CAD Tool Design Space Exploration via Bayesian Optimization

Yuzhe Ma
Chinese University of Hong Kong
yzma@cse.cuhk.edu.hk

Ziyang Yu
University of Hong Kong
yzy1996@connect.hku.hk

Bei Yu
Chinese University of Hong Kong
byu@cse.cuhk.edu.hk

***Abstract*—The design complexity is increasing as the technology node keeps scaling down. As a result, the electronic design automation (EDA) tools also become more and more complex. There are lots of parameters involved in EDA tools, which results in a huge design space. What's worse, the runtime cost of the EDA flow also goes up as the complexity increases, thus exhaustive exploration is prohibitive for modern designs. Therefore, an efficient design space exploration methodology is of great importance in advanced designs. In this paper we target at an automatic flow for reducing manual tuning efforts to achieve high quality circuits synthesis outcomes. It is based on Bayesian optimization which is a promising technique for optimizing black-box functions that are expensive to evaluate. Gaussian process regression is leveraged as the surrogate model in Bayesian optimization framework. In this work, we use 64-bit prefix adder design as a case study. We demonstrate that the Bayesian optimization is efficient and effective for performing design space exploration on EDA tool parameters, which has great potential for accelerating the design flow in advanced technology nodes.**

I. Introduction

Electronic design automation (EDA) tools play a vital role in pushing forward the VLSI industry. Nowadays, the design complexity keeps increasing as the technology node scales down. As a result, EDA tools correspondingly become more and more complex since more sophisticated algorithms and optimizations are incorporated to ensure timing closure, reliability and manufacturability. Essentially, the increasing complexity corresponds to the expanding amount of parameters involved in EDA tools, which implies a huge design space and requires rich expertise from the designers to achieve desired quality. Due to the underlying complicated optimization process, it is commonly seen that a subtle change of constraints would lead to large variations in final design performances, which makes designers could not rely on the intuition to explore the design space. What's worse, the runtime cost of synthesis flow also goes up as the complexity increases, and the exhaustive exploration is prohibitive for modern designs. Therefore, an efficient design space exploration methodology is of great importance in advanced designs.

Take the synthesis and physical design flow as an example. Given a specific design and a set of CAD tool scripts, parameterized by a vector representation $\boldsymbol{x}$, the synthesis flow is performed and the core metrics, including area, power, and delay, can be obtained. The aim is to find the most suitable parameters in CAD tool scripts that result in the best circuit performance after synthesis. Intuitively, it can be formulated as an optimization problem. However, the synthesis process is too sophisticated to be analytically modeled. Also, the solution space is too large. Therefore, designers can neither derive a closed-form solution, nor exhaustively search the solution space. Instead, the optimization should be conducted in an exploration manner in the design space.

978-1-7281-5758-0/19/$31.00

Data-driven approaches like machine learning and deep learning have been heavily applied in EDA field, including testability analysis, physical design, mask optimization and so on [1]–[3]. Besides these typical stages in design flow, there is a rich literature investigating how to perform design space exploration effectively and efficiently with learning-based methods. Linear regression and artificial neural network are adopted for multiprocessor systems-on-chip design [4]. Random forest is applied in high level synthesis (HLS) [5], [6]. Roy *et al*. propose to explore power efficient adders by leveraging support vector machine (SVM) to predict the power-performance-area (PPA) values [7]. Beyond that, a more efficient design space exploration flow for high performance adders is introduced in [8], which is based on an active learning flow with Gaussian process regression as the surrogate model.

Bayesian optimization is a promising technique for optimizing black-box functions that are expensive to evaluate, which promises significant automation such that both design quality and human productivity can be greatly improved. It has been applied in designing various sorts of circuits. A DNN accelerator is designed using Bayesian optimization for design parameters search [9]. Recently, Bayesian optimization has been heavily applied in analog circuits design as well [10]–[12], in which several approaches are introduced to improve the efficiency. Specifically, Lyu *et al*. propose to compose a multi-objective acquisition function to enhance the efficiency of the sampling step [10]. In [11], neural networks are leveraged to generate the feature representation explicitly, which reduce the prediction time complexity from $\mathcal{O}(n^2)$ of traditional kernel function-based computation to $\mathcal{O}(1)$.

In this paper we target at an automatic flow for reducing manual tuning efforts to achieve high quality circuits synthesis outcomes. We use 64-bit adder design as a case study. Fig. 1 demonstrates two significantly different views of the same implementation of a regular 64-bit Sklansky adder,

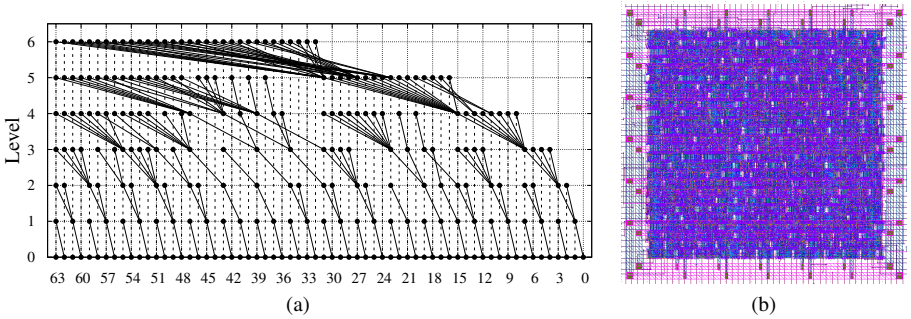

Fig. 1 (a) An example of architectural solution; (b) Corresponding physical solution.

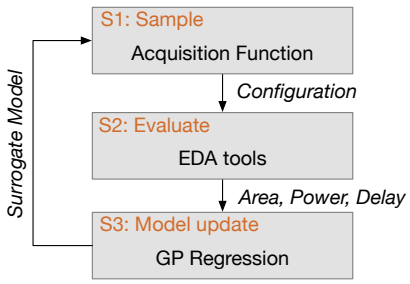

Fig. 2 Diagram of the proposed workflow.

in which Fig. 1(a) is the prefix structure in the front-end, and Fig. 1(b) is the result after this design goes through the back-end design process. Given a prefix architectural design, how to obtain efficiently a result of high quality after back-end design is a critical problem. There exist two main challenges. On one hand, it is observed that the final performance after logical synthesis and physical synthesis could vary significantly under subtle change of sensitive constraints. On the other hand, the design space is huge and it grows exponentially as more constraints are taken into consideration, which makes an efficient flow in high demand. To tackle these issues, we develop a Bayesian optimization-based approach for efficient exploration in the huge design space. Gaussian process regression (GPR) is leveraged as the surrogate model in our Bayesian optimization framework, which captures the correlation among the training samples. Based on the surrogate model, different acquisition functions are investigated to guide the exploration process to find the superior points. Various regular adder structures including legacy regular structures, synthesized structures and structures from commercial IP library are used for experimental evaluation. We demonstrate the Bayesian optimization is efficient and effective for performing design space exploration on CAD tool parameters, which has great potential for accelerating the design flow in advanced technology nodes.

The rest of this paper is organized as follows. Preliminary knowledge on Bayesian optimization and tool settings are introduced in Section II. Section III depicts the proposed design flow using Bayesian optimization. Experimental evaluations are presented in Section IV, and Section V concludes the paper.

II. Preliminaries

A. Bayesian Optimization

Essentially, Bayesian optimization is a statistical framework to optimize an arbitrary objective function. It is able to take advantage of the full information gained from past experiments to make the search much more efficient than human tuning. The core components of Bayesian optimization consist of a probabilistic surrogate model which is cheap to evaluate and provides our beliefs about the black-box function $f(\cdot)$, and an acquisition function which is designed to select the most informative one to query the label, such that the optimizer can be attained with a minimal number of expensive evaluations on the objective functions. After observing the output $f(\boldsymbol{x})$ of each new set of parameters $\boldsymbol{x}$ in each iteration, the surrogate model is updated such that it can better model the posterior distribution over the space of the objective function.

B. Gaussian Process Regression

The surrogate model plays a vital role in the Bayesian optimization framework. Gaussian Process (GP) model is typically adopted in building the surrogate models thanks to its capability to capture the uncertainty of the prediction. A Gaussian process is specified by its mean function and covariance function. Conventionally, the training process selects the parameters in the presence of training data such that the marginal likelihood is maximized. Then the Gaussian Process model can be obtained and the regression can be proceeded with supervised input [13]. A GP learner provides a Gaussian distribution $\mathcal{N}$ of the values predicted for any test input $\boldsymbol{x}$ by computing

$$\begin{aligned} m(\boldsymbol{x}) &= k(\boldsymbol{x},\mathbf{X})^{\top}(k(\mathbf{X},\mathbf{X})+\sigma_n^2\mathbf{I})^{-1}\mathbf{Y}, \\ \sigma^2(\boldsymbol{x}) &= k(\boldsymbol{x},\boldsymbol{x})-k(\boldsymbol{x},\mathbf{X})^{\top}(k(\mathbf{X},\mathbf{X})+\sigma_n^2\mathbf{I})^{-1}k(\boldsymbol{x},\mathbf{X}), \end{aligned} \tag{1}$$

where $\mathbf{X}$ is the training set, $\mathbf{Y}$ is the supervised information of trained set $\mathbf{X}$. In the case of CAD tool design space exploration, a prediction of a design objective consists of a mean and a variance from Gaussian Process regression. The mean value $m(\boldsymbol{x})$ represents the predicted value and the variance $\sigma(\boldsymbol{x})$ represents the uncertainty of the prediction. To make sure the model is close to the actual situation, additive noise ε is introduced on the black-box function value z. It is assumed that the noise is Gaussian distributed: $\varepsilon \sim \mathcal{N}(0,\sigma_n^2)$. The noisy observed value is: $y = z + \varepsilon$. k is covariance function characterizes the correlations between different samples in the process.

C. Tool Settings

The parameters in design space is depicted in TABLE I, including both the logical synthesis (LS) and the physical

TABLE I Design Parameters in CAD Tools

Parameter	Min.	Max.	Type	Stage
max_delay	0.1	0.5	float	LS
clock_period	1.0	2.0	float	LS
pin_load	0.002	0.006	float	LS
output_delay	0.1	0.5	float	LS
core_utilization	0.5	1.0	float	PD
core_aspect_ratio	1	3	integer	PD

design (PD) stages. The data types are not fixed and can be either floating value or integer value, depending on the specific parameters. These parameters essentially set constraints on the area, power and timing during the synthesis process, thus have great impacts on the ultimate performances.

D. Problem Definition

The design space exploration for CAD tools can be treated as an optimization problem. The objective is to minimizing the performance-power-area (PPA) values of a design, subject to that a set design constraints is satisfied. The variables of the problem are also constrained by a bounded set which indicates reasonable ranges of the tool parameters.

III. Methodology and Design Flow with Bayesian Optimization

A. Design Flow

Fig. 2 depicts the proposed iterative optimization flow. Each pass consists of a 3–step process and finally produces one sample point. In each iteration, it begins with solving the acquisition function, which returns a set of parameters. Since acquisition function is to guide the choice of the next most potential candidate sample, the choice of it depends on the specific case. A commonly used acquisition function is upper confidence bound (UCB) [14], which is defined as

$$\mathrm{UCB}(\boldsymbol{x}) = m(\boldsymbol{x}) + \kappa\sigma(\boldsymbol{x}), \tag{2}$$

where $m(\boldsymbol{x})$ and $\sigma(\boldsymbol{x})$ are the predictive value and uncertainty of GP defined as Equation (1), respectively. κ is a parameter that balances the exploitation and exploration. UCB can be considered as the weighted sum of the predictive value and uncertainty. When the weight κ is small, UCB tends to help select sample with low expected value. While with a large κ, this acquisition function is more likely to select samples with large uncertainty.

In the case of CAD tool design space exploration, the final objective functions are to be minimized, which is in conflict with UCB. Instead, lower confidence bound (LCB) function is leveraged as the acquisition function, which is defined as follows:

$$\mathrm{LCB}(\boldsymbol{x}) = m(\boldsymbol{x}) - \kappa\sigma(\boldsymbol{x}). \tag{3}$$

Besides LCB, several other acquisition functions are also commonly used. Probability of improvement (POI) evaluates the objective function f and finds the sample which is the most likely to attain improvements upon the best observed value f^*:

$$\begin{aligned}\mathrm{POI}(\boldsymbol{x}) &= \mathrm{P}(f(\boldsymbol{x}) \le f^* - \zeta) \\ &= \Phi(\frac{m(\boldsymbol{x}) - f^* + \zeta}{\sigma(\boldsymbol{x})}).\end{aligned} \tag{4}$$

Here Φ is the normal cumulative distribution function (CDF). ζ is a small trade-off parameter. Searching with POI ignores the margin of improvement and only selects those samples with higher probability of improving even if the improvement is smaller, which is prone to stucking at local optima.

An alternative acquisition function is expected improvement (EI). EI incorporates the margin of improvement by maximizing the expectation of the improvement. The improvement compared to the best observed sample can be written as:

$$\mathrm{I}(\boldsymbol{x}) = \max(0, f^* - f(\boldsymbol{x}) - \zeta). \tag{5}$$

Assuming the value of objective function $f(\boldsymbol{x})$ is normally distributed, $y \sim \mathcal{N}(m(\boldsymbol{x}), \sigma^2(\boldsymbol{x}))$, EI could be written as below:

$$\begin{aligned}\mathrm{EI}(\boldsymbol{x}) &= \mathrm{E}(\mathrm{I}(\boldsymbol{x})) \\ &= (f^* - m(\boldsymbol{x}) - \zeta)\Phi(\frac{f^* - m(\boldsymbol{x}) - \zeta}{\sigma(\boldsymbol{x})}) \\ &+ \sigma(\boldsymbol{x})\phi(\frac{f^* - m(\boldsymbol{x}) - \zeta}{\sigma(\boldsymbol{x})}).\end{aligned} \tag{6}$$

Here ϕ is the standard normal probability distribution function (PDF). ζ is the parameter set to trade off between exploration and exploitation.

After obtaining a set of parameters, the second step is to conduct the synthesis flow to obtain the corresponding performance of the target design under selected parameters. The synthesis flow involves logic synthesis and physical design, which makes this step the most time-consuming among all three steps. Typically, EDA tools are launched using Tcl scripts. Therefore, the selected parameters need to be translated into a representation so that the EDA tools can interpret. To do so, a Tcl script generator is built to provide such scripts based on the parameters.

The last step in each iteration is to update GP regression model using the evaluation result, i.e., area, power and delay values, which is GP model training. Assuming n targeted values $\boldsymbol{y}$ are already evaluated from the training set $\mathbf{X}$. The joint distribution of the $\boldsymbol{y} \in \mathbb{R}^n$ and the test sample value z can be written below:

$$\begin{bmatrix}\boldsymbol{y} \\ z\end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix}k(\mathbf{X},\mathbf{X}) + \sigma_n^2\mathbf{I} & k(\mathbf{X},\boldsymbol{x}) \\ k(\boldsymbol{x},\mathbf{X}) & k(\boldsymbol{x},\boldsymbol{x})\end{bmatrix}\right). \tag{7}$$

The mean function $m(\boldsymbol{x})$ and covariance function $\sigma^2(\boldsymbol{x})$ in Equation (1) can be derived from the above equation.

We optimize the GP regression model by maximizing the marginal likelihood $p(\boldsymbol{y}|\mathbf{X}, \boldsymbol{\mu})$. Here $\boldsymbol{\mu}$ represents the vector of all parameters contained in the model. This marginal likelihood is the marginalization over the black-box function

values $\boldsymbol{z}$. From Bayesian theory, this can be represented below:

$$p(\boldsymbol{y}|\mathbf{X},\boldsymbol{\mu}) = \int p(\boldsymbol{y}|\boldsymbol{z},\mathbf{X},\boldsymbol{\mu})p(\boldsymbol{z}|\mathbf{X},\boldsymbol{\mu})d\boldsymbol{z}. \tag{8}$$

In GP regression model, the likelihood term $p(\boldsymbol{y}|z,\mathbf{X},\boldsymbol{\mu})$ and prior $p(z|\mathbf{X},\boldsymbol{\mu})$ are all Gaussian. Finally we could derive the log marginal likelihood:

$$\begin{aligned}\log p(\boldsymbol{y}|\mathbf{X},\boldsymbol{\mu}) = &-\frac{1}{2}\boldsymbol{y}^\top(k+\sigma_n^2\mathbf{I})^{-1}\boldsymbol{y} - \frac{1}{2}\log|k+\sigma_n^2\mathbf{I}| \\ &-\frac{n}{2}\log 2\pi.\end{aligned} \tag{9}$$

The parameters in the model are updated after the log marginal likelihood are maximized in every iteration. Then the model can be leveraged by acquisition function to explore new point in the design space.

B. Multi-objective Optimization with Bayesian Optimization

The paradigm of conventional Bayesian optimization is natural for optimizing a single objective since the acquisition function is assumed to evaluate a single value. Regarding the design space exploration in circuits design, the most conventional objective is a vector of performance metrics, i.e., PPA. There exists strong trade-offs among these metrics in practice. Therefore, dedicated strategies are required to adopt Bayesian optimization to handle this problem. A straightforward way is to optimize a single metric value in the performance metric vector at a time in one Bayesian optimization procedure and select several better designs on each metric only. Then the whole Bayesian optimization procedure is repeated for multiple times, and each procedure finds superior designs on different metrics. The Pareto optimal ones in the PPA objective space can be picked manually by merging these selected designs.

Alternatively, scalarization can be performed to transform the three dimension vector of this three-dimensional space (delay vs. power vs. area) into a joint output as the regression target rather than using any single output [8], [15], which is formulated as

$$PPA = \alpha_1 \cdot Area + \alpha_2 \cdot Power + Delay. \tag{10}$$

Scalarization of the three metrics provides a weighted and linear relation among them. Intuitively, by changing the weight values, the Gaussian regression model will try to maximize the prediction accuracy on the most weighted direction. On the contrary, other metric directions will be predicted with less accuracy hence introducing relaxations to some extent. Therefore, we can actually explore a much larger space if we sweep the weight values over a wide range from 0 to large positive values [7]. The regression model will be guided to explore different best solutions which are Pareto-optimal. Once it is obtained, the designers can just select the most suitable designs according to specific constraints.

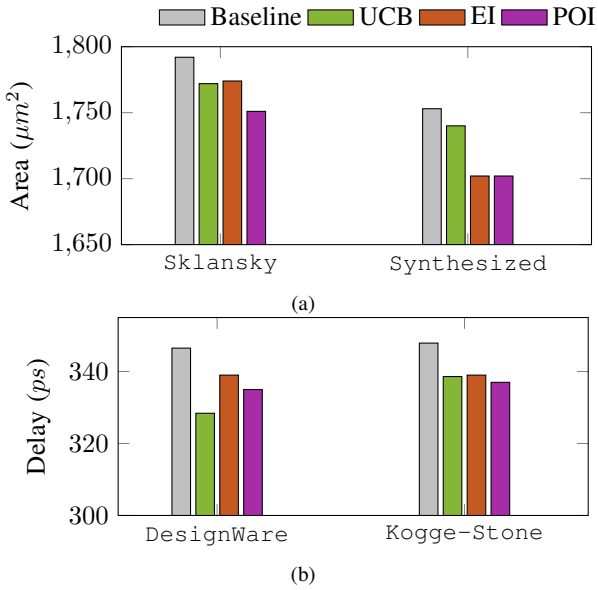

Fig. 3 Results with different acquisition functions.

IV. Experimental Results

We perform comprehensive experiments to validate the proposed Bayesian optimization approach for automatic design space exploration in back-end design flow. The experiments are conducted on a 2.2GHz Linux machine with 32GB memory. We use Design Compiler [16] (version F-2014.09-SP5) for logical synthesis, and IC Compiler [17] (version N-2017.09) for the placement and routing. "`tt1p05v125c`" corner and Non Linear Delay Model (NLDM) in 32nm SAED cell-library for LVT class [18] (available by University Program) is used for technology mapping. The core part in Bayesian optimization engine is implemented with Python. Within the engine, Tcl scripts are generated automatically based on the explored parameters, which are used to launch the back-end design tools. We use various adder designs, including regular structures like Sklansky adder and Kogge-Stone adder, synthesized adder structure obtained from [8], as well as the designs from commercial DesignWare IP libraries .

A. Results Comparison with Industrial Settings

A set of complete scripts for adder synthesis is obtained from industry, which can provide us a reasonable good performance as baseline. We develop a Bayesian optimization engine using the proposed strategy, with synthesis tools integrated in the flow. It starts with a random initialization, and the maximum iteration number is set to be 30. Firstly, we target at a single objective such as area or energy or delay. The results are presented in TABLE II. It can be seen that the Bayesian optimization can surpass the baseline performance within the limited budget.

Then we investigate the impacts of different acquisition functions. The results are demonstrated in Fig. 3. It can be observed that all acquisition functions can improve the

TABLE II Performance comparison between BO and baseline settings on single objective

Adder structure	Area (μm^2)		Energy (fJ/op)		Delay (ps)	
	Baseline	BO	Baseline	BO	Baseline	BO
`DesignWare`	2531	1873	8160	6220	346	328
`Sklansky`	1792	1772	6100	5020	356	350
`Kogge-Stone`	2563	2323	8780	6630	347	338
`Synthesized`	1753	1740	5900	5000	353	345
Average	2160.1	1927.4	7235.0	5717.5	350.9	340.7
Ratio	1.0	0.89	1.0	0.79	1.0	0.97

TABLE III Performance comparison between BO and baseline settings on multiple objectives

Adder structure	Baseline			BO		
	Area (μm^2)	Energy (fJ/op)	Delay (ps)	Area (μm^2)	Energy (fJ/op)	Delay (ps)
`DesignWare`	2531	8160	346	2473	8170	345
`Sklansky`	1792	6100	356	1791	5920	350
`Kogge-Stone`	2563	8780	347	2531	7980	340
`Synthesized`	1753	5900	353	1754	5940	350
Average	2160.1	7235.0	350.9	2137.9	7002.5	346.4
Ratio	1.0	1.0	1.0	0.989	0.967	0.987

performance compared to baseline settings, among which POI can lead to better performance in general.

Next we perform Bayesian optimization for multi-objective optimization. Since the acquisition function is defined over single value, we need to scalarize the output vector. In this experiment, the multiple objectives are merged into a single value by taking the weighted sum. The results are demonstrated in TABLE III, from which we can observe that the Bayesian optimization can achieve comparable or better results.

B. Results Comparison with Meta-heuristic Search

Generally, the CAD tool design space exploration problem can be treated as a black-box function optimization. For black-box function optimization, there are a few well-known and commonly leveraged evolutionary algorithms, e.g., genetic algorithm or simulated annealing. Thus, it is necessary to conduct comparison between the Bayesian optimization and the genetic algorithm. We plot the convergence curves for both Bayesian optimization and genetic algorithm on the same criterion during the optimization process, as shown in Fig. 4. It can be seen that the Bayesian optimization can reach lower values for both criteria. Besides, the search trajectory of the genetic algorithm is not as stable as that of Bayesian optimization. Therefore, the Bayesian optimization shows significant superiority to the genetic algorithm.

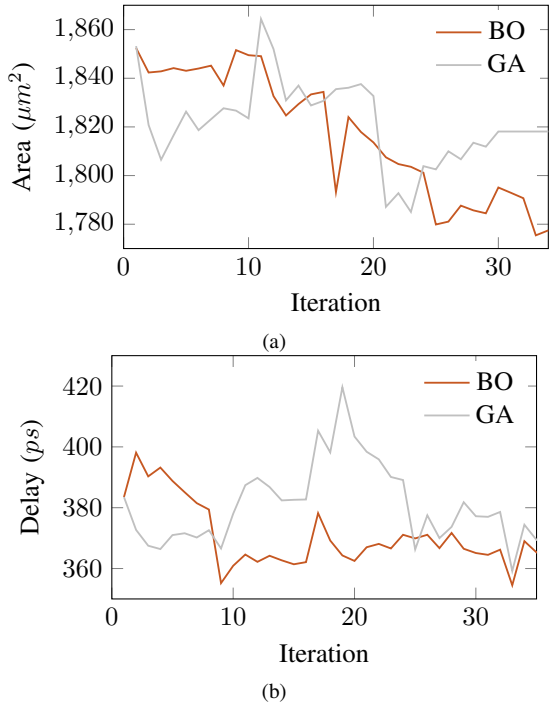

Fig. 4 Evolving curve of the adder performance.

C. Discussion

There are a few tips being observed during the experimental process, which have different effects on the final performance. Regarding the multi-objective optimization, we need to merge multiple objectives into a single scalar value. It should be noted that the values we obtained are not in the same scale, depending on the unit defined for each criterion. In order to avoid one objective value is dominated by another, scaling is considered necessary for those "small" values to ensure that all the values contribute closely to the weighed objective. During the experiments it is found that scaling up smaller

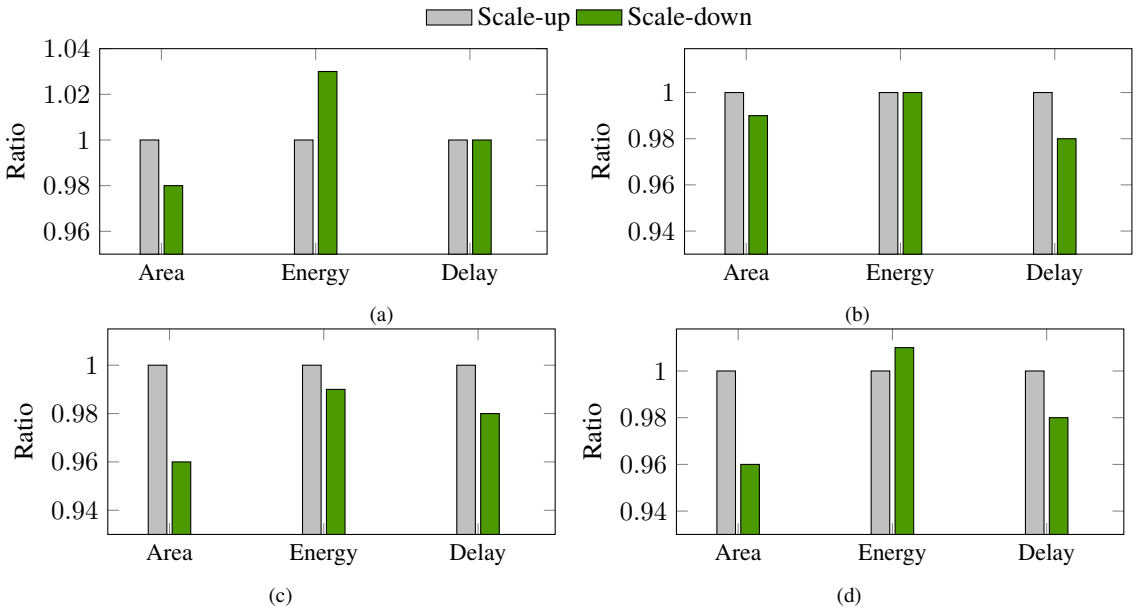

Fig. 5 The performance comparison of various designs with different scaling methods. (a) `DesignWare`; (b) `Kogge-Stone`; (c) `Sklansky`; (d) `Synthesized`.

values and scaling down large values may lead to different results. Fig. 5 shows the performance achieved using different scaling strategies. From the convergence points, most of the time scaling down larger values can achieve better results.

V. Conclusion

Bayesian optimization is a machine learning approach which can be applied for better design. In this practice, we adapt Bayesian optimization for multi-objective optimization to simultaneously minimize the PPA values of a design. In our experiments, BO substantially outperforms typical evolutionary algorithms. According to our study and experimental practice, there are still plenty of room for improvement. First, the design spaces on the front-end and back-end are separated, which may still lead to local optimal points. A natural way for improvement is to extend the design space to a unified exploration framework. The second thing is that we deal with multi-objective by simply using scalarization, which requires tuning efforts and tricks. A more elegant way to handle this would be of great help.

Acknowledgments

This work is supported by The Research Grants Council of Hong Kong SAR (Project No. CUHK24209017).

References

[1] Y. Ma, H. Ren, B. Khailany, H. Sikka, L. Luo, K. Natarajan, and B. Yu, "High performance graph convolutional networks with applications in testability analysis," in *Proc. DAC*, 2019, pp. 18:1–18:6.

[2] C.-W. Pui, G. Chen, Y. Ma, E. F. Young, and B. Yu, "Clock-aware ultrascale fpga placement with machine learning routability prediction," in *Proc. ICCAD*, 2017, pp. 929–936.

[3] H. Yang, S. Li, Z. Deng, Y. Ma, B. Yu, and E. F. Y. Young, "GAN-OPC: Mask optimization with lithography-guided generative adversarial nets," *IEEE TCAD*, 2020.

[4] G. Palermo, C. Silvano, and V. Zaccaria, "ReSPIR: a response surface-based pareto iterative refinement for application-specific design space exploration," *IEEE TCAD*, vol. 28, no. 12, pp. 1816–1829, 2009.

[5] H.-Y. Liu and L. P. Carloni, "On learning-based methods for design-space exploration with high-level synthesis," in *Proc. DAC*, 2013, pp. 50:1–50:7.

[6] P. Meng, A. Althoff, Q. Gautier, and R. Kastner, "Adaptive threshold non-pareto elimination: Re-thinking machine learning for system level design space exploration on FPGAs," in *Proc. DATE*, 2016, pp. 918–923.

[7] S. Roy, Y. Ma, J. Miao, and B. Yu, "A learning bridge from architectural synthesis to physical design for exploring power efficient high-performance adders," in *Proc. ISLPED*, 2017, pp. 1–6.

[8] Y. Ma, S. Roy, J. Miao, J. Chen, and B. Yu, "Cross-layer optimization for high speed adders: A pareto driven machine learning approach," *IEEE TCAD*, 2018.

[9] B. Reagen, J. M. Hernández-Lobato, R. Adolf, M. Gelbart, P. Whatmough, G.-Y. Wei, and D. Brooks, "A case for efficient accelerator design space exploration via bayesian optimization," in *Proc. ISLPED*, 2017, pp. 1–6.

[10] W. Lyu, F. Yang, C. Yan, D. Zhou, and X. Zeng, "Batch bayesian optimization via multi-objective acquisition ensemble for automated analog circuit design," in *Proc. ICML*, 2018, pp. 3312–3320.

[11] S. Zhang, W. Lyu, F. Yang, C. Yan, D. Zhou, and X. Zeng, "Bayesian optimization approach for analog circuit synthesis using neural network," in *Proc. DATE*, 2019, pp. 1463–1468.

[12] S. Zhang, W. Lyu, F. Yang, C. Yan, D. Zhou, X. Zeng, and X. Hu, "An efficient multi-fidelity bayesian optimization approach for analog circuit synthesis," in *Proc. DAC*, 2019, p. 64.

[13] C. E. Rasmussen and C. K. I. Williams, *Gaussian Process for Machine Learning*. The MIT Press, 2006.

[14] N. Srinivas, A. Krause, S. Kakade, and M. Seeger, "Gaussian process optimization in the bandit setting: no regret and experimental design," in *Proc. ICML*, 2010, pp. 1015–1022.

[15] K. Tumer and J. Ghosh, "Estimating the bayes error rate through classifier combining," in *Proc. ICPR*, vol. 2, 1996, pp. 695–699.

[16] "Synopsys Design Compiler," http://www.synopsys.com.

[17] "Synopsys IC Compiler," http://www.synopsys.com.

[18] "Synopsys SAED Library," http://www.synopsys.com/Community/UniversityProgram/Pages/32-28nm-generic-library.aspx, accessed 23-April-2016.