Research of Electronic Power Information System Based-on Multi-core Processors

FanJun Meng[1], HaiLong Wang[1], JinHong Cheng[2] and Jiong Xie[2]

[1]Computer & Information Engineering College, Inner Mongolia Normal University
[2]Inner Mongolia Electric Power Information and Communication Center

mfj007.meng@gmail.com

Abstract

With the development of electric information technology, the performance of online production and marketing system has become a bottleneck in the development of the power system. The introduction of power system analysis computing to solve the problem of real-time information system quickly analyze theoretically. However, in the practical application of the system, the processing power of multi-core hardware does not fulfill, and has gradually become the performance bottleneck of the whole system. This paper considers the characteristics of the power system software, design and power under the multi-core environment information system performance testing and tuning system software by using the PMU multicore-based detection methods. This paper analysis and detects of online production and marketing system bottlenecks, identifies the main reason which downgrades the system performance, and generates the adjustment technology and plan.

Keywords: *multi-core; performance; Performance Monitoring Unit*

1. Introduction

With the constant expansion of information systems and the improvement of on-line analysis of the requirements, the traditional calculation method has been an unprecedented challenge. Parallel processing technology was gradually introduced to the field of power system analysis and calculation since the 1980s, and it was becoming a fundamental solution to the real-time information system quickly analyze problems. Whatever the trend calculation, stability calculation or safety analysis, solving algebraic equations or differential equations are the most important problem in on-line analysis. Parallel solution of these equations can be divided into two categories - direct methods and iterative methods. The direct method is built on mathematical principles on the basis of matrix decomposition, matrix inversion. Such method does not exist for the convergence problem, but short vectors characteristic of the front and rear dependent and network matrix calculation process is generally difficult to achieve a high degree of parallelism and the communication overhead is greatly. Traditional iterative methods such as Jacobi, SOR method can be directly parallelization method, which are simple, small space requirements and easy to effectively realize the advantages of vector calculations. However because of the the slow rate of convergence, the number of iterations is significantly increased with the number of processors, it may shorten the processing time for large-scale systems. On the other hand, a system can be simplified using equivalents simplified model. But over-simplified calculation accuracy will be declined and the result is always unacceptable because of the huge error.

With the development of information technology and the rapid increasing of information, as well as the rapid development and widespread of opening market, more efficient, more accurate mathematical model was introduced into the power information system. More and more researchers were trying to find a new way which is different from the traditional method for solving the answer in the on-line information systems. Due to the information system increasing scale, more and more factors should be considered into the traditional single computing environment, especially for the high-dimensional sparse equations and the large-scale information systems. Even if amount advanced model was introduced into the information system, the single core or the linear processing was still very time-consuming. In fact, the large-scale electric power information system requires high speed static and dynamic security analysis and calculation rapidly. Most of the calculating problem should be solved in a short period of time. The calculating result is the basic data for the decision of adjustments and precautions on power information system, which is a significant factor for improving the reliability of the running system. Traditional serial single processor model was unable to meet the requirement of online computing and real-time control of large-scale information systems. How to import the multicore parallel computing architecture system and parallel algorithms computing power information system have become urgent tasks in the Academic research area and industry area.

2. Design and Implementation

In this section, we present the challenges and goals on designing multi-core performance testing debugging programs. Then, we discuss the components of this mechanism in detail.

2.1. Design Challenges

In the design of a multi-core performance testing debugging programs, we must first make a trade-off in the following aspects.

2.1.1. What to Analysis?

There are many factors affect the performance of multi-core parallel program, this paper mainly analyzes the communication cost between multi-core parallel programming threads, and the occupancy of shared resources of cache, and detect whether a conflict exists. Cache is a key factor affecting program performance. With the increasing of the number of core in CPU, on the one hand, because of the limited capacity of Cache makes serious competition and conflict between the parallel programs which shared the same resources. On the other hand, shared Cache play a decisive role in collaborating between the parallel program communications (IPC mechanisms, such as shared variables, shared memory, mutexes, *etc.*).

2.1.2. What is the Analysis mechanism?

The analysis program can dynamically monitor program execution while the program running, it also can analyze the result and log files after the program execution. Furthermore, the analysis program can run in the software level and run in the hardware layer. Under the dynamic monitoring mechanism, the running cost of the former is much higher than the latter, and the accuracy is much lower. However, the former is easier to achieve because it does not increase the hardware complexity.

Taking into account of the debugging the performance to find the bottlenecks on auxiliary diagnostic procedures, accurate indicators of the performance data and smaller development complexity runtime overhead is relatively more important to the performance debugging mechanism. In this paper, the analysis program was running at the hardware level and dynamic monitoring procedures.

2.1.3. How to analyze?

Analyzing the logic complexity will directly affect the debugging overhead and performance. Under the dynamic monitoring mechanism, Analyzing program not only minimize the structure overhead introduced by the hardware extensions, and also control the system's runtime overhead. This paper mainly focuses on the cache structural and coherence protocol and controller logic extensions so as to monitor the run-time communication and conflicts between parallel programs, shared cache occupation. We will weigh the overhead data accuracy both by word and in two different monitoring modes in design the analysis model.

2.1.4. How to apply?

The ultimate goal of performance tuning is to achieve better execution performance in the online running system which should take into account the design of hardware and software structure. In this paper, we provide a software interface via Performance Counter to export the experiment results. Taking the Process and thread scheduling as an example, the designed performance counters can detect which processes and threads communicated more frequently. So we can schedule the thread to the adjacent core, which can greatly improve performance. Indeed, the performance monitor will inevitably introduce additional overhead to the system, our counters provide extended instruction control software layer in order to support the running monitor in order to minimize run time overhead.

2.2. Design Method

In this paper, we design a set of software to monitor the performance of on-line system. This system can analysis the main reasons which downgrade the performance. According to the testing result, we can adjust the basis system parameters to reach optimization purposes. There are two detection method based on the current popular multi-core processors: such as (1) the multi-core detection method based on monitoring and statistical cache behaviors; (2) PMU based multi-core detection method.

The multi-core detection method based on monitoring and statistical cache behaviors monitor and record the communication behaviors between different cores. It is monitoring the behavior include the following four categories: (l) read the shared resources between the cores; (2) read and write shared resources between the cores; (3) Conflict between the cores; (4) Occupancy of shared cache. PowerPerMon record the monitoring and statistics information to the performance counter. This result can be easily access in software level through the extended instruction set (ISA). The third part program can fetch this status information and do further research via our interface. The user can take the appropriate optimization strategy for different purpose.

PMU based multi-core detection method is using the integrated micro-system event performance counters in modern processors to make the online records in cases with minimal overhead and minimal interference to the target program. So the calculating the micro events characteristics becomes possible.

Based on PMU and Cache Coherence, we plan to implement the PowerPerMon performance tuning tools into Linux system, its features include: (1) to support the acquisition program running on specific processes and directly event by counting or sampling method with the hardware behavioral characteristics; (2) to collect event data and program code corresponding to the position; (3) to provide two different type interface, a command line or a graphical windows.

According to the current multi-core architecture of information system in Inner Mongolia Power Company, we choose the PMU based multi-core detection methods. We design and develop a set of software using PMU model to analyze and detect the bottleneck of the current on-line system, looking for main reason which downgrade system performance and generating the tuning report.

2.3. PMU-based Multi-Core Detection Software Architecture

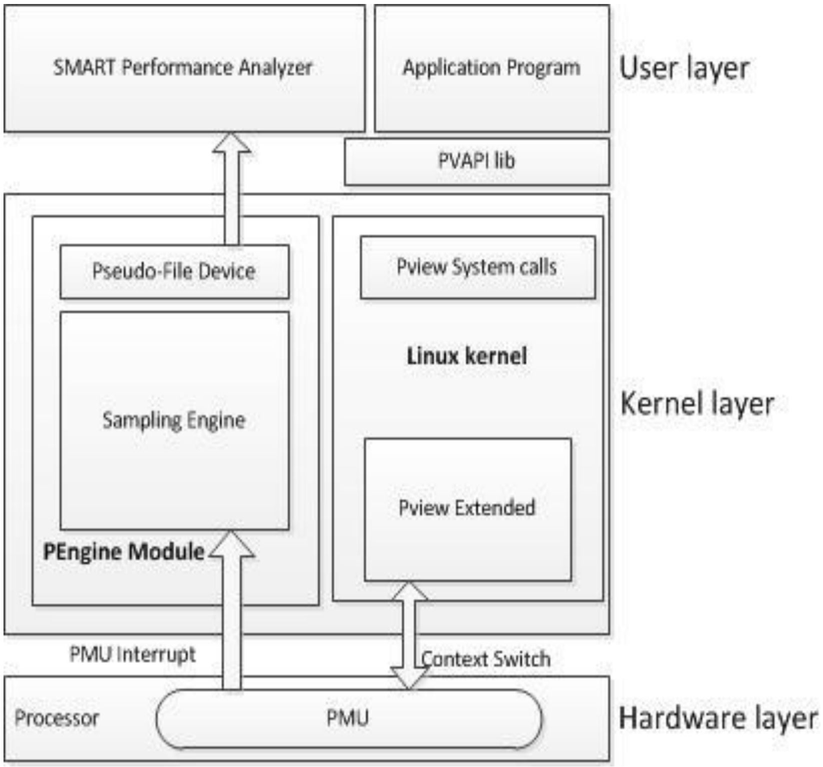

Figure 1. PMU-based Multi-core Detection Software Architecture

PowerPerMon tool set consists of two parts: the user layer in the front, the kernel layer in the rear. The front part includes the SMART Performance Analyzer. The rear part includes a PEngine module and the Pview system calls module in the Linux kernel. The user can communicate with Pseudo-device Device in the kernel layer via PVAPI library. PVAPI library packs the system call interface to get the processing information which shields the running details. The user can call PVAPI library function instead of directly use Pview system calls. The backend program is running in kernel mode which collects the hardware information and real-time processing data. This data is transmitted to the user layer via device interface or the corresponding system requests; the front part receives the data and then makes further analysis. The final result will be displayed and organize following the user’s request and presented with a friendly man-machine interface.

2.4. Steps of System Call Procedure

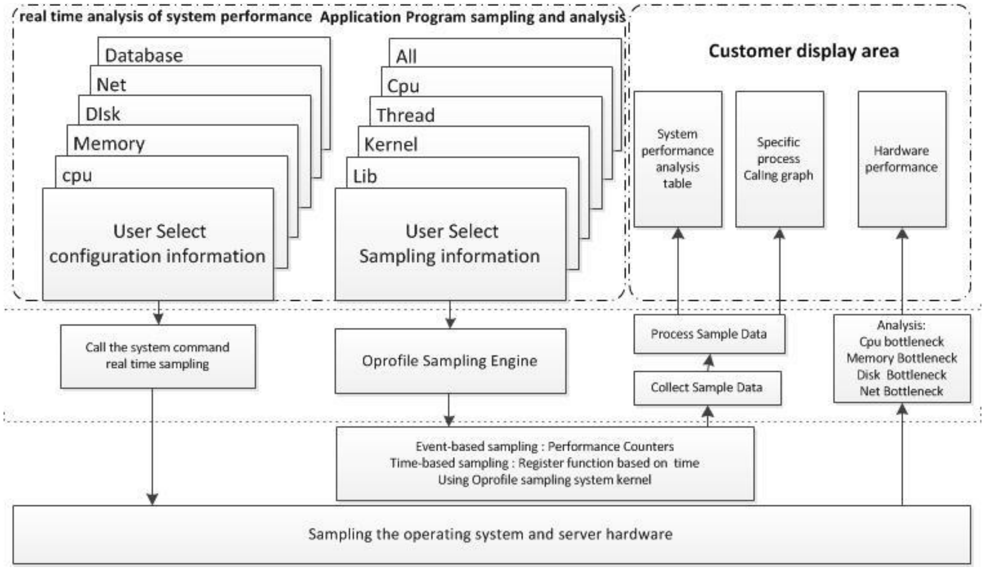
real time analysis of system performance
Application Program sampling and analysis
Database
Net
DIsk
Memory
cpu
User Select configuration information
All
Cpu
Thread
Kernel
Lib
User Select Sampling information
Customer display area
System performance analysis table
Specific process Caling graph
Hardware performance
Call the system command real time sampling
Oprofile Sampling Engine
Process Sample Data
Collect Sample Data
Analysis: Cpu bottleneck Memory Bottleneck Disk Bottleneck Net Bottleneck
Event-based sampling : Performance Counters
Time-based sampling : Register function based on time
Using Oprofile sampling system kernel
Sampling the operating system and server hardware

Figure 2. System Call Procedure

Figure 2 shows the system call procedure of our PowerPerMon. It includes three different parts: the presentation layer, application layer and kernel layer. Presentation layer includes customer configuration area and display area. The users can select the proper parameter, such as CPU memory size, network and database, to initialize this software and select the sampling data in configuration area. Application layer is running the system calls coming from the front interface. All the system call would be processed in the kernel level. Application layer fetch and then analyze the raw data, finding the bottleneck which downgrade the performance. The final result will be present on the display area in presentation layer.

The user selects the sampling configuration information to analyze the performance of the whole system. These configuration data include Lib, Kernel, Thread, CPU. In the application layer, PowerPerMon will automatically deploy Oprofile sampling engine according to the sampling configuration parameters set by the user. PowerPerMon will get the sampling data in the kernel layer. All the sampling data are then returned to the application layer data analysis and processing, and the final result will be presented in variety ways, such asthe table form , histograms graph, and process call graph.

2.5. The Characteristics and Advantages of Our PowerPerMon System

Compared with similar instruments, the strengths and characteristics of PowerPerMon include the following points.

The First, PowerPerMon supports both system-wide and direct counting and sampling methods of data collection for a specific process. Currently, in the application profiling and tuning system, system-wide sampling and process-specific performance event count are two of the most useful methods for data collection. We can also combine these two methods in one, such as using the performance event counts in overall process as the target analysis procedures rather than system-wide, to avoid randomness assumption fails sampling bias.

Second, PowerPerMon create a new process and load the target code to analysis the next program which is not running. In addition, PowerPerMon realize the attach and detach function to analyze the performance of running program in the specified periods.

Third, PowerPerMon provides performance analysis services via a unified interface -- pview system call. All control operations need a system call through this pview system call interface. These system calls can export specified process performance event statistics which the system running. In addition, the sampling process, data storage and transmission can be independently separated from the collected data by data collection engine.

Fourth, PowerPerMon fully supports all the multi-threaded program analysis in the kernel mode, it can record all the monitoring process performance statistics, and effectively manage a large number of dynamically created threads and revoked performance data. When the number of threads and activities performance events increased will not cause a substantial increase in the number of processor cores and record storage space as well as additional time overhead significantly increased.

3. Evaluation Performance

3.1. Experimental Environment

To evaluate the performance of the power information system, we run on a PowerPerMon . Table I summarizes the configuration of the cluster used as a test bed of the performance evaluation. Each computing node in the cluster is equipped with two dual-core 2.4 GHz Intel processors, 4GB main memory, 120GB SATA hard disk, and a Gigabit Ethernet network interface card.

Table I. Test-bed

CPU	Intel(R) Core(TM) i5-2430M CPU @ 2.40GHz(2401 MHz)
MainBoard	Dell
Memory	4.00 GB
Disk	WDC WD6400BPVT-75HXZT3
Graphics card	AMD Radeon HD 6630M

Experimental Result

Figure 7 shows PowerPerMon to demonstrate the impact of main memory on the performance of our power information system, we choose the main memory size of the computing nodes in our cluster to be 2 GB to 4 GB.

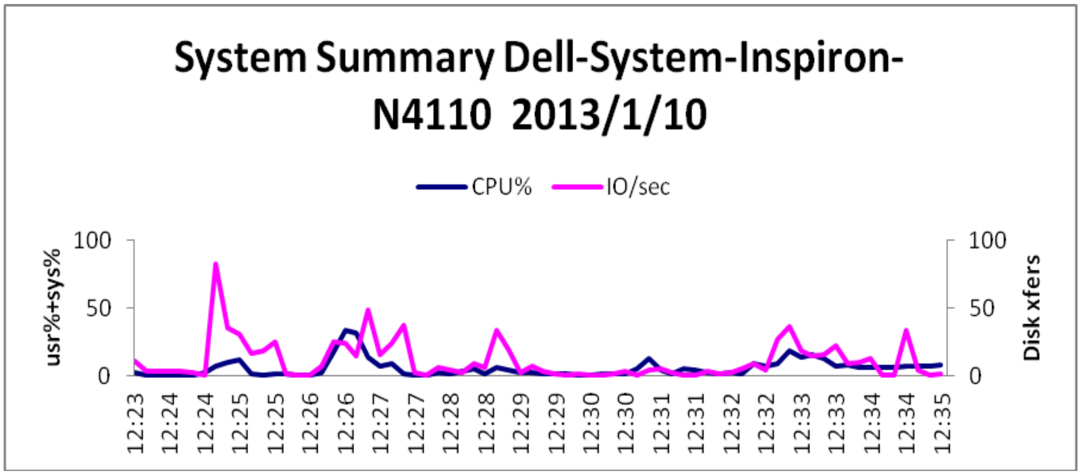

Figure 3. System Summary

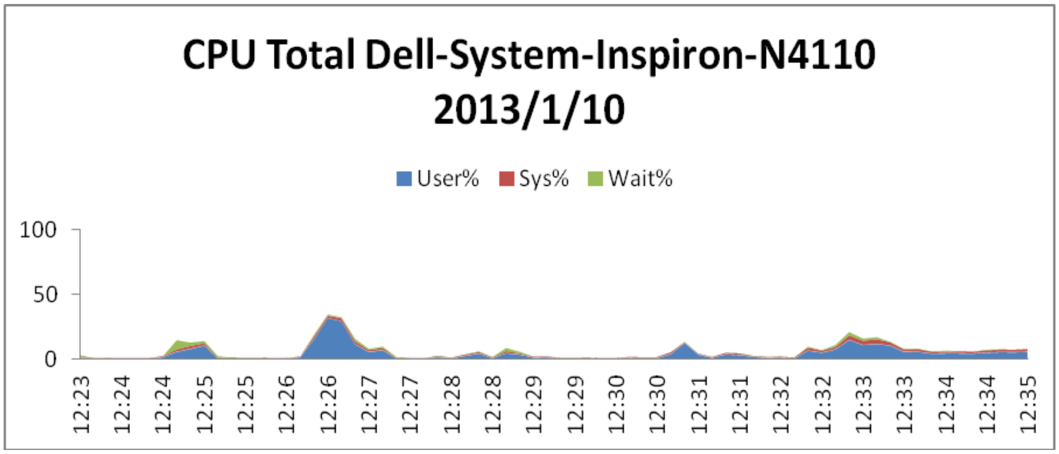

Figure 4. CPU Total

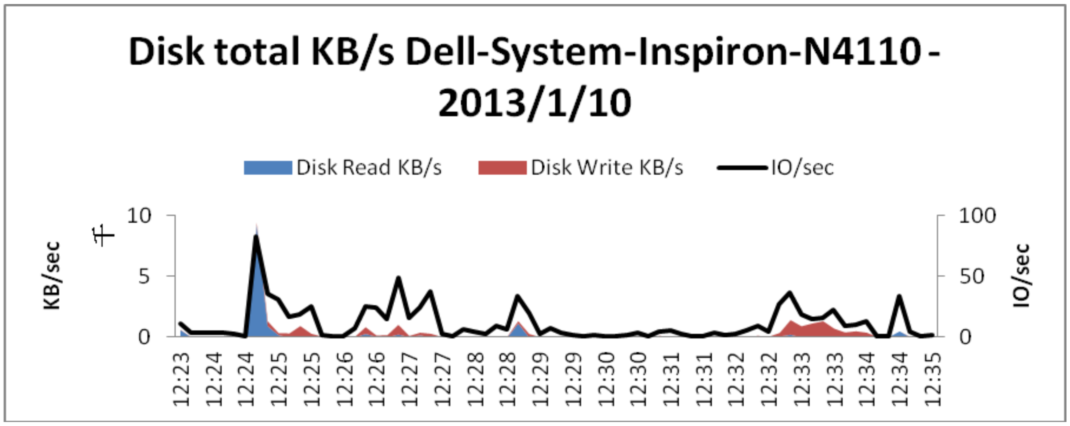

Figure 5. Disk Total KB/S

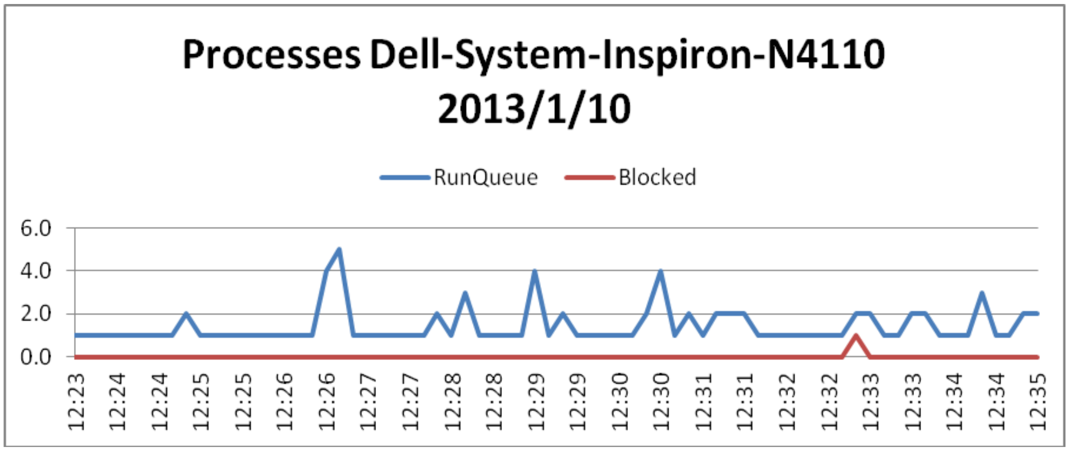

Figure 6. Processes

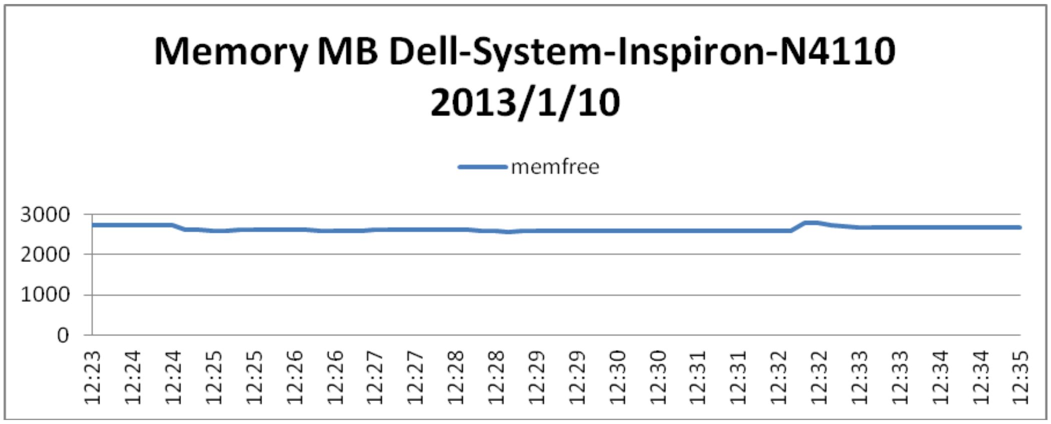

Figure 7. Memory MB

4. Conclusion

With the development of the Electronic power information technology, the overall performance of the multi-core software and equipment has become a bottleneck in the development of the power system. This paper considers the characteristics of the power system software, design and power under the multi-core environment information system performance testing and tuning system software by using the PMU multicore-based detection methods. This paper analysis and detects of online production and marketing system bottlenecks, identifies the main reason which downgrades the system performance, and generates the adjustment technology and plan. Our PowerPerMon can largely improve the performance of the existed system, and greatly improve the user experience.

Acknowledgements

This research was supported by Scientific Research Project of Higher Education of Inner Mongolia Autonomous Region, China (NJZY13052).

References

[1] IEEE Committee Report, Parallel processing in power system computation, IEEE Transactions on Power Systems, vol. 7, no. 2, **(1992)**, pp. 629-637.

[2] I. -R. Chen, “Effect Of Parallel Planning On System Reliability Of Real-Time Expert Systems”, IEEE Transactions On Reliability, vol. 46, no. 1, **(1997)**, pp. 211-218.

[3] A. Abur, “A Parallel Scheme for the Forward/Backward Substitutions in Solving Sparse Linear Equations”, IEEE Transaction on Power Systems, PWRS-3, vol. 4, no. 8, **(1988)**, pp. 119-127.

[4] F. L. Alvarado, D. C. Yu and R. Betancourt, “Partitioned Sparse A-1 Methods”, IEEE Transaction on Power Systems, vol. 5, no. 2, **(1990)**, pp. 218-235.

[5] J. E. Van Ness and G Molina, “Multiple Factoring in the Parallel Solution of Algebraic Equations”, EPRI EL-3893, **(1983)**.

[6] R. Betancourt and F. L. Alvarado, “Parallel Inversion of Sparse Matrices”, IEEE Transaction on Power Systems, vol. 3, no. 6, **(1986)**, pp. 426-433.

[7] M. K. Enns, W. F. Tinney and F. L. Alvarado, “Sparse Matrix Inverse Factors”, IEEE Transaction on Power Systems, vol. 5, no. 8, **(1990)**, pp. 311-324.

[8] D. J. Tylavsky and B. Gopalakrishnan, “Precedence Relationship Performance of an Indirect Matrix Solver”, IEEE/PES Summer Meeting, Long Beach, CA, **(1989)**.

[9] K. Lau, D. J. Tylavsky and A. B. Coarse, “Grain Scheduling in Parallel Triangular Factorization and Solution of Power System Matrices”, IEEE/PES Summer Meeting, Minneapolis, **(1990)** July.

[10] D. J. Tylavsky, “Quadrant Interlocking Factorization: A Form of Block L-U Factorization”, Proc. IEEE, vol. 3, no. 5, **(1986)**, pp. 254-263.

[11] A. Gomez and R. Betancourt, “Implementation of the Fast Decouple Load Flow on a Vector Computer”, IEEE/PES Winter Meeting, Atalanta, **(1990)**.

[12] J. E. Van Ness and D. J. Boratynska-Stadnicka, “A Partitioning Algorithm for Finding Eigenvalues and Eigenvectors”, Power Systems Computation Conference, Graz, Austria, **(1990)** August.

[13] D. J. Boratynska-Stadnicka, M. G. Lauby and J. E. Van Ness, “Converting and Existing Computer Code to A Hypercube Computer”, IEEE Power Industry Computer Applications Conference, Seattle, **(1989)** May.

[14] Y. Wang, “A New Parallel LU Decomposition Method”, IEEE Transaction on Power Systems, vol. 5, no. 9, **(1990)**, pp. 219-227.

[15] S. Y. Lee, H. D. ChiangK, G. Lee and B. Y. Ku, “Parallel Power System Transient Stability Analysis on Hypercube Multiprocessors”, IEEE Power Industry Computer Applications Conference, Seattle, **(1989)** May.

[16] M. LaScala, A. Bose, D. J. Tylavsky and J. S. Chai, “A Highly Parallel Method for Transient Stability Analysis”, IEEE Power Industry Computer Applications Conference, Seattle, **(1989)** May.

[17] F. M. Brasch, J. E. Van Ness and S. C. Kang, “Evaluation of Multiprocessor Algorithms for Transient Stability Problems”, EPRI-EL-947, 1987, vol. 20, no. 9, **(1996)**, pp. 22-25.

[18] Y. Jie, “Technology and Tool Design of Performance Tuning for Parallel Programming on Multi-core Platform”, University of Science and Technology of China, A thesis for master’s degree, **(2010)**.

[19] Y. Jie, X. Hengyang and An Hong, “Pview: A Novel Implementation of Fundamental Supports for Parallel Programs Performance Monitoring Based on PMU”, Computer Science, 2011, vol. 38, no. 2, pp. 288-292.

[20] Z. Xiaoyin, “Research debugging for multi-core architecture performance”, Fudan University, **(2009)**.

Author

Meng Fanjun received the BS and MS degrees in computer science from Inner Mongolia Normal University, China, in 1999 and 2007. Currently, his research interests include scheduling techniques and parallel algorithms for clusters, and also multi-core processors and software techniques for I/O-intensive applications.