

Simulating Complex Enterprise Workloads using Utilization Traces

Parthasarathy Ranganathan and Philip Leech

Hewlett Packard

partha.ranganathan@hp.com; phil.leech@hp.com

ABSTRACT

Simulating enterprise environments poses several challenges by virtue of the complex nature of enterprise applications and the systems that need to be modeled. In this paper, we propose an alternate approach to simulating such environments that uses resource utilization traces from real deployments in conjunction with high-level models that correlate resource utilization to system metrics like power and performance. We discuss the requirements and tradeoffs from such a simulation methodology and present results from the BladeSim simulator that instantiates this methodology. Our results show that this approach can enable a host of systems' optimization studies at the enterprise level.

INTRODUCTION

Enterprise compute environments have seen a lot of dramatic changes in the last decade [RJ05]. New applications have been driven by the adoption of the internet and the services models. Traditional business processing workloads such as enterprise resource planning (ERP), customer relationship management (CRM), transaction processing, data warehousing, and decision-support have also increased in importance (and complexity). In addition, server, storage, and network virtualization are being rapidly adopted, leading to greater workload consolidation and the blurring of traditional system boundaries. This increase in the number and complexity of applications has led to higher demands on enterprise IT systems. Large data centers have supplanted individual department-level compute clusters. At the same time, trends towards multi-core processors and modular blade servers have led to ever greater compute capacity being packed in smaller and smaller form factors.

All these trends in enterprise systems and workloads lead to corresponding challenges in evaluation methodologies for these environments. Ideally, when evaluating new system architectures, we would like to develop prototype implementations and deploy them in live enterprise environments running the targeted workloads. However, building prototype hardware, especially at scale, requires significant resources. Additionally, getting enterprises to trust their business applications to prototype hardware is a difficult proposition. Even if we were to mirror some of the software setup of real-world enterprise platforms, exercising this

setup to show behavior akin to a live deployment is challenging. Furthermore, it is hard to perform a detailed design space exploration with the one specific hardware implementation that a prototype would represent.

Given these drawbacks with building real hardware prototypes, simulation is often the only practical way to test architectural ideas and assess system performance. Simulators provide the flexibility to modify and analyze the impact of various architectural parameters and components as well as enable more detailed statistics collection than real hardware. These benefits make simulation useful even for projects that will eventually implement hardware.

However, simulation approaches pose several challenges as well. Evaluating system architectures in large-scale enterprise environments requires full-system modeling for large-scale parallel and distributed systems. The widely-used SimpleScalar toolset [BAS96] only models a single system and focuses primarily at the processor level. Other simulators, such as M5 [BD+06], GEMS [M+05], SimOS [R+95], and Simics [M+02], provide more detailed full-system simulation, but do not scale well for large-scale systems. In addition, these approaches face the same complexity with deploying and exercising real-world enterprise applications discussed earlier. Further, many of these simulators have slowdowns corresponding to 100-1000X the actual elapsed time. This either leads to long simulation times or requires significant scaling of the workload, often limiting the scope of the studies.

In this paper, we address these challenges. Specifically, we make two contributions.

- We propose an alternate simulation approach to perform architectural evaluation in enterprise environments. Our approach uses resource utilization traces as the workload input, and high-level models that correlate resource utilization to metrics like performance and power, to emulate the system behavior. We discuss how resource traces can be collected with little overhead on real enterprise deployments, and how additional system models can be created for new architectural configurations by

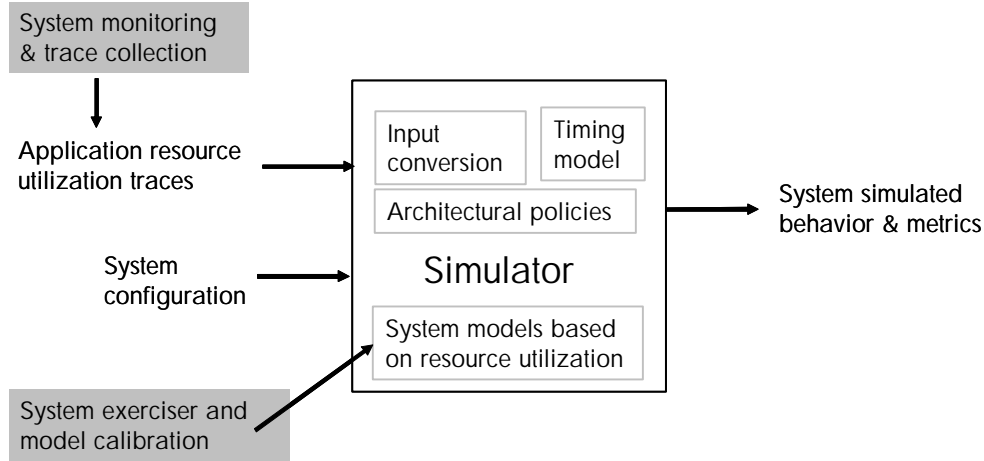


Figure 1: Resource-utilization based simulation approach.

simple offline experiments. Our approach can enable a host of systems' optimization studies at the enterprise level.

- We discuss our experiences with building a simulator, *BladeSim*, that instantiates this methodology. As one example of the application of this simulation methodology, we present results using *BladeSim* for performance-power tradeoffs in data centers using traces from more than 100 servers from nine enterprise deployments.

The rest of the paper is organized as follows. Section 2 provides a broad overview of our simulation approach. Section 3 discusses *BladeSim* and discusses results using *BladeSim*. Section 4 summarizes the paper and discusses ongoing work and future directions.

PROPOSED SIMULATION APPROACH

Figure 1 shows the overview of our proposed simulation approach. The key intuition behind our approach is that at an enterprise level, system performance can be approximated by understanding the relative utilization levels of various system components. Our approach therefore uses resource utilization traces as the workload input, and high-level models that correlate resource utilization to metrics like performance and power, to emulate the system behavior. We discuss these two aspects in greater detail below.

Resource utilization traces as proxy for workloads

To obtain resource utilization traces for our simulator, we leverage the observation that most commercially deployed systems have rich support for measuring system resource usage or allow simple software scripts to enable these in real-time with low overhead. For example, instrumentation frameworks such as HP's Open-

View, IBM's Tivoli, ganglia [SKMC03], etc., aggregate information from a variety of sources and present a unified monitoring and control interface to administrators. Consequently, it is relatively straightforward to obtain resource utilization traces from real-world environments. Specifically, we propose collecting information about CPU, memory, disk, and network utilization from workloads which can be replayed in our simulator. This approach allows us to bypass the complexity of deploying and exercising complex real-world applications, but still mimic the behavior of running these workloads.

Utilization-based system characterization models

However, there are several challenges with using traces that only capture resource utilization information. First, the input traces capture utilization on the system being measured. We need to determine the variation in resource utilization when a different system is modeled. For example, a resource utilization of 100% on a 1 GHz system could well correspond to a resource utilization of 50% on a 2 GHz system. The second, and more important, challenge is to be able to correlate resource utilization to performance. This will help us better understand the impact on application latencies and throughputs. Similarly, we need to correlate resource utilization to other metrics of interest such as the power consumed in the system.

To address both the challenges discussed above, we propose the creation of high-level system models that correlate resource utilization to the metric of interest. Figure 2 shows examples of two such models. We run systematic experiments using a synthetic load generator (such as *sstress* or *gamut* [MCFR05]), to execute a pre-determined synthetic stress kernel, while controlling the resource utilization in progressive steps.

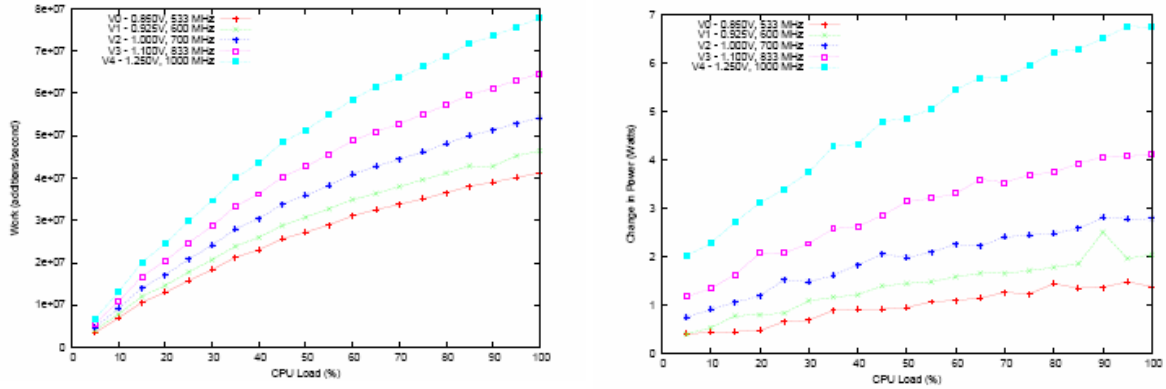


Figure 2: Illustrative examples showing models correlating resource utilization to power and performance

For example, we can exercise the processor with an compute-intensive loop at CPU utilizations from 0% to 100% in small increments. At each of these data points, we measure the metric of interest – e.g., power (using a power meter) and the performance (in terms of the useful work done, as reported by the benchmark). Based on these experiments, we create high-level models that correlate resource utilization to metrics such as power or performance.

For example, using the methodology described above, Figure 2(a) illustrates the variation of performance with CPU utilization for the experimental blade server discussed in the next section. The x-axis shows resource utilization levels (varied at 5% increments) and the y-axis shows relative performance. In this graph, the performance is measured in terms of adds/second, but this can be changed to other metrics such as throughput, latency, etc. Figure 2(b) similarly illustrates the variation of power with resource utilization.

Figures 2(a) and 2(b) additionally illustrate how we compute a series of models for different hardware configurations. In this case, the different configurations include five different voltage and frequency states for the blade server being simulated. For simulations involving heterogeneous servers, we anticipate having a model corresponding to each server class being modeled.

The models shown in Figures 2(a) and 2(b) can be used to address the two challenges described earlier. To “translate” resource utilization into performance, we just look up the corresponding entry in the model. To convert resource utilization in one system configuration to its equivalent in another system configuration, we convert the resource utilization to performance and then look up the entry for this performance in the new configuration. In the event that the simulated performance is higher than the maximum performance for that new configuration, we use the

maximum value and mark the system as having a “performance degradation” and measure the net loss in work done for that instant of time. For example, if a system transitions from a 1GHz configuration to a 500MHz configuration when the workload utilization (in the 1GHz case) was 100%, our model would show the new system utilization to be 100% but with the corresponding performance degradation. Focusing on the performance, as measured in these models, and not metrics like clock frequency, helps us get a more accurate picture of the actual performance degradation.

The examples above (and Figure 2) discuss fairly simple models to illustrate the idea. One can imagine more complex models to capture the correlations between resource utilization and system-level metrics. For example, Shivam et al [SBC06] discuss sophisticated methods using active learning techniques to generate performance models using resource utilization traces. Similarly, Economou et al [ERKR06], discuss more detailed methods to capture power models based on resource utilization. These models can be computed through one-time system exercising using synthetic benchmarks or through online learning algorithms in the context of live application deployments.

Benefits and tradeoffs

As can be seen from the discussion, our approach helps us address the two key challenges discussed earlier, namely the complexity of large enterprise workloads and the simulation time to model large scale-deployments of multiple servers. Workloads are now modeled as resource utilization traces and systems are modeled as look-up tables based on the characterization functions. This enables us to model complex workloads running on multiple servers without having to worry about simulation time constraints.

However, obviously, this approach is not applicable for all scenarios. In particular, our approach is based on

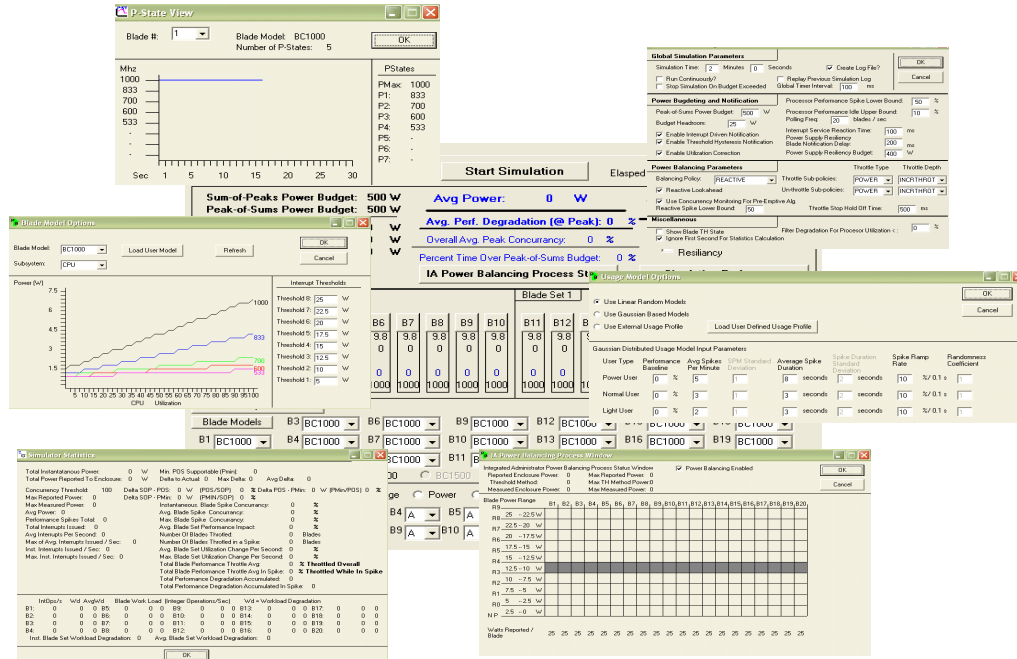


Figure 3: BladeSim GUI screen shots

high-level models of system behavior based on resource utilization traces. Consequently, using this simulation methodology to understand detailed component-level behavior will give erroneous results. Second, our approach approximates performance degradation purely in terms of the degraded capacity of the server to perform work, going from one configuration to another. This assumes a model where work is dropped when the system is unable to process it. The complex effects when work is buffered and queued are not modeled in our approach.

Finally, our work assumes that the system being evaluated can be characterized to obtain the model. This requires the need to have at a (prototype) system to run the calibration experiments or an analytical model of the variation of performance with resource utilization. Given that our goal is not to study detailed component-level interactions, but rather to study *complex* workloads on *large-scale* datacenter-like configurations, this approach is acceptable, but for evaluation of single systems, this may not be a reasonable model.

INSTANTIATION: THE BLADESIM SIMULATOR

In this section, we discuss our experiences with building and using a simulator, *BladeSim*, that instantiates the methodology discussed above. We have validated and used *BladeSim* for several evaluation studies. As one example, we present results using *BladeSim* for performance-power tradeoffs in data centers using

traces from more than 100 servers from nine enterprise deployments.

Experiment details

We use *BladeSim* to study “ensemble-level” power management [RLIC06] in the context of a blade enclosure. The key idea is to leverage non-synchronized power peaks over a collection of systems to reduce the power provisioning across a collection of systems. The goal is to study performance degradation when the system is throttled to enforce the power budget (through the use of voltage and frequency states), under varying policy and system parameter assumptions.

Our simulator (screenshots in Figure 3) is built based on the methodology described in the last section. It takes resource utilization traces as input and models the impact of different enclosure controller policies on performance and power under various load and policy conditions. The basic operation of the simulator is as follows. The main simulator loop operates on a timer that matches the time of the simulated system. When reading the input trace file, the simulator uses this timer to access resource utilization data at the corresponding time stamp associated with it.

We had a prototype blade system that we exercised to create high-level models that correlate resource utilization to power consumption, identify changes in resource utilization with changes in frequency, and convert changes in utilization to the corresponding impact on actual work done. To report performance

Site	Workload and trace length	Servers	Avg	90th %	Max	sum-peaks	Worst	Savings
1	Backend of pharmaceutical company	26	87	138	307	1128	2600	88%
2	Web hosting infrastructure for worldcup98 web site [2]	25	256	481	1166	1366	2500	53%
3	SAP-based business process application in large company	27	585	691	919	1654	2700	66%
4	E-commerce web site of a large retail company	15	83	166	234	591	1500	84%
5	Backend for thin enterprise clients - company 1	10	138	184	298	729	1000	70%
6	Backend for thin enterprise clients - company 2	14	102	159	287	1253	1400	80%
7	Front-end customer facing web site for large company	8	119	187	255	467	800	68%
8	Business processing workload in small company	3	78	132	225	278	300	25%
9	E-commerce web site of small company	4	90	136	197	228	400	51%
	All sites	132	1540	1872	2682	7694	13200	80%

Figure 4: Summary of the traces studied in the evaluation of ensemble-level power management using BladeSim. The last column shows the potential savings in resource (and power) provisioning from ensemble-level management [RLIC06].

Power budget	Algorithm	Site 1	Site 2	Site 3	Site 4	Site 5	Site 6	Site 7	Site 8	Site 9
400W	Preemptive	1%	1%	4%	1%	1%	2%	2%	5%	4%
425W		1%	1%	3%	1%	1%	2%	2%	5%	3%
450W		1%	1%	3%	1%	1%	2%	1%	4%	3%
400W	Reactive	0%	0%	0%	0%	0%	0%	0%	1%	0%
425W		0%	0%	0%	0%	0%	0%	0%	0%	0%
450W		0%	0%	0%	0%	0%	0%	0%	0%	0%

Figure 5: Simulator results for enclosure level power management for 9 enterprise traces.

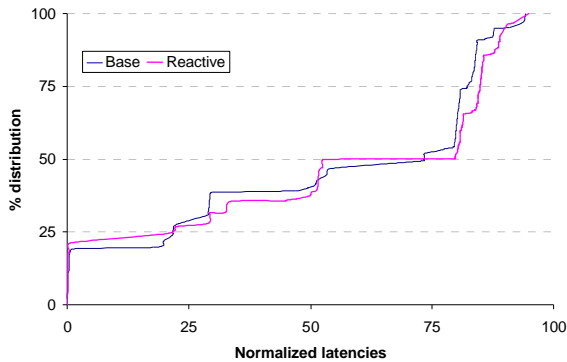


Figure 6: Variation in latencies for VNCplay.

degradation (workload slowdown) in this section, we compare the total work done across all blades in the enclosure across different policies. This metric represents the degradation for the entire solution.

Results

Using synthetic traces, we validated high-level properties of the simulator with the prototype and found results within small error ranges (within 5%). We also ran experiments using the vncplay benchmark (results in Figure 6) on both the simulator and the prototype blade and again, our results were similar.

Figure 5 summarizes simulation results for the 9 enterprise sites discussed in Figure 4. We consider three different values of the power budgets – 450W, 425W, and 400W – corresponding to enclosure-level

power budget reductions of 10%, 15%, and 20%. Note that this corresponds to equivalent reductions of approximately 25%, 37%, and 50% of the total processor power budget. As seen from the Figure, overall, enclosure-level power management can achieve power budget reductions with marginal reductions in total solution performance. (More details of the results can be found in [RLIC06].)

Figure 6 shows the data for interactive workloads, based on the VNCplay tool developed by Zeldovich et al [ZC05]. This tool records a user’s interactive session with a system and allows it to be replayed multiple times under different configurations. This benchmark allows us to measure the impact of our approach in an interactive GUI environment. When comparing the total execution time of the session with and without enclosure-level power management, our results showed relatively little variation (less than 1% even with 20% reduction in the enclosure power budget). This is consistent with the simulation data.

We also evaluated several different policies. In addition to the pre-emptive and reactive classification shown in Figure 4, we also considered two different approaches for which servers to throttle (and unthrottle): one where priority was given to servers at the highest power and one where priority was given to servers at the lowest utilization. Similarly, we considered two different approaches on the level to which to throttle the servers: either incrementally transition to the next power-state, or transition to the lowest or highest power state. For the nine real-world

enterprise traces, these algorithmic variations made little difference, because of the low utilizations and concurrencies. For the synthetic workloads with higher concurrencies, in general, deep throttling of a few servers consuming the highest power was desirable to throttling a large number of servers at low utilization. Our experiments varying the polling frequency and the interrupt service time showed that the algorithms are fairly robust at adapting to changes in these parameters. It is natural to use a proportional sharing model, in which the power budget is allocated to the contending workloads proportionally to weights assigned to each workload.

SUMMARY

In this paper, we discuss a simulation approach for enterprise environments that addresses the two key challenges of capturing workload complexity and modeling large number of systems. Our approach uses utilization traces that can easily be obtained on live enterprise deployments, but yet capture the complexity of resource usage in these workloads. We have also put together a collection of more than 100 real-world server application traces representing a spectrum of applications from nine different enterprise deployments. Second, we use high-level models that correlate resource utilization to metrics like power and performance and can scale to a large number of systems. We have developed such models for a variety of systems and we discuss how more models can be developed for future systems.

As an instantiation of this approach, we have built a simulator that implements this methodology and used it for several high-level studies of enterprise systems. In this paper, we present results from its use for one specific optimization – namely, ensemble-level power management. We are currently collecting more enterprise traces and are further refining our simulation approach. In the future, we plan on releasing our traces, models, and simulator to the broader academic community.

Overall, as enterprise workloads continue to get more and more complex, and deployments get spread over larger and larger number of machines, we believe approaches like ours that leverage real-world application traces and high-level system models will be an important component in the continuum of future simulation solutions.

ACKNOWLEDGEMENTS

We would like to thank Khaldoun Alzien, Nidhi Aggarwal, David Irwin, Hernan Laffitte, Charlie Shaver, and John Sontag for their input, feedback, and support of this work. We are indebted to Martin Arlitt as well as our enterprise customers for the real-world traces.

REFERENCES

- [BAS96] D. Burger, T.M. Austin, and S. Bennett, Evaluating Future Microprocessors: The SimpleScalar Tool Set, tech. report 1308, Computer Sciences Dept., Univ. of Wisconsin-Madison, 1996.
- [BD+06] Binkert, N.L.; Dreslinski, R.G.; Hsu, L.R.; Lim, K.T.; Saidi, A.G.; Reinhardt, S.K.m The M5 Simulator: Modeling Networked Systems, IEEE Micro, Volume 26, Issue 4, July-Aug. 2006 Page(s): 52 - 60
- [ERKR06] D. Economou, S. Rivoire, C. Kozyrakis, and P. Ranganathan, "Full-system Power Analysis and Modeling for Server Environments," Workshop on Modeling, Benchmarking, and Simulation (MoBS), June 2006
- [M+05] M.M.K. Martin et al., "Multifacet's General Execution-Driven Multiprocessor Simulator (GEMS) Toolset," Sigarch Computer Architecture News, vol. 33, no. 4, Sept. 2005, pp. 92–99.
- [M+02] P.S. Magnusson et al., "A Full System Simulation Platform," Computer, vol. 35, no. 2, Feb. 2002, pp. 50–58.
- [MCFR05] J. Moore, J. Chase, K. Farkas, and P. Ranganathan. "Data Center Workload Monitoring, Analysis, and Emulation," In the Eighth Workshop on Computer Architecture Evaluation using Commercial Workloads (CAECW), February, 2005.
- [RJ05] P. Ranganathan and N. Jouppi, "Enterprise IT trends and implications on system architecture research," Proceedings of the International Conference on High-Performance Computer Architecture (HPCA), 2005
- [RLIC06] P. Ranganathan, P. Leech, D. Irwin, and J. Chase, "Ensemble-level Power Management for Dense Blade Servers," Proceedings of the International Symposium on Computer Architecture (ISCA), June 2006
- [R+95] M. Rosenblum et al., "Complete Computer System Simulation: The SimOS Approach," IEEE Parallel & Distributed Tech., vol. 3, no. 4, Winter 1995, pp. 34–43.
- [SBC06] P. Shivam, S. Babu, J. Chase. Learning Application Models for Utility Resource Planning. In IEEE International Conference on Autonomic Computing (ICAC), June 2006.
- [SKMC03] F. D. Sacerdoti, M. J. Katz, M. L. Massie, and D. E. Culler. Wide area cluster monitoring with ganglia. In Proceedings of the IEEE Cluster 2003 Conference, 2003.
- [ZC05] N.Zeldovich and R.Chandra. Interactive Performance Measurement with VNCplay. In Proceedings of the FREENIX Track: USENIX Annual Technical Conference, April 2005.