AutoDMP: Automated DREAMPlace-based Macro Placement

Anthony Agnesina
aagnesina@nvidia.com
NVIDIA Corporation
Austin, TX, USA

Puranjay Rajvanshi
prajvanshi@nvidia.com
NVIDIA Corporation
Santa Clara, CA, USA

Tian Yang
tiyang@nvidia.com
NVIDIA Corporation
Santa Clara, CA, USA

Geraldo Pradipta
gpradipta@nvidia.com
NVIDIA Corporation
Santa Clara, CA, USA

Austin Jiao
ajiao@nvidia.com
NVIDIA Corporation
Santa Clara, CA, USA

Ben Keller
benk@nvidia.com
NVIDIA Corporation
Santa Clara, CA, USA

Brucek Khailany
bkhailany@nvidia.com
NVIDIA Corporation
Austin, TX, USA

Haoxing Ren
haoxingr@nvidia.com
NVIDIA Corporation
Austin, TX, USA

ABSTRACT

Macro placement is a critical very large-scale integration (VLSI) physical design problem that significantly impacts the design power-performance-area (PPA) metrics. This paper proposes AutoDMP, a methodology that leverages DREAMPlace, a GPU-accelerated placer, to place macros and standard cells concurrently in conjunction with automated parameter tuning using a multi-objective hyperparameter optimization technique. As a result, we can generate high-quality predictable solutions, improving the macro placement quality of academic benchmarks compared to baseline results generated from academic and commercial tools. AutoDMP is also computationally efficient, optimizing a design with 2.7 million cells and 320 macros in 3 hours on a single NVIDIA DGX Station A100. This work demonstrates the promise and potential of combining GPU-accelerated algorithms and ML techniques for VLSI design automation.

CCS CONCEPTS

• **Hardware → Placement**; • **Computing methodologies → Graphics processors**; **Artificial intelligence**.

KEYWORDS

VLSI placement, GPU acceleration, AI/ML

ACM Reference Format:
Anthony Agnesina, Puranjay Rajvanshi, Tian Yang, Geraldo Pradipta, Austin Jiao, Ben Keller, Brucek Khailany, and Haoxing Ren. 2023. AutoDMP: Automated DREAMPlace-based Macro Placement. In *Proceedings of the 2023 International Symposium on Physical Design (ISPD '23), March 26–29, 2023, Virtual Event, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3569052.3578923

1 INTRODUCTION

Modern digital chips integrate large numbers of macros, such as SRAMs and clock generators. These objects are typically much larger than the more numerous standard cells, which are the fundamental building blocks of digital designs. Thus, macros deserve special treatment due to their large effect on the floorplan of the chip, impacting many competing design objectives such as wirelength, power, and area. Traditionally, high-quality macro placements are obtained as part of floorplanning, decoupling the placement problem into first placing macros alone through manual or algorithmic effort, followed by placing the standard cells independently.

Recently, mixed-size placement, which places macros and standard cells simultaneously, has shown promising results compared to the two-step approach. The unified global view of the placeable objects can unlock new optimal locations for the macros for the multi-objective placement optimization problem. However, macro legalization is challenging for mixed-size placement, especially when many macros are tightly packed. Furthermore, expanding the design space can also increase the sub-optimality gap, thus requiring new effective and efficient design space exploration techniques.

In this work, we propose an open-source methodology called Automated DREAMPlace-based Macro Placement (*AutoDMP*) to find better macro placement solutions by efficiently searching a vast design space using ML-based multi-objective optimization and analytical mixed-size placers accelerated by graphics processing units (GPUs). The key contributions are summarized as follows:

- We propose using multi-objective Bayesian optimization to search the design space of macro placements efficiently. This design space is explored by tuning the parameters of a GPU-accelerated mixed-size placer and targeting three high-level power-performance-area (PPA) proxy objectives post-place, namely wirelength, cell density, and congestion.
- We propose a two-level PPA evaluation scheme to manage the complexity of the search space. Only the macro placements obtained during the search that are Pareto-optimal for the proxies are evaluated inside a commercial electronic design automation (EDA) tool with more advanced figures of merit obtained after timing optimization and routing.
- We propose enhancements to the mixed-size placement engine, the open-source analytical DREAMPlace [31] placer, to reduce legalization issues and significantly expand its design space, thereby increasing the potential achievable PPA.
- In a few hours on an NVIDIA DGX Station, we can generate various viable macro placements corresponding to different Pareto points, the quality of which is comparable to those designed by commercial tools. We demonstrate the best routed wirelength and timing results of any open-source tool on the open-source TILOS benchmarks [49].

The source code is released on GitHub[1].

[1]https://github.com/NVlabs/AutoDMP

ISPD '23, March 26–29, 2023, Virtual Event, USA
© 2023 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-9978-4/23/03.
https://doi.org/10.1145/3569052.3578923

2 RELATED WORK

Solving the macro placement problem has a rich and continuing history. We consider prior efforts in two broad categories: floorplanning, which fixes macro locations and then places standard cells separately, and mixed-size placement, which places both simultaneously.

Floorplanning addresses the problem of efficiently packing hard macros and soft clusters of standard cells to minimize wirelength and area. The floorplanning approach can be further divided into the time-honored simulated annealing and partitioning methods and the recently proposed reinforcement learning (RL)-based method.

Simulated annealing is widely applied in the traditional floorplan literature. The macro placement is represented by efficient data structures, such as sequence pairs [40], corner block lists [20], or tree-like structures [19, 35], notably B*-trees [7]. A new method [27] uses traditional sequence pairs but tries to mimic the behavior of human experts with a handcrafted objective function, exploiting logical hierarchy and dataflow for standard cell clustering. Simulated annealing is very flexible but suffers from poor scalability. Fast [9] and multilevel [10] approaches have been proposed to improve its scalability. The partitioning method performs top-down partitioning to construct the floorplan. Legalization is a crucial issue of the partitioning method. Initially, low-temperature annealing [44] and row-based legalization [45] were proposed to legalize the macros after partitioning. Later, look-ahead legalization [18] guaranteed legalizable partitions, and [51] deferred the decision-making when reconstructing partitions bottom up. The RL approach from Google, known as CircuitTraining [39], views the macro placement problem as a game where actions involve placing macros at discrete locations on a gridded canvas. The standard cells are clustered and placed with quadratic force-directed placement after macro placement. The proxy cost of the placement uses three coarse metrics, namely half-perimeter wirelength (HPWL), density, and congestion. Without considering the timing or global and detailed placements of standard cells, CircuitTraining achieves impressive macro placements, yielding solutions with competitive PPA to commercial solutions. However, the training and fine-tuning process is often long and consumes many hardware resources, i.e., multiple days and thousands of CPUs.

State-of-the-art mixed-size placers usually combine floorplanning techniques to pack and legalize macros and analytical placement techniques to handle millions of small standard cells. Simultaneous flows place macros and standard cells together, incorporating macro handling methods such as shredding [1, 5], shifting [11], and re-legalization [6]. During the global analytical placement, wirelength optimization and non-overlapping constraints are considered simultaneously as a nonlinear optimization problem [22, 26]. However, legalization is a fundamental challenge for the simultaneous method, as many overlaps remain after the placement stage. This issue is exacerbated in modern heterogeneous system-on-chip (SoC) designs in which a large proportion of the die is occupied by macros. Sometimes, a feasible placement can only be found by severely degrading the wirelength. On the other hand, sequential flows tentatively place macros and standard cells together, where standard cells might be clustered into soft blocks [52] to improve speed, discard the standard cell placement and optimize the macro placement only [8, 12]. Then a standard-cell placement with fixed macros is performed using established methods. This approach, widely used in the industry, is often more robust, especially when considering the effect of the power-ground (P/G) grid blockages, since it can guarantee feasible macro placements. Commercial EDA tools can execute both simultaneous and sequential flows.

3 PRELIMINARIES

DREAMPlace accelerated ePlace/RePlace [13, 37] algorithms with GPUs to produce state-of-the-art global placement quality, while ABCDPlace [32] implemented traditional sequential detailed placement techniques concurrently with GPUs. DREAMPlace formulates the global placement problem as a wirelength minimization problem under density constraints and solves it numerically through classical mathematical optimization (e.g., gradient descent) of the nonlinear unconstrained formulation:

$$\min_{\mathbf{x},\mathbf{y}} \sum_{e \in E} w_e \, \mathrm{WL}(e; \mathbf{x}, \mathbf{y}) + \lambda D(\mathbf{x}, \mathbf{y}), \tag{1}$$

where E is the set of nets, and $(\mathbf{x}, \mathbf{y})$ are the cell locations. The $\mathrm{WL}(e; \cdot)$ term is a smooth version of the HPWL of net e, and the smooth density function $D(\cdot)$ is computed as the potential energy of an electrostatic system where cells are modeled as charges. It is computed by solving Poisson's equation via spectral methods with a two-dimensional fast Fourier transform (FFT). The net weights w_e can be used to model timing constraints [30], for example. In addition, the Lagrange multiplier λ is progressively increased to ensure no overlaps among cells. Finally, DREAMPlace computes both wirelength and density gradients numerically using GPU-accelerated algorithms enabled by the PyTorch framework.

DREAMPlace can be used as a mixed-size placer, as it sees the global placement of macros and standard cells as the same problem from an optimization viewpoint. However, while it excels at solving the optimization problem, the current algorithm cannot guarantee that placements will be legalizable for designs with many macros. In addition, the placer's parameters and random seeding heavily influence the optimization's convergence and final objective value, resulting in very brittle and unpredictable results.

Sequential model-based optimization techniques [23] have been applied to EDA tools and flow parameter tuning [34, 50]. These techniques learn a surrogate model to predict performance and iterate between fitting a model and gathering additional data based on this model. Furthermore, they are known to be sample efficient. One such method is the tree-structured Parzen estimator (TPE) [4]. The TPE builds distributions of the "good" (G) and "poor" samples (P), where samples are classified based on their position in the objective space relative to the current Pareto front. The algorithm collects multiple random samples before building the internal distributions. These distributions follow the non-parametric multivariate density estimation model called the Parzen window [43]. The good distribution G helps draw many candidates, and the TPE picks the one with the greatest expected improvement $\propto G(x)/P(x)$. The benefit of TPE is that it can handle both discrete and continuous-valued parameters. The multi-objective TPE [42] extends the single-objective TPE to handle a multi-objective space with multi-dimensional partitioning.

4 AUTODMP FRAMEWORK

Our framework relies on high-level PPA proxies to guide the exploration of the macro placement space. These proxies are extracted after the legalization and detailed placement performed by ABCDPlace on DREAMPlace's simultaneous mixed-size global placement. We propose enhancements to the DREAMPlace engine and novel parameters to solve the macro legalization issues and expand the design space. We propose a two-level methodology with multi-objective Bayesian optimization to orchestrate the exploration of the vast placement space and achieve increased solution quality.

4.1 PPA Proxies

The placement significantly affects the results of the downstream physical design flow and often restricts numerous later stages, such as timing optimization and routing [25, 38]. Being central to the flow, many desirable characteristics during placement are antithetical. These include routability, timing, power consumption, or production cost. However, how to decide when each objective should matter during placement is still an open question [25]. Optimizing many goals at once during placement might be unproductive, so we instead focus on high-level cardinal placement objectives: wirelength, density, and congestion. CircuitTraining demonstrated the effectiveness of these metrics as PPA proxies. Moreover, our experiments show that timing and power optimization availability stems to a degree from these metrics. Furthermore, these objectives are relatively smooth, predictable, and can be very quickly estimated—an aspect of utmost importance to enable efficient exploration. In contrast, timing or total power is much more challenging to model, given many lengthy black-box downstream steps such as timing optimization, clock tree synthesis (CTS), and routing.

4.1.1 Wirelength. We use the rectilinear Steiner minimum tree (RSMT) as the first metric to judge the quality of a detailed placement. The RSMT achieves minimum wirelength to connect pins using 2D rectilinear edges only and is abundantly used by EDA tools during global and detailed routing. The RSMT is a loose lower bound of a net's routed wirelength due to routing effects to satisfy complex design rules and meet performance goals. However, without access to an actual router, the long-running algorithms of which can be chaotic, the RSMT combined with density and congestion can serve as a sufficiently accurate estimator of the wirelength. We compute the RSMTs of individual nets with FLUTE [16], a fast heuristic that can generate near-optimal wirelength Steiner trees using lookup tables.

4.1.2 Cell Density. Congestion-driven placers often decrease cell density where pin density is high, trading congestion for density. In DREAMPlace, the ePlace-based density formulation is used to optimize the system's potential energy assimilating cells with positive charges in an electrostatic analogy. The stable point of the potential energy corresponds to a uniform cell density. The optimization of the DREAMPlace objective in Equation 1 stops when the total electric overflow is under a certain threshold $T = 0.07$,

$$\text{Overflow} = \frac{\sum_{b \in B} \max(D_b(\mathbf{x}, \mathbf{y}) - A_b \cdot d_{\text{target}}, 0) A_b}{\sum_{c \in V_{mov}} A_c} \leq T, \tag{2}$$

where the floorplan region is decomposed into rectangular bins B, A_b and A_c are the area of bin b and cell c; D_b is the cell density of bin b, i.e., the sum of overlap areas between bin b and all cells; d_{target} is the target cell density, and V_{mov} is the subset of movable cells. While standard cells have a density/porosity close to 1 when calculating D_b, the macros get assigned a porosity of d_{target}. Thus, the final placement can reach a uniform density equal to d_{target} over the whole chip canvas. However, because the effective utilization is below d_{target}, fillers must be added to achieve the target density by filling the gaps between standard cells without excessively distributing the cells over all bins.

4.1.3 Congestion. Congestion is a traditional proxy for routability during placement, which is key to achieving predictable timing closure. Congested designs after the placement are more likely to have timing problems during the subsequent steps, especially during sign-off closure. We use a routing demand estimation technique called the rectangular uniform wire density (RUDY) [48] to model congestion, extended to consider the limited routing resources over macros. This is especially important when macros can be placed anywhere in the floorplan area, as nets may cross the macros and need to be routed over the macros. Because the RUDY map does not account for the fact that routing can be alleviated in neighborhood gcells, we smooth it by applying a Gaussian Blur kernel and estimate the congestion score as an average of the congestion of the top-10% most congested bins.

4.2 DREAMPlace Extensions

We propose extensions to DREAMPlace to improve the placement estimation and quality.

4.2.1 RUDY with Macro Blockages. The RUDY estimation provides a two-dimensional utilization map that can be readily and quickly computed after placement. However, the RUDY implementation inside DREAMPlace does not consider the obstructions of macros, which can force nets to be detoured outside the macro boundaries. Using the number of routing layers and minimum metal pitches, we compute the horizontal and vertical routing supplies $s(g)_{H/V}$ of a gcell g. We define per macro $m \in M$ a unit routing demand $\alpha(m)_{H/V} = \text{o}(m)_{H/V}/\text{a}(m)$, where $a(m)$ is the area of macro m, and $\text{o}(m)$ is the routing supply used by macro m's obstructions. Finally, the RUDY congestion map is obtained as

$$\text{RUDY}(g)_{H/V} = \frac{\sum_{e \in E} \frac{\text{OA}(e,g)}{\text{BBOX}(e)_{V/H}}}{\text{s}(g)_{H/V} - \sum_{m \in M} \alpha(m)_{H/V}\, \text{OA}(m, g)}, \tag{3}$$

where OA is the overlap area of g with macro m or the bounding box BBOX(e) of net e.

4.2.2 RISA Net Weights. Because the wirelength proxy is the RSMT, not HPWL, we use a weighted wirelength during global and detailed placements to increase correlation with Steiner routing. The net weights are obtained from RISA [15], where the RSMT is approximated based on the cardinality of the point set, and weights w_e are obtained through statistical simulations on random points and exact RSMT construction.

4.2.3 Gradient Descent Optimizer. The objective of Equation 1 is optimized in DREAMPlace through gradient descent (GD). To improve the convergence's success rate, we modify the initial learning rate computation, of the original form

$$\mathrm{lr}_{init} = \frac{\|z_0 - z_1\|_2}{\|\nabla f(z_1) - \nabla f(z_0)\|_2}, \quad z_1 = z_0 - \mathrm{lr}_0 \cdot \nabla f(z_0), \qquad (4)$$

where $z = (\mathbf{x}, \mathbf{y})$, f is the target minimization objective of Equation 1, lr_0 the user's initial learning rate parameter, and the gradient is preconditioned with the Jacobi diagonal method. We observed that this initialization method often resulted in null or anomalously large gradients at the early stages of the optimization, usually due to the target density setting. Therefore, if the rate in Equation 4 is abnormal, we perform a backtracking line search with Armijo–Goldstein stopping condition [3] to reset the initial learning rate to $\mathrm{lr}_{init} = \min_t f(z_0 - t\nabla f(z_0))$.

4.2.4 Heuristics Simplification. We remove multiple heuristics during global optimization, which activate quadratic penalty and inject entropy when reaching a high overflow plateau and stop placement based on a divergence check. These heuristics make placement unstable. Thus, we keep the simple first-order density penalty throughout the optimization. In addition, the stopping conditions are simplified to check the variation of the objective, HPWL, and density overflow values.

4.2.5 Macro Orientation Refinement. Standard cells and macros have their orientations fixed during placement inside DREAMPlace. Only at the end of detailed placement are cells flipped based on the orientation of their placement row. While this is inconsequential for standard cells, the orientations of macros can significantly impact wirelength due to the large displacement of their pins when their orientation is changed. We adopt a greedy refinement of macro orientations to account for this potential improvement [46]. Macros are flipped in the four directions, one after another. Each macro is considered once per iteration and flipped if it leads to positive HPWL improvement. The process stops when the incremental improvement is below 0.02%. While usually regarded as suboptimal, this method produces incremental improvements without runtime downsides.

4.3 Parameter Space

We consider the sixteen parameters for DREAMPlace listed in Table 1. From the original set of parameters of DREAMPlace, we propose new parameters to fix the legalization issues and expand the macro placement design space. Note that we use the target cell density d_{target} in DREAMPlace as both a tunable parameter—searched in a range below the area utilization a_{util}—and a target proxy metric. This section will show that the selected parameters are essential for the observed behavior of the DREAMPlace mixed-size placement.

4.3.1 Default DREAMPlace Parameters. We observed that the parameters of the weights schedules of the GD solver are vital for the convergence and quality of the placement optimization. Therefore, we propose to tune the initial learning rate (LR) and decay of the exponential LR schedule. We also adjust the gradient descent optimizer, choosing between two momentum optimizers, Adam [28] and Nesterov [41].

Table 1: Parameter space of DREAMPlace mixed-size placement. A star * denotes a novel parameter, and $\widehat{c_V}$ is the coefficient of variation summarizing the marginal effect of a parameter on the RSMT and congestion.

Parameter	Search Range	$\widehat{c_V}$ (%) RSMT	$\widehat{c_V}$ (%) Cong.	Divg. Rate
*horiz. initial position	[0.2, 0.8] (%)	2.2	0.9	0.0
*vert. initial position	[0.2, 0.8] (%)	2.0	1.1	0.0
*horiz. macro halo	technology dep.	1.8	1.3	0.0
*vert. macro halo	technology dep.	1.7	1.2	0.0
target density d_{target}	$[a_{\text{util}} - 0.2, a_{\text{util}}]$ (%)	-	-	-
density weight	$[1e^{-6}, 1.0]$	3.1	1.7	0.0
smooth HPWL model	{LSE, WA}	0.7	1.1	0.0
smooth HPWL initial γ_0	[0.10, 0.50]	5.1	1.9	0.0
GD initial LR lr_0	$[1e^{-4}, 1e^{-2}]$	1.4	1.0	0.0
GD LR decay	[0.99, 1.0]	6.7	2.3	53.2
GD optimizer	[Adam, Nesterov]	1.2	0.8	54.2
# horiz. global bins	{256, 512, 1024, 2048}	1.3	0.9	0.0
# vert. global bins	{256, 512, 1024, 2048}	3.1	1.3	21.1
λ update lower coeff. L	[0.90, 0.99]	4.2	1.9	0.0
λ update upper coeff. U	[1.01, 1.15]	27.0	7.5	1.8
λ update Δ HPWL$_{REF}$	$[1.5e^5, 5.5e^5]$	2.3	1.2	0.0

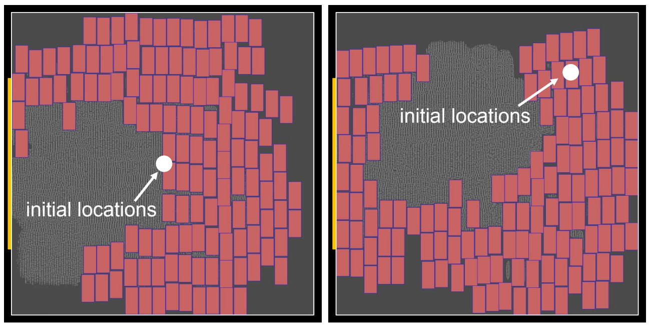

Figure 1: The large effect of the initial locations of macros and cells on the final placement landscape obtained with DREAMPlace.

We consider two variants of smooth HPWL, namely the log-sum-exp (LSE) [11] and weighted-average (WA) [21] wirelength models. Both approximations use a smoothing factor γ dynamically updated every iteration during global placement following [13], starting from γ_0, a tunable parameter. Moreover, the Lagrange multiplier of Equation 1 for global placement follows the ePlace [37] update rule that introduces three parameters, the HPWL reference Δ HPWL$_{REF}$ and the bounds $[L, U]$ of the multiplier. We also consider as parameters the global bin grids applied on the chip's floorplan to calculate the cell density map D_b.

4.3.2 Initial Locations. At the start of the placement, cells and macros concentrate into a single location, minimizing wirelength. Then, as the Lagrange multiplier of the density term gradually increases, cells spread to reach a minimum energy potential configuration at the end of the global placement. DREAMPlace originally

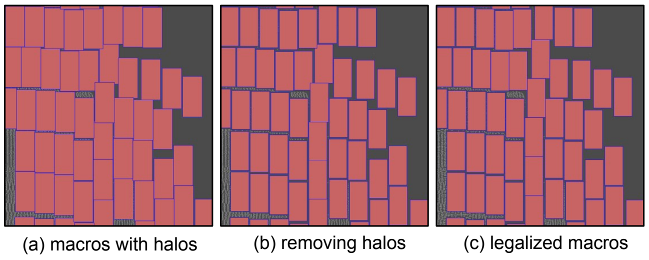

Figure 2: Easing legalization with macro halos. Removing macro halos leaves enough space for macros to be legalized without significantly disrupting the standard cells and other macros.

sets all initial positions of cells and macros at the center of the floorplan. However, as shown in Figure 1, adjusting the initial position can dramatically affect the shape of the final arrangement. Because macros are the first to spread—their gradient is usually most significant at the beginning—and stabilize, they also dictate the shape of the standard cell placement. Therefore, to expand the diversity of the placement solutions, we introduce two parameters to set the initial cells' location as a percentage of the width and height of the floorplan. Noise following the Gaussian distribution is also added to the initial positions to avoid null initial gradients.

4.3.3 Macro Halos. The DREAMPlace macro placements exhibit legalization issues. Movable macros tend to overlap or be closely placed at the end of the global placement, causing problems during legalization because macros are legalized first, ignoring the standard cells [33]. This causes significant disturbances to the macro and standard cell placements, with considerable and unrecoverable wirelength degradation.

We propose to add halos on the macro boundaries. As shown in Figure 2, these leave sufficient room in case the macros plus halos overlap at the end of global placement, as these are removed before macro legalization. This eases macro legalization and allows standard cells to be legalized in the haloed space.

4.3.4 Parameters' Effect. We carry out a sensitivity analysis to justify selecting the parameters of Table 1. We select a design and a good parameter set, and evaluate the marginal effect of each parameter on the PPA proxies by randomly sampling values inside the parameter's definition range, keeping other parameters constant. We report in the third column of Table 1 the coefficient of variation [47] to measure the dispersion of the parameters' effects on the RSMT and congestion defined as $\widehat{c_v}(x) = \sigma(x)/\mu(x)$, where σ is the standard deviation of proxy values x, and μ the sample mean. We fix the density during sensitivity analysis since it is both a parameter and a proxy metric. We see that most parameters affect the proxies. The fourth column reports the average number of times the global placement optimization diverged. These unsuccessful runs are not considered when computing $\widehat{c_v}$. Notably, the parameters related to the mathematical optimization, i.e., gradient descent, Lagrangian multiplier, and smooth HPWL factor, are essential to obtain superior and stable results. This calls for novel and more natural formulations. We leave this task as future work.

4.4 Design Space Exploration

While expanding the design space of placements through enriching the parameter space of DREAMPlace comes with potential PPA improvements, it also requires proper parameter-tuning and PPA evaluation procedures. Indeed, we observed that appropriate placement parameters depend on many design elements. For example, designs with many macros appear better optimized by Nesterov's method, while Adam exhibits more stable but sub-optimal optimization results in these cases. Unfortunately, creating a comprehensive guide for choosing the placement parameters seems unattainable. Moreover, the estimated PPA post-place might not correlate precisely with the post-route PPA from the EDA tool. Therefore, an efficient search framework is essential to obtain high-quality solutions consistently and quickly.

4.4.1 MOTPE. We use the MOTPE to search Pareto-optimal points on the axes of RSMT, density, and congestion. This method can straightforwardly and efficiently generate a Pareto front with points realizing multiple wirelength, density, and congestion trade-offs. It is more amenable than the scalarized single-objective method used, for example, in CircuitTraining as a reward, $\sum_{i=1}^{k} w_i f_i$, where weights $w_i > 0$ must be assigned to each objective f_i and swept to navigate on the Pareto front. The single-objective method might be time-consuming and sub-optimal, given that the correlation between weights and obtainable PPA is difficult to characterize.

4.4.2 Two-level Approach. To manage the latency and complexity of the accurate PPA feedback from the EDA tool, we use a two-level approach, as illustrated in Figure 3; (1) the multi-objective Bayesian optimization with MOTPE explores the vast placement space through tuning the DREAMPlace parameters, evaluating solutions on the proposed high-level metrics. (2) At the end of the sampling procedure, the most promising candidates from the Pareto front are evaluated with more advanced figures of merit (e.g., wirelength, timing, and power) inside the EDA tool after timing optimization and routing. The EDA flow can be stopped early in case of very unsatisfactory intermediate PPA quality.

The search procedure might produce many Pareto-optimal points. From there, we want to select diverse candidates corresponding to different placements to realize different PPA trade-offs. However, the MOTPE refines the internal distributions over time as it gathers more samples, reducing the spreading of the kernels of the good samples (Gaussian for continuous parameters and Aitchison-Aitken for categorical parameters). Therefore, many Pareto points might be obtained near the end of the search. However, these likely correspond to similar placement solutions. Moreover, it is impractical to evaluate many points in the EDA tool due to shared resource constraints, e.g., servers and tool licenses.

Thus, we propose to reduce the number of candidates by performing k-means clustering [36] of the 3D Pareto points. The postulate is that points close in the objective proxy space correspond to similar-looking placements. We select one representative per cluster, the point with the smallest RSMT, yielding $k = 5$ diverse candidates. Figure 4 shows the selection process for one of the evaluated benchmarks, where red points correspond to the macro placements candidates. These points are mapped to accurate PPA metrics with the commercial EDA tool.

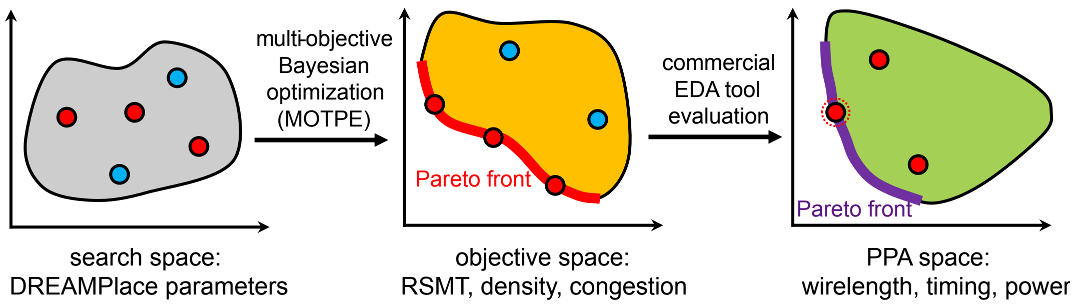
multi-objective Bayesian optimization (MOTPE)
commercial EDA tool evaluation
Pareto front
Pareto front
search space: DREAMPlace parameters
objective space: RSMT, density, congestion
PPA space: wirelength, timing, power

Figure 3: Conceptual view of the two-level exploration of the mixed-size placement landscape.

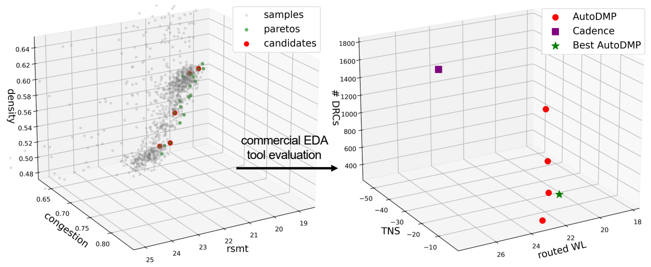

Figure 4: Selection of the macro placements candidates with k-means clustering of the 3D Pareto points on the BlackParrot benchmark. These points are mapped to the PPA space of the commercial EDA tool.

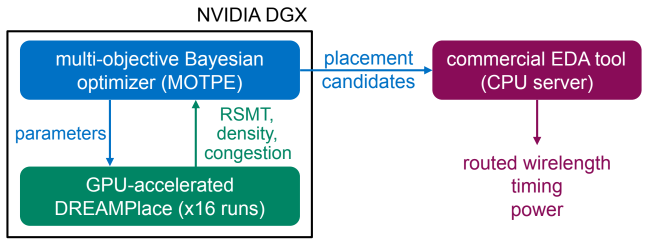
NVIDIA DGX
multi-objective Bayesian optimizer (MOTPE)
placement candidates
commercial EDA tool (CPU server)
parameters
RSMT, density, congestion
GPU-accelerated DREAMPlace (x16 runs)
routed wirelength
timing
power

Figure 5: The AutoDMP computation flow. The multi-objective Bayesian optimization runs on an NVIDIA DGX Station A100. Sixteen parallel processes are spawned to run the GPU-accelerated DREAMPlace during the search. The placement candidates are then fed to the EDA tool, which runs on a CPU server.

4.5 Infrastructure

The AutoDMP computation flow is shown in Figure 5. The Bayesian optimization with MOTPE is conducted on an NVIDIA DGX Station with four A100 GPUs, each with 80GB (HBM2e) memory. We run 16 jobs in parallel on the four GPUs, gathering 1,000 placement samples per design. Running multiple instances of the placing engine on one GPU offers throughput improvement (~40%), given that DREAMPlace and ABCDPlace largely underutilize the GPU memory.

We use the placement candidates in the physical design flow following the simultaneous and sequential mixed-size placement methodologies. We only keep the macro placement from AutoDMP in sequential mode and run on a CPU server the full place-opt with the EDA tool, which places all standard cells and runs pre-CTS optimization. We keep the macro and standard cell placements in the simultaneous flow and run place-opt incremental, which performs pre-CTS optimization only. In both cases, the macro placements are first legalized by the EDA tool to respect complex design rules not handled by ABCDPlace. We then fix the macros and perform P/G planning and routing, which can introduce soft and hard blockages around the macros and inside channels and notches. The subsequent steps include CTS, routing, and post-route optimization.

5 EXPERIMENTS

The AutoDMP framework is developed on top of DREAMPlace and ABCDPlace, using *Python/PyTorch* and *C++/CUDA* for the placement engine modifications and *Python* for the MOTPE optimization [24]. *Tcl/Bash* is used for the place-and-route flow inside the EDA tool, which closely follows the TILOS MacroPlacement flow [49] for open-source reproducibility. The implementations are done with the commercial tool *Cadence Innovus 21.15*, which runs on 16 Intel Xeon Gold CPUs with 80GB of memory allocated. The RTL designs are synthesized with *Cadence Genus 21.14* without physically-aware context.

We use the open-source benchmarks from the TILOS repository [49] to demonstrate the effectiveness of our approach. These modern mixed-size placement benchmarks include *Ariane*, a single-core RISC-V CPU, the *MemPool Group* and *BlackParrot* designs, many-core RISC-V CPUs with large amounts of on-chip SRAMs, and an *NVDLA* partition. Notably, *BlackParrot* and *MemPool Group* are challenging for placers, given their scale and diversity in macro shapes. In addition, we use the open-source process design kit NanGate 45nm [29] to match the current state of the TILOS report [14].

5.1 Benchmark Results

We report the results after post-route optimization to check for timing, design rule check (DRC) violations, power, and routed wirelength. While we report only the best PPA among the five Pareto candidates, the remaining candidates exhibit satisfactory PPA quality. Figure 4 shows that the macro placements of BlackParrot are all of consistent quality.

Table 2 summarizes the PPA results using the AutoDMP placements and the placements obtained using the reference TILOS flow, which uses Innovus in a sequential mixed-size fashion. The results show the potential of our methodology across mixed-size placement methodologies, benchmarks, target frequencies, and densities.

Table 2: Post-route PPA results on the TILOS benchmarks [49]. WL is the routed wirelength; WS is the worst slack; TNS is the total negative slack; flow time is the total runtime of the steps inside Cadence Innovus; search time is the runtime of the MOTPE carried out on an NVIDIA DGX Station A100. For AutoDMP, the top row corresponds to preserving only the macro placement, and the bottom row to keeping both macro and standard cell placement.

Benchmark	Metrics				Innovus Placements							AutoDMP Placements							
	# macros	# cells	freq. (MHz)	dens. (%)	cell area (mm^2)	power (mW)	WL (m)	WS (ns)	TNS (ns)	# DRC	flow time	cell area (mm^2)	power (mW)	WL (m)	WS (ns)	TNS (ns)	# DRC	search time	flow time
NanGate45																			
Ariane	133	88K	250	51	0.213	293.9	3.63	0.082	0.000	0	35m	0.213	294.3	3.57	0.034	0.000	0	1h20	35m
												0.213	294.4	3.56	0.036	0.000	0		40m
Ariane	133	88K	250	68	0.213	294.7	3.80	0.012	0.000	0	30m	0.213	294.5	3.70	0.116	0.000	0	1h20	30m
												0.213	294.6	3.59	0.045	0.000	0		35m
Ariane	133	97K	769	51	0.248	828.0	3.97	-0.159	-147	0	1h20	0.242	820.1	3.85	-0.127	-105	0	1h20	1h20
												0.239	814.5	3.87	-0.103	-81	0		1h40
Ariane	133	97K	769	68	0.244	824.9	4.10	-0.188	-102	0	1h10	0.243	822.5	3.92	-0.120	-97	0	1h20	1h
												0.241	816.8	3.94	-0.075	-34	0		1h30
NVDLA	128	145K	1,111	51	0.396	2,311	8.89	-0.004	-0.043	0	1h10	0.395	2,307	8.56	-0.006	-0.046	0	1h50	1h10
												0.396	2,308	8.56	-0.004	-0.011	0		1h20
BlackParrot	220	652K	357	68	1.827	2,136	22.26	-0.118	-70	1,156	3h20	1.822	2,136	22.28	-0.001	-0.002	558	2h20	3h30
												1.822	2,133	22.08	0.000	0.000	847		3h30
MemPool	320	2.7M	333	68	5.665	4,071	110.6	-0.361	-3,596	6,543	36h	5.760	4,091	111.0	-0.330	-2,913	2,651	3h30	33h
												5.769	4,087	112.0	-0.463	-4,708	8,819		28h
ASAP7 (parameters’ transfer result)																			
NVDLA	128	150K	1,111	51	0.240	535.9	1.97	0.000	0.000	136	1h30	0.237	533.7	2.02	0.000	0.000	268	0	1h10
												0.238	534.9	2.00	0.000	0.000	223		1h10

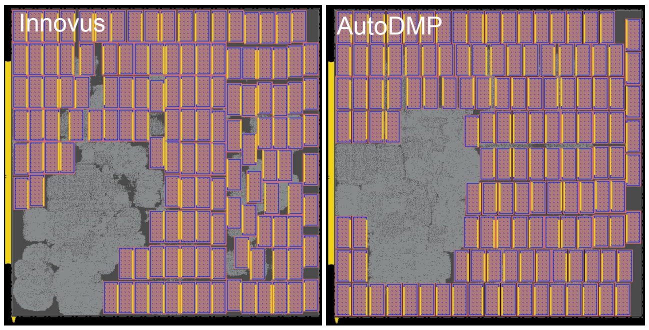

Figure 6: Pre-CTS placement layouts of the Ariane design using NanGate 45nm process (freq. = 769 MHz, density = 68%).

Please note that we report this data only to demonstrate the viability of our approach, not to compare it with the commercial EDA tool, since this benchmark is limited and the industrial design flows are more complicated. Please also refer to the TILOS GitHub [49] for comparison with other open-source tools, which achieve inferior quality compared to the EDA tool.

Figure 6 shows the placement layouts of the Ariane benchmark obtained with Cadence Innovus and AutoDMP. Interestingly, the macro placements are not dissimilar and follow traditional human-like rules with macros on the periphery and away from the IO pins, leaving continuous space for the standard cells. This is reassuring in that our placer makes sensible decisions.

Figure 7 shows the logical to physical maps and cell density maps of the macro and standard cell placements of the MemPool Group benchmark obtained with Cadence Innovus and AutoDMP.

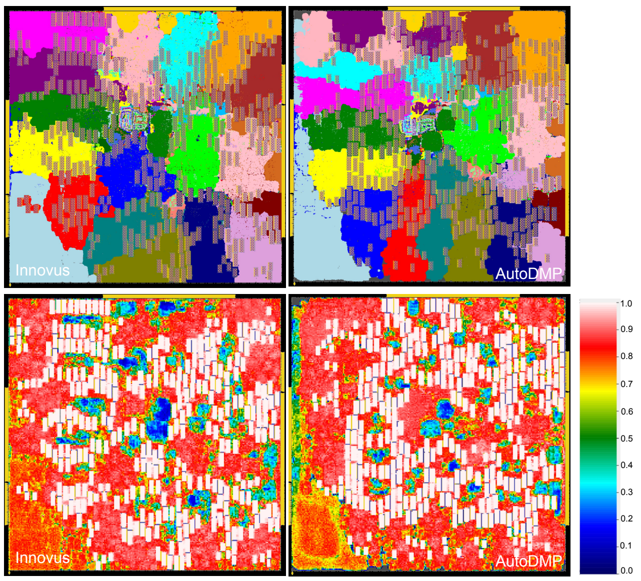

Figure 7: Pre-CTS placements of the logical groups and cell densities of the MemPool Group designs using NanGate 45nm process (freq. = 333 MHz, density = 68%). Congestion (H/V): Innovus (2.66%/1.54%), AutoDMP (3.48%/1.86%).

The macro placements are quite different, along with the physical arrangement of the logical groups. However, the AutoDMP placer physically places logical groups of cells together, like the commercial tool. The cell density maps and congestion reported by the tool show a slightly more routing-difficult design for the downstream steps, which is anticipated, given that our placement algorithm is not congestion-driven.

5.2 Discussions

Here we propose explanations for the results and analyze the transferability of the optimal parameter sets.

5.2.1 Simultaneous/Sequential Mixed-Size Placement. We notice PPA differences between the simultaneous and sequential mixed-size flows for AutoDMP. The listed pros and cons of the two methods from the perspective of legalization and placement blockages can explain the disparities. However, another critical factor is at play: DREAMPlace's macro placements are evaluated with proxies based on the simultaneous standard cell placement from DREAMPlace. However, the standard cells are placed with the EDA tool for the sequential flow. The miscorrelation between the two algorithms' assumptions may be responsible for the observed sub-optimality.

This premise reinforces the need for flexible and broad proxies and a simple wirelength-driven placement, avoiding congestion or timing over-optimization [2], where correlation with the EDA tool algorithms is of higher impact since it depends on the behaviors of the EDA tool global router and timing optimizer.

5.2.2 PPA Improvements. We achieved PPA results of good quality with wirelength, timing, and power benefits. However, we can further improve PPA by making the search more PPA-aware, e.g., with timing and routability considerations, or by increasing the search space with new parameters. We leave as future work the development of more accurate proxy extraction, such as ML-based predictors or using GPU-accelerated global routers to estimate wirelength and congestion without runtime downsides.

It is essential to recognize that the internal engines of the EDA tool consistently achieve high-quality macro placements through the use of historic, long-living parameter sets that are continuously and heuristically refined. However, because we can now afford to explore the design space with GPUs, we can help find more last-mile improvements from the tool and achieve different PPA trade-offs.

Despite not using timing information to drive or evaluate placements during the search, our placements achieve good timing quality, which we propose is driven by one or more of the following:

- Apart from MemPool, the benchmarks are not congestion-critical, easing timing closure.
- The lower-density solutions obtained from the multi-objective search allow the commercial tool to find advantageous, timing-driven placement locations. The timing optimizer also has more room to resize and insert buffers without reaching the maximum density threshold.

5.2.3 Scalibility Benefits. The runtime penalty on the overall flow runtime of the AutoDMP search is reduced for larger designs and tighter timing constraints. This trend favors AutoDMP even further when tackling industrial-size designs, whose study is left as future work. The scalability of the methodology diminishes the need for learning transferable parameter sets since we can directly optimize the macro placement with AutoDMP of every new design within a few hours. The scalability of the search can also unlock new front-end and back-end methodologies, such as early prototyping of design floorplans. Since AutoDMP can produce high-quality floorplans for the EDA tool, it can be used as quick high-level feedback.

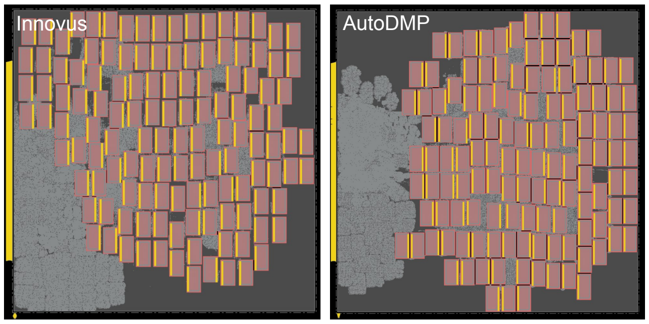

Figure 8: Pre-CTS placement layouts of the NVDLA design using ASAP 7nm process (freq. = 1.11 GHz, density = 51%). The AutoDMP macro placement was obtained by reusing one of the best parameter sets found during MOTPE on NanGate45, illustrating the transferability of parameters across technology nodes.

5.2.4 Parameters' Transferability. Transferring good parameter sets across design changes can improve productivity, as the physical design flow must be rerun when the RTL, floorplan, or technology changes. We investigate transferring optimal parameters found for NVDLA on NanGate 45nm to the more advanced open-source ASAP 7nm process [17]. Only the macro halo sizes are modified by scaling them down to match the new technology. The PPA results at the bottom of Table 2 show that parameters may be transferable across technology nodes. We found that this also holds with different target frequencies. Even if the direct transferability of parameter sets is not needed for successful use of AutoDMP, reusing the best parameters might help reduce or prune the search space and obtain Pareto-optimal points quicker.

6 CONCLUSION

We propose a new open-source methodology called AutoDMP to improve the quality of mixed-size VLSI placement. The macro and standard cell placement space is automatically and efficiently explored by tuning the augmented parameters of the GPU-accelerated DREAMPlace placement engine with multi-objective Bayesian optimization. As a result, we generate placement solutions in a few hours, achieving PPA comparable to commercial tools and superior to open-source academic tools. Our advances can help the turn-around time of early-stage architectural exploration and assess more accurately and efficiently floorplan modification decisions.

REFERENCES

[1] S. N. Adya and I. L. Markov. Combinatorial techniques for mixed-size placement. *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, 10(1):58–90, 2005.
[2] C. Alpert, Z. Li, G.-J. Nam, C. N. Sze, N. Viswanathan, and S. I. Ward. Placement: Hot or not? In *Proceedings of the International Conference on Computer-Aided Design*, pages 283–290, 2012.
[3] L. Armijo. Minimization of functions having Lipschitz continuous first partial derivatives. *Pacific Journal of mathematics*, 16(1):1–3, 1966.
[4] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. Algorithms for hyper-parameter optimization. *Advances in neural information processing systems*, 24, 2011.

[5] U. Brenner, M. Struzyna, and J. Vygen. BonnPlace: Placement of leading-edge chips by advanced combinatorial algorithms. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 27(9):1607–1620, 2008.

[6] T. F. Chan, K. Sze, J. R. Shinnerl, and M. Xie. mPL6: Enhanced multilevel mixed-size placement with congestion control. *Modern Circuit Placement: Best Practices and Results*, pages 247–288, 2007.

[7] Y.-C. Chang, Y.-W. Chang, G.-M. Wu, and S.-W. Wu. B*-Trees: A New Representation for Non-Slicing Floorplans. In *ACM/IEEE Design Automation Conference (DAC)*, pages 458–463, 2000.

[8] H.-C. Chen, Y.-L. Chuang, Y.-W. Chang, and Y.-C. Chang. Constraint graph-based macro placement for modern mixed-size circuit designs. In *2008 IEEE/ACM International Conference on Computer-Aided Design*, pages 218–223. IEEE, 2008.

[9] T.-C. Chen and Y.-W. Chang. Modern floorplanning based on fast simulated annealing. In *Proceedings of the 2005 international symposium on Physical design*, pages 104–112, 2005.

[10] T.-C. Chen, Y.-W. Chang, and S.-C. Lin. IMF: Interconnect-driven multilevel floorplanning for large-scale building-module designs. In *ICCAD-2005. IEEE/ACM International Conference on Computer-Aided Design, 2005.*, pages 159–164. IEEE, 2005.

[11] T.-C. Chen, Z.-W. Jiang, T.-C. Hsu, H.-C. Chen, and Y.-W. Chang. NTUplace3: An analytical placer for large-scale mixed-size designs with preplaced blocks and density constraints. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 27(7):1228–1240, 2008.

[12] T.-C. Chen, P.-H. Yuh, Y.-W. Chang, F.-J. Huang, and D. Liu. MP-trees: A packing-based macro placement algorithm for mixed-size designs. In *Proceedings of the 44th annual Design Automation Conference*, pages 447–452, 2007.

[13] C. Cheng, A. B. Kahng, I. Kang, and L. Wang. RePlAce: Advancing Solution Quality and Routability Validation in Global Placement. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 38(9):1717–1730, 2019.

[14] C.-K. Cheng, A. B. Kahng, S. Kundu, Y. Wang, and Z. Wang. Assessment of Reinforcement Learning for Macro Placement. In *ACM International Symposium on Physical Design (ISPD)*, 2023.

[15] C.-L. E. Cheng. RISA: Accurate and efficient placement routability modeling. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 690–695, 1994.

[16] C. Chu and Y.-C. Wong. FLUTE: Fast Lookup Table Based Rectilinear Steiner Minimal Tree Algorithm for VLSI Design. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 27(1):70–83, 2008.

[17] 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*, 53:105–115, 2016.

[18] J. Cong, M. Romesis, and J. R. Shinnerl. Fast floorplanning by look-ahead enabled recursive bipartitioning. In *Proceedings of the 2005 Asia and South Pacific Design Automation Conference*, pages 1119–1122, 2005.

[19] P.-N. Guo, C.-K. Cheng, and T. Yoshimura. An O-tree representation of non-slicing floorplan and its applications. In *Proceedings 1999 design automation conference*, pages 268–273. IEEE, 1999.

[20] X. Hong and S. Dong. Non-slicing floorplan and placement using corner block list topological representation. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 51, 2004.

[21] M.-K. Hsu, V. Balabanov, and Y.-W. Chang. TSV-aware analytical placement for 3-D IC designs based on a novel weighted-average wirelength model. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2013.

[22] M.-K. Hsu and Y.-W. Chang. Unified analytical global placement for large-scale mixed-size circuit designs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 31(9):1366–1378, 2012.

[23] F. Hutter, H. H. Hoos, and K. Leyton-Brown. Sequential Model-Based Optimization for General Algorithm Configuration. In *Proceedings of the 5th International Conference on Learning and Intelligent Optimization*, LION'05, page 507–523, Berlin, Heidelberg, 2011. Springer-Verlag.

[24] S. Izquierdo, J. Guerrero-Viu, S. Hauns, G. Miotto, S. Schrodi, A. Biedenkapp, T. Elsken, D. Deng, M. Lindauer, and F. Hutter. Bag of baselines for multi-objective joint neural architecture search and hyperparameter optimization. In *8th ICML Workshop on Automated Machine Learning (AutoML)*, 2021.

[25] A. B. Kahng. Advancing placement. In *ACM International Symposium on Physical Design (ISPD)*, pages 15–22, 2021.

[26] A. B. Kahng, S. Reda, and Q. Wang. Architecture and details of a high quality, large-scale analytical placer. In *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 891–898, 2005.

[27] A. B. Kahng, R. Varadarajan, and Z. Wang. RTL-MP: Toward Practical, Human-Quality Chip Planning and Macro Placement. In *ACM International Symposium on Physical Design (ISPD)*, pages 3–11, 2022.

[28] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[29] J. Knudsen. Nangate 45nm open cell library. *CDNLive, EMEA*, 2008.

[30] P. Liao, S. Liu, Z. Chen, W. Lv, Y. Lin, and B. Yu. DREAMPlace 4.0: Timing-driven global placement with momentum-based net weighting. In *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2022.

[31] Y. Lin, S. Dhar, W. Li, H. Ren, B. Khailany, and D. Z. Pan. DREAMPlace: Deep Learning Toolkit-Enabled GPU Acceleration for Modern VLSI Placement. In *ACM/IEEE Design Automation Conference (DAC)*, 2019.

[32] Y. Lin, W. Li, J. Gu, H. Ren, B. Khailany, and D. Z. Pan. ABCDPlace: Accelerated Batch-based Concurrent Detailed Placement on Multi-threaded CPUs and GPUs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, pages 1–1, 2020.

[33] Y. Lin, D. Z. Pan, H. Ren, and B. Khailany. DREAMPlace 2.0: Open-Source GPU-Accelerated Global and Detailed Placement for Large-Scale VLSI Designs. In *2020 China Semiconductor Technology International Conference (CSTIC)*, pages 1–4. IEEE, 2020.

[34] H.-Y. Liu and L. P. Carloni. On learning-based methods for design-space exploration with High-Level Synthesis. In *2013 50th ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 1–7, 2013.

[35] Y.-C. Liu, T.-C. Chen, Y.-W. Chang, and S.-Y. Kuo. MDP-trees: multi-domain macro placement for ultra large-scale mixed-size designs. In *Proceedings of the 24th Asia and South Pacific Design Automation Conference*, pages 557–562, 2019.

[36] S. Lloyd. Least squares quantization in PCM. *IEEE transactions on information theory*, 28(2):129–137, 1982.

[37] J. Lu, H. Zhuang, P. Chen, H. Chang, C.-C. Chang, Y.-C. Wong, L. Sha, D. Huang, Y. Luo, C.-C. Teng, et al. ePlace-MS: Electrostatics-Based Placement for Mixed-Size Circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 34(5):685–698, 2015.

[38] I. L. Markov, J. Hu, and M.-C. Kim. Progress and challenges in VLSI placement research. *Proceedings of the IEEE*, 103(11):1985–2003, 2015.

[39] A. Mirhoseini, A. Goldie, M. Yazgan, J. W. Jiang, E. Songhori, S. Wang, Y.-J. Lee, E. Johnson, O. Pathak, A. Nazi, et al. A graph placement methodology for fast chip design. *Nature*, 594(7862):207–212, 2021.

[40] H. Murata, K. Fujiyoshi, S. Nakatake, and Y. Kajitani. VLSI module placement based on rectangle-packing by the sequence-pair. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 12:1518–1524, 1996.

[41] Y. Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2003.

[42] Y. Ozaki, Y. Tanigaki, S. Watanabe, and M. Onishi. Multiobjective Tree-Structured Parzen Estimator for Computationally Expensive Optimization Problems. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, GECCO '20, page 533–541, New York, NY, USA, 2020. Association for Computing Machinery.

[43] E. Parzen. On estimation of a probability density function and mode. *The annals of mathematical statistics*, 33(3):1065–1076, 1962.

[44] A. Ranjan, K. Bazargan, S. Ogrenci, and M. Sarrafzadeh. Fast floorplanning for effective prediction and construction. *IEEE transactions on very large scale integration (VLSI) systems*, 9(2):341–351, 2001.

[45] P. G. Sassone and S. K. Lim. A novel geometric algorithm for fast wire-optimized floorplanning. In *ICCAD-2003. International Conference on Computer Aided Design*, pages 74–80. IEEE, 2003.

[46] C.-W. Sham, F. Y. Young, and C. Chu. Optimal cell flipping in placement and floorplanning. In *ACM/IEEE Design Automation Conference (DAC)*, pages 1109–1114, 2006.

[47] R. Sokal and F. Rohlf. Biometry 3rd ed WH Freeman and Company. *New York*, 1995.

[48] P. Spindler and F. M. Johannes. Fast and accurate routing demand estimation for efficient routability-driven placement. In *IEEE/ACM Proceedings Design, Automation and Test in Eurpoe (DATE)*, pages 1–6. IEEE, 2007.

[49] TILOS-AI-Institute. MacroPlacement: an open-source, transparent implementation and assessment of Google Brain's Circuit Training. https://github.com/TILOS-AI-Institute/MacroPlacement.

[50] 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 *2020 25th Asia and South Pacific Design Automation Conference (ASP-DAC)*, pages 19–25, 2020.

[51] J. Z. Yan and C. Chu. DeFer: Deferred Decision Making Enabled Fixed-Outline Floorplanner. In *ACM/IEEE Design Automation Conference (DAC)*, pages 161–166, 2008.

[52] J. Z. Yan, N. Viswanathan, and C. Chu. Handling complexities in modern large-scale mixed-size placement. In *Proceedings of the 46th Annual Design Automation Conference*, pages 436–441, 2009.