

Late Breaking Results: RL-Based Macro Placement with Cell Clustering and Rudy Modeling for Routability Optimization

Youwen Wang*, Hao Gu*, Xinglin Zheng, Keyu Peng, Ziran Zhu[†]

National ASIC System Engineering Research Center, Southeast University, Nanjing, China

Email: {220236428, 230218727, 220236395, kypeng, zrzh}@seu.edu.cn

Abstract—Macro placement critically impacts the routability of modern very large scale integration (VLSI) circuits. While reinforcement learning (RL) has emerged as a promising solution, existing approaches often optimize solely for HPWL while neglecting standard cell information, leading to severe routability degradation. To overcome these limitations, we propose a novel congestion-aware RL-based macro placement method that jointly optimizes wirelength and routability. Specifically, we introduce a hypergraph-based standard cell clustering technique to preserve netlist connectivity and utilize the RUDY mask for high-fidelity congestion estimation. To empower the RL agent, we construct a Swin Transformer-based U-Net architecture, which enhances feature extraction and transferability across designs. To effectively guide this agent, we also incorporate a local overlap-allowed strategy to expand the search space, combined with a multi-objective reward function that balances wirelength and congestion metrics. Experimental results on the ISPD 2015 benchmarks demonstrate that our method significantly outperforms state-of-the-art RL methods, routability-driven analytical placers, and the commercial tool Innovus, achieving superior macro placement quality.

Index Terms—Electronic design automation, physical design, reinforcement learning, macro placement, routability.

I. INTRODUCTION

Macro placement is a critical stage in VLSI physical design that directly determines downstream routing quality. Traditional methods, including partitioning-based, simulated annealing, and analytical approaches, provide foundational solutions but often suffer from sub-optimal placement quality or high computational cost due to mismatched objectives, slow convergence, or approximated optimization functions. Recently, RL-based approaches such as AlphaChip [1], ChiPFormer [2] and MaskRegulate [3] have shown promise by formulating macro placement as a sequential decision-making problem. However, these methods generally overlook standard cell information, rely on coarse grid representations, and optimize only proxy metrics like HPWL. This neglect of fine-grained congestion limits routability and degrades post-routing performance.

To address these challenges, we propose a congestion-aware RL-based macro placement method that integrates standard cell clustering and RUDY modeling to improve routability and jointly optimize wirelength.

II. THE PROPOSED METHOD

Fig.1 illustrates the overall flow of our proposed RL-based macro placement method. Our method is built upon an online learning framework with PPO strategy [4]. Specifically, this method starts with a preprocessing, where standard cells are clustered into macros and jointly placed together with existing macros. To explicitly model routing congestion, we generate the RUDY mask that serves as critical congestion features to guide our RL agent. Subsequently, the core macro placement stage utilizes a novel RL agent that incorporates a U-Net architecture with a Swin Transformer backbone [5]. This agent processes multi-channel input masks to output action probability matrix, guided by a local overlap-allowed strategy and a multi-objective reward function. Following macro placement, we leverage DREAMPlace [6] for fast legalization and mixed-size placement optimization.

This work was partially supported by the National Key Research and Development Program of China under Grant 2022YFB4400500, and the National Natural Science Foundation of China under Grant 92373207. (*Youwen Wang and Hao Gu contributed equally to this work.) ([†]Corresponding author: Ziran Zhu.)

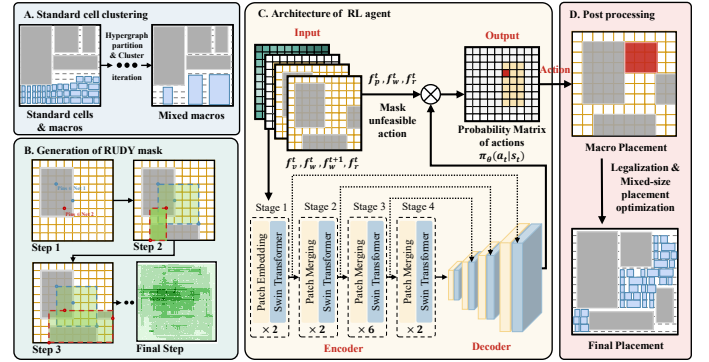


Fig. 1. Overview of our proposed method.

A. Standard Cell Clustering

To incorporate standard cells information into macro placement, we first cluster standard cells into macro-like blocks, as illustrated in Fig.1-A. The target cluster count k is derived from the ratio of total standard cell area to the average macro area. We formulate this process as a hypergraph partitioning problem. Let $H = (V, E)$ denote the hypergraph, where vertices V represent cells and hyperedges E represent nets. To prioritize critical connections, each net e is assigned a weight $\omega(e) = 1/(|e| - 1)$, where $|e|$ denotes the degree (number of connected cells) of net e . The clustering objective is to minimize the total weighted cut cost Φ subject to area balance constraints:

$$\begin{aligned} \text{minimize } \Phi &= \sum_{e \in E_{cut}} \omega(e), \\ \text{s.t. } \mathcal{A}(V_i) &\leq \frac{1+\epsilon}{k} \mathcal{A}(V), \quad \forall i \in \{1, \dots, k\} \end{aligned} \quad (1)$$

where E_{cut} is the set of hyperedges spanning multiple partitions, $\mathcal{A}(\cdot)$ denotes the area function, and ϵ represents the maximum allowable imbalance ratio. Through this recursive optimization, standard cells are hierarchically grouped into high-connectivity clusters. Finally, each cluster is modeled as a square macro with an area equivalent to $\mathcal{A}(V_i)$, forming a set of mixed macros for the subsequent RL process.

B. Visual Representation with RUDY Mask

To explicitly capture congestion information during macro placement, we introduce the RUDY mask, a density-based feature map derived from the Rectangular Uniform wire DensitY model [7]. In contrast to prior works [2, 3] that rely solely on macro-level connectivity, we integrate standard cell netlist information to achieve more accurate congestion estimation. The mask is constructed by iterating through nets and accumulating wire density values within their respective pin bounding boxes. After normalization, this process yields a continuous 224×224 matrix (visualized in Fig.1-B), where higher values correspond to elevated routing demand. This high-fidelity representation embeds fine-grained routability awareness directly into the RL state space.

C. Detailed Model Architecture Design

Fig.1-C illustrates the architecture of our RL agent, which employs a PPO2-based Actor-Critic framework [4, 8]. Distinguishing our work from prior CNN-based approaches [3, 9], we construct a U-Net backbone using the Swin Transformer [5]. By leveraging self-attention

TABLE I
EXPERIMENT RESULT OF MACRO PLACEMENT ON ISPD2015 BENCHMARK.

| Benchmark | ChiPFormer [2] | | | MaskRegulate [3] | | | DREAMPlace 4.0 [6] | | | Innovus | | | Ours | | |
|--------------------|----------------|-------------------|-------------------|------------------|-------------------|-------------------|--------------------|-------------------|-------------------|-------------|-------------------|-------------------|--------------------------|-------------------|-------------------|
| | WL↓ | CR _H ↓ | CR _V ↓ | WL↓ | CR _H ↓ | CR _V ↓ | WL↓ | CR _H ↓ | CR _V ↓ | WL↓ | CR _H ↓ | CR _V ↓ | WL↓ | CR _H ↓ | CR _V ↓ |
| mgc_des_perf_a | 2.71 | 5.11% | 16.05% | 3.44 | 20.25% | 25.05% | 2.86 | 6.53% | 22.50% | 2.47 | 2.75% | 2.73% | 2.31 ² | 0.63% | 1.75% |
| mgc_edit_dist_a | 4.71 | 13.08% | 20.09% | 4.93 | 37.00% | 55.72% | 5.14 | 31.24% | 42.31% | 4.52 | 9.04% | 12.26% | 4.55 | 6.37% | 10.40% |
| mgc_fft_a | 1.01 | 3.22% | 1.28% | 1.83 | 50.17% | 22.88% | 1.08 | 5.30% | 0.96% | 0.91 | 4.38% | 0.91% | 0.96 | 3.66% | 1.04% |
| mgc_fft_b | 1.24 | 10.74% | 4.67% | 2.29 | 66.76% | 24.71% | 1.35 | 21.76% | 4.58% | 1.20 | 8.02% | 4.72% | 1.22 | 7.23% | 2.94% |
| mgc_matrix_mult_a | 4.93 | 2.72% | 2.51% | 9.79 | 38.59% | 27.79% | 4.69 | 2.08% | 8.61% | 5.08 | 13.85% | 10.08% | 4.47 | 1.70% | 2.06% |
| mgc_matrix_mult_b | 5.60 | 6.97% | 17.53% | 9.41 | 83.57% | 41.24% | 4.95 | 5.66% | 4.78% | 4.97 | 7.08% | 5.17% | 4.83 | 4.49% | 3.88% |
| mgc_matrix_mult_c | 4.99 | 10.80% | 6.12% | 9.37 | 58.00% | 48.24% | 5.68 | 12.71% | 11.53% | 6.70 | 30.94% | 10.83% | 4.80 | 9.23% | 3.08% |
| mgc_pci_bridge32_a | 0.52 | 3.05% | 6.13% | 0.59 | 4.71% | 10.27% | 0.67 | 3.21% | 5.02% | 0.49 | 3.20% | 3.36% | 0.52 | 2.83% | 6.09% |
| mgc_pci_bridge32_b | 0.68 | 0.58% | 0.63% | 0.83 | 6.85% | 3.04% | 1.06 | 1.36% | 2.02% | 1.25 | 2.46% | 1.69% | 0.63 | 0.26% | 0.31% |
| mgc_superblue11_a | 49.69 | 2.54% | 0.28% | 82.32 | 1.49% | 0.80% | 53.21 | 2.14% | 2.80% | 58.56 | 4.69% | 0.89% | 48.41 | 0.82% | 0.38% |
| mgc_superblue12 | 50.66 | 2.80% | 2.79% | 50.04 | 1.05% | 2.93% | 52.18 | 4.02% | 18.03% | 49.42 | 3.24% | 12.92% | 49.05 | 1.33% | 2.33% |
| mgc_superblue14 | 28.49 | 1.83% | 0.72% | 44.65 | 3.82% | 6.69% | 30.28 | 1.61% | 0.85% | 29.86 | 2.34% | 0.64% | 26.99 | 1.63% | 0.48% |
| mgc_superblue16_a | 30.07 | 3.39% | 0.94% | 45.71 | 9.53% | 4.41% | 33.22 | 20.29% | 1.43% | 32.65 | 11.62% | 0.84% | 29.96 | 3.19% | 0.94% |
| mgc_superblue19 | 22.29 | 3.47% | 0.96% | 26.12 | 0.84% | 1.28% | 22.10 | 4.76% | 3.46% | 23.22 | 4.50% | 5.27% | 21.78 | 1.28% | 0.91% |
| Average | 14.83 | 5.02% | 5.76% | 29.04 | 21.21% | 16.05% | 15.60 | 8.76% | 9.28% | 15.81 | 7.72% | 5.17% | 14.32 | 3.19% | 2.61% |
| Ratio | 1.04 | 1.57 | 2.21 | 2.03 | 6.65 | 6.14 | 1.09 | 2.75 | 3.55 | 1.10 | 2.42 | 1.98 | 1.00 | 1.00 | 1.00 |

¹ WL value ($\times 10^6$ μm) is the wirelength reported by Innovus.

² The optimal value for each metric in every case across all methods is highlighted in bold black in the table.

³ The gray highlight indicates that MaskRegulate failed to generate a legal placement and are excluded from average metric calculations.

mechanisms, this architecture captures long-range spatial dependencies more effectively than conventional CNNs, thereby enhancing scalability and transferability on complex placement. The encoder adopts a four-stage hierarchical structure that progressively downsamples the input state at placement step t , comprising view (f_v^t), wire (f_w^t, f_w^{t+1}), and RUDY (f_r^t) masks. A symmetric decoder subsequently upsamples these features via skip connections to generate the action probability matrix. This output is constrained by the position (f_p^t), wire (f_w^t), and RUDY (f_r^t) masks to restrict the valid action space and guide macro placement decisions.

To further enhance placement flexibility and allow masks to accommodate mixed macros, we introduce a local overlap-allowed strategy. Under this scheme, non-integer macros are aligned to grid boundaries, while temporary overlaps are permitted within local regions to expand the action space. This relaxation allows the agent to learn more realistic spatial relationships compared to strict non-overlapping constraints.

Furthermore, to jointly optimize wirelength, congestion, and overlap, we formulate a unified multi-objective reward function. Since solely minimizing HPWL often lead to severe routing congestion, we incorporate a RUDY-based congestion metric as an auxiliary objective, alongside overlap penalties to enforce legality. While HPWL reduction $\Delta\text{HPWL}(s_t)$ and overlap penalties $\text{Overlap}(s_t)$ are computed step-wise, congestion is evaluated only at the terminal state to avoid conflicts with immediate HPWL reduction. Unlike RALF [10], which relies on costly iterative net-overlap computation, we estimate congestion penalty $R_c(s_T)$ by aggregating the RUDY values of the top-5% congested grid cells. The overall RL reward for an episode R_T is defined as:

$$R_T = \underbrace{\sum_{t=1}^T (\Delta\text{HPWL}(s_t) + \alpha \cdot \text{Overlap}(s_t))}_{\text{step-wise (accumulated) reward}} + \underbrace{\beta \cdot R_c(s_T)}_{\text{final congestion reward}} \quad (2)$$

α and β are weighting coefficients balancing our multi-objectives.

D. Final Placement Evaluation

After extensive training, our RL agent converges to generate the final mixed macro placement. Following this stage, DREAMPlace is employed for fast legalization and mixed-size placement optimization. Finally, the macro placement results are exported to a commercial EDA tool for cell placement, routing and routability evaluation.

III. EXPERIMENTAL RESULTS

In our experiments, we evaluated our method using the ISPD 2015 contest benchmark suite [11], excluding fence region constraints to focus on macro placement. The framework is implemented in Python with PyTorch and executed on a Linux workstation (Ubuntu 20.04) equipped with dual 64-core CPUs and an NVIDIA RTX 4090 GPU. For our reward configuration, the weighting coefficients are set to $\alpha = -2 \times 10^6$

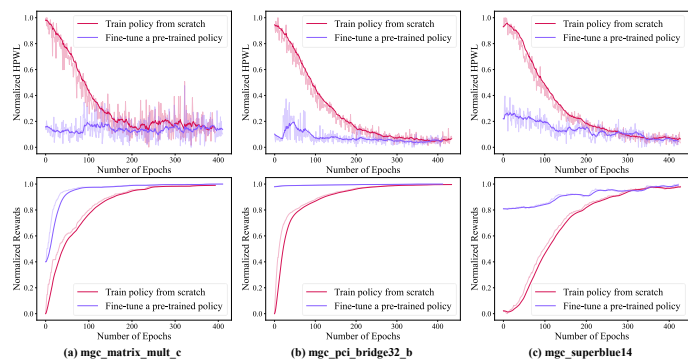


Fig. 2. Transferability evaluation of our RL-based model.

and $\beta = -1.5$. Following the commercial evaluation flow described in [12], we first generate the initial macro placement via our RL agent, followed by legalization and mixed-size placement in DREAMPlace [6]. The design is then exported to commercial tool Cadence Innovus 20.1 [13]. To validate the superiority of our initial macro placement and ensure experimental fairness, we re-implement cell placement and routing by executing `unplaceAllInsts`, `place_design`, and `earlyGlobalRoute` sequentially.

As shown in Table I, we compare our approach with SOTA RL-based methods, ChipFormer [2] and MaskRegulate [3], utilizing their open-source implementations. Our method demonstrates consistently superior routability, achieving average reductions of 1.83% in CR_H (horizontal congestion) and 3.15% in CR_V (vertical congestion) over ChipFormer, with even more substantial gains of 18.02% in CR_H and 13.44% in CR_V over MaskRegulate. Wirelength performance remains highly competitive, yielding improvements of 3.42% and 50.68% over these two baselines, respectively. Furthermore, we extend the comparison to the routability-driven analytical placer DREAMPlace and the Innovus (executing `place_design -concurrent_macros`) [13]. Our approach significantly outperforms both baselines, reducing CR_H by 5.57% over DREAMPlace and 4.53% over Innovus, while lowering CR_V by 6.67% and 2.56%, respectively. Meanwhile, wirelength is shortened by 8.21% and 9.42%. These results highlight the effectiveness of our congestion-aware framework in enhancing both placement optimization and downstream routability.

Finally, to evaluate transferability of our RL-based model, we deploy the model pre-trained exclusively on `mgc_superblue19` to other unseen designs. As illustrated in Fig.2, leveraging these pre-trained weights provides a superior initialization with lower HPWL, accelerates convergence, and significantly reduces the overall training overhead across benchmarks of varying scales.

REFERENCES

- [1] Azalia Mirhoseini, Anna Goldie, Mustafa Yazgan, Joe Wenjie Jiang, Ebrahim Songhori, Shen Wang, Young-Joon Lee, Eric Johnson, Omkar Pathak, Azade Nazi, et al. A graph placement methodology for fast chip design. *Nature*, 594(7862):207–212, 2021.
- [2] Yao Lai, Jinxin Liu, Zhentao Tang, Bin Wang, Jianye Hao, and Ping Luo. ChiPFormer: Transferable chip placement via offline decision transformer. In *Proceedings of ACM International Conference on Machine Learning (ICML)*, pages 18346–18364. PMLR, 2023.
- [3] 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. *Advances in neural information processing systems (NeurIPS)*, 37(93943):140565–140588, 2024.
- [4] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- [5] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10012–10022, 2021.
- [6] Peiyu Liao, Dawei Guo, Zizheng Guo, Siting Liu, Yibo Lin, and Bei Yu. Dreamplace 4.0: Timing-driven placement with momentum-based net weighting and lagrangian-based refinement. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, 42(10):3374–3387, 2023.
- [7] Peter Spindler and Frank M Johannes. Fast and accurate routing demand estimation for efficient routability-driven placement. In *Proceedings of IEEE Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 1–6. IEEE, 2007.
- [8] Vijay Konda and John Tsitsiklis. Actor-critic algorithms. *Advances in Neural Information Processing Systems (NeurIPS)*, 12, 1999.
- [9] Yao Lai, Yao Mu, and Ping Luo. MaskPlace: Fast chip placement via reinforced visual representation learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 35(1744):24019–24030, 2022.
- [10] Jakob Ratschenberger and Harald Pretl. Ralf: A reinforcement learning assisted automated analog layout design flow. *Authorea Preprints*, 2024.
- [11] Ismail S Bustany, David Chinnery, Joseph R Shinnerl, and Vladimir Yutsis. ISPD 2015 benchmarks with fence regions and routing blockages for detailed-routing-driven placement. In *Proceedings of IEEE/ACM Symposium on International Symposium on Physical Design (ISPD)*, pages 157–164, 2015.
- [12] Chung-Kuan Cheng, Andrew B Kahng, Sayak Kundu, Yucheng Wang, and Zhiang Wang. Assessment of reinforcement learning for macro placement. In *Proceedings of the 2023 International Symposium on Physical Design (ISPD)*, pages 158–166, 2023.
- [13] Cadence Design Systems, Inc. INNOVUS Implementation System, version 20.1, San Jose, CA, USA. 2020.