

MCMCF-Router: Multi-capacity Ordered Escape Routing Algorithms for Grid/Staggered Pin Array

ZHENYI GAO, Computer Science and Technology, Tsinghua University, Beijing, China
SHEQIN DONG, Computer Science and Technology, Tsinghua University, Beijing, China
ZHICONG TANG, Computer Science and Technology, Tsinghua University, Beijing, China
WENJIAN YU, Computer Science and Technology, Tsinghua University, Beijing, China

Ordered escape routing (OER), which means that the pins need to be routed to the boundary of a pin array in a given order, is an important research topic in PCB design. Although OER has been widely investigated, most works assume that the routing capacity between two adjacent pins is just 1 and the structure of the pin array is a grid pin array. In this article, we focus on multi-capacity ordered escape routing (MC-OER) both in grid pin arrays and staggered pin arrays, which means that multiple wires are allowed to pass through between two adjacent pins. We first propose a multi-capacity multi-commodity flow (MC-MCF) model for the MC-OER problem. To accelerate the routing process, MCMCF-Router is proposed. In MCMCF-Router, a wiring resources driven partition strategy is proposed to reduce the problem size, followed by the approach based on routing conflicts. These approaches largely accelerate the MC-MCF model based method and increase the routability with minimal sacrifice on wire length. Experiments on various cases (with up to 525 pins) show that the proposed method achieves 100% routability within reasonable time (<810 seconds). Compared to the state-of-the-art works for single-capacity OER problems, MCMCF-Router performs similarly well or better.

CCS Concepts: • **Hardware → PCB design and layout**; *Wire routing*; Package-level interconnect;

Additional Key Words and Phrases: Ordered escape routing, multi-capacity, multi-commodity flow

ACM Reference Format:
Zhenyi Gao, Sheqin Dong, Zhicong Tang, and Wenjian Yu. 2024. MCMCF-Router: Multi-capacity Ordered Escape Routing Algorithms for Grid/Staggered Pin Array. *ACM Trans. Des. Autom. Electron. Syst.* 29, 6, Article 103 (October 2024), 23 pages. https://doi.org/10.1145/3695253

The preliminary version of this work was presented at the 24th International Symposium on Quality Electronic Design (ISQED) in 2023.

Authors' Contact Information: Zhenyi Gao, Computer Science and Technology, Tsinghua University, Beijing, China; e-mail: gzy22@mails.tsinghua.edu.cn; Sheqin Dong (Corresponding author), Computer Science and Technology, Tsinghua University, Beijing, Beijing, China; e-mail: dongsq@tsinghua.edu.cn; Zhicong Tang, Computer Science and Technology, Tsinghua University, Beijing, Beijing, China; e-mail: tzc21@mails.tsinghua.edu.cn; Wenjian Yu (Corresponding author), Computer Science and Technology, Tsinghua University, Beijing, Beijing, China; e-mail: yu-wj@tsinghua.edu.cn.



This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivs International 4.0 License.

© 2024 Copyright held by the owner/author(s).
ACM 1084-4309/2024/10-ART103
https://doi.org/10.1145/3695253

1 Introduction

With the development of integrated circuits, chiplets, and advanced packaging, the escape routing problems at the packaging or **printed circuit board (PCB)** level have been attracting much more attention [1–3]. Some pioneering works employing rule-based approaches [4, 5] and network flow based approaches [6–8] were proposed to tackle the escape routing problems. However, they do not consider the order of pins at the boundary, following the assumption of **unordered escape routing (UER)** problems. **Ordered escape routing (OER)** is different from UER, and becoming more important for the design of packaging and PCB routing. Given a board with a pin array, OER aims to route the escaped pins to the boundary of the pin array, following the given clockwise order. With the increase of pin number and constraints, the problem becomes difficult. Therefore, an efficient OER method is in great demand.

Traditionally, the pin array is placed to form a **grid pin array (GPA)**, and most of the work considers OER under the GPAs (Figure 1(a) presents an example). Tomioka and Takahashi [9] proposed a method that can solve the problem while satisfying monotonicity. Fang et al. [10] proposed a method that can solve cyclic OER problems based on **integer linear programming (ILP)**. In 2008, Luo an Wong [11] converted the OER problem into the Boolean satisfiability (SAT) problem and solved it using an SAT solver. Later on, a simple partitioning method was proposed to improve the efficiency of the SAT method by exploiting the characteristics of OER problems in industry [12]. In 2010, Yan et al. [13] proposed a more efficient method based on hierarchical bubble sorting. However, it reduces the size of solution space and loses many solutions of the problem. It should be pointed out that none of these works can solve the OER problem with an optimization objective.

In 2015, Satter and Naveed [14] proposed a network flow method that works to maximize the number of escape pins. Then, Jiao and Dong [15] proposed the **min-cost multi-commodity flow (MMCF)** model, which is the first model that minimizes the total wire length for OER problems. In a recent study, Liao and Dong [16] proposed a compact MMCF model with a partitioning strategy to improve efficiency and achieved more than 100X speedup over the method of Jiao and Dong [15]. With the development of machine learning, neural networks have begun to be applied to the field of PCB designs. Chen et al. [17] proposed a convolutional neural network based method, which predicts the probability of the use of grids in the solution of OER problems, reduces the solution space of OER problems, and achieves an acceleration of 4X to 370X over the methods in other works [11, 15]. However, this method is suited to the problems with a fixed pin array size and is therefore short in robustness and generalization. Moreover, the method relies on the SAT method and the MMCF model for constructing the training dataset, which is time consuming. To overcome these drawbacks, Chen et al. [18] proposed a reinforcement learning based method, where a Monte Carlo tree search framework was adopted for design space exploration. It also uses a neural network to guide the Monte Carlo tree search process and achieves up to 19.4X acceleration over the methods of other works [15, 16]. It should be pointed out that these works assume that the routing capacity between two adjacent pins is just 1, which is called the **single-capacity ordered escaped routing (SC-OER)** problem. None of them consider the **multi-capacity ordered escape routing (MC-OER)** problem with an optimization objective (see Figure 1(b) for an example). Recently, Gao et al. [19] extended the MMCF model to the **multi-capacity multi-commodity flow (MC-MCF)** model, which supports multi-capacity between two adjacent pins and can solve large-scale MC-OER problems within reasonable time [19].

With the ever-increasing pin number in the OER problem, the **staggered pin array (SPA)** structure has emerged to accommodate more pins/signals in a given area [5, 20]. The SPA is formed by shifting specific columns (or rows) of the GPA (see Figure 1(c) and (d) for the SC-OER and MC-OER problems in SPAs, respectively). The UER approaches in other works [6–8] for

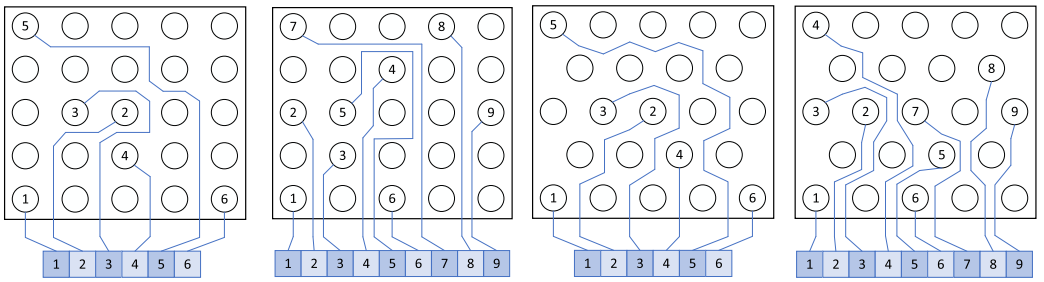


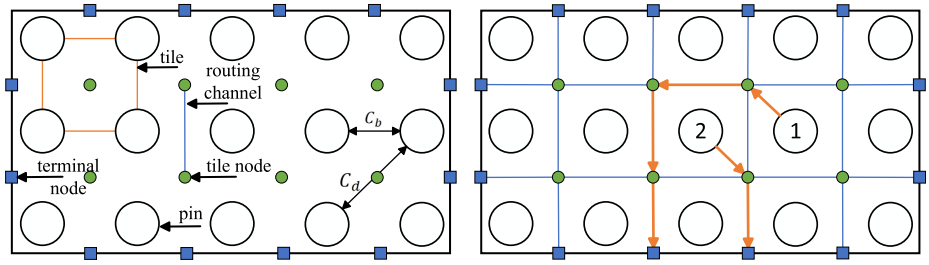
Fig. 1. An illustration of OER in GPAs and SPAs.

SPA structures cannot satisfy the ordering constraint in OER problems. The OER approaches for SPA structures are in demand, but there are few in the literature. Only Jiao and Dong [21] ever applied the MMCF model to the SC-OER problem in SPAs, as an extension of their previous work [15].

In this work, we consider both GPA and SPA structures, and develop efficient algorithms (MCMCF-Router) to tackle the SC-OER and MC-OER problems. The main contributions and results of this work are as follows:

— We propose a model supporting MC-OER—that is, the MC-MCF model—which is based on recursive reconstructions of three main structures in the MMCF model, and apply it to the MC-OER problem with objective of minimizing wire length. The proposed MC-MCF model considers both GPA and SPA structures.
— Based on the MC-MCF model, a set of algorithms (named *MCMCF-Router*) are proposed to tackle OER problems by combining several acceleration approaches. In MCMCF-Router, a **wiring resources driven partition strategy (WRDPS)** is proposed to reduce the problem size, which considers wiring resources in the routing network as the basis for partition. While routing non-escaped pins, MCMCF-Router allows reuse of wiring resources, which generates routing conflicts. These routing conflicts help MCMCF-Router detect which pins are grabbing wiring resources mutually and need to be fine-tuned. These approaches largely accelerate the MC-MCF model based method and increase the routability with minimal sacrifice on wire length.
— Experimental results show that MCMCF-Router achieves 100% routability in various testcases and can solve large-scale MC-OER problems (>500 pins) both in GPAs and SPAs in relatively short runtime (<810 seconds). Compared to the basic MC-MCF model [19], the approaches in MCMCF-Router achieve an average speedup of more than 3,900X. Compared to the state-of-the-art works for SC-OER problems [15, 18, 21], the proposed MCMCF-Router costs similar or less time with similar or shorter wire length.

The rest of this article is organized as follows. The structure of the two pin arrays, the definition of the OER problem, and the properties of the MMCF method are introduced in Section 2. In Section 3, we propose the MC-MCF model for solving the MC-OER problem both in GPAs and SPAs. To handle the challenge of excessive computational cost of the MC-MCF model on large-scale MC-OER problems, MCMCF-Router is proposed in Section 4. Then, experiment results are presented in Section 5. Finally, we draw our conclusion in Section 6. Some preliminary results of this article were presented in our prior work [19]. We extend these results with the consideration of the SPA and the acceleration technique based on routing conflicts, along with relevant experimental results.



(a) Grid pin array. (b) A routing example via routing network.

Fig. 2. Some concepts for GPAs.

2 Background

In this section, we first review the concepts of OER in GPAs and SPAs, and the problem formulation of related OER problems. Then, we present the MMCF problem and its relationship to the OER problem.

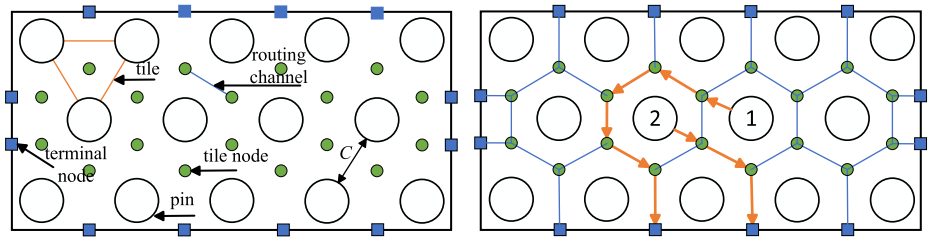
2.1 Problem Formulation

An $m \times n$ GPA consists of m rows and n columns. Each row and each column of pins are aligned. Every four adjacent pins form a square, called a *tile*, and we assume that there is a *tile node* at the center of each tile, as shown in Figure 2(a). The terminal nodes are placed at the boundary of the GPA, serving as exits of the pin array, which are aligned with the tile nodes closest to the boundary. We call the edge between adjacent tile nodes (or between tile nodes and terminal nodes) a *routing channel*. There are two kinds of routing capacities in GPAs: C_b refers to the number of wires that can pass through the routing channel and C_d refers to the number of wires that can pass through the tile node in a diagonal direction. In this work, we assume that $C_b \leq \lfloor \frac{C_d+1}{2} \rfloor$. The routing channels form a routing network, and the escaped pins are routed through the routing network to the boundary of the pin array (see Figure 2(b)).

The SPA has been proposed to accommodate the increasing density of pins on PCBs. Various types of SPAs are available in industrial PCB designs. The following type will be considered in this article. An $m \times n$ SPA consists of m rows. There are n pins in each odd row and n-1 pins in each even row. Any three adjacent pins form an equilateral triangle, which is also called a *tile*, as shown in Figure 3(a) [20]. We also assume that there is a *tile node* at the center of each tile. The definitions of routing channels and terminal nodes are similar to those in GPAs. There is only one kind of routing capacity in SPAs: C refers to the number of wires that can pass through the routing channel. The routing channels also form a routing network, and Figure 3(b) illustrates a routing example via the routing network.

As earlier, the OER problem in GPAs can be defined as follows. Given an $m \times n$ GPA with a sequence of pins $P = \{p_1, p_2, \ldots, p_n\}$, route all of the pins to the boundary of the GPA without wire crossing (*non-crossing constraint*) and violating the two routing capacities (*capacity constraint*), while satisfying the given clockwise at the boundary of the GPA (*ordering constraint*). When C_b=1, the problem can be called an *SC-OER* problem. When C_b is greater than 1, we call the problem an *MC-OER* problem. For the example in Figure 1, C_b=1 in Figure 1(a) and C_b=2 in Figure 1(b).

The definition of OER in SPAs is similar to that in GPAs. If C=1, the problem is called an *SC-OER* problem. When C is greater than 1, the problem can be also called an *MC-OER* problem. For the example in Figure 1, C=1 in Figure 1(c) and C=2 in Figure 1(d). If all pins are required to be escaped on a single side of the pin array, the problem is called a *1-side OER* problem. Similarly, we can



(a) Staggered pin array. (b) A routing example via routing network.

Fig. 3. Some concepts for SPAs.

define the 2-side, 3-side, and 4-side OER problems. The optimization objective can be minimizing the wire length, minimizing the number of wiring layers, or maximizing the number of escaped pins. In this work, we focus on minimizing the wire length.

2.2 The MMCF Model

The MMCF problem is a fundamental optimization problem in graph theory and network flows. In this problem, commodities originating from different sources need to be transported through a network to various destinations, aiming to minimize the overall transportation cost while satisfying capacity constraints on the edges. The MMCF problem has various applications across different domains, such as transportation networks, telecommunication networks, and supply chain management. Figure 4 shows an example of the MMCF problem where the source node is node 1 and the destination node is node 8.

Then, the MMCF problem can be formally defined as follows. Given a directed graph $G = (V, E)$, where V is the set of nodes and E is the set of edges. Each edge $(i, j) \in E$ has a non-negative capacity $u_{i,j}$ and a cost per unit flow $c_{i,j}$. A set of commodities K is transported through the graph, and each commodity k is associated with a source node s_k, a destination node t_k, and a demand d_k. The objective is to find flow variables $f_{i,j}^k$ representing the amount of commodity k flowing from node i to node j while satisfying the demand of commodities and the capacity constraint of directed edges such that the overall transportation cost is minimized.

Now, we formally state the MMCF problem as a linear programming problem as follows:

$$\min_{f_{i,j}^k} \quad \sum_{k\in K,(i,j)\in E} f_{i,j}^k c_{i,j} \tag{1}$$

$$\text{s. t.} \quad \sum_{(i,j)\in E} f_{i,j}^k - \sum_{(j,i)\in E} f_{j,i}^k = \begin{cases} d_k, i = s_k \\ -d_k, i = t_k \\ 0, otherwise \end{cases}, \forall k \in K, \forall i \in V. \tag{2}$$

$$\sum_{k\in K} f_{i,j}^k \le u_{i,j}, \forall (i,j) \in E. \tag{3}$$

In this formulation, constraint (2) ensures that the flow of commodities leaving each node and entering each node is balanced. Constraint (3) ensures that the flow of all commodities through the edge is less than its capacity.

The OER problem can be transformed to the MMCF problem [15]. Additional tile nodes and terminal nodes are added in the pin array to form the routing network, as shown in Figures 2 and 3. Each escaped pin is a source node in the MMCF problem, which ships one unit of commodities to the boundary of the pin array such that the commodity flow can be regarded as the wires

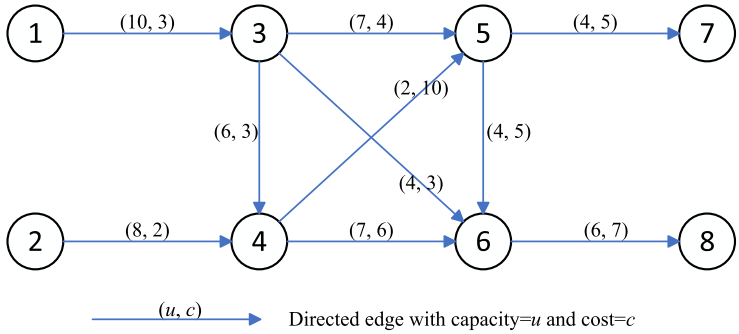


Fig. 4. An example of the MMCF problem where the source node is node 1 and the destination node is node 8.

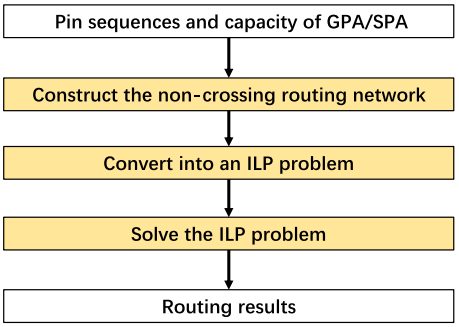


Fig. 5. The workflow of the MC-MCF model.

of escaped pins. In this case, the flow variables $f_{i,j}^{k} \in \{0, 1\}$ so that the MMCF problem can be regarded as an ILP problem. However, the OER problem has more constraints than the MMCF problem—that is, the ordering constraint and the non-crossing constraint. The ordering constraint can be satisfied by setting the destination nodes of commodities in a given clockwise order. The non-crossing constraint can be satisfied by reconstructing the routing network [15] or adding additional constraints to the formulation of the MMCF problem [16]. However, both Jiao and Dong [15] and Liao and Dong [16] only consider the non-crossing constraint in the SC-OER problem. In this article, we will present a universal theorem to guide the construction of routing networks to ensure that the non-crossing constraint is satisfied.

3 The MC-MCF Model for MC-OER

In this section, we propose the MC-MCF model for solving MC-OER problems both in GPAs and SPAs. The workflow of the MC-MCF model is shown in Figure 5. To use the MC-MCF model to solve MC-OER problems while minimizing the wire length, we first construct the non-crossing routing network and convert the MC-OER problem into an ILP problem which aims to minimize the total cost. Then, we use the ILP solver (e.g., `Gurobi` [22]) to solve it and finally convert the ILP solution into the solution of the OER problem. Note that although the MC-MCF model is specially designed for MC-OER problems, it still could solve SC-OER problems. Therefore, in the following, we consider the SC-OER problem to be a simplification of the MC-OER problem.

To ensure that the non-crossing constraint is satisfied, we first present the non-crossing determination theorem. Based on it, the method of constructing a non-crossing multi-capacity commodity

flow graph is then presented, which serves as the routing network of the MC-OER problem. Then, we present the formulation of the ILP problem for the MC-MCF model. Finally, a brief discussion on the computational cost of the MC-MCF model is presented.

3.1 Non-Crossing Determination Theorem

For a commodity flow graph G, we use $F = \{f_1, f_2, \dots\}$ to denote the set of all solutions of the MMCF problem. For any $f \in F$, there is a corresponding solution r of the OER problem. $R = \{r_1, r_2, \dots\}$ denotes the set of all corresponding solutions of the OER problem. $N = \{n_1, n_2, \dots\}$ and $E = \{e_1, e_2, \dots\}$ denote the set of nodes and the set of edges in G, respectively. If there is no crossing wire in any $r \in R$, the commodity flow graph G is called a *non-crossing commodity flow graph*. Before stating the theorem, we first give two lemmas for the node non-crossing and edge non-crossing conditions.

Lemma 1 (Node Non-Crossing). *For any node $n \in N$ in a commodity flow graph G, if its degree is not larger than 3, then there is no crossing wire at n in any corresponding OER solution $r \in R$.*

Proof. The solution of an MMCF problem satisfies the flow balance property (see Equation (2)). We will discuss the commodity flow through node n based on this. If there is no commodity flow through node n, it is clear that there is no crossing wire at n. Therefore, we will only discuss the case where there is some commodity flow through n.

For any $n \in N$ and $r \in R$, when the degree of n is 0, no commodity flow passes through n, thus no wires pass through n in r, so there is no crossing wire.

When the degree of n is 1, at most one commodity passes though n. In this case, one wire passes through n in r at most, so there is no crossing wire.

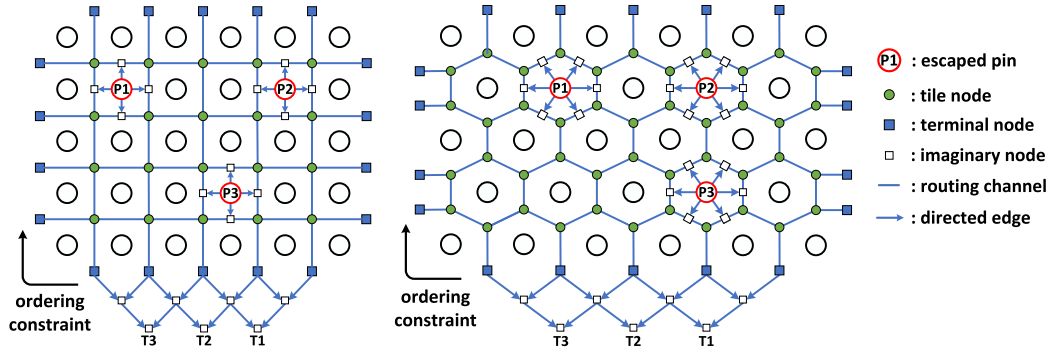
When the degree of n is 2, n_1 and n_2 denote the adjacent nodes of n. When n is a source (or destination) node, we denote the commodity flows out (or in) as $n \rightarrow n_1$ (or $n_1 \rightarrow n$). Another commodity flow passing through edge $n \leftrightarrow n_2$ flows in but cannot flow out. Therefore, no commodity flow passes through edge $n \leftrightarrow n_2$. Then, only one wire passes through n in r. In the case where n is not a source (or destination) node, the commodity flow passes through n is $n_1 \rightarrow n \rightarrow n_2$ (or $n_2 \rightarrow n \rightarrow n_1$). Based on the flow balance property, the two units of commodity flow belong to one commodity, which means that only one wire passes through n in r. Then, there is no crossing wire when the degree of n is 2.

When the degree of n is 3, n_1, n_2, and n_3 denote the adjacent nodes of n. When n is a source (or destination) node, we denote the commodity flow out (or in) as $n \rightarrow n_1$ (or $n_1 \rightarrow n$), and the route $n_2 \leftrightarrow n \leftrightarrow n_3$ can be simplified to $n_2 \leftrightarrow n_3$. Therefore, no crossing wire is generated whether or not there is commodity flow passing through $n_2 \leftrightarrow n_3$. In the case where n is not a source node or destination node, if there is a unit of legal commodity flow passing through n, it will occupy two edges (we take $n \leftrightarrow n_1, n \leftrightarrow n_2$ as an example), and another unit of commodity flow through edge $n \leftrightarrow n_3$ flows in but cannot flow out so that edge $n \leftrightarrow n_3$ is free. Additionally, the commodity flow on $n_1 \leftrightarrow n \leftrightarrow n_2$ also belongs to one commodity. Then, there is only one wire pass through n in r when the degree of n is 3.

Above all, if the degree of n is less than 3, there is no crossing wire. □

Lemma 2 (Edge Non-Crossing). *For any edge $e \in E$ in a commodity flow graph G, if its capacity is 1, then there is no crossing wire at e in any corresponding OER solution $r \in R$.*

Proof. For any $e \in E$ and and $r \in R$, since the capacity of e is 1, at most one unit of commodity flow passes through e, which means that at most one wire passes through e in r. Therefore, there is no crossing wire at e. □



(a) The overall routing network for GPAs. (b) The overall routing network for SPAs.

Fig. 6. The overall routing network.

Theorem 1 (Non-Crossing Determination Theorem). *For a commodity flow graph G, if the degree of any $n \in N$ is less than or equal to 3 and the capacity of any $e \in E$ is 1, the commodity flow graph is a non-crossing commodity flow graph.*

Proof. Combining the statements of Lemmas 1 and 2, and according to the definition of non-crossing commodity flow graph, we can derive this theorem. □

3.2 Non-Crossing Routing Network for MC-OER

In this sub-section, based on Theorem 1, we will extend and generalize for the three main structures (*routing channel, terminal node*, and *tile node*) in the routing network shown in Figures 2(b) and 3(b) to construct non-crossing commodity flow graphs supporting multi-capacity.

3.2.1 The Overall Routing Network. The overall routing networks for GPAs and SPAs are shown in Figure 6(a) and (b), respectively. Each escaped pin is a source node in the MMCF problem, which ships one unit of commodities to the boundary of the pin array. For each escaped pin (see the red nodes in Figure 6(a)) in GPAs, four imaginary nodes are added on its four adjacent routing channels so that each routing channel is divided into two short routing channels. Then, we add four directed edges starting at the escaped pin and ending at the imaginary nodes. Similarly, there are six routing channels adjacent to the escaped pin in SPAs (see the red nodes in Figure 6(b)). Therefore, for each escaped pin in SPAs, six imaginary nodes and six directed edges are added.

The terminal nodes are placed at the boundary of the GPA, serving as exits of the pin array (or destination nodes of commodities). However, the number of terminal nodes is usually greater than that of escaped pins, which means that the number of destination nodes is not equal to that of source nodes in the MMCF problem. Therefore, some directed edges and imaginary nodes are added (see the bottom of Figure 6). This special structure (which is called *ordering transformation* [15]) decreases the number of nodes one by one until it equals the number of escaped pins. In this case, the ordering constraint can be satisfied by setting destination nodes of commodities in a given order. For example, in Figure 6, we set T1, T2, and T3 as destination nodes of escaped pins P1, P2, and P3, respectively. Note that for 2-side, 3-side, and 4-side OER problems, we combine the sequence of terminal nodes on each side into a single sequence, then apply the ordering transformation to the combined sequence of terminal nodes. For example, if the pins in Figure 6 are allowed to escape to the bottom and right sides of the pin array, the terminal nodes on the two sides are combined into a single sequence with nine terminal nodes. Then, the ordering

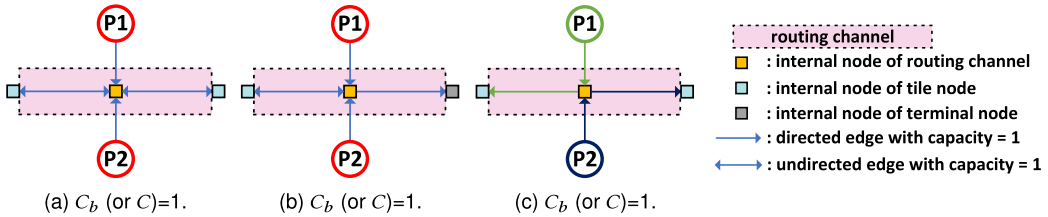


Fig. 7. The construction of single-capacity routing channels.

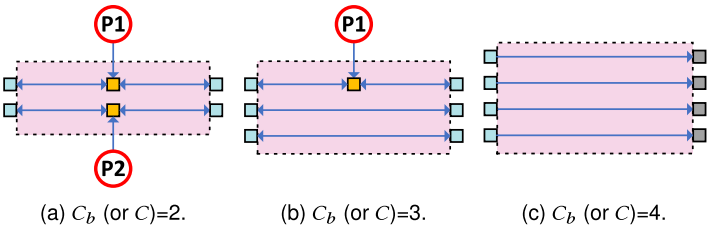


Fig. 8. The construction of multi-capacity routing channels.

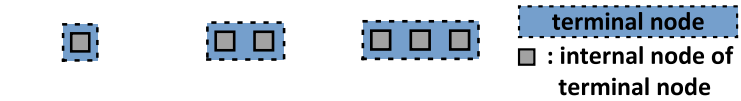
transformation is applied to the nine terminal nodes. Additionally, the non-crossing constraint and capacity constraint can be satisfied by the special structures of tile nodes and routing channels.

3.2.2 The Construction of Routing Channels. The commodity flow is transmitted between tile nodes or between tile nodes and terminal nodes via routing channels. Routing channels are also connected to escaped pins. Notice that routing channels in GPAs and SPAs follow the same construction. The capacity of routing channels is equal to capacity C_b in GPAs (or capacity C in SPAs). There are two types of routing channels. One is a routing channel connecting two tile nodes, and the other is a routing channel connecting a tile node and a terminal node. The single-capacity routing channel is the basic structure of routing channels, and we first introduce the single-capacity routing channel.

The single-capacity routing channel is composed of a internal node and two edges as shown in Figure 7 where all edges have a capacity of 1. When the internal node is connected to an internal node of a tile node, the connected edge is an undirected edge (see Figure 7(a)). When the internal node is connected to an internal node of a terminal node, the connected edge is a directed edge (see Figure 7(b)). Additionally, the escaped pin node is connected to the internal node with a directed edge. Note that as the capacity of the edges is 1, the two units of commodity flow need to flow out via different directions after flowing through the internal node (see Figure 7(c), where the green edges represent the commodity flow of P1 and the blue edges represent the commodity flow of P2). Therefore, although the internal node has a degree of 4, it still can be guaranteed that there is no wire crossing at the internal node.

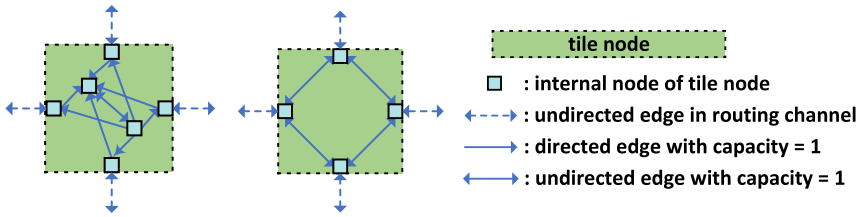
The multi-capacity routing channel with capacity = C_b (or C) consists of C_b (or C) single-capacity routing channels, as shown in Figure 8. The routing channels in Figure 8(a), (b), and (c) have a capacity of 2, 3, and 4, respectively. When the pin node near the routing channel is not an escaped pin node, the internal node is deleted and the two edges are replaced by one edge, as shown in Figure 8(b) and (c). Notice that the escaped pin node only connects the nearest routing channel.

3.2.3 The Construction of Terminal Nodes. The terminal nodes serve as exits of the pin array (or destination nodes of commodities). In the case of multi-capacity, each terminal node contains



(a) C_b (or C)=1. (b) C_b (or C)=2. (c) C_b (or C)=3.

Fig. 9. The construction of terminal nodes.



(a) C_d=1. (b) C_d=2.

Fig. 10. The construction of tile nodes with C_d=1 and 2 in GPAs.

multiple exits, which is determined by the capacity of its connected routing channel (C_b in GPAs or C in SPAs). Therefore, a terminal node consists of several internal nodes, and each internal node serves as an exit of the pin array. The structure of terminal nodes is shown in Figure 9.

3.2.4 The Construction of Tile Nodes. Tile nodes serve as traffic hubs in the routing network and need to complete the transit of commodity flow while satisfying the non-crossing constraint. In GPAs, tile nodes need to complete the transit of commodity flow for four neighboring branches, whereas in SPAs, it is required to complete the transit for three branches. Similar to routing channels, high-capacity tile nodes derive from low-capacity tile nodes. We first introduce the construction of tile nodes in GPAs.

The capacity of tile nodes is equal to capacity C_d in GPAs. The low-capacity (C_d=1 and 2) tile nodes are shown in Figure 10. For the tile node with C_d=1 (see Figure 10(a)), four internal nodes are added to connect the undirected edges in routing channels. The commodity flow passing through the tile node with C_d=1 needs to pass through the middle undirected edge. As C_d=1, at most one unit of commodity flow passes through the tile node. Therefore, even though the construction violates Theorem 1, it still can be guaranteed that there is no wire crossing at the internal nodes. For the tile node with C_d=2 (see Figure 10(b)), all internal nodes follow Theorem 1 and no wire crossing will occur.

The construction of tile nodes is recursive. It is easy to obtain a design with C_d=n by adding nested nodes around a design with C_d=$n-2$, as shown in Figure 11. For example, the tile node with C_d=1 is nested inside the tile node with C_d=3 (see Figure 11(a)), and the tile node with C_d=2 is nested inside the tile node with C_d=4 (see Figure 11(b)). Similarly, we can obtain tile nodes with C_d=5 and 6, as shown in Figure 11(c) and (d), respectively. For tile nodes with even capacity, except for the center sub-structure with C_d=1, all internal nodes follow Theorem 1. Based on the previous discussion of the tile node with C_d=1, it still can be guaranteed that there is no wire crossing at the internal nodes. For tile nodes with odd capacity, all internal nodes follow Theorem 1 so that no wire crossing will occur. We also show the method of connecting tile nodes and routing channels. From Figure 11, it can be seen that before connecting with the routing channel, there are $\lfloor \frac{C_d+1}{2} \rfloor$ internal nodes with degree = 2 in each direction of the tile nodes. According to Theorem 1, only the internal nodes with degree = 2 can be connected to routing channels. Therefore, the tile nodes

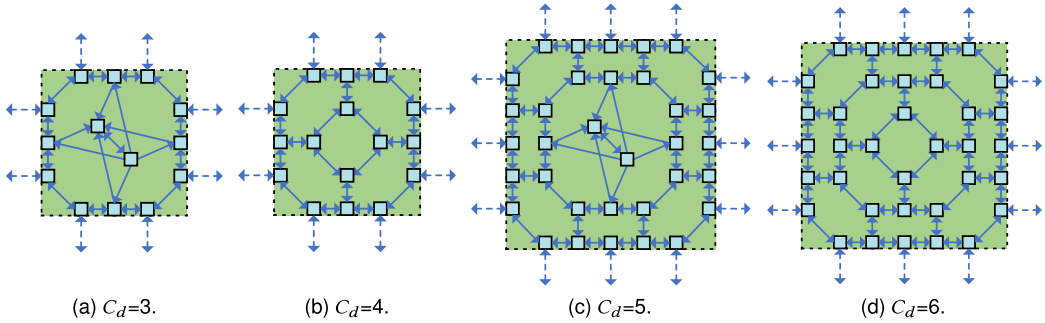


Fig. 11. The construction of tile nodes with C_d=3, 4, 5, and 6 in GPAs.

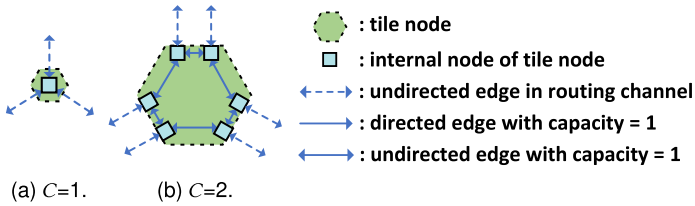


Fig. 12. The construction of tile nodes with C=1 and 2 in SPAs.

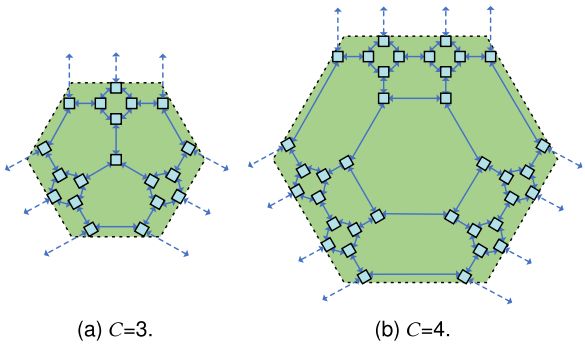


Fig. 13. The construction of tile nodes with C=3 and 4 in SPAs.

with C_d can be connected to the routing channels with $C_b \leq \lfloor \frac{C_d+1}{2} \rfloor$. For example, the tile nodes in Figure 11(a) and (b) are connected to routing channels with C_b=2.

The capacity of tile nodes is equal to capacity C in SPAs. The low-capacity (C=1 and 2) tile nodes are shown in Figure 12. For the tile node with C=1, an internal node is added to connect three undirected edges in routing channels (see Figure 12(a)). For the tile node with C=2, two internal nodes are added for each branch. Then, an undirected edge is added to connect adjacent groups of two internal nodes (see Figure 12(b)). Theorem 1 ensure that there is no wire crossing.

Similar to tile nodes in GPAs, we can easily obtain a tile node with C=n by adding nested nodes around a tile node with C=n − 2, as shown in Figure 13. For example, the tile node with C=1 is nested inside the tile node with C=3 (see Figure 13(a)). Note that all internal nodes of tile nodes have a degree of 3, so it can be ensured that there is no wire crossing.

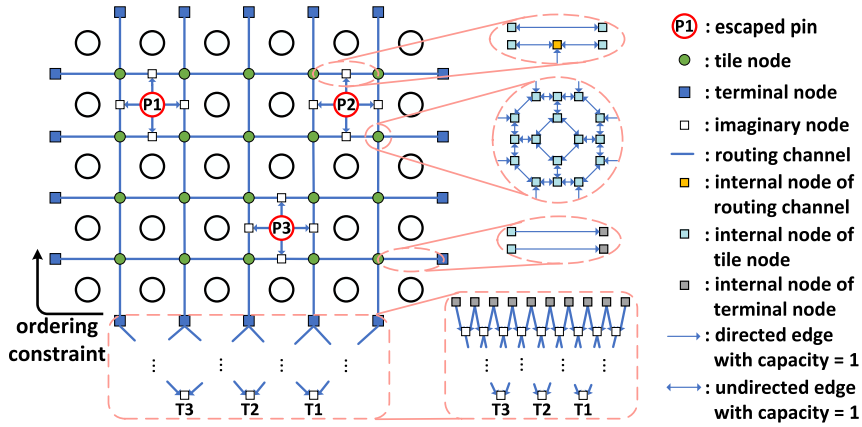


Fig. 14. Illustration of the multi-capacity commodity flow graph for MC-OER in GPAs (C_d=2C_b=4).

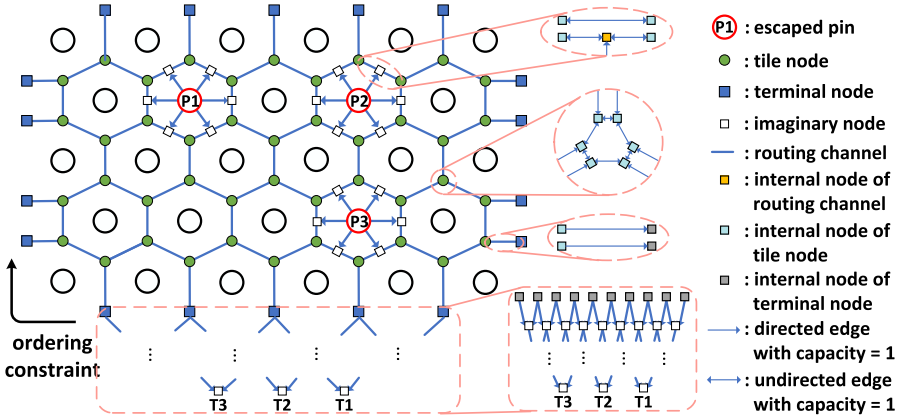


Fig. 15. Illustration of the multi-capacity commodity flow graph for MC-OER in SPAs (C=2).

Based on the preceding extension structures, we propose the MC-MCF graph, which serves as the routing network of MC-OER. For GPAs, we show the detailed structure of the commodity flow graph by taking an example of C_d=2C_b=4 (Figure 14). Note that in the case of multi-capacity, the terminal node contains multiple internal nodes. Therefore, the ordering transformation is applied to internal nodes rather than terminal nodes. Similarly, the MC-MCF graph for MC-OER in SPAs is shown in Figure 15 (C=2).

3.3 ILP Formulation for MC-MCF

Based on the multi-capacity commodity flow graph, the MC-OER problem can be transformed to the MMCF problem. Note that the preceding commodity flow graph is not a directed graph. Every undirected edge is replaced by two directed edges, and we obtain a directed graph. Then, we denote the directed graph as $G = (V, E)$, where V is the set of nodes and E is the set of edges. Each edge $(i, j) \in E$ has a capacity of 1 and a cost per unit flow $c_{i,j}$. A set of commodities P (which serves as the set of pins in the MC-OER problem) is transported through the graph, and each commodity p is associated with a source node s_p (escaped pin nodes), a destination node t_p (the imaginary nodes in the last row of the ordering transformation), and one unit of demand. The objective is to find

flow variables $f_{i,j}^{p}$ representing the amount of commodity p flowing from node i to node j while satisfying the demand of commodities and the capacity constraint of directed edges such that the overall transportation cost (wire length) is minimized. Note that $f_{i,j}^{p} \in \{0, 1\}$.

Now, we state the ILP formulation as follows:

$$\min_{f_{i,j}^{p}} \quad \sum_{p \in P, (i,j) \in E} f_{i,j}^{p} c_{i,j} \tag{4}$$

$$\text{s. t.} \quad \sum_{(i,j) \in E} f_{i,j}^{p} - \sum_{(j,i) \in E} f_{j,i}^{p} = \begin{cases} 1, i = s_p \\ -1, i = t_p \\ 0, otherwise \end{cases}, \forall p \in P, \forall i \in V. \tag{5}$$

$$\sum_{p \in P} f_{i,j}^{p} \leq 1, \forall (i,j) \in E. \tag{6}$$

$$\sum_{p \in P} f_{i,j}^{p} + f_{j,i}^{p} \leq 1, \forall (i,j) \in E. \tag{7}$$

In this formulation, constraint (5) ensures that the flow of commodities leaving each node and entering each node is balanced and the ordering constraint is satisfied. Constraint (6) ensures that the capacity constraint is satisfied. Constraint (7) ensures that the undirected edge is not reused. Additionally, the non-crossing constraint is naturally satisfied by the special structure of tile nodes and routing channels.

3.4 Discussion on Computational Cost

Since the ILP problem has been proven to be NP-complete [23], we discuss the complexity of the MC-MCF model by analyzing the number of constraints in the ILP problem. The ILP problem in Section 3.3 contains $O(|E||P|)$ constraints, where $|E|$ and $|P|$ denote the number of directed edges in the routing network and the number of escaped pins, respectively. An $m \times n$ GPA with C_d=2C_b contains $O(mn)$ routing channels and tile nodes. According to the recursive structure of tile nodes (or routing channels), it is easy to find that each tile node (or routing channel) contains $O(C_b^2)$ nodes and directed edges. Then, an $m \times n$ GPA with C_d=2C_b contains $O(mnC_b^2)$ nodes and directed edges. Therefore, the ILP problem for MC-OER in a GPA contains $O(mnC_b^2|P|)$ constraints. Similarly, the ILP problem for MC-OER in an SPA contains $O(mnC^2|P|)$ constraints. With the increasing scale of the pin array and capacity, the number of constraints in the ILP problem increases rapidly, leading to unacceptable computational cost for solving the problem.

4 MCMCF-Router for MC-OER

In this section, to handle the challenge of excessive computational cost of the MC-MCF model on large-scale MC-OER problems, we propose MCMCF-Router, as shown in Figure 16. MCMCF-Router first divides the given MC-OER problem into several sub-problems of MC-OER via the WRDPS. Then, the MC-MCF model presented in Section 3 is applied to solve all sub-problems of MC-OER. Most of the pins are routed to the boundary of the pin array after the preceding two steps. The pins that are not routed are called *non-escaped pins*. Then, MCMCF-Router will complete the routing of non-escaped pins using remaining wiring resources. For some of the pins that are difficult to route, MCMCF-Router allows reuse of wiring resources, which generates *routing conflicts*. These routing conflicts help MCMCF-Router detect which pins are grabbing wiring resources mutually and need to be fine-tuned. Finally, some of the pins are rerouted to remove routing conflicts.

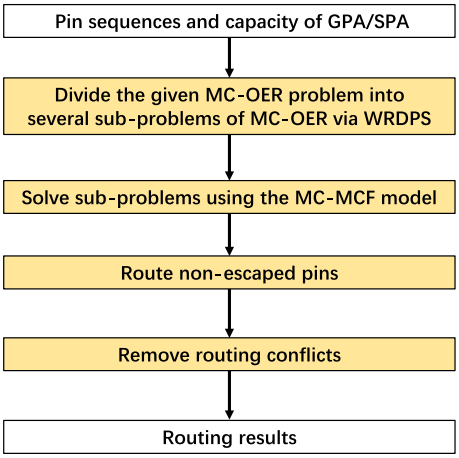


Fig. 16. The workflow of MCMCF-Router for MC-OER.

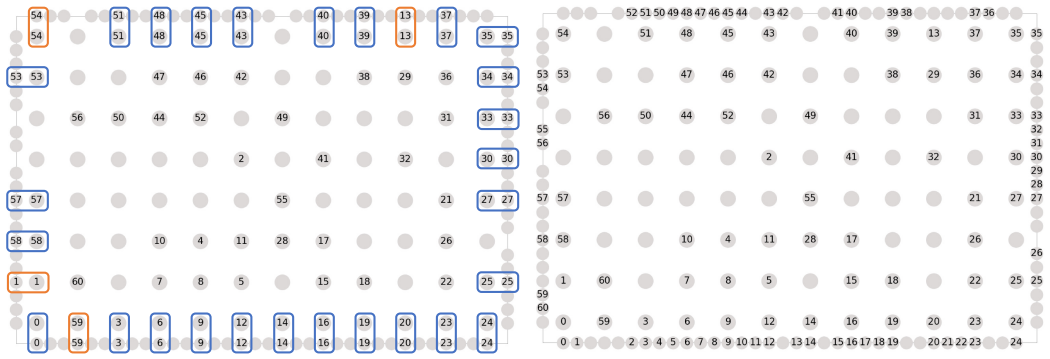
We will introduce how MCMCF-Router accelerates the routing process by using an MC-OER problem in a GPA as an example. For MC-OERs in SPAs, MCMCF-Router also follows the workflow shown in Figure 16. Due to the difference in array structure between SPAs and GPAs, there are differences in some details, which we will point out. It should be noted that since the OER problem is split into several sub-problems, MCMCF-Router can only obtain the optimal solutions of these sub-problems. Therefore, MCMCF-Router is not guaranteed to get the global optimal solution (with minimum wire length). However, the experiments show that MCMCF-Router largely accelerates the MC-MCF model based method and increases the routability with minimal sacrifice on wire length.

4.1 Wiring Resources Driven Partition Strategy

The basic idea of the WRDPS is to reduce the problem size and speed up the routing process by dividing the original MC-OER problem into several sub-problems of MC-OER based on the pin distribution characteristic. In previous works (e.g., [12, 16]), only the clustering characteristic of pins are considered. In the work of Chen et al. [17], a neural network is used to extract high-dimensional routing features. However, the method in their work [17] can only be applied to fixed-scale SC-OERs in GPAs.

In MCMCF-Router, the WRDPS considers wiring resources (edges and nodes) in the routing network as a basis for partition and contains three steps: *assigning escape exits for pins, constructing regions*, and *selecting regions*.

Assigning Escape Exits for Pins. The assigning step focuses on assigning the closest possible escape exits to pins without violating the ordering constraint. Note that internal nodes of terminal nodes serve as escape exits of the routing network. The pins closest to the pin array boundary are easier to route, so assigning them the closest escape exits could save wiring resources. Therefore, the assigning step contains three sub-steps. First, we assign escape exits to pins that are closest to the pin array boundary. Second, illegal allocation pairs are removed (e.g., if the ordering constraint is not satisfied). Third, the preceding allocation divides the pin sequence and escape exit sequence into some sub-sequences, and each sub-sequence of pins corresponds to one sub-sequence of escape exits. For each pair of these sub-sequences, the allocation is executed according to the principle of minimizing the total distance between pins and escape exits.



(a) Substep (1) of assigning escape exits for pins. (b) The result of executing all three substeps.

Fig. 17. An illustration of the approach for assigning escape exits for pins.

We take an MC-OER problem in a GPA as an example (Figure 17), which has a capacity of C_d=2C_b=4, 61 pins, and anti-clockwise ordering constraint. These escaped pins are represented by labeled larger gray nodes while escape exits are represented by smaller gray nodes at the boundaries of the GPA. If an escape exit is assigned, it will be labeled by the pin number. The result of sub-step (1) is shown by the blue (or orange) boxes in Figure 17(a). In sub-step (2), illegal allocation pairs are removed, which are marked by orange boxes. In sub-step (3), the allocation is executed between the corresponding sub-sequences of pins and escape exits. For example, the allocations for pin 0 and pin 3 make a sub-sequence of pins containing two pins (pin 1 and 2) and a sub-sequence of escape exits containing four escape exits. Two out of four escape exits will be assigned to pin 1 and pin 2 according to the principle of minimizing the total distance. Finally, the overall assigning result is shown in Figure 17(b). Afterward, we use A_k to denote the escape exit assigned to pin p_k.

Constructing Regions. Based on the result of the assigning step, the approach of constructing regions is as follows. First, initialize region (R_i^k). In this step, it is expected that the region includes pins that are as close as possible to the center pin (p_k) and numbered close to k. R_i^k is the rectangular area containing exactly the pins numbered in $[k - w_1, k + w_2]$, $0 \le w_1, w_2 \le w_{max} = 2$,

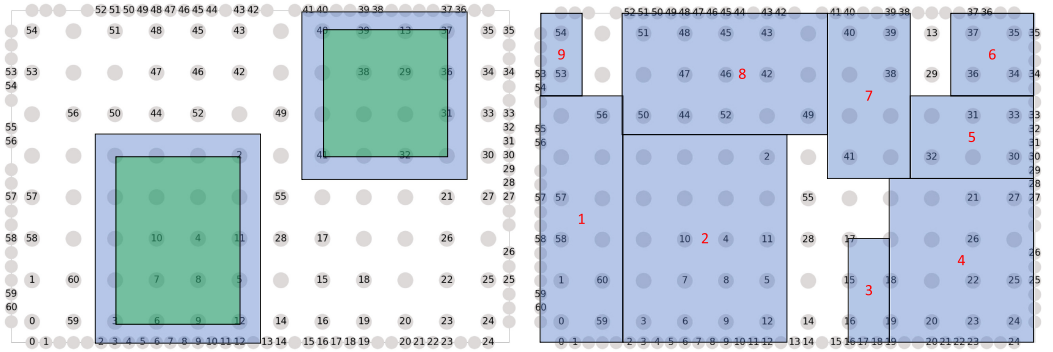
$$w_1 = \max\{w : \text{pins numbered in } [k - w, k] \text{ satisfy the distance constraint}\}, \tag{8}$$

$$w_2 = \max\{w : \text{pins numbered in } [k, k + w] \text{ satisfy the distance constraint}\}. \tag{9}$$

Pin s satisfies the distance constraint if and only if $|x_s - x_k| \le L \wedge |y_s - y_k| \le L$, where (x_k, y_k) denotes the coordinates of pin k. Second, expand region (R_e^k). This step is to provide sufficient wiring resources for pins. R_i^k is expanded in all directions by 0.5 unit lengths to obtain R_e^k. Note that in SPAs, the length of the extension in the vertical direction is $\frac{\sqrt{3}}{4}$. Further, if R_e^k does not include A_k, expand it to include A_k exactly. As shown in Figure 18(a), the green regions indicate R_i^4, R_i^{39} and the blue regions indicate R_e^4, R_e^{39}.

Selecting Regions. After the region construction step, we need to select some regions as the partition result. These regions need to satisfy certain properties: *pin continuity, mutual exclusivity*, and *optimality*. Pin continuity means that the region includes continuous pin numbers.

In Figure 18(a), R_e^4 satisfies pin continuity while R_e^{39} violates pin continuity. Mutual exclusivity indicates that there is no intersection between the selected regions, and optimality means that these selected regions need to include as many pins as possible. We denote $\mathcal{R} = \{R_1, R_2, \ldots, R_l\}(l \le n)$ as regions that satisfy the pin continuity property. The selection result can be determined by



(a) Example of constructing regions. (b) The result of selecting regions.

Fig. 18. An illustration of constructing and selecting regions.

a dynamic programming process. We denote $\mathrm{dp}[a, b]$ as the intermediate selection result for $\{R_a, \ldots, R_b\}(1 \le a \le b \le l)$, and $\mathrm{dp}[a, b]$ can be determined as follows:

$$\mathrm{dp}[a, b] = (\mathrm{dp}[a, m-1] - B_m) \cup R_m \cup (\mathrm{dp}[m+1, b] - B_m), \tag{10}$$

where $B_m(a \le m \le b)$ denote the set of regions in $\mathcal{R}$ that violate mutual exclusivity with R_m and $|R|$ denotes the number of pins in R. The m can be determined as follows:

$$m = \underset{a \le i \le b}{\mathrm{argmax}}\{i : |\mathrm{dp}[a, i-1] - B_i| + |R_i| + |\mathrm{dp}[i+1, b] - B_i|\}. \tag{11}$$

Note that the initial condition for dynamic programming is $\mathrm{dp}[i, i] = \{R_i\}(i = 1, 2, \ldots, l)$ and the final selection result is $\mathrm{dp}[1, l]$. To make the region include more pins, we can repeat the region construction and region selection process several times with different w_{max}. For example, we set $w_{max} = 2$ for the first process and set $w_{max} = 1$for the second process. After the preceding steps, the final result of the WRDPS is shown in Figure 18(b) where 56 out of 61 pins are in the nine regions. The nine regions are labeled from 1 to 9 according to the ordering constraint.

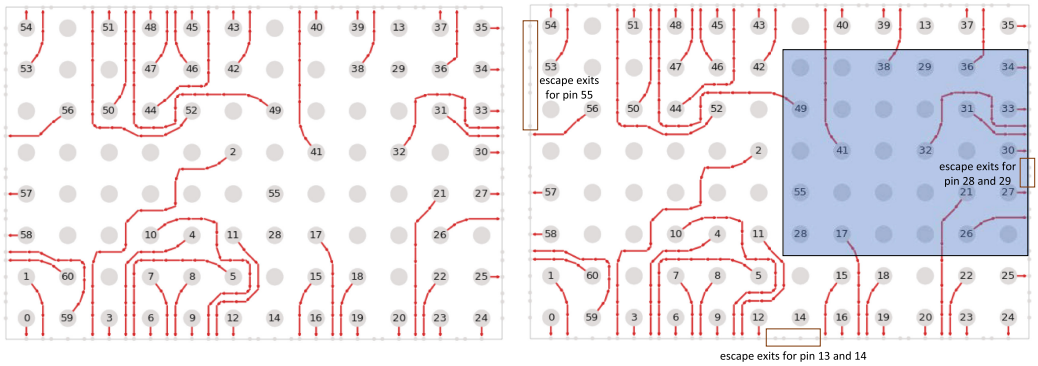
4.2 Solving Sub-Problems Using the MC-MCF Model

The WRDPS generates and filters out a number of disjoint regions. Each region contains pins with continuous pin numbers such that it can be assumed that the region corresponds to a sub-problem of the MC-OER problem. For each sub-problem, we only construct the routing network with the wiring resource in the region and apply the MC-MCF model to it. These sub-problems are solved in a given order according to the ordering constraint. For the MC-OER with anti-clockwise ordering constraint in Figure 18(b), MCMCF-Router will solve these sub-problems sequentially based on the number of regions and obtain the intermediate routing result shown in Figure 19(a). Note that for each sub-problem, a limit of solution time will be set based on the problem size.

After this step, the pins that are not routed are called *non-escaped pins*. There are two kinds of non-escaped pins. One is those pins outside the regions produced by the WRDPS. The other is those pins inside the region but not routed within the time limit. In Figure 19(a), the non-escaped pins are pins 13, 14, 28, 29, and 55. They all are pins outside the regions.

4.3 Routing Non-Escaped Pins

Non-escaped pins will naturally form several sequences of pins according to the order. Additionally, each sequence of non-escaped pins corresponds to an sequence of unused escape exits. For the case in Figure 18, there are three sequences of non-escaped pins (see Figure 19(b)) (i.e., from pin 19 to pin



(a) The intermediate routing result before routing non-escaped pins.

(b) Some concepts during routing non-escaped pins.

Fig. 19. An illustration of routing non-escaped pins.

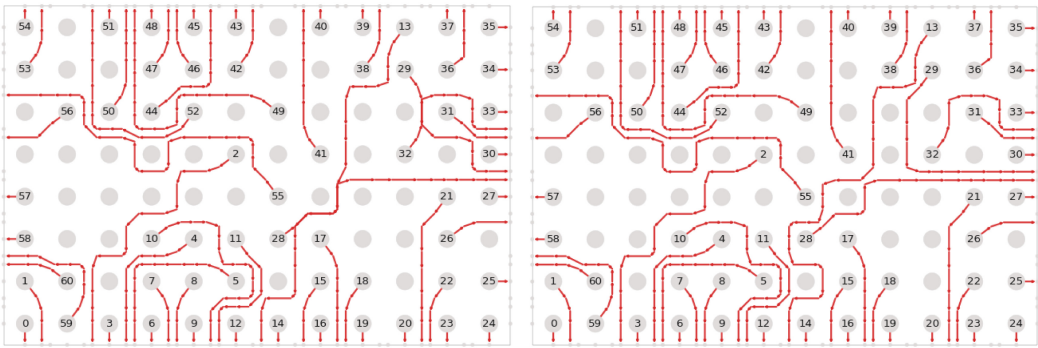
20, from pin 28 to 29 and pin 55). The sequences of unused escape exits are marked by the brown rectangles in Figure 19(b). Each sequence of non-escaped pins is expected to escape through the corresponding sequence of unused escape exits to ensure that the ordering constraint is satisfied. Then, the following approaches will be applied to each sequence of non-escaped pins:

(1) A region that contains exactly the sequence of non-escaped pins and corresponding sequence of unused escape exits is constructed. The region is expanded by 0.5 unit lengths in all directions (in Figure 19(b), the region for pin 28 and 29 is marked by a blue rectangle). Note that in SPAs, the length of the extension in the vertical direction is $\frac{\sqrt{3}}{4}$. Then, we apply the MC-MCF model to the region. In this step, only remaining wiring resources can be used. If the MC-MCF model fails to find a legal solution, we turn to step (2).
(2) The region in step (1) will be expanded by two unit lengths in all directions, which means that we provide more wiring resources to the sequence of non-escaped pins. Note that in SPAs, the length of the extension in the vertical direction is $\sqrt{3}$. Then, we apply the MC-MCF model to the region. In this step, only remaining wiring resources can be used. If the MC-MCF model fails to find a legal solution, we turn to step (3).
(3) As mentioned in Section 4, MCMCF-Router can only obtain the optimal solutions of the sub-problems of OERs. Therefore, in the intermediate routing results, there may be unreasonable wiring resource occupation, which makes some of the sub-problems of OERs unsolvable. To overcome the drawback, in this step, we allow reuse of wiring resources when applying the MC-MCF model to the region in step (2). This step may cause reuse of edges (or nodes) in the routing network, which generates *routing conflicts*. However, these routing conflicts let us know which pins are grabbing wiring resources mutually. The information is helpful to detect which pin needs to be fine-tuned to get a higher routability.

For the case in Figure 19, we apply the preceding approaches for pins 13 and 14, pins 28 and 29, and pin 55 in turn and get the intermediate routing result shown in Figure 20(a). Note that if we first route pins 28 and 29, then route pins 13 and 14, no routing conflicts occur. However, in practice, it is difficult to know which routing order will not cause routing conflicts.

4.4 Removing Routing Conflicts

After routing non-escaped pins, all pins have been routed to the boundary. In the intermediate routing result, if pin a and pin b occupy the same edge (or node) in the routing network, the pin



(a) The routing conflicts after routing non-escaped pins. (b) The routing result after removing routing conflicts.

Fig. 20. An illustration of removing routing conflicts.

pair (a, b) is called a *conflict pair* ($a < b$). For the case in Figure 20(a), there are two conflict pairs—that is, (13, 28) and (29, 32). The goal of removing routing conflicts is to remove all conflict pairs. For each conflict pair (a, b), the following approach is applied:

(1) The routing result of pin a is deleted. Then, the MC-MCF model is applied to route pin a with remaining wiring resources. If the MC-MCF model fails to find a legal solution, we turn to step (2).
(2) The routing result of pin b is deleted. Then, the MC-MCF model is applied to route pin b with remaining wiring resources. If the MC-MCF model fails to find a legal solution, we turn to step (3).
(3) The routing result of pins from a to b is deleted. Then, the MC-MCF model is applied to route the pin sequence from a to b with remaining wiring resources.

For the case in Figure 20(a), the MC-MCF model is applied to route pin 13 to remove the conflict pair (13, 28). Then, the MC-MCF model is applied to route pin 29 to remove the conflict pair (29, 32). Finally, MCMCF-Router obtains the legal routing result shown in Figure 20(b).

5 Experimental Results

We have implemented MCMCF-Router, the MMCF model [15, 21], and the ConDri(P) model [16] in Python. `Gurobi` 9.5 is used as our ILP solver [22]. All experiments are carried out on a machine with an Intel Xeon E5-2630 CPU@2.40 GHz and 512 GB of memory. For the ILP solver, a maximum of eight threads are allowed for `Gurobi`. The experiments are divided into two groups for SC-OER problems and MC-OER problems, respectively. In the SC-OER experiment, MCMCF-Router is compared with the MMCF model [15, 21], the ConDri(P) model [16], and NeuroEscape [18], and the time limit is set to be 1 hour. In the MC-OER experiment, MCMCF-Router is compared with the basic MC-MCF model presented in Section 3 and the method in prior work [19]. The time limit is set to be 168 hours (i.e., 1 week). The testcases are generated randomly, from published works (e.g., [12, 15, 18, 21]) or from industrial PCB design.

5.1 Results on SC-OER Problem

The comparison of MCMCF-Router, the MMCF model, and the ConDri(P) model for SC-OER in GPAs is shown in Table 1. Size(#pin) indicates the grid number of GPAs and the number of pins to be routed. WL denotes the total wire length, which is calculated by the criteria in the work of Liao and Dong [16]. We have tested the cases from other works [12, 15, 18, 21]. Sg7 and Sg8 are

Table 1. Comparison of MCMCF-Router, the MMCF Model, and the ConDri(P) Model for the SC-OER Problem in GPA

Case	Scale(#pin)	(C_b, C_d)	MMCF [15]		ConDri(P) [16]		MCMCF-Router	
			WL	Time (s)	WL	Time (s)	WL	Time (s)
Sg1	8 × 6(24)	(1, 2)	**44**	0.95	**44**	0.42	**44**	**0.35**
Sg2	8 × 8(28)	(1, 2)	**70**	2.28	**70**	0.82	**70**	**0.67**
Sg3	10 × 6(25)	(1, 2)	54.5	1.15	**53.5**	0.34	**53.5**	**0.23**
Sg4	20 × 21(42)	(1, 2)	–	>3,600	168	4.85	**165**	**5.54**
Sg5	25 × 26(60)	(1, 2)	–	>3,600	318	23.9	**311**	**22.7**
Sg6	50 × 50(130)	(1, 2)	–	>3,600	**1,160**	80.7	1167	**57.6**
Sg7	20 × 20(53)	(1, 2)	–	>3,600	312.5	103	**311.5**	**77.8**
Sg8	20 × 20(58)	(1, 2)	–	>3,600	317	10.8	**317**	**6.96**
Sg9	30 × 31(70)	(1, 2)	–	>3,600	480	20.5	**477**	**4.32**
Sg10	75 × 75(151)	(1, 2)	–	>3,600	–	>3,600	**2,102.5**	**182**

Cases Sg3 through Sg6 are the same as cases p1 through p4 in prior work [19].

from Luo and Wong [12]; Sg3 and Sg9 are from Jiao and Dong [15]; Sg4, Sg5, and Sg6 are also from Jiao and Dong [21]; and Sg10 is from Chen et al. [18]. Notice that the dataset of Chen et al. [18] is not available to us. So we just reproduce the case Sg10 based on Figure 5 in their work [18]. For these SC-OER problems, our method reaches 100% routability in the given time limit, whereas the ConDri(P) model [16] completes only nine cases and the MMCF model [15] completes only three cases. Notice that these testcases are originally for SC-OER, so the method in the work of Liao and Dong [16] has achieved the solution close to optimal. In addition, as the partition strategy in the work of Liao and Dong [16] is different from that in this work, the former might produce more optimal result than MCMCF-Router (e.g., on case Sg6). Overall, although the proposed MCMCF-Router has the unique advantage of solving MC-OER problems, it performs similarly as well as or better than the algorithm in the work of Liao and Dong [16] for SC-OER problems. The runtime is shorter, whereas the total wire length of result is similar or even shorter.

The proposed method is also compared with the most recent NeuroEscape [18]. As its source codes and dataset are not available, this comparison is only for case Sg10 and based on the experimental results given in the work of Chen et al. [18]. We see that NeuroEscape completes the routing of Sg10 with wire length of 2,108 in 445 seconds, from their work [18]. With the comparison to the results of MCMCF-Router in Table 1 (time: 182 seconds, wire length: 2,102.5), it is revealed that the proposed method is faster than NeuroEscape with shorter wire length, at least on this case.

Among the state-of-the-art works for SC-OER, only the MMCF [21] model can solve SC-OER problems in SPAs. Therefore, for SC-OER in SPAs, we compare MCMCF-Router with the MMCF model, with the results shown in Table 2. Note that Ss3 and Ss4 are selected from the work of Jiao and Dong [21]. From the table, we can see that our method reaches 100% routability in the given time limit, whereas the MMCF model completes only five cases. For all test cases, MCMCF-Router achieves shorter runtime with equal or shorter wire length compared to the MMCF model.

5.2 Results on the MC-OER Problem

The results of MC-OER with the proposed methods are listed in Tables 3 and 4, where MC-MCF$_0$ denotes a direct implementation of the MC-MCF model in Section 3, and MC-MCF denotes the method presented in prior work [19]. #Node denotes the number of nodes in the routing network of the MC-MCF model. In MCMCF-Router, the given MC-OER is divided into several sub-problems

Table 2. Comparison of MCMCF-Router and the MMCF Model for the SC-OER Problem in SPA

Case	Scale(#pin)	C	MMCF [21]		MCMCF-Router	
			WL	Time (s)	WL	Time (s)
Ss1	5 × 5(16)	1	**32.5**	0.42	**32.5**	**0.30**
Ss2	5 × 5(9)	1	**14.5**	0.46	**14.5**	**0.45**
Ss3	8 × 7(24)	1	**67**	2.98	**67**	**0.13**
Ss4	15 × 11(36)	1	**180**	35.4	**180**	**0.33**
Ss5	33 × 33(92)	1	465.5	2,802	**461.5**	**1.07**
Ss6	33 × 33(98)	1	–	>3,600	**515**	**1.42**
Ss7	44 × 43(132)	1	–	>3,600	**946**	**1.81**
Ss8	42 × 41(126)	1	–	>3,600	**869**	**9.86**
Ss9	54 × 53(162)	1	–	>3,600	**1154**	**22.9**
Ss10	52 × 55(156)	1	–	>3,600	**1,238.5**	**50.8**

Table 3. Comparison of MCMCF-Router and the Original Methods of the MC-MCF Model for the MC-OER Problem in GPA

Case	Scale(#pin)	(C_b, C_d)	MC-MCF$_0$			MC-MCF [19]		MCMCF-Router				
			WL	Time	#Node	WL	Time	WL	Time	#Node(#Num)	#AvgNode	Sp.
Mg1	10 × 5(28)	(2, 4)	**37**	20.0s	2,365	**37**	0.76s	**37**	**0.64s**	1,543(3)	309	31.3X
Mg2	8 × 8(43)	(2, 4)	**98.5**	3198s	1,142	**98.5**	183s	**98.5**	**38.9s**	2,763(3)	345	82.2X
Mg3	10 × 10(32)	(2, 4)	**90**	38.2h	3,399	**90**	158s	**90**	**44.0s**	4,316(5)	863	3,125X
Mg4	23 × 24(93)	(2, 4)	342.5	168h	38,406	338.5	206s	**334.5**	**83.8s**	16,930(21)	806	7,217X
Mg5	24 × 24(93)	(2, 4)	421.5	168h	40,758	412.5	1,171s	**407.5**	**178s**	21,286(18)	1,183	3,398X
Mg6	50 × 50(300)	(2, 4)	–	>168h	153,559	1,617	2,032s	**1,545**	**790s**	90,600(86)	1,053	>766X
Mg7	50 × 50(308)	(2, 4)	–	>168h	148,743	1,724	1,388s	**1,707**	**385s**	92,740(85)	1,091	>1,571X
Mg8	30 × 14(90)	(3, 6)	421	168h	16,365	413	160s	**409**	**138s**	41,943(24)	1,748	4,383X
Mg9	30 × 30(158)	(3, 6)	–	>168h	47,615	**641**	360s	643	**159s**	70,467(43)	1,639	>3,804X
Mg10	20 × 20(171)	(4, 8)	472.5	168h	120,864	469.5	746s	**459.5**	**212s**	94,243(42)	2,244	2,853X

Cases Mg1 through Mg10 are the same as cases b1 through b10 in prior work [19].

of MC-OER and the MC-MCF model is applied to each sub-problem. Each time the MC-MCF model is used, we record the number of nodes in the routing network. Then, we can calculate the total number of nodes in these routing networks and the number of sub-problems, which are denoted by #Node(#Num). #AvgNode denotes the average number of nodes in these routing networks. For example, during the routing of case Mg10, MCMCF-Router constructs 42 sub-problems and the total number of nodes in the 42 routing networks is 94,243. Sp. denotes the speedup ratio of MCMCF-Router with respect to the MC-MCF$_0$.

From the tables, we can see that the direct implementation of the MC-MCF model can only solve 7 out of 10 MC-OER problems in GPAs and 5 out of 10 MC-OER problems in SPAs in 168 hours (i.e., 1 week). For relatively large cases, the runtime of the MC-MCF model is unacceptable (>168 hours). For these MC-OER problems, MC-MCF$_0$ finds the optimal solution in Mg1 through Mg3 and Ms1 through Ms4, where the runtime is less than 168 hours. For those cases with a running time of 168 hours, MC-MCF$_0$ only finds the sub-optimal solution. The number of nodes in the routing network validates the complexity analysis in Section 3.4 (see column 6). We also find that, to get an MC-OER solution via `Gurobi` in a relatively short runtime, the number of nodes in a single routing network is required to be less than the level of several thousands. The runtime is also related to the number of escaped pins in the routing network. Compared to case Mg1, case Mg2 has less nodes in the

Table 4. Comparison of MCMCF-Router and the Original Method of the MC-MCF Model for the MC-OER Problem in SPA

Case	Scale(#pin)	C	MC-MCF$_0$			MCMCF-Router				
			WL	Time	#Node	WL	Time	#Node(#Num)	#AvgNode	Sp.
Ms1	5 × 5(18)	2	**59**	193s	585	**59**	**1.14s**	1,645(6)	274	169X
Ms2	19 × 17(102)	2	**328.5**	50.2h	22,791	**328.5**	**32.5s**	10,023(23)	436	5,561X
Ms3	19 × 17(110)	2	**321**	47.2h	20,488	**321**	**6.14s**	8,561(22)	389	27,674X
Ms4	16 × 15(91)	2	**284**	38.2h	15,725	288	**34.4s**	7,281(20)	364	3,998X
Ms5	29 × 27(170)	2	712	168h	43,280	**692**	**113s**	16,656(38)	438	5,352X
Ms6	54 × 53(323)	2	–	>168h	141,315	**1,802**	**493s**	46,156(72)	641	>1,227X
Ms7	73 × 73(525)	2	–	>168h	184,228	**4,765.5**	**180s**	136,654(216)	633	>3,355X
Ms8	73 × 73(518)	2	–	>168h	187,726	**4,921.5**	**366s**	145,616(224)	650	>1,652X
Ms9	33 × 33(254)	3	–	>168h	158,416	**1,772**	**213s**	117,195(68)	1,723	>2,839X
Ms10	53 × 53(434)	3	–	>168h	393,423	**2,840**	**807s**	163,640(133)	1,230	>749X

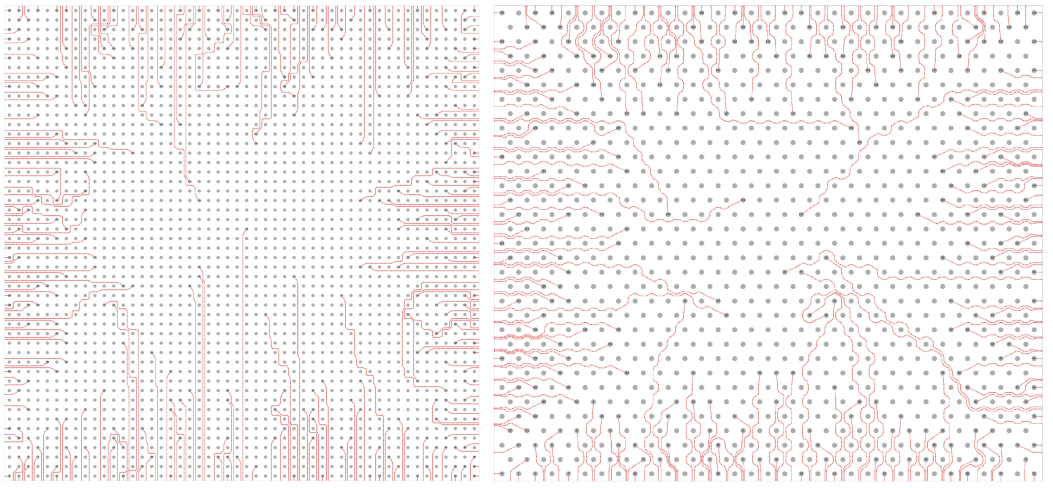
routing network (see lines 1 and 2 of column 6 in Table 3) and more escaped pins (see lines 1 and 2 of column 2 in Table 3). However, the runtime of case Mg2 (3,198 seconds) is much longer than that of case Mg1 (20.0 seconds). Therefore, reducing the number of escaped pins in a single routing network also helps improve the efficiency of routing. MCMCF-Router reduces both the number of nodes (see column "#Node(#Num)") and the number of escaped pins in a single routing network. Therefore, compared to MC-MCF$_0$, MCMCF-Router brings several tens to more than thousands of times of speedup (see the last column).

According to the analysis in Section 4, MCMCF-Router is not guaranteed to find the optimal solution. However, MCMCF-Router still reaches the theoretically optimal wire length for cases Mg1 through Mg3 and Ms1 through Ms3. For the other cases, MCMCF-Router largely accelerates the direct implementation of the MC-MCF model and increases the routability with minimal sacrifice on wire length. Taking case Ms4 as an example, we see that compared to the MC-MCF model, the wire length increases by 1.4% (from 284 to 288) while the runtime decreases by 99.9% (from 38.2 hours to 34.4 seconds). As an extension of prior work [19], the MCMCF-Router proposed in this work has much better performance. From Table 3, we can see that, compared to the MC-MCF proposed in prior work [19], MCMCF-Router brings an acceleration of 3.16X and reduces the wire length by 1.0% on average.

Above all, for these MC-OER problems, MCMCF-Router achieves 100% routability. In addition, its runtime is less than 810 seconds for each case. Furthermore, the results show that the increase of the complexity does not lead to a significant increase in runtime. Even for the largest test cases (Mg6 and Mg7, more than 300 pins in GPAs; Ms7 and Ms8, more than 500 pins in SPAs), the proposed method solves them in 810 seconds. The routing results for cases Mg7 and Ms9 are shown in Figure 21. Notice that these cases have no solution if setting the single-capacity constraint. In other words, the methods in other works [15, 18, 21] cannot solve these MC-OER problems.

6 Conclusion

In this work, an MC-MCF model was proposed for the MC-OER problem with an optimization objective of minimizing the wire length in both the GPA and SPA. To accelerate the routing process, MCMCF-Router was proposed. In MCMCF-Router, a WRDPS was proposed to reduce the problem size, followed by the approach based on routing conflicts. These approaches largely accelerated the MC-MCF model based method and increased the routability with minimal sacrifice on wire length. Although the proposed MCMCF-Router has the unique advantage of solving MC-OER problems,



(a) Case Mg7. (b) Case Ms9.

Fig. 21. The routing results produced by MCMCF-Router.

compared to the state-of-the-art works for SC-OER problems, MCMCF-Router performed similarly well or better. Experiments on various MC-OER cases (with up to 525 pins) showed that the proposed method achieves 100% routability within reasonable time (<810 seconds).

References

[1] Alberto Sangiovanni-Vincentelli, Zheng Liang, Zhe Zhou, and Jiaxi Zhang. 2023. Automated design of chiplets. In *Proceedings of the International Symposium on Physical Design (ISPD'23)*. 1–8.

[2] Ting-Chou Lin, Devon Merrill, Yen-Yi Wu, Chester Holtz, and Chung-Kuan Cheng. 2021. A unified printed circuit board routing algorithm with complicated constraints and differential pairs. In *Proceedings of the Asia and South Pacific Design Automation Conference (ASP-DAC'21)*. 170–175.

[3] Yi-Hao Cheng, Tao-Chun Yu, and Shao-Yun Fang. 2020. Obstacle-avoiding length-matching bus routing considering nonuniform track resources. *IEEE Transactions on Very Large Scale Integration Systems* 28, 8 (2020), 1881–1892.

[4] Michio Horiuchi, Eiji Yoda, and Yukiharu Takeuchi. 2000. Escape routing design to reduce the number of layers in area array packaging. *IEEE Transactions on Advanced Packaging* 23, 4 (2000), 686–691.

[5] Rui Shi and Chung-Kuan Cheng. 2006. Efficient escape routing for hexagonal array of high density I/Os. In *Proceedings of the Design Automation Conference (DAC'06)*. 1003–1008.

[6] Tan Yan and Martin D. F. Wong. 2009. A correct network flow model for escape routing. In *Proceedings of the Design Automation Conference (DAC'09)*. 332–335.

[7] Yuan-Kai Ho, Hsu-Chieh Lee, and Yao-Wen Chang. 2013. Escape routing for staggered-pin-array PCBs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 32, 9 (2013), 1347–1356.

[8] Pei-Ci Wu and Martin D. F. Wong. 2013. Network flow modeling for escape routing on staggered pin arrays. In *Proceedings of the Asia and South Pacific Design Automation Conference (ASP-DAC'13)*. 193–198.

[9] Yoichi Tomioka and Atsushi Takahashi. 2006. Monotonic parallel and orthogonal routing for single-layer ball grid array packages. In *Proceedings of the Asia and South Pacific Design Automation Conference (ASP-DAC'06)*. 642–647.

[10] Jia-Wei Fang, Chin-Hsiung Hsu, and Yao-Wen Chang. 2007. An integer linear programming based routing algorithm for flip-chip design. In *Proceedings of the Design Automation Conference (DAC'07)*. 606–611.

[11] Lijuan Luo and Martin D. F. Wong. 2008. Ordered escape routing based on Boolean satisfiability. In *Proceedings of the Asia and South Pacific Design Automation Conference (ASP-DAC'08)*. 244–249.

[12] Lijuan Luo and Martin D. F. Wong. 2009. On using SAT to ordered escape problems. In *Proceedings of the Asia and South Pacific Design Automation Conference (ASP-DAC'09)*. 594–599.

[13] Jin-Tai Yan, Chung-Wei Ke, and Zhi-Wei Chen. 2010. Ordered escape routing via routability-driven pin assignment. In *Proceedings of the Great Lakes Symposium on VLSI (GLSVLSI'10)*. 417–422.

[14] Kashif Sattar and Anjum Naveed. 2015. Ordered escape routing using network flow and optimization model. In *Proceedings of the Conference on Automation, Robotics, and Applications (ICARA'15)*. 563–568.

[15] Fengxian Jiao and Sheqin Dong. 2016. Ordered escape routing for grid pin array based on min-cost multi-commodity flow. In *Proceedings of the Asia and South Pacific Design Automation Conference (ASP-DAC'16)*. 384–389.
[16] Zhaopo Liao and Sheqin Dong. 2020. A constraint-driven compact model with partition strategy for ordered escape routing. In *Proceedings of the Great Lakes Symposium on VLSI (GLSVLSI'20)*. 393–398.
[17] Zhiyang Chen, Weiqing Ji, Yihao Peng, Datao Chen, Mingyu Liu, and Hailong Yao. 2021. Machine learning based acceleration method for ordered escape routing. In *Proceedings of the Great Lakes Symposium on VLSI (GLSVLSI'21)*. 365–370.
[18] Zhiyang Chen, Tsung-Yi Ho, Ulf Schlichtmann, Datao Chen, Mingyu Liu, Hailong Yao, and Xia Yin. 2023. NeuroEscape: Ordered escape routing via Monte-Carlo tree search and neural network. In *Proceedings of the International Conference on Computer-Aided Design (ICCAD'23)*. 01–09.
[19] Zhenyi Gao, Sheqin Dong, Zhicong Tang, and Wenjian Yu. 2023. MC-MCF: A multi-capacity model for ordered escape routing. In *Proceedings of the International Symposium on Quality Electronic Design (ISQED'23)*. 1–7.
[20] Kan Wang, Huaxi Wang, and Sheqin Dong. 2013. Escape routing of mixed-pattern signals based on staggered-pin-array PCBs. In *Proceedings of the International Symposium on Physical Design (ISPD'13)*. 93–100.
[21] Fengxian Jiao and Sheqin Dong. 2018. Ordered escape routing with consideration of differential pair and blockage. *ACM Transactions on Design Automation of Electronic Systems* 23, 4 (2018), 1–26.
[22] Gurobi Optimization LLC. 2024. Gurobi Optimizer Reference Manual. Retrieved September 19, 2024 from https://www.gurobi.com
[23] Qiang Ma and Martin D. F. Wong. 2012. NP-completeness and an approximation algorithm for rectangle escape problem with application to PCB routing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 31, 9 (2012), 1356–1365.

Received 30 April 2024; revised 26 July 2024; accepted 26 August 2024