2024 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)

Global Routing Optimization Analysis based on Deep Reinforcement Learning

Ying-Tzu Chen, Wei-Kai Cheng
Department of Information and Computer Engineering
Chung Yuan Christian University
Taoyuan City, Taiwan
g11177002@cycu.edu.tw, wkcheng@cycu.edu.tw

2024 IEEE Asia Pacific Conference on Circuits and Systems (APCCAS) | 979-8-3503-7877-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/APCCAS62602.2024.10808325

***Abstract*—Integrated Circuit (IC) design is a multi-stage and complex process, with each stage from requirements analysis to final manufacturing presenting its unique challenges. As technology and market demands evolve, the complexity and difficulty of IC design continue to increase, necessitating more robust automated design systems to assist engineers in overcoming these challenges. This paper focuses on global routing, a critical step in IC design responsible for determining the approximate paths of signal lines on a chip to optimize resource allocation, signal integrity, and power control. Global routing has always been a challenging aspect of IC physical design. Common solutions primarily rely on heuristic-driven greedy methods, which have strict limitations. We aim to find better solutions to minimize total wire length (WL) and edge overflow (OF). This paper references an IC global routing method based on Deep Reinforcement Learning (DRL) and improves the traditional Deep Q-Network (DQN) algorithm to a Double Deep Q-Network (DDQN) to address issues in IC global routing. The method aims to use DDQN to enhance the stability and performance of DQN in handling complex routing problems. Experimental results compare DDQN with the commonly used A* search algorithm, demonstrating the optimized performance of DDQN in global routing tasks, especially in partially edge-exhausted problems, where DDQN excels and successfully optimizes total wire length.**

Keywords—Global Routing, Deep Reinforcement Learning, Machine Learning

I. INTRODUCTION

Integrated Circuit (IC) design is a multi-stage and complex process that involves many technical challenges and issues. From requirement analysis to final manufacturing, each stage has its unique challenges. With continuous technological advancements and changing market demands, the complexity and difficulty of IC design are constantly increasing. Designers need to consider multiple factors comprehensively and use advanced design methods and tools to ensure the success of the design. We need more powerful automated design systems to help engineers address the increasingly challenging IC design problems.

The routing process in IC design refers to determining the connections between various components in the circuit. This process can be divided into two stages: global routing and detailed routing. Global routing focuses on macro-level planning, determining the approximate paths and directions of the main signal lines, optimizing the allocation of routing resources, and avoiding routing congestion[1]. Detailed routing focuses on micro-level implementation, determining the specific paths of each signal line, making fine adjustments and verifications, and ensuring that all connections meet design specifications and manufacturing requirements. This paper will focus on the aspect of global routing.

From past to present, global routing has always been a challenging problem in IC physical design. In practical operations, it may be necessary to integrate a very large number of components and networks onto a single chip, making this problem extremely challenging. Due to the difficulty of the global routing problem and the increasing demand for more complex designs, there has been extensive research and exploration in global routing in IC design. However, current solutions mainly rely on heuristic-driven greedy methods, which primarily address situations with strict constraints on the problems to be solved, such as sequential network routing after network sorting [2]. We aim to find better solutions that can minimize objective functions of interest, such as total wire length (WL) and edge overflow (OF).

This paper refers to an IC global routing method based on Deep Reinforcement Learning (DRL) [3]. We modify the algorithm by replacing the deep Q-network (DQN) algorithm with a double deep Q-network (DDQN). We analyze its conjoint optimization mechanism and effectively address the issues encountered in IC global routing. Finally, we compare the results of different algorithms. Deep Reinforcement Learning (DRL) is a machine learning technique that combines deep learning and reinforcement learning. In today's technological advancements, DRL has become an important direction in artificial intelligence research because it can learn and perform complex tasks in high-dimensional and dynamic environments. This technology is widely used in areas such as gaming, robot control, and autonomous driving[4,5]. We chose Double Deep Q-Network (DDQN) as the base RL algorithm because DDQN can optimize some of the issues generated by the Deep Q-Network (DQN). At the end of the experiments, we will compare the solutions obtained by DDQN with those obtained by the commonly used A* search algorithm. Based on several evaluation criteria, it can be observed that DDQN offers better optimization than A* search.

II. BACKGROUND

A. Global Routing

Global routing is a critical step in the integrated circuit (IC) design process. It is responsible for determining the approximate paths of signal lines on the chip, ensuring that the connections between circuit modules meet design requirements. The main goals of global routing include resource allocation, signal integrity, timing optimization, and

This work was supported in part by National Science and Technology Council, Taiwan. (Grant number NSTC 113-2221-E-033-030)

979-8-3503-7877-1/24/$31.00 ©2024 IEEE

power consumption control. Common global routing algorithms include the A* algorithm, Dijkstra's algorithm, Rip-up and Reroute, minimum spanning tree-based algorithms, and greedy algorithms[6]. In this section, we will explain the algorithms used in this paper.

The A* algorithm is based on heuristic search, using a heuristic function to estimate the minimum cost from the current node to the target node. It can be used to find the shortest path from the starting point to the target pin. The greedy algorithm plans the path for each signal line in a certain order, choosing the currently optimal path. It quickly generates an initial routing solution but requires subsequent adjustments. The minimum spanning tree algorithm is used to find a tree that connects all points with the minimum total edge weight. It first constructs a minimum spanning tree and then refines it into specific routing paths.

In the global routing process, capacity refers to the maximum number of wires allowed to pass through a specific area without exceeding design and manufacturing constraints, thus preventing congestion or design rule violations. If the number of wires passing through a certain edge exceeds its capacity, this excess is called overflow. Overflow can lead to increased signal delay, higher power consumption, and signal integrity issues. The goal of global routing is to reduce congestion, meaning the total overflow, and to minimize the total length of all wires in the routing process, which includes the physical length of all signal lines connecting different components or modules.

In most single networks, there are multiple pins that need to be connected. We can use methods such as minimum spanning tree or rectilinear Steiner tree to decompose the multi-pin network problem into a set of two-pin connection problems [7,8]. Then, we use the A* search algorithm to solve the two-pin connection problems. Finally, we merge these solutions into a solution for the single network.

B. Deep Neural Network

Deep Neural Networks (DNNs) are a type of artificial neural network characterized by having multiple hidden layers, allowing them to automatically extract high-level features from data. DNNs are the core technology of deep learning and are widely applied in fields such as image recognition, speech recognition, and natural language processing. The structure of a DNN includes the input layer, which is responsible for receiving external data, with the number of nodes determined by the dimensionality of the input data. The hidden layers contain multiple neurons (nodes), with each neuron performing nonlinear transformations through activation functions. The number and size of these hidden layers determine the depth and capacity of the network. Finally, the output layer is used to produce the final results, with the number of nodes and activation functions varying depending on the specific task. However, despite their strong representational capabilities and the advantage of automatic feature extraction, DNNs also face challenges such as high data requirements, significant computational resources, and the risk of overfitting. Through reasonable model design and optimization methods, the full potential of DNNs can be harnessed to solve various complex data analysis problems.

In Deep Reinforcement Learning (DRL), DNNs play a crucial role in approximating value functions, policy functions, or model dynamics. Through DNNs, DRL can automatically extract features and handle high-dimensional complex data, enabling a range of applications from image recognition to robotic control. However, training DNNs in DRL also faces challenges such as stability, balancing exploration and exploitation, and computational resource demands. Therefore, certain techniques are needed to improve stability and efficiency, such as target networks and experience replay.

C. Deep Reinforcement Learning

Deep Reinforcement Learning (DRL) is an artificial intelligence approach that combines the techniques of Deep Learning and Reinforcement Learning. DRL leverages DNNs to process and learn from complex, high-dimensional data, and uses Reinforcement Learning to make decisions and optimize strategies. This method has achieved remarkable success in various fields, including gaming, robotic control, and autonomous driving. In the global routing process, capacity refers to the maximum number of wires allowed to pass through a specific area without exceeding design and manufacturing constraints, thus preventing congestion or design rule violations. If the number of wires passing through a certain edge exceeds its capacity, this excess is called overflow. Overflow can lead to increased signal delay, higher power consumption, and signal integrity issues. The goal of global routing is to reduce congestion, meaning the total overflow, and to minimize the total length of all wires in the routing process, which includes the physical length of all signal lines connecting different components or modules.

In DRL, as illustrated in Fig. 1, the agent is the entity that learns and makes decisions. It takes actions in the environment and receives feedback from the environment. The current state (s) of the environment is the situation the agent is in at a certain moment. The agent decides the next action to take based on this state. Whenever the agent performs an action, the environment provides a reward (r), which measures the quality of the action. The core goal of DRL is to learn a policy (π), which dictates the actions to take in each state.

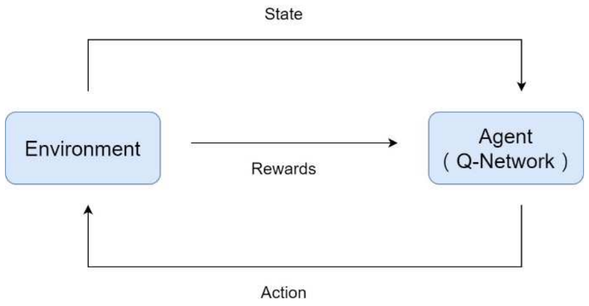


Fig. 1. Diagram of DRL Architecture.

To evaluate the quality of the policy, DRL uses the value function (V) and the action value function (Q). This part is integrated with the DNN architecture, often referred to as the Q-network.[9] The value function estimates the long-term return of being in a certain state, while the action value function estimates the long-term return of taking a specific action in a certain state. The evaluation of these functions is usually achieved through trial and error and updating.

III. Methodology

Fig. 2 illustrates the global routing method based on DDQN. First, a set of data with specific size, complexity, and constraints needs to be generated. We refer to a dataset generation method [2]. These generated data are stored in individual text files. Each file records relevant information, such as the capacity of edges in the grid, the dimensions of the grid, the networks that need routing, and the x, y, z coordinates of each network's pins. After generating these files, they are read and parsed. Each network is then decomposed into a set of two-pin problems using the minimum spanning tree algorithm. Subsequently, A* algorithm routers and DDQN algorithm routers are used to solve these numerous two-pin problems. It is important to note that the results from the A* algorithm router are provided as burn-in memory for the DDQN algorithm router, 像预训练 which facilitates a faster convergence process for DDQN. After solving these two-pin problems, the solutions for each network are merged. The final merged results provide the complete network solution, which is then analyzed to evaluate its quality.

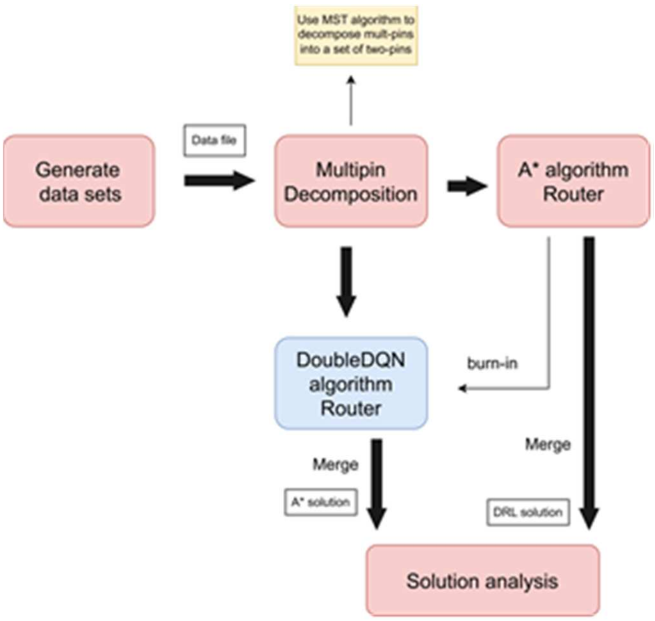


Fig. 2. Workflow for Solving Global Routing Using DDQN.

When decomposing all the two-pin problems, two Deep Q-Networks are utilized to solve all the two-pin problems during the iterative learning process. The framework of DDQN is shown in Fig. 3. The Q-network acts as an agent interacting with the environment. The environment provides state information to the Q-network. Next, the agent evaluates the Q-values of all possible next states. Finally, based on the ϵ-greedy algorithm, an action is selected and executed, thereby changing the environment. The agent then takes a "step" in the environment. The reward is calculated based on the new state, and the edge capacity information is updated. Each transition during the training process is recorded in the replay buffer. These transitions are used for backpropagation whenever the weights in the Q-network are updated iteratively.

The DDQN algorithm is an improved version of the DQN algorithm [5]. This method is an enhancement to traditional Q-learning and DQN, aimed at reducing the overestimation of Q-values, thereby improving the algorithm's performance. Both DDQN and DQN are algorithms used to solve reinforcement learning problems, differing only in the way the Q-values are computed within the Agent. They have some key differences in the way Q-values are updated. This paper focuses on the DDQN algorithm. DDQN uses two deep neural networks, a Main Q-Network and a Target Q-Network. These two networks have the same structure but different weights. The Main Q-Network is used to select actions, while the Target Q-Network is used to evaluate the target Q-values of these actions. This separation is mainly to reduce the overestimation problem caused by using a single Q-Network for Q-value updates in DQN, thus improving learning stability and effectiveness.

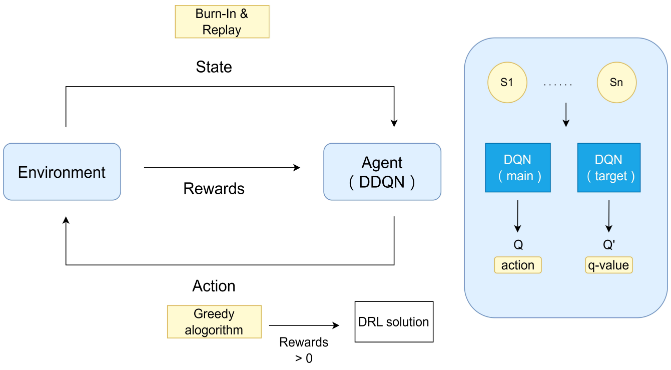


Fig. 3. Workflow of the Router based on DDQN.

As in (1), Q' is the Q-value of the Target Q-Network, used to calculate the target value. Q is the Q-value of the Main Q-Network, which can be considered the Q-value of the current state (s) and action (a). $\arg\max_{a'} Q(s', a')$ is the best action for the next state selected by the Main Q-Network. α is the learning rate, r is the immediate reward obtained after executing action (a), and γ is the discount factor used to account for the influence of future rewards. Intuitively, if the Q-value is overestimated, it means the action chosen by Q is overestimated, but as long as Q' does not overestimate the action chosen by Q, it is fine. Even if Q' overestimates a particular action, as long as Q does not select that action, there is no problem. The two networks balance each other.

$$Q(s,a) \leftarrow Q(s,a)+\alpha[r+\gamma Q'(s',\arg\max_{a'} Q(s',a'))-Q(s,a)] \quad (1)$$

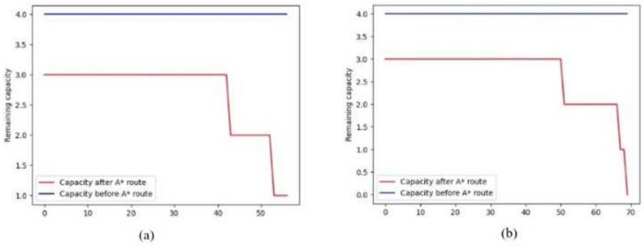
IV. Experiment

The main advantage of using Deep Reinforcement Learning (DRL) with double Q-networks for routing tasks lies in the model's ability to conjointly consider sub-tasks, such as the networks and pins to be routed. This optimization is a distinctive feature of the DDQN model. Considering that the effective utilization of edges in A* routing is a critical aspect, we categorize the problems into two types in this study. The first type is called type I problems or non-edge-exhaustion problems, involving situations where edges with positive capacity are not fully utilized according to A* routing. The second type is called type II problems or partial edge-exhaustion problems, encompassing situations where at least some edges with positive capacity are fully utilized by the A* routing algorithm. This categorization helps in understanding the different challenges posed by each type of problem and highlights the areas where DDQN can offer significant improvements.

We conducted experiments using the parameter settings in Table I. Fig. 4 shows the edge utilization graphs for the two types of problems based on A* solutions. Fig. 4(a) represents type I problems, and Fig. 4(b) represents type II problems before and after A*. It can be observed that type I problems require fewer networks to be routed compared to type II problems. This indicates that type I problems do not fully utilize any edges after A*, while type II problems do fully utilize some edges after A*.

Table I. Parameter Settings of the DDQN Model

Parameter	value	Parameter	value
Learning rate	1×10^{-4}	Batch size	32
Buffer size	50,000	Burn-in size	10,000
γ (discount factor)	1.0	Burn-in memory	A*-based
ϵ-greedy	0.05	Max episodes	5,000



(a)

(b)

Fig. 4. Two Types of Edge Utilization Graphs Based on A* Solutions.

Next, we conducted experiments on type II problems using the same parameter settings. Table II shows the results for different map sizes. Notably, the results indicate that in most cases, the overflow (OF) remains at zero, while the wire length (WL) shows significant optimization. Therefore, this demonstrates that the DDQN router generally provides superior solutions compared to the A* router when dealing with type II (partial edge-exhaustion) problems. This superior performance is evident in the consistent reduction of wire length, highlighting the effectiveness of the DDQN approach in optimizing complex routing challenges in these specific scenarios.

Table II. Results for Type II Problems on Different Map Sizes

Router	size	OF	WL
A*	8×8×2	0	347
DDQN	8×8×2	0	329
A*	16×16×2	0	1608
DDQN	16×16×2	0	1597

V. Conclusion

This paper explores an IC global routing method based on deep reinforcement learning (DRL), specifically improving the traditional deep Q-network (DQN) algorithm to a double deep Q-network (DDQN). Experimental results show that DDQN provides more optimized solutions for complex routing problems compared to the A* algorithm, especially in addressing partial edge exhaustion (type II) issues, where DDQN performs excellently. DDQN not only enhances the stability and performance of the model but also successfully optimizes total wire length (WL) in most cases. This indicates that DDQN is an effective IC global routing solution that better addresses the challenges of modern IC design. Furthermore, the use of DDQN demonstrates its potential in handling high-dimensional and complex problems, particularly in large-scale IC design. The methods adopted in this paper are not limited to the IC design field; their core ideas and techniques can also be applied to other engineering fields that require global routing optimization, such as network design and robot path planning. Future research can further explore other DRL techniques and optimization strategies to enhance the automation and efficiency of the IC design process. For example, integrating multiple reinforcement learning techniques or introducing more environmental features to enhance the model's learning capability could be considered. Additionally, expanding to three-dimensional routing problems and dynamic design requirements will also be worthwhile directions to explore.

Overall, the DRL method based on DDQN shows its powerful advantages in IC global routing, providing a feasible and efficient solution for complex system design. This will bring new perspectives and opportunities to the IC design field and help advance the further development of automated design technologies.

References

[1] Yuxuan Pan, Zhonghua Zhou, André Ivanov, “Routability-driven Global Routing with 3D Congestion Estimation Using a Customized Neural Network,” 2022 23rd International Symposium on Quality Electronic Design.

[2] Jiang Hu, and Sachin S. Sapatnekar, S. S., “A Survey on Multi-Net Global Routing for Integrated Circuits,”Integration, 31(1) , 2001 ,pp. 1–49.

[3] Haiguang Liao, Wentai Zhang, Xuliang Dong, Barnabas Poczos, Kenji Shimada, Levent Burak Kara,” A Deep Reinforcement Learning Approach for Global Routing,” J. Mech. Des. Jun 2020.

[4] Huaxin Pei, Jiawei Zhang, Yi Zhang, Huile Xu, Li Li, “Self-Organized Routing for Autonomous Vehicles via Deep Reinforcement Learning,” IEEE Transactions on Vehicular Technology, 73(1), January 2024.

[5] Wei Jin, Qifeng Xie, Wenkang Wan, Yoshiki Yamaguchi, Dejene M. Sime, Kai Sheng, “A Comparative Study on DRL-Based Autonomous Driving Under Single-Vehicle and Human-Vehicleroad Coordination,” IEEE 2023 2nd ICARCE, December 2023.

[6] Geetanjali Udgirkar, G Indumathi, “VLSI global routing algorithms: A survey,” IEEE INDIACom., 31 October 2016.

[7] M. Cho, K. Lu, K. Yuan and D. Pan, “Boxrouter 2.0: architecture and implementation of a hybrid and robust global router,” Computer-Aided Design 2007. ICCAD 2007. IEEE/ACM International Conference on.

[8] Ho, J.-M., Vijayan, G., and Wong, C.-K., “New Algorithms for the Rectilinear Steiner Tree Problem,” IEEE Trans. Comput. Aided Des. Integr. Circuits Syst., 9(2), February 1990.

[9] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M., 2013, “Playing Atari With Deep Reinforcement Learning,” preprint arXiv:1312.5602