Fast Multi-Objective Algorithmic-Design Co-Exploration for FPGA-based Accelerators

Kumud Nepal, Onur Ulusel, R. Iris Bahar, and Sherief Reda
School of Engineering
Brown University, Providence, RI 02912
{*kumud_nepal,onur_ulusel,iris_bahar,sherief_reda*}*@brown.edu*

***Abstract*—The reconfigurability of Field Programmable Gate Arrays (FPGAs) makes them an attractive platform for accelerating algorithms. Accelerating a particular algorithm is a challenging task as the large number of possible algorithmic and hardware design parameters lead to different accelerator variant implementations, each with its own metrics such as performance, area, power, and arithmetic accuracy characteristics. To identify these parameters that optimize the accelerator for certain metrics, we propose techniques for fast design space exploration and non-linear multi-objective optimization (e.g., minimize power under arithmetic inaccuracy bounds). Our methodology samples a small part of the design space and uses measurements from the sampled implementations to train mathematical models for the different metrics. To automate and improve the model generation process, we propose the use of L_1-regularized least squares regression techniques. To demonstrate the effectiveness of our approach, we implement a high-throughput real-time accelerator for image debluring. We demonstrate the accuracy (e.g., within 8% for power modeling) of our modeling techniques and their ability to identify the optimal accelerator designs with large speed-ups (340×) in comparison to brute-force enumeration.**

I. Introduction

FPGA-based accelerators are enjoying ubiquitous use. One particularly important application is that of real-time image processing, which is used for surveillance, scientific research, camera technologies and automotive industries [1]. With this prolific use comes an increased demand for higher computational capabilities.

Many of these high-performance systems are also used in highly resource constrained environments, where reduced power consumption becomes imperative. Therefore, adding more hardware resources to solve the throughput problem may not lead to a feasible solution. We observe that image processing accelerators offer many algorithmic and hardware design parameters, which when properly chosen, can lead to outcomes with the desired power, design area and arithmetic accuracy. However selecting these parameters is not an easy task and requires evaluation of the whole design space.

Previous work on accelerating design space exploration mainly follows two different approaches: reducing the number of configurations to be evaluated or design space evaluation via modeling. The relationship between architectural parameters and the speed of FPGA implementations are explored in [2]. Jiang *et al.* use a least squares regression analysis to estimate the power and area consumptions of specific computation units of an implementation [3]. Their work is similar to the work done by Lee *et al.* in the computer architecture domain with regression based models for micro-architectural design space exploration [4]. Prior work has also been done on optimizing certain design metrics, such as throughput and power, after design exploration. Chen *et al.* minimize power dissipation for an FPGA implementation by doing careful allocation of functional units and registers [5]. Other related work by Sing *et al.* optimize the FPGA architecture for performance and power [6].

While all these prior approaches are effective in their own way, in this paper we aim to improve the optimization process by proposing new methodologies for architecture and hardware design co-exploration and optimization for accelerating algorithms in FPGA-based platforms. In particular, our contributions are as follows:

- We develop regression-based techniques to train mathematical models for various implementation metrics such as power, area, and arithmetic accuracy. We automate the process of identifying the best model for each metric by using L_1-regularized least squares to assess possible interactions between design variables.
- We perform multi-objective optimization formulations by leveraging the best models developed from L_1 regularization to show different important design optimizations, such as minimizing power consumption under maximum arithmetic error tolerance.
- We develop an actual FPGA-based accelerator design for deblurring images acquired from unmanned aerial vehicles. We analyze the effectiveness of algorithmic and hardware-level design choices to train our models.
- We sample a number of implementation variants for the accelerators and use real measurements to train and evaluate our regression-based models for different metrics. We show that our best models predict these metrics within 90% of measured values, while achieving 340× speedup over brute-force design space exploration. We then use these models within our proposed numerical optimization framework to optimize the accelerator using a number of objectives and constraint scenarios.

II. Modeling and Optimization Methodology

Designers are interested in exploring tens of algorithmic and hardware design parameters without doing explicit enumeration. This lets them identify the optimal values for these design parameters that meet the target metrics such as performance, power, area, and arithmetic accuracy targets.

To speed up design exploration, we propose an approach similar to [3], [7], where the large design space is sampled and then regression analysis and statistical inference are used to create mathematical models that estimate the target metrics over the entire design space. These sample combinations are implemented in the design and the resultant metrics characterized from real measurements (e.g., power) and/or from synthesis tool results (e.g., area). The characterized results are then used as a training set to generate the scalable models.

To train the models and obtain variable coefficients, the mathematical models are fit to the measured samples using least squares estimation. Note that while the model could be non-linear in its variables, it is linear in its coefficients. If $\mathbf{X}$ is the design matrix and $\mathbf{y}$ is the set of observations, then the model coefficients, $\hat{\mathbf{c}}$, that minimize the total squared error, i.e., $||\mathbf{y} - \mathbf{X}\mathbf{c}||_2$, is given by

$$\hat{\mathbf{c}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y},$$

where $\mathbf{X}^T$ is the transpose of $\mathbf{X}$. While this regression technique has been used by previous works in design exploration, we identified a number of shortcomings:

- If there are interactions between two or more variables, a model guessed from merely the individual relationships between isolated design parameters and the design metric would not be accurate.
- To capture the interactions between different algorithm and design parameters, the designers might need to make educated guesses on the interactions between the variables to identify the appropriate terms in the model. While educated guesses are usually guided by design of experiment methods, they can still introduce error.

To address this limitation and to automate the process of identifying the closest model to the measurements, we propose the use of L_1*-norm based regularization*. In this case, we start with a model that captures all possible interactions between the algorithm-design parameters. Then we can solve for $\hat{\mathbf{c}}$ by minimizing

$$||\mathbf{y} - \mathbf{X}\mathbf{c}||_2^2 + \lambda||\mathbf{c}||_1,$$

where $||\mathbf{y}-\mathbf{X}\mathbf{c}||_2^2$ is the total squared error, and $||\mathbf{c}||_1$ represents the L_1-norm, or the summation of the absolute value of all coefficients of $\mathbf{c}$, i.e., $\sum_{i=1}^{n}|c_i|$. The minimization of the L_1 norm of $\mathbf{c}$ attempts to *sparsify* the coefficients $\mathbf{c}$ [8]. Coefficients that get relatively small numerical values indicate interaction terms that are not important towards estimating the target metrics. By suppressing interactions or features that are irrelevant to the model during training, we avoid the problem of overfitting the model.

By choosing a regularization parameter λ which minimizes the error for a design objective after cross validation between a predicted and measured set of outputs, the unwarranted complexity of the design matrix that stems from overfitting irregularities can be reduced.

Once the best model representing the design objective is obtained, we will be able to do the following:

- Plug in different design parameters to estimate design metrics, with each design metric (e.g., power, area, arithmetic accuracy) having its own model.
- Incorporate into non-linear optimization formulations with an objective and under one or more model constraint(s). If a designer is studying two metrics, say y_A and y_B, these formulations will enable him/her to carefully design an architecture with focus on one or the other design variable, making it more efficient in the direction of either objective.

Multi-objective optimization problems mentioned above can be solved using standard non-linear optimizing techniques as presented in [9] . With an objective function for y_A (e.g, power) to be optimized under a constraint function y_B (e.g., arithmetic inaccuracy) bounded by constant M, our optimization could look like

$$\text{minimize } y_A(\mathbf{x}) \text{ subject to } y_B(\mathbf{x}) \leq M, \mathbf{x} \in \mathbb{R}^n$$

We solve this optimization formulation using *interior point* algorithms.

III. Image Processing Application

To evaluate our methodology we consider a case study for image deblurring acceleration. Image deblurring is performed by a filtering operation over the image, which is one of the fundamental operations of image processing applications. The accelerator is deployed within a real-life image processing system mounted on an unmanned air vehicle system for surveillance. The real-life setting of the accelerator has put strenuous requirements on its throughput, power, area, and arithmetic accuracy which motivated the need for our proposed modeling and multi-objective optimization methodology.

The filtering operation is performed as

$$I_D(i,j) = \sum_k \sum_l I_0(i+k, j+l)H(k,l),$$

where $I_D(i,j)$ and $I_0(i,j)$ are the deblurred and original pixel intensities at coordinates (i,j) and $H(k,l)$ is the deblur filter value at index (k,l). Our implementation uses input (I_0) and output (I_D) images with 12-bit pixels and deblur filters of varied sizes and fixed-point bit-widths. We will refer to the deblur filter H as the *kernel* from hereon.

The block diagram of the deblurring hardware is given in Fig. 1. Dedicated DSP units on the FPGA are used

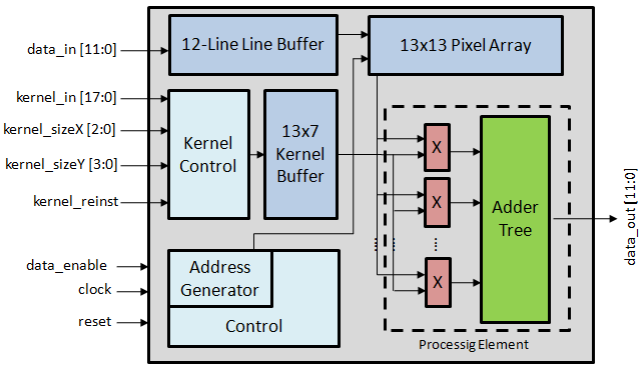

Figure 1. Top-level block diagram for deblur architecture.

as processing elements (PEs). This architecture deblurs 8 pixels/cycle running at 125 MHz on our FPGA board. Some design and algorithmic parameters present in this architecture are as follows:

1. Kernel Bit-Width (algorithm parameter). Different bit-width selections do not have any effect on the area and throughput of the design due to the fixed width allocated for DSP inputs; however both power and accuracy of the design varies with different bit-widths.

2. Kernel Size (algorithm parameter). The kernel size used for filtering is dynamically configured and the DSP blocks not being used for smaller kernels are clock gated to reduce power. Thus, both accuracy and power of our design vary with changing kernel size but area will remain unaffected.

3. DSP Pipeline Depth (design parameter). As the depth of each DSP pipeline decreases, the number of required DSP groups increase to perform the same number of computations. The smaller pipeline depths require fewer delay registers for synchronization but use extra DSP slices for the addition of computed partial sums. We use the average DSP pipeline depth as a design variable.

4. Time-Division Multiplexing (design parameter). Time-division multiplexing enables the DSP blocks to run at a frequency potentially faster than the rest of the FPGA system [10].

IV. Experimental Results

Our experimental prototype system uses a 40 nm Xilinx XC6VLX240T FPGA with 240,000 logic elements and 768 DSP blocks. Xilinx ISE Design Suite 12.4 is used for physical synthesis and Mentor Graphics Modelsim 10.0c is used for functional and timing simulations of the design. MATLAB is used for regression and optimization. To evaluate our accelerator performance, we use a number of sample images that are captured from the aerial vehicle platform. As for design *metrics*, we measure area in terms of number of DSP blocks, arithmetic accuracy in terms of the *mean square error* (MSE) between sample image data and deblurred output, and power in terms of *incremental power consumption* of the prototype board. The incremental power is the difference between the reset state power and the execution state power of the design. This approach accounts for the additional system (e.g., FPGA and memory) power that is associated with the computations of our accelerator.

We use the four design choices discussed earlier as parameters of our algorithm-design space. For time-division multiplexing, we use factors of 1, 2, and 4 which correspond to a DSP clock frequency of 125, 250 and 500 MHz in our case. We take four different choices of average DSP pipeline depths between 3.3 and 11.5. These depths are calculated by dividing the total number of DSPs used in all pipeline blocks by the number of blocks used. For kernel bit-width, we vary the parameter from 8 bits to 18 bits. We also pick four random kernel sizes between 5×3 and 13×7. The combinations of parameters create a design space with $3 \times 8 \times 11 \times 45 = 11{,}880$ possible design points that potentially lead to different accelerator variants.

Full physical synthesis (which includes placement and routing) of an accelerator variant takes about two hours on our quad-core based system, which puts limitations on the ability to execute a brute-force exploration of all accelerator variants. This motivates the need for fast design space exploration and optimization. To obtain our samples, we fully synthesize and implement 50 accelerator variants with different parameter permutations; i.e., we only sample $\frac{50}{11880} = 0.42\%$ of the entire design space.

A. Modeling Results

We first analyze the closeness between the results of different regression models for power, area, and accuracy against the measurements we obtained from our samples. We compared four generic models (linear, linear with interactions, pure quadratic and quadratic with interactions) and one optimized model based on L_1 regularization for each metric. We separate our measurements into a *training subset* and *query subset* and verify the performance of our model by doing random cross-validation 1000 times. The estimation accuracy of the different models is given in Fig. 2 for a training size of 36, 9, 23 samples for power, area, and accuracy models. The results show that our L_1 based model outperforms other models and is able to achieve estimation errors of 7.48%, 2.38% and 9.22% for power, area, and accuracy respectively. This estimation also achieves approximately 340× speedup compared to brute-force design space exploration.

B. Multi-Objective Optimization Results

We consider two optimization formulations:

1. Minimizing Power Under Accuracy Constraints. We set up the objective function to be equal to the mathematical model for power obtained from L_1 regularization, and set a constraint function based on the mathematical model of

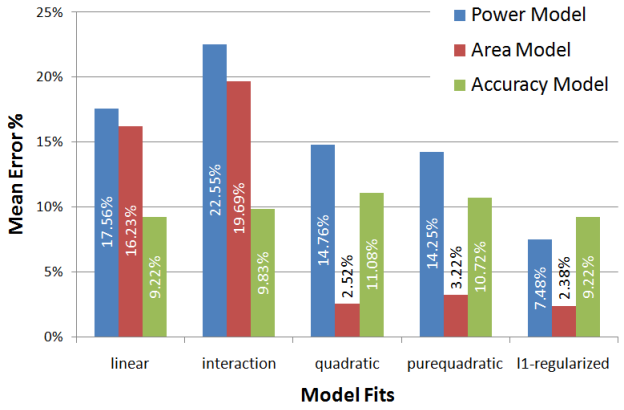

Figure 2. Comparison of mean error percentage using different model fits for power estimation, area and arithmetic accuracy models.

arithmetic accuracy. We experiment with setting different accuracy values. For each constraint value, we solve the numerical optimization problem as discussed in Section II using MATLAB. The results from our experiments are given in Fig. 3. We label solution points with the identified design parameters. The results from the numerical optimization are intuitive as they show that relaxing the accuracy constraint leads to reduced power dissipation. While it is impossible to verify the optimality of these implementations without brute-force exploration, we show high *fidelity* of our optimization results by implementing the designs with the identified optimal parameters and evaluating their actual measurements. The dotted red line in Fig. 3 gives the results from the actual implementation.

2. Minimizing Area within a Power Budget. In this second formulation we use the mathematical model for the design area (as measured by number of DSPs) as an objective function, and use the power function as a constraint. We use different numerical values over a range for the power constraint. The results from numerical optimization show the trend that *minimum* number of DSPs reduces as we relax the allowed power threshold. Results from optimization formulations show a high-fidelity trend with measured results.

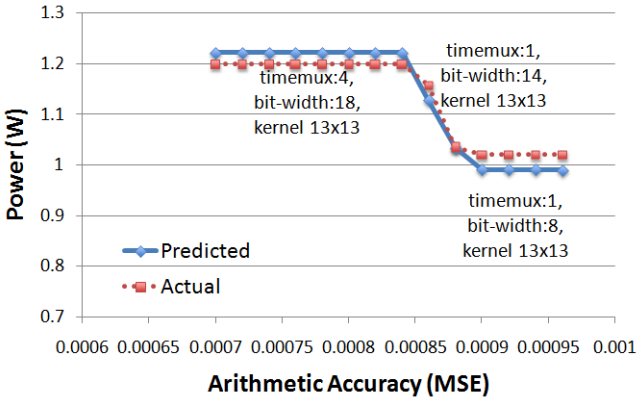

Figure 3. Trade-off between power and accuracy of the system.

V. Conclusions

In this paper we explored techniques for fast design space exploration and multi-objective design optimization for accelerators implemented in FPGAs. We formulated scalable models using fast regression techniques that predict various design metrics such as power, arithmetic accuracy, performance, and area. We proposed automatic techniques to devise the best model using L_1-regularized least squares estimation. For the given image deblurring accelerator design, the proposed models predict the implementation metrics within 8% of measured power values, 10% within the output arithmetic accuracy, and within 3% of actual FPGA resources used. With these accurate models in hand, we are able to come up with numerical optimization formulations that give directly the optimal design parameters under various objectives and constraints. These multi-objective optimizations help the designers identify the parameters of the design space that would give leverage to efficient trade-off between design metrics.

VI. Acknowledgments

This work was supported in part by a grant from Object Video through DARPA grant number W31P4Q-10-C-0139 and an equipment grant from Xilinx.

References

[1] S. S. B. Brainslav KisaCanin and S. Chai, *Embedded Computer Vision*. London: Springer, 2009.

[2] J. Das, S. Wilton, P. Leong, and W. Luk, "Modeling post-techmapping and post-clustering fpga circuit depth," in *Field Programmable Logic and Applications, 2009. FPL 2009. International Conference on*, pp. 205 –211, 31 2009-sept. 2 2009.

[3] T. Jiang, X. Tang, and P. Banerjee, "Macro-models for high level area and power estimation on fpgas," in *Proceedings of the 14th ACM Great Lakes symposium on VLSI*, GLSVLSI '04, (New York, NY, USA), pp. 162–165, ACM, 2004.

[4] B. Lee and D. Brooks, "Roughness of microarchitectural design topologies and its implications for optimization," in *High Performance Computer Architecture, 2008. HPCA 2008. IEEE 14th International Symposium on*, pp. 240 –251, February 2008.

[5] D. Chen, J. Cong, Y. Fan, and Z. Zhang, "High-level power estimation and low-power design space exploration for fpgas," in *Design Automation Conference, 2007. ASP-DAC '07. Asia and South Pacific*, pp. 529 –534, January 2007.

[6] L. C. Sing and H. Yajun, "Design space exploration for arbitrary fpga architectures," in *Embedded Software and Systems, 2005. Second International Conference on*, p. 7 pp., December 2005.

[7] B. C. Lee and D. M. Brooks, "Accurate and efficient regression modeling for microarchitectural performance and power prediction," *SIGOPS Oper. Syst. Rev.*, vol. 40, pp. 185–194, October 2006.

[8] S. J. Kim, K. Koh, M. Lustig, S. Boyd, and D. Gorinevsky, "An interior-point method for large-scale l1-regularized logistic regression," *Journal of Machine Learning Research*, vol. 2007, 2007.

[9] R. H. Byrd, J. C. Gilbert, and J. Nocedal, "A trust region method based on interior point techniques for nonlinear programming," *MATHEMATICAL PROGRAMMING*, vol. 89, pp. 149–185, 1996.

[10] Xilinx, *ML605 Hardware User Guide*, 2011.