Design Space Characterization in Micro-Architecture Design and Implementation

John Brunhaver
Stanford University
jbrunhav@stanford.edu

Haowei Zhang
Stanford University
zhanghw@stanford.edu

***Abstract*—Modern VLSI designs contain both micro-architecture parameters and implementation parameters. These can be used to facilitate verification and relaxed design specifications. We concentrate on extending prior work in understanding design parameterization and using those design knobs to make global optimizations.**

This paper discusses the application of machine learning techniques to improve the efficiency and quality of the design space characterization and optimization. Specifically, we propose improvements to the circuit energy vs delay characterization.

I. Introduction

In this report we will discuss the contribution to NUMBERS made this quarter as a part of our CS229 project.

1) We built a framework to query the design parameters in order to map its full design space.
2) We used non linear least squares regression to characterize the energy delay trade-off of Circuits. This characterization reduced the design space parameters into a single pseudo-parameter representing a design trade-off in energy and delay.
3) We examined the training and test error rates required for effective fitting of theses functions.

The trend in micro-architecture design over the past decade has been to increase the number of parameters specifying the delay, composition, data structures, algorithms, and protocols of the Hardware Description Language (HDL) blocks. Much of this has been a function of increasing pressures from verification and late design specification. Combined with the large number of implementation, synthesis, and place-n-route parameters the design space is significant. The opportunity here is to capitalize on the design parameterization to provide optimized designs without requiring significant engineering effort.

This design space optimization problem [1] can be described abstractly as a three step problem. First, sample portions of the design space. Second, given these samples characterize the design parameter's affect on design cost and performance. Third, using this characterization and a set of constraints, optimize the design parameters for a given cost function. Unfortunately generating design space samples can be quite expensive.

For example generating one sample for a simple microprocessor execution core through the simulations and synthesis would be measured on the order of days. For a deeply parameterized design with a complicated feature space could take many months to brute force generate the cost, performance, parameter pairs required to build a low error analytic expression.

Our optimization framework extends prior work developing methodologies to guide the design of highly parameterized architectures [2]. Our goals are to both build design optimization into this methodology (GENESIS) and to improve the methodology through a better understanding of the requirements for optimization.

Our approach has been guided by prior work OPTIMACE [1]. One of our goals is to integrate many of the OPTIMACE strategies into GENESIS. Specifically, we adopt the approach of building analytic characterizations at the leaf nodes of the design hierarchy and then lifting these characterizations up into the circuit independent evaluations of the parent nodes.

In the case of our microprocessor execution core this would mean developing analytic expressions to describe floating point units, register files, caches, etc. Each of these smaller units may only take hours to build reasonable analytic models for their performance, cost, and parameters. Fast simulation also measured in hours can be used to build a cost performance model for the architecture independent of the circuit parameters. The two sets of models can be combined later to provide the full estimation required.

Our goal is to alter GENESIS to include a design space exploration component (NUMBERS). NUMBERS should start by integrating the OPTIMACE into the configurable design framework. NUMBERS will focus on providing the physical part of the co-design optimization problem. The architecture part exists as a later extension to this project.

NUMBERS should provide: circuit energy vs delay regression, the circuit energy and delay vs design parameters regression, and the architectural performance and costs regression. Additionally NUMBERS should provide a sampling function to facilitate the generation of new sample points for improved characterization or optimization speed and quality.

NUMBERS however must be design agnostic, and capable of building these models based on deep parameterizations without a full understanding of the underlying architectures. This is a departure from OPTIMACE, which focused on lightly parameterized microprocessors. For example, one of the target designs is an ASIC that implements the processing required for aperture radar.

In this report we will discuss the contribution to NUMBERS made this quarter as a part of our CS229 project. Additionally, we will discuss some future work, possible optimizations, and project goals as this report doubles as a milestone report for a three quarter project.

II. Prior Work

Current approaches to VLSI design are currently migrating from a set of fixed design choices towards flexible designs which encode designer intent [3]. The approach we take for our designs is to leverage the elaboration and generation engine GENESIS [2]. This provides a framework in which the design elaboration and parameterization is specified in Perl code mixed into the HDL. The generator evaluates these HDL designs and extracts XML configurations briefly describing the topology and parameters of the design.

Our goal has been to take this framework and incorporate both architectural optimization knowledge [4]–[6] and circuit optimization [7]–[9] to build joint architectural and circuit optimizers [1], [10].

O. Azizi created an integrated optimization framework that performs a co-exploration of architectural and circuit-level design spaces [1], [10]. In this framework, large architectural design spaces are modeled with statistically sampling and regression. The underlying circuits are characterized using stratified sampling and regression to create a circuit library. Then a joint architecture-circuit model is created by linking these two design spaces with a regression based on posynomial functions. Geometric programming is then utilized to evaluate the posynomials and minimize a given cost function. Using this framework, various architectures including single-issue in-order processor and quad-issue out-of-order processor have been compared. This comparison led those authors to suggest a strategy to achieve the high energy and area efficiency microprocessors [10].

Our work for CS229 sought to improve the circuit models from O. Azizi's work in order to deal with deeper design hierarchies and slightly more complicated design spaces.

While we did not have an opportunity to evaluate these machine learning approaches due to limitations in infrastructure and time, we believe that many of the machine learning approaches to combinatorial optimization may be useful in increasing the efficiency of our flows [11]–[14]. Additionally, we expect that it will be useful to evaluate the accuracy interval of many of estimates as function of the domain [15]. These intervals can be used both as a constraint in minimizing characterization error (sampling rates would dynamically vary with the complexity of the underlying design space) and in minimizing optimization error (recursive characterization would be required in regions where optima are expected to exist).

III. Numbers: Circuit Characterization Flow

In order to get a single design point representing the energy cost of a given circuit parameterization at a specific delay (each parameterization has its own continuous function describing a trade-off in delay and the energy required for that operation), we must run elaboration, simulation, synthesis, place, and route. We examine energy and delay pairs as these can be incorporated into the architectural model to find performance per watt and similar metrics used for system optimization. Elaboration accepts the parameters and produces and a fixed HDL design for those parameters. Simulation evaluates both correctness and performance in the HDL simulator while extracting the toggle rates for accurate energy estimation in later stages. Synthesis maps the HDL to a gate level netlist and sizes the gates to minimize an expected delay. Place and route determines the physical location of these gates and the wires that connect them (we did not run place and route as it nearly triples data collection time and disk space).

To build the characterization of a circuit for a given function we have a wrapper for the elaboration tools to extract the parameterization space from GENESIS. This is required as the design space has been encoded into the HDL but it has not be explicitly stated. Instead the design space exists as a set of parameter evaluations that must be executed in order to understand their extent.

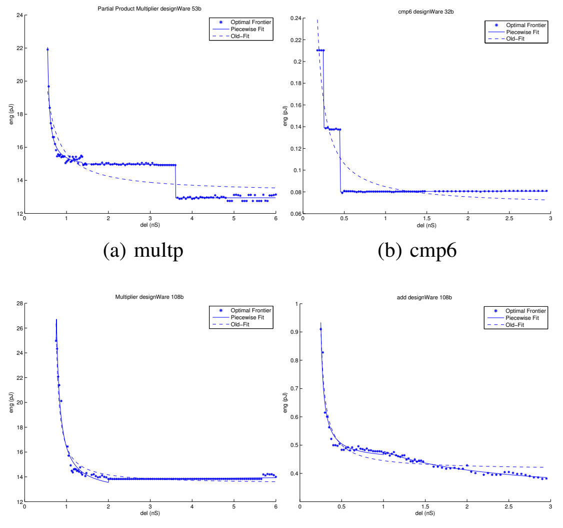
Given a set of energy delay pairs we need to preprocess the data to find the Pareto points. Unfortunately, because the tools produce noisy results we cannot reject all non-Pareto points. We should keep points that have a sufficiently high probably of being a design which exists on the true Pareto frontier but is simply perturbed by noise. We use a manually tuned relative threshold to nearby points in order to achieve this affect.

With the Pareto frontier we would like to create an analytic function for optimization and interpolation. Our basis function is a shifted and scaled hyperbola.

$$E^{(i)} = \frac{\theta_1}{\theta_2 - D^{(i)}} + \theta_3 \tag{1}$$

These can be grouped piecewise to describe discontinuous functions. To do this we fit K piecewise elements to the Pareto frontier where K is the number of piecewise elements that can be supported by the data. We felt that one piece for every 16 data points was reasonable, though this wasn't well explored. We split the domain into K probable pieces, including empty pieces, and fit single shifted and scaled hyperbola to that portion of the space. As least squares minimization of a non-linear function is non-convex, and subject to local optima, we initialize the parameters with estimates for the two asymptotes. Additionally we constrain the solution to make sure that the asymptotes occur to the left and below of the actual data (to guarantee that singularities do not occur in the domain) and that the scale factor is positive to guarantee the knee shape. This process is repeated to encompass all possible partitions.

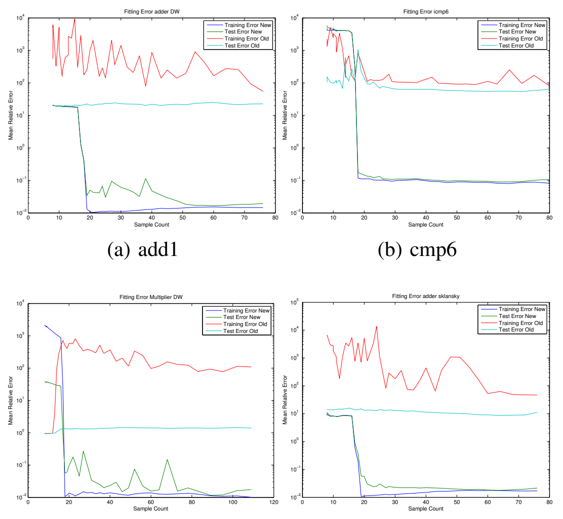
1) Generate energy delay pairs across parameter space.
2) Prefilter design points for probable Pareto designs
3) K Piecewise hyperbola regression
    a) For each possible partitions of the function domain
        i) Estimate initial parameters algebraically
        ii) Calculate parameter constraints



(a) multp (b) cmp6

(c) mult (d) add

Fig. 1: Energy vs Delay Pareto frontier fits for a 53 bit partial product multiplier, 32 bit six function comparator, a 53 bit multiplier, and a 108 bit adder. With the exception of the comparator these units generally compose a floating point multiply accumulate unit. New fits represent our method and old fits represent a fit using 1 piece.



(a) add1 (b) cmp6

(c) mult (d) add2

Fig. 2: Training Error and Testing Error for the one piece and a general K piece wise fit. The average relative error seen for 128 stratified random samplings consisting of the number of samples indicated on the x axis. Note that the data is fairly noisy and is only accurate to an order of magnitude.

iii) Fit each piece individually against its portion of the domain.

IV. Results

Figure 1 provides four examples of piecewise fits on a large number of data points and compares them to a fit using only one hyperbola. Visually the curves adhere well to the data. There are some unfortunate discontinuities in the curve fit that should be resolved in order to increase the utilities of these fits in later optimization stages that expect monotonically decreasing functions. Note that this is one of the better cases as there is a relatively large amount of data relative to the curves.

Figure 2 shows the training and test error for these points. Note that the extent of the x-axis represents that total number of points available. Each point was evaluated 128 times using a random stratified subsample of the Pareto frontier. The training error and test error results are quite noisy and seem to be accurate to about an order of magnitude. Both approaches should have about the same error around 8-16 samples as this will be the region that our algorithm uses one hyperbola to the data. We expected a similar drop around 32, but did not see one. We suspect that the fitting algorithm tends to favor only two hyperbolas. In any case it appears that about 20 points is required in order to achieve mean errors of 1%. In terms of automation this means that we can stop filling in the Pareto frontier after seeing about 20 well stratified points.

V. Future Circuit Characterization Work

There are a number of optimizations left that could be done to further optimize the characterization of circuit delay and energy.

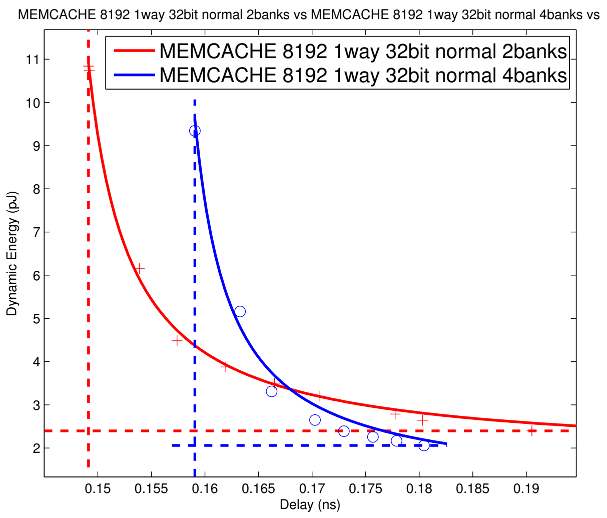
A better Pareto pre-filter could have a small but helpful impact on the quality of the results. With a greater number of points available for fitting the fewer total simulations are required. To improve the filter we suspect that a probabilistic filter based on a characterization of the design tool noise would be more robust than a simple relative threshold.

While the scaled and shifted hyperbola provides both the general shape we are looking for and the asymptotic energy and delay behavior, it isn't clear that it captures behavior like linearly increasing dynamic energy with delay and sharp knees. Providing other functions as a basis may be useful. For example:

$$E^{(i)} = \frac{\theta_1}{(\theta_2 - D^{(i)})^{\theta_4}} + \theta_3 \qquad (2)$$

The current algorithm for selecting the best separations in the piecewise function is often arbitrary using random initialization and iteration to converge to local minims. However these separations are, visibly, likely to occur near places where the second discrete derivative is high. This might provide more robust convergence and significantly reduce the runtime of the current algorithm.

Currently we utilize the squared error of the data. However, this isn't quite the error that we are actually attempting to minimize. We are interested in the maximum relative error to the nearest point on the fitted function. Additionally, we are more interested in the fitting that occurs at the extremes of the



functions, near the minimum delay, minimum energy, and the knee of the function.

Finally, we would like to include many of our insights here into fitting functional parameters that are exposed to upper levels of the hierarchy.

For structure such as caches and buffers, delay and energy also depend on the storage capacity. To capture this dependency in circuit library, a more complex fitting function is needed. This could be done by first starting with basic tradeoff curve, then parameterize each of the fit parameters with monomials.

Where the primed variables and various bs are new fit parameters. In this way, a new function is produced that defines the cost as the joint function of its size and delay.

Generally, creating fitted energy models as a joint function of size and delay can be harder to produce than with the simple energy-delay trade-offs since there are more data points that need to be simultaneously captured. To achieve acceptable accuracies in the cases of caches, ranges of sizes are often restricted to a limited set of values.

VI. Research Questions

- Is it possible to derive analytic expressions of higher levels of the hierarchy in terms of lower level characteristics of the design. (IE would it be effective to sample sparsely at high levels in the design and densely in lower levels of the design to build characterizations of).
- These new piecewise functions introduce a non-convexity into the original formulation. How do we deal with this in a way that doesn't require increased simulation or computation.

VII. Conclusion

Thank you for the CS229 experience. We learned a lot and feel like we have a good start on a bigger project.

For CS229 we:

1) We built a framework to query the design parameters in order to map its full design space.
2) We used non linear least squares regression to abstract implementation parameters from the energy delay Pareto frontier of leaf cells in the design hierarchy.
3) We examined the training and test error rates required for effective fitting of theses functions.

Acknowledgments

Thank you to Ofer Shacham and Omid Azizi for providing some great problems to work on and some great advice along the way

References

[1] O. Azizi, "Design and optimization of processors for energy efficiency: A joint architecture-circuit approach," Ph.D. dissertation, Stanford University, 2011.

[2] O. Shacham, "Chip multiprocessor generator: Automatic generation of custom and heterogeneous compute platforms," Ph.D. dissertation, Stanford University, 2011.

[3] O. Shacham, O. Azizi, M. Wachs, W. Qadeer, Z. Asgar, K. Kelley, J. Stevenson, S. Richardson, M. Horowitz, B. Lee, A. Solomatnikov, and A. Firoozshahian, "Rethinking digital design: Why design must change," *Micro, IEEE*, vol. 30, no. 6, pp. 9 –24, nov.-dec. 2010.

[4] T. Stanley and T. Mudge, "Systematic objective-driven computer architecture optimization," in *Advanced Research in VLSI, 1995. Proceedings., Sixteenth Conference on*, mar 1995, pp. 286 –300.

[5] E. İpek, S. A. McKee, R. Caruana, B. R. de Supinski, and M. Schulz, "Efficiently exploring architectural design spaces via predictive modeling," in *Proceedings of the 12th international conference on Architectural support for programming languages and operating systems*, ser. ASPLOS-XII. New York, NY, USA: ACM, 2006, pp. 195–206. [Online]. Available: http://doi.acm.org/10.1145/1168857.1168882

[6] B. Lee and D. Brooks, "Illustrative design space studies with microarchitectural regression models," in *High Performance Computer Architecture, 2007. HPCA 2007. IEEE 13th International Symposium on*, feb. 2007, pp. 340 –351.

[7] A. Hartstein and T. R. Puzak, "The optimum pipeline depth for a microprocessor," *SIGARCH Comput. Archit. News*, vol. 30, pp. 7–13, May 2002. [Online]. Available: http://doi.acm.org/10.1145/545214.545217

[8] ——, "The optimum pipeline depth considering both power and performance," *ACM Trans. Archit. Code Optim.*, vol. 1, pp. 369–388, December 2004. [Online]. Available: http://doi.acm.org/10.1145/1044823.1044824

[9] M. H. Dinesh Patil, "Joint supply, threshold voltage and sizing optimization for design of robust digital circuits," Tech. Rep.

[10] O. Azizi, A. Mahesri, S. J. Patel, and M. Horowitz, "Area-efficiency in cmp core design: co-optimization of microarchitecture and physical design," *SIGARCH Comput. Archit. News*, vol. 37, pp. 56–65, July 2009. [Online]. Available: http://doi.acm.org/10.1145/1577129.1577138

[11] J. A. Boyan and A. W. Moore, "Using prediction to improve combinatorial optimization search," in *In Proc. of 6th Int'l Workshop on Artificial Intelligence and Statistics*, 1997.

[12] ——, "Learning evaluation functions for global optimization and boolean satisfiability," in *In Proc. of 15th National Conf. on Artificial Intelligence (AAAI*. AAAI Press, 1998, pp. 3–10.

[13] ——, "Learning evaluation functions to improve local search."

[14] A. S. Weigend and D. A. Nix, "Predictions with confidence intervals (local error bars)," 1994.

[15] D. L. Shrestha and D. P. Solomatine, "2006 special issue: Machine learning approaches for estimation of prediction interval for the model output," *Neural Netw.*, vol. 19, pp. 225–235, March 2006. [Online]. Available: http://dx.doi.org/10.1016/j.neunet.2006.01.012