Chapter 4

# CHARACTERIZATION OF BUS TRANSACTIONS FOR SPECWEB96 BENCHMARK

Prasant Mohapatra
*Department of Computer Science and Engineering*
*Michigan State University, East Lansing, MI 48824*
prasant@cse.msu.edu

Hariharan Thantry
*Department of Computer Science and Engineering*
*Michigan State University, East Lansing, MI 48824*
thantryh@cse.msu.edu

Krishna Kant
*Server Architecture Laboratory*
*Intel Corporation, Beaverton, OR 97006*
krishna.kant@intel.com

**Abstract** In this Chapter, we have characterized the bus traffic generated by the SPECweb96 benchmark in a four-processor SMP system with NT/4 operating system. We observe that the benchmark is very network intensive, in terms of computation and bandwidth consumption. The study shows that the transactions across the four processors are well balanced and do not have any long-range dependencies. There is a substantial amount of locking contention in the benchmark as is reflected by a large number of hit-modified transactions. In addition, a significant amount of spatial locality exists in the bus transactions that can be exploited through efficient cache organization. The inferences from this study can be used for designing efficient Internet servers.

**Keywords:** SPECWeb96, Cache pollution, bus traffic characterization, contention, locking, Bus Invalidates(BILs), Bus Read Invalidates(BRILs)

# 1. INTRODUCTION

The explosive growth in the use of Word Wide Web (WWW) necessitates the development of a common basis to compare and evaluate the performance of Internet servers. SPECweb96 was the first standardized benchmark developed by the Standard Performance Evaluation Corporation (SPEC) [1] for evaluating web server performance. It is a simple benchmark that serves requests from clients for a static set of files. SPECweb96 will be superseded by SPECweb99 that will include dynamic web traffic. Since the proportion of accesses to static files still dominates in many web servers, the results of this study based on the SPECweb96 benchmark is useful for a majority of servers.

A few attempts have been made to characterize the web traffic [2] - [10]. These studies have analyzed and characterized the behavior of the request patterns to web servers. The design of performance study of the NCSA web server is reported in [11]. The Internet traffic also has an impact on the intercommunication behavior within a server, especially the ones that use multiple processors and a hierarchical memory organization. In fact, most of the current generation Internet servers belong to this category. To the best of our knowledge, there has been no study reported on the characterization of the processor-memory intercommunication and the bus transactions due to the Internet traffic.

Our goal is not to characterize the Internet traffic, but to analyze the behavior of bus transactions within a symmetric multiprocessor (SMP) server using the SPECweb96 benchmark. The processors in an SMP system are interconnected to the memory through a front-side bus (FSB). We monitor the traffic on this bus and characterize the traffic flowing through it due to the SPECweb96 benchmark workload. Specifically, we focus on characterizing the locking/contention and locality (both spatial and temporal) behavior of the transactions. These behavior were analyzed on the basis of the type of transactions (hit, miss, hit-modified, explicit and implicit writes, etc.) observed on the FSB. We observed a significant amount of locking contention generated by the benchmark. Furthermore, a substantial amount of spatial locality exists in the bus transactions that can be exploited through efficient cache management strategies. However, the current cache implementations seem to get polluted resulting in higher miss rates. Thus, this study will enable designers to identify bottlenecks and consider issues that can enhance the performance of web servers for SPECweb96 type of transactions.

The rest of the chapter is organized as follows. We describe the system configuration and the data collection method in Section 2. A preliminary analysis of traces is done in Section 3. In Section 4, we report the general characteristics of the bus traffic. The temporal characteristics followed by the spatial characteristics are reported in Sections 5 and 6, respectively. The conclusions are outlined in Section 7.

## 2. SYSTEM DESCRIPTION AND TRACING

We use an SMP-based web server for our experiment for collecting the traces of bus transactions. Here, we describe the details of the system configuration and the experimental set-up. The traces were collected from a 4-processor Intel Pentium II Xeon based Web server using a HP bus-probe and logic analyzer. The Xeon family of processors is designed for high end servers and consists of a 512K L2 cache. It follows the MESI cache coherence protocol, which is common among most multiprocessor systems [12]. The system had Microsoft NT4.0/SP4 operating system and Microsoft Internet Information Server (IIS) v4.0 for providing HTTP service. The SPECweb96 workload was generated using eight Intel Pentium Pro clients (running NT4.0/SP4), each configured for four load generation threads.

By booting up the server with different number of enabled processors, we were able to obtain traces for one, two, and four-processor cases. Using the onboard jumpers, the processors were also made to run at 300, 400 and 500 MHz core frequencies in order to study the impact of the processor speed on bus transactions. In all cases, processor modules with 1 MB of second level cache were used. The server BIOS has the capability of disabling the first and/or second level processor caches, and thereby obtain a trace of the traffic entering the caches; however, in the data analyzed here, this capability was not used.

One limitation of the tracing system was the inability to offload the trace in real-time. Consequently, the tracing system must be stopped when the available onboard buffer becomes full. Therefore, it was not possible to take very long traces. For the analysis purposes, we stitched together about 10 trace segments, each segment covering a period of about 60 ms for 4P/500 MHz case, and about 240 ms for 1P/300 MHz case. Switching from one trace segment to the next introduces some irregularities and may not properly preserve the locality properties. However, the lengths of the individual segments are much larger than the number of segments, which makes the effect of these irregularities negligible.

## 3. TRACE ANALYSIS

In SPECweb96, each client process repeats the following cycle: sleeps for some fixed amount of time, issues a HTTP GET request, waits for the complete file to be received, and then repeats the cycle. The server throughput is measured as the number of operations (or transactions) per second. Each request is also logged into the HTTP log.

The bus transactions obtained were grouped in the following categories for analysis:

1. *Aggregate bus transactions:* It represents the raw aggregation of all the bus transaction types. The aggregate bus transactions gives a measure of the amount of system load.

2. *Chipset transactions:* These transactions appear on the bus because of the I/O operations of the chipset. We differentiate the chipset transactions from the rest because of the long-range dependencies in address access patterns observed in the chip-set transactions. The study of the locality issues and contention traffic must include only the transactions due to the processors activity on the FSB.

3. *Code reads:* Code reads represents the instruction read operations from the memory.

4. *Data reads:* Data reads refer to the bus transactions for reading data from the memory. The code reads and data reads are differentiated to study their locality behavior individually.

5. *Data writes:* The bus transactions for writes could be because of explicit writes or implicit writes. Explicit data writes occur, when a modified address line is evicted, either because the L2 cache is full, or the data line has been requested in an exclusive mode by some other processor. Implicit data writes are due to the requests for sharing of data that is held in an exclusive mode at a processor.

6. *Bus invalidation line (BIL):* When a native processor wants to write to a shared data, it issues a BIL transaction to invalidate the corresponding line at other caches and the copy in the memory.

7. *Bus Read invalidation line (BRIL):* When a processor wants to modify a data that is not available in its own cache, or it needs to acquire a lock variable, it issues a BRIL transaction. As a result of this transaction, a line is read into the local cache and its copies at all other processors and memory are invalidated.

8. *Hit-modified (HITM):* The HITM transactions are due to hits in remote caches to the lines that are modified by the remote memory. The lock contentions creates a lot of HITM transactions in the FSB.

9. *Clean hit (HIT) transactions:* The HIT transactions are due to the transactions that find a clean copy in a remote cache. The intensity of HIT transactions reflect the amount of data sharing in the benchmark.

The time series and histogram plots were obtained in both temporal and spatial domain for analysis. The time is expressed in terms of the FSB clock cycle and the addresses reflect the physical address. We derived the autocorrelation

functions to study the long-range dependence behavior of the transactions. We have analyzed the characterizations in three categories: general, temporal, and spatial, and the details are reported in the following sections. All the results shown in this chapter are measured using 500 MHz core frequency. We have obtained the results with other frequencies. The trend and behavior of the bus transactions are the same and are not dependent on the core frequency.

## 4. GENERAL CHARACTERISTICS

In terms of its overall characteristics, the benchmark is fairly network intensive, both in terms of required network bandwidth and in terms of CPU time spent on networking activities. This observation was made during the trace collection by recording the bus and CPU utilizations. Figure 1 shows the time series and histogram plots of the aggregate bus transactions. For the time series, the arrival rates are plotted with respect to the bus clock cycles. The histogram is generated on the basis of the frequency of the arrival rates. It was observed that the aggregate bus transactions comprise of about 51% reads, 22% read-invalidates (BRILs), 21% bus-invalidates (BILs), and 5% explicit write backs. These results are summarized in Figure 2. Reads dominate bus transactions, which in turn are dominated by the HITM transactions. The hit ratio of read transactions (in other processors' caches) is about 84%, of which 75% are HITMs. Among all writebacks, about 89% are implicit and the rest are explicit. The high percentage of HITM and implicit writeback transactions indicate a very high degree of data contention among the four processors. The high proportion of BRILs indicate the possibility of high level of contentions for locks. The hit ratios for the native caches are about 60% for reads and 30% for writes. The lower hit ratio is primarily because of possible cache pollutions and the inability of the cache organization to exploit the available locality. The locking and contention issues can be further investigated by analyzing the traffic in temporal and spatial domains.

## 5. TEMPORAL CHARACTERIZATION

The aggregate transactions on the bus had a mean rate of 0.122 per FSB clock cycle with almost negligible variation (0.0005). The autocorrelation function decays very quickly indicating that there is no long-range dependence in the aggregated bus transactions. The behavior of bus traffic generated by each of the processor is almost identical and the characteristics are quite similar to that of the aggregated traffic. The mean rate of traffic generated by each of the processor is 0.025 per FSB clock cycle with negligible variance and has no long-range dependence. The time series plot and the corresponding histogram of all the processors, individually as well as collectively, exhibit very similar behavior. The chip-set transactions have a mean rate of about 0.021 which is

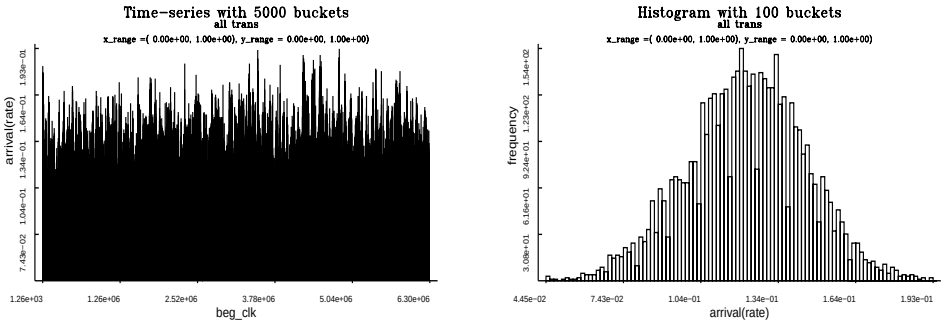


*Figure 1* Timeseries and histograms of all bus transactions in the 4P system. “beg_clk" refers to the beginning clock cycle of the FSB.

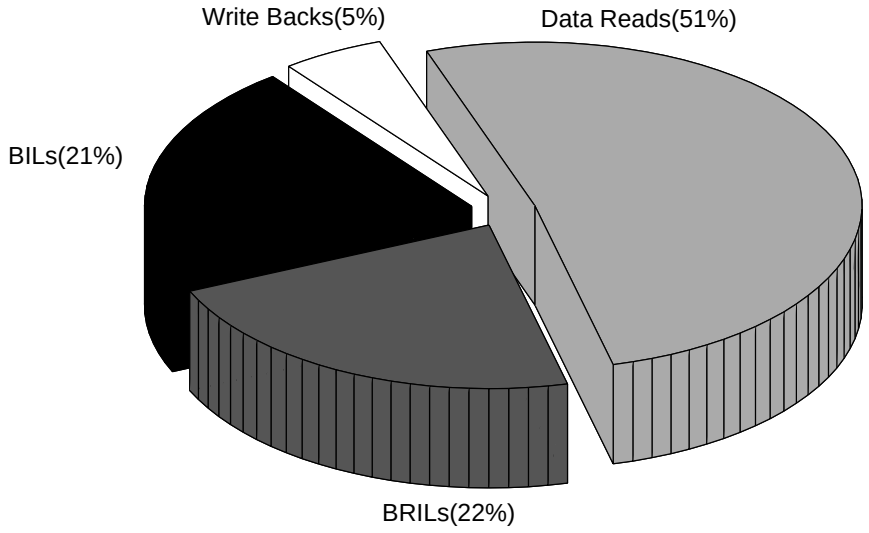


*Figure 2* Relative proportions of the aggregate transactions.

very close to the traffic generation rate of a single processor. It can thus be inferred that the load across all the processors in a multiprocessor environment seemed to be well-balanced for the SPECweb96 benchmark.

The code reference behavior is also identical across all the processors and does not have any long-range dependencies in the temporal domain. The code references are very low compared to the data references. The data-read bus transactions generated by all the processors is about 0.04 FSB per clock cycle with negligible variance and is evenly spread across all the processors. The

autocorrelation function of these transactions also decays fast indicating the absence of long-range dependencies in the read transactions of a single processor.

Unlike other bus transactions, the data-write transactions, which are about one-tenth of the data read transactions, exhibit very slowly decaying autocorrelations. This behavior may be attributed to the fact that data writes are given low priority (in case of conflict, you will get the line from the write buffer) and are buffered in the write back buffer. Upon finding the bus idle, these transactions are executed in spurts (see Figure 3) and thus display a behavior that has a slowly decaying autocorrelation function. The bursty nature of the data writes creates long-range dependence for this type of traffic. Similar to the other transactions, the data write transactions are identical across all the processors, and thus exhibit the same behavior.

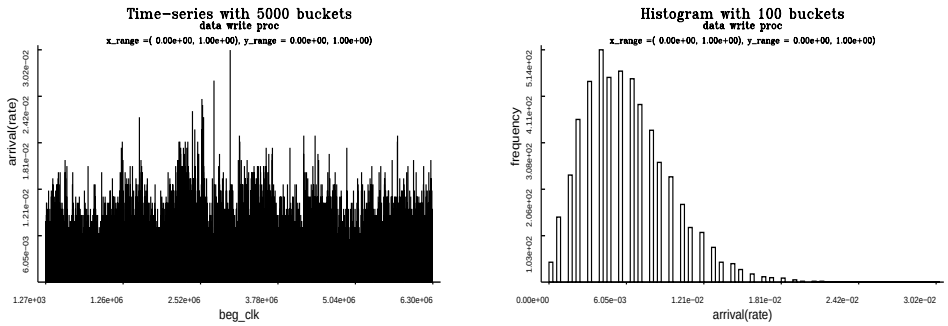


*Figure 3* Timeseries and histograms of data writes in the 4P system. "beg_clk" refers to the beginning clock cycle of the FSB.

The bus invalidations are of two types: BIL (bus invalidate line) and BRIL (bus read invalidate line). Both BILs and BRILs have a mean rate of 0.026 per FSB clock cycle and have very low variance. This rate is relatively high considering the data read and write rates, which reflects higher levels of sharing and/or contentions. The rate of BRILs has a higher impact on the performance as it quantifies the magnitude of contentions. BILs are associated with the write operations as discussed in Section 3. Neither BILs nor BRILs have any significant correlations for a single processor or for all the transactions taken collectively from all the processors.

It is observed that a significant amount of contentions for locks exists in the SPECweb96 benchmark traces. It is not possible to do an exact analysis of locking behavior with the traces with the L2 cache turned on. We have estimated the proportion and effect of the locking behavior. BRILs in quick succession (within 100 FSB cycles) to the same cache line by different processors were used as indications of possible lock contentions. Although this approach does

not necessarily identify locks, it may be a very close approximation. Using this approach, it was observed that about 13% of the BRILs were related to lock contentions. Since the proportion of BRILs was significant, we can infer that there is a lot of lock contentions in the SPECweb96 benchmark traces. The average contention length is about 240 FSB clock cycles, and the contending bus requests were spinning on the locks for an average period of 130 FSB clock cycles. This high degree of locking (both in number and duration) impact may be reduced through efficient code scheduling.

## 6. SPATIAL CHARACTERIZATION

We have characterized the spatial locality in terms of spatial distance. The spatial distance is defined as the difference in the physical addresses between two consecutive transactions of the same type.

The bus traffic for all transactions when observed on an aggregated basis show significant level of spatial locality. In fact, about 20% of all the bus transactions are within 10 cache lines apart (in terms of the addresses); a significant portion of those, about 75%, are within a single cache line, and more than 90% are within a difference of only four cache lines. About 80% of the chipset transactions have a spatial distance of only one cache line as expected. After excluding the chipset transactions, about 10% of bus transactions are within 10 cache lines apart, of which about 80% are within four cache lines. These results can be viewed graphically in Figure 4. For any single processor, this ratio is more than 85%. However, we need to examine this locality in terms of different transactions. The aggregated locality cannot be exploited without considering each of the transactions in isolation for a single processor.

There is a significant amount of spatial locality in code access transactions by individual processors as shown in Figure 5. Furthermore, the degree of spatial locality is similar across all the processors. The relative spatial distances are shown in Figure 7. On the average, about 40% of the code access transactions have a spatial locality of less than 4 cache lines, of which more than 70% of the transactions are within a single cache line. These statistics reflect a very high level of spatial locality, which is not exploited by the cache organization. The temporal relationship of these accesses needs to be analyzed in order to develop techniques for exploiting this high level of code locality. If these spatially close codes have substantial temporal locality, then it could be exploited through prefetching or by increasing the cache line size.

The data read transactions for the individual processors are shown in Figure 6. In this case also the behavior of the transactions are indistinguishable with respect to the processors. In the data read transactions shown in Figure 7, about 14% of the transactions are within a single cache line, and about 25% of the transactions are within four cache lines. These levels of locality are quite

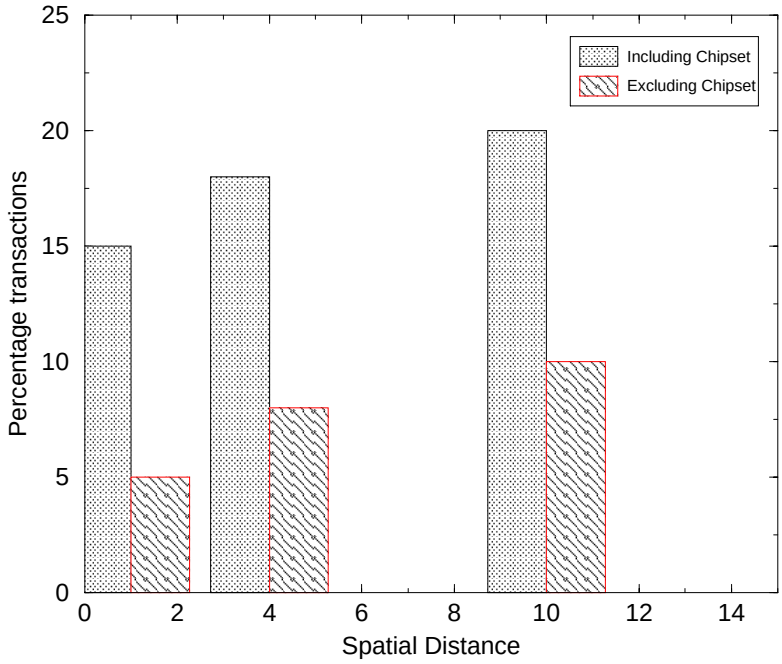


*Figure 4* Spatial locality with and without chipset transactions.

high and could be exploited by using larger cache lines. However, we need to examine the access pattern correspondingly in the temporal domain and also with respect to the other transactions. A little less than one percent of the data read transactions have a spatial distance of zero. It should be noted that this distance is between two consecutive data read transactions; there may be other types of bus transactions between them. However, since the data read transactions are very frequent, it was observed that very few of the other types of transactions were embedded within consecutive read accesses to the same address.

About 15% of the data write transactions on the bus have a spatial distance of four or fewer cache lines. More than one-half of these transactions have a spatial distance of zero. Data write transactions are about an order of magnitude less frequent than the data read transactions. So the likelihood of other types of transactions being interleaved between consecutive write transactions is relatively higher. If there are many such transactions, then the data may be getting evicted between two successive write transactions. A proportion of these data writes may be involved in lock contentions. Thus after a write transaction, the line may be getting invalidated by some other processor.

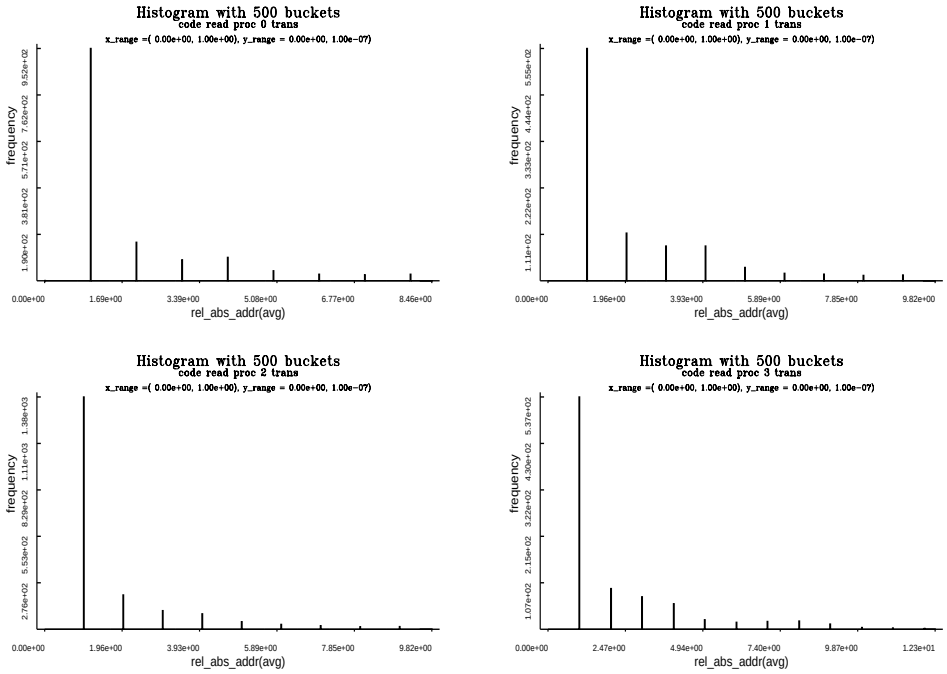


*Figure 5* Spatial locality in code reads.
"rel_abs_addr" refers to the relative absolute addresses.

Similar to the read and write transactions, the BILs and BRILs also exhibit spatial locality. About 8% of the aggregate BIL and BRIL transactions have a spatial distance of one cache line. A negligible number of BIL transactions have a spatial distance of zero. However, about 1.7% of the BRIL transactions have a spatial distance of zero for each of the processors. Most of these transactions are due to the locking contentions. These locking contentions are potential performance bottlenecks and may increase with the increase in the number of processors. Thus the increase in cache line size will reduce the BRILs and may thus have a positive impact on the overall performance, provided the larger lines do not introduce a lot of false sharing and contentions. The high proportion of BILs and BRILs also contributes to the cache miss ratio as more and more data gets invalidated polluting the cache. The locking contentions also reduce the proportion of cached data that gets reused for an extended length of time.

The HITM transactions constitute a very high proportion of all reads (about 75%). About 25% of these transactions have a spatial contiguity of four or less number of cache lines. In fact, about 15% of HITM transactions have a spatial distance of one cache line. These results are summarized in Figure 8. Back to back HITMs have negative impact on the performance as it creates bubbles

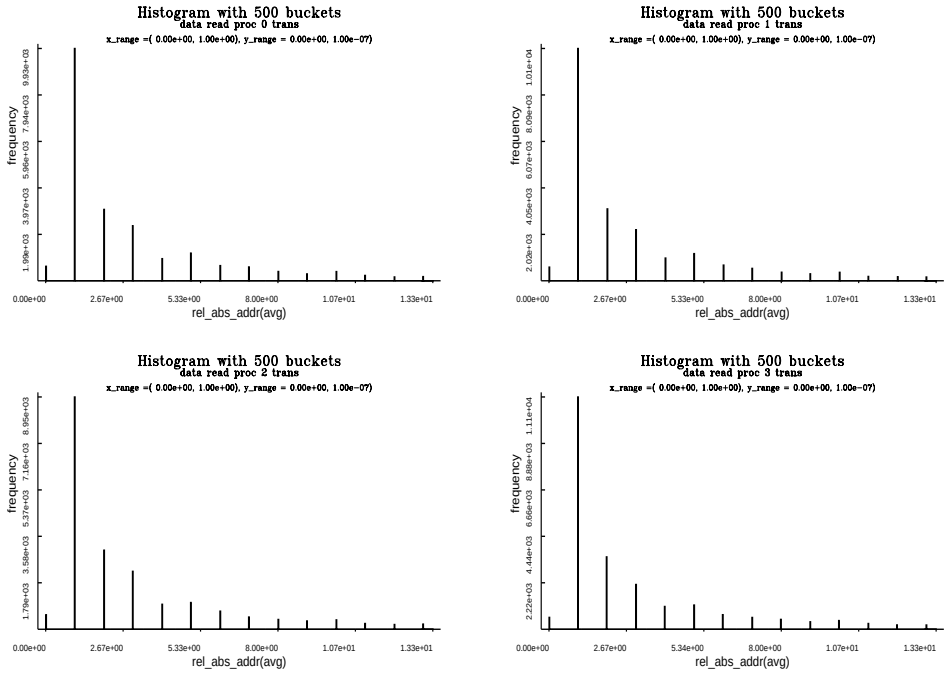


*Figure 6* Spatial locality in data reads.
"rel_abs_addr" refers to the relative absolute addresses.

in the data read lines and may also fill up the write buffer. Note that a spatial distance of one does not necessarily means that the transactions were back to back. There may be other types of transactions interleaved between them. The number of clean hits is usually one-tenth of that of the HITMs, and about 10% of clean hits are within a spatial distance of four cache lines. The HITM transactions reflect the sharing of data among the processors and it can thus be inferred that a significant amount of data sharing is done in the SPECweb96 transactions. Once again, the spatial locality of both the clean hits and HITM transactions can be exploited by larger cache lines provided they also exhibit locality in the temporal domain.

## 7. CONCLUSIONS AND FUTURE WORK

The traffic characterization reported in this Chapter exposes the locking and locality behavior of the bus transaction generated by the SPECweb96 benchmark on an SMP system. The study also analyzes the mixtures of various types of transactions and their dependencies in the temporal and spatial domains. In future, we plan to analyze the bus traces with the level-2 (L2) cache turned off. Such a study will enable us to characterize the hierarchical caching behavior

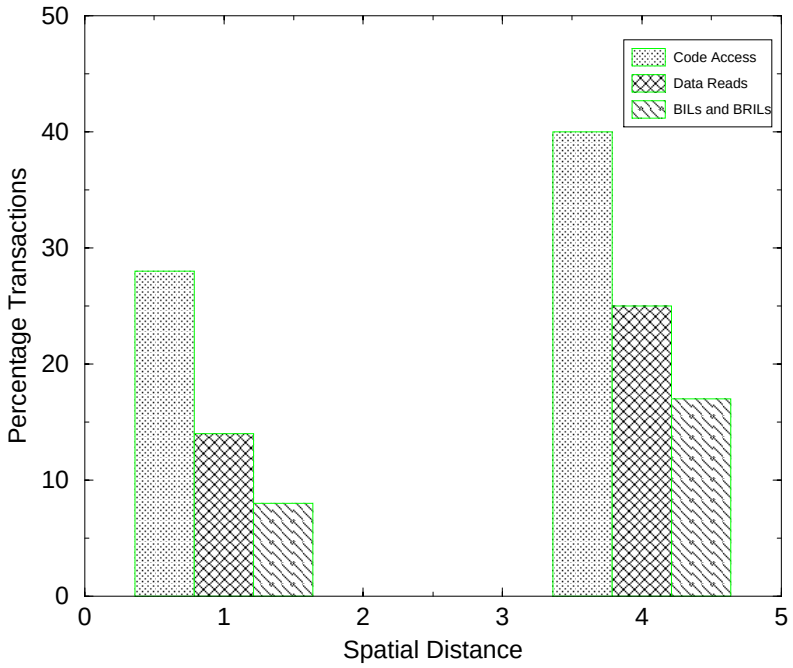


*Figure 7* Spatial characterization of individual transactions.

and we will be able to identify the possible causes of cache pollution. In addition, we need to examine the spatial and temporal locality in an integrated manner to identify the proportion of the locality of transactions that can be exploited through efficient cache organization. Several architectural alternatives can be examined to exploit the available spatial and temporal locality in the SPECweb96 bus traces. These alternatives include the L1 cache size, L2 cache size, and cache line size.

## Acknowledgments

This work was done while Mohapatra was visiting Intel Corporation during the summer of 1999, and Thantry was working as a summer intern for Intel Corporation in 1999.

## References

[1] SPEC Report, “An Explanation of the SPECweb96 Benchmark," `www.spec.org/osg/web96/webpaper.html`.

[2] M.F. Arlitt and C.L. Williamson, "Web Server workload characterization; The search for invariants," *ACM SIGMETRICS Conference on Measurement and Modeling of Computer Systems,* pp. 126-137, May 1996.

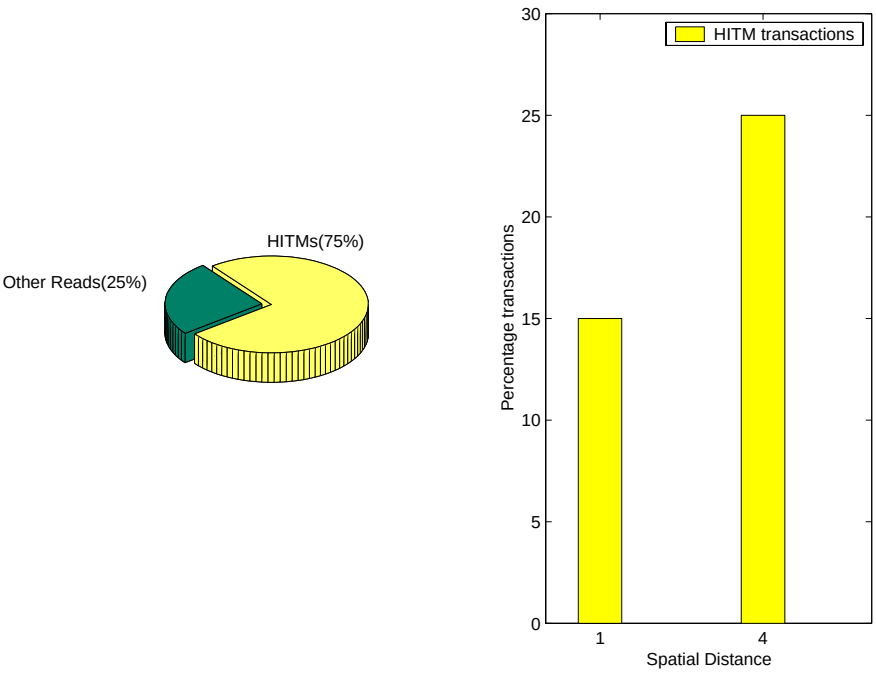


*Figure 8* Spatial characterization of HITM transactions.

[3] C. Cunha, A. Bestavros and M. Crovella, "Characteristics of www client-based traces," CS Tech Report BU-CS-95-010, Boston University, April 1995.

[4] M.E. Crovella and A. Bestavros, "Self-similarity in world wide web traffic: Evidence and possible causes," *IEEE/ACM Transactions on Networking*, 1997.

[5] L. P. Slothouber, "A model of web server performance," *Fifth International World Wide Web Conference*, (Paris, France), May 1996.

[6] Y. Hu, A. Nanda and Q. Yang, "Measurement, analysis and performance improvement of the apache web server," *IEEE International Performance, Computing and Communications Conference*, Phoenix/Scottsdale, Arizona, February 1999.

[7] V. Almeida, A. Bestavros, M. Crovella and A. de Oliveira, "Characterizing reference locality in the WWW," *Fourth International Conference on Parallel and Distributed Information Systems*, Miami Beach, FL, IEEE, December 1996.

[8] V. Paxon, “End-to-End Routing Behavior in the Internet," *IEEE Trans. on Networking*, vol. 5, no. 5, pp. 601-615, Oct. 1997.

[9] K. Park, G.T. Kim, and M.E. Crovella, “On the relationship between file sizes, transport protocols, and self-similar network traffic”, *Proc. 4th Int. Conf. Network Protocols(ICNP’96),* Oct. 1996, pp 171-180.

[10] V.Paxson and S.Floyd, “Wide-area traffic: The failure of Poisson modelling”, *IEEE/ACM Trans. Networking*, vol. 3, pp. 226-244, June 1995.

[11] T. Kwan, R. McGrath, and D. Reed, “NCSA’s World Wide Web Server: Design and Performance”, *IEEE Computer*, Vol. 28, No. 11, pp. 68-74, November 1995.

[12] D. E. Culler, J. P. Singh, and A. Gupta, *Parallel Computer Architecture: A Hardware/Software Approach*, Morgan Kaufmann Publishers, 1999.