MOSTAR: Multi-Stage Hierarchical Bayesian Optimization for Substructure-Aware High-Dimensional Analog Circuit Sizing

Weijian Fan[1,2†], Haoyi Zhang[2†], Weibin Lin[5], Runsheng Wang[2,3,4], Yibo Lin[2,3,4*]
[1]School of Software and Microelectronic [2]School of Integrated Circuits, Peking University
[3]Beijing Advanced Innovation Center for Integrated Circuits [4]Institute of Electronic Design Automation, Peking University, Wuxi, China [5]The College of Mechatronics and Control Engineering, Shenzhen University

***Abstract*—Analog circuit sizing is a critical challenge due to increasing circuit complexity and diverse performance requirements. Existing algorithms struggle with poor scalability in high-dimensional spaces and frequent convergence to local optima. To address these limitations, we propose MOSTAR, a multi-stage hierarchical Bayesian optimization framework that integrates a local-to-global GNN (L2G-GNN). L2G-GNN identifies circuit substructures and adds symmetric constraints to the circuit. MOSTAR employs additive Gaussian processes and stage-adaptive constrained acquisition function to improve scalability in high-dimensional circuits. Furthermore, its dynamic search space adjustment strategy helps avoid local optima during optimization. Experiments show that our L2G-GNN achieves a substructure identification accuracy of 97.22%, and MOSTAR achieves an optimization performance improvement ranging from 1.04× to 4.13× on three basic circuits and two high-dimensional circuits, highlighting its efficacy in automating complex analog circuit sizing.**

***Index Terms*—Bayesian Optimization, high-dimensional circuits, L2G GNN, MOSTAR**

I. Introduction

Analog circuit design usually requires designers to invest significant time in determining the size of circuit components due to the increased complexity from shrinking scales and diverse performance specifications. To speedup the design process, automated analog circuit sizing techniques have become increasingly important. One common approach is to formulate the circuit sizing task as a single-objective optimization problem using a comprehensive Figure of Merit (FOM) [1], and then employ conventional algorithms to optimize it [2], [3], [4]. Alternatively, a more practical strategy is to solve it as a multi-objective optimization problem directly with corresponding algorithms [5], [6]. However, these methods struggle with high-dimensional circuits due to their poor scalability. Thus, developing an efficient method for high-dimensional constrained circuit optimization is crucial.

Existing work on optimization algorithms for analog circuit sizing can be categorized into three classes: heuristic algorithms, reinforcement learning-based optimization algorithms, and constrained Bayesian optimization algorithms.

Firstly, existing heuristic algorithms [7], [8] like NSGA-II [9] and MOEA/D [10] optimize multiple objectives to generate Pareto-optimal solutions for analog circuit sizing. NSGA-II uses non-dominated sorting and crowding distance, while MOEA/D decomposes problems into weighted subproblems. However, their effectiveness drops in high-dimensional circuits due to complexity.

Secondly, reinforcement learning (RL) algorithms optimize analog circuit sizing using reward-driven feedback. For instance, [11] uses deep RL with symbolic filters for efficient sizing. [12] introduces a prioritized RL framework with non-uniform sampling for data-efficient optimization. GCN-RL [13] combines graph convolutional networks with RL for transferable transistor sizing. PVTSizing [14] employs TuRBO-RL-based batch sampling for PVT-robust sizing via trust region Bayesian optimization. However, RL methods face challenges: (1) formulating sizing as a Markov Decision Process increases complexity, and (2) high simulation data and resource demands limit their practicality for smaller teams.

Thirdly, constrained Bayesian optimization (CBO) algorithms [15], [16] use probabilistic models for analog circuit sizing, requiring less data and lower complexity than RL for simpler EDA tasks. Notable methods include PESMOC [17], reducing Pareto set entropy; USEMOC [18], using two-stage optimization; and MACE [19], leveraging multiple acquisition functions. For high-dimensional problems, hypervolume improvement-based (HVI) CBO methods like MORBO [20], optimizing parallel trust regions; LoCoMOBO [21], using local surrogate models with Thompson Sampling; and ABCMOBO [22], employing Expected HVI [23] with dynamic reference points, are proposed. However, CBO often gets trapped in local optima, scales poorly in high-dimensional circuits, and HVI-based methods rely on EHVI sampling, limiting efficacy under strict constraints.

According to the analysis of the above algorithms, the current challenges mainly include circuit complexity, the tendency to get trapped in local optima, and poor scalability in high-dimensional circuits. To address these challenges, we propose MOSTAR, a novel approach using a local-to-global GNN to reduce circuit complexity and a multi-stage hierarchical Bayesian optimization method to mitigate local optima and enhance scalability in high-dimensional circuits. The primary contributions of this paper are as follows:

1) We propose a local-to-global GNN to identify circuit substructures and incorporate symmetric constraints, thereby reducing the complexity of the high dimensional circuit.
2) We propose MOSTAR, a multi-stage hierarchical Bayesian optimization framework that leverages additive Gaussian processes as the surrogate model, employs a stage-adaptive constrained acquisition function for optimization, and integrates a dynamic strategy to update the

†Equal Contribution
*Corresponding author: yibolin@pku.edu.cn

search space adaptively. This framework addresses the challenges of poor scalability and local optima.

3) Experimental results show our local-to-global GNN achieves 97.22% prediction accuracy on our dataset. Moreover, MOSTAR demonstrates superior optimization in both basic and high-dimensional circuits, with improvements ranging from 1.04× to 4.13×.

II. Preliminaries

A. Problem Formulation

The analog circuit sizing problem can be formulated as a constrained optimization task [24] aiming to minimize an objective function while meeting multiple constraints, expressed as:

$$\begin{aligned} &\min f_0(\mathbf{x}), \quad \mathbf{x} \in \mathcal{X} \\ &\text{s.t.} \quad f_j(\mathbf{x}) \geq C_j, \quad j = 1, \ldots, n_c \end{aligned} \tag{1}$$

Here, $\mathcal{X} = \{\mathbf{x} \in \mathbb{R}^d \mid \mathbf{x_L} \leq \mathbf{x} \leq \mathbf{x_U}\}$ defines the search space of design variables, set by Process Design Kit (PDK) rules and designer experience. The vector $\mathbf{x}$ includes d device size variables, with $\mathbf{x_L}$ and $\mathbf{x_U}$ as lower and upper bounds. $f_0(\mathbf{x})$ is the objective function, and $f_j(\mathbf{x}) \geq C_j$ are the constraint functions with minimum values C_j.

B. Bayesian Optimization

Bayesian optimization (BO) [25], [26] is an efficient method for optimizing expensive black-box functions, using a surrogate model to approximate the objective and an acquisition function to select the next candidate point. Gaussian Process Regression (GPR) [27] is a common surrogate model. For an input $\mathbf{x}$ with d dimensions and an unknown function $y = f(\mathbf{x}) + \epsilon$, where $\epsilon \sim N(0, \sigma_n^2)$, given N samples in a dataset $D = \{\mathbf{X}, \mathbf{y}\}$, GPR provides the posterior for any input $\mathbf{x}^*$:

$$\mu(\mathbf{x}^*) = m(\mathbf{x}) + k(\mathbf{x}^*, \mathbf{X}) \left[K + \sigma_n^2 I\right]^{-1} (\mathbf{y} - m(\mathbf{x})) \tag{2}$$

$$\sigma^2(\mathbf{x}^*) = k(\mathbf{x}^*, \mathbf{x}^*) - k(\mathbf{x}^*, \mathbf{X}) \left[K + \sigma_n^2 I\right]^{-1} k(\mathbf{X}, \mathbf{x}^*) \tag{3}$$

where $\mu(\mathbf{x}^*)$ is the predicted mean, $\sigma(\mathbf{x}^*)$ is the standard deviation, $k(\mathbf{x}^*, \mathbf{X})$ are kernel values, and $K = k(\mathbf{X}, \mathbf{X})$ is the kernel matrix. We set $m(\mathbf{x}) = 0$ and use the RBF kernel [28] for simplicity. Common acquisition functions, such as Expected Improvement (EI), Probability of Improvement (PI), and Upper Confidence Bound (UCB), guide the optimization process.

C. Graph Representation of the Netlist

In our work, We use an undirected bipartite graph $G(V, E)$ to model the circuit netlist [29], with vertex set V split into V_e (components like transistors, resistors) and V_n (nets connecting components), and edge set E linking V_e to V_n, ensuring bipartiteness. For example, Figure 1 shows a current mirror circuit with nets as blue vertices, components as green vertices, and edges labeled with a three-bit tag (e.g., '001' for drain connection). By converting the netlist into this graph structure, Graph Neural Networks (GNNs) can process circuits.

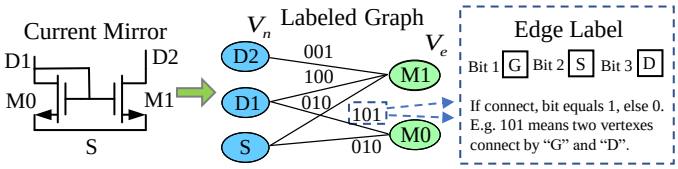


Fig. 1: The bipartite graph representation of the current mirror.

III. Algorithm

A. Overview of MOSTAR

MOSTAR comprises two key components: L2G-GNN and MOSTAR. The former (detailed in Section B) includes a netlist dataset generation method and the L2G-GNN model. The latter, MOSTAR integrates additive Gaussian processes, a stage-adaptive constrained acquisition function, and a dynamic search space adjustment strategy (Sections C-E).

The overall workflow of our method is as follows: Initially, the L2G-GNN recognizes relevant substructures in the target analog circuit. Following this identification, we add symmetry constraints to certain substructures. Finally, we apply MOSTAR to optimize the resulting circuit, where the dynamic search space adjustment strategy iteratively refines the search space during the optimization.

B. Local-to-Global GNN

1) Data Generation Method: The training of Graph Neural Network (GNN) models is constrained by the scarcity of circuit netlist datasets. We investigated open-source datasets [30], [31], [32], but found they lack proper classification, substructure annotations, and clear netlist structures. To overcome these challenges, we developed a program to generate annotated netlist datasets, enabling effective GNN training.

Firstly, we collect common circuit substructures (e.g., current mirrors, differential pairs), creating a netlist for each. These are then encapsulated as Python functions and stored in a substructure library. Using this, we build single, two, and three-stage operational amplifier (opamp) circuits, forming an opamp library. Finally, leveraging both libraries, we construct diverse circuit types, such as comparators and low-dropout regulators (LDOs). Figure 2 illustrates this dataset construction process.

The generated dataset has a unified format: a circuit netlist as input and a set of labeled components as output. For example, for a five-transistor OTA, the input is its netlist, and the output is: P_Current_Mirror:[MM1, MM2], N_Different_Pair:[MM3, MM4], N_Current_Source:[MM5].

2) Local-to-Global GNN(L2G-GNN): GNNs are commonly used for circuit structure identification [33], [34], [35]. However, different GNNs have different limitations: GCN [36] uses 1st-order filters for efficient neighbor feature extraction, but may miss long-range dependencies; GraphSAGE [37] uses convolutional aggregators for local filter approximation, offering aggregation flexibility, but its performance is function-sensitive and it struggles with complex relationships; CR-GCN [38] learns subgraph features with GIN and processes connections with GAT [39] to capture circuit structure, but it is complex and lacks generalization ability.

Inspired by CR-GCN and to address the aforementioned limitations, we introduce L2G-GNN. This model uses a local GNN

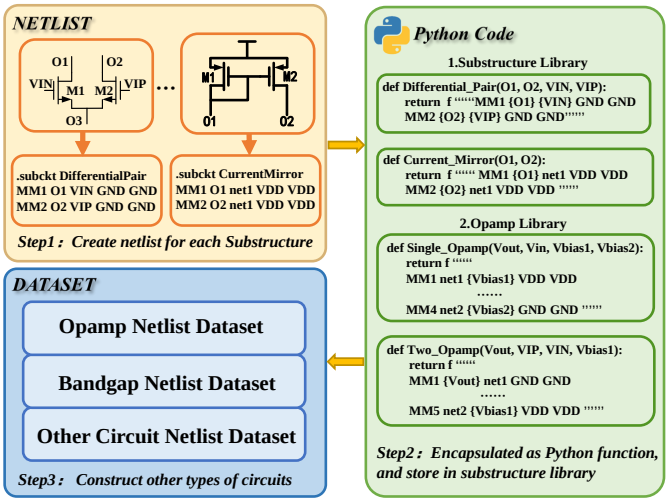


Fig. 2: The construction process of the circuit netlist dataset.

for intra-module feature extraction, an MLP for local feature processing, and a global GNN for inter-module relationship capture. During our experiments, we observed that GCNs are well-suited for substructure identification. Therefore, we use GCN as the building blocks for the local GNN (two GCN layers with ReLU after the first) and the global GNN (five GCN layers with ReLU after the first four).

The L2G-GNN workflow, illustrated in Figure 3, begins by converting the circuit netlist into a bipartite graph G. For each node in G, we extract its 1-hop subgraph, representing the directly connected circuit elements. All extracted subgraphs are then processed by the shared local GNN layer. Subsequently, an average pooling operation is performed over the processed subgraphs associated with each node, resulting in a new graph G' that retains the original bipartite structure. G' is then passed through the MLP layer, yielding G'', which effectively integrates local information learned from the subgraphs. Finally, an element-wise sum of G'' and the original graph G is computed, and the resulting graph is processed by the global GNN layer to produce the circuit structure's prediction.

3) Incorporation of Symmetrical Constraints: Based on the aforementioned L2G-GNN model, we are able to identify substructures within the circuit. In our experiments, we apply symmetric constraint conditions to two specific structures in the circuit: the current mirror and the differential pair. Specifically, we impose the constraint that the length and width of the two transistors in symmetric positions must be equal.

C. Additive GP-Based Surrogate Model

A primary challenge in analog circuit sizing is the curse of dimensionality, where the performance of traditional BO [40] degrades significantly in high-dimensional space. To address this, our approach is based on the fact that an analog circuit sizing problem can be effectively optimized by leveraging its hierarchical structure [41], [42]. Specifically, we employ the Additive Gaussian Process (Add-GP) model [43] and utilize the specific hierarchical information of the circuit to address this problem.

The Add-GP model enables us to reformulate the high-dimensional objective function $f(\mathbf{x})$ as a sum of lower-dimensional components:

$$f(\mathbf{x}) = \sum_{i=1}^{P} f^{(i)}(\mathbf{x}^{(\mathcal{X}_i)}). \tag{4}$$

Here, $\mathbf{x}^{(\mathcal{X}_i)} \in \mathcal{X}_i$ are sub-variables in subspace $\mathcal{X}_i$, with $\bigcup_{i=1}^{P} \mathcal{X}_i = \mathcal{X}$. The number of partitions P equals the circuit's hierarchical levels (e.g., Rail-to-Rail Opamp includes first stage, second stage, and bias, $P = 3$, Figure 6(d)). Each subspace $\mathcal{X}_i$ contains parameters of one level (e.g., $\mathcal{X}_1$ represents the parameter in the first stage).

Each function $f(\mathbf{x})$ models a performance metric (such as gain, bandwidth) and consists of P independent GPs with mean $\mu^{(i)}$ and kernel $k^{(i)}$, so:

$$f(\mathbf{x}) \sim \mathrm{GP}\left(\sum_{i=1}^{P} \mu^{(i)}, \sum_{i=1}^{P} k^{(i)}\right). \tag{5}$$

For M performance metrics, $M \times P$ GPs are used.

The hyperparameters of the Add-GP model are optimized by maximizing the log marginal likelihood of the observed data $\mathbf{y}$, which is given by:

$$\mathcal{L} = -\frac{1}{2}\log|\Sigma| - \frac{1}{2}\mathbf{y}^T\Sigma^{-1}\mathbf{y} - \frac{n}{2}\log(2\pi), \tag{6}$$

with $\Sigma = K + \sigma^2 I$, $K = \sum_{i=1}^{N} k^{(i)}(\mathbf{x}_w^{(\mathcal{X}_i)}, \mathbf{x}_y^{(\mathcal{X}_i)})$. Posterior mean and variance at $\mathbf{x}^{*(\mathcal{X}_i)}$ are:

$$\begin{aligned} &\mu^{(i)}(\mathbf{x}^{*(\mathcal{X}_i)}) = k^{(i)}(\mathbf{x}^{*(\mathcal{X}_i)}, \mathbf{X}^{(\mathcal{X}_i)})^T\Sigma^{-1}\mathbf{y}, \\ &\sigma^2(\mathbf{x}^{*(\mathcal{X}_i)}) = k^{(i)}(\mathbf{x}^{*(\mathcal{X}_i)}, \mathbf{x}^{*(\mathcal{X}_i)}) \\ &\quad - k^{(i)}(\mathbf{x}^{*(\mathcal{X}_i)}, \mathbf{X}^{(\mathcal{X}_i)})^T\Sigma^{-1}k^{(i)}(\mathbf{X}^{(\mathcal{X}_i)}, \mathbf{x}^{*(\mathcal{X}_i)}). \end{aligned} \tag{7}$$

This approach mitigates the curse of dimensionality by splitting high-dimensional problems into the sum of low-dimensional problems. For circuits with lower dimensionality, there is no need to use an additive structure; instead, a standard GP is sufficient for modeling.

D. Stage-Adaptive Constrained Acquisition Function

In analog circuit sizing, we aim to minimize an objective performance while satisfying multiple constraints. We propose a stage-adaptive constrained acquisition function with two stages: (1) **feasible point search**, to quickly identify a point satisfying all constraints, and (2) **objective function minimization**, to optimize the objective from the feasible point.

Feasible Point Search Stage: If no initial samples satisfy the strict constraints, this stage starts. We focus on finding feasible points by maximizing each constraint's Probability of Improvement (PI):

$$\text{Maximize} \quad \Phi\left(\frac{C_1 - \mu_1(x)}{\sigma_1(x)}\right) \cdots \Phi\left(\frac{C_{n_c} - \mu_{n_c}(x)}{\sigma_{n_c}(x)}\right) \tag{8}$$

Where C_j is the constraint threshold, ensuring feasibility before objective optimization.

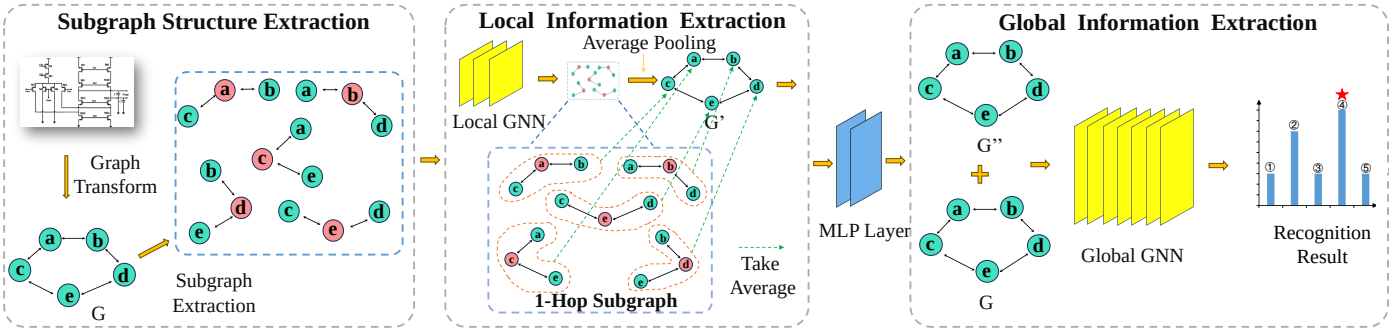


Fig. 3: The workflow of L2G-GNN.

Objective Function Minimization Stage: When the first feasible point is present in the initial dataset or obtained from the previous stage, we continue to optimize the objective function. The goal here is to simultaneously consider both the satisfaction of the constraint conditions and the improvement of the objective function:

$$\text{Maximize} \quad \sum_{j=1}^{nc}\left(\frac{\min(\mu_j(x), C_j)}{C_j} + \lambda\left(\frac{\min(\mu_j(x), r\cdot C_j)}{C_j} - 1\right)\right), \\ -\Phi\left(\frac{f_{\text{best}} - \mu_0(x)}{\sigma_0(x)}\right) \tag{9}$$

Where the first term balances constraint satisfaction, λ is a reward-punishment factor, r controls reward scaling, and the second term minimizes the objective by maximizing improvement over the current best value f_{best}. Multi-objective algorithms like NSGAII optimize these functions, achieving efficient circuit designs.

E. Dynamic Search Space Adjustment Strategy

Trapping into local optima due to a fixed search space is a key challenge in BO. To address this, we propose a strategy that dynamically changes the search space by adjusting its center.

After simulating a candidate point $\mathbf{x}_{\text{can}}$, we check if it satisfies constraints and improves the objective performance over f_0^{best}. If so, a better solution $\mathbf{x}_{\text{bet}}$ is found. The search space center is updated as:

$$\mathbf{x}_{\text{new}} = \mathbf{x}_{\text{cur}} + g \cdot \eta$$

where the gradient $g = \mathbf{x}_{\text{bet}} - \mathbf{x}_{\text{cur}}$ and η is dynamically adjusted using the following decay function:

$$\eta = \eta_{\min} + (\eta_{\max} - \eta_{\min})e^{-\gamma t} \tag{10}$$

Here, $\eta_{\min}$ and $\eta_{\max}$ are the step size bounds, γ is the decay factor, and t is the time step. As iterations progress, t increases, reducing the learning rate gradually. When a better solution $\mathbf{x}_{\text{bet}}$ is found, t resets to 0.

The new search space is defined as $\mathcal{X} = (\mathbf{x}_l, \mathbf{x}_u) = (\mathbf{x}_{\text{new}} \times (1-r), \mathbf{x}_{\text{new}} \times (1+r))$, where $r = 50\%$ sets the fluctuation range.

MOSTAR uses Add-GP as a surrogate model, optimizes with a stage-adaptive constraint acquisition function, and adaptively updates the search space using a dynamic search space adjustment strategy. The overall workflow of MOSTAR is shown in Algorithm 1.

Algorithm 1 MOSTAR Algorithm

Input: Design space $\mathcal{X}$, initial data $\mathcal{D}$, iterations N, hierarchy number P.
Output: Best sizing solution
for $i = 1$ **to** N **do**
  Partition $\mathcal{D}$ into P subsets: $\mathcal{D} = \bigcup_{j=1}^{P} \mathcal{D}^{(j)}$, where $\mathcal{D}^{(j)} = \{\mathbf{X}^{(\mathcal{X}_j)}, \mathbf{Y}\}$
  for $j = 1$ **to** P **do**
    Train M sub-GPs $SubGP_k^{(j)}$ on $\mathcal{D}^{(j)}$, $(k \in [1, M])$
    if feasible point in $\mathcal{D}$ exists **then**
      $\mathbf{x}^{(\mathcal{X}^j)} \leftarrow \arg\max$(Acq. Func. 2, Eq. 9) in $\mathcal{X}_j$
    else
      $\mathbf{x}^{(\mathcal{X}^j)} \leftarrow \arg\max$(Acq. Func. 1, Eq. 8) in $\mathcal{X}_j$
    end if
  end for
  Combine P points $\{\mathbf{x}^{(\mathcal{X}^j)}\}_{j=1}^{P}$ to get candidate $\mathbf{x}_i$.
  Simulate $\mathbf{x}_i$ to get observation $\mathbf{y}_i$.
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{x}_i, \mathbf{y}_i)\}$
  if $\mathbf{x}_i$ is feasible and $f_0(\mathbf{x}_i) < f_0^{best}$ **then**
    $\mathcal{X} \leftarrow$ Dynamic Search Space Adjustment Strategy
    $f_0^{best} \leftarrow f_0(\mathbf{x}_i)$ (Update the current best value)
  end if
end for
return Best sizing solution in $\mathcal{D}$.

IV. Experimental Results

In this section, we validate our proposed algorithm through three experiments. First, we verify the L2G-GNN's accuracy on substructure identification. Second, an ablation study evaluates our dynamic search space and symmetry-handling strategies. Finally, five sizing experiments on three basic and two high-dimensional circuits demonstrate the algorithm's effectiveness and scalability. All experiments are performed on a CentOS workstation (Intel Xeon Gold 5218R CPU, 128GB RAM) using Python/GPytorch, and all circuits are built based on the TSMC 65nm library.

A. Circuit Substructure Identification Experiment

To evaluate the efficiency of our proposed L2G-GNN model in identifying substructures, we conduct a dedicated experiment focused on circuit substructure identification. In this experiment, the recognition accuracy of L2G-GNN is compared against several GNN architectures, namely GCN [36], DeepGCN, GraphSAGE [37], and CR-GCN [38].

The experiment is performed using a custom-built dataset comprising 1870 test circuits, meticulously designed based on established circuit schematics and informed by authoritative resources such as [44], [45], [46], [47], and other pertinent literature.The dataset contains various analog circuits, including

1359 opamps, 176 LDOs, 324 bandgap references, and 11 other circuit types. The netlist dataset is divided into an 80% training dataset and a 20% testing dataset.

The circuit substructure identification results are summarized in Table I and the prediction accuracy curves are shown in Figure 4, revealing that our L2G-GNN achieves the highest accuracy at 97.22%. This significantly outperforms GCN at 87.48%, DeepGCN at 90.64%, and the lower accuracies of GraphSAGE at 67.15% and CR-GCN at 65.40%, clearly indicating L2G-GNN's superior performance in this task.

TABLE I: The accuracy comparison of different GNN models.

Model	GCN[36]	DeepGCN	GraphSAGE[37]	CR-CRN[38]	**L2G-GNN**
Accuracy	87.48%	90.64%	67.15%	65.40%	**97.22%**

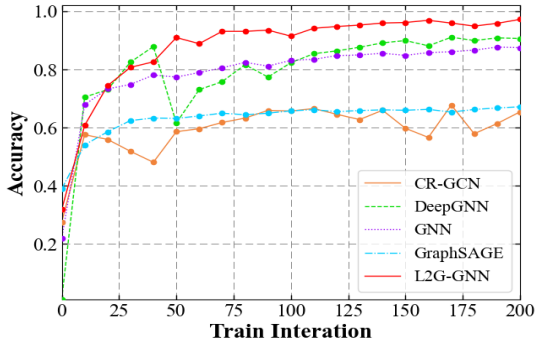


Fig. 4: Accuracy curves comparison of different GNN models.

B. Ablation Study of DSSAS and SC

To assess the individual benefits of the dynamic search space adjustment strategy (DSSAS) and the incorporation of symmetrical constraints (SC) within the MOSTAR framework, we perform an ablation study on a two-stage circuit. The results are shown in Figure 5. It is observed that the complete MOSTAR method, integrating DSSAS and SC, exhibits the best optimization performance, achieving the lowest predicted current. The complete MOSTAR method achieves a 1.69× improvement compared to the version without DSSAS. Notably, the complete MOSTAR achieves a 13.14× performance improvement compared to the version without SC. In summary, the experimental data clearly support the effectiveness of DSSAS and SC in achieving optimized analog circuit sizing.

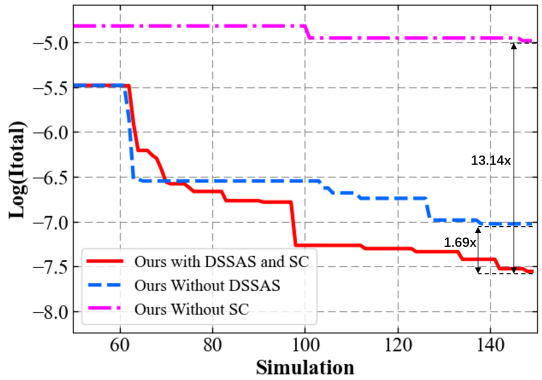


Fig. 5: Results of ablation experiments on DAASA and SC.

C. Circuit Sizing Experiments

We evaluate our algorithm's sizing performance on a diverse suite of five circuits—three basic and two high-dimensional—benchmarking it against several baseline methods including NSGAII [9], MOEA/D [10], PVTSizing [14], MACE [19], and ABCMOBO [22]. The circuit topologies and terget performance comparisons are shown in Figure 6(a) to Figure 6(j), with final results in Table II. For this analysis, improvement is defined as the ratio of the worst-performing model's objective function value to the specific model's value.

The design parameters and constraints for each circuit are as follows: For a two-stage operational amplifier (opamp), the constraints are a voltage gain (Gain) $>$ 40 dB, gain-bandwidth product (GBW) $>$ 5 MHz, and phase margin (PM) $>$ 60°. The goal is to minimize current by optimizing 28 parameters, using 50 initial samples and 100 optimization simulations.

For a three-stage opamp, the constraints are a gain $>$ 90 dB, GBW $>$ 2 MHz, and PM $>$ 60°. The goal is to minimize current by optimizing 25 parameters, using 100 initial samples and 100 optimization simulations.

For a bandgap circuit, the constraints are a TC $<$ 30 ppm/°C and current $<$ 55 μA. The goal is to minimize output noise by optimizing 36 parameters, using 100 initial samples and 100 optimization simulations.

For a rail-to-rail opamp, the constraints are a gain $>$ 50 dB, GBW $>$ 5 MHz, PM $>$ 60°, and input/output swing $>$ 3 V. The goal is to minimize current by optimizing 94 parameters, using 100 initial samples and 150 optimization simulations.

For a gain-boosted opamp, the constraints are a gain $>$ 100 dB, GBW $>$ 10 MHz, and PM $>$ 60°. The goal is to minimize current by optimizing 114 parameters, using 100 initial samples and 200 optimization simulations.

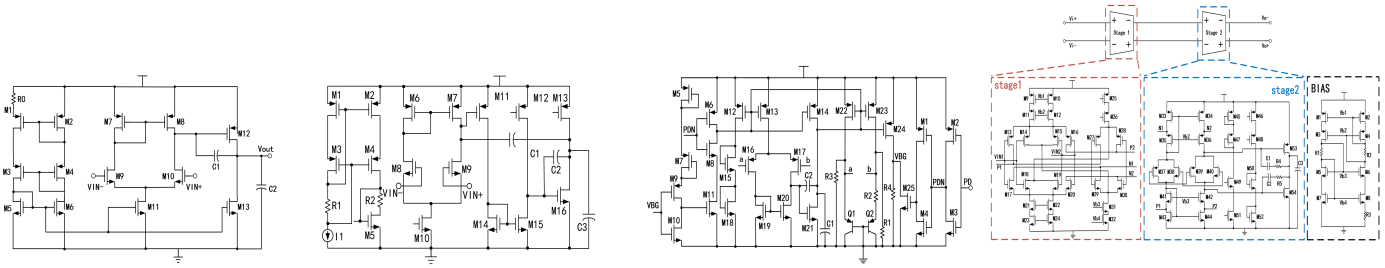
Add symmetry constraint: Following structure identification by L2G-GNN, symmetry constraints are applied to all five circuits, targeting components like current mirrors and differential pairs. We add 10, 6, 4, 44, and 52 constraint pairs for the two-stage opamp, three-stage opamp, bandgap circuit, rail-to-rail opamp, and gain-boosted opamp, respectively. For fair comparison, all baseline algorithms incorporate these same symmetry constraints.

Optimization result analysis: In the two-stage opamp, MOSTAR demonstrates superiority by achieving the lowest current consumption, with a 4.124× improvement compared to MOEAD. The improvements of the other methods are as follows: 2.290× for NSGA-II, 2.678× for PVTSizing, 1.921× for MACE, and 2.040× for ABCMOBO.

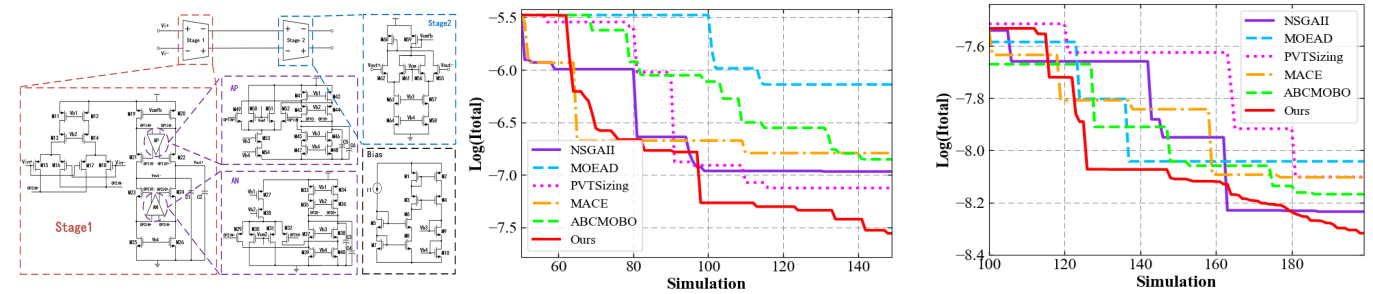
In the three-stage opamp, MOSTAR maintains its leading position with the lowest current, attaining a 1.317× improvement over MOEAD. The improvements of the other methods are: 1.213× for NSGA-II, 1.061× for PVTSizing, 1.063× for MACE, and 1.134× for ABCMOBO.

In the bandgap circuit, MOSTAR achieves the lowest noise, showing a 1.039× improvement over NSGA-II, which is taken as the reference. The improvements of the other methods are: 1.030× for MOEAD, 1.011× for PVTSizing, 1.026× for MACE, and 1.009× for ABCMOBO.

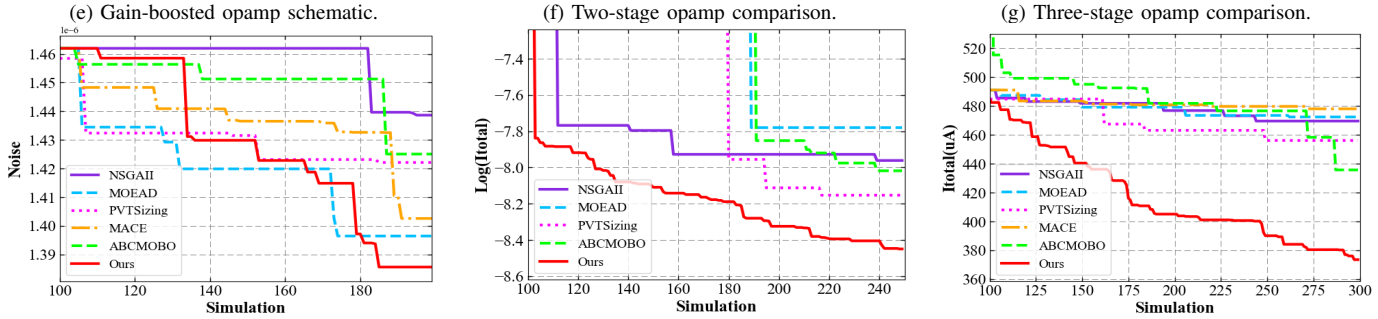
In the rail-to-rail opamp, MOSTAR takes the lead with the lowest current, resulting in a 1.955× improvement over MOEAD. PVTSizing shows a notable 1.452× improvement,



(a) Two-stage opamp. (b) Three-stage opamp. (c) Bandgap circuit. (d) Rail-to-rail opamp schematic.



(e) Gain-boosted opamp schematic. (f) Two-stage opamp comparison. (g) Three-stage opamp comparison.



(h) Bandgap circuit comparison. (i) Rail-to-rail opamp comparison. (j) Gain-boosted opamp comparison.

Fig. 6: Schematics and comparison results of five module-based circuits.

TABLE II: The optimization results and statistics for all five circuits, arranged in two sections for improved readability.

Method	**2-Stage OpAmp** (16 Dev, 28 Param)						**3-Stage OpAmp** (21 Dev, 25 Param)						**Bandgap Circ.** (33 Dev, 36 Param)				
	Gain	GBW	PM	I/μA	I(mean)	Improv.	Gain	GBW	PM	I/μA	I(mean)	Improv.	TC	I	No	No(mean)	Improv.
Constraint	>40dB	>5M	>60°	Min	Min	–	>90dB	>2M	>60°	Min	Min	–	<30	<55μA	Min	Min	–
NSGA II [9]	46.86	6.08	66.51	763	941	2.29×	105.5	5.41	69.88	255	265	1.21×	23.25	50.28	1.46e-6	1.44e-6	1.00×
MOEAD [10]	41.41	5.31	66.41	2046	2156	1.00×	109.4	3.65	71.90	339	321	1.00×	23.26	44.43	1.40e-6	1.40e-6	1.03×
PVTSizing [14]	41.79	5.34	72.73	872	805	2.68×	112.0	6.51	61.49	294	302	1.06×	28.80	46.66	1.43e-6	1.42e-6	1.01×
MACE [19]	42.51	6.85	61.23	1055	1122	1.92×	108.3	5.73	60.15	291	302	1.06×	27.22	50.96	1.40e-6	1.40e-6	1.03×
ABCMOBO [22]	41.08	5.13	70.19	1124	1057	2.04×	110.7	4.46	66.66	287	283	1.13×	25.45	39.73	1.44e-6	1.43e-6	1.01×
MOSTAR	41.10	5.40	65.56	**495**	**523**	**4.13**×	97.25	9.13	84.77	**239**	**244**	**1.32**×	29.69	51.19	**1.38e-6**	**1.38e-6**	**1.04**×

Method	**R-to-R OpAmp** (62 Dev, 94 Param)							**Gain-Boost OpAmp** (70 Dev, 114 Param)					
	Gain	GBW	PM	In/Out	I/μA	I(mean)	Improv.	Gain	GBW	PM	I/μA	I(mean)	Improv.
Constraint	>50dB	>5M	>60°	>3	Min	Min	–	>100dB	>10M	>60°	Min	Min	–
NSGA II [9]	53.82	7.69	75.91	3.00	307	348	1.20×	101.5	12.09	79.63	463	470	1.02×
MOEAD [10]	53.77	9.65	72.94	3.08	356	418	1.00×	100.4	13.64	76.50	477	472	1.01×
PVTSizing [14]	54.12	21.55	74.24	3.13	302	288	1.45×	100.9	18.17	69.05	464	456	1.05×
MACE [19]				*fail*				101.0	14.09	61.58	472	478	1.00×
ABCMOBO [22]	57.87	19.45	61.04	3.15	230	329	1.27×	108.8	11.64	61.05	445	441	1.09×
MOSTAR	67.61	8.16	65.00	3.18	**201**	**214**	**1.95**×	104.8	17.88	71.58	**379**	**374**	**1.28**×

followed by ABCMOBO (1.269×) and NSGA-II (1.199×). MACE failed in this optimization task.

In the gain-boosted opamp, MOSTAR again secures the lowest current, with a 1.280× improvement over MACE (the reference). The improvements of the remaining methods are: 1.085× for ABCMOBO, 1.048× for PVTSizing, 1.018× for NSGA-II, and 1.012× for MOEAD.

V. Conclusion

In this paper, we propose MOSTAR, a multi-stage hierarchical Bayesian optimization framework integrating L2G-GNN for substructure identification and symmetry constraints. MOSTAR improves scalability in high-dimensional scenarios with Add-GP and stage-adaptive constrained acquisition function, and avoids local optima using a dynamic search space adjustment strategy. Experiments show L2G-GNN achieves 97.22% substructure identification accuracy, while MOSTAR yields 1.04×–4.13× optimization improvements across three basic and two high-dimensional circuits, demonstrating potential for high-dimensional analog circuit sizing.

VI. Acknowledgement

This work was supported in part by the Natural Science Foundation of Beijing, China (Grant No. Z230002) and 111 project (B18001).

References

[1] R. Phelps *et al.*, "Anaconda: simulation-based synthesis of analog circuits via stochastic pattern search," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, pp. 703–717, 2002.

[2] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-international conference on neural networks*, vol. 4. ieee, 1995, pp. 1942–1948.

[3] V. Feoktistov, *Differential evolution*. Springer, 2006.

[4] P. I. Frazier, "A tutorial on bayesian optimization," *arXiv preprint arXiv:1807.02811*, 2018.

[5] B. Liu *et al.*, "Efficient multi-objective synthesis for microwave components based on computational intelligence techniques," in *Proceedings of the 49th annual design automation conference*, 2012, pp. 542–548.

[6] W. Lyu *et al.*, "Multi-objective bayesian optimization for analog/rf circuit synthesis," in *Proceedings of the 55th Annual Design Automation Conference*, 2018, pp. 1–6.

[7] M. Fakhfakh, "A novel alienor-based heuristic for the optimal design of analog circuits," *Microelectronics Journal*, vol. 40, pp. 141–148, 2009.

[8] M. Fakhfakh *et al.*, "A novel heuristic for multi-objective optimization of analog circuit performances," *Analog integrated circuits and signal processing*, vol. 61, pp. 47–64, 2009.

[9] K. Deb *et al.*, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, pp. 182–197, 2002.

[10] Q. Zhang and H. Li, "Moea/d: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on evolutionary computation*, vol. 11, pp. 712–731, 2007.

[11] Z. Zhao and L. Zhang, "Deep reinforcement learning for analog circuit sizing," in *2020 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2020, pp. 1–5.

[12] N. K. Somayaji *et al.*, "Prioritized reinforcement learning for analog circuit optimization with design knowledge," in *2021 58th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2021, pp. 1231–1236.

[13] H. Wang *et al.*, "Gcn-rl circuit designer: Transferable transistor sizing with graph neural networks and reinforcement learning," in *2020 57th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2020, pp. 1–6.

[14] Z. Kong *et al.*, "Pvtsizing: A turbo-rl-based batch-sampling optimization framework for pvt-robust analog circuit synthesis," in *Proceedings of the 61st ACM/IEEE Design Automation Conference*, 2024, pp. 1–6.

[15] S. Amini *et al.*, "Constrained bayesian optimization: A review," *IEEE Access*, 2024.

[16] V. Perrone *et al.*, "Constrained bayesian optimization with max-value entropy search," *arXiv preprint arXiv:1910.07003*, 2019.

[17] S. Daulton *et al.*, "Parallel bayesian optimization of multiple noisy objectives with expected hypervolume improvement," *Advances in Neural Information Processing Systems*, vol. 34, pp. 2187–2200, 2021.

[18] S. Belakaria *et al.*, "Uncertainty aware search framework for multi-objective bayesian optimization with constraints," *arXiv preprint arXiv:2008.07029*, 2020.

[19] S. Zhang *et al.*, "An efficient batch-constrained bayesian optimization approach for analog circuit synthesis via multiobjective acquisition ensemble," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, pp. 1–14, 2021.

[20] S. Daulton *et al.*, "Multi-objective bayesian optimization over high-dimensional search spaces," in *Uncertainty in Artificial Intelligence*. PMLR, 2022, pp. 507–517.

[21] K. Touloupas and P. P. Sotiriadis, "Locomobo: A local constrained multiobjective bayesian optimization for analog circuit sizing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, pp. 2780–2793, 2021.

[22] X. Zhao *et al.*, "Asynchronous batch constrained multi-objective bayesian optimization for analog circuit sizing," in *2024 29th Asia and South Pacific Design Automation Conference (ASP-DAC)*. IEEE, 2024, pp. 872–877.

[23] S. Daulton *et al.*, "Differentiable expected hypervolume improvement for parallel multi-objective bayesian optimization," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9851–9864, 2020.

[24] A. F. Budak *et al.*, "Practical layout-aware analog/mixed-signal design automation with bayesian neural networks," in *2023 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*. IEEE, 2023, pp. 1–8.

[25] B. Shahriari *et al.*, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, vol. 104, pp. 148–175, 2015.

[26] E. Brochu *et al.*, "A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," *arXiv preprint arXiv:1012.2599*, 2010.

[27] C. Williams and C. Rasmussen, "Gaussian processes for regression," *Advances in neural information processing systems*, vol. 8, 1995.

[28] M. Kanagawa *et al.*, "Gaussian processes and kernel methods: A review on connections and equivalences," *arXiv preprint arXiv:1807.02582*, 2018.

[29] K. Kunal *et al.*, "Gana: Graph convolutional network based automated netlist annotation for analog circuits," in *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2020, pp. 55–60.

[30] K. Kunal *et al.*, "Align: Open-source analog layout automation from the ground up," in *Proceedings of the 56th Annual Design Automation Conference 2019*, 2019, pp. 1–4.

[31] Z. Dong *et al.*, "Cktgnn: Circuit graph neural network for electronic design automation," *arXiv preprint arXiv:2308.16406*, 2023.

[32] Z. Tao *et al.*, "Amsnet: Netlist dataset for ams circuits," in *2024 IEEE LLM Aided Design Workshop (LAD)*. IEEE, 2024, pp. 1–5.

[33] K. Kunal *et al.*, "Gnn-based hierarchical annotation for analog circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 42, pp. 2801–2814, 2023.

[34] Y. Yamakaji *et al.*, "Circuit2graph: Circuits with graph neural networks," *IEEE Access*, 2024.

[35] Z. Wu and I. Savidis, "Circuit-gnn: A graph neural network for transistor-level modeling of analog circuit hierarchies," in *2023 IEEE International Symposium on Circuits and Systems (ISCAS)*. IEEE, 2023, pp. 1–5.

[36] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[37] W. Hamilton *et al.*, "Inductive representation learning on large graphs," *Advances in neural information processing systems*, vol. 30, 2017.

[38] V. Pathak *et al.*, "Analog circuit classification using graph convolution networks," in *2024 20th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*. IEEE, 2024, pp. 1–4.

[39] P. Velickovic *et al.*, "Graph attention networks," *stat*, vol. 1050, pp. 10–48 550, 2017.

[40] B. Shahriari *et al.*, "Taking the human out of the loop: A review of bayesian optimization," *Proceedings of the IEEE*, vol. 104, pp. 148–175, 2015.

[41] I. Abel and H. Graeb, "Fuboco: Structure synthesis of basic op-amps by functional block composition," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 27, pp. 1–27, 2022.

[42] D. Stefanovic and M. Kayal, *Structured analog CMOS design*. Springer Science & Business Media, 2008.

[43] K. Kandasamy *et al.*, "High dimensional bayesian optimisation and bandits via additive models," in *International conference on machine learning*. PMLR, 2015, pp. 295–304.

[44] B. Razavi, *Design of analog CMOS integrated circuits*, 2005.

[45] T. C. Carusone *et al.*, *Analog integrated circuit design*. John Wiley & Sons, 2011.

[46] H. Camenzind, *Designing analog chips*. Virtualbookworm Publishing, 2005.

[47] P. E. Allen and D. R. Holberg, *CMOS analog circuit design*. Elsevier, 2011.