Attention-Based EDA Tool Parameter Explorer: From Hybrid Parameters to Multi-QoR metrics

Donger Luo[1], Qi Sun[2], Qi Xu[3], Tinghuan Chen[4], Hao Geng[1,5§],
[1]Shanghaitech University [2]Zhejiang University [3]University of Science and Technology of China
[4]The Chinese University of Hong Kong, Shenzhen
[5]Shanghai Engineering Research Center of Energy Efficient and Custom AI IC

***Abstract*—Improving the outcomes of very-large-scale integration design without altering the underlying design enablement, such as process, device, interconnect, and IPs, is critical for integrated circuit (IC) designers. Parameter tuning for electronic design automation (EDA) tools is an emerging technology for improving the final design Quality-of-Result (QoR). However, many complex heuristics have been accreted upon previous complex heuristics integrated into tools, resulting in a vast number of tunable parameters. Even worse, these parameters include both continuous and discrete ones, making the parameter tuning process laborious and challenging. In this paper, we propose an attention-based EDA tool parameter explorer. A self-attention mechanism is developed to navigate the parameter importance. A hybrid space Gaussian process model is leveraged to optimize continuous and discrete parameters jointly, capturing their complex interactions. In addition, considering multiple QoR metrics and the large amount of time required to invoke EDA tools, a customized acquisition function based on expected hypervolume improvement (EHVI) is proposed to enable multi-objective optimization and parallel evaluation. Experimental results on a set of IWLS2005 benchmarks demonstrate the effectiveness and efficiency of our method.**

I. Introduction

The ever-growing design complexity and time-to-market pressure for good Quality-of-Result (QoR), such as timing, power, area, etc., are challenging IC design flows and designers. Therefore, electronic design automation (EDA) tools for logic synthesis and physical design must constantly evolve to meet ever-increasing design demands. As presented in Fig. 1, parameter tuning for electronic design automation (EDA) tools, especially for logic and physical synthesis tools, is an emerging technology to achieve good QoR metrics without too much human intervention. Specifically, the designers have access to countless parameter-option and parameter combinations owing to the unstopping upgrading and incorporating algorithms into tools. Thus, even for experienced designers, finding the optimal or near-optimal parameter combination within a huge tool parameter space is a challenge. Unfortunately, time-consuming EDA tools invocation makes it impossible to exhaustively enumerate all possible parameter combinations and find the optimal or near-optimal one. Automatically tuning parameter settings is therefore essential.

In recent years, several works have been proposed that enable automatic parameter tuning utilizing heuristics or machine learning techniques. FlowTuner [1] incorporates the ant colony optimization (ACO) algorithm into a co-evolutionary framework. The AutoTuner platform [2] integrates several optimization algorithms, such as evolutionary algorithms and tree-structured Parzen estimators. coupled with an early stopping mechanism to speed up FPGA design. [3] harnesses a tensor decomposition-based recommender algorithm to tune some synthesis and physical design tool parameters. Agnesina *et al.* In [4], the active learning-based approach exploit the feature importance sampling and XGBoost regressor is developed for tool parameter tuning. By incorporating the transfer Gaussian process model, PPAtuner [5] can learn the transfer knowledge from the existing tool parameter combinations autonomously.

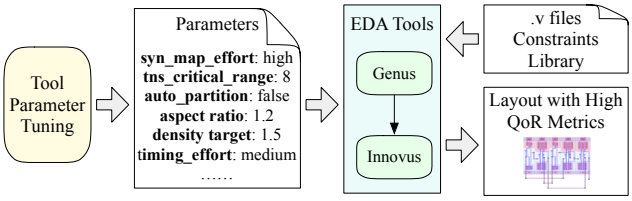

Fig. 1 The visualizations of EDA tool parameter tuning.

In the tuning scenario, long synthesis and physical design tool runtimes can be a bottleneck in collecting sufficient samples. Bayesian optimization, which performs efficient global searches by balancing exploration and exploitation, is a promising tuning framework. It has achieved some success on some EDA problems. For example, Zhang *et al.* [6] propose a Bayesian optimization approach for analog circuit synthesis using neural network. [7] solves the parameter selection problem of EDA tools through Bayesian optimization, while PTPT [8] establishes the tuning framework based on multi-objective Bayesian optimization and multi-task Gaussian models.

It is worth noticing that parameters of EDA tools consist of both discrete and continuous parameters, as shown in Fig. 1. Prior arts often simplify the issue by transforming one type of parameters into another. Although this makes optimization convenient, it prevents continuous parameters from unleashing their potential in achieving optimal design QoR metrics. To tackle the issue, we propose the principle method of constructing diffusion kernels in a hybrid space using the formula of additive kernel functions, which enables Bayesian optimization to solve the parameter tuning problem in a discrete-continuous hybrid space. We also apply the self-attention mechanism

§ Corresponding author

979-8-3503-4859-0/DATE24/© 2024 EDAA

to process the input of Bayesian optimization, so that the importance relationship between parameters can be better captured. To the best of our knowledge, this is the first time to develop a hybrid parameter tuning algorithm with applying the self-attention mechanism. Additionally, considering multiple QoR metrics and the large amount of time required to invoke EDA tools, a customized acquisition function based on expected hypervolume improvement (EHVI) is proposed to enable multi-objective optimization and parallel evaluation. Our main contributions are as follows:

- A hybrid-space multi-objective Bayesian optimization algorithm is designed for tuning tools in a more practical scenario.
- Through the self-attention mechanism, the weights of important parameters are navigated to better obtain the optimal solution.
- A customized acquisition function based on EHVI is used, and parallel computing is used to evaluate multiple points simultaneously to accelerate computation.

The rest of the paper is organized as follows. Section II introduces some prior knowledge of Bayesian optimization and provides a problem formulation. Section III sketches the whole optimization flow. Section IV describes the technical details of hybrid space Bayesian optimization. Section V depicts our self-attention mechanism and parallel evaluation, while Section VI presents the experimental results followed by conclusion in Section VII.

II. Preliminaries

A. Bayesian Optimization

Bayesian optimization (BO) is a sequential design strategy for the global optimization of noisy black-box functions. It builds surrogate models to mimic the unknown black-box objective functions, *i.e.*, the complicated EDA flow in our context. Gaussian process (GP) model is widely used as the surrogate model, which provides a posterior distribution of functions given prior and observed data.

B. Multi-objective Bayesian Optimization

Real-world optimization problems often involve multiple conflicting objectives. Multi-objective Bayesian optimization (MOBO) extends the standard BO to tackle such multi-objective problems. Let functions $\{f_i(\boldsymbol{x})\}_{i=1}^{m}$ denote the m-dimension QoR metrics to be minimized and $\boldsymbol{X}$ denotes the parameter space. A parameter vector $\boldsymbol{x}^* \in \boldsymbol{X}$ is said to dominate $\boldsymbol{x}' \in \boldsymbol{X}$, denoted as $\boldsymbol{x}^* \succeq \boldsymbol{x}'$, if the following two conditions are met in this minimization problem:

1) $f_i(\boldsymbol{x}') \geq f_i(\boldsymbol{x}^*), \forall i \in \{1, ..., m\}$,
2) There exists at least one j such that $f_j(x') > f_j(\boldsymbol{x}^*)$.

The set of parameter vectors that are not dominated by other vectors is called the Pareto-optimal set, denoted as $\boldsymbol{X}_{\text{Pareto}}$, which is the target of MOBO:

$$\boldsymbol{X}_{\text{Pareto}} = \{\boldsymbol{x}^* \in \boldsymbol{X} | \nexists \boldsymbol{x}' \in \boldsymbol{X}, \boldsymbol{x}' \succeq \boldsymbol{x}^*\}. \quad (1)$$

If $\boldsymbol{y}^* \succeq \boldsymbol{y}'$, it means that $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ dominate $(\boldsymbol{x}', \boldsymbol{y}')$. For the problem with m optimization objectives, Bayesian optimization builds m surrogate models for these objectives, denoted as $\boldsymbol{M}$, to mimic the unknown black-box objective functions. Further, an acquisition function $Ac(\boldsymbol{M}, \boldsymbol{x})$ is built based on the surrogate models and estimates the quality of a parameter vector $\boldsymbol{x}$ with respect to finding the Pareto set, without going through the time-consuming EDA flow.

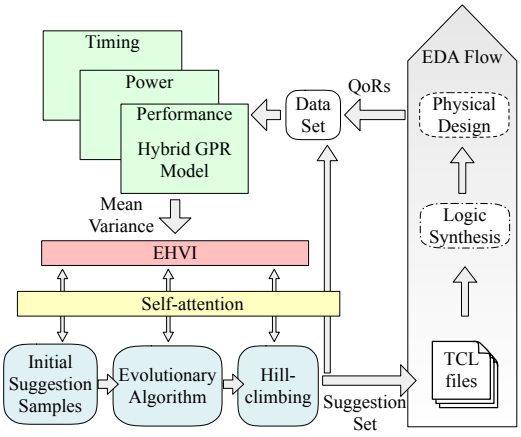

Fig. 2 The architecture of our Explorer.

C. Problem Formulation

In the context of parameter tuning, the parameter space is of hybrid types, both continuous and discrete. Thus, a parameter vector $\boldsymbol{x} = (\boldsymbol{x}^c, \boldsymbol{x}^d)$ contains $\boldsymbol{x}^c \in \mathbb{R}^c$ representing continuous variables and $\boldsymbol{x}^d \in \mathbb{Z}^d$ representing discrete variables, while the 3 metrics: performance, power and area should be optimized simultaneously.

Problem 1 (Parameter Tuning for EDA Design Tool)**.** *Given the boundary of hybrid parameters of EDA tools and the QoR metrics to be optimized, the objective of EDA tool parameter tuning is to automatically search the Pareto-optimal parameter configurations which bring about the high design quality concerning multiple QoR metrics like delay versus power/area and delay versus power versus area.*

III. Hybrid Space Bayesian Optimization Flow

In this section, we introduce a novel multi-objective hybrid Bayesian optimization approach to efficiently handle hybrid (discrete and continuous) parameter spaces. The goal of this approach is to simultaneously optimize multiple conflicting objectives, thereby providing a set of Pareto-optimal solutions.

The overall architecture of our approach is shown in Fig. 2. The EDA flow takes in suggested set of parameters as TCL (tool command language) files and then operates the logic synthesis and physical design. After that, the QoR information of the design under current parameters can be read from report files. This QoR information is appended to the data set as $\boldsymbol{Y}$ along with the set of parameters as $\boldsymbol{X}$, which was transformed using the attention function. A hybrid Gaussian process regression model is built using the data set for each metric of the QoR. These models offer mean and variance information for EHVI acquisition to select the next set of

Algorithm 1 Multi-objective Hybrid BO Approach

Input: Hybrid input domain $\boldsymbol{X}$, surrogate, acquisition function $Ac(\cdot,\cdot)$, tool flow $\mathbb{EDA}$, attention function $atten(\boldsymbol{x})$, optimization budget t_{max}, number of samples K in each step.
Output: The set of Pareto-optimal solutions $\boldsymbol{X}_{\text{Pareto}}$.

1: **Initialization**: sample initial input $\boldsymbol{X}_0$ with $\boldsymbol{X}_0 \subset \boldsymbol{X}$, $\boldsymbol{Y}_0 = \mathbb{EDA}(\boldsymbol{X}_0)$, $\boldsymbol{D}_0 = \{\boldsymbol{X}_0, atten(\boldsymbol{X}_0), \boldsymbol{Y}_0\}$;
2: **for** $t \leftarrow 1$ to t_{max} **do**
3: Build m surrogate models: $\boldsymbol{M}_{t;1,\ldots,m} \leftarrow \mathcal{GP}(atten(\boldsymbol{X}_{t-1}), \boldsymbol{Y}_{t-1})$;
4: Select the next candidate set to evaluate: $\boldsymbol{X}_t \leftarrow$ Top K $Ac(\boldsymbol{M}_{t;1,\ldots,m}, atten(\boldsymbol{x}))$, $\forall \boldsymbol{x} \in \boldsymbol{X}$;
5: Fixing discrete parameters $\boldsymbol{x}_d$, optimize continuous parameters $\boldsymbol{x}_c$ through evolutionary algorithm;
6: Fixing continuous parameters $\boldsymbol{x}_c$, optimize discrete parameters $\boldsymbol{x}_d$ through hill-climbing;
7: Evaluate next candidate set $\boldsymbol{Y}_t = \mathbb{EDA}(X_t)$;
8: Update $\boldsymbol{D}_t \leftarrow \{\boldsymbol{X}_t, atten(\boldsymbol{X}_t), \boldsymbol{Y}_t\}$;
9: Update $\boldsymbol{X} \leftarrow \boldsymbol{X} \backslash \boldsymbol{X}_t$;
10: **end for**
11: Select Pareto vectors $\boldsymbol{X}_{\text{Pareto}}$ according to $\boldsymbol{D}_{t_{max}}$;
12: **return** $\boldsymbol{X}_{\text{Pareto}}$.

suggestions. The next set of suggestions to evaluate is selected from random samples, they are then transformed using the attention function, and each gets an EHVI score. The K samples with top K scores will be optimized with evolutionary algorithm and hill-climbing, and become the next suggested set of parameters.

The proposed approach is encapsulated in Algorithm 1, which outlines the steps taken in each iteration of the optimization process. In line 1, the algorithm begins by randomly sampling the initial parameter vectors $\boldsymbol{X}_0$ from the parameter space. The parameter vectors are then fed into the EDA tools to run designs to get ground-truth performance values $\boldsymbol{Y}_0$. For simplicity, we denote the EDA tools as $\mathbb{EDA}$. The $\boldsymbol{X}_0$ is then transformed using the attention function to learn better inputs, as discussed in Section V. The initial data set is $\boldsymbol{D}_0 = \{\boldsymbol{X}_0, atten(\boldsymbol{X}_0), \boldsymbol{Y}_0\}$. In each iteration (line 3 to 9), we build a set of GP models $\boldsymbol{M}$, one for each objective (line 3). The next candidate set for evaluation is composed of parameter vectors with the top-K $Ac(\boldsymbol{M}, atten(\boldsymbol{X}_0))$ values over the current GP models. The selected candidates are then fed into the EDA tools to evaluate the performance, and the results are added to the sample set for the next iteration (line 4). This process is repeated until the maximum number of iterations is reached. Finally, the algorithm returns the set of all Pareto-optimal solutions from the sampled set. Note that given the sampled set $\boldsymbol{D}_{t_{max}}$, it is easy to select its Pareto-optimal vectors according to Equation (1) (line 11).

In a nutshell, this proposed method allows for uncertainty-aware exploration of the parameter space, and enables the identification of the Pareto-optimal parameter configurations.

IV. Hybrid space Gaussian process

In this section, we detail the methodology we employed to carry out Bayesian optimization (BO) for optimizing QoR metrics in a hybrid space, which includes both continuous and discrete parameters. This is achieved using the concept of additive hybrid diffusion kernels.

A. Kernels of Continuous Space Gaussian Processes

Gaussian processes (GPs) allow us to mimic unknown functions in continuous space to make predictions given some observed data. Given a set of single-objective observations $\boldsymbol{D} = (\boldsymbol{x}_i, y_i)_{i=1}^N$, the kernel $k(\boldsymbol{x}, \boldsymbol{x}')$ defines the covariance between the function values at two input points $\boldsymbol{x}$ and $\boldsymbol{x}'$. In GPs, the prior over functions can be written as:

$$f(\boldsymbol{x}) \sim \mathcal{GP}(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')), \tag{2}$$

where $m(\boldsymbol{x})$ is the mean function and $k(\boldsymbol{x}, \boldsymbol{x}')$ is the kernel function. A common choice of kernel in continuous space is the radial basis function (RBF) kernel, also known as the squared exponential kernel:

$$k(\boldsymbol{x}, \boldsymbol{x}') = \sigma^2 \exp\left(-\frac{||\boldsymbol{x} - \boldsymbol{x}'||^2}{2s^2}\right), \tag{3}$$

where σ^2 is the signal variance and s is the length scale. The kernel function and its associated hyperparameters have a significant impact on the GP's predictive performance, making the choice of an appropriate kernel essential in GP modeling.

B. Diffusion Kernels for Discrete Spaces

The diffusion kernel is powerful to characterize the discrete parameter space by constructing a graph for the discrete parameter vectors and simulating the diffusion or random walk process over the graph. Denote the graph as $\boldsymbol{G}$, with each node being a parameter vector. Two nodes are connected if their parameter vectors have only one different element value. Denote the node set as $\boldsymbol{V}$ (*i.e.*, the set of discrete parameter vectors), the edge set as $\boldsymbol{E}$, the adjacency matrix as $\boldsymbol{A}$, and the degree matrix as $\boldsymbol{D}$, respectively. Then the normalized form of the Laplacian matrix is defined as:

$$\boldsymbol{L}(\boldsymbol{G}) = \mathbf{I} - \boldsymbol{D}^{(-1/2)}\boldsymbol{A}\boldsymbol{D}^{(-1/2)}, \tag{4}$$

where $\mathbf{I}$ is the identity matrix. The normalized Laplacian scales the influence of each node by its degree. The diffusion kernel over our graph node set $\boldsymbol{V}$ is defined in Equation (5):

$$\boldsymbol{K}(\boldsymbol{V}, \boldsymbol{V}) = \boldsymbol{\Phi} \exp(-\gamma \boldsymbol{\Pi}) \boldsymbol{\Phi}^T, \tag{5}$$

where γ is a hyper-parameter, $\boldsymbol{\Phi} = [\phi_1, \cdots, \phi_{|\boldsymbol{V}|}]$ and $\boldsymbol{\Pi} = [\pi_1, \cdots, \pi_{|\boldsymbol{V}|}]$ are the eigenvector matrix and eigenvalue matrix of $\boldsymbol{L}$, respectively. The diffusion kernel has a solid mathematical foundation based on the Laplacian operator and the heat diffusion process, and it can be computed efficiently in closed forms for large graphs [9].

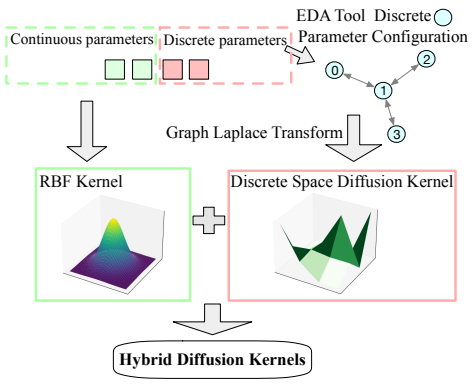

Fig. 3 Construction of Hybrid Diffusion Kernels. For the RBF kernel, X and Y axis in the picture represent the value of the two continuous inputs, respectively, while the Z axis represents the value of Equation (3) with $\boldsymbol{x}'$ fixed to $(0, 0)$. For the discrete space diffusion kernel, the two discrete parameters are first transferred into a graph where each node of the graph represents a combination of discrete parameters. In the picture, X and Y axis represents the graph inputs, while the Z axis represents the value of Equation (5).

C. Hybrid Diffusion Kernels

Hybrid diffusion kernels are designed to handle hybrid input spaces. These kernels are constructed by combining the continuous and discrete diffusion kernels in a way that allows for seamless integration of both types of input spaces.

To construct hybrid diffusion kernels, we use the general formulation of additive Gaussian process kernels, which defines an additive hybrid diffusion kernel over hybrid spaces. The key idea is to assign a base kernel for each input dimension $i \in 1, 2, \cdots, n_c + n_d$, where n_c and n_d represent the number of discrete and continuous parameters in hybrid space $\boldsymbol{X}$. RBF kernel and discrete diffusion kernel act as base kernels for continuous and discrete input dimensions, respectively, as shown in Fig. 3. The overall kernel is constructed by summing all possible orders of interactions between these base kernels.

The overall additive hybrid diffusion kernel $\mathcal{K}_{HBS}(\boldsymbol{x}, \boldsymbol{x}')$ over hybrid spaces is defined as:

$$\mathcal{K}_{HBS} = \sum_{p=1}^{n_c+n_d} \left(\theta_p^2 \sum_{i_1, \cdots, i_p} \prod_{d=1}^{p} k_{i_d} \left(\boldsymbol{x}_{i_d}, \boldsymbol{x}'_{i_d} \right) \right), \quad (6)$$

where θ_p is a hyper-parameter associated with each additive kernel, $1 \leq i_1 < i_2 < \cdots < i_p \leq n_c + n_d$ denotes the possible selections of kernels, k_{i_d} is the base kernel for the input dimension i_d. By using hybrid diffusion kernels, we can effectively perform Bayesian optimization over input spaces with mixed continuous and discrete parameters, which is particularly useful for applications like EDA tool tuning.

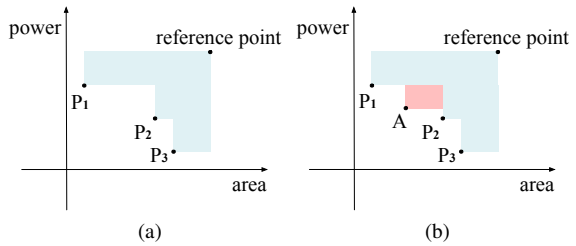

Fig. 4 Example of hypervolume improvement in 2-d space.

V. Attention Mechanism and Parallel Evaluation based on EHVI

A. Attention Mechanism in Bayesian Optimization

Attention mechanism, originating from the field of deep learning, is an effective mechanism to encode the dependencies between different parts of input data. In EDA tool parameter tuning, we use it to model the relationships between different parameters of our hybrid input parameter space.

Since the attention mechanisms we designed for discrete and continuous parameters are the same, we exploit $\boldsymbol{x}$ to represent a parameter configuration vector either discrete or continuous. x_i represents the i-th element in vector (i.e., tool parameter). The attention module computes the importance scores between all elements of input vectors, as defined in Equation (7).

$$Importance(x_i) = \sum_{j=1}^{n} x_i \cdot x_j, \quad (7)$$

then the weighted version of original input parameter is updated to the GP model, as shown in Equation (8).

$$atten(x_i) = x_i \cdot Importance(x_i). \quad (8)$$

The attention design allows our Bayesian optimization algorithm to consider complex dependencies between different parameters in the input space while it is easy to compute.

B. Parallel Evaluation based on EHVI

Given a set of Pareto-optimal parameter vectors $\boldsymbol{X}_{\text{Pareto}} = \{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n\}$, the corresponding performance set is $\boldsymbol{Y}_{\text{Pareto}} = \{\boldsymbol{y}_1, \cdots, \boldsymbol{y}_n\} \in \mathbb{R}^m$, where m denotes the number of objectives. A newly sampled parameter vector $\boldsymbol{x}$ is regarded as Pareto-optimal if its performance vector $\boldsymbol{y}$ is not dominated by the current $\boldsymbol{Y}_{\text{Pareto}}$. Correspondingly, the improvement of the hypervolume resulting from sampling the Pareto-optimal $(\boldsymbol{x}, \boldsymbol{y})$ is defined in Equation (9).

$$I(\boldsymbol{y}) = \int_{\mathbb{R}^m} [\boldsymbol{y} \succeq \boldsymbol{y}' \wedge \forall i \in 1, \ldots, n : \boldsymbol{y}_i \not\succeq \boldsymbol{y}'] \mathrm{d}\boldsymbol{y}', \quad (9)$$

which computes the hypervolume improvement brought by previously found $\boldsymbol{Y}_{\text{Pareto}}$.

Fig. 4 is an example of hypervolume improvement in the 2-d space. Given a reference point and Pareto-optimal P_1, P_2, and P_3, the area covered in blue is the hypervolume of the Pareto sets. Point A, which is not dominated by the Pareto-optimal points, can bring an improvement to the hypervolume, so the size of the area in red is the hypervolume improvement.

TABLE I Benchmark Designs

Name	#Cells	Clock Period (ps)
des3_area	3097	700
b15	12014	1000
b19	48398	1200
vga	73898	400

TABLE II Examples of Parameters

Parameters	Stage	Range
syn_generic_effort	synthesis	high/medium/low
syn_map_effort		high/medium/low
tns_critical_range		0-15
auto_partition		true/false
aspect ratio	floorplan	0.5-2.0
density target		0.5-1.0
place_detail_wire_length_opt_effort	placement	high/medium/none
place_global_cong_effort		auto/high/medium/low
place_global_timing_effort		high/medium
drouteUseMultiCutViaEffort	routing	high/medium/low
routeWithLithoDriven		true/false

The expected hypervolume improvement EHVI is defined as the expectation of $I(\boldsymbol{y})$ with respect to the posterior predictive distribution of the GP:

$$\text{EHVI}(\boldsymbol{y}) = \mathbb{E}_{p(\boldsymbol{y}|\boldsymbol{D})}[I(\boldsymbol{y})], \tag{10}$$

where $\boldsymbol{D}$ denotes the observed data. We then select the next sample point to be the one that maximizes the EHVI:

$$\boldsymbol{y}^* = \arg\max_{\boldsymbol{y}} \text{EHVI}(\boldsymbol{y}). \tag{11}$$

The reference point is a virtual hyperparameter set by ourselves, which is only for ease of computation. Generally, it is set as a very bad performance value point.

In our algorithm, the EHVI acquisition function is used to assign scores to a large number of points randomly selected from the hybrid space. The K points with top K EHVI scores are selected for further optimization. Specifically, the discrete part of these K selected points undergoes optimization through a hill-climbing algorithm, while the continuous part is optimized using an evolutionary algorithm. After the optimization, these points are evaluated in parallel. By integrating parallel evaluation, our algorithm is able to quickly and efficiently explore and exploit the hybrid space, making it highly effective for the problem of Bayesian optimization in hybrid spaces.

VI. Experimental Results

A. Experimental Setup and Benchmarks

We test our Explorer with a VLSI design flow consisting of Cadence Genus and Innovus (version 20.1), on a platform with 2 Xeon Gold 6226R CPU processors. Due to the space limitation, our experiments are conducted on 4 representative IWLS2005 benchmark designs [10] (shown in TABLE I) under a 7-nm technology node [11]. Specifically, vga and b19 are the two largest designs in the benchmark which can be synthesized under this technology node. According to the experience, 27 potential design performance-relative parameters are selected and tuned in our experiments, among which 4 are continuous parameters, and 23 are discrete parameters. Minimum clock period, total power and total area are chosen as the metrics to be optimized simultaneously. Examples of the parameters are shown in TABLE II.

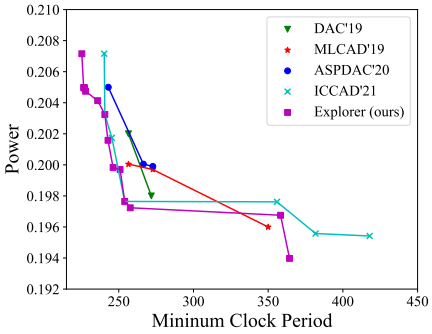

Fig. 5 The visualizations of Pareto frontiers in two QoR metric spaces of des_area;

The following metrics are used to compare each parameter tuning method on all 4 designs: hypervolume (HV), maximum performance improvement (MPI1), maximum power improvement (MPI2), and maximum area improvement (MAI). These metrics are good at assessing the ability of each method to explore the tool parameter spaces. Runtime of the algorithm isn't listed because it's only a drop in the bucket (about 3 mins) compared with EDA tools (3 to 6 hours) in each iteration. To compute the hypervolume, we use $(900, 1, 800)$, $(1200, 20, 110000)$, $(1500, 5, 30000)$, $(600, 100, 150000)$ as the reference points for the 4 designs respectively based on the range of metrics.

To fairly and truly evaluate the performance state of our work, we compare it to the following state-of-the-art methods by exploring the parameter space of the selected designs using 50 iterations:

- DAC'19 [3]: The model is pre-trained based on 30 highest reward configurations from historical design records. For 20 iterations of online training, the model's recommendations are employed to guide sampling process.
- MLCAD'19 [7]: A Bayesian optimization method implemented with the package GPyOpt [12].
- ASPDAC'20 [4]: A feature-importance sampling and tree-based method.
- ICCAD'21 [1]: A cooperative co-evolutionary controller and ant colony optimization-based method.

B. Comparisons Against SOTA Works

TABLE III shows the comparison results of Explorer with existing methods. Baseline setting is a parameter set with all parameters set to the highest effort. It can be seen that our Explorer averagely behaves the best on MPI1, MPI2, and MAI. Moreover, the Explorer achieves better hypervolume than existing methods on all benchmarks, which indicates that the proposed method can find better tool parameter configurations when considering multiple objectives concurrently.

To improve the readability and reproducibility, some results from the Pareto frontiers of parameter configurations des_area given by each method are listed in TABLE IV

TABLE III Comparisons between Explorer and SOTA works on the 4 benchmarks, with data in parentheses representing the ratios of these values to the values produced by Explorer

Design	Metric	DAC'19	MLCAD'19	ASPDAC'20	ICCAD'20	Explorer (ours)
des3_area	MPI1(%)	12.69	17.33	3.33	15.92	**18.20**
	MPI2(%)	2.54	2.04	0.05	3.05	**4.04**
	MAI(%)	0.28	0.28	0	0	**0.67**
	$HV(10^3)$	46.3(0.94)	45.7(0.95)	38.9(0.79)	44.7(0.91)	**48.9**(1)
b15	MPI1(%)	1.84	2.62	4.20	4.10	**5.46**
	MPI2(%)	2.56	2.87	2.69	3.46	**3.60**
	MAI(%)	**6.82**	6.51	6.03	5.94	6.61
	$HV(10^3)$	718.7(0.94)	733.7(0.96)	738.1(0.97)	728.6(0.95)	**759.3**(1)
b19	MPI1(%)	1.30	2.76	3.01	3.89	**4.58**
	MPI2(%)	1.41	2.43	0.66	1.21	**2.90**
	MAI(%)	0.47	0.56	1.18	0.98	**1.23**
	$HV(10^3)$	349.5(0.80)	411.2(0.94)	368.7(0.84)	428.9(0.98)	**435.3**(1)
vga	MPI1(%)	21.10	7.51	**24.56**	15.89	23.99
	MPI2(%)	1.53	0.65	6.60	5.45	**7.69**
	MAI(%)	0.15	0.73	2.20	3.66	**4.40**
	$HV(10^3)$	412.2(0.84)	446.7(0.91)	457.5(0.93)	450.9(0.91)	**490.3**(1)
Average MPI1(%)		9.23	7.55	8.77	9.95	**13.05**
Average MPI2(%)		2.01	1.99	2.5	3.29	**4.55**
Average MAI(%)		1.93	2.02	2.35	2.64	**3.22**
Average $HV(10^3)$		381.6(0.88)	409.3(0.94)	400.8(0.92)	413.2(0.95)	**433.4**(1)

TABLE IV Parameter sets outputted by each method for design `des_area`, along with corresponding minimum clock period(ps), total power(mW) and total area(μm^2)

	syn_generic_effort	syn_map_effort	tns_critical_range	aspect ratio	place_detail_wire _length_opt_effort	place_global_ timing_effort	Minimum clock period	Power	Area
baseline	high	high	15	1	high	high	275.7	0.203	591.8
DAC'19 [3]	high	high	15	1	medium	high	267.3	0.201	591.8
MLCAD'19 [7]	high	medium	8	1	low	high	273.4	0.199	592.2
ASPDAC'20 [4]	high	medium	8	1	high	meidum	266.5	0.200	591.8
ICCAD'21 [1]	high	high	15	0.8	high	low	287.4	0.199	590.1
Explorer (ours)	high	medium	14	1.13	medium	high	250.9	0.198	588.0

along with partial associated parameters. The visualizations of the Pareto frontiers in power versus minimum clock period predicted by the methods on benchmark `des_area` is displayed in Fig. 5. The purple square points refer to our method. It visually shows that the parameter configurations we find dominate those found by other algorithms.

VII. Conclusion

In this paper, for the first time, we propose an attention mechanism-based EDA tool parameter explorer surrogated with a hybrid space Gaussian process model. The explorer navigates the weights of important parameters through the attention mechanism and constructed a hybrid space multi-objective Bayesian optimization process that is more closely aligned with real-world scenarios. Moreover, parallel computing is used to accelerate computation by evaluating multiple points simultaneously. Experimental results on 4 IWLS2005 benchmarks under an advanced 7nm technology node have demonstrated the effectiveness of the proposed framework.

Acknowledgment

This work is sponsored by Shanghai Pujiang Program (Project No. 22PJ1410400).

References

[1] R. Liang, J. Jung, H. Xiang, L. Reddy, A. Lvov, J. Hu, and G.-J. Nam, "FlowTuner: A Multi-Stage EDA Flow Tuner Exploiting Parameter Knowledge Transfer," in *Proc. ICCAD*. IEEE, 2021, pp. 1–9.

[2] J. Jung, A. B. Kahng, S. Kim, and R. Varadarajan, "METRICS2. 1 and Flow Tuning in the IEEE CEDA Robust Design Flow and OpenROAD ICCAD Special Session Paper," in *Proc. ICCAD*, 2021, pp. 1–9.

[3] J. Kwon, M. M. Ziegler, and L. P. Carloni, "A Learning-Based Recommender System for Autotuning Design Flows of Industrial High-Performance Processors," in *Proc. DAC*, 2019, pp. 1–6.

[4] Z. Xie, G.-Q. Fang, Y.-H. Huang, H. Ren, Y. Zhang, B. Khailany, S.-Y. Fang, J. Hu, Y. Chen, and E. C. Barboza, "FIST: A Feature-Importance Sampling and Tree-Based Method for Automatic Design Flow Parameter Tuning," in *Proc. ASPDAC*, 2020, pp. 19–25.

[5] H. Geng, Q. Xu, T.-Y. Ho, and B. Yu, "PPATuner: pareto-driven tool parameter auto-tuning in physical design via gaussian process transfer learning," in *Proc. DAC*, 2022, pp. 1237–1242.

[6] 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.

[7] Y. Ma, Z. Yu, and B. Yu, "CAD tool Design Space Exploration via Bayesian Optimization," in *Proc. MLCAD*, 2019, pp. 1–6.

[8] H. Geng, T. Chen, Y. Ma, B. Zhu, and B. Yu, "PTPT: Physical Design Tool Parameter Tuning via Multi-objective Bayesian Optimization," *IEEE TCAD*, vol. 42, no. 1, pp. 178–189, 2022.

[9] R. I. Kondor and J. Lafferty, "Diffusion kernels on graphs and other discrete structures," in *Proc. ICML*, vol. 2002, 2002, pp. 315–322.

[10] C. Albrecht, "IWLS 2005 benchmarks," in *International Workshop for Logic Synthesis (IWLS): http://www.iwls.org*, 2005.

[11] L. T. Clark, V. Vashishtha, L. Shifren, A. Gujja, S. Sinha, B. Cline, C. Ramamurthy, and G. Yeric, "ASAP7: A 7-nm finFET predictive process design kit," *Microelectronics Journal*, vol. 53, pp. 105–115, 2016.

[12] N. Knudde, J. van der Herten, T. Dhaene, and I. Couckuyt, "GPflowOpt: A Bayesian optimization library using TensorFlow," *arXiv preprint arXiv:1711.03845*, 2017.