COARSE-TO-FINE MACRO PLACEMENT VIA EVOLUTIONARY SEARCH AND CRITICAL MACRO TUNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Macro placement is a critical stage in chip physical design that substantially affects downstream implementation quality. Recent search-based methods improve existing layouts through partial reconstruction, but quality-biased or spatially restricted macro selection can limit the diversity of reconstruction proposals, potentially hindering escape from local optima. Moreover, coarse-grid representations restrict placement precision. To address these challenges, we propose C2FPlace, a **C**oarse-to-**F**ine macro **Place**ment framework that integrates population-based evolutionary search with fine-grained refinement. During coarse-grained optimization, tournament selection chooses promising parents from randomly sampled groups of layouts, and stochastic partial rip-up and re-place generates offspring by sampling macro subsets across the entire layout. A two-phase schedule samples reconstruction ratios from a higher range early in the search and a lower range later, supporting broad exploration followed by more conservative refinement. During fine-grained optimization, critical macro tuning enables positional adjustments beyond the coarse grid to obtain additional half-perimeter wirelength (HPWL) reduction. Experiments on the ISPD2005 benchmark show that C2FPlace reduces HPWL by 17.82% over EGPlace and 17.86% over RollPlace on average. On the ICCAD2025 benchmark, C2FPlace achieves the best average ranking among the compared methods under the evaluated power, performance, and area (PPA) metrics.

1 INTRODUCTION

Macro placement is a critical stage in chip physical design that significantly affects downstream design closure. In modern VLSI systems, high-quality macro placement facilitates subsequent standard cell placement, clock tree synthesis, and detailed routing. Conversely, poor macro placement can degrade implementation quality even after extensive downstream optimization Kang et al. (1994); MacMillen et al. (2000). The macro placement problem is NP-hard, with tightly coupled decisions arising from macro dimensions, connectivity, and non-overlap constraints. Analytical placers address placement through continuous optimization and gradient-based numerical techniques Cheng et al. (2019); Lin et al. (2019); Liao et al. (2022). Although these methods offer strong scalability, their solutions can depend on initialization and subsequent legalization, which may alter the optimized placement objectives.

Recent reinforcement learning (RL) and black-box optimization (BBO) methods have shown promising results for macro placement Mirhoseini et al. (2021); Lai et al. (2022; 2023); Shi et al. (2023); Geng et al. (2024); Xue et al. (2024); Li et al. (2025); Zhou et al. (2025); Geng et al. (2025). Beyond constructing layouts sequentially, recent search-based approaches improve existing solutions through partial reconstruction while retaining the positions of unselected macros, as shown in Table 1. EGPlace Li et al. (2025) maintains a population of layouts and uses fitness-based probabilities to select parents for evolution. RollPlace Zhou et al. (2025) organizes candidate layouts in a Monte Carlo search tree and uses the Upper Confidence Bound applied to Trees (UCT) criterion for candidate selection. These methods demonstrate the value of maintaining multiple candidates and reusing existing placement structures.

Despite these advances, generating sufficiently diverse reconstruction proposals remains an important challenge. In EGPlace, parent selection follows a softmax distribution over fitness values, while

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Table 1: Comparison of macro placement methods and their search mechanisms.

Method	Parent selection	Macro selection	Update scale	Granularity
MaskPlace (Lai et al., 2022)	N/A	N/A	N/A	Coarse grid
ChiPFormer (Lai et al., 2023)	N/A	N/A	N/A	Coarse grid
WireMask-BBO (Shi et al., 2023)	1+1 EA	N/A	All	Coarse grid
EfficientPlace (Geng et al., 2024)	UCT	N/A	N/A	Coarse grid
MaskRegulate (Xue et al., 2024)	Fixed	N/A	All	Coarse grid
EGPlace (Li et al., 2025)	Fitness-based	Score-guided	Fixed ratio	Coarse grid
RollPlace (Zhou et al., 2025)	UCT	Rectangle	Region-based	Coarse grid
C2FPlace (ours)	Tournament	Random subset	Two-phase ratio	**Coarse-to-fine**

macro selection is guided by scores derived from wirelength, congestion, and overlap. Although both selections are stochastic, their preference for current quality estimates may concentrate search effort on similar candidates and macro subsets. RollPlace samples two coordinates to define a rectangular region and repositions all macros within that region. Such region-based selection couples macro inclusion to spatial proximity, restricting the subsets that can be reconstructed together in a single update. These selection biases may limit exploration of alternative macro combinations and make it harder to escape local optima. This motivates combining quality-aware parent selection with macro-subset sampling that is not tied to current macro scores or a single rectangular region.

The extent of reconstruction introduces a further exploration–exploitation trade-off. Repositioning a large subset of macros permits broader coordinated changes, but also exposes more established placement relationships to modification. Repositioning a small subset retains more of the current structure, but offers fewer opportunities for substantial rearrangement. A fixed reconstruction ratio, as used in the reported EGPlace experiments, does not explicitly reflect these differing needs across search stages. Varying the reconstruction scale over time offers a way to encourage broad exploration early in the search and preserve more of the evolved layout during later refinement.

A second challenge arises from placement granularity. Many recent methods discretize the placement canvas into a coarse grid, such as 224×224, to make the search space tractable. While this representation reduces optimization complexity, it also restricts available macro coordinates. Even after effective coarse-grid search, finer positional adjustments may further improve wirelength, particularly for macros whose pins determine net bounding boxes. Combining coarse-grained exploration with targeted fine-grained refinement can exploit these remaining opportunities without repeating global search at a finer resolution.

To address these challenges, we propose C2FPlace, a coarse-to-fine macro placement framework centered on population-based evolutionary search. C2FPlace first generates multiple initial layouts through repeated WireMask-guided Lai et al. (2022); Shi et al. (2023) greedy placement. During coarse-grained optimization, tournament selection randomly samples a group of individuals and chooses the best among them as the parent. This provides quality-aware selection based on relative ranking within each sampled group, without requiring a softmax transformation of fitness values. Offspring are generated through stochastic partial rip-up-and-replace: a reconstruction ratio is sampled, and the corresponding subset of macros is randomly selected across the entire layout for repositioning, while unselected macros retain their positions. A two-phase schedule draws ratios from a higher range during the initial search phase and a lower range during the subsequent phase. Together, random parent-group sampling, random macro-subset selection, and varying reconstruction ratios provide multiple sources of search variation, while tournament selection and elitist replacement favor promising layouts. The schedule shifts the search from broad reconstruction toward more conservative refinement.

Following coarse-grained search, C2FPlace performs fine-grained critical macro tuning to further reduce half-perimeter wirelength (HPWL). This stage selectively adjusts critical macros beyond the coarse-grid coordinates while preserving the global arrangement established by evolutionary search. Coarse-grained optimization thus provides the main search capability, and fine-grained tuning complements it with additional positional refinement. Our main contributions are summarized as follows:

- We propose a population-based evolutionary search strategy that combines tournament selection with stochastic partial rip-up and re-place. Random macro-subset sampling across the layout promotes diverse reconstruction proposals, while a two-phase reconstruction-ratio schedule supports broad exploration followed by more conservative refinement.
- We propose a coarse-to-fine framework that integrates population-based search on a coarse placement grid with fine-grained critical macro tuning, enabling additional HPWL reduction through positional adjustments beyond the grid.
- Experiments on the ISPD2005 benchmark C2FPlace reduces HPWL by 17.82% over EGPlace and 17.86% over RollPlace on average. On the ICCAD2025 benchmark, C2FPlace achieves the best average ranking among the compared methods under the evaluated PPA metrics.

2 Related Work

2.1 Learning-based Methods

Learning-based methods, particularly reinforcement learning (RL), learn policies for constructing or refining macro placements. GraphPlace Mirhoseini et al. (2021) formulates macro placement as a sequential Markov decision process, while DeepPR Cheng & Yan (2021) and PRNet Cheng et al. (2022b) incorporate feedback from standard-cell placement and routing. MaskPlace Lai et al. (2022) uses pixel-wise masks to represent placement constraints and wirelength information, and GoodFloorPlan Xu et al. (2021) combines graph convolutional networks with sequence-pair representations. To improve learning efficiency, ChiPFormer Lai et al. (2023) adopts offline RL with placement-data pretraining, whereas EfficientPlace Geng et al. (2024) integrates Monte Carlo Tree Search with RL. More recently, MaskRegulate Xue et al. (2024) focuses on iterative refinement and incorporates regularity priors into its state representation and reward design to improve placement quality and downstream PPA.

2.2 Optimization-based Methods

Optimization-based approaches include partition-based, analytical, and meta-heuristic methods. Partition-based methods Breuer (1977); Agnihotri et al. (2005); Roy et al. (2006) recursively divide the placement region and netlist, while Hier-RTLMP Kahng et al. (2024) applies hierarchical decomposition to complex designs. Analytical methods use quadratic Spindler & Johannes (2007); Spindler et al. (2008); Lin et al. (2015); Li et al. (2016); Cheng et al. (2022a) or nonlinear formulations Chen et al. (2008); Kahng & Qinke Wang (2005); Hsu et al. (2014). Electrostatic approaches include ePlace-MS Lu et al. (2015b), ePlace Lu et al. (2015a), and RePlAce Cheng et al. (2019); other scalable frameworks include PeF Li et al. (2022), DREAMPlace Lin et al. (2019), and Xplace Liu et al. (2024). These methods generally require subsequent legalization to obtain feasible layouts. Meta-heuristic approaches include evolutionary search Cohoon & Paris (1987) and simulated annealing Shunmugathammal et al. (2020); Kahng et al. (2022). AutoDMP Agnesina et al. (2023) uses Bayesian optimization to tune DREAMPlace Lin et al. (2019), while WireMask-BBO Shi et al. (2023) combines black-box search with WireMask-guided greedy placement. EGPlace Li et al. (2025) evolves a population through score-guided macro repositioning, and RollPlace Zhou et al. (2025) combines Monte Carlo tree search with order exchanges and repeated rollouts. Building on partial reconstruction, C2FPlace combines tournament selection with scheduled random-subset repositioning, followed by fine-grained critical macro tuning.

3 Preliminary

3.1 Problem Formulation

The input of a macro placement problem is a netlist, which contains wire nets and module information, such as macro sizes and pin offsets. A wire net connects a group of pins from different modules. A placement solution is represented as $P = \{(x_i, y_i)\}_{i=1}^{N}$, where (x_i, y_i) denotes the position of the i-th macro and N is the total number of macros. Since exact routing evaluation is computationally expensive, heuristic metrics such as half-perimeter wirelength (HPWL) are commonly used to

guide macro placement. HPWL provides an efficient wirelength estimation by converting routing optimization into a geometric calculation Mirhoseini et al. (2021); Garey & Johnson (1977). This paper focuses on minimizing HPWL under non-overlap constraints, formulated as:

$$\min_{P} \text{ HPWL}(P), \quad \text{s.t. Overlap}(x, y, w, h) = 0 \tag{1}$$

where x and y represent macro positions, and w and h represent macro width and height, respectively. HPWL is defined as the sum of the half-perimeters of the bounding boxes for all wire nets in the netlist, which is formulated as:

$$\text{HPWL} = \sum_{e_i \in E} \left[\left(\max_{p \in e_i} x_p - \min_{p \in e_i} x_p \right) + \left(\max_{p \in e_i} y_p - \min_{p \in e_i} y_p \right) \right], \tag{2}$$

where E is the set of all wire nets, and p denotes a pin in net e_i with coordinates (x_p, y_p).

3.2 WireMask-Based Placement Operator

C2FPlace employs a WireMask-based placement operator Lai et al. (2022); Shi et al. (2023) as the basic placement primitive. Given a macro to be placed, the operator first identifies legal locations satisfying boundary and non-overlap constraints. For each legal location, WireMask estimates the induced wirelength by computing the HPWL of nets connected to the macro, and the location with the minimum estimated wirelength is selected. Specifically, the placement region is discretized into a $G \times G$ grid. For a macro m, the legality mask $\mathbf{L}_m$ indicates whether each grid location satisfies placement constraints, while the WireMask $\mathbf{W}_m$ records the estimated HPWL cost of placing m at each legal location. The selected position is obtained by:

$$(i_m, j_m) = \arg\min_{(i,j)} \{ \mathbf{W}_m(i, j) \mid \mathbf{L}_m(i, j) = 1 \}. \tag{3}$$

Here, (i_m, j_m) denotes a coarse-grained grid index, which must be mapped to physical coordinates. Fine-grained tuning evaluates the same wirelength cost on its fine-grained candidate set. This placement operator is consistently used in initialization, evolutionary search, and critical macro tuning, providing a unified optimization primitive throughout C2FPlace.

4 Method

Figure 1 illustrates the overall pipeline of C2FPlace. Starting from a population of diverse WireMask-guided initial placements, C2FPlace performs evolutionary optimization in two stages. The coarse-grained stage uses tournament selection and a staged exploration-to-exploitation reconstruction strategy to explore diverse solutions while preserving high-quality structures. The resulting layouts are further refined by fine-grained critical macro tuning to correct discretization errors from coarse-grid placement. WireMask-guided greedy placement serves as the common placement operator throughout the search.

4.1 Initial Population Generation

The optimization starts from a diverse population generated by the WireMask-based placement operator. Instead of relying on a single individual initialization, C2FPlace independently constructs 20 initial placement solutions through greedy WireMask-guided macro placement.

During initialization, macros are placed according to a deterministic ordering heuristic that prioritizes larger and more strongly connected macros. This strategy alleviates early resource contention and produces stable placement configurations. Crucially, when multiple locations offer equivalent optimality, the algorithm selects randomly among them. By repeatedly applying the greedy placement procedure, C2FPlace obtains a diverse initial population, providing multiple promising starting points for subsequent evolutionary optimization.

4.2 Coarse-Grained Evolutionary Search

C2FPlace optimizes the initial population through a steady-state evolutionary search. At each iteration, five individuals are randomly sampled for tournament selection, and the one with the lowest

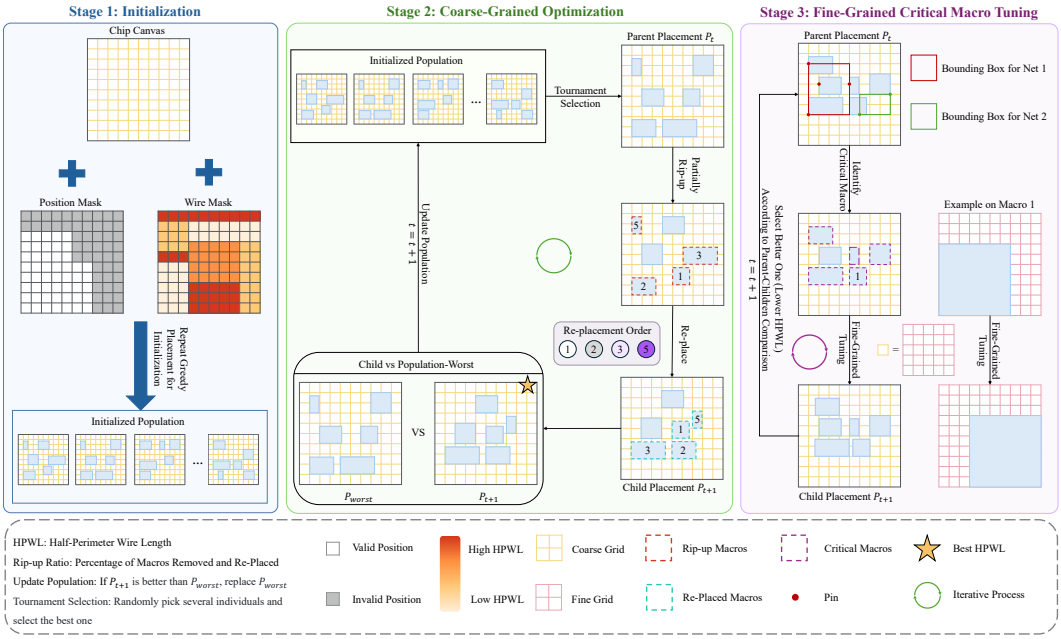


Figure 1: Overview of C2FPlace. The framework initializes a population of diverse greedy placements, evolves them through stochastic evolutionary search with partial rip-up and re-place, and performs critical macro tuning for further HPWL reduction.

HPWL is chosen as the parent. The parent undergoes partial rip-up and re-place to generate an offspring, which is evaluated and incorporated into the population through elitist update, as described in Section 4.2.2.

4.2.1 Mutation Strategy

Inspired by evolutionary optimization, C2FPlace generates new candidate placements through stochastic mutation rather than complete reconstruction. Our mutation strategy consists of three complementary components: partial rip-up and re-place, staged exploration-to-exploitation scheduling, and heuristic-guided reinsertion.

Partial Rip-up and Re-place Instead of reconstructing the entire placement, C2FPlace performs optimization by modifying only a subset of macros in each iteration. Specifically, a subset of macros is randomly selected and removed from the current placement, while all remaining macros remain fixed. Compared with reconstruction, this strategy reduces optimization cost while preserving high-quality placement structures accumulated during previous iterations, thereby providing a better balance between search diversity and solution stability.

Exploration-to-Exploitation To balance these competing objectives,C2FPlace uses a two-phase schedule for exploration-to-exploitation schedule. At the first phase, the rip-up ratio is sampled from a high-valued interval, providing stronger perturbation for global exploration. At the second phase, The sampling interval shifts toward lower values, enabling a transition toward exploitation.

Heuristic-Guided Reinsertion After determining the rip-up ratio, the selected macros are removed from the current placement and reinserted one by one using the WireMask-based placement operator described in Section 3.2. C2FPlace follows a topology-aware heuristic ordering that prioritizes macros with higher connectivity and larger area.

4.2.2 POPULATION UPDATE

After mutation, the generated offspring is evaluated based on its HPWL. To preserve high-quality solutions while continuously improving the population, C2FPlace adopts an elitist replacement strategy. Specifically, the offspring is compared with the worst individual in the current population and replaces it only when achieving better placement quality. Algorithm 1 summarizes the complete coarse-grained evolutionary search procedure.

4.3 CRITICAL MACRO TUNING

Although coarse-grained evolutionary search effectively explores the macro placement space, the obtained solution may still contain suboptimal positions for macros with significant impact on wirelength. This is mainly because coarse-grid representation limits placement flexibility and introduces discretization errors, especially for macros determining net bounding-box boundaries. Therefore, C2FPlace performs critical macro tuning to selectively refine influential macros while preserving the optimized global placement structure.

Different from the global exploration performed in evolutionary search, critical macro tuning focuses on local improvement around high-quality solutions. To balance refinement flexibility and placement precision, C2FPlace adopts a dual-granularity refinement strategy, including structure-preserving coarse-grid displacement and fine-grid offset refinement.

4.3.1 CRITICAL MACRO IDENTIFICATION

For each net, C2FPlace identifies macros that define the four boundaries of its HPWL bounding box as shown on the right side of Figure 1. Since displacing these macros can affect the dimensions of the bounding box, they exert the most significant influence on total wirelength.

For net n with pin set P_n, the bounding box boundaries are defined as:

$$x_{\min}(n) = \min_{p \in P_n} p.x, \quad x_{\max}(n) = \max_{p \in P_n} p.x, \quad y_{\min}(n) = \min_{p \in P_n} p.y, \quad y_{\max}(n) = \max_{p \in P_n} p.y. \tag{4}$$

A macro is classified as an extreme provider if it hosts at least one pin lying on any of these four boundaries. To exclude nets whose bounding box is governed by a single macro, we only consider nets with multiple extreme providers. The critical macro set is then formulated as:

$$\mathcal{M}_{\text{crit}} = \bigcup_{\substack{n \\ |E_n| \geq 2}} E_n, \tag{5}$$

where E_n represents the set of extreme providers for net n.

4.3.2 DUAL-GRANULARITY REFINEMENT

After identifying the critical macro set, C2FPlace considers two complementary placement strategies for refinement. The first allows larger-scale movement to escape unfavorable coarse-grid positions, while the second focuses on recovering fine-grained precision within the current coarse cell.

Structure-Preserving Coarse-Grid Displacement Although coarse-grid placement provides efficient global exploration, a macro may still be trapped at a suboptimal coarse-grid location. Therefore, C2FPlace first performs coarse-grid displacement to explore neighboring coarse locations. Different from unrestricted movement, the displacement is constrained to avoid disturbing surrounding macros. Specifically, the movement from the original location to a candidate location is feasible only when the swept region of the macro does not overlap with other placed macros. Among all feasible candidates, the position with the minimum incremental HPWL is selected. This constraint enables larger-scale local exploration while maintaining the optimized placement structure.

Fine-Grid Offset Refinement After coarse-grid displacement, C2FPlace further improves placement precision through fine-grid offset refinement. Instead of changing the coarse-grid location, this strategy searches fine-grained offsets within the current coarse grid cell to compensate for discretization errors. For each critical macro, all feasible offsets within the corresponding coarse cell are evaluated, and the position with the minimum incremental HPWL is selected. This refinement improves placement accuracy while avoiding unnecessary perturbations to the global solution.

Table 2: HPWL ($\times 10^5$) comparison of different macro placement methods on the ISPD2005 benchmark. Lower HPWL values indicate better placement quality. The results of baseline methods are taken from EGPlace and RollPlace. Performance metrics are reported as mean values with standard deviations (mean $\pm$ std), except for the deterministic method NTUPlace3. The best-performing results are highlighted in **bold**, while the second-best results are marked in brown. "*w/*" and "*w/o*" denote the settings with and without Stage 3 critical macro tuning.

Method	adaptec1	adaptec2	adaptec3	adaptec4	bigblue1	bigblue2	bigblue3	bigblue4
SP-SA	18.84 ± 4.62	117.36 ± 8.73	115.48 ± 7.56	120.03 ± 4.25	5.12 ± 1.43	N/A	164.70 ± 19.55	N/A
NTUPlace3	26.62	321.17	328.44	462.93	22.85	N/A	455.53	N/A
RePlAce	16.19 ± 2.10	153.26 ± 29.01	111.21 ± 11.69	37.64 ± 1.05	2.45 ± 0.06	N/A	119.84 ± 34.43	N/A
DREAMPlace	15.81 ± 1.64	140.79 ± 26.73	121.94 ± 25.05	$\mathbf{37.41 \pm 0.87}$	2.44 ± 0.06	N/A	107.19 ± 29.91	112.95 ± 4.03
GraphPlace	30.10 ± 2.98	351.71 ± 38.20	358.18 ± 13.95	151.42 ± 9.72	10.58 ± 1.29	N/A	357.48 ± 47.83	N/A
DeepPR	19.91 ± 2.13	203.51 ± 6.27	347.16 ± 4.32	311.86 ± 56.74	23.33 ± 3.65	N/A	430.48 ± 12.18	433.90 ± 5.26
MaskPlace	7.62 ± 0.67	75.16 ± 4.97	100.24 ± 13.54	87.99 ± 3.25	3.04 ± 0.06	N/A	90.04 ± 4.83	130.15 ± 10.94
ChiPFormer	6.62 ± 0.05	67.10 ± 5.46	76.70 ± 1.15	68.80 ± 1.59	2.95 ± 0.04	N/A	72.92 ± 2.56	94.90 ± 2.50
EfficientPlace	5.94 ± 0.04	46.79 ± 1.60	56.35 ± 0.99	58.47 ± 1.61	2.14 ± 0.01	10.48 ± 0.25	58.38 ± 0.54	88.01 ± 2.71
WireMask-EA	6.15 ± 0.05	64.38 ± 4.43	58.18 ± 1.04	59.52 ± 1.71	2.15 ± 0.01	19.47 ± 0.17	59.85 ± 3.39	87.81 ± 3.30
EGPlace	5.72 ± 0.01	37.69 ± 1.08	60.13 ± 1.83	56.08 ± 0.43	2.20 ± 0.01	10.40 ± 0.34	52.41 ± 8.16	67.14 ± 2.42
RollPlace	5.93 ± 0.07	34.83 ± 1.81	54.88 ± 3.08	64.66 ± 3.49	2.05 ± 0.03	10.89 ± 0.22	53.56 ± 7.31	69.67 ± 2.70
C2FPlace *w/o*	$\mathbf{5.60 \pm 0.17}$	29.13 ± 1.91	51.19 ± 2.05	43.67 ± 1.21	$\mathbf{2.04 \pm 0.01}$	7.50 ± 0.28	37.98 ± 4.32	56.29 ± 1.29
C2FPlace *w/*	$\mathbf{5.60 \pm 0.17}$	$\mathbf{29.06 \pm 1.90}$	$\mathbf{51.06 \pm 2.05}$	43.39 ± 1.22	$\mathbf{2.04 \pm 0.01}$	$\mathbf{7.43 \pm 0.27}$	$\mathbf{37.90 \pm 4.34}$	$\mathbf{56.18 \pm 1.29}$

4.3.3 Critical Macro Tuning Procedure

During each tuning iteration, C2FPlace first applies coarse-grid displacement to the critical macros. It then re-computes the critical macro set and applies fine-grid offset refinement. The candidate placement is accepted only if its total HPWL is lower than that of the current placement.The WireMask-based placement operator described in Section 3.2 evaluates the incremental HPWL of each feasible candidate, and the best position is selected for reinsertion.

To ensure monotonic improvement, C2FPlace adopts an elitist update strategy. The refined placement replaces the current solution only when it achieves lower HPWL. Therefore, the critical macro tuning stage progressively improves local placement quality while preserving the globally optimized placement structure. The pseudocode of Stage 3 is shown in Appendix A.1.

5 Experiments

5.1 Benchmarks, Baselines and Settings

For evaluation, we use the ISPD2005 benchmark Nam et al. (2005) to assess macro placement performance and the ICCAD2025 benchmark Lu et al. (2025) to evaluate large-scale placement capability and post-placement PPA. We compare C2FPlace with the packing-based SP-SA Murata et al. (1996), representative analytical placers (NTUPlace3 Chen et al. (2008), RePlAce Cheng et al. (2019), and DREAMPlace Lin et al. (2019)), RL-based approaches (GraphPlace Mirhoseini et al. (2021), DeepPR Cheng & Yan (2021), MaskPlace Lai et al. (2022), ChiPFormer Lai et al. (2023), and EfficientPlace Geng et al. (2024)), and optimization-based methods (WireMask-EA Shi et al. (2023), EGPlace Li et al. (2025), and RollPlace Zhou et al. (2025)). Detailed settings of C2FPlace are provided in Appendix A.3.

5.2 ISPD2005 Results

5.2.1 Placement Quality Comparison

We evaluate macro placement quality on the ISPD2005 benchmark containing eight circuits. Following previous works, all macros are placed for `adaptec1-4`, `bigblue1`, and `bigblue3`, while the first 1024 macros are considered for the large-scale `bigblue2` and `bigblue4`. The HPWL results are reported in Table 2.

As shown in Table 2, C2FPlace with Stage 3 achieves the lowest HPWL on seven of eight circuits, whereas DREAMPlace performs best on `adaptec4`. Across the eight benchmarks `adaptec1-4`, `bigblue1-4`, C2FPlace reduces HPWL by 2.10%, 22.90%, 15.08%, 22.63%, 7.27%, 28.56%, 27.69%, and 16.32% compared with EGPlace, and achieves HPWL reductions of 5.56%, 16.57%, 6.96%, 32.90%, 0.49%, 31.77%, 29.24%, and 19.36% relative to RollPlace, respectively. Averaged

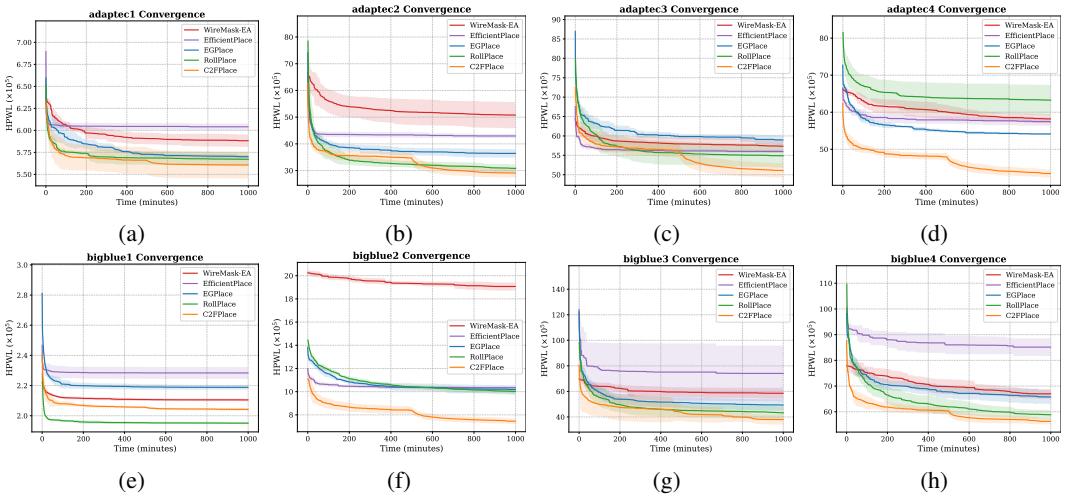


Figure 2: Convergence of best-so-far HPWL over runtime for methods rerun on our AMD EPYC 9654 CPU and RTX 6000 ada under a unified 1000-minute budget. Solid lines denote mean values, and shaded regions indicate standard deviation intervals over five independent runs.

over all benchmarks, C2FPlace reduces HPWL by 17.82% against EGPlace and 17.86% against RollPlace, demonstrating the effectiveness of the proposed coarse-to-fine population-based evolutionary macro placement framework. The visualization for `adaptec3` is shown in Appendix B.4.

Furthermore, we perform ablation experiments to evaluate the contribution of Stage 3. On average, Stage 3 brings a 0.31% HPWL reduction compared to without Stage 3. These results confirm that Stage 3, which performs fine-grained tuning of critical macros, can further reduce HPWL beyond the previous stages. However, the improvement is negligible in certain scenarios (e.g., `adaptec1` and `adaptec4`), where the gain is almost zero. This demonstrates the effectiveness of the proposed coarse-to-fine population-based evolutionary macro placement framework.

5.2.2 Convergence Analysis

As shown in Figure 2, all methods are evaluated under the same 1000-minute runtime budget, with all other hyperparameters following their original settings, and C2FPlace allocates this budget as 500 min for Stage 2 Phase 1, 400 min for Stage 2 Phase 2, and 100 min for Stage 3. C2FPlace rapidly reduces HPWL in the early stage and continues to improve throughout optimization, demonstrating effective global exploration while preserving flexibility for later refinement.

The convergence curves also demonstrate the benefit of staged partial rip-up. By exploring diverse local reconstruction trajectories, C2FPlace avoids premature stagnation and maintains steady HPWL improvements, particularly on large and highly connected circuits. The subsequent low-ratio updates and Stage 3 critical macro tuning further reduce residual wirelength overhead, confirming the effectiveness of the coarse-to-fine optimization strategy.

5.3 ICCAD2025 Results

To assess the scalability and downstream physical design quality of C2FPlace, we further evaluate it on the ICCAD2025 benchmark. We exclude `ariane` (136 macros) and apply C2FPlace and search-based baselines to standard-cell placement on the remaining five circuits, with the largest design containing over 12K cells. Following the official flow, we use OpenROAD v2.0 with the provided ASAP7 technology library to obtain placement and PPA metrics. Note that Placement HPWL is from the placer, while Detailed Placement HPWL is measured after legalization and detailed placement where the latter is typically larger due to row/site alignment and design-rule constraints.

As shown in Table 3, C2FPlace with Stage 3 achieves the lowest placement HPWL on all five circuits, rwith an average per-circuit reduction of 37.92% relative to the best baseline for each circuit. Compared with C2FPlace without Stage 3, it reduces placement HPWL by 2.11% on average,

Table 3: Intermediate and PPA metrics on the ICCAD2025 benchmarks with a 1000-minute runtime budget. "*w/*" and "*w/o*" denote the settings with and without Stage 3 critical macro tuning.

Benchmark	Method	Intermediate Metrics		PPA Metrics			
		Placement	Detailed Placement	Timing Performance			Power ↓
		HPWL ↓	HPWL ↓	WNS ↑	TNS ↑	NVP ↓	
ac97_top	WireMask-EA	106125	115299	-350.88	-262524.84	**1889**	**9.18e-02**
	EGPlace	84244	85090	-305.20	-257896.27	1897	1.16e-01
	RollPlace	154668	158904	-484.65	-272175.41	1892	1.06e-01
	C2FPlace *w/o* stage 3	34778	35014	**-279.11**	**-216087.58**	1893	1.34e-01
	C2FPlace *w/* stage 3	**33837**	**34164**	-292.12	-234145.58	1893	1.35e-01
aes	WireMask-EA	38660	58318	-177.86	-48986.96	547	4.26e-02
	EGPlace	32228	34538	-121.73	-13915.60	431	3.97e-02
	RollPlace	57638	68329	-243.78	-49069.95	564	4.31e-02
	C2FPlace *w/o* stage 3	28518	30546	-93.51	-10578.63	331	3.88e-02
	C2FPlace *w/* stage 3	**28071**	**30267**	**-85.26**	**-8798.57**	**329**	**3.86e-02**
aes_cipher_top	WireMask-EA	101754	119643	-293.34	-61544.95	596	9.77e-02
	EGPlace	103438	105149	-168.46	-32984.75	498	9.15e-02
	RollPlace	323464	340744	-768.79	-271215.72	935	1.52e-01
	C2FPlace *w/o* stage 3	54767	55137	-140.75	-26973.63	394	8.15e-02
	C2FPlace *w/* stage 3	**53654**	**54123**	**-136.97**	**-25533.20**	**387**	**8.12e-02**
des	WireMask-EA	9435	13090	-102.95	-2972.96	44	1.73e-02
	EGPlace	8224	8863	-94.09	-2490.54	**33**	1.64e-02
	RollPlace	23463	26892	-230.60	-9451.57	95	2.10e-02
	C2FPlace *w/o* stage 3	7631	7890	-77.57	-2310.32	34	**1.61e-02**
	C2FPlace *w/* stage 3	**7476**	**7781**	**-77.14**	**-2273.51**	34	**1.61e-02**
pci_bridge32	WireMask-EA	195019	254100	-782.01	-1018531.31	3250	1.72e-01
	EGPlace	185046	188753	-612.53	-555181.25	3285	1.31e-01
	RollPlace	281574	305189	-822.14	-658746.44	**2835**	1.52e-01
	C2FPlace *w/o* stage 3	74777	78749	**-413.01**	**-354008.03**	3151	**1.14e-01**
	C2FPlace *w/* stage 3	**73108**	**77547**	-471.76	-394838.00	3237	1.21e-01
Average Rank	WireMask-EA	3.80	4.00	4.00	4.20	3.40	3.60
	EGPlace	3.20	3.00	3.00	3.00	3.40	3.00
	RollPlace	5.00	5.00	5.00	4.80	3.60	4.20
	C2FPlace *w/o* stage 3	2.00	2.00	1.60	1.60	2.20	**2.00**
	C2FPlace *w/* stage 3	**1.00**	**1.00**	**1.40**	**1.40**	**2.00**	**2.00**

and the corresponding average reduction after detailed placement is 1.62%. For downstream PPA, C2FPlace with Stage 3 achieves the best average ranking in WNS (1.40), TNS (1.40), and NVP (2.00), while maintaining competitive power efficiency. These results demonstrate the scalability of C2FPlace and the effectiveness of Stage 3. However, Stage 3 improves average WNS and TNS rankings but degrades timing on `ac97_top` and `pci_bridge32`, showing that HPWL reduction does not always translate proportionally to downstream timing and power.

6 Conclusion

We propose C2FPlace, a coarse-to-fine macro placement framework that combines tournament selection, stochastic macro-subset reconstruction, and a two-phase reconstruction-ratio schedule with critical macro tuning. During the coarse-grained optimization stage, C2FPlace performs evolutionary search through stochastic partial rip-up and re-place, enabling diverse solution exploration while progressively refining promising placement structures. After global optimization, C2FPlace performs fine-grained critical macro tuning to compensate for discretization errors introduced by coarse-grid placement and further improve placement quality. Experiments demonstrate the effectiveness and scalability of C2FPlace. On the ISPD2005 benchmark, C2FPlace achieves the best average HPWL among all compared methods and outperforms the state-of-the-art methods EGPlace and RollPlace by 17.82% and 17.86%, respectively. Furthermore, evaluations on the ICCAD2025 benchmark demonstrate that C2FPlace scales to large-scale placement scenarios with thousands to tens of thousands of movable modules and achieves the best overall average ranking in both intermediate placement metrics and downstream PPA evaluations. These results demonstrate that C2FPlace provides an effective and scalable solution for modern macro placement by combining evolutionary search for global exploration with critical macro tuning for local precision optimization.

AI USE STATEMENT

In this work, generative AI tools were exclusively used for manuscript polishing and grammatical refinement. No AI tools were utilized for research design, implementation, data analysis, result interpretation, or scientific content generation. All final content has been fully reviewed and approved by the authors, who take full responsibility for this submission.

REFERENCES

Anthony Agnesina, Puranjay Rajvanshi, Tian Yang, Geraldo Pradipta, Austin Jiao, Ben Keller, Brucek Khailany, and Haoxing Ren. AutoDMP: Automated DREAMPlace-based macro placement. In *Proceedings of the International Symposium on Physical Design*, pp. 149–157, 2023.

Ameya R. Agnihotri, Satoshi Ono, and Patrick H. Madden. Recursive bisection placement: Feng shui 5.0 implementation details. In *Proceedings of the International Symposium on Physical Design*, pp. 230–232, 2005.

Melvin A Breuer. A class of min-cut placement algorithms. In *Proceedings of the Design Automation Conference*, pp. 284–290, 1977.

Tung-Chieh Chen, Zhe-Wei Jiang, Tien-Chang Hsu, Hsin-Chen Chen, and Yao-Wen 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*, pp. 1228–1240, 2008.

Chung-Kuan Cheng, Andrew B. Kahng, Ilgweon Kang, and Lutong Wang. RePlAce: Advancing solution quality and routability validation in global placement. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1717–1730, 2019.

Chung-Kuan Cheng, Chia-Tung Ho, and Chester Holtz. Net separation-oriented printed circuit board placement via margin maximization. In *Proceedings of the Asia and South Pacific Design Automation Conference (ASP-DAC)*, pp. 288–293, 2022a.

Ruoyu Cheng and Junchi Yan. On joint learning for solving placement and routing in chip design. In *Proceedings of Advances in Neural Information Processing Systems*, pp. 16508–16519, 2021.

Ruoyu Cheng, Xianglong Lyu, Yang Li, Junjie Ye, Jianye Hao, and Junchi Yan. The policy-gradient placement and generative routing neural networks for chip design. In *Proceedings of Advances in Neural Information Processing Systems*, pp. 26350–26362, 2022b.

James P Cohoon and William D Paris. Genetic placement. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 956–964, 1987.

M. R. Garey and D. S. Johnson. The rectilinear Steiner tree problem is NP-complete. *SIAM Journal on Applied Mathematics*, pp. 826–834, 1977.

Zijie Geng, Jie Wang, Ziyan Liu, Siyuan Xu, Zhentao Tang, Mingxuan Yuan, Jianye Hao, Yongdong Zhang, and Feng Wu. Reinforcement learning within tree search for fast macro placement. In *Proceedings of the International Conference on Machine Learning*, pp. 15402–15417, 2024.

Zijie Geng, Jie Wang, Ziyan Liu, Siyuan Xu, Zhentao Tang, Shixiong Kai, Mingxuan Yuan, Jianye Hao, and Feng Wu. LaMPlace: Learning to optimize cross-stage metrics in macro placement. In *Proceedings of the International Conference on Learning Representations*, 2025.

Meng-Kai Hsu, Yi-Fang Chen, Chau-Chin Huang, Sheng Chou, Tzu-Hen Lin, Tung-Chieh Chen, and Yao-Wen Chang. NTUplace4h: A novel routability-driven placement algorithm for hierarchical mixed-size circuit designs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1914–1927, 2014.

A.B. Kahng and Qinke Wang. Implementation and extensibility of an analytic placer. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 734–747, 2005.

Andrew B. Kahng, Ravi Varadarajan, and Zhiang Wang. RTL-MP: Toward practical, human-quality chip planning and macro placement. In *Proceedings of the International Symposium on Physical Design*, pp. 3–11, 2022.

Andrew B Kahng, Ravi Varadarajan, and Zhiang Wang. Hier-RTLMP: A hierarchical automatic macro placer for large-scale complex IP blocks. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1552–1565, 2024.

E.Q. Kang, Rung-Bin Lin, and E. Shragowitz. Fuzzy logic approach to VLSI placement. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, pp. 489–501, 1994.

Y. Lai, Y. Mu, and P. Luo. MaskPlace: Fast chip placement via reinforced visual representation learning. In *Proceedings of Advances in Neural Information Processing Systems*, 2022.

Y. Lai, J. Liu, Z. Tang, B. Wang, J. Hao, and P. Luo. Chipformer: Transferable chip placement via offline decision transformer. In *Proceedings of the International Conference on Machine Learning*, pp. 18346–18364, 2023.

Wuxi Li, Shounak Dhar, and David Z. Pan. UTPlaceF: A routability-driven FPGA placer with physical and congestion aware packing. In *Proceedings of the International Conference on Computer-Aided Design*, pp. 1–7, 2016.

Ximeng Li, Keyu Peng, Fuxing Huang, and Wenxing Zhu. Pef: Poisson's equation-based large-scale fixed-outline floorplanning. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 2002–2015, 2022.

Zhao Li, Ji Zhang, Jun Gao, et al. EGPlace: An efficient macro placement method via evolutionary search with greedy repositioning guided mutation. In *Proceedings of the Forty-second International Conference on Machine Learning*, pp. 13237–13255, 2025.

Peiyu Liao, Siting Liu, Zhitang Chen, Wenlong Lv, Yibo Lin, and Bei Yu. DREAMPlace 4.0: Timing-driven global placement with momentum-based net weighting. In *Proceedings of the Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 939–944, 2022.

Tao Lin, Chris Chu, and Gang Wu. POLAR 3.0: An ultrafast global placement engine. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pp. 520–527, 2015.

Yibo Lin, Shounak Dhar, Wuxi Li, Haoxing Ren, Brucek Khailany, and David Z Pan. DREAMPlace: Deep learning toolkit-enabled GPU acceleration for modern VLSI placement. In *Proceedings of the Annual Design Automation Conference*, pp. 1–6, 2019.

Lixin Liu, Bangqi Fu, Shiju Lin, Jinwei Liu, Evangeline FY Young, and Martin DF Wong. Xplace: An extremely fast and extensible placement framework. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1872–1885, 2024.

Jingwei Lu, Pengwen Chen, Chin-Chih Chang, Lu Sha, Dennis Jen-Hsin Huang, Chin-Chi Teng, and Chung-Kuan Cheng. ePlace: Electrostatics-based placement using fast Fourier transform and Nesterov's method. *ACM Transactions on Design Automation of Electronic Systems*, pp. 1–34, 2015a.

Jingwei Lu, Hao Zhuang, Pengwen Chen, Hongliang Chang, Chin-Chih Chang, Yiu-Chung Wong, Lu Sha, Dennis Huang, Yufeng Luo, Chin-Chi Teng, et al. ePlace-MS: Electrostatics-based placement for mixed-size circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 685–698, 2015b.

Yi-Chen Lu, Rongjian Liang, Wen-Hao Liu, and Haoxing Ren. 2025 ICCAD CAD Contest Problem C: Incremental placement optimization beyond detailed placement: Simultaneous gate sizing, buffering, and cell relocation. In *Proceedings of the IEEE/ACM International Conference On Computer-Aided Design (ICCAD)*, pp. 1–3, 2025.

Don MacMillen, Raul Camposano, Dwight Hill, and Thomas W Williams. An industrial view of electronic design automation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1428–1448, 2000.

Azalia Mirhoseini, Anna Goldie, Mustafa Yazgan, Joe Wenjie Jiang, Ebrahim Songhori, Shen Wang, Young-Joon Lee, Eric Johnson, Omkar Pathak, Azade Nova, et al. A graph placement methodology for fast chip design. *Nature*, pp. 207–212, 2021.

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*, pp. 1518–1524, 1996.

G.-J. Nam, C. J. Alpert, P. Villarrubia, B. Winter, and M. Yildiz. The ISPD2005 placement contest and benchmark suite. In *Proceedings of the 2005 International Symposium on Physical design*, pp. 216–220, 2005.

J.A. Roy, S.N. Adya, D.A. Papa, and I.L. Markov. Min-cut floorplacement. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1313–1326, 2006.

Y. Shi, K. Xue, L. Song, and C. Qian. Macro placement by wire-mask-guided black-box optimization. In *Proceedings of Advances in Neural Information Processing Systems*, 2023.

M. Shunmugathammal, C. Christopher Columbus, and S. Anand. A novel B* tree crossover-based simulated annealing algorithm for combinatorial optimization in VLSI fixed-outline floorplans. *Circuits, Systems, and Signal Processing*, pp. 900–918, 2020.

Peter Spindler and Frank M Johannes. Fast and accurate routing demand estimation for efficient routability-driven placement. In *Proceedings of 2007 Design, Automation & Test in Europe Conference & Exhibition*, pp. 1–6. IEEE, 2007.

Peter Spindler, Ulf Schlichtmann, and Frank M Johannes. Kraftwerk2—a fast force-directed quadratic placement approach using an accurate net model. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 1398–1411, 2008.

Qi Xu, Hao Geng, Song Chen, Bo Yuan, Cheng Zhuo, Yi Kang, and Xiaoqing Wen. GoodFloorplan: Graph convolutional network and reinforcement learning-based floorplanning. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 3492–3502, 2021.

Ke Xue, Ruo-Tong Chen, Xi Lin, Yunqi Shi, Shixiong Kai, Siyuan Xu, and Chao Qian. Reinforcement learning policy as macro regulator rather than macro placer. In *Proceedings of Advances in Neural Information Processing Systems*, pp. 140565–140588, 2024.

Qi Zhou, Guojun Liu, Guangzhi Qi, Ming Lu, Jiechu Liu, Zhongli Liu, Jianqun Yang, and Xingji Li. RollPlace: Improving macro placement via monte carlo rollout search. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pp. 3155–3168, 2025.

A Implementation Details

A.1 Detailed Optimization Procedures

This section provides the detailed optimization procedures of the coarse-to-fine refinement process. The complete pseudocode is presented in Algorithms 1 and 2. The optimization consists of two stages: coarse-grained evolutionary search and fine-grained critical macro tuning.

Algorithm 1 Coarse-Grained Evolutionary Search

Require: Initial population $\mathcal{P}$, macro set $\mathcal{M}$, iterations T, ratio intervals R
Ensure: Optimized placement $P^\star$

for $t = 1, \ldots, T$ **do**
  Randomly sample $\mathcal{T}$ individuals from $\mathcal{P}$
  Select parent:

$$P_{\text{par}} \leftarrow \arg\min_{P_i \in \mathcal{T}} \text{HPWL}(P_i)$$

  Sample rip-up ratio $\rho \sim \mathcal{U}(R)$
  $k \leftarrow \max(1, \lfloor \rho |\mathcal{M}| \rfloor)$
  Randomly select k macros: $\mathcal{S} \leftarrow \text{Sample}(\mathcal{M}, k)$
  Remove $\mathcal{S}$ from P_{par} to obtain partial layout $\widetilde{P}$
  Preserve original relative order: $\pi \leftarrow \text{SortByIndex}(\mathcal{S})$
  for each macro $m \in \pi$ **do**
    $c^\star \leftarrow \arg\min_{c \in \mathcal{L}_c(m|\widetilde{P})} W_m(c|\widetilde{P})$
    Reinsert m into $\widetilde{P}$
  end for
  Find worst individual:

$$P_{\text{worst}} \leftarrow \arg\max_{P_i \in \mathcal{P}} \text{HPWL}(P_i)$$

  if $\text{HPWL}(\widetilde{P}) < \text{HPWL}(P_{\text{worst}})$ **then**
    Replace P_{worst} with $\widetilde{P}$
  end if
end for
$P^\star \leftarrow \arg\min_{P_i \in \mathcal{P}} \text{HPWL}(P_i)$
return $P^\star$

A.2 Benchmark Circuit Statistics

Table 4 summarizes the circuit statistics of the ISPD2005 Nam et al. (2005) and ICCAD2025 Lu et al. (2025) benchmarks used in our experiments. For the ISPD2005 circuits, C2FPlace optimizes the macro subset, and thus the number of modules to place corresponds to the number of macros. For bigblue2 and bigblue4, we optimize only 1024 macros, following the setting described in the main paper. All macros are placed for the other six ISPD2005 circuits.

For the ICCAD2025 benchmarks, C2FPlace is applied to standard-cell placement, and "Modules to Place" therefore denotes the number of standard cells optimized by C2FPlace. The ariane circuit is excluded from the C2FPlace experiments because its placement setting differs from the other ICCAD2025 circuits. The table also reports the numbers of ports, nets, pins, and the corresponding area utilization to provide a detailed overview of the benchmark characteristics.

A.3 Hyperparameter Settings

Table 5 summarizes the hyperparameter settings used in C2FPlace. The coarse-grained evolutionary search is performed in two phases with different rip-up ratios. The first phase uses relatively large rip-up ratios to encourage broader exploration, while the second phase gradually reduces the rip-up ratio for more localized refinement. We use a population size of 20 and a tournament size of 5 throughout the evolutionary search.

Algorithm 2 Critical Macro Tuning via Structure-Preserving Coarse-Grid Displacement and Fine-Grid Offset Refinement

Require: Initial placement P, netlist $\mathcal{N}$, iterations T_3
Ensure: Optimized placement P

for $t = 1, \ldots, T_3$ **do**
  $\widetilde{P} \leftarrow P$
  Structure-Preserving Coarse-Grid Displacement
  Identify bounding-box critical macros $\mathcal{M}_{\mathrm{crit}}$ from $\mathcal{N}$ and $\widetilde{P}$
  for each macro $m \in \mathcal{M}_{\mathrm{crit}}$ **do**
    $(\mathbf{c}_m, \boldsymbol{\delta}_m) \leftarrow \mathrm{GetPosition}(m, \widetilde{P})$
    Remove m from $\widetilde{P}$
    Generate legal coarse locations $\mathcal{C}_m^{\mathrm{coarse}}$ satisfying footprint and structure-preserving constraints
    $\mathbf{c}_m^\star \leftarrow \arg\min_{\mathbf{c} \in \mathcal{C}_m^{\mathrm{coarse}}} W_m(\mathbf{c}, \boldsymbol{\delta}_m \mid \widetilde{P})$
    Reinsert m at $(\mathbf{c}_m^\star, \boldsymbol{\delta}_m)$ into $\widetilde{P}$
  end for
  Fine-Grid Offset Refinement
  Update bounding-box critical macros $\mathcal{M}_{\mathrm{crit}}$ from $\mathcal{N}$ and $\widetilde{P}$
  for each macro $m \in \mathcal{M}_{\mathrm{crit}}$ **do**
    $(\mathbf{c}_m, \boldsymbol{\delta}_m) \leftarrow \mathrm{GetPosition}(m, \widetilde{P})$
    Remove m from $\widetilde{P}$
    Generate feasible fine offsets $\mathcal{D}_m^{\mathrm{fine}}(\mathbf{c}_m)$ within the current coarse footprint
    $\boldsymbol{\delta}_m^\star \leftarrow \arg\min_{\boldsymbol{\delta} \in \mathcal{D}_m^{\mathrm{fine}}(\mathbf{c}_m)} W_m(\mathbf{c}_m, \boldsymbol{\delta} \mid \widetilde{P})$
    Reinsert m at $(\mathbf{c}_m, \boldsymbol{\delta}_m^\star)$ into $\widetilde{P}$
  end for
  if $\mathrm{HPWL}(\widetilde{P}) < \mathrm{HPWL}(P)$ **then**
    $P \leftarrow \widetilde{P}$
  end if
end for
return P

Table 4: Circuit statistics of the ISPD2005 and ICCAD2025 benchmarks.

Circuit	Macros	Standard Cells	Modules to Place	Ports	Nets	Pins	Area Util (%)
adaptec1	543	210904	543	0	221142	944053	55.62
adaptec2	566	254457	566	0	266009	1069482	74.46
adaptec3	723	450927	723	0	466758	1875039	61.51
adaptec4	1329	494716	1329	0	515951	1912420	48.62
bigblue1	560	277604	560	0	284479	1144691	31.58
bigblue2	23084	534782	1024	0	577235	2122282	32.43
bigblue3	1298	1095514	1298	0	1123170	3833218	66.81
bigblue4	8170	2169183	1024	0	2229886	8900078	35.68
ac97_top	0	7750	7750	132	7806	28406	56.23
aes	0	4393	4393	389	4653	17029	57.06
aes_cipher_top	0	11630	11630	388	11890	40834	64.64
ariane	136	105594	0	495	108198	441645	59.89
des	0	2317	2317	197	2441	7549	56.09
pci_bridge32	0	12091	12091	368	12254	45333	59.46

Table 5: Hyperparameter settings of C2FPlace.

Hyperparameter	Value
Population size	20
Tournament size	5
Stage 2 Phase 1 iterations	5,000
Stage 2 Phase 1 rip-up ratio	$[0.4, 0.7]$
Stage 2 Phase 2 iterations	5,000
Stage 2 Phase 2 rip-up ratio	$[0.1, 0.4]$
Critical macro tuning iterations	50
Coarse grid size	512
Random seeds	$\{0, 1, 2, 3, 4\}$
Hardware	AMD EPYC 9654 CPU

After the coarse-grained search, we perform 50 iterations of fine-grained critical macro tuning. Unless otherwise specified, all experiments use a coarse grid size of 512 and five random seeds $\{0, 1, 2, 3, 4\}$. The experiments are conducted on an AMD EPYC 9654 CPU.

A.4 PPA Evaluation Flow

We use OpenROAD v2.0 to evaluate the downstream PPA performance of the placements generated by C2FPlace. For each placement, the corresponding DEF and SDC files are loaded together with the ASAP7 technology and standard-cell libraries. We then perform detailed placement and estimate the parasitics based on the resulting placement. Finally, we report the total negative slack (TNS), worst negative slack (WNS), power, and the number of endpoints with setup violations (NVP). The complete Tcl script used for this evaluation is shown in Fig. 3, ensuring a consistent evaluation flow across all tested circuits.

```
set def_file $::env(DEF_FILE)
set sdc_file $::env(SDC_FILE)

foreach libFile [glob "../ASAP7/LIB/*nldm*.lib"] {
   read_liberty $libFile
}

read_lef ../ASAP7/techlef/asap7_tech_1x_201209.lef

foreach lef [glob "../ASAP7/LEF/*.lef"] {
   read_lef $lef
}

read_def $def_file
read_sdc $sdc_file

source ../ASAP7/setRC.tcl

detailed_placement
estimate_parasitics -placement

report_tns
report_wns
report_power

set nvp_setup [sta::endpoint_violation_count max]
puts "NVP (setup violation count): $nvp_setup"
```

Figure 3: OpenROAD v2.0 Tcl script for PPA evaluation.

B ADDITIONAL RESULTS

To evaluate the sensitivity of C2FPlace to key design choices, we conduct an ablation study on eight ISPD2005 benchmark Nam et al. (2005) circuits, focusing on coarse-grid granularity and rip-up ratio scheduling. We compare the proposed staged random-ratio schedule with ten fixed-ratio settings ranging from 0.1 to 1.0. Each heatmap cell in Fig. 4 reports the mean final-best HPWL over five random seeds.

To aggregate results across circuits, we use normalized HPWL instead of directly averaging raw HPWL values, since circuit scales vary significantly across the benchmark suite. For circuit c, coarse-grid size g, and ratio setting s, let $H_{c,g,s}$ denote the five-seed mean HPWL. The normalized metric is defined as:

$$\bar{R}_{g,s} = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \frac{H_{c,g,s}}{\min_{s' \in \mathcal{S}} H_{c,g,s'}}, \tag{6}$$

where $\mathcal{C}$ denotes the eight circuits and $\mathcal{S}$ contains all ratio settings, including fixed ratios and the staged random schedule. A lower $\bar{R}_{g,s}$ indicates better overall performance, while $\bar{R}_{g,s} - 1$ represents the average relative degradation from the best setting for each circuit.

B.1 IMPACT OF COARSE-GRID GRANULARITY

We first investigate the influence of coarse-grid granularity on placement quality. As shown in Fig. 4, increasing the grid size generally improves HPWL by providing finer placement resolution. However, the improvement is not monotonic and varies across circuits and rip-up ratios. The numerical scales are reported separately in the subcaptions, where `adaptec1` and `bigblue1` use $\times 10^5$ while the remaining circuits use $\times 10^6$.

The optimal granularity is circuit dependent. For example, `adaptec2` achieves its best result at `cg256`, while `adaptec3` and `bigblue2` obtain the lowest HPWL at `cg512`; the remaining circuits prefer `cg1024`. These results indicate that finer grids do not always guarantee better placement quality, as the effectiveness of granularity depends on the interaction between resolution and perturbation strength. Therefore, coarse-grid granularity should be considered as an important component of the coarse-to-fine optimization process rather than a fixed parameter shared across all circuits.

B.2 EFFECTIVENESS OF RANDOM RIP-UP RATIO

We evaluate the staged random-ratio schedule using the normalized mean HPWL. The random schedule achieves the best performance at `cg256`, `cg512`, and `cg1024`, with normalized HPWL values of 1.026, 1.038, and 1.069, respectively. Its advantage becomes more evident on finer grids: at `cg512` and `cg1024`, it outperforms the best-performing fixed ratio 0.6 (1.038 vs. 1.048 and 1.069 vs. 1.082). For coarse grids, fixed ratios remain slightly competitive, with fixed ratio 0.7 achieving the best result on `cg128` and fixed ratio 0.6 on `cg224`.

Averaged over all coarse-grid sizes, the staged random schedule achieves the lowest normalized HPWL of 1.039, followed by fixed ratio 0.6 with 1.044. These results demonstrate that the staged schedule provides a robust performance across different grid granularities without requiring circuit-specific ratio tuning. While individual fixed ratios may still achieve better results on specific cases, the random schedule offers a strong overall trade-off under a unified optimization strategy.

B.3 ANALYSIS OF RATIO SCHEDULING STRATEGY

The fixed-ratio results show that no single ratio consistently achieves the best HPWL across all benchmark circuits. The optimal fixed ratio varies significantly, ranging from 0.2 on `adaptec4` to 1.0 on `adaptec1` and `bigblue3`, indicating that the preferred perturbation strength depends on circuit characteristics and grid granularity. It should be noted that for all fixed-ratio baselines, the rip-up ratio in Stage 2 Phase 2 is set to half of that used in Stage 2 Phase 1.

C2FPlace addresses this sensitivity with a staged random-ratio schedule. The global-exploration stage samples rip-up ratios from $[0.4, 0.7]$, while the local-refinement stage uses $[0.1, 0.4]$. This design provides stronger perturbations during exploration and more conservative updates during refinement, while avoiding dependence on a manually selected fixed ratio.

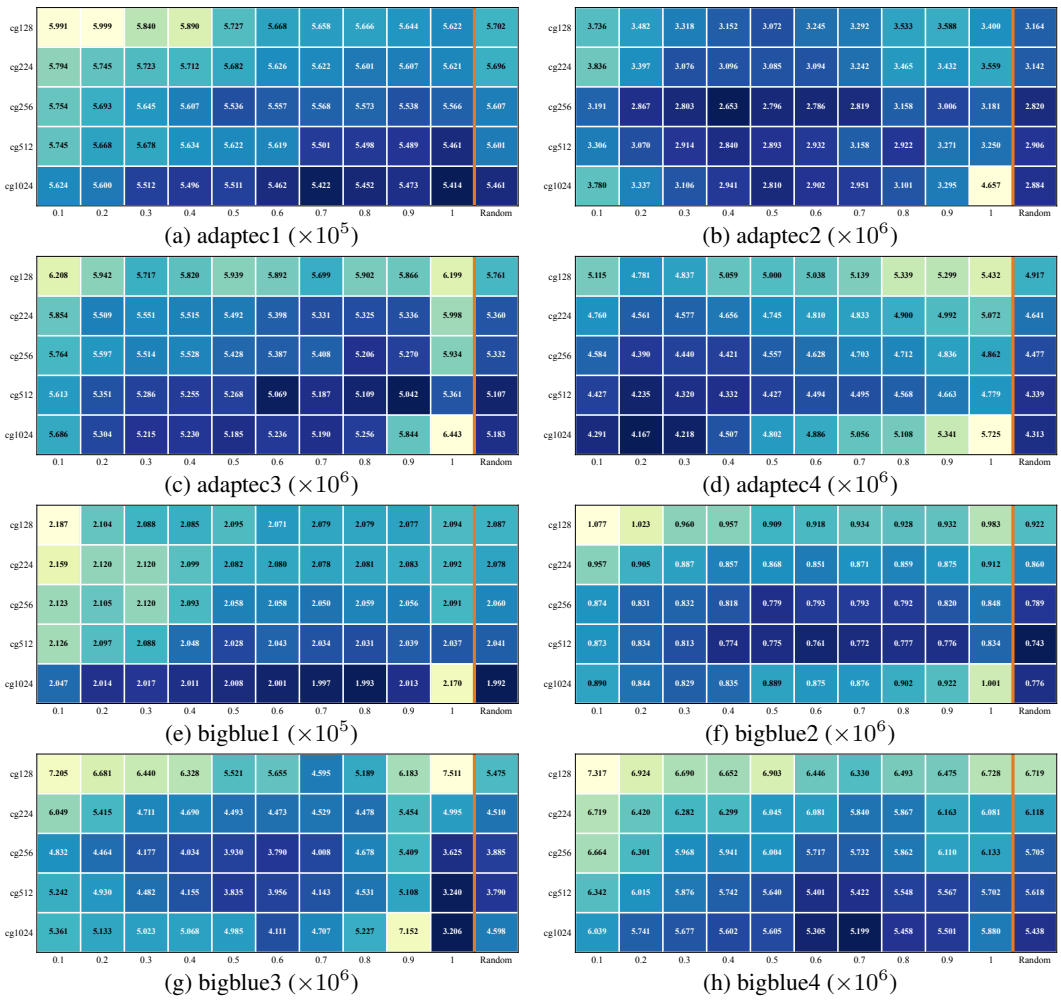


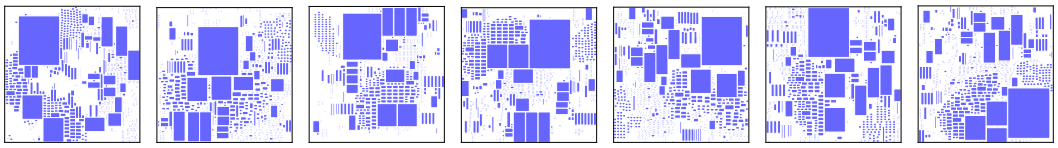
Figure 4: Mean final-best HPWL over five seeds for fixed and random ratio settings. Columns in each panel correspond to fixed-ratio settings, followed by the random-ratio strategy after the orange separator; rows correspond to coarse-grid sizes.

Overall, the normalized results demonstrate that the staged schedule achieves the best average performance across different coarse-grid sizes, particularly on `cg256`, `cg512`, and `cg1024`. Although fixed ratios remain competitive on `cg128` and `cg224`, the proposed schedule provides a robust default across diverse benchmarks without circuit-specific tuning.

B.4 Visualization Results

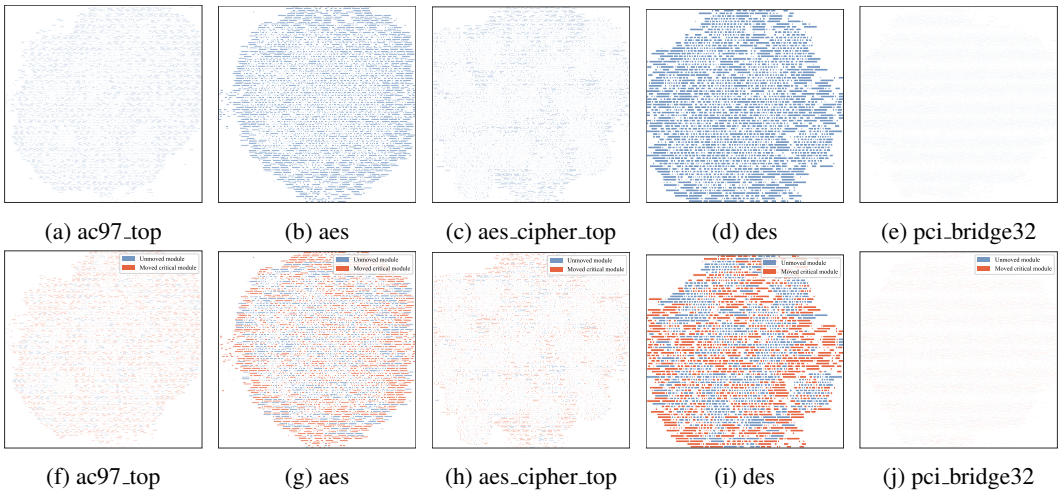
Figure 5 presents a visual comparison of macro placements produced by different methods on `adaptec3`.

Figure 6 visualizes the placement refinement process (Stage 3) of C2FPlace on five ICCAD2025 benchmark circuits. It can be observed that a substantial number of critical macros are moved during the refinement process, and these movements are accepted only when they lead to improved solutions. This demonstrates that Stage 3 can effectively further reduce HPWL.



(a) DMPlace 7.239×10^6 (b) MaskPlace 8.151×10^6 (c) WireM-EA 5.864×10^6 (d) EffiPlace 5.551×10^6 (e) EGPlace 5.761×10^6 (f) RollPlace 5.251×10^6 (g) **C2FPlace** 4.791×10^6

Figure 5: Macro placement visualization for `adaptec3` circuit comparing C2FPlace with five recent methods. Abbreviations: DMPlace = DREAMPlace, WireM-EA = WireMask-EA, EffiPlace = EfficientPlace. Blue rectangles denote macros. The placement quality is evaluated through HPWL value, where lower HPWL values indicate superior performance.



(a) ac97_top (b) aes (c) aes_cipher_top (d) des (e) pci_bridge32

(f) ac97_top (g) aes (h) aes_cipher_top (i) des (j) pci_bridge32

Figure 6: Placement visualization of five ICCAD2025 benchmark circuits. The first and second rows correspond to Stage 2 and Stage 3 optimization, respectively. Blue modules remain unchanged, while red indicate tuned critical modules.