

Characterization of Web Server Workloads for Three Generations of IBM PowerPC Microarchitectures

Pattabi Seshadri
Dr. Lizy K. John

University of Texas at Austin

Department of Electrical and Computer Engineering (UT-ECE)

In Collaboration With: Alex Mericas, Processor Performance, Advanced CEC Engineering, IBM Austin Server Group

OBJECTIVES

Web server workloads have been characterized as having a treelike execution path and thus a higher branch misprediction ratio than other types of workloads such as are represented by SPECfp and SPECint, which typically contain more loops¹. Thus, it would stand to reason that speculative and out of order execution would be less effective for web server workloads than for other workloads that exhibit more predictable branching structures. The aim of this work is to discover whether current microarchitectural design features, speculative and out of order execution in particular, can be effective for these types of applications. To this end, three different types of web server workloads—a HTML page return, a CGI script, and a Java servlet—will be run on three generations of IBM PowerPC microarchitectures—the PowerPC 604e, the PowerPC RS64-II, and the POWERIII—in order to characterize these workloads on three very different microarchitectures that vary in complexity. (SPECWeb 99 may also be used, as it has been updated to include benchmarks that approximate the execution characteristics of dynamic web server workloads such as Java servlets and CGI scripts. Initial plans are to use the Apache Web Server [which includes Java servlet and CGI capabilities], with the inclusion of SPECWeb99 a possible later step.) Measurements will be made using the built-in hardware performance monitor on each machine. The goal of the project is to discover which microarchitectural techniques are useful/ necessary/ ineffective as regards the optimal performance of web server workloads. These findings will hopefully suggest possible directions for the design of future web server microprocessors.

EXPERIMENTAL SETUP

This section gives an overview of the three PowerPCs and the three web server workloads used in this study.

The PowerPC 604e² is a 32-bit, superscalar, out of order, speculative execution, in order completion machine. It has two single cycle integer units, one multiple cycle integer unit, one branch unit, one condition register unit, one load/ store unit, and one pipelined three stage floating point unit. It has a six stage pipeline. The PowerPC 604e can fetch, dispatch, and retire up to four instructions per cycle. For branch prediction, it uses a 64 entry BTAC and a 512 entry, two-bits-per-entry branch history table. It has a 32 KB, four way set associative L1 instruction cache, a 32 KB, four way set associative, non-blocking L1 data cache, and a 256 KB unified L2 cache. It also has two Block Array Translation Units, one for instructions and one for data, that provide quick address translation to large contiguous irregularly sized blocks of memory. The processor clock is 332 MHz.

1. Radhakrishnan and Rawson, 1998.
2. IBM Corporation and Motorola Corporation, 1998.

The PowerPC RS64-II¹ is a 64-bit, superscalar, in order, speculative execution machine and is targeted specifically for commercial applications such as web servers. It has one single cycle integer unit, one multiple cycle integer unit, one four stage pipelined floating point unit, one branch unit, and one load/ store unit. It has a five stage pipeline. The PowerPC RS64-II can fetch, dispatch, and retire up to four instructions per cycle. Unlike the POWERIII and PowerPC 604e, it does not employ any form of dynamic branch prediction. Rather, it prefetches up to 8 instructions from the branch target into a branch target buffer during normal execution, predicts the branch not taken and continues to fetch from the current instruction stream, and then, once the branch is resolved in the dispatch stage, either continues fetching from the current instruction stream with no penalty or flushes the instructions fetched after the branch and starts fetching from the branch target buffer, with a penalty of at most one and often zero cycles. The PowerPC RS64-II also has a 64KB, two way set associative L1 instruction cache, a 64KB L1 data cache (associativity information not available), and a 4MB, four way set associative unified L2. The processor clock is 340 MHz.

The POWERIII², the most complex of the three processors, is a 64-bit, superscalar, out of order, speculative execution, in order completion machine. It is targeted mainly for scientific and commercial applications. It has two single cycle integer units, one multiple cycle integer unit, one branch/ condition register unit, two load/ store units, and two three stage pipelined floating point units. It can fetch up to four, dispatch up to eight, and complete up to four instructions in the same cycle. It has a 256 entry BTAC and a 2048 entry (bits per entry not available) branch history table for use in branch prediction. Its memory system consists of a 32 KB, 128 way set associative, two way interleaved L1 instruction cache, a 64 KB, 128 way associative, four way interleaved L1 data cache, and a 4MB, four way set associative unified L2. All caches are nonblocking. The POWERIII is designed with separate buses to memory and L2 for greater memory bandwidth. To the same end, it can perform up to four data cache operations in the same cycle (with certain restrictions). The POWERIII also employs a data prefetching mechanism which detects sequential data access patterns and prefetches cache lines to match these patterns. The processor clock is 222 MHz.

The three web server workloads used in this work (HTML, Java servlet and CGI) are representative of real world static and dynamic web server workloads. These are the same simple workloads used by Radhakrishnan and Rawson³ and Radhakrishnan and John⁴. To understand the execution characteristics of these three workloads, it is necessary to understand how each type of web server workload generates the data that is sent to the HTTP client. An HTML page return is static. In other words, when an HTML server receives an HTTP request for an HTML page, it simply sends the requested page, which is already on disk or in memory, to the client that requested it. CGI scripts and Java servlets are dynamic in that once an HTTP request is received, some computation is done by the server to create data on the fly that is then sent back to the client. Of course, the exact mechanisms by which CGI scripts and Java servlets dynamically create data are different.

Since each of these web server workloads have different mechanisms by which they service client requests, it is to be expected that they exhibit different performance characteristics. In previous workload characterization work done on the Intel PentiumPro and Pentium using the same workloads as will be used in this study⁵, the Java servlet was shown to have the lowest CPI, the lowest instruction and data cache miss rates,

-
1. Borkenhagen and Storino, 1999.
 2. Papermaster, Dinkjian, Mayfield, Lenk, Ciarfella, O'Connell, and DuPont, 1998.
 3. Radhakrishnan and Rawson, 1998.
 4. Radhakrishnan and John, 1999.
 5. Radhakrishnan and Rawson, 1998.

and the highest frequency of parallel instruction issue. It was followed in performance in all of these categories by the CGI script and then by the HTML server. Also, interestingly, Radhakrishnan and Rawson found that, while each of the server workloads exhibited treelike branch behavior with lower than average predictability, the Java servlet had a higher branch prediction ratio (and, as to be expected, a higher ratio of instructions completed to instructions dispatched) than the other two workloads. This is surprising since interpreted Java programs typically have higher branch misprediction rates. It will be useful to confirm or contradict the branch behavior on the three PowerPC microarchitectures with the observed behavior on the PentiumPro.

METHODOLOGY

Client requests will be generated by an automatic client generator, which will be run on a remote machine. The client generator allows the selection of the number of simultaneous client requests and the total number of requests. For the purposes of this study, all of the requests in a given run will be simultaneous, since the load on the server is determined by the number of simultaneous client requests. The number of simultaneous requests will be varied in order to expose any bottlenecks or nonscalability in performance. Each workload will be run with 20, 50, and 100 client requests.

The web server software used will be Apache Web Server 1.3.12, the CGI script interpreter built into Apache Web Server, and Apache JServ 1.1.2 (a module of Apache Web Server) with JDK 1.1.8 from IBM and JSDK2.0 from Sun Microsystems.

The files requested by the clients will be simple, example pages included in the Apache software. The reasons for this are twofold: to allow quicker measurements and to ensure that one type of server workload is not more complex than the other. The essential characteristics of each type of workload should not depend on what files are being requested by the clients. Further, these three workloads are reasonably representative of real world web transactions, as they include both static and dynamic transactions.

Performance measurements will be made using the built in performance monitors in each machine. A performance monitor is hardware built into the machine that can collect various measurements during execution such as cycles, instructions completed, instructions dispatched, branch mispredictions, branch frequency, access latency in each level of cache, etc. IBM has developed an API that can be used to program and collect data from any PowerPC performance monitor. Calls from this API will be used to interface with the performance monitor.

Since each performance monitor has a limited number of counters, and there are far more relevant events to measure than there are counters, several runs are required to gather all of the desired measurements. The arrangement of measurements will be designed to count as many measurements which are relevant to each other as possible in the same run. For example, the number of branches will be counted simultaneously with the number of branches mispredicted. One series of runs to collect all of the necessary measurements will be henceforth referred to as a cycle. Ten cycles of measurements will be run for each workload on each machine for each of the three values of simultaneous client requests (20, 50, and 100). Thus, 10x3x3x3, or 90 measurement cycles, will be run in total, 30 cycles on each machine.

Once results are collected, a trimmed mean will be taken from the ten values for each measurement (i.e., the highest and lowest of the ten values will be discarded, and a mean will be taken from the remaining eight values).

EVALUATION OF RESULTS

Since these three PowerPC microarchitectures are significantly different from each other in terms of cache size, fetch, dispatch, and completion width, superscalarity, branch prediction mechanisms, execution order, etc., not much can be gleaned from simply comparing the performance of the processors on these workloads. Therefore, the focus of this investigation will be to discover how efficiently each processor is using its own available resources and exploiting ILP. Central focus will be given to the efficiency of speculation and how successfully each processor can predict branch direction and recover from mispredictions. Attention will also be given to how efficiently available superscalarity is used, as well as whether any particular pipeline stages, functional units, or data structures within the microarchitecture constitute bottlenecks to performance. In other words, each processor's performance will be evaluated according to its own ideal performance, not the performance of another machine.

STATUS

The performance monitor interface software and scripts to automate data collection have been tested on a PowerPC RS64-II and a PowerPC 604e system. Preliminary problems in data collection are being solved in consultation with the IBM Server Group. Work is being done right now to obtain access to all three machines from IBM. Measurements will be made as access to these systems is obtained. Currently, a PowerPC RS64-II system has been obtained and is being worked on. The current plan is for all of the measurements to be completed by Fall 2000.

REFERENCES

- Radhakrishnan, R. and Rawson, F. L. III, "Characterizing the Behavior of Windows NT Web Server Workloads Using Processor Performance Counters", Presented at the First Workshop on Workload Characterization, November 1998, Dallas, Texas, Also appears in *Workload Characterization: Methodology and Case Studies*, IEEE Computer Society Press, edited by Lizy Kurian John and Ann Marie Maynard, pp. 76-85.
- Radhakrishnan, R. and John, L. K., "A Performance Study of Modern Web Applications," Proceedings of the European Parallel Processing Conference (Euro-Par 99), Lecture Notes in Computer Science, Springer, pp. 239-247.
- Papermaster, M., Dinkjian, R., Mayfield, M., Lenk, P., Ciarfella, B., O'Connell, F. and DuPont, R. "POWERIII: Next Generation 64-bit PowerPC Processor Design", White Paper, IBM Corporation, 1998.
- Borkenhagen, J. and Storino, S., "4th Generation 64-bit PowerPC-Compatible Commercial Processor Design", White Paper, IBM Corporation, <http://www.rs6000.ibm.com/resource/technology/nstar.html>, January 1999.
- "PowerPC 604e Microprocessor User's Manual", IBM Corporation and Motorola Corporation, 1998.