LINPACK: Power- Performance Analysis of Multi-Core Processors using OpenMP

Vijayalakshmi Saravanan
VIT University, India

Mohan Radhakrishnan
HCL, Canada

Mukund Sankaran
SASTRA University, India

D.P Kothari
Formerly with I.I.T. Delhi,
Currently with Raisoni Group of Institutions, Nagpur

ABSTRACT
With the advent of multi-core technology, scientific and high performance computing research is becoming increasingly dependent upon efficient parallel programming techniques. LINPACK is a mathematical software package which used for solving linear equations. The purpose of this paper is to compare the sequential and parallel implementations of LINPACK and analyze the results concurrently with energy consumption. A major emphasis is given here to find an efficient parallel programming method on multi-core processors for performance and power gains based on the obtained execution time. We discuss the techniques and algorithms involved in achieving high performance by reducing execution time through OpenMP parallelization on multi-core. The results of multi-core performance are found to be encouraging.

General Terms

Multi-core Architecture, LINPACK and Parallelization**.**

Keywords
Multi-core, OpenMP, LINPACK, Performance-Power, Sequential and Parallel Equations.

1. INTRODUCTION
LINPACK is a mathematical software package for solving problems in linear algebra: mainly dense linear systems and most of the parallel applications involved with looping concepts and matrix problems. LINPACK is not an efficient method to solve these problems due to its algorithm and resulting software accesses to memory problems. It spends too much time in moving data instead of doing useful floating point operations. Therefore parallelization must be done by using efficient programming techniques. OpenMP is one such effective practice for reducing execution time and increasing the performance on parallel machines. It has been broadly used in parallelizing scientific programs.

As computer architects turn to multi-core CPU and GPUs, there arises a need in parallel programming to increase the performance. So far no efficient approach has yet developed for parallel software. However OpenMP has long been used for performance oriented parallel machines. OpenMP (open multi-processing) is an API introduced in 1997 to standardize programming extensions for shared memory on multi-core machines over Linux environment. It provides data coherency among the caches and main memory of the multi-core architecture. OpenMP exploits thread level parallelism on multi core architecture and thus reduces the communication cost.

The structure of the paper is as follows: Section II presents the related work in this field. Section III covers the overview of the proposed work and IV & V discusses the proposed methodology. Finally, a detailed discussion and analysis of simulation results on multi-core processors is presented in section VI and some conclusions based on the analysis are reported in section VII.

2. RELATED WORK
With the recent advancements in multi-core technology, the processor industry has underwent a major shift from sequential to parallel programming in order to obtain significant performance and power gains. This tremendous shift from sequential (single core) to multi-core has lead to evolutionary changes in linear algebra software from block algorithms of LAPACK [1], [2].

Multi-core offers explicit support for executing multiple threads in parallel and thus reduces the idle time. This has opened up new domains to extract parallelism. The traditional methods and benchmarks which are used are often inadequate for multi-core processors. This in turn leads us to find the best benchmarks for parallel machines. The parallel systems with distributed memory could be the key issue in multi-core processors. The systems with more number of linear equations need appropriate methods and benchmarks. To illustrate this issue with the parallel solution of Gaussian elimination method, Brent studied the issues of distributed memory systems for linear algebra computations [3], [4]. On the other hand there is a need to study the best algorithms on multi-core/parallel machines and the performance metrics of numerical algorithms [5], [6], [7], [8].The research on hybrid multi-core and GPU architectures are also emerging trends in the multi-core era [9]. In [2], the performance analysis of various computers using linear equations is reported.

The latest shift in multi-core leads to excessive power consumption due to the ever increasing clock frequencies. We need to have proper algorithm or hardware level support to keep all the cores busy in order to retain high performance levels. So far all the works discuss about the performance algorithms and their benchmarks. Our unique approach is to find the performance and power of various numbers of (N) linear equations on multi-core CPU's using OpenMP programming techniques.

3. OVERVIEW OF PROPOSED WORK
LINPACK is a package of mathematical software which is used to provide the solution of linear equations using Gaussian elimination with partial pivoting. LINPACK uses column based algorithms in order to increase the efficiency, which is determined by running a computer program that solves linear equations.

OpenMP (open multi-processing) is an API over Linux environment which provides data coherency among the caches and main memory of the multi-core architecture. It consists of a set of compiler directives, library routines and environment variables that direct run time behavior and enable multiprocessor programming in C or C++, FORTRAN. It currently supports platforms such as Windows, Linux and UNIX operating systems. OpenMP works on the basis of multi-threaded concepts. Therefore, it takes advantage of executing different threads in these processing elements to reduce the communication cost between the threads. Having multiple cores on a chip allows us to extract thread level parallelism in a program which provides the benefit of increase in performance of the single program. [10], [11].

In our proposed work, we compared the various number of LINPACK equations on sequential and parallel programs using OpenMP. We used OpenMP as the programming language for our analysis, due to its parallelization efficiency on multi-core processors. The schematic diagram of our proposed work is shown in Fig.1.

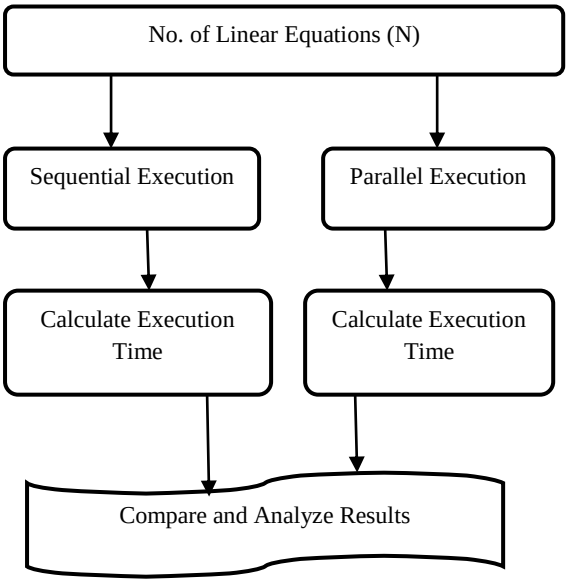

Fig. 1. Modules of Parallel Algorithm

3.1 System Specifications

3.1.1 Hardware Requirements

- **Processors**: Intel Core 2 & Intel Pentium M
- **CPU:** 2.13GHz & 1.60GHz
- **RAM:** 2 GB & 1 GB

3.1.2 Software Requirements

- **Operating System:** Windows XP/Vista
- **Software:** Visual Studio 2005
- Intel C++ compiler
- C-Free Compiler

4. PROPOSED METHODOLOGY

The methodology is as follows:

- The parallel algorithm, which is used to solve a non-singular n by n linear system using Gaussian elimination with partial pivoting, is developed. Gaussian elimination is equivalent to triangular factorization. It produces an upper triangular matrix and a lower triangular matrix.
- This parallel algorithm consists of forward elimination (F. Elimination) and back substitution (B. Substitution) phase. In forward elimination phase, first the Pivot element is identified as the largest absolute value among the coefficients in the first column. Then Exchange the first row with the row containing that element. Then eliminate the first variable in the second equation using normalization .When the second row becomes the pivot row, search for the coefficients in the second column from the second row to the nth row and locate the largest coefficient. Exchange the second row with the row containing the largest coefficient. Continue this procedure till (n-1) unknowns are eliminated.
- The backward substitution phase is concerned with the actual solution of the equations and uses the back substitution process on the reduced upper triangular system.

5. DESIGN MODULES

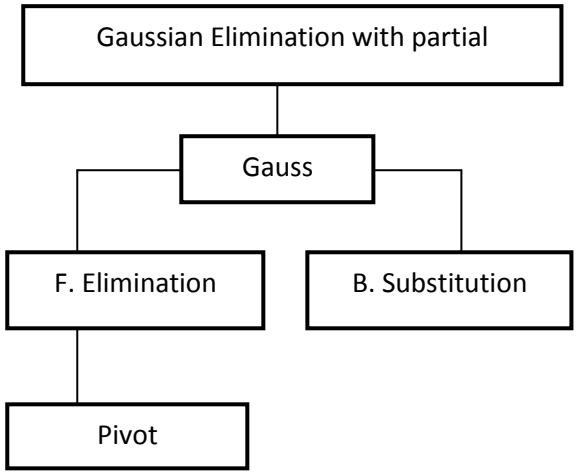

Fig. 2. Modules of Parallel Algorithm

5.1 Steps Involved in Gaussian Elimination with Partial Pivoting

It comprises of two phases are shown in Fig.2.

5.1.1 Forward Elimination Phase

This phase is concerned with the manipulation of equations in order to eliminate some unknowns from the equations and produce an upper triangular system.

- Search and locate the largest absolute value among the coefficients in the first column.
- Exchange the first row with the row containing that element.
- Then eliminate the first variable in the second equation using normalization.
- When the second row becomes the pivot row, search for the coefficients in the second column from the second row to the nth row and locate the largest coefficient. Exchange the second row with the row containing the large coefficient.
- Continue this procedure till (n-1) unknowns are eliminated.

5.1.2 Back Substitution Phase

This phase is concerned with the actual solution of the equations and uses the back substitution process on the reduced upper triangular system.

5.2 Implementation

- First study the typical behavior of LINPACK benchmark loops.
- From the behavior of the loops we derive a framework for manual parallelization.
- By using this framework, we can implement a specific code in parallel programming language (OpenMP).
- By applying OpenMP commands and threads to that particular program, we can derive a program which can be run in multi-core processors.
- We have implemented our program in OpenMP by studying several examples in OpenMP like Matrix Multiplication, Vector Multiplication, Hello World, Loop Sharing, Arithmetic Expressions etc.
- After running the program in multi-core processors, we found the increased performance in parallel execution of linear equations.

5.3 Performance Energy Calculation

Performance and power constraint are the important issues in high performance and scientific computing with multi-core. We have considered the dynamic power consumption of the single and dual-core processors in order to analyze the power and performance gains of LINPACK equations (based on the execution time taken).

We have taken the peak power consumption of Intel Pentium M and Intel Core 2 processor [12] and calculated the Energy consumption.

Intel Pentium M @ 1.6GHz: Peak power 24.5 W
Intel Core 2 @ 2.13GHz: Peak power 134 W

Let us define E be the total energy consumption and E.T be the total execution time of the given program. Ptotal is the total power, Pstatic is the static power and Pdynamic is the dynamic power consumed.

$$E = E.T * P_{total} \quad (1)$$

$$P_{total} = P_{dynamic} + P_{static} \quad (2)$$

We considered only the peak dynamic power consumption of the processors by taking the theoretical approximate values. By using (1) and (2), we calculated the energy consumption of the sequential and parallel execution as shown in Table.1 and Table.2 of various numbers of linear equations.

5.4 Tabulation of Performance & Energy Calculation of Linear Equations

Table 1. Tabulation of Performance Calculation of Linear Equations

Number of Equations (n)	Execution Time of Sequential (in seconds)	Execution Time of Parallel (in seconds)
1000	3	3
2000	23	26
3000	85.9998	67.9998
4000	216	157.00002
5000	397.99998	285
6000	679.0002	509.00004
7000	1044.996	780
8000	1156.9998	1098
9000	1620.996	1578
10000	2306.004	2124

Table 2. Tabulation of Energy Consumption of Linear Equations

Number of Equations (n)	Energy Consumption of Sequential (W)	Energy Consumption of Parallel (W)
1000	73.5	402
2000	637	2948
3000	2106.9951	9111.9732
4000	5292	21038.00268
5000	9750.99951	38190
6000	16635.5049	68206.0536
7000	25602.402	104520
8000	28346.4951	147132
9000	39714.402	211452
10000	56497.098	284616

5.5 Screenshots of Sequential and Parallel execution of LINPACK equation n=1000

The screenshots of both sequential and parallel execution with OpenMP LINPACK equations for n=1000 are shown in the following figures Fig. 3 and Fig. 4.

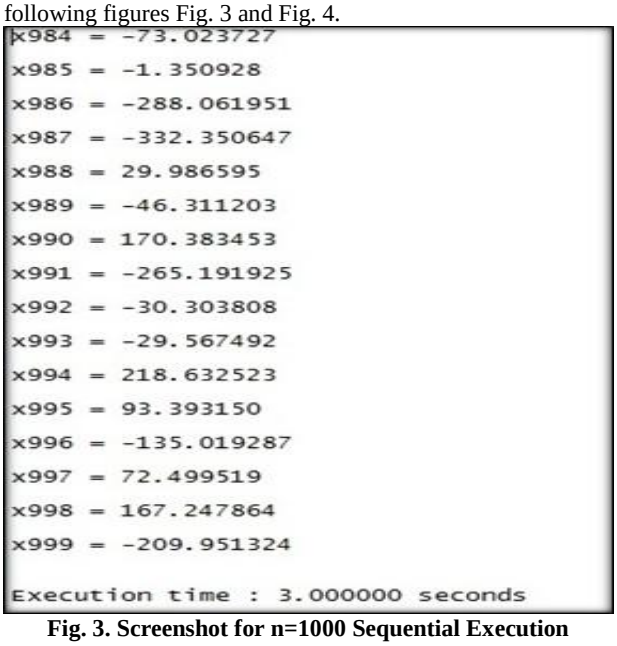

Fig. 3. Screenshot for n=1000 Sequential Execution

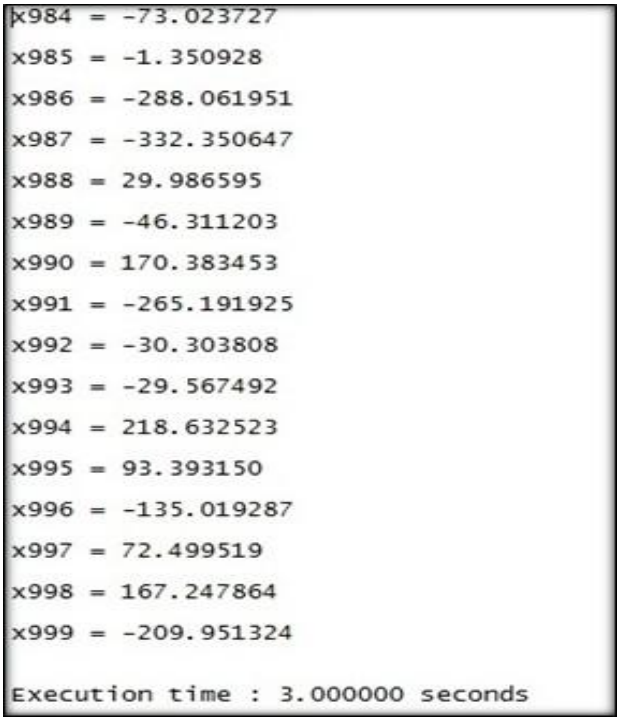

Fig. 4. Screenshot for n=1000 Parallel Execution using OpenMP

5.6 Screenshots of Sequential and Parallel execution of LINPACK equation n=2000

The screenshots of both sequential and parallel execution with OpenMP LINPACK equations for n=2000 are shown in the following figures Fig. 5 and Fig. 6.

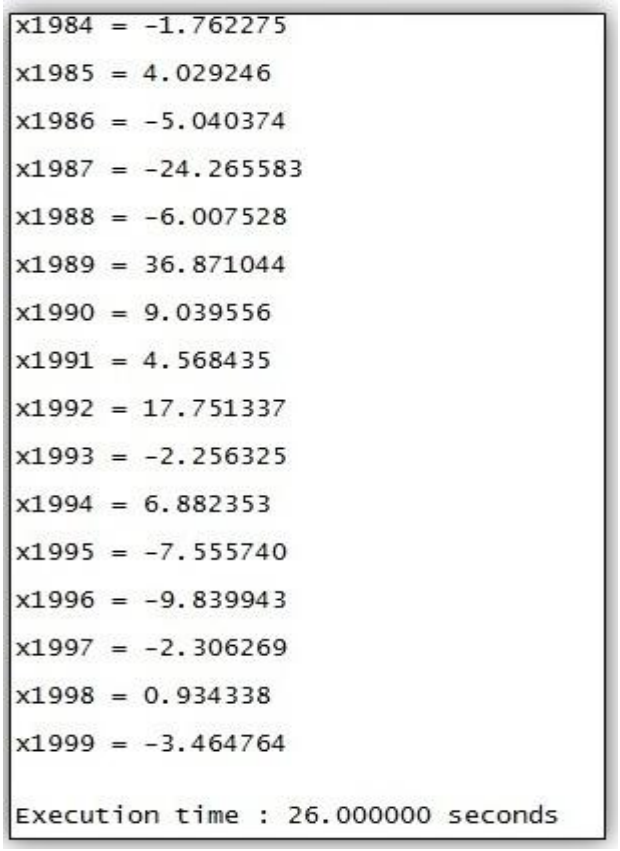

Fig. 5. Screenshot for n=2000 Sequential Execution

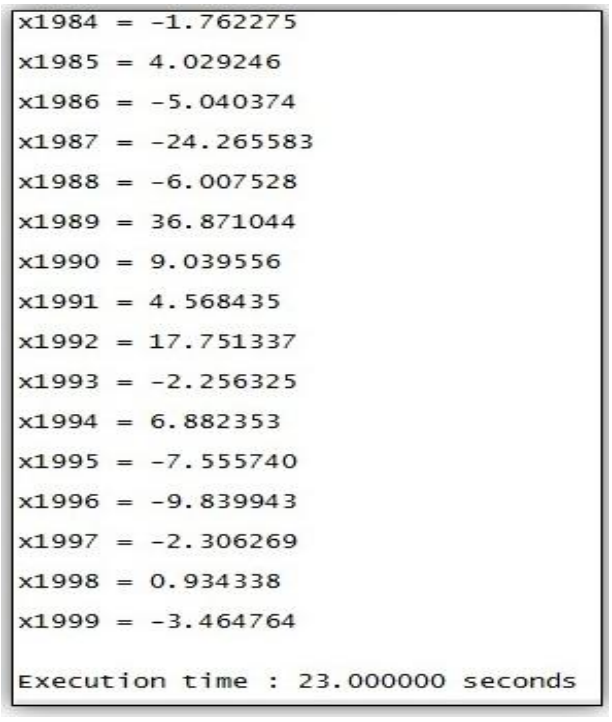

Fig. 4. Screenshot for n=2000 Parallel Execution using OpenMP

6. ANALYSIS AND DISCUSSION

The work has successfully solved linear equations using OpenMP on multi-core machines. We see the performance enhancement by reduced execution time of parallel execution and increased energy consumption due to its per core power consumption. From the Table I & II, we have shown the execution time of sequential and parallel implementations in which we calculated the energy consumption and performance of 'n' number of linear equations. The sample screen shots of both outputs are shown in Fig.3 - Fig.6 and the corresponding performance and power consumption results graphs are drawn in Fig.7 & 8.

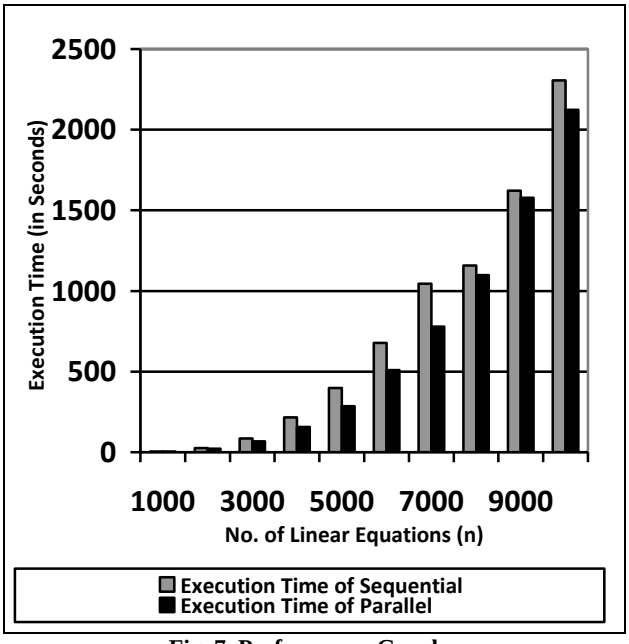

Fig. 7. Performance Graph

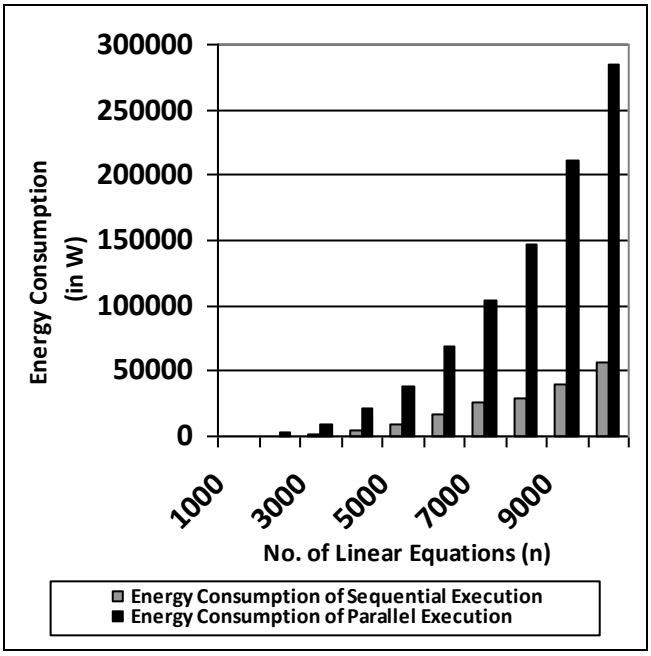

Fig. 8. Performance Graph

7. CONCLUSION AND FUTURE ENHANCEMENTS

In this work, we studied how OpenMP programming techniques are beneficial to multi-core architectures. We also solved linear equations using OpenMP to improve performance by reducing execution time. The future enhancement of this work is highly laudable as parallelization using OpenMP is gaining popularity these days. This work will be carried out in the near future for the real time implementation over a large scale.

8. ACKNOWLEDGMENTS

The authors would like to thank anonymous reviewers for their valuable comments and IASc (Indian Academy of Science, Bangalore).

9. REFERENCES

[1] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen, LAPACK Users' Guide. Philadelphia, PA: Society for Industrial and Applied Mathematics, third ed., 1999.

[2] J. J. Dongarra, "Performance of various computers using standard linear equations software (LINPACK benchmark report)," LINPACK Benchmark Report CS-89-85, University of Tennessee Computer Science Technical Report, 2011.

[3] R. P. Brent, "The LINPACK benchmark on the AP 1000:" in Proceedings of Frontiers '92 (McLean, Virginia, October 1992), pp. 128-135, IEEE Press, 1992.

[4] R. P. Brent, "Parallel algorithms in linear algebra, algorithms and architectures," in Proceedings of the Second NEC Research Symposium (held at Tsukuba, Japan, August 1991, (SIAM, Philadelphia), pp. 54-72, August 1993.

[5] K. A. Gallivan, R. J. Plemmons, and A. H. Sameh, "Parallel algorithms for dense linear algebra computations," SIAM Rev., vol. 32, pp. 54-135, March 1990.

[6] J. J. Dongarra, P. Luszczek, and A. Petitet, "The LINPACK benchmark: Past, present, and future. Concurrency and Computation: Practice and Experience", vol. 15, p. 2003, 2003.

[7] A. Buttari, J. Langou, J. Kurzak, and J. Dongarra, "A class of parallel tiled linear algebra algorithms for multicore architectures," tech. rep., 2007.

[8] K. A. Gallivan, W. Jalby, A. D. Malony, and H. A. G. Wijshoff, "Performance prediction for parallel numerical algorithms.," International Journal of High Speed Computing, vol. 3, no. 1, pp. 31-62, 1991.

[9] M. Horton, S. Tomov, and J. Dongarra, "A class of hybrid LAPACK algorithms for multi-core and GPU architectures," in Proceedings of the 2011 Symposium on Application Accelerators in High-Performance Computing, SAAHPC '11, (Washington, DC, USA), pp. 150-158, IEEE Computer Society, 2011.

[10] V. Packirisamy and H. Barathvajasankar, "OpenMP in multi-core architectures," 2005.

[11] K. Psarris, "Program analysis techniques for transforming programs for parallel execution," Parallel Computing, vol. 28, no. 3, pp. 455-469, 2002.

[12] Simcha Gochman et.al, "The Intel Pentium M processor: micro-architecture and performance," Intel Technology Journal, vol. 07, pp. 20-37, May 21 2003.

AUTHORS PROFILE

Vijayalakshmi Saravanan is an Assistant Professor (Sr), VIT University; India.She is a recipient of Erasmus Mundus (EURECA) Programme as an Exchange student from India at Malardalen University, Sweden. Currently, she holds a position as visiting researcher at Ryerson University, Canada. She holds a Bachelor of Engineering Degree in Electrical and Electronics Engineering and Master of Science Degree in Information Technology from Bharathiar University & Manonmaniam Sundaranar University (Now Anna University), India. Her research interests include Multi-core Low Power Design Exploration, Power-Aware Processor Design, and Computer Architecture. She has taken part of her research studies one course work at University of Rochester, USA. She is serving as a Technical Evangelist for Asia Open Source Software Community, CICC Japan. She is a Member of IEEE, ACM, CSI and a Board member of N2WOMEN (Networking Networking Women) IEEE/ACM Women in Engineer and she is a Chair for IEEE-WIE VIT affinity group, India

Mohan Radhakrishnan is currently working as a Sr.Technical Architect in HCL Canada. He has more than ten years of technical experience in designing, administrating and supporting Microsoft enterprise and VMware environments. He is currently working on R&D level projects in data center server and network implementation, support and administration, thorough grasp of development principles and best practices. He is also a Member of IEEE and VMware

Mukund Sankaran is currently pursuing a B.Tech in Computer Science and Engineering at SASTRA University, India. He was selected as a Summer Research Fellow by the Indian Academy of Sciences in the year 2011. His current research interests are in the areas of Computer Architecture and Database Systems

Dr. D.P. Kothari is a Senior Professor and Advisor to the Chancellor, VIT University, Vellore and named IEEE fellow in 2011. Earlier, he was Head, Centre for Energy Studies, IIT Delhi (1995-97), and Principal, Visvesvaraya Regional

Engineering College, Nagpur (1997- 98). He has been Director i/c, IIT Delhi (2005) and Deputy Director (Administration) (2003-06). Earlier, (1982-83 and 1989), he was a visiting fellow at RMIT, Melbourne, Australia. He obtained BE, ME and PhD degrees from BITS, Pilani. He is a Fellow of the Institution of Engineers (India), Fellow of National Academy of Engineering (FNAE), Fellow of National Academy of Sciences (FNASc), Life Member ISTE (LMISTE). Professor Kothari has published/presented 640 papers in national and international journals/conferences. He has authored/co-authored 22 books including Power System Optimization, Modern Power System Analysis, Electric Machines, Power System Transients, Theory and Problems of Electric Machines, Renewable Energy Sources and Emerging Technologies, and Power System Engineering. His research interests include Optimal Hydro-thermal Scheduling, Unit Commitment, Main-tenance Scheduling, Energy Conservation (loss minimization and voltage control), and Power Quality and Energy Systems Planning and Modeling. He has received the National Khosla award for Lifetime Achievements in Engineering for 2005 from IIT Roorkee. The University Grants Commission (UGC) has bestowed UGC National Swami Pranavananda Saraswati award for 2005 on Education for outstanding scholarly contribution. The World management congress, New Delhi conferred Life time achievement award for "Educational Planning and Administration on 30th December 2009.