Model Discovery for Energy-Aware Computing Systems: An Experimental Evaluation

Zhichao Li, Radu Grosu, Koundinya Muppalla, Scott A. Smolka, Scott D. Stoller, and Erez Zadok
Department of Computer Science, Stony Brook University

***Abstract*—**
We present a model-discovery methodology for energy-aware computing systems that achieves high prediction accuracy. Model discovery, or system identification, is a critical first step in designing advanced controllers that can dynamically manage the energy-performance trade-off in an optimal manner. Our methodology favors Multiple-Inputs-Multiple-Outputs (MIMO) models over a collection of Single-Input-Single-Output (SISO) models, when the inputs and outputs of the system are coupled in a nontrivial way. In such cases, MIMO is generally more accurate than SISO over a wide range of inputs in predicting system behavior. Our experimental evaluation, carried out on a representative server workload, validates our approach. We obtained an average prediction accuracy of 77% and 76% for MIMO power and performance, respectively. We also show that MIMO models are consistently more accurate than SISO ones.

***Keywords*-energy; performance; system identification; control theory; file compression**

I. Introduction

The carbon footprint of the IT industry, though only 2% of the world economy, is estimated to be equal to that of the entire aviation industry [2]. Making matters worse, server and data-center energy use has been growing rapidly in recent years [18]. Concerns about energy consumption in the computing arena have led to the emergence of *energy-aware computing systems*, where energy, or power, is a first-class citizen in the design process [8], [16], [17].

The goals of energy-aware system design include saving energy without sacrificing performance, and supporting flexible, dynamic trade-offs between energy consumption and performance. Accurate models of energy consumption and performance are prerequisites for any foundation for the design of energy-aware systems.

Such models are also prerequisites for the application of *control theory* to energy-aware systems. We advocate the use of control theory in this context, as it has the potential to yield advanced controllers that dynamically manage the energy-performance trade-off in an optimal manner.

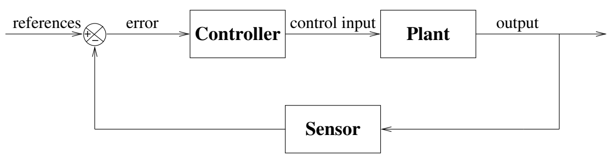


Figure 1. Plant with feedback controller

Applying control theory to computing systems is a three-step process [7]: (1) discover an accurate model of the plant (i.e., the computing system to be controlled) using *system identification*; (2) use the plant model to design and implement a feedback controller for the plant as shown in Figure 1; and (3) apply the controller to real systems (i.e., backup system, mail system, etc.) and validate the functionality of the whole system experimentally.

The process of model discovery for energy-aware systems, in advance of controller design, is complicated by a number of factors. We recently concluded an extensive year-long study [12] of the energy consumption and performance of a *file-compression server*, a representative workload involving both substantial CPU usage and disk I/O. We analyzed the effects of several input parameters, including compression algorithm, compression level, file type, persistent storage media, CPU Dynamic Voltage and Frequency Scaling (DVFS) level, and disk I/O scheduler—all under the Linux operating system.

Our experimental results identified three factors that complicate the system's energy and performance profiles: (1) *nonlinearity*, which makes the application of traditional control-theoretical techniques challenging; (2) *instability*, referring to significant fluctuations in outputs when inputs are held relatively constant; and (3) *multi-dimensionality*, referring to the vast number of possible inputs, outputs, and internal system states.

In this paper, we present a model-discovery methodology that mitigates the complexities we identified in [12] to achieve accurate plant models of energy-aware systems. Key to our approach is to use only *numeric* model parameters as inputs and outputs, and treat *non-numeric* model parameters as part of the workload. Our methodology favors MIMO models over a collection of SISO models, when the system's inputs and outputs are coupled in a nontrivial way. Our experimental evaluation, performed on a representative file-compressor server, validates our approach and shows that MIMO models are consistently more accurate than SISO ones.

II. Related Work

Control theory has been applied to database systems [4], storage systems [10], [11], Web servers [3], [15], and data centers [13], [19]–[21] to provide QoS (e.g., performance and power) guarantees. Abdelzaher et al. surveyed the application of feedback control to software systems [1]. Most of the models considered in these approaches are SISO, with only a few using MIMO.

Hellerstein et al. cconstructed a MIMO model for Apache Web servers, and proved that a single MIMO model outperformed a collection of SISO models in terms of prediction and control [3]. The MIMO advantage was also observed by Wang et al. in the context of high-density servers [19]. We obtained similar results for compression systems attempting to deal with the trade-offs between power consumption and performance.

III. Methodology

This section describes our methodology for system identification. Due to space limits, we do not discuss controller design and implementation in this paper.

We discuss several decisions that need to be made as part of system identification. For each system identification, we obtain a dataset by varying the inputs as described below and measuring the outputs. The first half of each dataset is used for training (i.e., system identification). We use the `pem` command in MATLAB's System Identification Toolbox [14] to identify the system under the following state-space model:

$$x(n+1) = Ax(n) + Bu(n) + Kw(n) \quad \text{(III.1a)}$$
$$y(n) = Cx(n) + Du(n) + w(n) \quad \text{(III.1b)}$$

where $u(n)$ are the inputs, $y(n)$ are the outputs, $x(n)$ are the internal states of the plant, and $w(n)$ is a white Gaussian noise representing uncontrollable inputs (e.g., execution of default system daemons) and output measurement errors at time n. Term $x(n+1)$ is the next internal state of the plant. All $u(n)$, $y(n)$, $x(n)$, and $x(n)$ are arrays. Matrices A, B, C, D, and K denote the significance or weight that each internal state and element in the input, output, and Gaussian noise has in determining the next state and output of the system.

We validate the model using the second half of the dataset. By computing the coefficient of determination R^2, we measure how accurately the model predicts the measured values in the second half of the dataset.

Sampling time: The sampling time is the period of time at which sensors measure the current system status. It is chosen based on the time scale of the behavior of the system of interest, in order to obtain an accurate system model. We experimentally observed that a sampling time of 10 seconds works well for our compression system.

Model order: We use state-space models. For such models, the *model order* is the number of components (dimensions) of the state space. A larger order usually results in a model with higher accuracy but greater complexity; conversely, a smaller order usually results in a simpler model but with lower accuracy. Therefore, choosing a suitable model order is a trade-off between model accuracy and model complexity. In our experiments, we choose the model order in a systematic way as follows: start with a first-order model, repeatedly increase the order by one, and terminate when the improvement in prediction accuracy is less than 10%. This procedure leads to the orders given in Table I.

Model type: As shown in Figure 2, the system model has two inputs (CPU frequency and Number of workers) and two outputs (Power consumption and Performance). The system can be modeled using one MIMO model as shown in Figure 2(a), or two separate SISO models as shown in Figure 2(b). SISO1 is achieved by fixing the number of worker threads to 2 (average number of CPU cores), while SISO2 is achieved by fixing the CPU frequency at level 4 (the average of the possible DVFS values). We evaluate all of these models in Section IV.

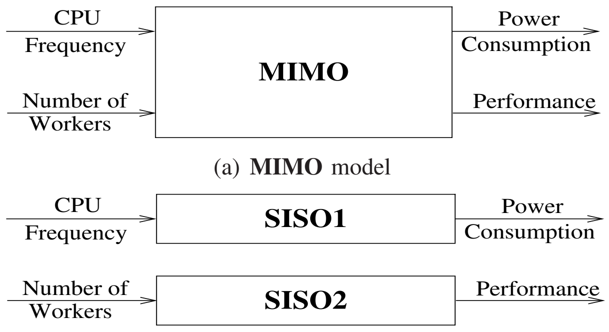
Input space: To obtain accurate models, the input sequences used in the experiments must thoroughly exercise the system's possible behaviors in the input range of interest. A simple approach that works well in practice is to vary each input *sinusoidally*, using relatively prime periods for different inputs [7]. We approximate this approach using simpler triangular instead of sinusoidal waves. For our file-compressor system, we vary the CPU frequency across all 8 possible values, with a period of 19 seconds. We vary the number of worker threads from 1 to 4 (there are 4 cores), with a relatively prime period of 13 seconds.

Normalization: The model is usually estimated using Recursive Least-Squares (RLS) estimation [6]. If the measurements have a large constant component, RLS tries to accurately predict this constant component and may thus fail to capture relatively small output changes due to input changes [9]. Therefore, both the input and output values should be normalized to zero-mean before applying the RLS technique.

IV. Evaluation

In this section, we introduce our experimental setup, compare MIMO and SISO prediction accuracies, and provide an in-depth analysis of our results.

We conducted our experiments on a Dell PowerEdge R710 server with one quad-core 2.395GHz Intel® Xeon^{TM} Nehalem CPU with dynamic frequency and voltage scaling (DVFS) support: 8 different frequencies with a difference of 1MHZ for the top 2 frequencies, and a difference of 133MHZ for the remaining 7 frequencies with Turbo Mode on. The machine has 24GB RAM, out of which we only used 2GB to force I/O to take place. We use the SAS disk of the server for the experiments. The server was running the Linux 2.6.18 kernel with the `acpi_cpufreq` module installed to enable software control of the CPU frequency.



(a) **MIMO** model

(b) Two separate **SISO** models

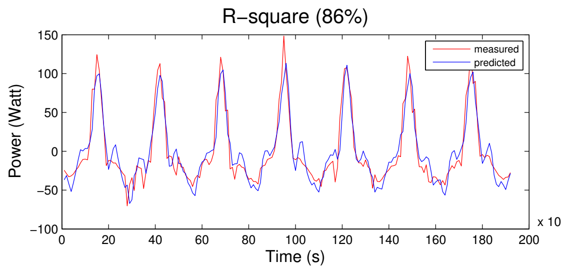
Figure 2. Our MIMO model and two-SISO models

We connected the server to a WattsUP Pro ES in-line power meter [5], which measures the power drawn by a device plugged into the meter, with resolution of 0.1 Watts. We used the `wattsup` Linux utility to download the recorded data from the meter over a USB interface to the test machine.

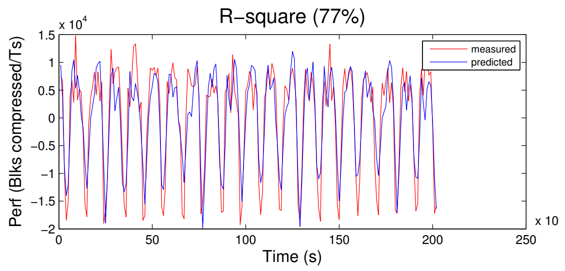
For our file-compression server, the worker threads keep compressing files until all files are compressed. We use a set of 8,000 text files produced by decompressing all Linux kernel tar-balls and stripping non-ASCII characters. Each of the text files is 10MB in size. Compression is performed by `gzip` at compression level 7, which is rather CPU-intensive, and hence creates a stronger relationship between the number of workers and performance.

We built the MIMO model from a dataset, called the multi-dimensinal data, obtained by varying both inputs and recording, for each sampling period, the average power consumption during the sampling period and the number of blocks written to disk during the sampling period. We built the SISO models from datasets called uni-dimensional data, obtained similarly except that one input is varied and the other input is fixed.

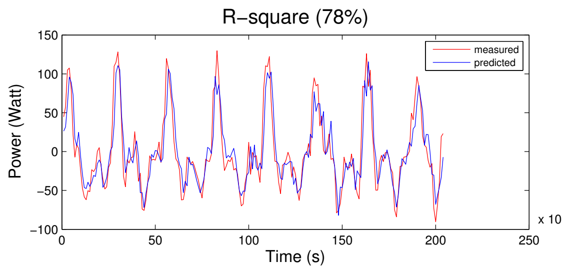
In the following discussion, we use accuracy numbers from the first run as representative numbers unless otherwise noted. We show the accuracy of the SISO models evaluated with uni-dimensional data in Figures 3(a) and 3(b). The two SISO models have good prediction accuracy (77% and 86%, respectively). This demonstrates that when CPU



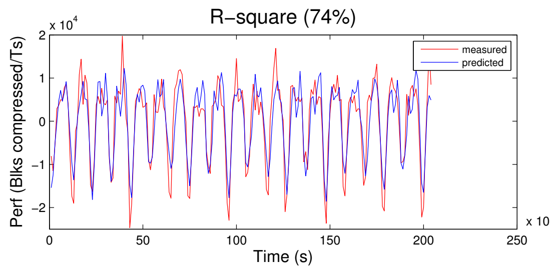
(a) **SISO1** model evaluated with uni-dimensional data



(b) **SISO2** model evaluated with uni-dimensional data



(c) **MIMO** model evaluated with multi-dimensional data for Power



(d) **MIMO** model evaluated with multi-dimensional data for Performance

Figure 3. Evaluation for one run of a MIMO model and two separate SISO models. Ts is the sampling time we use in the experiments.

frequency is the only varying input, it is a good indicator for power consumption; when the number of workers is the only varying input, it is a good indicator for performance. We also achieved an accuracy of 67% for number of workers vs. power, and an accuracy of 36% for CPU frequency vs. performance. This indicates that both inputs are coupled with both outputs in a nontrivial way. The accuracy of the MIMO model evaluated with multi-dimensional data appears in Figures 3(c) and 3(d). The accuracy is 78% and 74% for power and performance, respectively.

Although the SISO models achieve good accuracy when evaluated on uni-dimensional data (i.e., one input is held constant), in real usage of the system, we want to control both inputs. How accurate are SISO controllers in this context? To answer this question, we evaluate the SISO models with multi-

Model	Fixed Input	Order	Accuracy
MIMO	N/A	3	Power: 77% Perf: 76%
SISO1	1 worker	1	Power: 73%
SISO1	2 workers	1	Power: 73%
SISO1	3 workers	1	Power: 73%
SISO1	4 workers	1	Power: 71%
SISO2	2395MHz Freq	1	Perf: 43%
SISO2	1995MHz Freq	2	Perf: 61%
SISO2	1596MHz Freq	1	Perf: 44%

Table I
EVALUATION OF MIMO AND SISO MODELS WITH MULTI-DIMENSIONAL DATA.

dimensional data (specifically, with the second half of the multi-dimensional data). As expected, the SISO models generally have lower accuracy in this setting, because they do not take the variation of the other input into consideration, and both inputs are coupled with both outputs in the system, to varying degree. We ran each experiment multiple times and report averages unless otherwise noted. The results are summarized in Table I, quantify this effect and also demonstrate that, in some cases, the accuracy of the SISO model evaluated with multi-dimensional data is very sensitive to the value chosen for the constant input in the uni-dimensional data used to build the SISO model. Because real-world data *is* multi-variate, it suggests that SISO models are less suitable for production settings.

For example, in the case of SISO1, when the number of workers is fixed at 4, the power prediction accuracy is worse than that of the MIMO model. For other fixed numbers of workers, the accuracy results are comparable. In the case of SISO2, for all cases, the accuracy results are worse than that of the MIMO model. This shows that if the two separate SISO models are generated with appropriate training data, they can have accuracy comparable with that of the MIMO model for some of the metrics (e.g., power), but the two separate SISO models generally have lower accuracy than the MIMO model. Overall, MIMO models are more resilient across a wide range of training data.

V. CONCLUSIONS

As we have shown [12], even simple systems that de/compress files exhibit complex performance and energy profiles. Control theory holds great promise for the automated and optimal control of such energy-aware systems. Before designing a suitable controller, one must first identify a system model.

In this paper, we have presented a methodology for system identification, applied it to a representative file-compressor server, and experimentally evaluated the accuracy of SISO and MIMO models of power consumption and performance. Although potentially more complex, MIMO offers consistently higher prediction accuracy (76–77%). SISO was at best comparable to MIMO (73%), and at worse close to half as accurate (43%). Collectively, our results help to understand, quantify, and compare the behavior of MIMO and SISO models of energy-aware systems under representative workloads.

Based on the extensive data set we collected in [12], we plan to evaluate additional MIMO models, especially those with more than two inputs/outputs, and those that explore other dimensions (e.g., different compression algorithms and compression levels). We also plan to design and implement actual SISO/MIMO controllers and evaluate their ability to trade off power consumption and performance. An important research question is to quantify the costs (e.g., CPU, memory, power) for executing more complex MIMO-based controllers, and to investigate the effect of the model order on these costs.

VI. ACKNOWLEDGEMENTS

This work is sponsored in part by NSF awards CCF-0937854, CCF-0926190, CCF-1018459, and CNS-0831298; AFOSR grant number FA0550-09-1-0481; and an IBM Faculty Award. This work performed in part under the Stony Brook Advanced Energy Research and Technology Center (AERTC, www.aertc.org).

REFERENCES

[1] T. F. Abdelzaher, J. A. Stankovic, C. Lu, R. Zhang, and Y. Lu. Feedback performance control in software services. *IEEE Control Systems Magazine*, 23(3):74–90, 2003.

[2] J. Chang, J. Meza, P. Ranganathan, C. Bash, and A. Shah. Green server design: Beyond operational energy to sustainability. In *Proceedings of the 2010 International Conference on Power Aware Computing and Systems*, HotPower'10, 2010.

[3] Y. Diao, N. Gandhi, J. L. Hellerstein, S. Parekh, and D. M. Tilbury. Using MIMO feedback control to enforce policies for interrelated metrics with application to the apache web server. In *Proceedings of the Network Operations and Management Symposium*, pages 219–234, 2002.

[4] Y. Diao, J. L. Hellerstein, A. J. Storm, M. Surendra, S. Lightstone, S. Parekh, and C. Garcia-Arellano. Using MIMO linear control for load balancing in computing systems. In *2004 American Control Conferences*, 2004.

[5] Watts up? PRO ES Power Meter. www.wattsupmeters.com/secure/products.php.

[6] S. Haykin. *Adaptive Filtering Theory*. Prentice Hall, 2002.

[7] J. L. Hellerstein, Y. Diao, S. Parekh, and D. M. Tibury. *Feedback Control of Computing Systems*. Wiley-IEEE Press, 2004.

[8] N. Joukov and J. Sipek. GreenFS: Making enterprise computers greener by protecting them better. In *Proceedings of the 3rd ACM SIGOPS/EuroSys European Conference on Computer Systems 2008 (EuroSys 2008)*, Glasgow, Scotland, April 2008. ACM.

[9] C. Karamanolis, M. Karlsson, and X. Zhu. Designing controllable computer systems. In *Proceedings of the 10th Conference on Hot Topics in Operating Systems*, 2005.

[10] M. Karlsson, C. Karamanolis, and X. Zhu. Triage: Performance differentiation for storage systems using adaptive control. *ACM Trans. Storage*, 1(4), 2005.

[11] H. D. Lee, Y. J. Nam, K. J. Jung, S. G. Jung, and C. Park. Regulating I/O performance of shared storage with a control theoretical approach. In *NASA/IEEE Conference on Mass Storage Systems and Technologies (MSST)*. IEEE Society Press, 2004.

[12] Z. Li, R. Grosu, P. Sehgal, S. A. Smolka, S. D. Stoller, and E. Zadok. On the energy consumption and performance of systems software. In *Proceedings of the Fourth Israeli Experimental Systems Conference (ACM SYSTOR '11)*, Haifa, Israel, May/June 2011. ACM.

[13] H. C. Lim, S. Babu, and J. S. Chase. Automated control for elastic storage. In *Proceeding of the 7th International Conference on Autonomic computing*, ICAC '10, pages 1–10. ACM, 2010.

[14] L. Ljung. *System Identification (2nd ed.): Theory for the User*. Prentice Hall, 1999.

[15] Y. Lu, A. Saxena, and T. F. Abdelzaher. Differentiated caching services; a control-theoretical approach. In *21st International Conference on Distributed Computing Systems*, pages 615–624, 2001.

[16] D. Narayanan, A. Donnelly, and A. Rowstron. Write off-loading: practical power management for enterprise storage. In *Proceedings of the 6th USENIX Conference on File and Storage Technologies (FAST 2008)*, 2008.

[17] S. Gurumurthi and A. Sivasubramaniam and M. Kandemir and H. Franke. DRPM: Dynamic Speed Control for Power Management in Server Class Disks. In *Proceedings of the 30th Annual International Symposium on Computer Architecture*, pages 169–181, 2003.

[18] U.S. EPA. Report to Congress on Server and Data Center Energy Efficiency. Public Law 109-431, August 2007.

[19] X. Wang, M. Chen, and X. Fu. MIMO power control for high-density servers in an enclosure. *IEEE Trans. Parallel Distrib. Syst.*, 21:1412–1426, 2010.

[20] X. Wang and Y. Wang. Coordinating power control and performance management for virtualized server clusters. *IEEE Trans. Parallel Distrib. Syst.*, 22:245–259, February 2011.

[21] Y. Wang, X. Wang, M. Chen, and X. Zhu. Power-efficient response time guarantees for virtualized enterprise servers. In *Proceedings of the 2008 Real-Time Systems Symposium*, pages 303–312. IEEE Computer Society, 2008.