Statistical Inference for Efficient Microarchitectural and Application Analysis

Benjamin C. Lee :::: bclee@seas.harvard.edu
School of Engineering and Applied Sciences :::: Harvard University

Problem and Motivation

Parameter space exploration and optimization is a costly endeavor for both hardware and software systems, requiring mechanisms to accurately predict the effectiveness of particular hardware designs or software configurations. Existing approaches to prediction, such as cycle-accurate microprocessor simulation or analytical performance modeling, are unscalable as trends in computer architecture and high-performance computing drive exponential increases in the sizes of parameter spaces.

Hardware Domain. Current approaches to microarchitectural design space exploration are often inefficient and ad hoc due to the significant computational costs of modern microprocessor simulator infrastructure. Performance and power simulation quickly becomes intractable as design space sizes increase. Further exacerbating these simulation costs, designers increasingly target differentiated market segments, each with particular metric priorities. Designers might implement different compromises between latency, throughput, power, and temperature depending on application and operating cost factors specific to each market segment. This increasing metric diversity may also lead to non-intuitive design optima that occupy very different regions of the design space. For example, IBM POWER5 implements relatively wide pipelines, INTEL CORE implements relatively deep pipelines, and SUN ULTRASPARC T1 implements relatively simple cores with in-order execution. This last example illustrates regions of the design space that become increasingly viable in the multiprocessor domain. Metric and design diversity, combined with a shift toward multiprocessors, drive exponential increases in design space size.

Software Domain. Tunable algorithmic parameters enable performance optimization as applications, such as numerical methods, are ported between computing platforms. For example, the optimal configuration of locality-exploiting optimizations (e.g., loop tiling) is highly dependant on memory hierarchy design. Similarly, the choice of communication algorithms used to coordinate computation will depend on the underlying interconnect topology. Analytical models that predict optimization effectiveness are increasingly difficult to formulate as system and algorithmic complexity obscure trends in the performance topology. Furthermore, analytical models often use simplifying assumptions about the target platform or application inputs. As a result, analytical models may capture high-level trends (i.e., performance bounds, scalability), but are less effective for producing accurate predictions of performance for any particular combination of tuning parameter values. Thus, algorithmic and tuning complexity, combined with microarchitectural design diversity, drive exponential increases in parameter space sizes for performance tuning.

Statistical Inference. Techniques in statistical inference are necessary for a scalable measurement methodology to address these fundamental challenges, modestly reducing detail for substantial gains in speed and tractability. This paradigm (1) specifies a large comprehensive parameter space, (2) selectively measures a modest number of sampled points from that space, and (3) more efficiently leverages measurements by deriving regression models to identify trends and optima. Spatial sampling decouples the number of measurements required to identify a trend from the size of a parameter space. Regression models provide computationally efficient predictions from these sparsely measured spaces.

Background and Related Work

Statistical Methodology. Eeckhout, *et. al.* , study statistical simulation for simplifying workloads in architectural simulation [Eeckhout 2003]. Workloads are profiled to construct smaller, synthetic benchmarks with statistically similar characteristics. Introducing sampling and statistics into simulation frameworks reduce accuracy for gains in speed and tractability. While Eeckhout suggests this trade-off for benchmarks input to simulators, we propose this trade-off for resulting simulator outputs to reduce the number of required simulations. Ipek, *et. al.* , predict the performance of microarchitectural designs using artificial neural networks with automated adaptive sampling based on network accuracy [Ipek 2006]. Similarly, Joseph, *et. al.* , construct neural networks with Latin hypercube sampling to increase sampling coverage [Joseph 2006]. In contrast, we predict both performance and power, enabling the application of predictive models to practical design optimization. Although our approach to model derivation requires greater statistical analysis, the resulting spline-based regression models may be more computationally efficient than neural networks [Lee 2007b].

Microarchitectural Design. This work presents a case study in identifying optimal cores for heterogeneous multiprocessors. In prior work, Kumar, *et. al.* studied this problem for a more modest design space, evaluating design alternatives with exhaustive, detailed simulation [Kumar 2006]. In homogeneous multiprocessor studies, Davis, *et. al.* , suggest in-order cores are performance optimal and Huh, *et. al.* , suggest larger out-of-order cores maximize throughput [Davis 2005, Huh 2001]. Each of these design spaces is relatively modest as experience and intuition were used to prune each space prior to simulation. However, such pruning may lead to conclusions that simply reinforce prior intuition and may not generalize to the broader space. In contrast, this work comprehensively considers a large design space, thereby enabling the discovery of unexpected optima.

Application Tuning. Carrington, *et. al.* , develop a framework for predicting scientific computing performance and demonstrate its application for numerical linear algebra (HPL) and an ocean modeling simulation [Carrington 2003]. Their automated approach relies on a convolution method that maps an application signature onto a machine profile. Kerbyson, *et. al.* , present an analytical model that accurately predicts performance and scaling characteristics of SAGE, a multidimensional hydrodynamics code with adaptive mesh refinement [Kerbyson 2001]. Their application-centric approach requires static code analysis with separate, detailed models for each target application. Pan, *et. al.* , apply linear regression and Vaswani, *et. al.* , survey non-linear regression techniques to assess the impact of tunable compiler flags on SPEC CPU benchmark performance [Pan 2004, Vaswani 2007]. In contrast, this work examines tunable algorithmic parameters for parallel scientific computing kernels.

Uniqueness of Approach

The proposed parameter space exploration paradigm (1) specifies a large, comprehensive parameter space, (2) selectively measures a modest number of points sampled from that space, and (3) more efficiently leverages measurements using techniques in statistical inference to identify trends and optima. This paradigm is applied to both hardware and software domains, demonstrating the broad effectiveness of non-linear regression for predicting microprocessor energy efficiency and parallel application performance. The proposed paradigm emphasizes integration of statistical analysis and domain-specific knowledge. The modeler specifies the model's functional form based on her prior knowledge of the significance and interaction of parameters. This enables the construction of models logically consistent with the modeler's knowledge of the domain and contrasts with more automated, black-box approaches such as artificial neural networks. The proposed model derivation process also emphasizes exploratory data analysis. Clustering, association, correlation, and significance analyses ensure efficiency (i.e., only significant parameters used) and robustness (i.e., unbiased predictions). The approach is summarized here and detailed in prior publications [Lee 2006, Lee 2007a, Lee 2007b].

Parameter Space. In the hardware domain, this work considers permutations of values for seven design parameters in the microprocessor design space (pipeline depth, superscalar width, register file, reservation stations, instruction cache, data cache, L2 cache) that constitute a space with 375,000 points. In the application domain, this work considers performance tuning for Semicoarsening Multigrid (SMG) and High-Performance Linpack (HPL), parallel numerical methods that solve discretized differential equations and dense linear systems, respectively. This high-resolution parameter space of more than 1 million points each is defined by permutations of six tunable SMG parameters (processor count, workload size in three dimensions) and six tunable HPL parameters (square matrix block size, row/column processor counts, four algorithmic options for LU decomposition).

Spatial Sampling. Samples are obtained uniformly at random for the parameter space definitions for measurement. 1,000 microprocessor design samples are simulated with Turandot, a cycle-accurate trace-driven simulator based on the IBM POWER4/5, to obtain performance and power estimates. 600 application configuration samples are timed when executing on clusters of the IBM Blue Gene/L (BGL) and the Intel Xeon (ALC, MCR from different vendors) physically located at Lawrence Livermore National Laboratory. The required number of samples is determined empirically based on resulting model accuracy.

Regression Modeling. Regression captures a response as a weighted sum of observed predictor values plus random error. Observations are obtained by sampling uniformly at random from the parameter space. Given these samples, regression weights are determined with the method of least squares.

:::: Predictor Interaction. In some cases, the effect of predictor *x1* on response *y* depends on the value of predictor *x2* and vice versa. The interaction between two predictors may be modeled by constructing a third predictor *x3 = x1 * x2*. Domain-specific knowledge is used to specify such interactions so the resulting model is consistent with modeler's prior knowledge of the space. For example, pipeline depth likely interacts with cache sizes. As the L2 cache size decreases, memory stalls per instruction will increase and instruction throughput gains from pipelining will be impacted.

:::: Non-linearity. Non-linear transformations may reduce error and capture non-linear effects. This work considers restricted cubic splines on the predictors in which the predictor domain is divided into intervals with endpoints called knots. Cubic polynomials are fit for each interval to obtain a piecewise cubic polynomial. Increasing knot count may improve fit at the risk of overfitting. Our models increase knot count with predictor significance, quantified by correlation analyses.

:::: Derivation. The statistically rigorous derivation applies association and correlation analyses before model specification to prune unnecessary, ineffective predictors thereby improving model efficiency. Specifically, we consider the following design process for regression modeling:

1. **Hierarchical Clustering:** Clustering examines correlations between potential predictors and enables elimination of redundant predictors. Predictor pruning controls model size, thereby reducing risk of over-fitting and improving model efficiency during formulation and prediction.
2. **Association Analysis:** Scatterplots qualitatively capture trends of predictor-response relationships, revealing the degree of non-monotonicity or non-linearity. Scatterplots with low response variation as predictor values change may suggest predictor insignificance, enabling further pruning.
3. **Correlation Analysis:** Correlation coefficients quantify the relative strength of predictor-response relationships observed in the scatterplots of association analysis. These coefficients impact our choice in non-linear transformations for each predictor.
4. **Model Specification:** Domain-specific knowledge is used to specify predictor interaction. The correlation analysis is used to specify the degree of flexibility in non-linear transformations. Predictors more highly correlated with the response will require more flexibility since any lack of fit for these predictors will impact overall model accuracy more. Given the model's functional form, least squares determines regression coefficients.
5. **Assessing Fit:** The R-squared statistic quantifies the fraction of response variance captured by the model's predictors. Larger R-squared suggests a better fit to training data. Normality and randomness assumptions for model residuals are validated using quantile-quantile plots and scatterplots to ensure a lack of bias. Lastly, predictive ability is assessed by predictions on randomly selected validation points.

Results and Contributions

Hardware Domain. Figure 1 presents boxplots of the error distributions from microprocessor performance and power predictions of 100 validation points. The validation set is obtained uniformly at random and separately from the training set. The three horizontal lines in the box indicate quartiles, vertical lines illustrate the spread, and circles indicate outliers. The performance model achieves median errors ranging from 3.7 percent (ammp) to 11.0 percent (mesa) with an overall median error across all benchmarks of 7.2 percent. Power models are slightly more accurate with median errors ranging from 3.5 percent (mcf) to 7 percent (gcc) and an overall median of 5.4 percent.

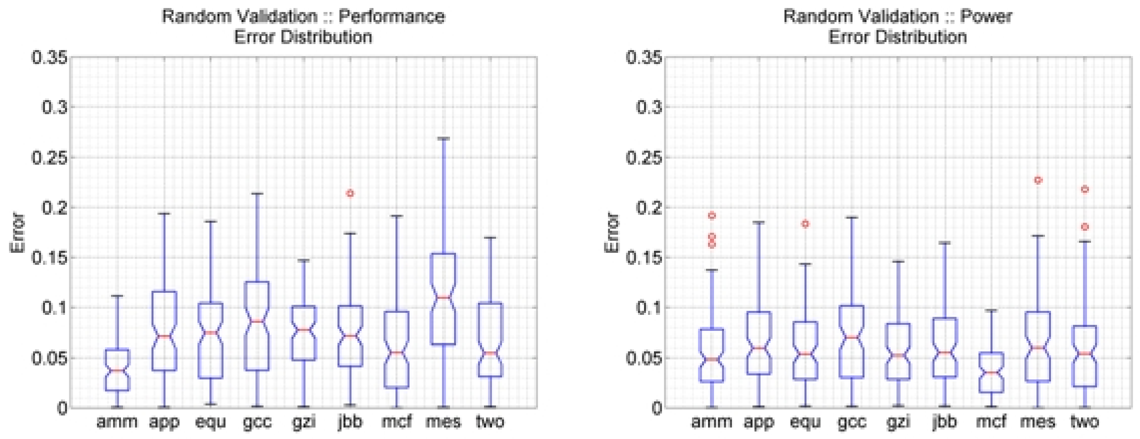


Fig 1 :: *Distribution of regression error for microprocessor (L) performance and (R) power predictions of 100 validation designs.*

In a case study, these models are used to identify optimal core compromises for heterogeneous multicore architectures. Regression models identify BIPS-cubed per watt maximizing architectures for each of nine SPEC benchmarks. This optimization, intractable via simulation for such a large design space, is enabled by regression. A K-means analysis clusters similar architectures, identifying core designs that best mitigate penalties of design compromises. Figure 2 plots the delay and power of the nine per benchmark optima executing their benchmarks. The four compromise designs are depicted as circles and their configurations are specified. Spatial locality between each centroid and its cluster elements suggest delay and power penalties from these architectural compromises are small.

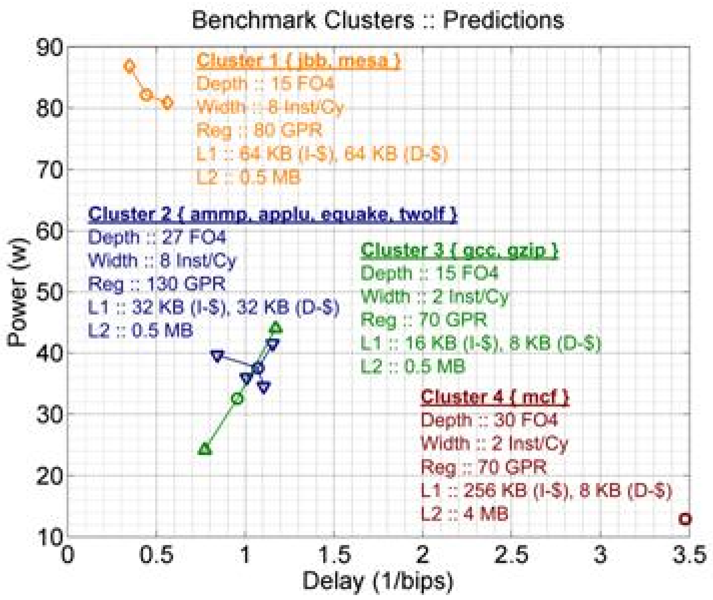
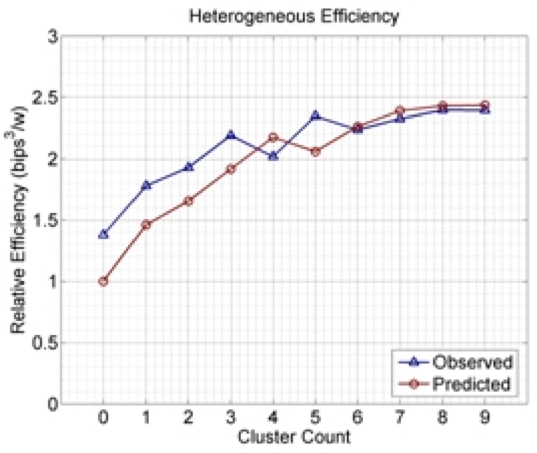


Fig 2 :: *Delay and power for per benchmark optima (radial points) and resulting K-means compromises (circles).*

Figure 3(L) plots predicted and simulated BIPS-cubed per watt efficiency gains for the benchmark average with increasing heterogeneity, quantified by cluster count in the K-means algorithm. Efficiency is presented relative to a POWER4-like baseline (cluster count 0). We observe diminishing marginal returns in heterogeneity beyond 4 cores. The 4 cores of Figure 2 are predicted to benefit efficiency by 2.2x, 8 percent less than the theoretical upper bound of 2.4x that is achievable only from the much greater heterogeneity of 7 to 9 cores. The benefits for 9 different cores is the theoretical upper bound on heterogeneity benefits as each benchmark executes on its optimal core.



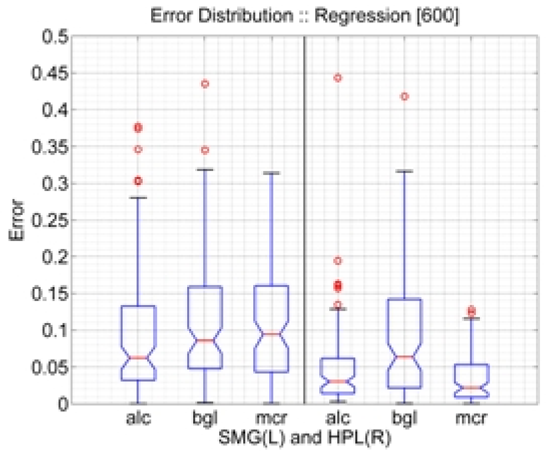


Fig 3 :: *(L) Efficiency trends from varying multicore heterogeneity and (R) distribution of regression error for performance of 100 validation designs.*

Software Domain. Figure 3(R) indicates median errors between 2.2 percent (hpl-mcr) and 9.4 percent (smg-mcr). 75 percent of predictions achieve error rates of 16 percent or less. HPL predictions (median errors of 2.2 to 5.2 percent) are more accurate than SMG predictions on the same platform (median errors of 6.3 to 9.4 percent). Outlier error not shown includes three predictions for SMG on ALC with errors of 56.7, 62.6, and 98.0 percent.

In a case study of performance prediction, Figure 4 illustrates the performance topologies predicted by regression models as significant predictors are varied. These topologies illustrate the marginal performance impact from changing the processor workload in the *z*-dimension (*nz*) for SMG and changing the block size (*nb*) for HPL. For example, SMG solve time increases with the processor workload in the *z*-dimension under limited processor resources (*pz*=1). Regression predicts greatest HPL solve time reductions from increasing block sizes when block size is small and processor count in the *x*-dimension is large. These predicted topologies may be particularly useful in optimization heuristics such as gradient descent.

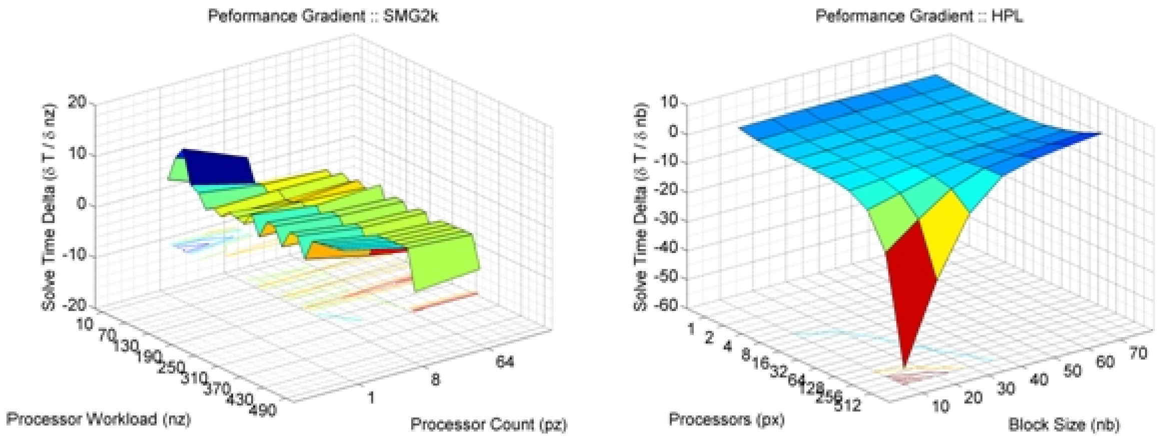


Fig 4 :: *Performance topologies for (L) Semicoarsening Multigrid and (R) High-Performance Linpack.*

Conclusion. Spatial sampling and statistical inference enables computationally efficient predictions for metrics of interest by controlling the number of measurements required to identify trends within large parameter spaces. This efficiency enables new capabilities in parameter space optimization not possible under more conventional approaches to measurement, such as heterogeneous multicore design and performance topology visualization.

Acknowledgements

Benjamin Lee is a doctoral candidate at Harvard University advised by Professor David Brooks. This work is supported by NSF grant CCF-0048313 (CAREER), Intel, and IBM. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation, Intel Corporation, or IBM Corporation.

References

Carrington, L., *et. al.* (2003): "A performance prediction framework for scientific applications." *Proceedings International Conference on Computational Science Workshop on Performance Modeling and Analysis*.

Davis, J., *et. al.* (2005): "Maximizing CMP throughput with mediocre cores." *Proceedings of International Conference on Parallel Architectures and Compilation Techniques*.

Eeckhout, L., *et. al.* (2003): "Statistical simulation: Adding efficiency to the computer designer's toolbox." *IEEE Micro*.

Huh, J., *et. al.* (2001): "Exploring the design space of future CMPs." *Proceedings of International Conference on Parallel Architectures and Compilation Techniques*.

Ipek, E., *et. al.* (2006): "Efficiently exploring architectural design spaces via predictive modeling." *Proceedings of International Conference on Architectural Support for Programming Languages and Operating Systems*.

Joseph, P., *et. al.* (2006): "A predictive performance model for superscalar processors." *Proceedings of International Symposium on Microarchitecture*.

Kerbyson, D., *et. al.* (2001): "Predictive performance and scalability modeling of a large-scale application." *Supercomputing* .

Kumar, R., *et. al.* (2006): "Core architecture optimization for heterogeneous chip multiprocessors." *Proceedings of International Conference on Parallel Architectures and Compilation Techniques*.

Lee, B., *et. al.* (2006): "Accurate and efficient regression modeling for microarchitectural performance and power prediction." *Proceedings of International Conference on Architectural Support for Programming Languages and Operating Systems*.

Lee, B., *et. al.* (2007a): "Illustrative design space studies with microarchitectural regression models." *Proceedings of International Symposium on High-Performance Computer Architecture.*.

Lee, B., *et. al.* (2007b): "Methods of inference and learning for performance modeling of parallel applications." *Proceedings of Symposium on Principles and Practice of Parallel Programming*.

Pan, Z. *et. al.* (2004): "Rating compiler optimizations for automatic performance tuning." *Supercomputing*.

Vaswani, K. *et. al.* (2007): "Microarchitecture sensitive empirical models for compiler optimizations." *Proceedings of International Symposium on Code Generation and Optimization*.