Available online at www.sciencedirect.com

SciVerse ScienceDirect

Physics Procedia 33 (2012) 1029 – 1035

Physics Procedia

2012 International Conference on Medical Physics and Biomedical Engineering

Memory Performance Characterization of SPEC CPU2006 Benchmarks Using TSIM1

Fucen Zeng, Lin Qiao, Mingliang Liu, and Zhizhong Tang

Department of Computer Science and Technology,Tsinghua University,Beijing, China
Email: zengfucen@gmail.com, qiaolin@tsinghua.edu.cn, liuml07@gmail.com, tzz-dcs@tsinghua.edu.cn

Abstract

This paper uses TSIM, a cycle accurate architecture simulator, to characterize the memory performance of SPEC CPU2006 Benchmarks under CMP platform. The experiment covers 54 workloads with different input sets, and collects statistical information of instruction mixture and cache behaviors. By detecting the cyclical changes of MPKI, this paper clearly shows the memory performance phases of some SPEC CPU2006 programs. These performance data and analysis results can not only help program developers and architects understand the memory performance caused by system architecture better, but also guide them in software and system optimization.

Keywords:chip multi-processor; memory characterization; cache miss; performance; SPEC CPU2006 benchmarks

1.Introduction

There is a common belief that one of the key performance bottlenecks of system architecture is memory hierarchy. Architects and program developers have a growing need of understanding memory behaviors of workloads, such as the average miss rate of a specific program, the performance phases, the number of read/write instructions and the rate of correct prefetches, to optimize architectures or to develop high performance programs.

As far as architecture design is concerned, an architecture simulator is such an indispensable tool that it is, in fact, the foundation of quantitative analysis of architecture design, optimization and study. Simulation gives the architects a quick and accurate performance evaluation of a wide range of architectures, which

[1] This work is partially supported by the National High Technology Research and Development Program of China (863 Program) under Grant No. 2008AA01Z108, and National Science Foundation under Grand No.60773149.

 doi:10.1016/j.phpro.2012.05.169

certainly, reduce the cost and time of a project. For example, both of memory behavior characterization and exploratory research employ efficient simulation techniques in comparing the performance of one design policy with that of another.

Many studies on characterizing memory behaviors have been presented in recent years. Some researchers mainly use architecture simulation methods, while others prefer the performance monitor unit packaged in the operation system. Jaleel [1] has made a valuable research on memory system optimization for CMPs and workload characterization of SPEC 2000 CPU and SPEC 2006 CPU suites, using instrumentation-driven simulation (IDS). Li, et al. [2] have characterized performance of SPEC 2006 benchmarks both on the Intel platform and on the AMD one, using the performance events collected by the performance monitor unit. They have compared and analyzed performance differences caused by features of architectures and optimization technologies on the two platforms. Lin, et al. [3] has characterized memory behavior on emerging RMS (recognition, mining, and synthesis) workloads for future multi-core processors. They have also explored the LLC design space for multi-threaded RMS workloads by examining the working set size, data sharing behavior, and spatial data locality. The interested reader will also enjoy the correlative work in [4].

Recently, Bach, et al. [5] has used software instrumentation technique to efficiently analyze parallel programs. According to their work, developers can build tools to examine dynamic behaviors including data races, memory system behavior, and parallelizable loops, using Pin tool. Bienia [6], Bhadauria [7], et al. characterize the PARSEC benchmarks (Princeton Application Repository for Shared-Memory Computers). Their characterization shows that the benchmark suite covers a wide spectrum of working sets, locality, data sharing, synchronization and off-chip traffic.

Most of these studies usually give out many useful performance data, statistical charts of memory behaviors, and also several design suggestions for kinds of workloads or specific applications. Architects or software designers may take full use of these analyses. However, few high accurate performance data or statistical results for CMP platform have been reported.

TSIM (Tsinghua SIMulator) [8], as a trace-driven architecture simulator, uses the binary instrumentation technique as its front-end. It focuses on CMPs system and provides an extensible framework to explore the behaviors of on-chip memory subsystem. One of advantages of TSIM is that it is cycle accurate, which as a result, provides a detailed internal simulation and ensures the accuracy of data. TSIM presents an extensible approach to exploring behaviors of on-chip subsystems. A TSIM user can configure simulation parameters freely, such as cache level, cache size, block size, cache associability, cache resources, hit latency, replacement policy and coherence protocol, etc. Last but not the least, by introducing the concept of statistical meta metrics, TSIM separates the analysis stage from the simulation process per se, and this provides a great facilitation for a user to sample the performance metrics for further analysis.

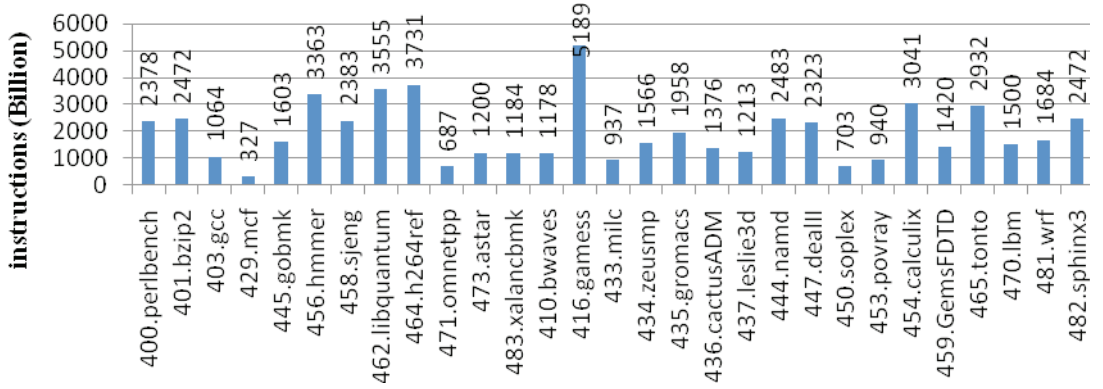

Figure 1. Dynamic Instruction Count of SPEC CPU 2006 Benchmarks

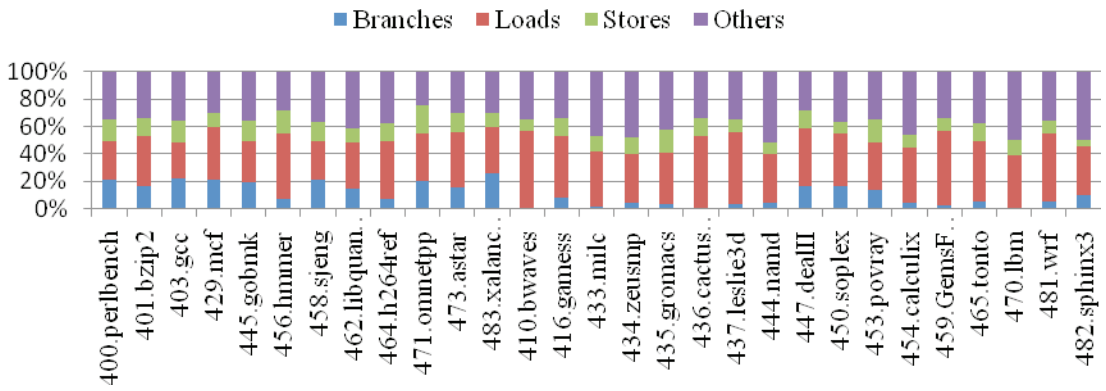

Figure 2. Instruction Mixture of SPEC CPU 2006 Benchmarks

2.Experimental Methodology

SPEC CPU 2006, released by Standard Performance Evaluation Corporation, is designed to provide performance measurements that can be used to compare compute-intensive workloads on different computer systems. SPEC CPU2006 contains two benchmark suites: CINT2006 for measuring and comparing compute-intensive integer performance, and CFP2006 for floating point performance.

This paper uses TSIM to characterize the on-chip memory behaviors under the CMP platform for the SPEC2006 CPU benchmark suite and the NPB-OMP suite (Nas Parallel Benchmarks, OpenMP version). In our experiments, more than fifty representative programs of kinds of application areas have been characterized.

Several important performance measures have been investigated: such as the instruction mixture, MPKI (Miss Per 1000 Instructions), and rate of correct prefetches of each SPEC CPU 2006 program. After that, some representative programs are randomly selected to detecting their performance phases. These results are useful for optimizing architectures and improving the performance of programs.

In order to make the analysis brief, the simulation has been set up to skip the first one billion instructions and then to run the following one billion ones.

The detailed experimental configuration of TSIM is listed in Table I.

TABLE I. THE EXPERIMENT CONFIGURATION SUMMARY

Configuration Name	Value
Cache Size of Simulator	32KB
Cache Block Size	64B
Cache Ports	4 ports for w/r
Number of MSHRs	4
Hit Latency	3 cycles
Load Latency	1 cycle
Way of Associativity	4-way
Replacement Policy	LRU

3.Memory Performance

The speed gap between processor and memory has become the most important factor influencing system performance. This section gives the characterization results from several aspects, including instruction mixture, cache behaviors, performance phases, and cache sharing behaviors.

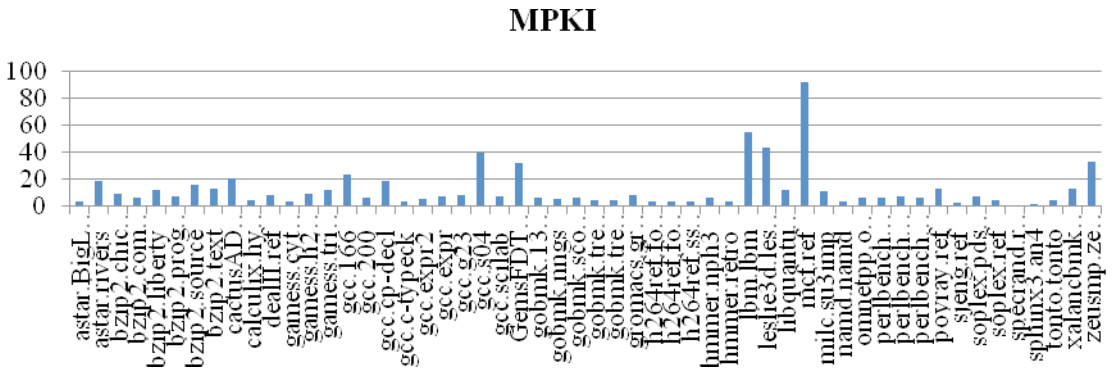

Figure 3. MPKI of SPEC CPU 2006 Programs

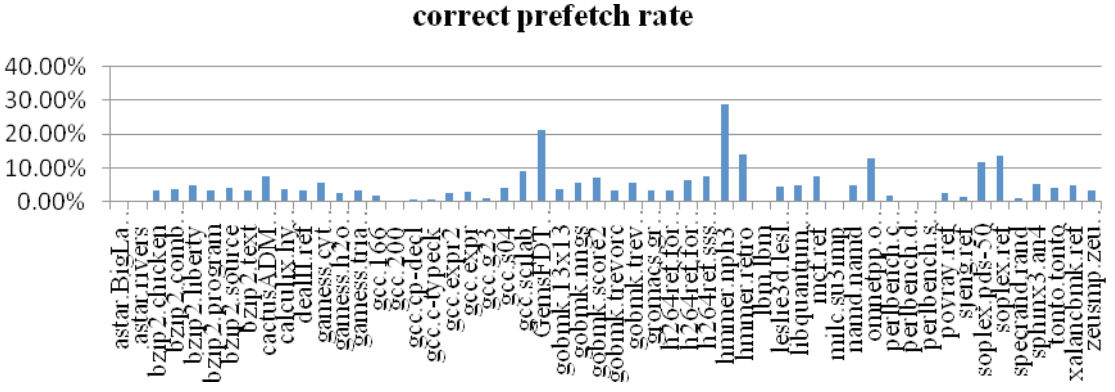

Figure 4. Correct prefetch rate of SPEC CPU 2006 programs using stride prefetch

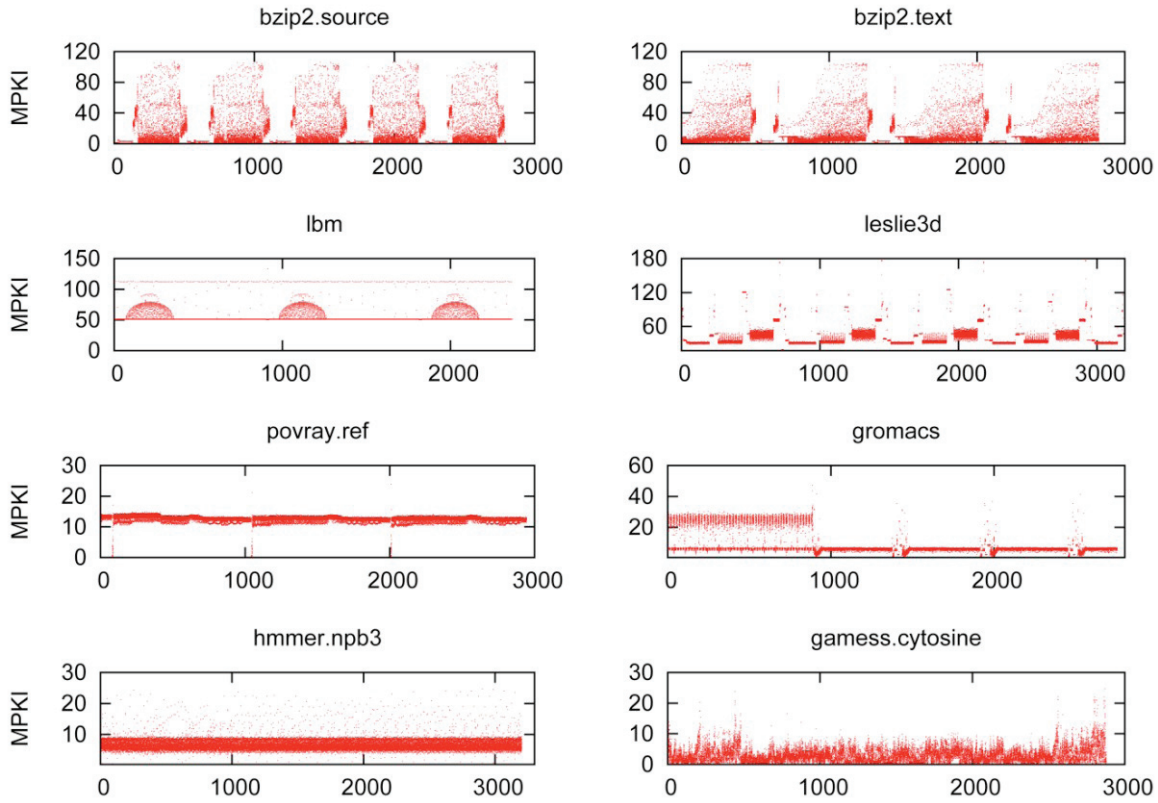

Figure 5. Detect the MPKI Phases of the selected eight representative SPEC CPU 2006 Benchmarks. The unit of x-axis is million cycles.

3.1.Instruction Mixture

Figure 1 shows the dynamic instruction count of each program in the SPEC CPU 2006 suite. And Figure 2 illustrates the instruction mixture of the programs, including branches, loads, stores and other instructions.

As shown in the figures above, the percentages of loads/stores instructions of all integer programs are between 41% and 65%, while the floating-point programs are between 40% and 64%. For almost every SPEC CPU integer program, the percentage of branch instructions is closed to 20%, except *456.hmmer* and *464.h264ref*. While for most of SPEC CPU floating-point programs, the percentages of their branch instructions are less than 10%, except *447.dealII*, *450.soplex* and *453.povray* that have nearly 15%. This implies that higher parallelism of these programs can be exploited, especially for those with large dynamic basic block sizes.

3.2.Cache Behaviors

The metric which reflects the cache performance best is misses per 1,000 instructions (MPKI). Miss rate of a cache represents its utilization, where MPKI statistically denotes the mean one of each 1,000 read/write instruction. Figure 3 shows the MPKI of SPEC CPU 2006 programs; the MPKI metrics of about ten SPEC CPU 2006 workloads are more than 20%, including *astar.rivers, cactusADM, gcc.166, gcc.cp-decl, gcc.s04,GemsFDT, lbm, leslie3d, mcf.ref*, and *zeusmp*.

In literature, cache prefetching technique has been being expected to improve cache performance greatly. But unfortunately, failed prefetch can also pollute caches, and as a result, cause performance degradation. Stride prefetch, as a typical prefetch algorithm, is mainly used to eliminate compulsory/capacity cache miss. Stride prefetch believes that if a memory address is missed, an address that is offset by a distance from the missed address is likely to be missed in near future. Figure 4 shows the correct prefetch rate of SPEC CPU 2006 programs when stride prefetch algorithm is used. As we can see, Stride prefetch mechanism is well performed on *gcc.scilab, GemeFDT, hmmer.npb3, hmmer.retro, omnetpp, soplex.pds-50*, and *soplex.ref*. The correct prefetching rates of all these programs are greater than 10%, which of course, brings a great performance improvement for cache MPKI.

3.3.Performance Phases of SPEC CPU 2006 suite

Figure 5 shows the MPKI changes of eight randomly selected representative benchmarks of SPEC CPU 2006, via the TSIM's sampling tool, where a clear panorama of the workloads' cache behaviors can be observed. The performance phases of each program have been shown by cyclical changes of MPKIs in Figure 5.

In the experiment above, the MPKIs of some SPEC CPU 2006 programs change cyclically, while others remain stable. For program *bzip2* with input sets *source, text*, a MPKI phase periodically appears nearly every 500 million cycles, and remains alive when other data sets are used. And in similar, program *lbm*'s performance phase appears every nearly 1,000 million cycles, *leslie3d*'s nearly every 800 million cycles, and *povray.ref*'s also nearly every 1,000 million cycles. Program *gromacs* is a interesting case; whose MPKI remains stable in the first 800 million cycles and periodically appears every 500 million cycles after that, while *hmmer.npb3*'s MPKI all the same. Contrary to others, program *gamess.cytosine* has no any regular changes.

4.Conclusion

This paper presents on-chip memory behavior characterization of SPEC CPU 2006 benchmarks under CMP platform using a cycle accurate simulator, TSIM. The instruction mixture, MPKI and the correct stride prefetch rate of SPEC 2006 programs are characterized. After that, cyclical MPKI changes are detected, which clearly shows the performance phases of the SPEC CPU 2006 programs. Our experimental results and corresponding analysis are valuable to help programmers understand the performance, caused by architectures, and optimize programs better. Architects can also benefit from this analysis when making architecture design.

References

[1] A. Jaleel, “Memory characterization of workloads using instrumentation-driven simulation - a pin-based memory characterization of the spec cpu2000 and spec cpu2006 benchmark suites,” VSSAD, Tech. Rep., 2007.

[2] S. Li, B. Cheng, X. Gao, L. Qiao, and Z. Tang, “Performance characterization of spec cpu2006 benchmarks on intel and amd platform,” in Proceedings of the 2009 First International Workshop on Education Technology and Computer Science, vol. 2, 2009.

[3] J. Lin, Y. Chen, W. Li, A. Jaleel, and Z. Tang, “Understanding the memory behavior of emerging multi-core workloads,” in Proceedings of Eighth International Symposium on Parallel and Distributed Computing, 2009.

[4] Y. Chen, W. Li, and J. M. Lin, “Memory characterization of emerging recognition-mining-synthesis workloads for multicore processors,” in Workshop for Computer Architecture Evaluation of Commerical Workloads (CAECW), co-located with HPCA’08, 2008.

[5] M. Bach, M. Charney, and R. Cohn, “Analyzing parallel programs with pin,” IEEE Computer, March 2010.

[6] C. Bienia, S. Kumar, J. P. Singh, and K. Li, "The parsec benchmark suite: Characterization and architectural implications," in Proceedings of the 17th International Conference on Parallel Architectures and Compilation Techniques, Oct. 2008.

[7] M. Bhadauria, V. Weaver, and S. A. McKee, "A characterization of the parsec benchmark suite for cmp design," Computer Systems Laboratory, Cornell University, Ithaca, NY 14853, Tech. Rep., 2008.

[8] M. Liu, L. Qiao, Y. Chen, F. Zeng, and C. Zhang, "An extensible memory simulation framework for chip multiprocessors," in Proceedings of the 2nd International Conference on Computer Science and Software Engineering (CSSE), Wuhan, China, Dec. 2009.