by increasing the clock speed of the processor. Also, another established fact is by increasing the number of cores, speedup can be further enhanced. However, again, if we increase the number of cores then there are other inevitable problems with memory and cache coherence. And communication between cores also would have to be considered. Researchers have also focused on interconnection networks. But the interconnection network tend to get overloaded if parallelism exist at instruction level. It will create the overhead of communication between instructions, which is a function of inter-dependency.

Raghavan Raman [4] in his PhD thesis has focused - how compiler can support to obtain the parallelism in the program. Hence it is specified that the advent of multi core processors has lead to the emergence of different kinds of programming models to utilise the parallel processing power available. Parallel programming models exist in three categories. These categories are based on their approach to exploit the parallelism; (a) Programming language approach (b) directive based approach and (c) library approach. In (a) execute the task asynchronously, and for scheduling the task, we need to create threads, and, threads are expensive.

Damian A. Mallon [5] in his research paper specified the importance of APIs for the inter process communication. This paper specifies that the current trend of multi-core architectures emphasise the need of parallelism too. This research work evaluated the performance on multi-core between unified parallel c (UPC) and openMP on multi-core architectures. Result showed that MPI is best choice for multi-core systems, with both hybrid shared and distributed memory, because it takes highest advantage of

H. Selvaraj et al. (eds.), *Progress in Systems Engineering: Proceedings of the Twenty-Third International Conference on Systems Engineering*, Advances in Intelligent Systems and Computing 1089,
DOI 10.1007/978-3-319-08422-0_59,

data locality. Here data locality is a key performance factor (parameter).

Dmitri Perelman [6], in his work "Exploiting Parallelism of Multi-Core Architectures", shows the importance of parallelism in multi core performance. He has specified that if threads(program parameter) can be synchronised efficiently, and data exchange between the threads is proper then we can exploit the parallelism. Thus the biggest issue of multi-core architecture in this scenario is synchronisation. He has introduced transitional memory to synchronise the multi-threaded program, but the drawback of this technique is that for the implementation of the scheme are requires to add a small auxiliary translational cache memory.

Kakoullie [7] in 2012 specified that hot spot is the main issue in the performance of multi-core architecture. Hot spot is a router which is responsible for the data exchange among the cores in multi-core processor. Researcher has focused on the issues, which are responsible for the creation of hot-spot in the multi-core systems. He has specified that if hot spot will be generated in inter-connection network, it will affect the performance of the multi-core architecture. Prediction of hot-spot in the network is a difficult task, because generation of hot-spot depends on the nature of the application. For the prevention of hot-spots, the technique described in his paper is based on the artificial neural network(ANN). ANN based technique predicts and avoids the hot spots in the network.

Tudor, B.M. and Young Meng Teo [8] in 2011 specified, that memory contention is a big issue in the performance of the multi-core architecture. They have specified that in a program, with large memory requirement, memory contention increased as 1000 %. Tests have conducted on 24 cores on an inter NUMA system. These tests have been done on SP.C program. Results shown in his paper are- if problem size is small then there is less cache miss, then when less main memory access, then less traffic and also busty traffic. If size of the problem is large, then there is more cache miss, more main memory access, and non-busty traffic. Model proposed in this paper has following limitations- accuracy is decreased for the programs, where less degree of contention is existing. Second limitation is accuracy is decreased for the programs, for low memory requirement. This model is best for the programs, where the memory requirement is higher.

Shahrivari, S & Sharifi, M, [9] specified that Task-Oriented Programming, can help in increasing the performance of the multi-core architecture. They have mentioned that programming languages available for the multi-core & distributed systems are not efficient. To increase the performance of the multi-core architectures, we should have to use new programming models like task-oriented programming. But in task-oriented programming for making the program we need to create threads, and creation of thread involves the system calls, which degrades the performance of the system. Drawback of task oriented programming is that in task oriented programming task and required data both are exists in remote computer and remote computer executes the task and returns it to the needy computer. If data is available far away from the task, it will put higher penalty on the performance of the multi-core architecture.

Bini et al., [10] in their research work specified the concept of virtualization for testing the performance of independent applications. In virtualization concept some part of the whole resource is assigned to a application. Application has the illusion that he is using whole resource. This concept is useful for the applications like mobile phone where resource is precious, so we need to provide the resources to the applications on the basis of their importance. For example there are some applications like display of mobile phone which has high demand of memory, so we must have to give the higher priority to this device when assigning resources like memory and CPU time. Virtualization (parameter) concept can be helpful on identification of those applications which have more requirement of resources.

Khan, et al., [11] in their research work "Improving Multi-Core Performance Using Mixed-Cell Cache Architecture" specified that cache memory can have two type of cells. Robust cell and non-robust cell. Robust cell required more energy because robust cells stores the data which are required write operation in cell. Non-robust cell stores data which are required reading operation in cell. Robust-cell required more energy(parameter) as compared to the non-robust cell.

Ubal, R et al., [12] have proposed a tool for the analysis of multi-core architecture performance. Tool has focused on the analysis of three major performance elements of multi-core architecture-processor cores, memory hierarchy, interconnection network. Tool has visualize graphically how cores executing threads, how many cores are ideal, how interconnection network is utilised by cores for inter-core communication. With the help of this tool we can identify the multi-core architecture performance parameters.

Durate, F, and Wong, S [13] specified the Accelerator scheme, with the help of this scheme data movement between main memory and cache memory can be increased, it can increase the performance of multi-core architecture. Limitation of this scheme is that, it can improve the performance of multi-core architecture only in the case when we are doing copy operation between the main memory and the cache memory. But this scheme is not applied when real time updation is done in main memory and the cache memory.

Magnus Broberg [14] has developed a tool, with the help of the tool a parallel program can be developed in sequential

machine. Tool provides the parallel processing environment in uni-processor. A programmer when writes their program in that virtual parallel environment, he feels that he is actually working in parallel processor. Tool has provided opportunity to make the program for multi-core architecture.

Julian Bui et al., [15] has specified in their research paper, that cache size, cache protocols, associate number etc. are all important parameters for performance, among which cache size is thought to be the most important parameter for performance. Cache size has the largest impact on cache performance and energy consumption.

The following parameters [16] throughput, execution time, energy, memory bandwidth, scalability, inter-core communication, CPU clock speed, number of cores and cache coherence are the important parameters for performance improvement of multi-core architecture.

N. Ramasubramanian et al., [17] has specified that cache memory plays a crucial role in deciding the performance of multi-core system. He has used M5sim tool for the cache memory parameters simulation with multi-core processor parameters like L1 and L2 cache size verses CPU frequency.

Ram prashad mohanty et al., [18] has specified that the evaluation of performance is dependent on internal network e.g. ring network and hybrid network. He has used the parameters, execution time and speed-up, to show the performance of ring network and hybrid network.

The above review had been summarised in the following: Some common identified parameters are listed below

Some common identified parameters are listed below [15–18].

In the literature review we have found following conclusions

[1]. All attempts address some isolated issues and some common issues
[2]. Little about programs and programming style.
[3]. No investigation for hybrid and integrated applications-like mobile, Internet and security.
[4]. No research on instruction types, other characteristics of process and their performance.
[5]. No research on issues of fault tolerance and real-time systems.
[6]. Nothing on load balancing.
[7]. Less research on inter-connection network.
[8]. Less on special support processors.
[9]. No special hardware.

With the above literature survey, we have come with following research objectives, given in below.

3 Scope & Objectives

We Visualise the multi-core processor architecture with above perspectives then

I. Identifying more effective and comprehensive hardware software system parameters.
II. Understanding correlation between parameters.
III. Exploring the scope of balancing the parameters and trade-off optimising the throughput.
IV. Exploring the possibility of on-line parameter optimisation through signalling mechanism.
V. Simulation and analysis which can visualise all parameter tuning operation.
VI. Obtaining the interface standard(protocol) between Intelligent parameter tuning device(IPTD) and multi-core architecture for the interrupts.
VII. Obtaining the performance issues when many parameters and many applications will affect/influence to each other, dependencies.

4 Description of Research Work

Based on the objectives, a research plan has been prepared. Performance of multi-core architecture in totality can be visualised as:

Performance of the multi-core architecture = relation(System parameters + program parameters + data parameters + environment parameters + instruction pattern)

Parameter: A parameter, in its common meaning, is a characteristic, feature, or a measurable factor that can help in defining effect on a particular system. measurable factor which helps to define a particular system.

(1) **System Parameters:** Parameters which exist in the components of the multi-core architecture, e.g. cache memory, main memory, cores in the processor, interconnection network.
(2) **Program Parameters:** parameters which exist in the program by its virtue. e.g. Execution time, parallelism.
(3) **Data Parameters:** parameters related to the principal of locality.
(4) **Environmental Parameters:** parameters which the surroundings of a physical system that may interact with the system. e.g. Energy, temperature, radiations.
(5) **Instruction Pattern:** it means for the execution of instruction, how much data is required. With the help of

instruction pattern information, we can mix the instructions, CPU bound instructions, I/O bound instructions. We can perform data fetching operation for the instructions which has required more data.

4.1 Proposed Scope of Performance Improvement

Temperature balancing in Multi-core Architecture scope for performance improvement: we are proposing a novel approach here for the performance improvement of multi-core architecture, and extra core for monitoring the temperature on cores of multi-core processor. If in any core it was found that the temperature in the core is high because large number of instructions are executing in the core. Load should be automatically balanced, with the core which has less temperature. So some instructions should be transferred to the core which has less temperature. In the figure 4.1 given below it is shown that, because temperature in core 1 is high so some instructions must have to be transferred to the core 4. For monitoring the temperature a special device (extra core) should be used which is responsible to watch continuously the temperature of all the cores existing in the multi-core processor.

Hybrid cores and applications scope for performance improvement/ balancing. We are proposing here RISC + CISC instruction set architecture both should be existing in the core. CISC and RISC both should be used in the dynamic form. Core must have capability to change their mode from CISC instruction set architecture to RISC instruction set architecture as per the characteristics of the application. If application contains instructions which if could be executed in RISC architecture instead of CISC, then core should have to change their mode from CISC to RISC. In this approach we are suggesting that core functionality can be adjusted depending upon load characteristic. If we will implement this technique, performance of the multi-core architecture will be improved. In the diagram 4.2 it is shown that integrated application can be scheduled from CISC mode to RISC mode and from RISC mode to CISC mode.

Common shareable hardware and impact on performance. We are proposing shareable hardware which stores mixed instructions. we are using slack time, (a cycle that can be used for some other operation by the CPU, as instruction is not required that cycle for their execution may be that instruction is doing I/O activity that time) can be used to bring data in advanced for the other instruction stored in the instruction cache. The multi-core processor can utilize this concept for improving the performance. Here slack-time may be a parameter.

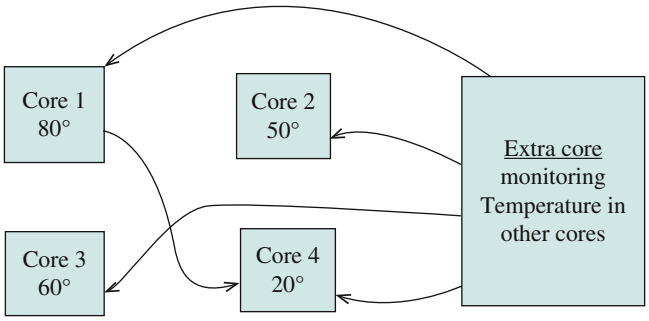

Fig. 1 Proposed approach for temperature control in Multi-core architecture

Intelligent Parameter Tuning device for performance improvement of Multi-core architecture. As our objective of research is to improve the performance of the multi-core architecture, for that we have proposed an intelligent device, IPTD(Intelligent Parameter tuning device). IPTD will act as supporting interface for the multi-core architecture for the performance improvement. As depicted in figure 4.1 any application will be given as a input to the multi-core architecture, machine will execute the application and simultaneously IPTD will watch the whole process of execution. If IPTD finds that any parameter is trying to degrade the performance, it will tune it with the help of the parametric interrupt to the multi-core architecture.We can test the performance of one application, two applications or n applications. We can analyse the performance of multi-core architecture one application to one parameter, one application to many parameter, many application to many parameters and many applications to one parameter. Intelligent parameter Tuning device is a firmware, we need to embedded the parameters in the chip. IPTD device has also a capability to generate the interrupt to the multi-core machine on the basis of parameters. Here parameters are

Functioning of IPTD and multi-core architecture. As shown in the figure 4.3 an application is used as a input to the multi architecture machine. Multi-core architecture machine executes the application. IPTD starts keen observation of the execution of application, with all their parameters and stores the throughput of the machine. As IPTD founds that performance is degrading, it tries that if he can change the some parameter value and performance will again come in the same label. Below we have given two cases of parameters tuning.

Different cases for the parameters tuning.

CASE 1

If IPTD founds that hot spots have been created in the interconnection network, IPTD will change their parallelism parameter and it will stop other parallel parts of the program to be given to the multi-core machine. And when he see that

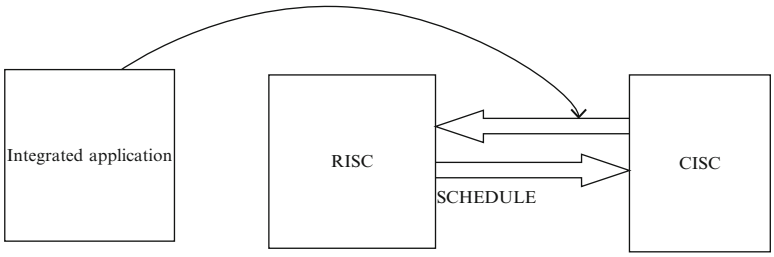

Fig. 2 Proposed Hybrid cores RISC & CISC

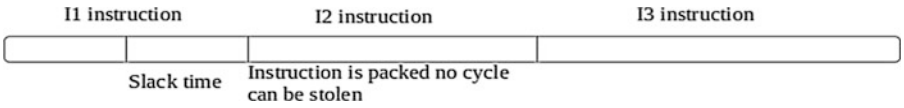

Fig. 3 Proposed Common shareable hardware

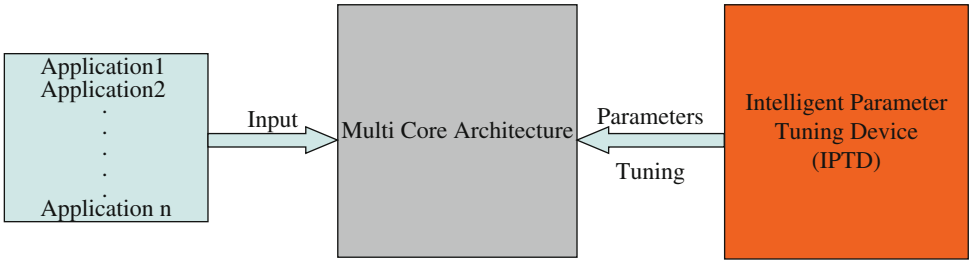

Fig. 4 Proposed Intelligent parameter tuning device

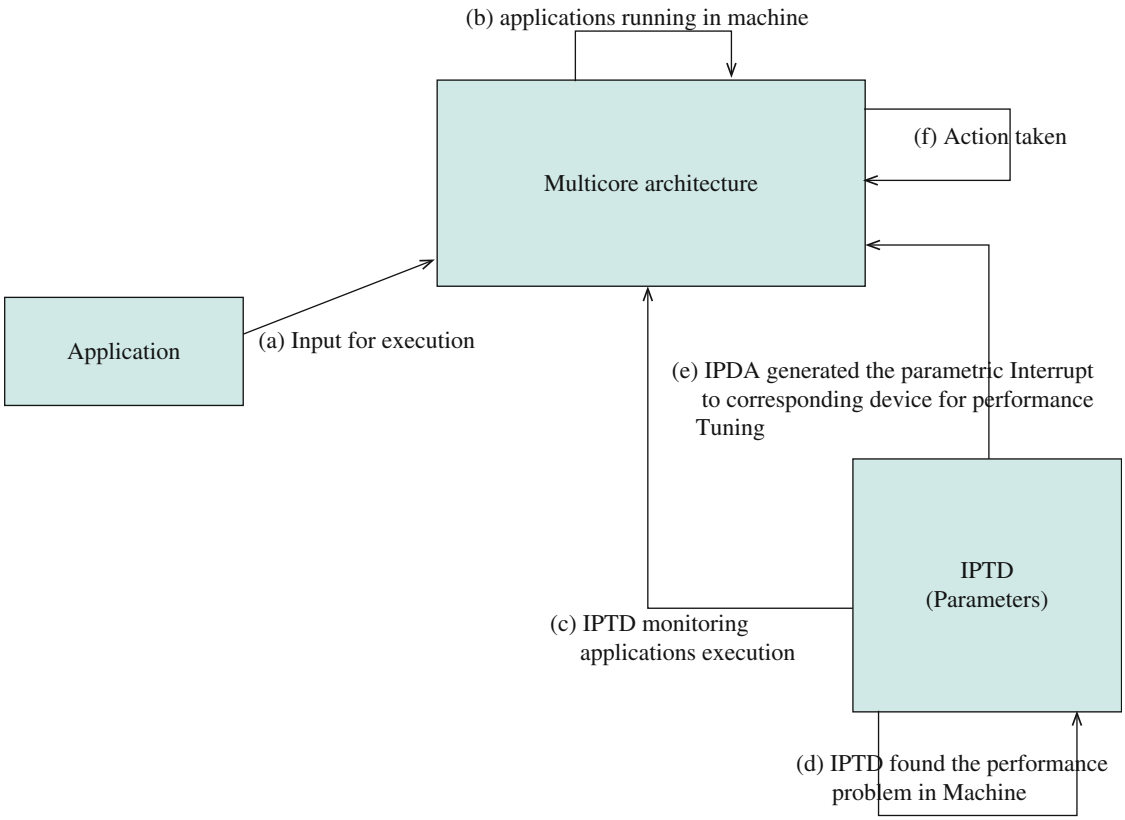

Fig. 5 proposed working of Intelligent parameter tuning device(IPTD).

Table 1 Parameters Description & Correlation.

S. No	Parameter	Parameter description	Correlation with other parameters
1	Speed-up	In parallel computing, speedup refers to how much a parallel algorithm is faster than a corresponding sequential algorithm.	Parallelism
2	Parallelism	Scope of parallel instruction in program.	Speed-up
3	Data locality	Data locality is a, typical memory reference feature of regular programs.	Execution time
4	No of threads	A thread is a, single sequential flow of control within a program.	Inter-connection network
5	Percentage of memory contention	A situation in which, two different programs, or two parts of a program, try to read items in the same block of memory at the same time.	Memory band-width
6	Virtualization	Virtualization, means to create a virtual version of a device or resource, such as a server, storage device, network.	No. of threads
7	Percentage of robust-cell required	Robust-cell, is a cell in cache memory, which is requires more energy.	Energy
8	No of processor cores	Total small CPUs, exist in single chip.	Speed-up
9	Inter-connection Network.	Network used to connect processor cores.	Percentage of memory contention
10	Memory Accelerator	An approach by which, data movement between main memory and cache memory can be increased.	Cache access time
11.	Memory Hierarchy	A ranking of computer memory devices, with devices having the fastest access time at the top of the hierarchy, and devices with slower access times but larger capacity and lower cost at lower levels.	Execution time.

now performance is up to the mark, it will again increase the parallelism parameter.

CASE 2

Suppose that IPTD have observed that, because of memory contention performance is decreasing, he may instruct to the multi-core machine that lot of memory operations are going on so, stop for some time this. In this case again IPTD will have to tune the parallelism parameter.

CASE 3

Suppose that IPTD have observed that the interconnection network have higher traffic because of finer level of granularity, (instructions level parallelism), all the cores are trying to communicate to each other, because of that the performance is degrading. IPTD will tune the compiler and instruct to the compiler for finding the course grain parallelism, so that traffic in the interconnection network can be reduced. After some time when traffic is normal in the interconnection network IPTD will again tune the compiler for the finer grain parallelism (instruction level parallelism).

5 Conclusion

It is clear therefore, that performance of multi-core architecture can be improved with the help of parameters tuning (adjustment). In this research, we will identify the performance parameters, and analyse them for the performance improvement of multi-core architecture. Analysis of performance parametes can provide us various opturnaties on performance improvement in future. If the proposed IPTD will be implemented in future it will be a seprate harware device for performance improvement.

Acknowledgements The authors thanks to management of Chameli Devi Group of Institutions, Indore, M.P. India for providing excellent research enviourment in the college. A Special thanks to the Chairman **Shri Vinod Kumar Agrawal ji**, CDGI Indore, India for motivating to the faculties of the institutioon on research activities.

References

1. http://software.intel.com/en-us/blogs/2008/12/31/top-10-challenges-in-parallel-computing
2. Advanced Computer Architecture, parallelism, scalability, programmability, Kai hwang 2nd ed. McGraw-Hill, 01-Feb-2003
3. Bryan Schauer "Multi core Processors – A Necessity" Pro Quest Discovery Guides, September 2008
4. Raghavan Raman, "Compiler Support for Work-Stealing Parallel Runtime Systems", Ph.D. dissertation, Dept. Computer Science, Rice University, Houston, Texas, May 2009
5. Damian A. Mallon, et al., "Performance Evaluation of MPI, UPC and OpenMP on Multi-core Architectures" in under Project TIN2007-67537-C03-02, Spain.
6. Dmitri Perelman, "Exploiting Parallelism of Multi-Core Architectures", Ph.D dissertation, Department of Electrical Engineering, Israel Institute of technology, Haifa Israel September 2012.

7. Kakoullie, E. et al., "Intelligent Hot-spot Prediction for Network-on-Chip-Based Multi-core Systems" in computer aided Design of Integrated Circuits and Systems, IEEE Transactions on, vol.61 no.3, pp. 418 – 431, 2012.
8. Tudor, B.M. And Young Meng Teo, "Understanding Off-Chip Memory Contention of Parallel Programs in Multi-core Systems", IEEE Conference in Parallel Processing (ICCP), Taipei, Singapore, 2011, pp. 602 - 611
9. Shahrivari, S & Sharifi, M, "Task-Oriented Programming: A Suitable Programming Model for Multi-core and Distributed Systems", IEEE Conference in Parallel and Distributed Computing (ISPDC), Cluj Napoca, Iran, 2011, pp. 139 - 144.
10. Bini, E et al., Resource Management on Multi-core Systems: The ACTORS Approach, published in IEEE Conference in Micro, 2011, pp. 72-81.
11. Khan, et al. "Improving Multi-Core Performance Using Mixed-Cell Cache Architecture", in High Performance Computer Architecture (HPCA2013), IEEE 19th International Symposium, Shenzhen, China, 2013, pp. 119 - 130
12. Ubal, R et al. "Multi2Sim: A Simulation Framework to Evaluate Multi-core-Multithreaded Processors", in Computer Architecture and High Performance Computing,19th International Symposium, Rio Grande do Sul, 2007 pp. 62-68
13. Durate, F and Wong, S "Cache-Based Memory Copy Hardware Accelerator for Multi-core Systems", Published in Computers, IEEE Transactions, (Volume:59, issue:11), 2010, pp. 1494-1507.
14. Magnus Broberg, "Performance prediction and improvement techniques for parallel programs in multi processors", Ph.D dissertation, Department of software Eng. & computer SC., Blekinge institute of technology Sweden, 2002
15. Julian Bui et al.,"Understanding Performance Issues on both Single Core and Multi-core Architecture", ACM conference in Computer Organizatione'07, Charlottesvill, 2007
16. http://www.cse.wustl.edu/~jain/cse567-13/ftp/multicore
17. N. Ramasubramanian et al. "Performance of cache memory subsystems for multi-core architectures", in International Journal of Computer Science Engineering and Applications, (IJCSEA) on vol.1, no.5, October 2011
18. Mohanty et al. "Performance Evaluation of Multi-core Processors with Varied Interconnect Networks", in Advanced Computing Networking and Security (ADCONS), 2nd International Conference, Mangalore, India, 2013, pp. 7-11