JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

IMPLEMENTATION OF MACHINE LEARNING IN THE FIELD OF VLSI PLACEMENT: A REVIEW

[1]Bindushree V, [2]R Jaya Gowri

[1]Student, [2]Associate Professor
[1]Electonics and communication,
[1]BMS College of Engineering, Bengaluru, India

Abstract: — In order to change the entire chip design methodology, the Very Large-Scale Integration (VLSI) sector has begun implementing machine learning (ML) techniques in design automation. The time and effort required to comprehend and process the data within and across different abstraction levels via automated learning algorithms is reduced when machine learning techniques, such as reinforcement learning, are used in VLSI design and placement. It thus increases IC yield, shortens production lead times, and ensures accurate placement. In light of this, a thorough examination of numerous Reinforcement Learning-related aspects in the context of VLSI placement has been carried out. We have also considered the recent machine learning and deep learning techniques incorporated in VLSI.

Index Terms - Reinforcement learning, machine learning, algorithms, placement, design.

I. INTRODUCTION

The process of integrating thousands of transistors onto a semiconductor chip can be described as very large-scale integration. VLSI first appeared in the 1970s, when advanced level processor chips were in their infancy. The most common VLSI devices are microprocessors and microcontrollers.

As previously indicated, the term "Very Large-Scale Integration" (VLSI) was first used to describe integrated circuits in the 1970s. The amazing expansion of the electronics industry in terms of sophistication and the volume of devices produced annually has been made possible by numerous developments in the field of VLSI. The existence of the chip design industry has led to significant growth in the domains of telecommunications, control systems, consumer electronics, high performance computing, missiles, etc [1]. These applications experience processing speeds and application access that are unheard of, and VLSI makes this all feasible. As long as there are inventions and a very fast rate of growth in the VLSI sector to support those inventions, there will always be a market for these products.

Due to the large cell count and great accuracy in how the cells are put in a chip that are required for Very Large-Scale Integration, the idea of traditional design that is done manually is rendered obsolete [2]. Any designer would find it extremely difficult to complete a project of this size without the aid of technology. Electronic Design Automation (EDA) tools were introduced as a result, assisting designers in increasing design and verification efficiency. The development of several tools for each level of VLSI is the main goal of EDA. However, having EDA tools alone is insufficient for design because it necessitates a fundamental understanding of VLSI and its characteristics. This may aid in bridging the gap between specification and chip production. The creators ought to be able to formulate a problem and an algorithm for the same problem and then develop a program to automate physical design on a computer.

The stage that handles the placement and routing of a chip in an IC design is called Physical Design, which is at the tail end of the VLSI flow. The circuit representations from the earlier phases are transformed into shapes, and these shapes are subsequently formed into various metal layers. An IC Layout is the name given to this geometric illustration.

A VLSI circuit's design must go through several stages because of its complexity. Placement is carried out after logic synthesis but before CTS and routing in accordance with the ASIC Design flow. Therefore, it is understood that each standard cell and macro has a specific placement on the chip. The placement stage is where this intricate procedure of inserting about a million cells and several macros takes place [3].

The rows are first introduced so that the standard cells can be placed before Placement. Additionally, the common cells are positioned in these rows during placement. It is necessary to take timing requirements and net lengths into account while positioning the standard cells [4] Otherwise, many iterative loops must be performed to enhance the current location, which may or may not enhance the timing. Runtime will once more suffer as a result of this.

Reinforcement Machine learning includes learning. Agents learn about reward and punishment systems on their own in this situation [18]. It's about using observations to determine the best course of action or path to achieve the greatest rewards and the least amount of punishment.

This paper is divided into the subsequent sections, where Section I gives the Introduction, Section II depicts the extensive Literature Review done so far in Physical Design, Section III includes a summarization of papers in tabular form and Section IV contains the conclusion.

II. LITERATURE REVIEW

An immense amount of research has been done regarding the implementation of ML algorithms in the VLSI placement. This review takes into consideration various ML models, Reinforcement learning techniques, deep RL techniques and algorithms implemented to boost the performance of placement tools, optimize, and effortlessly automate the placement process in the PD flow.

According to Z. Wang et al. [1] reward and state transition functions of dynamic settings may change over time, which is why this work addresses the incremental RL problem in continuous spaces for these environments. The aim was to switch from the initially learned policy in the original environment to a new one whenever the environment changes. With the incremental learning process, authors present a two-step strategy to increase adaptability: policy relaxation and importance weighting. A proper exploration of the new environment is the first goal of the policy relaxation mechanism, which achieves this by lowering the behavior expectations for a few learning episodes to a consistent level. This results in a better long-term adaption by reducing the conflict between the new knowledge and the previously held beliefs, they're adapted to. The second step is the application of an importance weighting technique based on the finding that episodes with greater returns are more consistent with the new environment and therefore contain more novel information. In order to encourage the prior optimal policy to be quickly replaced by a new one that works in the new environment, they provide larger weights during parameter update to episodes that contain more new information. Traditional navigation challenges and intricate locomotion tasks with various configurations were the subjects of experiments. The outcomes demonstrated that the suggested approach could manage a variety of dynamic situations and deliver a substantially faster learning process.

A. Agnesina et al. [2] the physical design flow depends on the placement's quality. A human engineer often devotes a significant amount of time to fine-tuning the various settings of a commercial placer in order to meet PPA goals. In order to optimize the placement settings of a commercial EDA tool, this study suggests a deep reinforcement learning (RL) architecture. Researchers create an autonomous agent that is taught exclusively by RL via self-search and learns to tune parameters optimally without the assistance of humans or domain expertise. Researchers combine manually created characteristics from graph topology theory with graph embeddings produced by unsupervised Graph Neural Networks to generalize to unseen netlists. The sparsity of the data and the latency of the placement runs are overcome by their RL algorithms. When compared to a human engineer and a state-of-the-art tool auto-tuner, their trained RL agent improves wirelength on unseen netlists by up to 11% and 2.5%, respectively, in just one placement iteration (20X and 50X less iteration).
A. Mansoor et al. [3] have implemented unique placement method (RS3DPlace) based on Simulated Annealing (SA) and Reinforcement Learning (RL), which is the first machine learning strategy for Monolithic 3D ICs (M3D). RS3DPlace rapidly calculates a draught solution using RL's capacity for learning, which SA then uses to produce a better final solution. Although the gate-level M3D design style is the focus of the present implementation, it may be applied to other M3D design styles as well as other 2D and 3D physical design optimization issues. We evaluated RS3DPlace for 8-128-bit MUX-based right arithmetic shifter circuits and a circuit with non-regular connections in comparison to Mux-based shifters, which are optimized in 2- layered M3D technology, to demonstrate the efficiency of the technique. Additionally, according to experimental findings, the total cost function is on average 16% better than it is with Random Initialized SA (Rand SA).
Mrinal Mathur [4] demonstrates that solving time-consuming placement-based activities requires focusing on complicated, industry-wide problems with a big impact. They provide a fresh RL-based method for placing the macros quickly and effectively to maximize PPA values. Designers demonstrate that they have produced placements with improved outcomes and outperformed state-of-the-art baselines. These findings demonstrate that their agent reduced wirelength without incurring any additional training costs and generalized well when compared to EDA technologies.

S. F. Almeida et al. [5] the placement engine may generate an impractical routing solution as a result of the search for wirelength optimization, necessitating the repetition of earlier processes and raising the total project cost. Due of its cheap computing cost, placement algorithms have historically used pin density to determine routability. This has turned out to be inefficient at advanced technology nodes, nevertheless, because of tighter production regulations and complicated standard cell layouts. Although routeability is a topic that many placement strategies aim to solve, the issue is that these models rely on certain heuristics or designer expertise. As a result, researchers provide a methodology based on machine learning for addressing routeability during the placement stage.

According to R. Manimegalai et al. [6] for the Placement and Routing problem on 3D-FPGA, a Reinforcement Learning-based solution based on Support Vector Machines is proposed in this study. their experimental findings on typical benchmark circuits show that their method leads in efficient routing with reduced connection length and channel width. The performance of the RL and SVM combination is very good, even with only a few training trajectories. Fixed routing method is assumed in this paper. RL methods can also be used to enhance the current routing algorithm. For the Placement and Routing problem on 3D-FPGA, a Reinforcement Learning-based solution based on Support Vector Machines is proposed in this study. their experimental findings on typical benchmark circuits show that their method leads in efficient routing with reduced connection length and channel width. The performance of the RL and SVM combination is very good, even with only a few training trajectories. Fixed routing method is assumed in this paper. RL methods can also be used to enhance the current routing algorithm.

R. Cheng et al [7] when it comes to macro placement, reinforcement learning is been used, but when it comes to standard cell placement, which is more laborious and time-consuming, a more efficient gradient-based optimization technique is used and effectively integrate this technique into the end-to-end pipeline. Researchers go one step farther and create previously unresearched joint positioning and routing using reinforcement learning. A two-view based embedding model is created in particular to combine global and local information, and distillation is created to enhance exploration. This solver's effectiveness is demonstrated by experimental results on public benchmarks. Due to the GNN's scaling problem and the huge action space for reinforcement learning, this model can only place a moderate amount of macros at the moment.

L. Bai et al. [8] after a floorplan has been constructed, machine learning algorithms for time prediction are used in this study. The parameters from the gate-level netlist, constraint files, and floorplan files are examined to choose and abstract the features for the models. We investigate the traditional machine learning techniques including neural networks, support vector machines (SVM), and ensemble

machine learning. To forecast the time of SoC, the appropriate regression models are used. The suggested concept is validated using the testcase created by open-source IP core. The outcomes demonstrate that, among the learning models tested in this paper, the hybrid ensemble learning model has the greatest prediction performance.

E. C. Barboza et al. [9] have introduces a pre-routing timing prediction technique based on machine learning. According to experimental findings, it can achieve accuracy close to post-routing sign-off analysis. It decreases the false positive rate in detecting timing violations by around 2/3 compared to a commercial pre-routing time estimation tool.

Mozhzhukhina A [10] The use of contemporary artificial intelligence technologies in the field of microcircuit design is discussed in this article. It supports the need for and provides a brief explanation of some artificial intelligence techniques and technologies, such as deep reinforcement learning (DRL) in connection with formalizing the task as a Markov decision process and neural networks in connection with representing LSI as a weighted graph with feature vectors. Also included is a breakdown of the most popular neural network designs. Along with a brief explanation of the steps, a technique for positioning LSI elements at the stage of topological design based on DRL employing graph neural networks (GNN) is also described.

The characteristics in this machine learning model include placement-specific data .and the clogged global route. Y. -Y. Huang et al. [11] use the model to forecast where the precise routing violations will be, input that knowledge back into the placement algorithm, and then provide a new placement result. According to experimental findings, the suggested techniques can significantly reduce the frequency of DRC breaches as compared to the outcome of the original placer.

In order to accommodate objective functions with inference-time changing weights, Fu-Chieh [12] suggests flexible multiple-objective reinforcement learning (MORL) with a single pretrained model. Their macro placement outcomes demonstrate that MORL may efficiently produce the Pareto frontier of many objectives.

P. Esmaeili et al. [13] have demonstrated how RL may be applied to dramatically shorten DP runtimes while retaining Quality-of-Result (QoR). Designers create three different RL models using Actor-Critic, Deep Q-Learning, and Tabular Q-Learning. These models are tested using the 12 ISPD contest benchmarks and evaluated by incorporating them into GPlace3.0, a cutting-edge analytic FPGA placement tool. Their findings reveal that when compared to the algorithmic-based detailed placer in GPlace3.0, the models exhibit total runtime savings between 2x and 3.5x with equivalent QoR.

To forecast congestion hotspots, S. Liu et al. [14] provide a fully convolutional network model in this study. To obtain a more route-friendly result, we next integrate this prediction model into the DREAM Place placement engine. The
experimental results using ISPD2015 benchmarks demonstrate that our suggested technique, when compared to the state-of-the-art, can reduce congestion rate by up to 9.05% and routed wirelength by 5.30% due to the improved accuracy of the prediction model.

Yao Lai et al. [15] have proposed a MaskPlace, an RL-based placement approach that learns position, wirelength, and view information. It facilitates the model's effective and efficient operation without constricting the search space. They create a direct incentive function based on real-world examples and achieve satisfying outcomes for all important parameters. This effort can speed up placement and prevent module overlaps that aren't wanted.

By equating the analytical placement problem to the process of training a neural network, Y. Lin et al. [16] have provided a revolutionary GPU-accelerated placement framework called DREAM Place. DREAM Place, which is built upon the widely used deep learning toolkit PyTorch, can outperform the state-of-the-art multithreaded placer RePlAce in terms of global placement speed without sacrificing quality by about 40%.

By completing a series of trials and utilizing convolutional neural networks to extract spatial data from various locations, Lei Deng et al. [17] a reinforcement-learning-based technique to automatically optimize core placement using deep deterministic policy gradient. The proposed method, according to experimental results, increases throughput by 1.99 and reduces latency by 50.5% when compared to naive sequential placement; when compared to simulated annealing, which is a useful technique for approximating the global optima in a very large search space, our method increases throughput by 1.22 and decreases latency by 18.6%.

A. Mirhoseini et al. [18] their approach can learn from the past and get better over time at producing ideal locations for chip blocks that haven't been seen before, especially as it trains over more chip blocks. The agent is trained to place the nodes of a chip network onto a chip canvas in order to achieve these results. They formulate placement as a reinforcement learning (RL) problem. The supervised task of predicting placement quality serves as the foundation for their reinforcement learning (RL) policy, allowing it to generalize to unseen blocks. They were able to create rich feature embeddings of the input netlists by creating a neural architecture that can precisely predict reward across a wide range of netlists and their placements. Then, in order to facilitate transfer learning, they employ this architecture as the encoder of their policy and value networks. Their goal is to reduce PPA (power, performance, and area), and they demonstrate it in less than six hours.

Gandhi et al. [19] new methods of routing have been provided in the Alpha-Router and Alpha-PD-Router programmers. Without utilizing any external optimization algorithms, these frameworks develop the optimization function autonomously. The reduction of running time and memory usage when routing each benchmark is a benefit of employing the RL-based design. After a single training session with the data generated by the RL algorithm, these routing tools will be able to produce results. Therefore, no outside data is needed. A modest 5X5 grid circuit with three nets has been the subject of experiments to demonstrate the effectiveness of this model as a proof of concept. NNET parameters have been discovered experimentally as the first contribution. The performance of the Tick Tack Toe game settings influenced by AlphaGo Zero and the Alpha-router has been compared. It is demonstrated that parameters found with Alpha-router provide more wins as the difficulty of pin placement increases. With Alpha-PD-second Router's contribution, 99 test cases and 116 DRCs out of 177 DRCs are resolved in 51.6 seconds.

At order to fix the short violation in the routing stage, a routing model named Alpha-PD-Router is provided by U. Gandhi et al. [20]. One of the first RL-based frameworks, Alpha-PD-Router doesn't require users to have any prior understanding of the problem environment or outside information for physical design. The Alpha-PD-Router is built on a two-player cooperative game model that was trained on a tiny circuit and, in the testing phase, successfully resolves 75 violations out of 99 instances of 2 pins net arrangements. They have used straightforward 2-pin net routing issues to show the viability of this self-sufficient collaborative game strategy. Their objective is to extend this game theory-based approach to provide superior results on real routing benchmarks.

A. Goldie et al. [21] have discussed deep reinforcement learning for placement optimization. Deep RL, which permits domain adaptation and direct optimization of non-differentiable objective functions, is a potential strategy for handling combinatorial problems. The difficulty of training RL rules is partly caused by the fragility of gradient updates and the high cost of reward evaluation. In this work, researchers introduce deep RL, describe the placement problem as an RL problem, and present methods for developing RL agents

that perform well. Designers forecast a shift toward more efficient RL-based domain adaptation methods, where graph neural networks will play a significant role in facilitating both improved sample efficiency and more optimum placements.

Andrew B et al. [22] have examined machine learning possibilities with an emphasis on IC physical implementation. Examples of applications are: (1) eliminating irrational design and modelling margins through correlation mechanisms; (2) accelerating design convergence through predictors of downstream flow outcomes that consider both tools and design instances; and (3) corollaries like optimizing the use of design resource licences and available schedule.

III. COMPARATIVE PERFORMANCE OF ML IN VLSI PLACEMENT

Table I here, shows the study of the different Machine learning model used, the type of placement, the algorithm used, comparison with which technology and results of different papers reviewed above.

Table 3.1: Study of implementing ML in Vlsi placement

SL.no	Title	Machine learning model	Type placement	Algorithm used	Compared with	Result
1	[1]	Deep RL	Placement	Graph neural network	Human design, Multi-Armed Bandit	20min to do over all placement, 11% and 2.5% wirelength improvements
2	[2]	RL	placement	RS3DPlace	Random Initialized SA (RandSA).	16% improvement in overall cost function
3	[3]	Deep RL	placement	Proximal Policy Optimization	Manual and RePLACE	Improvement in time,area, power and wirelength
4	[4]	RL	Detailed Placement	Monte Carlo Search Tree	Manual procedure	Routing violations are reduced
5	[5]	RL	Placement	fixed routing algorithm	Mondrian algorithm	improvement in wirelength Term1-44.46% 2large- 22.02%
6	[6]	RL	placement	gradient based optimization technique	DREAMPlace	Effective improvement in timings
7	[7]	RL	placement	BP algorithm	BP algorithm	Time optimization
8	[8]	RL	placement	pre-routing timing prediction approach	commercial pre-routing timing estimation	largely reduce the pessimism and improve the accuracy of pre-routing prediction
9	[9]	Deep RL	placement	Graph neural network	Greedy algorithm	Improvement in timings
10	[10]	RL	Detailed Placement	SVM and Refinement Placement	VDA placer	short violations is reduced
11	[11]	RL	Placement	MOPPO	fixed-preference PPO	best improvement in training
12	[12]	Incremental RL	Placement	policy relaxation and importance weighting	Reacher, Swimmer, Hopper, and HalfCheetah	faster adaptation to various dynamic environments than the baselines.
13	[13]	RL	Detailed Placement	Tabular Q-Learning, Deep Q-Learning, and Actor-Critic.	DOISM	total runtime improvements between 2x to 3.5x and similar QoR
14	[14]	Deep RL	Global placement	fully convolutional network model (a routability-driven placer)	NTUplace4dr, DREAMPlace	9.05% reduction in congestion rate and 5.30% reduction in routed wirelength
15	[15]	RL	Placement	MaskPlace	GraphPlace	it achieves 60%- 90% wirelength reduction and guarantees zero overlaps.
16	[16]	Deep RL	Global placement	DREAMPlace	RePlAce	around 40× speedup

17	[17]	RL	placement	DDPG	Simulated anneling	achieves 1.99× increase in throughput and 50.5% reduction in latency
18	[18]	Deep RL	placement	Policy Network Architecture	Simulated anneling and RePIAce	PPA optimization ,6hours to complete placement
19	[19]	RL	Placement	MCTS	AlphaGo Zero	116 DRC's out of 177 drc's are resolved in 99 test cases in 51.6 secs.
20	[20]	RL	placement	Alpha-PD-Router	A-star algorithm	75 violations in 99 cases of 2 pins net arrangements in the testing phase. Took 3hrs
21	[21]	Deep RL	Placement	Policy gradient optimization	GNN	Efficient placement

IV. Conclusion

This paper mainly focuses on different machine learning methods for VLSI placement. A detailed review of how RL has been implemented VLSI placement is shown. It comprehends many algorithms which helps it better placements where the machine learns on its own without much human intervention. Also, the major goal in physical design placement is to add RL algorithms in order to increase its efficiency but to accomplish that, there is also a requirement. This paper can serve as a reference for future research purposes regarding RL implementation in the VLSI placement.

.

References

[1] Z. Wang, H. -X. Li and C. Chen, "Incremental Reinforcement Learning in Continuous Spaces via Policy Relaxation and Importance Weighting," in IEEE Transactions on Neural Networks and Learning Systems, vol. 31, no. 6, pp. 1870-1883, June 2020

[2] A. Agnesina, K. Chang and S. K. Lim, "VLSI Placement Parameter Optimization using Deep Reinforcement Learning," 2020 IEEE/ACM International Conference on Computer Aided Design (ICCAD), 2020, pp. 1-9.

[3] A. Mansoor and M. Chrzanowska-Jeske, "RS3DPlace: Monolithic 3D IC placement using Reinforcement Learning and Simulated Annealing," 2022 IEEE International Symposium on Circuits and Systems (ISCAS), Austin, TX, USA, 2022, pp. 394-398

[4] Mrinal Mathur,"Routing and Placement of Macros using Deep Reinforcement Learning"

[5] S. F. Almeida, J. Luís Güntzel, L. Behjat and C. Meinhardt, "Routability-Driven Detailed Placement Using Reinforcement Learning," 2022 IFIP/IEEE 30th International Conference on Very Large-Scale Integration (VLSI-SoC), Patras, Greece, 2022, pp. 1-2

[6] R. Manimegalai, E. Siva Soumya, V. Muralidharan, B. Ravindran, V. Kamakoti and D. Bhatia, "Placement and routing for 3D-FPGAs using reinforcement learning and support vector machines," 18th International Conference on VLSI Design held jointly with 4th International Conference on Embedded Systems Design, Kolkata, India, 2005, pp. 451-456,

[7] R. Cheng and J. Yan, "On Joint Learning for Solving Placement and Routing in Chip Design." arXiv, 2021.

[8] L. Bai and L. Chen, "Machine-Learning-Based Early-Stage Timing Prediction in SoC Physical Design," 2018 14th IEEE International Conference on Solid-State and Integrated Circuit Technology (ICSICT), Qingdao, China, 2018, pp. 1-3,

[9] E. C. Barboza, N. Shukla, Y. Chen and J. Hu, "Machine Learning-Based Pre-Routing Timing Prediction with Reduced Pessimism," 2019 56th ACM/IEEE Design Automation Conference (DAC), Las Vegas, NV, USA, 2019, pp. 1-6.

[10] Mozhzhukhina A. "VLSI element placement method based on deep reinforcement learning" International Journal of Professional Science №2-2022

[11] Y. -Y. Huang, C. -T. Lin, W. -L. Liang and H. -M. Chen, "Learning Based Placement Refinement to Reduce DRC Short Violations," 2021 International Symposium on VLSI Design, Automation and Test (VLSI-DAT), Hsinchu, Taiwan, 2021, pp. 1-4

[12] Fu-Chieh Chang "Late Breaking Results: Flexible Chip Placement via Reinforcement Learning" DAC '22: 59th ACM/IEEE Design Automation Conference ,2022

[13] P. Esmaeili, T. Martin, S. Areibi and G. Grewal, "Guiding FPGA Detailed Placement via Reinforcement Learning," 2022 IFIP/IEEE 30th International Conference on Very Large-Scale Integration (VLSI-SoC), Patras, Greece, 2022, pp. 1-6,

[14] S. Liu, Q. Sun, P. Liao, Y. Lin and B. Yu, "Global Placement with Deep Learning-Enabled Explicit Routability Optimization," 2021 Design, Automation & Test in Europe Conference & Exhibition (DATE), Grenoble, France, 2021, pp. 1821-1824,

[15] Yao Lai, Yao Mu, Ping Luo "MaskPlace: Fast Chip Placement via Reinforced Visual Representation Learning" Computer Vision and Pattern Recognition,2022

[16] 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," 2019 56th ACM/IEEE Design Automation Conference (DAC), Las Vegas, NV, USA, 2019, pp. 1-6.

[17] Wu, Lei Deng, Guoqi Li, and Yuan Xie. 2020. Core Placement Optimization for Multi-chip Many-core Neural Network Systems with Reinforcement Learning. ACM Trans. Des. Autom. Electron. Syst. 26, 2, Article 11 (March 2021)

[18] A. Mirhoseini et al., "Chip Placement with Deep Reinforcement Learning." arXiv, 2020

[19] Gandhi, "A Reinforcement Learning-Based Framework to Generate Routing Solutions and Correct Violations in VLSI Physical Design," M.S. Thesis, Sch. of Electrical and Computer Engineering, University of Calgary., Calgary, Ontario, Alberta, 2020.

[20] U. Gandhi, I. Bustany, W. Swartz, and L. Behjat, "A Reinforcement LearningBased Framework for Solving Physical Design Routing Problem in the Absence of Large Test Sets," 2019 ACM/IEEE 1st Workshop on Machine Learning for CAD (MLCAD). IEEE, Sep. 2019

[21] A. Goldie and A. Mirhoseini, "Placement Optimization with Deep Reinforcement Learning," Proceedings of the 2020 International Symposium on Physical Design. ACM, Mar. 20, 2020

[22] Andrew B. Kahng. 2018. Machine Learning Applications in Physical Design: Recent Results and Directions. In Proceedings of the 2018 International Symposium on Physical Design (ISPD '18). Association for Computing Machinery, New York, NY, USA, 68–73