On the Evaluation of x86 Web Servers using Simics: Limitations and Trade-Offs

Francisco J. Villa, Manuel E. Acacio, and José M. García

Universidad de Murcia, Departamento de Ingeniería y Tecnología de Computadores
30071 Murcia (Spain)
{fj.villa,meacacio,jmgarcia}@ditec.um.es

Abstract. In this paper, we present our first experiences using *Simics*, a simulator which allows full-system simulation of multiprocessor architectures. We carry out a detailed performance study of a static web content server, showing how changes in some architectural parameters affect final performance. The results we have obtained corroborate the intuition of increasing performance of a dual-processor web server opposite to a single-processor one, and at the same time, allow us to check out *Simics* limitations. Finally, we compare these results with those that are obtained on real machines.

1 Introduction

Multiprocessor systems are increasingly being used for executing commercial applications, among which we can find web servers or On-Line Transaction Processing (OLTP) applications. As a consequence of the use of multiprocessors in these fields, simulating multiprocessor architectures running commercial applications accurately becomes important. Opposite to scientific applications, there are some characteristics of commercial workloads that make their simulation challenging. In particular, the activity of the operating system is very important, as well as the interaction with memory hierarchy, storage system and communication network. *Simics* [1] is a full-system simulator which allows us to simulate all these aspects and obtain accurate simulation results.

In this paper, we use *Simics* to evaluate three different architectures executing a static web content server, being *Apache* the web server and *httperf* the utility which places the workload at the server.

2 Related work

Up to not long ago, the methodology used for evaluating commercial workloads in multiprocessors consisted in firstly generating memory references of applications, and then, using these references to feed a user-level simulator. For example, in [2] Ranganathan *et al.* study the performance of OLTP and decision support systems based on this methodology.

The appearance of full-system simulators, like *SimOS* [3] or *Simics* [1], has significantly simplified the evaluation of commercial workloads, as these simulators allow modelling elements such as the operating system, the I/O subsystem and so on.

Recently, several studies have appeared in which *Simics* is used as the simulation tool employed for the evaluation. In [4, 5], it is presented an exhaustive study of several commercial applications, including a static content web server and the TPC-C benchmark. The authors also identify one of the problems concerned with simulation of commercial applications: the variability they show.

3 Simulation results and limitations

In this Section, we present the results that we have obtained using *Simics* and compare them to the results obtained using real machines. In our evaluations, we have considered three different server architectures: two single-processor architectures with L2 cache sizes of 512 KB and 1024 KB respectively, and a dual-processor architecture in which each processor has a L2 cache of 512 KB. In the case of real machines, the single-processor architecture with a L2 cache of 1024 KB has not been analysed.

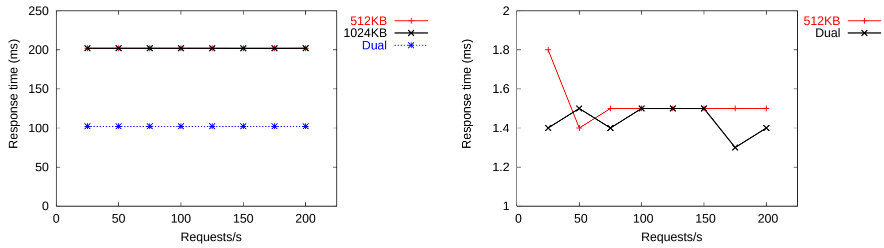
We measure the response time of *Apache* in each case as a function of the number of requests that are received. For this, we have executed 1000 requests referred to 10 web pages with an average page size of 537 bytes. This page size has been selected in order to avoid the influence of the interconnection network on the results.

We have carried out eight tests for each sever architecture, in which the total number of requests that *Apache* must process has been set to 25, 50, 75, 100, 125, 150, 175 and 200 respectively. Starting with the results of the simulations, Figure 1(a) shows the average response time that has been obtained in each case. This metric is provided by *httperf*. As we can see, the dual-processor server has greater performance than those that employ a single-processor, with an average response time of approximately half the response time of the single-processor severs (which show almost the same response time).

On the other hand, Figure 2(a) shows the evolution of the number of requests that are dispatched as a function of the total number of requests. This metric is provided by the *Apache* server. Although the dual-processor server is able to dispatch more requests than the single-processor architectures, the performance difference is lower than the observed for the response time.

Once we have seen how *Simics* can help us to analyse the behavior of a commercial web server, we want to check how accurate are the results the simulator provides. For this, we have repeated the experiments, but this time we have employed real computers. Figures 1(b) and 2(b) show the results we have observed for these tests.

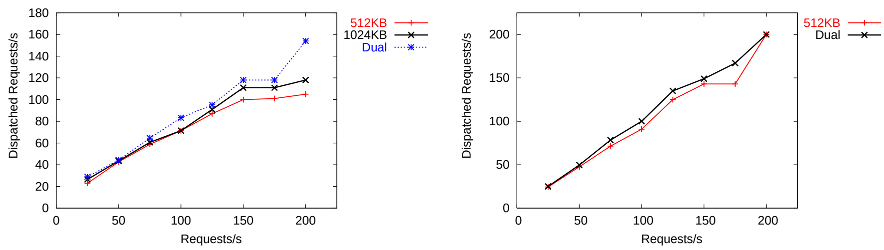
Comparing these results to the obtained with *Simics*, we find that there are notable differences between them. In the case of the response time, it is scaled down by a factor of almost 100. In fact, the performance difference between dual and single-processor real servers is negligible. Something similar occurs with the number of requests that are dispatched. Although simulation results showed that the dual-processor server could sustain a larger request per second rate than the single-processor one, in the real environment we find that for the experiments we have carried out, single and dual-processor servers provide almost the same results in terms of the number of requests that are dispatched.



(a) Simulation results

(b) Real results

Fig. 1. Average response time as a function of the requests received per second.



(a) Simulation results

(b) Real results

Fig. 2. Dispatched requests per second as a function of the requests received per second.

Therefore, we can conclude that the low detail level when modeling x86-like processors prevents *Simics* from be able to reproduce the results that would be reached in the real world. Specifically, *Simics* doesn't implement out-of-order execution for these processors. In this way, we think that the *x86-Simics* machine is appropiate as functional simulator but not as timing simulator.

4 Additional information obtained with *Simics*

Using *Simics* we can easily obtain statistics of the processor and memory hierarchy, one of the main advantages of the simulator compared to real machines, for which collecting these measures is harder. In this Section, we analyse CPU and cache statistics, exploring their influence in the performance of the architectures that are evaluated.

4.1 CPU statistics

The first important fact is that the number of instructions executed in user mode is 50 times lower than the number of instructions executed in supervisor mode. Comparing the statistics obtained for the single-processor server with a L2 cache of 1 MB to the dual-processor server, we notice that the number of instructions executed in user mode is almost the same in the two cases, but it is distributed between the two processors in the case of the dual-processor server. It does not happen the same with the instructions executed in supervisor mode, since in this case each CPU executes the same number of instructions than the single-processor server. These numbers corroborate the important influence that the operating system has on the final results.

4.2 Cache statistics

The most noticeable difference is the increase in the L2 cache miss rate found for the single-processor architecture with a L2 of 512 KB, compared to the single-processor architecture with a L2 of 1024 KB. The increasing in the number of L1 cache invalidations is also a remarkable result. This fact is a consequence of the increased number of replacements (what is caused by the larger number of misses), which leads to invalidate more L1 blocks in order to maintain the inclusion property.

Finally, in the case of the dual-processor server configuration, the large number of L1 cache invalidations must be considered again, although the explanation is just as before. Regarding miss rates, they are just like the preceding ones for the first level caches, whereas for the second level ones this rate ranges between the values that are obtained for the single-processor configuration with a L2 cache of 1024 KB and the values obtained for the configuration with a L2 cache of 512 KB.

5 Conclusions

In this paper, we have introduced the evaluation of a functional simulator which allows us to simulate all the aspects that are critical in the execution of commercial workloads, such as the I/O subsystem and the operating system. However, we have found that the simulator does not provide an accurate model for the x86 family of processors, which leads to obtain different results than those that would be obtained using real computers. We think that the impossibility of using an out-of-order execution model for this family has a negative influence in the results that we have obtained.

References

1. Magnusson, P. S. *et al.*: Simics: A Full System Simulation Platform. IEEE Computer **35** (2002) 50–58
2. Ranganathan, P. *et al.*: Performance of Database Workloads on Shared-Memory Systems with Out-of-Order Processors. In: ASPLOS-VIII. (1998) 307–318
3. Rosemblum, M. *et al.*: Complete Computer System Simulation: The SimOS Approach. IEEE Parallel and Distributed Technology: Systems and Applications (1995) 34–43
4. Alameldeen, A.R. *et al.*: Simulating a $2M Commercial Server on a $2K PC. IEEE Computer **36** (2003) 50–57
5. Alameldeen, A.R. *et al.*: Evaluating Non-deterministic Multi-threaded Commercial Workloads. In: CAECW-02 (2002) 30–38