Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Review:

A review of automatic schematic generation techniques and their application to PCBs*

Jie YANG, Kai QIAO, Jian CHEN, Chen CHEN, Lixiang GUO, Bin YAN‡

PLA Information Engineering University, Zhengzhou 450000, China

E-mail: evil126126@126.com; qiaokai1992@gmail.com; ybspace@hotmail.com

Received July 27, 2024; Revision accepted Mar. 25, 2025; Crosschecked

Abstract: The Printed Circuit Board (PCB) stands as the cornerstone of electronic equipment, with its schematic holding paramount importance for system performance and reliability. In light of the pervasive use of electronic devices in society, concerns regarding maintenance, safety, backdoors, and other latent issues have garnered significant attention. Automatic schematic generation (ASG), with its distinct capability for generating circuit schematics autonomously, not only plays a pivotal role in Electronic Design Automation (EDA) but also aids in deciphering the fundamental principles of PCB equipment to effectively address these underlying issues. However, constrained by the increasingly sophisticated manufacturing processes of PCBs and the inherent legal and ethical controversies surrounding reverse engineering, the development of related technologies faces notable bottlenecks. To break through technical barriers and advance technological progress, this paper comprehensively combs through the existing ASG, in-depth description of the core algorithms of the technology – layout and routing, and for the application of the technology in PCB reverse engineering, analyzes in detail the current challenges and problems faced. Around these challenges, feasible solutions are discussed in this paper, with the aim of promoting the research of automatic PCB schematic generation technology and contributing new strength to EDA and PCB reverse engineering automation.

Unedited

Key words: Automatic schematic generation; Layout; Routing; Printed circuit board; Reverse engineering; Automation

https://doi.org/10.1631/FITEE.2400612 **CLC number:**

1 Introduction

The rapid advancement of electronic information technology has rendered electronic devices an essential infrastructure in contemporary society, which greatly promotes the facilitation of life and work. In this development process, the printed circuit board (PCB) assumes a pivotal role. As the "nervous system" of electronic devices, it is the carrier of the circuits, providing a foundation for the realization of device functions.

However, as electronic technology progresses and equipment functions become more complex, PCBs face new challenges such as potential hardware Trojan insertions, intellectual property rights verification, and the maintenance of older equipment (Rematska and Bourbakis, 2016; Botero et al., 2021). Therefore, conducting reverse engineering research on PCBs to accurately reconstruct their original circuit diagrams has become a key strategy to tackle these challenges. Through reverse engineering, designers can analyze circuit structures, optimize design schemes, and ultimately enhance the performance and safety of electronic devices.

In PCB reverse engineering, schematic generation stands out as a critical phase. Serving as a visual representation of the circuit, the schematic illustrates the interconnections among electronic components. It not only forms the foundation for un-

‡ Corresponding author

* Project supported by the fund of Laboratory for Advanced Computing and Intelligence Engineering, China

ORCID: Jie YANG, https://orcid.org/0000-0001-7933-8220

© Zhejiang University Press 2025

derstanding and analyzing circuits but also serves as a valuable guide for circuit design and optimization (Yang et al., 2024). However, the complexity and time-consuming nature of this process should not be underestimated, especially when dealing with large-scale, high-density PCBs. Manual generation of schematic diagrams becomes notably challenging in such scenarios. Therefore, the study of automatic schematic generation (ASG) is not only an important prerequisite to promote the automation of PCB reverse engineering but also an important technical demand to enhance the potential safety of electronic devices.

Despite the strides made in research on ASG over the past decades, the extensive adoption of this technology in the realm of PCBs is hindered by the complex and irregular nature of PCB schematic structures, leading to various challenges. To tackle these challenges and promote the extensive adoption of ASG in PCBs, this paper provides a review of the current research situation, fundamental procedures, and prevalent algorithms of ASG. Building upon this foundation, the paper further discusses the specific difficulties of ASG in PCB reverse engineering and analyzes the possible solutions. The purpose of this research is to promote the further development of ASG in PCB reverse engineering and provide technical support for the application expansion and security of PCB electronic products.

In this paper, Section 1 provides a brief overview of automatic schematic technology, including its basic concepts, current situation, and technical process. Sections 2 and 3 introduce the core algorithms of ASG, i.e., layout and routing technologies, respectively, and comb them. Section 4 discusses the difficulties and solutions of ASG in PCB reverse engineering, aiming to advance the development of the field. Finally, Section 5 summarizes the entire text.

2 Overview of ASG

The technology of ASG holds a significant position in the realm of PCB reverse engineering. This section first introduces the basic concepts of ASG, including traditional forms of schematic generation and basic methods of ASG. Subsequently, the current situation of ASG is briefly analyzed, and the main classifications of ASG are summarized from the perspective of different circuit forms. Finally, the fundamental process of ASG is briefly introduced.

2.1 Basic concepts

Schematic diagrams, as a graphical carrier describing the connection relationship between electronic components, are the cornerstone for in-depth understanding and analysis of circuits (Stok and Koster, 1989). Traditionally, the schematic generation process has been predominantly manual, relying on the designer's expertise and meticulous manual interventions. Leveraging advanced electronic design automation (EDA) tools such as Altium Designer and Integrated Synthesis Environment (ISE) for inputting netlists and component details, a preliminary graphical depiction of components is crafted, followed by a step-by-step manual refinement of circuit layouts and component interconnections. This process is cumbersome and time-consuming, ranging from a few weeks to a few months, making it challenging to adapt to the increasing complexity and timeliness requirements of modern electronic devices.

Therefore, the research and implementation of ASG assume paramount importance. This technology endeavors to enhance the efficiency and precision of schematic generation without human intervention, employing automated layout and routing algorithms to construct circuit configurations aligning with design requisites and ensuring precise component interconnections (Jehng et al., 1991). This approach not only significantly reduces human involvement in the schematic generation cycle, thereby reducing time and labor costs, but also paves the way for large-scale and high-density reverse schematic generation. Such advancements hold profound implications in fortifying the security of electronic devices.

2.2 Situation

ASG is currently at a critical turning point. Although the technology has been accumulated and applied, its development still faces some core problems. Its core algorithms have made very few breakthroughs in the past decades, which is slow compared to the overall the pace of innovation in electronics technology. With the rapid innovation of electronic technology, the inherent ASG is still limited to the generation of simple circuits, and in practical applications, especially for complex and high-density circuits, the outcomes are often meager or

Unedited

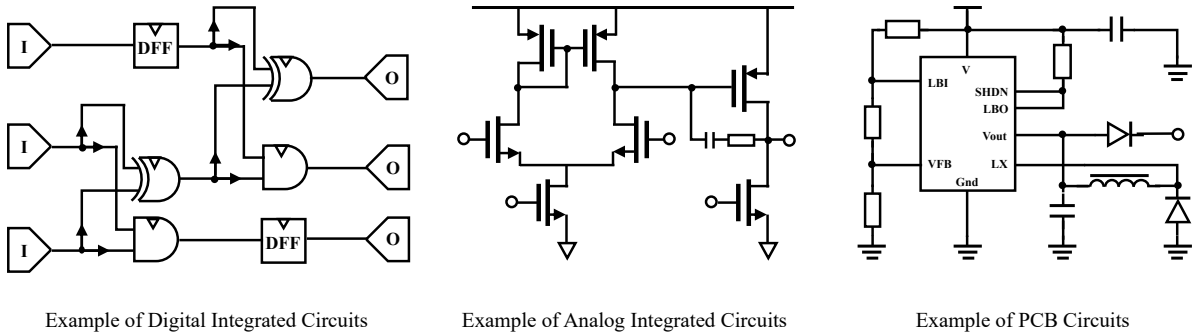

Fig. 1 Examples of different circuit types

even unattainable.

Reverse engineering, especially PCB reverse engineering, involves in-depth analysis of existing products to obtain design and process information, which is prone to intellectual property and ethical disputes (Sharma et al., 2022). Automated schematic generation is an important part of reverse engineering. Concerns about the legality and ethics of reverse engineering in both academia and industry have led to some limitations in the development of ASG. Moreover, the swift evolution of electronic devices has heightened the complexity of relevant technologies, thereby exacerbating the challenges associated with research in this area. Consequently, due to a combination of internal and external factors, the innovation of ASG technology has been slow, which has hindered its broader application in the electronic design processes.

Despite numerous challenges, the significance of ASG cannot be overlooked, and its importance still merits acknowledgment. As a crucial method for netlist visualization, this technology significantly contributes to EDA (Lageweg, 1998). It serves as an essential tool for various tasks, including hardware security protection (Tehranipoor and Koushanfar, 2010) and the maintenance of legacy equipment (Mata et al., 2006). The evolution of ASG is intricately tied to the advancements in EDA throughout history. Since the 1970s, ASG has found extensive applications in electronic design, primarily focusing on the automation of layout and wiring tasks. In recent years, rapid advancements in deep learning technology have significantly enhanced ASG performance and broadened its application scope, solidifying its integral role in contemporary EDA tools. An illustrative example is Synopsys's DSO.ai tool, which achieves automated optimization of chip layout and routing design through the integration of a reinforcement learning engine. Similarly, Cadence's Innovus system utilizes the cutting-edge GigaPlace engine to enable timing slack-driven layout optimization.

Overall, according to different circuit types, the technology can be categorized into two groups: ASG in integrated circuits and ASG in PCBs (Botero et al., 2021). Integrated circuits typically consist of numerous general-purpose circuits integrated into a small chip to achieve a specific chip function. Based on the functional structure of the chip, they can be further classified into analog integrated circuits (AICs), digital integrated circuits (DICs), and other forms. In contrast, PCBs, in circuit form, are more macroscopic than integrated circuits. They serve as the carriers of integrated circuits, usually with the chip as the core, to construct complex peripheral circuits and thereby realize the product's functions. Fig.1 illustrates typical styles of DICs (Fu et al., 2020), AICs (Wang et al., 2020), and PCB circuits (Yang et al., 2024). The structure of DICs typically follows a layer-by-layer arrangement, where component symbols are placed based on the sequential order of rows and columns from left to right. AICs, on the other hand, demonstrate a mirror-symmetric configuration, where component symbols are placed based on the signal flow direction from top to bottom. The structure of PCB circuits exhibits a circular arrangement focused on the chip, adhering to the sequential placement from left to right and top to bottom, thereby forming a circular peripheral circuit. Table 1 summarizes the structural characteristics of different circuit types.

The advent of deep reinforcement learning (DRL) technology has led to its widespread appli-

Unedited

Table 1 ASG techniques for different circuit types

	Circuit type	Structural characteristics
Automatic	DIC	Progressive
Schematic	AIC	Symmetrical
Generation	PCB	Circular flared

cation across various fields, including games (Mnih et al., 2013), biology (Jumper et al., 2021), and chip design (Goldie and Mirhoseini, 2020). This technology has demonstrated remarkable efficacy in decision-making and layout optimization tasks, thereby presenting new opportunities for advancing ASG. DRL possesses the capability to autonomously learn and optimize complex tasks by simulating human decision-making processes. It effectively addresses intricate circuit layout and wiring challenges encountered during schematic generation, offering the potential to overcome existing technical bottlenecks. Moreover, chip layout design and ASG exhibit a high degree of compatibility, as both domains emphasize component placement and routing optimization. In recent years, significant breakthroughs (Liao et al., 2020; Cheng and Yan, 2021; Roy et al., 2021) in chip layout design have been achieved through the application of DRL, and these advancements serve as a vital reference for research in ASG.

Unedited

2.3 Process

The ASG takes a circuit netlist as an input and outputs a circuit schematic that maintains consistent electrical relationships with the netlist. The netlist, a detailed circuit description file, specifies the components in the circuit and their wire network connections. Displayed graphically, the output circuit schematic illustrates the circuit's structure and connections, facilitating circuit analysis. The technical process, depicted in Fig.2, involves netlist parsing, component layout, routing, and circuit labeling. Layout and routing, as the core processes, are crucial. Initially, the input circuit netlist is read, and the component information and connection relationships are analyzed, forming a hypergraph $H = (U, V)$, which is used as the input for subsequent layout and routing. Among them, $U = \{u_1, u_2, u_3, ..., u_n\}$ denotes the set of components and $N = \{n_1, n_2, n_3, ..., n_m\}$ denotes the set of signal networks.

ASG will then layout any $u_i \in U$. This process is carried out on an empty canvas, with the typical goal of positioning all components in the set on the canvas with minimal placement cost, resulting in clear, easy-to-view layout outcomes. The layout process is commonly divided into global layout and local layout, also known as logical layout and geometric layout (Jehng et al., 1991; Nam-Hoon Kim et al., 2000). Global layout serves to establish the abstract component locations and offers macro placement guidance, while local layout specifies the actual geometric coordinates of the components on the canvas and addresses any potential rule violations during the process.

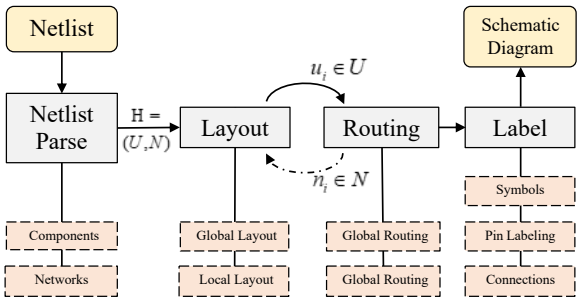

Fig. 2 ASG process. ASG, automatic schematic generation

Routing is generally performed after layout. It refers to the process of interconnecting circuit elements on the canvas based on the circuit connection relationships outlined in the netlist. The signaling network $n_i \in N$ contains information about the endpoints (pins) with connection relationships to direct the routing process. Routing is typically categorized into global and local routing. Global routing serves to establish a preliminary routing of the network, while local routing refines the initial routing from global routing to resolve inter-wire congestion and overlap, enhancing the schematic's readability.

Upon finishing the layout and routing, the automatic generation method will add suitable symbols and annotations to the circuit schematic diagram based on the type and attributes of the circuit components. This typically includes component symbols, pin labeling, connection point labeling, and more, all aimed at aiding designers in comprehending and analyzing the circuit.

3 Layout algorithms

The layout algorithm, responsible for generating the initial layout of the circuit schematic, is a key step in the ASG. This algorithm sequentially places

the components in the circuit netlist on the flat canvas based on schematic design norms. The process entails thorough consideration of component connectivity, layout density, routing length, and other factors to ensure that the initial layout meets design constraints while offering enough routing paths. Fundamentally, the layout algorithm tackles a combinatorial optimization problem. In recent decades, research on layout algorithms can be mainly classified into heuristic-based algorithms and DRL-based algorithms. This section presents an overview of layout algorithms in ASG focusing on these two aspects.

3.1 Heuristic-based layout algorithm

Abstracting the schematic layout process as a mathematical problem, heuristic-based layout algorithms model and design solutions based on specific rules or natural principles to achieve a satisfactory outcome. In the realm of ASG, classical heuristic-based layout algorithms encompass the barycenter algorithm (Swinkels and Hafer, 1990), the bubble algorithm based on value propagation (Jehng et al., 1991), and the simulated annealing algorithm (Steinbrunn et al., 1997).

Swinkels and Hafer argued that the generation of DICs that meet visualization and specification requirements is an NP-hard optimization problem, and typically, one can only find a standard solution rather than the optimal solution. Therefore, they proposed a logical layout algorithm for DICs based on an expert knowledge system and a barycenter algorithm. Usually, the logical layout can be divided into two phases: horizontal and vertical ordering. It is used to determine the logical position of component symbols in the layout. In the horizontal layout phase, a recursive graph traversal algorithm is used to ensure the signal flow direction of the circuit follows the left-to-right layout norm. This algorithm detects and removes feedback loops from component connections and determines the columns where the component symbols are located. Once column positions are assigned to all component symbols, the vertical sorting phase assigns rows where the component symbols are located based on the barycenter algorithm. This algorithm calculates the average row position with respect to connected symbols or the row position where the center of gravity is located to determine the row position of the current symbol. Finally, the knowledge system is utilized to find a standard layout based on heuristic layout criteria such as the number of crossings between networks and the number of wire bends.

The barycenter algorithm is a classical sorting method utilized in logical layout. It exhibits a time complexity of $O(n)$ and a space complexity of $O(1)$ when sorting n elements. This algorithm is characterized by its rapid computational speed and minimal space requirements. Consequently, it finds a broad array of applications in the automatic generation of schematics for DICs. The bubble algorithm, based on value propagation proposed by Jehng et al., is categorized as a variant of the barycenter algorithm. During the horizontal sorting phase of the logical layout, circuit leveling is used to assign each symbol its respective column position from left to right. During the vertical sorting phase, the bubble algorithm based on value propagation is employed to assign the symbols their respective rows. The algorithm's process is depicted in Fig.3. At the outset, a value is set for each input symbol according to its initial row position. Subsequently, each symbol in the netlist calculates its value by averaging the values of the connected symbols. Once all symbols have been assigned values, each column is reordered according to the symbol value, thereby assigning column positions to each symbol. In comparison to the barycenter algorithm, this algorithm exhibits a higher time complexity of $O(n^2)$ due to the need to compare all symbols at each iteration. However, it compensates for this increased complexity by generating layout results with fewer signal line crossings.

In schematic generation for AICs, Arsintescu et al. employed a rule-based heuristic approach (Arsintescu, 1996). They identified symmetric symbol pairs and arranged them in a mirror pattern. These pairs were then placed on a grid and the layout was optimized by minimizing wire length and wire bends. The heuristic rules acted as supervisors to ensure smooth layout paths during generation. In addition, the vertical layout method proposed by Garg et al. (2008) assigns vertical positions to function block symbols in a top-to-bottom manner according to the signal flow direction incrementally. By ensuring both horizontal and vertical alignment in symbol placement, the method is able to avoid crossover and overlap between networks as much as possible. Wu et al. proposed a function block symbol layout method with the aim of minimizing wire length (Wu, 2009).

Unedited

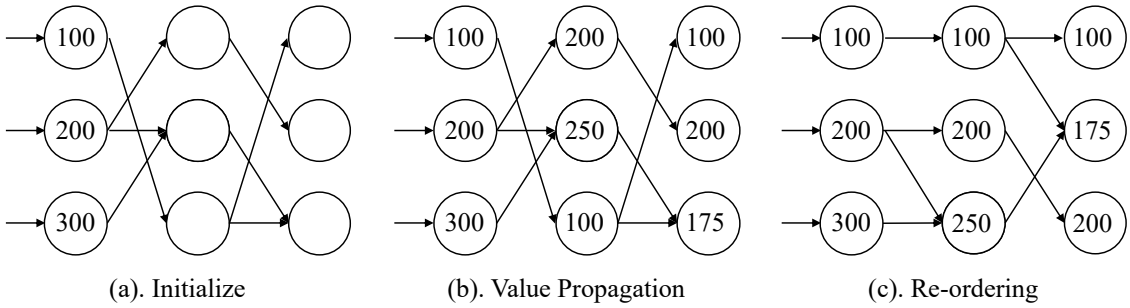

Fig. 3 Bubbling algorithm based on value propagation

This method integrates symmetry, alignment, and other constraints during the placement of component symbols.

The advantage of these heuristic methods lies in their ability to generate layouts using explicit rules and strategies. These rules and strategies can be adjusted and optimized to meet specific generation needs with high interpretability. Additionally, feasible standard solutions can be generated quickly to ensure layout efficiency. However, the heuristic rules used in these methods lack the capability to achieve the global optimal solution, making it challenging to ensure the quality of the layout.

The simulated annealing algorithm, inspired by the physical process of material annealing, aims to find the globally optimal solution through probabilistic searching. This algorithm involves randomly exploring the solution space and has been extensively applied to address layout optimization challenges. Regarded as a high-performance heuristic algorithm, it was utilized by Lee et al. for the automatic generation of DICs (Lee and McNamee, 1989). The method focuses on optimizing symbol row sequences within the layout space by iteratively and randomly swapping the positions of symbols in the same column to reduce wire crossings. Its detailed algorithmic pseudo-code can be found in Supplementary Algorithm 1. Theoretically, the algorithm possesses the capability to approximate a globally optimal solution as the termination temperature T_f approaches 0. The time complexity of this algorithm is contingent upon the temperature settings and the number of iterations conducted in each round. It is mathematically represented as $O\left(\frac{L\log(T_f/T_0)}{\log(r)}\right)$.

Although the simulated annealing algorithm has shown some potential in layout, subsequent research on ASG has not been thoroughly investigated. One reason is its low search efficiency, especially in large-scale schematic layout tasks, leading to slow convergence and unsatisfactory results. In contrast, the simulated annealing algorithm has excelled in very-large-scale integrated circuit design. The optimization of its search efficiency has been studied (Sechen, 1988; Sergey et al., 2019; Katsuki et al., 2022), which greatly overcomes the challenge of slow convergence speed. These studies provide valuable insights into the application of the algorithm in ASG. Further optimizations to the simulated annealing algorithm are anticipated to enhance the quality of ASG while ensuring a certain level of search efficiency and convergence speed. This advancement is expected to enhance the practical application value of the layout algorithm for ASG.

3.2 Layout algorithms based on DRL

Layout algorithms based on DRL have become a recent research hotspot in the realm of ASG. The algorithms tackle the schematic generation problem by converting it into a combinatorial optimization problem. Through guidance from the reward function, reinforcement learning agents are trained to determine the placement of component symbols, thereby achieving the schematic layout.

The method proposed by Hsh et al., based on building block classification and reinforcement learning, is a classic example of automatic AIC schematic generation algorithms (Hsu and Lin, 2022). This approach achieves intelligent component placement through the interaction between the agent and the layout environment. At each step, the agent observes the canvas's current state, feeding this information into the neural network along with embedded features detailing building blocks and signal flow to receive layout suggestions. Subsequently, guided by

Unedited

the neural network's output, the agent executes suitable placement actions for the component symbols on the canvas. Through continuous iteration and guided by the reward function, the reinforcement learning agent engages in trial-and-error processes and self-improvement within the solution space. This allows the agent to autonomously discover the optimal layout strategy, leading to the best layout outcomes. Among them, the reward function covers the optimization of several metrics, such as the area of building blocks, network crossings, wire bends, and the total length of wires, to ensure that the schematic layout meets the requirements of circuit generation. Supplementary Algorithm 2 shows the process of the layout algorithm based on reinforcement learning.

In the realm of reverse automatic generation of PCB schematics, Yang et al. (2024) introduced a reinforcement learning layout method centered on aesthetic evaluation metrics (AEM), called AEM-PCB Reverser. The method is shown in Fig.4. A key feature of this method is the introduction of metrics that objectively quantify the quality of PCB schematic diagrams. These metrics consider factors like the normality and readability of layout and routing, ensuring a high level of consistency between the metric scores and the designer's subjective evaluation. By incorporating these metrics into the reward function, the reinforcement learning agent is guided to perform layout optimization. This addresses the problem that PCB schematics are difficult to generate automatically due to the lack of objective standards in the past. It provides a reliable metric foundation for the automatic generation and optimization of PCB schematics.

Overall, in ASG, the layout algorithm based on DRL demonstrates good performance. It not only discovers the optimal solution in the layout space but also operates solely through agents spontaneously without manual parameter adjustments. However, a drawback is that the state and action space of reinforcement learning experiences an exponential expansion as schematic density increases. This poses challenges such as reduced convergence speed or even difficulty in converging in complex schematic environments. Taking the method of Hsu et al. as an example, while yielding some results, was limited by the schematic's component count not exceeding 40. This underscores the constraints of layout algorithms based on DRL when confronted with complex

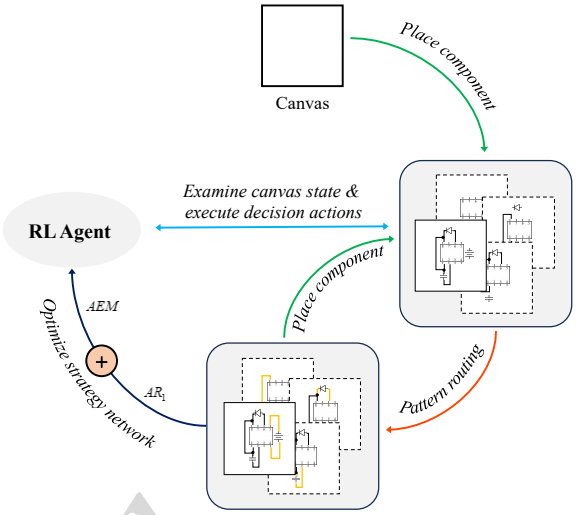

Fig. 4 The method of the AEM-PCB reverser. AEM, aesthetic evaluation metric

schematics.

However, despite facing challenges in ASG due to increasing schematic density, the layout algorithms based on DRL still hold significant potential for generating complex schematics. This potential is exemplified by the widespread application of this algorithm in chip layout design (Agnesina et al., 2020; Vashisht et al., 2020; Lai et al., 2023; Shi et al., 2023; Zhong et al., 2024a). These applications integrate layout problems with DRL, aiming to optimize the power, performance and area (PPA) of chip designs, significantly improving the floorplanning results. It is well known that chip layout designs typically deal with hundreds of millions of transistors, and their complexity is even greater. To mitigate layout density, transistors are often grouped and abstracted into higher-level standard and macro cells. Following this approach, the chip layout algorithm employs DRL-based layout for fewer, more functional macro-cells, while heuristics enhance layout speed for a larger number of standard cells. This hierarchical layout strategy serves as a valuable reference for reducing layout density and enhancing the algorithm's generation efficiency in complex and densely-packed circuits within ASG.

This section reviews the layout algorithms for ASG, which mainly covers two major categories: heuristic-based layout algorithms and DRL-based layout algorithms. It summarizes the advantages and shortcomings of various layout algorithms in Table 2.

Unedited

Table 2 Advantages and shortcomings of various layout algorithms

Category	Names	Advantages		Shortcomings
Heuristic-based	Heuristics with rules	Higher interpretability, more quickly.		Lack of ability to obtain optimal solutions.
	Simulated annealing	Strong search capability for optimal solutions.		Slow convergence and sensitive to parameters.
DRL-based	The method of Hsu et al.	Embedded building block features for high quality.	Insensitive to parameters and able to obtain optimal solutions.	The search space expands dramatically with increasing schematic density.
	AEM-PCB Reverser	Objective PCB schematic quality metrics are introduced.		

4 Routing algorithms

The core of routing involves path searching, which entails using a specific search strategy to locate the optimal path meeting certain constraints in a complex routing setting. According to the different order of wire network arrangement, routing algorithms can be divided into two categories: sequential routing algorithms and simultaneous routing algorithms.

Sequential routing algorithms follow a specific iterative order to wire the network step by step. Common sequential routing algorithms are maze search algorithms, and pattern routing algorithms. In addition, reinforcement learning-based routing algorithms that appear in recent years adaptively adjust the routing strategy by employing agents to control the direction of the routing also belong to sequential routing algorithms. Simultaneous routing algorithms, in contrast, simultaneously route all the networks and typically rely on integer linear programming (ILP) to search for routing paths under multiple optimization objectives.

This section provides an overview of the main routing algorithms, including those used in ASG. The algorithms are categorized based on sequential and simultaneous routing approaches from an algorithmic perspective.

4.1 Sequential routing algorithm

During the routing process, the sequential routing algorithm arranges the networks in a specific order. It interconnects the endpoints in the wire network one by one, selecting a relatively better routing path in each interconnection. These paths are optimized through multiple iterations to achieve the best routing results. Classical sequential routing algorithms include Lee's algorithm, A* algorithm, the pattern routing algorithm, and the minimum Steiner tree algorithm.

4.1.1 Lee's algorithm and A* algorithm

Lee's algorithm (Abel, 1972) is a shortest path method that calculates the routing cost between a source and a target point through wave propagation. The algorithm employs a breadth-first strategy, starting from the source point and calculating the routing cost of its neighboring points. It then uses these neighboring points as a new base point to continue spreading outward, calculating the routing cost of the neighboring points from the source point. This process repeats until the base point reaches the target point, marking the end of the spreading. Finally, the algorithm selects the routing path with the lowest cost. Due to the necessity of calculating the routing cost for all points between the source and target points, Lee's algorithm has high computational complexity, requires significant memory resources, and operates at a slow speed.

In contrast, the A* algorithm (Hart et al., 1968) improves upon Lee's algorithm by searching for the shortest path more efficiently. It achieves this by incorporating a heuristic function to direct the routing search. The A* algorithm calculates the sum of the actual cost from the source point to the current point and the estimated cost from the current point to the target point. This sum serves as a reference value, guiding the selection of the next base point with the smallest total cost. The algorithm then proceeds to compute the total costs of the neighboring points of this base point and continues this process, gradually directing the search toward the target point. Unlike Lee's algorithm, the A* algorithm only computes the total costs along the search path, reducing computational complexity to some extent and enhancing

Unedited

routing speed. Supplementary Algorithm 3 shows the process of the A* algorithm.

Frezza and Levitan (1993) introduced the schematics placement and routing method (SPAR) for ASG. The method employs the A* algorithm for path searching in global routing. It calculates the actual cost based on path congestion and the Manhattan length from the source to the base point. The estimated cost is set at 0.8 times the Manhattan length from the base point to the nearest endpoint in the network. By using a low estimate, the method ensures an optimal solution for path searching.

4.1.2 Pattern routing algorithm

Despite significant improvements based on Lee's algorithm, the traditional A* algorithm still faces the challenge of high computational and memory overhead for complex routing. This challenge stems from the fact that the A* algorithm is based on a search-based pathfinding approach, and complex routing over long distances leads to high time and memory overhead.

In contrast, unique advantages are demonstrated by the pattern routing algorithm (Kastner et al., 2000). The algorithm significantly reduces complexity by employing predefined routing patterns to guide the pathfinding process, and it achieves a time complexity of only $O(1)$. Several common routing patterns in pattern routing, such as I-type, L-type, Z-type, and U-type, are demonstrated in Fig.5. The routing patterns can also be customized for various application scenarios or circuit characteristics to ensure the applicability and scalability of pattern routing. However, compared to A* algorithms, flexibility is lost by pattern routing algorithms, resulting in relatively lower routing quality. Taking L-pattern routing as an example, only two routing paths can be selected. In extreme cases, both paths may not be ideal, which can adversely affect the quality of the routing results.

Combining pattern routing with maze search algorithms is an efficient way to tackle this issue, where pattern routing is applied for quick routing at a reasonable cost, and maze search is utilized to provide better routing results when the path cost is too high. This method was used in the generation of AICs by Hsu and Lin (2022). The routing process was greatly accelerated by using a pattern routing algorithm to connect the component symbols on the schematic canvas. For components that could not be fully connected by pattern routing, the maze search approach was utilized as a supplement to routing. In addition to yielding good routing results, this integrated use of maze search and pattern routing algorithms greatly lowers the method's complexity, providing a fast and effective strategy for generating schematics.

4.1.3 Right-angle minimum Steiner tree algorithm

Both A* algorithms and pattern routing algorithms are usually used to deal with the routing of two-ended wire networks only, i.e., they can effectively solve the pathfinding problem with only one source and target point. However, for more complex multi-terminal wire network routing, these algorithms appear to be inadequate. To cope with this challenge, the commonly adopted routing method is the right-angle minimum Steiner tree algorithm.

The Right-Angle minimum Steiner tree (RMST) algorithm implements the routing of a multi-terminal wire network by introducing auxiliary Steiner points in the network with the goal of minimizing the routing cost. To ensure the rationality of the routing, Steiner points are typically selected in the right-angle space of the multi-terminal wire network. This means that the Steiner point is always aligned horizontally or vertically with either endpoint in the multi-terminal wire network.

Let the nodes in the multi-terminal network be defined as a terminal set of size k. The RMST algorithm employs dynamic programming to construct the minimum tree, with the primary source of computational complexity stemming from this approach. Central to dynamic programming is the idea of decomposing complex problems into simpler subproblems, which can then be recombined. In RMST, the terminal set is partitioned, and these subsets are enumerated to form subtrees. Ultimately, these subtrees are merged to derive the minimum Steiner tree for the terminal set. This process has a computational complexity of $O(n3^k)$. Consequently, while RMST is capable of addressing the routing issues in multi-terminal networks, it faces challenges related to high computational complexity and the difficulty of achieving optimal solutions.

To address these challenges, researchers have proposed heuristic-based RMST algorithms (Hanan, 1966). A Steiner tree heuristic routing algorithm

Unedited

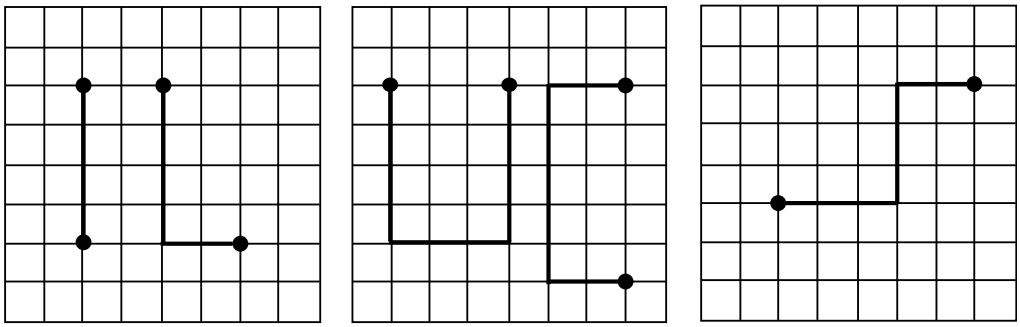

Fig. 5 Pattern routing style

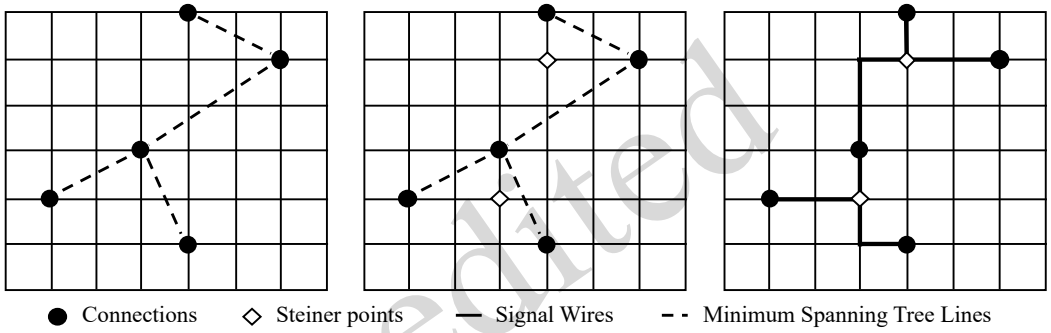

Fig. 6 Iteration-based Steiner tree heuristic routing

Unedited

based on continuous iteration was proposed by Lee and McNamee (1992). This algorithm utilizes minimum spanning trees to adjust the rectangular space and reduce the generation complexity of Steiner trees. As shown in Fig. 6, the method first generates a minimum spanning tree for a multi-terminal network and then iteratively works on edges with common nodes in the spanning tree. If there is a Steiner point in the rectangular space of these common edges, the two edges are replaced with three edges connected to that point. This process continues iteratively until no new Steiner points are generated. Finally, the line edges are converted to straight edges to generate the routing result. The rectangular space is a set of nodes aligned horizontally or vertically with the end points of edges.

Additionally, Chu et al. introduced a fast lookup table-based RMST algorithm (Chu and Yiu-Chung Wong, 2008), significantly reducing the algorithm's time complexity at the expense of some storage space. For multi-terminal wire networks with nine or fewer endpoints, the optimal RMST can be directly obtained through table lookup. For networks with fewer than 100 endpoints, the algorithm can still achieve suboptimal solutions with high accuracy.

This section presents various common sequential routing algorithms, each with distinct advantages and disadvantages, capable of yielding different outcomes based on the scenario. Apart from the aforementioned algorithms, routing algorithms leveraging DRL have emerged as significant methods recently.

The DRL global routing method proposed by Liao et al. (2020) controls routing direction through agents to determine routing paths, surpassing the A* algorithm and showcasing the broad application potential of DRL in routing planning problems. Meanwhile, the HubRouter algorithm introduced by Du et al. (2023) represents a novel generative approach to global routing. This method is structured into two distinct phases. In the first phase, it employs image generation models, including Generative Adversarial Networks (GAN) and Variational Auto-Encoders (VAE), to create hubs. The second phase integrates the RSMT algorithm with a deep learning model

to effectively connect to these hubs. This strategy significantly enhances performance by thoroughly merging traditional algorithms with advanced deep learning techniques. Furthermore, the PreRoutGNN algorithm proposed by Zhong et al. (2024b) incorporates a pre-training technique within the routing process. This integration effectively addresses the challenges of signal decay and error accumulation that arise in long-timing paths, achieving state-of-the-art results in the spare prediction tasks of several realistic circuits. These advancements underscore the evolving trend of deep learning technology in the routing domain and pave the way for new explorations in both theoretical research and practical applications of future routing technologies.

4.2 Simultaneous routing algorithm

Sequential routing algorithms execute the routing of each network one by one in a predetermined order. The front-end routing may have a detrimental impact on subsequent routing, leading to diverse outcomes based on different sequences (Abel, 1972). At times, regional congestion may arise due to constraints in wiring paths, posing challenges in attaining optimal outcomes in sequential wiring.

Simultaneous routing algorithms offer greater flexibility compared to sequential routing algorithms. They are not constrained by a specific routing order; instead, they process all the networks in the circuit simultaneously. This approach helps address suboptimal solution issues that may arise from following a predetermined routing order. Typically rooted in ILP, this algorithm treats edges or points of the routing networks as independent variables. It comprehensively considers various routing constraints, constructs the objective function and constraints, and determines the optimal routing scheme by minimizing the objective function while meeting the constraints. Ou et al. (2014) introduced a simultaneous routing approach, which integrates the four constraints of symmetry, concentricity, topology matching, and length matching into the AIC routing problem. By employing an ILP formulation and binary decision variables (0–1), they achieved precise edge assignments in routing, leading to superior results compared to sequential routing strategies.

However, the complexity of ILP is intricately linked to the complexity of both the objective function and the constraints. Consequently, common ILP algorithms, such as the branch-and-bound method and the cut-plane method, typically exhibit exponential time complexity in most scenarios. This characteristic renders this class of algorithms computationally inefficient. To tackle this issue, Wu et al. (2015) introduced a hybrid strategy. They employ a rapid sequential routing algorithm as an initial step to generate candidate routing paths for each wire network. Subsequently, an ILP formula with binary decision variables is constructed to determine the adoption of specific candidate paths. This strategy skillfully balances the quality and efficiency of routing, significantly improving the efficiency of simultaneous routing by reducing the solution space.

In general, the simultaneous routing algorithm exhibits its superiority in improving routing quality. Through the utilization of a global optimization approach, the algorithm has the capability to identify the optimal solution while simultaneously adhering to multiple constraints. However, the simultaneous routing algorithm demonstrates a relatively high time cost. Subsequent research efforts could further explore the enhancement of the simultaneous routing algorithm to improve its solution efficiency while maintaining routing quality.

This section provides an overview of routing algorithms for ASG, covering two main categories, sequential routing algorithms and simultaneous routing algorithms. Table 3 summarizes the advantages and shortcomings of the various routing algorithms.

5 PCB Automatic Schematic Generation

The typical structure of a PCB is centered around a main chip as the core, multiple secondary chips as sub-cores, and surrounded by active and passive components forming the peripheral circuit. The main chip typically serves as the central component of the overall circuit, controlling and coordinating the operation of the entire system. Secondary chips are tasked with executing specific functions, such as signal processing and power management. Peripheral circuits encompass passive components, such as resistors, capacitors, and inductors, and active components, such as transistors and operational amplifiers. In contrast to integrated circuits, the schematic structure of a PCB is more macroscopic, and the shapes and sizes of components are differ-

Unedited

Table 3 Advantages and shortcomings of various routing algorithms

Category	Name	Advantages	Shortcomings
Sequential routing	Maze search routing	Detailed path search, good ability to obtain optimal solutions, and good routing quality.	High computational complexity and memory footprint, slow computation.
	Pattern routing	Low computational complexity and routing cost, allowing for fast routing.	Fixed-mode routing, lack of flexibility, and poor routing quality.
	Right-angle minimum Steiner tree	Capable of handling the routing of multi-terminal wire networks.	NP-complete problem with high algorithmic complexity.
	DRL-based routing	Strong global routing capability, spontaneous optimization without manual intervention.	Agent training is slow and routing is inefficient.
Simultaneous routing	Totally simultaneous routing	Synchronized routing avoids adverse effects caused by routing sequences.	The computational complexity increases dramatically with the increase of optimization conditions.
	Hybrid routing	Predefined candidate paths reduce the optimization solution space and improve routing speed.	Somewhat reduces the routing quality.

ent, which is complex and special. In the past, the ASG method was primarily based on DICs or AICs, characterized by high regularity and displaying progressive and symmetrical layout patterns, distinct from PCB circuits. Consequently, the conventional automatic schematic layout and routing techniques pose challenges when directly employed for PCB generation tasks.

This section addresses the technology of ASG in PCBs, discusses its metrics, and analyzes the current difficulties and challenges encountered in this field. Based on this analysis, viable solutions are explored to offer valuable references and insights for the future advancement of automatic PCB schematic generation technology.

5.1 Metrics

Schematic drawings represent a fusion of artistic aesthetics and technical accuracy, where their legibility plays a crucial role in ensuring the precise communication of information. Designers infuse their distinct perspectives and styles into the creation of schematics, which undoubtedly adds to the complexity and subjectivity of evaluating schematics (Jehng et al., 1991). Nevertheless, they will always follow a common set of design norms when creating schematics. For example, the flow of signals usually follows the principle of left-to-right, and the number of wire crossings and bends is minimized. These established norms serve the purpose of ensuring the clarity, consistency, and precision of schematics, as well as facilitating mutual comprehension and utilization of designs by various designers. Hence, although the evaluation of schematics may differ based on personal aesthetics and viewpoints, adhering to standard design norms is unquestionably crucial for ensuring the quality of schematics.

These design norms play a crucial role in the generation of schematics as they are essential prerequisites for the automated optimization of layout and routing. Traditional heuristics commonly employ optimization objectives, such as minimizing the number of wire crossings and bends and reducing the wire length in the routing, to generate schematics that meet the design norms. The intricate nature of the PCB schematic structure necessitates a more comprehensive metric for optimization objectives, as traditional criteria may not adequately capture all aspects of its design norms. For instance, the optimization objective of minimizing wire crossings and bends makes it difficult to express the design specification for the flow of signals from left to right and top to bottom in the PCB schematic design. Similarly, efforts to reduce wire length often result in overly compact component arrangement in the layout, thereby diminishing the schematic's readability.

In order to solve the problem of the lack of objective metrics for evaluating PCB schematic quality, Yang et al. (2024) introduced the AEM as a method for objectively assessing the quality of PCB schematics, which is based on the layout and routing norms found in PCB schematic diagrams. Through a comprehensive analysis of common regular features in numerous schematic diagrams, they identify the fun-

Unedited

damental elements that influence the attributes of PCB schematic diagrams. These attributes encompass clarity, readability, and neatness. Subsequently, by employing mathematical quantification methods and the refinement of relevant formulas, the evaluation metric was effectively developed to align closely with the subjective assessments of designers, maintaining a high level of consistency with them. This marks the initial step toward supporting the automatic generation of PCB schematics by establishing a foundation for the metric. The formula is presented as follows:

$$AEM = \sum \frac{\sum_{i=1}^{m} R_{Li} \times D_{WBi} \times D_{NCi}}{m} \tag{1}$$

$$R_{Li} = \frac{R_{bi}}{\alpha} + \frac{(\alpha - 1)(R_{bi} - P_i)}{\alpha} \tag{2}$$

where m denotes the number of routing, D_{WB} and D_{NC} denote the discount factors determined by the number of routing bends and the number of network crossings, R_b is affected by the alignment of the components and denotes the baseline fraction, P denotes the penalty fraction to constrain a single routing to be in a given range of lengths, and α is the single routing robustness factor, which is generally taken to be a value of 2 in the experiments.

5.2 Difficulties and discussion

5.2.1 Reverse residual netlist processing is difficult

In the process of forward design, the transformation from the PCB schematic to the netlist involves comprehensive information about electronic components, such as component footprints, parameters, and the interconnection relationships. In contrast, when generating schematic diagrams in reverse engineering, the netlist is typically obtained by scanning the PCB image for deduction. This process can easily result in information loss, which is predominantly evident in indeterminate component footprints, missing parameters, and ambiguous connections, leading to the circuit netlist obtained with residuals. These residual netlists are crucial for ensuring the accuracy and completeness of schematic generation. Therefore, the automatic restoration of the schematic based on the residual netlists poses a significant challenge in reverse engineering for PCB schematic generation.

The utilization of graph neural network technology for matching and identifying residual networks has proven to be an effective approach in tackling the challenges associated with generating schematic diagrams for residual netlists. During the PCB design process, the circuit design builds its peripheral circuits based on the core chip, and these peripheral circuits are usually derivatives of the chip's reference circuits. Therefore, there are similarities in the connection relationships and schematic structures between the peripheral circuits on the PCB and the reference circuit. By training the graph neural network model with ample reference circuit data, it can learn the underlying correlations among component types, parameters, and connections. This enables the model to supplement components and connections in the residual netlists, offering precise and complete circuit netlists for PCB schematic generation, thereby reducing the complexity of reverse circuit schematic generation.

In addition, the image recognition method employing convolutional neural networks is used for analyzing the PCB image. When integrated with the component library, this method can effectively identify the types and footprints of components in the circuit, thereby supplementing any missing parameter information in the residual netlist. The component library is a collection of information containing various electronic component footprints and types, device parameters, and pin parameters. Through the utilization of an image recognition method, the identifiers of components in the PCB image are recognized, and the corresponding footprints and parameters are matched from the component library. Convolutional neural networks, known for their superior performance in image recognition tasks, have the potential to significantly improve the recognition accuracy and matching precision of missing component information in PCB circuits. This process aids in filling in the gaps in the netlist's information and improving the clarity and readability of the generated PCB schematic.

5.2.2 Difficulty in generating complex and dense schematics

In PCBs, especially in densely populated ones with numerous components and complex connections, fast layout and routing are crucial for generating clear and readable PCB schematics. This is

Unedited

essential to guarantee their efficient implementation and application in PCB reverse engineering.

However, global optimization algorithms used for layout and routing, such as layout algorithms based on DRL or heuristic layout algorithms based on simulated annealing, necessitate conducting numerous searches in the global space. This results in slow convergence of the algorithms. When confronted with tasks of generating PCB schematics, the complexity of the layout increases non-linearly as the quantity of components sharply rises. This causes the above algorithms to be stretched in terms of processing efficiency and may not even be able to complete the schematic generation. Therefore, improving the efficiency of complex and intensive schematic generation while ensuring the quality of layout and routing is one of the current PCB reverse engineering problems that needs to be solved.

The incorporation of heuristic rules into global algorithms has been identified as an effective strategy for enhancing search efficiency. These rules are usually based on expert knowledge and have the capability to remove numerous invalid sub-solutions within the solution space through straightforward and efficient computation. By restricting the search scope, efficient pruning can be implemented during the global search process. In tasks involving the intensive automatic generation of PCB schematics, the utilization of heuristic rules to eliminate actions that do not adhere to design norms or lead to suboptimal results can significantly decrease the search space of the global layout and routing algorithm. This enhances the efficiency of the generation process.

By simplifying netlists and partitioning functional modules within dense PCB schematics, it is possible to substantially reduce the search depth of global algorithms. Positive PCB design generally adheres to the "top-down" approach, starting from the overall circuit function and systematically decomposing it into distinct peripheral circuit or functional modules. Subsequently, the components in each specific module are placed and routed to form a complete circuit schematic design. This design concept is characterized by its well-organized structure, which can greatly simplify complex problems and improve design efficiency.

The algorithm for netlist partitioning based on graphs (Hong et al., 2023) utilizes a graph neural network to automatically analyze and understand the graph structure of complex circuit netlists. The algorithm segregates the circuit netlist into different modules based on the functional and structural characteristics of the circuit. This process facilitates the potential for automated generation of schematic representations in a "top-down" manner. In PCBs, Meng et al. used an attribute-based netlist partitioning approach (Meng and Zheng, 2022), segmenting the PCB netlist into functional modules. Followed by the independent layout and routing of each module, it can significantly decrease the global search algorithm's depth, leading to a significant enhancement in the efficiency of automatically generating densely populated schematics for PCBs. It is important to note that the division results significantly influence the comprehensibility of the schematic. Therefore, the accuracy and effectiveness of the netlist partitioning algorithm are pivotal factors that need to be taken into account in this strategy.

Overall, schematic generation and reduction based on residual netlist is an important feature and difficulty of PCB reverse engineering. Improving the accuracy and reliability of automatic PCB schematic generation technology with a residual netlist is an important prerequisite to enable designers to understand and read the reverse schematic effectively. Efficiently addressing the challenge of generating intensive PCB schematics is fundamental to the engineering and application of this technology. It enhances the breadth and speed of automated generation, facilitating its widespread application in practical engineering. There exists a progressive relationship between the two issues; the current PCB ASG technology is an important research direction.

6 Conclusions

This paper provides an overview of ASG, detailing its process, methodology, and core algorithms, specifically focusing on layout and routing. Drawing from the current state of automatic PCB schematic generation, this paper delves into three aspects: objective evaluation metrics, difficult challenges, and feasible solutions. The goal is to propel the future development of automatic PCB schematic generation.

The core algorithms for ASG are layout and routing. For layout, this paper introduces heuristic-based layout algorithms and DRL-based layout al-

Unedited

gorithms. Heuristic-based layout algorithms offer high interpretability and faster layout speed, but face challenges in achieving the optimal layout. Although the simulated annealing method can achieve the global optimal solution, it encounters low search efficiency. Further research on the convergence speed of the simulated annealing algorithm can enhance the application and development of ASG.

Layout algorithms based on DRL operate without human intervention and depend solely on the autonomous learning of the agent. They exhibit low reliance on the initial layout. However, as circuit complexity increases, the action space and state space expand significantly, posing challenges for convergence in learning. The solution to this issue lies in the exploration of hierarchical schematic generation algorithms.

For routing, this paper introduces sequential routing algorithms and simultaneous routing algorithms. The sequential routing algorithm typically conducts path searches for each wire network individually, exhibiting low algorithmic complexity, resulting in faster routing speed. Nevertheless, varying routing sequences may lead to suboptimal solutions. The simultaneous routing algorithm addresses the issue of suboptimal solutions due to routing sequences. However, the computational complexity associated with linear optimization for multiple objectives is notably higher compared to that of sequential routing, resulting in lower routing efficiency. Therefore, further exploration of routing algorithm efficiency while maintaining quality is a significant research area in the realm of ASG.

This paper also emphasizes the discussion on ASG for PCB reverse engineering. The structure of the PCB schematic is unique, and the shape and size of its components are different. The challenges faced in the automated generation of schematics for PCB reverse engineering are significantly influenced by the uniqueness and differentiation. Under the premise of having an automatic optimization metric, the schematic generation problem of residual netlist is the primary challenge in PCB reverse engineering. The identification and matching algorithm based on deep learning is an effective way to complement the residual netlist and improve the accuracy of PCB schematic generation. Furthermore, the engineering application of PCB schematic generation technology faces a significant challenge of low efficiency in intensive PCB schematic generation. Heuristic algorithms and modular generation serve as crucial breakthroughs in tackling this challenge.

Contributors

Bin YAN designed the research. Jie YANG and Kai QIAO drafted the paper. Chen CHEN helped organize the paper. Lixiang GUO and Jian CHEN revised and finalized the paper. Jie YANG and Kai QIAO contribute equally to the article.

Conflict of interest

All the authors declare that they have no conflict of interest.

References

Abel LC, 1972. On the ordering of connections for automatic wire routing. IEEE Trans Comput, C-21(11):1227-1233. https://doi.org/10.1109/T-C.1972.223482

Agnesina A, Chang K, Lim SK, 2020. VLSI placement parameter optimization using deep reinforcement learning. Proc IEEE/ACM Int Conf on Computer Aided Design, p.1-9.

Arsintescu BG, 1996. A method for analog circuits visualization. Proc Int Conf on Computer Design VLSI in Computers and Processors, p.454-459. https://doi.org/10.1109/ICCD.1996.563593

Botero UJ, Wilson R, Lu HW, et al., 2020. Hardware trust and assurance through reverse engineering: a survey and outlook from image analysis and machine learning perspectives. https://doi.org/10.48550/arXiv.2002.04210

Cheng RY, Yan JC, 2021. On joint learning for solving placement and routing in chip design. Proc 35th Int Conf on Neural Information Processing Systems, article 1262.

Chiueh TC, 1991. HERESY: a hybrid approach to automatic schematic generation (for VLSI). Proc European Conf on Design Automation, p.419-423. https://doi.org/10.1109/EDAC.1991.206438

Cho M, Pan DZ, 2007. BoxRouter: a new global router based on box expansion and progressive ILP. IEEE Trans Comput Aided Des Integr Circuits Syst, 26(12):2130-2143. https://doi.org/10.1109/TCAD.2007.907003

Chu C, Wong YC, 2008. FLUTE: fast lookup table based rectilinear Steiner minimal tree algorithm for VLSI design. IEEE Trans Comput Aided Des Integr Circuits Syst, 27(1):70-83. https://doi.org/10.1109/TCAD.2007.907068

Du XB, Wang CH, Zhong RZ, et al., 2023. HubRouter: learning global routing via hub generation and pin-hub connection. Proc 37th Int Conf on Neural Information Processing Systems, article 3435.

Eick M, Strasser M, Lu K, et al., 2011. Comprehensive generation of hierarchical placement rules for analog integrated circuits. IEEE Trans Comput Aided Des Integr Circuits Syst, 30(2):180-193. https://doi.org/10.1109/TCAD.2010.2097172

Unedited

Ferreira A, Lourenco N, Martins R, et al., 2016. Automated analog IC design constraints generation for a layout-aware sizing approach. 13th Int Conf on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design, p.1-4. https://doi.org/10.1109/SMACD.2016.7520740

Frezza ST, Levitan SP, 1993. SPAR: a schematic place and route system. IEEE Trans Comput Aided Des Integr Circuits Syst, 12(7):956-973. https://doi.org/10.1109/43.238032

Fu RL, Zhang ZM, Tang GM, et al., 2020. Design automation methodology from RTL to gate-level netlist and schematic for RSFQ logic circuits. Proc on Great Lakes Symp on VLSI, p.145-150. https://doi.org/10.1145/3386263.3406898

Garg B, Agrawal A, Sehgal R, et al., 2008. Partitioning, floor planning, detailed placement and routing techniques for schematic generation of analog netlist. Proc IEEE East-West Design & Test Symp, p.379-382. https://doi.org/10.1109/EWDTS.2008.5580148

Goldie A, Mirhoseini A, 2020. Placement optimization with deep reinforcement learning. Proc Int Symp on Physical Design, p.3-7. https://doi.org/10.1145/3372780.3378174

Gu JQ, Jiang ZX, Lin YB, et al., 2020. DREAMPlace 3.0: multi-electrostatics based robust VLSI placement with region constraints. Proc IEEE/ACM Int Conf on Computer Aided Design, p.1-9.

Hanan M, 1966. On Steiner's problem with rectilinear distance. SIAM J Appl Math, 14(2):255-265. https://doi.org/10.1137/0114025

Hart PE, Nilsson NJ, Raphael B, 1968. A formal basis for the heuristic determination of minimum cost paths. IEEE Trans Syst Sci Cybern, 4(2):100-107. https://doi.org/10.1109/TSSC.1968.300136

Hong XN, Lin T, Shi YQ, et al., 2023. GraphClusNet: a hierarchical graph neural network for recovered circuit netlist partitioning. IEEE Trans Artif Intell, 4(5):1199-1213. https://doi.org/10.1109/TAI.2022.3198930

Hsu HY, Lin MPH, 2022. Automatic analog schematic diagram generation based on building block classification and reinforcement learning. Proc ACM/IEEE 4th Workshop on Machine Learning for CAD, p.43-48. https://doi.org/10.1109/MLCAD55463.2022.9900093

Ivanova M, Rozeva A, Ninov A, et al., 2022. Reinforcement learning at design of electronic circuits: review and analysis. Proc 5th Artificial Intelligence and Cloud Computing Conf, p.275-284. https://doi.org/10.1145/3582099.3582140

Jehng YS, Chen LG, Parng TM, 1991. ASG: automatic schematic generator. Integration, 11(1):11-27. https://doi.org/10.1016/0167-9260(91)90004-5

Jumper J, Evans R, Pritzel A, et al., 2021. Highly accurate protein structure prediction with AlphaFold. Nature, 596(7873):583-589. https://doi.org/10.1038/s41586-021-03819-2

Kastner R, Bozorgzadeh E, Sarrafzadeh M, 2000. Predictable routing. IEEE/ACM Int Conf on Computer Aided Design. IEEE/ACM Digest of Technical Papers, p.110-113. https://doi.org/10.1109/ICCAD.2000.896459

Katsuki K, Shin D, Onizawa N, et al., 2022. Fast solving complete 2000-node optimization using stochastic-computing simulated annealing. 29th IEEE Int Conf on Electronics, Circuits and Systems, p.1-4. https://doi.org/10.1109/ICECS202256217.2022.9971124

Kim NH, Kim KS, Choi KM, et al., 2000. RightTopologizer: an efficient schematic generator for multi-level optimization. Proc 13th Annual IEEE Int ASIC/SOC Conf, p.387-391. https://doi.org/10.1109/ASIC.2000.880769

Kunal K, Madhusudan M, Sharma AK, et al., 2019. ALIGN: open-source analog layout automation from the ground up. Proc 56th Annual Design Automation Conf, article 77. https://doi.org/10.1145/3316781.3323471

Lageweg CR, 1998. Designing An Automatic Schematic Generator for A Netlist Description. Technical Report No. 1-68340-44(1998)03, Laboratory of Computer Architecture and Digital Techniques, Delft University of Technology, Delft, The Netherlands.

Lai Y, Liu JX, Tang ZT, et al., 2023. ChiPFormer: transferable chip placement via offline decision transformer. Proc 40th Int Conf on Machine Learning, article 757.

Lee TD, McNamee LP, 1989. Structure optimization in logic schematic generation. IEEE Int Conf on Computer-Aided Design Digest of Technical Papers, p.330-333. https://doi.org/10.1109/ICCAD.1989.76964

Lee TD, McNamee LP, 1992. Aesthetic routing for transistor schematics. IEEE/ACM Int Conf on Computer-Aided Design, p.35-38. https://doi.org/10.1109/ICCAD.1992.279400

Liao HG, Zhang WT, Dong XL, et al., 2020. A deep reinforcement learning approach for global routing. J Mech Des, 142(6):061701. https://doi.org/10.1115/1.4045044

Lin YB, Pan DZ, Ren HX, et al., 2020. DREAMPlace 2.0: open-source GPU-accelerated global and detailed placement for large-scale VLSI designs. China Semiconductor Technology Int Conf, p.1-4. https://doi.org/10.1109/CSTIC49141.2020.9282573

Lin YB, Jiang ZX, Gu JQ, et al., 2021. DREAMPlace: deep learning toolkit-enabled GPU acceleration for modern VLSI placement. IEEE Trans Comput Aided Des Integr Circuits Syst, 40(4):748-761. https://doi.org/10.1109/TCAD.2020.3003843

Liou GH, Wang SH, Su YY, et al., 2018. Classifying analog and digital circuits with machine learning techniques toward mixed-signal design automation. 15th Int Conf on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design, p.173-176. https://doi.org/10.1109/SMACD.2018.8434884

Liu H, Li C, He SB, et al., 2022. Simulated annealing particle swarm optimization for a dual-input broadband GaN Doherty like load-modulated balance amplifier design. IEEE Trans Circuits Syst II: Express Briefs, 69(9):3734-3738. https://doi.org/10.1109/TCSII.2022.3173608

Liu LT, Kuo MT, Cheng CK, et al., 1995. A replication cut for two-way partitioning. IEEE Trans Comput Aided Des Integr Circuits Syst, 14(5):623-630. https://doi.org/10.1109/43.384426

Liu ST, Sun Q, Liao PY, et al., 2021. Global placement with deep learning-enabled explicit routability optimization.

Unedited

Design, Automation & Test in Europe Conf & Exhibition, p.1821-1824.
https://doi.org/10.23919/DATE51398.2021.9473959

Mata RC, Azmib S, Daudc R, et al., 2006. Reverse engineering for obsolete single layer printed circuit board (PCB). Int Conf on Computing & Informatics, p.1-7.
https://doi.org/10.1109/ICOCI.2006.5276552

Meng D, Zheng YL, 2022. Circuit partitioning for PCB netlist based on net attributes. Int Conf on Machine Learning and Cybernetics, p.31-36.
https://doi.org/10.1109/ICMLC56445.2022.9941328

Mnih V, Kavukcuoglu K, Silver D, et al., 2013. Playing Atari with deep reinforcement learning.
https://doi.org/10.48550/arXiv.1312.5602

Naveen B, Raghunathan KS, 1993. An automatic netlist-to-schematic generator. IEEE Des Test Comput, 10(1):36-41. https://doi.org/10.1109/54.199803

Oh Y, Kim K, Lee E, et al., 2008. Automatic generation of diagrams for system-on-chip architectures. Int SoC Design Conf, p.I-322-I-325.
https://doi.org/10.1109/SOCDC.2008.4815637

Ou HC, Chien HCC, Chang YW, 2014. Nonuniform multilevel analog routing with matching constraints. IEEE Trans Comput Aided Des Integr Circuits Syst, 33(12):1942-1954.
https://doi.org/10.1109/TCAD.2014.2363394

Rematska G, Bourbakis NG, 2016. A survey on reverse engineering of technical diagrams. 7th Int Conf on Information, Intelligence, Systems & Applications, p.1-8. https://doi.org/10.1109/IISA.2016.7785372

Roy R, Raiman J, Kant N, et al., 2021. PrefixRL: optimization of parallel prefix circuits using deep reinforcement learning. 58th ACM/IEEE Design Automation Conf, p.853-858.
https://doi.org/10.1109/DAC18074.2021.9586094

Sechen C, 1988. VLSI Placement and Global Routing Using Simulated Annealing. Springer, New York, USA, p.181-243. https://doi.org/10.1007/978-1-4613-1697-8

Sergey G, Daniil Z, Rustam C, 2019. Simulated annealing based placement optimization for reconfigurable systems-on-chip. IEEE Conf of Russian Young Researchers in Electrical and Electronic Engineering, p.1597-1600.
https://doi.org/10.1109/EIConRus.2019.8657251

Settaluri K, Haj-Ali A, Huang QJ, et al., 2020. AutoCkt: deep reinforcement learning of analog circuit designs. Design, Automation & Test in Europe Conf & Exhibition, p.490-495.
https://doi.org/10.23919/DATE48585.2020.9116200

Sharma A, Dyrkolbotn GO, Overlier L, et al., 2022. A state-of-the-art reverse engineering approach for combating hardware security vulnerabilities at the system and PCB level in IoT devices. IEEE Physical Assurance and Inspection of Electronics, p.1-7.
https://doi.org/10.1109/PAINE56030.2022.10014884

Shi YQ, Xue K, Lei S, et al., 2023. Macro placement by wire-mask-guided black-box optimization. Proc 37th Int Conf on Neural Information Processing Systems, article 299.

Steinbrunn M, Moerkotte G, Kemper A, 1997. Heuristic and randomized optimization for the join ordering problem. VLDB J, 6(3):191-208.
https://doi.org/10.1007/s007780050040

Stok L, Koster GP, 1989. From network to artwork. Proc 26th ACM/IEEE Conf on Design Automation Conf, p.686-689. https://doi.org/10.1145/74382.74504

Swinkels GM, Hafer L, 1990. Schematic generation with an expert system. IEEE Trans Comput Aided Des Integr Circuits Syst, 9(12):1289-1306.
https://doi.org/10.1109/43.62774

Tehranipoor M, Koushanfar F, 2010. A survey of hardware Trojan taxonomy and detection. IEEE Des Test Comput, 27(1):10-25.
https://doi.org/10.1109/MDT.2010.7

Vashisht D, Rampal H, Liao HG, et al., 2020. Placement in integrated circuits using cyclic reinforcement learning and simulated annealing.
https://doi.org/10.48550/arXiv.2011.07577

Wang HR, Wang K, Yang JC, et al., 2020. GCN-RL circuit designer: transferable transistor sizing with graph neural networks and reinforcement learning. 57th ACM/IEEE Design Automation Conf, p.1-6.
https://doi.org/10.1109/DAC18072.2020.9218757

Wu CY, Graeb H, Hu J, 2015. A pre-search assisted ILP approach to analog integrated circuit routing. 33rd IEEE Int Conf on Computer Design, p.244-250.
https://doi.org/10.1109/ICCD.2015.7357110

Wu YP, 2009. Novel method of analog circuit schematic synthesis. IEEE 8th Int Conf on ASIC, p.1209-1212.
https://doi.org/10.1109/ASICON.2009.5351191

Yan JC, Lyu XL, Cheng RY, et al., 2022. Towards machine learning for placement and routing in chip design: a methodological overview.
https://doi.org/10.48550/arXiv.2202.13564

Yang J, Qiao K, Shi SH, et al., 2024. AEM-PCB reverser: circuit schematic generation in PCB reverse engineering using reinforcement learning based on aesthetic evaluation metric. IEEE Trans Comput Aided Des Integr Circuits Syst, 43(5):1608-1612.
https://doi.org/10.1109/TCAD.2023.3340869

Zhong RZ, Du XB, Kai SX, et al., 2024a. FlexPlanner: flexible 3D floorplanning via deep reinforcement learning in hybrid action space with multi-modality representation. Proc 38th Int Conf on Neural Information Processing Systems, p.49252-49278.

Zhong RZ, Ye JJ, Tang ZT, et al., 2024b. PreRoutGNN for timing prediction with order preserving partition: global circuit pre-training, local delay learning and attentional cell modeling. Proc 38th AAAI Conf on Artificial Intelligence, p.17087-17095.
https://doi.org/10.1609/aaai.v38i15.29653

Unedited