論文 / 著書情報
Article / Book Information

Title	UEO Channel Routing Algorithm to Alleviate Local Congestion for Generalized Channels
Authors	Zezhong Wang, Hiroto Nakayama, Masayuki Shimoda, Atsushi Takahashi, Kosuke Yanagidaira, Chikaaki Kodama
Citation	IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E108-A, No. 9, pp. 1241-1250
Pub. date	2025, 9
Copyright	Copyright(C)2025 IEICE

PAPER

UEO Channel Routing Algorithm to Alleviate Local Congestion for Generalized Channels

Zezhong WANG[†a)], **Hiroto NAKAYAMA**[†], ***Nonmembers***, **Masayuki SHIMODA**[†b)], ***Member***, **Atsushi TAKAHASHI**[†c)], ***Fellow***, **Kosuke YANAGIDAIRA**[††], ***Nonmember***, ***and*** **Chikaaki KODAMA**[††], ***Senior Member***

SUMMARY Global routing is one of the most crucial steps for design closure in the physical design of VLSI. This paper proposes a routing algorithm, called UEO algorithm, for generalized channels to achieve a small local congestion which is mainly dedicated to global routing for CMOS circuits designed to match 3D bonding technology. In our generalized channel formulation, due to tight global horizontal routing capacity, the connection of a net is restricted to a single-trunk Steiner tree. Routing algorithms proposed for the generalized channel so far achieve a small total vertical wire length while achieving the minimum number of used tracks, but they do not take a local vertical congestion into account, and the completion of detailed routing may suffer from a large local vertical congestion. The proposed UEO algorithm iteratively determines the assignment of trunks of nets based on the net priority proposed in this paper to achieve a small local vertical congestion. In experiments, it is confirmed that UEO achieves a small local vertical congestion, and that this work contributes to achieve design closure of routing design for 3D VLSI.

key words: *generalized channel, local congestion, parallel wire length*

1. Introduction

Global routing is a critical physical design step to obtain high-performance chips in very-large-scale-integrated circuits (VLSI) design flow [1]. Recently, 3-dimensional (3D) chip manufacturing has been investigated extensively, and one of the key challenges in 3D integration is interconnection technology between vertically stacked 2D chips [2]–[4]. In recent 3D flash memory, bonding pads are used for communication between a complementary metal-oxide-semiconductor (CMOS) chip and a flash memory chip on it, and are regarded as obstacles during the routing design for the CMOS chip. Figure 1 shows a top view of a routing layer of a part of the CMOS chip that includes bonding pads for 3D integration, which are regularly placed as an array inside the routing area.

Due to CMOS chip architecture and design policy of 3D flash memory, the routing area in the CMOS chip contains bottleneck regions of connections in one direction where the routing capacity is tight and the pins of a net are preferred to be connected by a single-trunk Steiner tree if the connection of the net passes a bottleneck region.

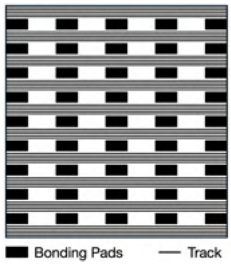


Fig. 1 A routing layer for horizontal wires with tight capacity due to regularly placed bonding pads [5].

Despite the effectiveness of conventional global routing algorithms in scenarios where routing layers contain few or no obstacles, they struggle to efficiently utilize layers with regularly placed obstacles. Techniques such as annealing, negotiation, or machine learning become impractical due to their significant computational demands.

To effectively address challenges found in global routing of CMOS chip for 3D flash memory, the concept of a generalized channel was introduced [6], [7]. This model features horizontal tracks defined in a routing layer for horizontal wires which span the routing area, allows pins of a nets to be placed inside the routing area, and all pins of a net are connected using a single trunk Steiner tree (STST). The vertical wire that connects a pin of a net and the trunk of the net is routed in a routing layer for vertical wires. The routing resource for vertical wires is not so tight compared to the routing resource for horizontal wires, but is limited.

To alleviate the global vertical congestion in generalized channels, algorithms that reduce the total vertical wire length were proposed [5], [6]. Even though these algorithms achieve a small total vertical wire length by using the minimum number of horizontal tracks, they do not take a local vertical congestion into account, and the completion of detailed routing may suffer from a large local vertical congestion. In order to achieve the design closure, a local vertical congestion has to be small enough.

In this paper, we propose UEO (Under, Enclose, Over) algorithm for generalized channels where no constraint except prohibiting an overlap of trunks is enforced. UEO connects pins of each net by a STST, and achieves a small local vertical congestion by using the minimum number of horizontal tracks. It is built upon the Greedy Routing Framework with Guarantee (GRFG) presented in [5]. GRFG ensures the

Manuscript received October 18, 2024.
Manuscript revised February 9, 2025.
Manuscript publicized February 28, 2025.
[†]Institute of Science Tokyo, Tokyo, 152-8550 Japan.
[††]Kioxia Corporation, Yokohama-shi, 247-8585 Japan.
a) E-mail: zezhong@eda.ict.eng.isct.ac.jp
b) E-mail: shimoda@ict.eng.isct.ac.jp
c) E-mail: atsushi@ict.eng.isct.ac.jp
DOI: 10.1587/transfun.2024EAP1144

completion of assignment of trunks of nets to tracks when the required number of tracks is provided and no constraint except overlap is enforced on the assignment. UEO assigns trunks of nets to tracks iteratively under GRFG according to the net priority proposed in this paper. Even though UEO is heuristic and the optimality in terms of the minimum maximum local vertical congestion is not guaranteed, UEO achieves a small local vertical congestion effectively.

This paper is organized as follows: Sect. 2 provides the background and defines the problem. Section 3 introduces the proposed UEO algorithm. Section 4 presents the experimental results, comparing the performance of UEO with other existing methods. Finally, Sect. 5 presents the conclusion.

2. Preliminaries

This section outlines essential concepts, including the problem background, the total parallel length for assessing local vertical congestion, and the definition of the Generalized Channel Routing Problem (GCRP).

2.1 Problem Background

To enable efficient routing on a layer with tight global horizontal capacity due to regularly placed obstacles, the routing layer is modeled as a generalized channel. A generalized channel is characterized by predefined horizontal tracks spanning the entire routing area, ensuring obstacle avoidance and preserving global horizontal routing capacity, effectively eliminating obstacles within the channel. This generalized channel model is similar to the classical channel employed in standard cell designs with a few metal layers. The routing problems associated with these models are termed the *generalized channel routing problem (GCRP)* and the *classical channel routing problem (CCRP)*, respectively [6].

In CCRP where pins are placed on channel boundaries, several algorithms [8]–[19] have been developed. For example, routing for rectangle channel using HV-rule in which each layer uses only either horizontal or vertical wires has been discussed in [8]–[13], [15], while a routing algorithm for L-shaped channel using three-layer has been proposed in [14]. In [16]–[19], routing algorithms for bottleneck channel in which horizontal wires share the same track in adjacent layers to satisfy many connection requirements in a small region have been proposed. However, these algorithms are not directly applicable to GCRP. GCRP allows pins to be placed at any location within the channel.

In a primitive CCRP defined for detailed routing with two metal layers in standard cell design, horizontal and vertical constraints enforced on nets play an important role to define the behavior of algorithms for CCGP. However, in a primitive GCRP, which is defined for global routing for CMOS chips for 3D integration where the vertical routing capacity is not so tight and pin locations are not absolute, no constraint except prohibiting an overlap of horizontal wires is enforced. It is reasonable to assume that a conflict among vertical wires that can be observed in a simple routing representation can be resolved in the followed detailed placement and routing.

To connect multiple pins of a net, the Steiner minimal tree (SMT) is used in a Euclidean plane, while the rectilinear Steiner minimal tree (RSMT) is commonly used in VLSI routing [20]–[22]. However, RSMT typically uses multiple horizontal wires, which may consume the horizontal routing resource redundantly, and is not ideal for layers with limited horizontal routing capacity. Dogleg routing [9] mitigates this by only using horizontal wires obtained by segmenting a horizontal trunk into sections, and is used to reduce the number of tracks needed under vertical constraints in CCRP. Despite its benefits, dogleg routing increases vertical wires and vias, and consumes vertical routing resource much, and makes the following detailed routing difficult. Conversely, the *single-trunk Steiner tree (STST)* uses a single trunk, using the minimum horizontal routing resources and minimizing vertical wires and vias.

It is not difficult to obtain a STST for a single net. While, it is not easy to obtain multiple STSTs without overlapping of horizontal trunks of nets that achieve small congestion even if no constraint except prohibiting an overlap of horizontal trunks is enforced. There are several ways to obtain multiple STSTs without overlapping of horizontal trunks. Left-Edge [8] effectively assigns the trunks of nets to the minimum number of tracks using the leftmost principle. However, it is less effective at alleviating the vertical congestion. BCA [6] is specifically designed to minimize the total vertical length for two-pin nets, and its advanced version, SDG [5], addresses multi-pin nets. Despite these advancements, both algorithms may result in high local vertical congestion. The study in [23]–[25] discusses routing algorithms within GCRP using wires of various widths, whereas this paper focuses on routing with unit-width wires. Although these approaches enhance the assignments, they do not prioritize the minimization of local vertical congestion.

Congestion occurs when the demand for routing resources exceeds the available capacity in a specific area of the chip. Higher congestion leads to potential routing failures. To alleviate congestion, several algorithms have been proposed both in placement and routing phases. During placement, the DLManage model [26] employs heat maps for dynamic congestion management, while the PDT approach [27] focuses on minimizing congestion. In the routing stage, various algorithms [28]–[32] have been developed to reduce local congestion. The PathFinder router [30] is notable for balancing performance and routability, while another algorithm [31] minimizes congestion through estimates. Additionally, a model [32] has been proposed for effective use in integer linear programming (ILP) to reduce congestion. However, these algorithms cannot be directly applied to GCRP.

2.2 Generalized Channel Routing Problem

In GCRP, each net is connected by STST which is charac-

terized by the y-coordinate of the trunk of the net. The primal objective of GCRP in this paper is to minimize the total vertical local congestion while assigning all trunks of nets to given tracks without overlapping. The vertical local congestion is evaluated by the length of the common sections of vertical wires that are close to each other.

GCRP is an assignment task of trunks of nets to tracks where inputs are sets of nets, $N_{\text{in}} = \{n_1, n_2, \ldots, n_m\}$, and horizontal tracks, $T_{\text{in}} = \{t_1, t_2, \ldots, t_k\}$. A net n consists of the set of pins $n = \{p_1, p_2, \ldots, p_{|n|}\}$ which are to be connected by wires. The net n to which pin p belongs is denoted by $n(p)$.

The y-coordinate of track t is represented by $y(t)$. The coordinate of pin p is represented by $(x(p), y(p))$. The minimum and maximum x-coordinate of pins in net n are denoted by $x_{\min}(n) = \min_{p \in n} x(p)$ and $x_{\max}(n) = \max_{p \in n} x(p)$, respectively. The horizontal interval of net n is defined as $I(n) = [x_{\min}(n), x_{\max}(n)]$.

A track assignment $\mathcal{A}$ that assigns the trunk of net n to track t is represented by $\mathcal{A}(n) = t$. A *horizontal constraint* (HC) between two nets n and n' is defined if $I(n) \cap I(n') \neq \emptyset$. The trunks of nets with HC cannot be assigned to the same track.

The vertical wire of pin p of net n when the trunk of n is assigned to track t is represented by $v(p,t)$ where the range of $v(p,t)$ is $[y(p), y(t)]$ if $y(t) > y(p)$, and $[y(t), y(p)]$ otherwise. The length of vertical wire of pin p is $w^y(p,t) = |y(p) - y(t)|$.

The length of a net is the sum of the lengths of horizontal wire and vertical wires of the net. The length of net n when the trunk of n is assigned to track t is formulated as $w(n,t) = w^x(n) + w^y(n,t)$, where $w^x(n) = x_{\max}(n) - x_{\min}(n)$ and $w^y(n,t) = \sum_{p \in n} w^y(p,t)$. The horizontal length w^x of a net is independent of track assignment. The total vertical length (y-length) of nets in a net set N by a track assignment $\mathcal{A}$ is denoted by $w^y(N, \mathcal{A}) = \sum_{n \in N} w^y(n, \mathcal{A}(n))$.

In this paper, the congestion and the local congestion are used to evaluate a track assignment. The congestion of the routing obtained by a track assignment $\mathcal{A}$ is evaluated by the total y-length

$$w^y(N_{\text{in}}, \mathcal{A}) = \sum_{n \in N_{\text{in}}} w^y(n, \mathcal{A}(n)).$$

The local congestion is evaluated by the common section of vertical wires with small distance.

If the vertical wires $v(p,t)$ and $v(p',t')$ belonging to different nets have the common vertical section and the horizontal distance of them is less than the threshold d_{th} (that is, $|x(p) - x(p')| \leq d_{\text{th}}$), then we define that they cause a local congestion by the parallel wire. Otherwise, we regard no local congestion is caused by them. The threshold d_{th} is a user-defined parameter which would be set to suit the design situation.

Let $P(N)$ be the set of nearby pin pairs of net set N in terms of threshold d_{th} which is given as $P(N) = \{(p,p') \mid p \in n \in N, p' \in n' \in N, n \neq n', |x(p) - x(p')| \leq d_{\text{th}}, y(p) \geq y(p')\}$. The length of parallel wire $PL(v(p,t), v(p',t'))$ by

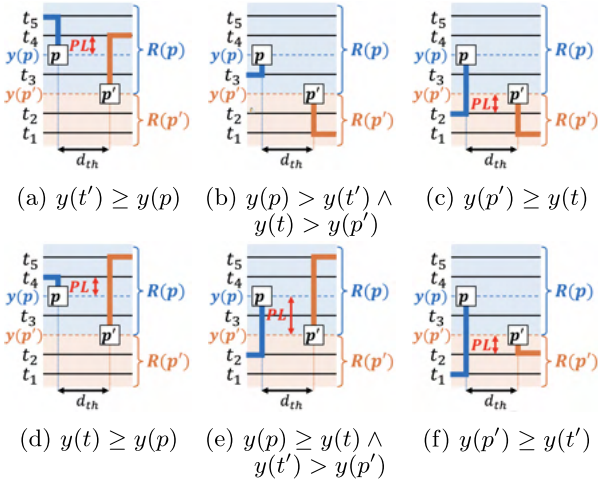


Fig. 2 The length of parallel wire (PL).

$v(p,t)$ and $v(p',t')$ where $(p,p') \in P(N_{\text{in}})$ is defined as follows: In cases that $y(t) > y(t')$,

$$PL(v(p,t), v(p',t')) = \begin{cases} y(t') - y(p) & \text{if } y(t') \geq y(p), \quad \text{(a)} \\ 0 & \text{if } y(p) > y(t') \wedge y(t) > y(p'), \quad \text{(b)} \\ y(p') - y(t) & \text{otherwise (if } y(p') \geq y(t)); \quad \text{(c)} \end{cases}$$

In cases that $y(t') \geq y(t)$,

$$PL(v(p,t), v(p',t')) = \begin{cases} y(t) - y(p) & \text{if } y(t) \geq y(p), \quad \text{(d)} \\ \min(y(p), y(t')) - \max(y(p'), y(t)) & \\ \qquad \text{if } y(p) \geq y(t) \wedge y(t') > y(p'), \quad \text{(e)} \\ y(p') - y(t') & \text{otherwise (if } y(p') \geq y(t')). \quad \text{(f)} \end{cases}$$

Figures 2 illustrate the six cases in terms of the combination of two vertical wires, and the length of each parallel wire (PL) is shown. Cases where $y(t) > y(t')$ are shown in Figs. 2(a), 2(b), and 2(c), corresponding to (a) $y(t') \geq y(p)$, (b) $y(p) > y(t') \wedge y(t) > y(p')$, and (c) $y(p') \geq y(t)$, respectively. Similarly, cases where $y(t') \geq y(t)$ are illustrated in Figs. 2(d), 2(f), and 2(g), corresponding to (d) $y(t) \geq y(p)$, (e) $y(p) > y(t) \wedge y(t') > y(p')$, and (f) $y(p') \geq y(t')$, respectively.

The *total parallel length (TPL)* of N_{in} by track assignment $\mathcal{A}$ is defined as

$$TPL(N_{\text{in}}, \mathcal{A}) = \sum_{(p,p') \in P(N_{\text{in}})} PL(v(p, \mathcal{A}(n(p))), v(p', \mathcal{A}(n(p')))).$$

Note that no parallel wire occurs by $v(p,t)$ and $v(p',t')$ where $y(p) \geq y(p')$ if either $n(p) = n(p')$, $|x(p) - x(p')| > d_{\text{th}}$, or $y(t) > y(t') \wedge y(p) > y(t') \wedge y(t) > y(p')$.

This paper assumes each pin is a point. If accounting for pin width is necessary in practical scenarios, the model

can be adjusted to represent pins as intervals, for example, with corresponding modifications to the algorithm.

The objective of GCRP in this paper is to minimize the total local congestion, which is evaluated by TPL, while assigning all nets N_{in} to the given tracks T_{in}. GCRP to minimize the total local congestion is defined as follows:

GCRP for the total local congestion
Instance: Net set N_{in}, Track set T_{in}.
Output: Assignment $\mathcal{A}: N_{\text{in}} \to T_{\text{in}}$.
Objective: Minimization of $TPL(N_{\text{in}}, \mathcal{A})$.
Constraints:

1. $\mathcal{A}(n) \in T_{\text{in}}, \forall n \in N_{\text{in}}$,
2. $I(n_i) \cap I(n_j) = \emptyset$ if $\mathcal{A}(n_i) = \mathcal{A}(n_j), \forall n_i, n_j \in N_{\text{in}}$.

This problem seems to be NP-hard since there is an example that a simple greedy algorithm fails to obtain an optimal solution, but it is currently unknown to the best of our knowledge. Even though the minimization of TPL does not necessarily reduce an actual local congestion, it is used as the first step to alleviate the local congestion.

In order to complete the track assignment of nets in N_{in} to T_{in}, N_{in} and T_{in} have to meet the density constraint (DC) defined below. We assume that N_{in} and T_{in} meet DC.

Let N ($\subseteq N_{\text{in}}$) and T ($\subseteq T_{\text{in}}$) be a set of unassigned nets during the track assignment procedure and a set of tracks to which no net is assigned so far and to which N will be assigned, respectively. *Density* at x of N is the number of nets in N whose interval contains x and is denoted by $d(x, N) = |\{n \in N \mid x \in I(n)\}|$, and the maximum density of N is defined as $D(N) = \max_x d(x, N)$. *Capacity* of T is the number of tracks in T and is denoted by $|T|$. *Slack* at x of N and T is defined as the difference between capacity and density at x and is represented by $s(x, T, N) = |T| - d(x, N)$.

A feasible track assignment of N to T exists if and only if $D(N) \leq |T|$ is satisfied. We call $D(N) \leq |T|$ the *density constraint* (DC) of N for T. It is said that N meets DC for T if $D(N) \leq |T|$.

Critical zone (CZ) of channel for N and T is the set of x-coordinates of channel where there is no slack and is denoted as $Z(T, N) = \{x \mid s(x, T, N) \leq 0\}$. In any feasible track assignment $\mathcal{A}$ of N to T, a critical zone is covered by trunks, that is, for any $x \in Z(T, N)$ and for any track $t \in T$, there exists the trunk of a net $n \in N$ that contains x at t, that is, $\mathcal{A}(n) = t$ and $x \in I(n)$.

In order to accomplish the routing by determining track assignment track by track in a greedy manner, CZ has to be covered by trunks in each track. Algorithms that follow the Greedy Routing Framework with Guarantee (GRFG) [5] such as Left-Edge [8], BCA [6], SDG [5], and the proposed UEO in this paper cover CZ by trunks, and complete the routing if N_{in} and T_{in} meet DC.

3. UEO Channel Routing Algorithm

In this paper, we propose UEO (Under, Enclose, Over) routing algorithm for generalized channels that achieves a small TPL of vertical wires by using the minimum number of horizontal tracks. It is built upon the *Greedy Routing Framework with Guarantee (GRFG)* [5] which ensures the completion of the assignment of trunks of nets when the required number of tracks is provided. UEO assigns trunks to tracks iteratively according to UEO net priority proposed, and CZ of each track is covered under GRFG. Even though UEO is a heuristic and the optimality in terms of the minimum TPL is not guaranteed, UEO achieves a small TPL effectively.

3.1 Relief-Zone

Parallel wires by vertical wires occur in terms of nearby pin pairs. The occurrence of parallel wires depends on the track assignment of trunks of nets. A parallel wire may not occur if the trunks of nets are assigned to appropriate tracks.

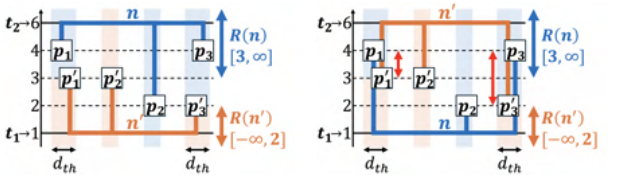
The *relief-zone of a pin* is introduced here to indicate tracks so that the occurrence of parallel wires is expected to be prevented if the trunks of nets are assigned to a track in the relief-zone of each pin. The relief-zone of pin p of a net, which is denoted by $R(p)$, is the range of y-coordinates. It is recommended that the trunk of the net to which p belongs be assigned to a track whose y-coordinate is contained in $R(p)$. The relief-zone of net n, which is denoted by $R(n)$, is defined in terms of the relief-zones of pins of n.

If all the trunks of nets are assigned to a track in its relief-zone, then no parallel wire occurs. Our algorithm assigns the trunk of a net to a track within the relief-zone of the net as much as possible to avoid the occurrence of parallel wires, even though it may be impossible to assign all trunks of nets to tracks in their relief-zones.

There are various ways to define the relief-zone of a pin. In this paper, we design the track assignment algorithm so that the trunks of nets are assigned one by one from the bottom track to the top track, and the relief-zone of pin p is defined to fit the situation that trunks are preferred to be assigned to a lower track. The definition of the relief-zone of pin p used in this paper is formally defined as follows: $R(p) = [R_{\min}(p), R_{\max}(p)]$ where $R_{\min}(p) = \max(\max_{(p,p') \in P(N_{\text{in}})} y(p'), -\infty)$ and $R_{\max}(p) = y(p)$ if $\{p' \mid (p', p) \in P(N_{\text{in}})\}$ is not empty, $R_{\max}(p) = \infty$ otherwise.

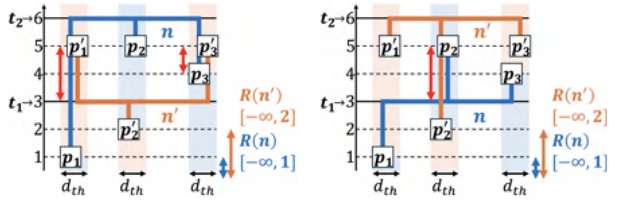
The relief-zone of a pin focuses on the range of track assignment for each individual pin. To extend this concept to the net level, the *relief-zone of a net* is defined. The intersection of relief-zones of pins of a net is defined as the relief-zone of the net if it is not empty. Otherwise, the relief-zone of the net is set to the relief-zone of pin p of the net whose upper boundary is the minimum among the relief-zones of pins of the net. The relief-zone of net n is represented by $R(n) = [R_{\min}(n), R_{\max}(n)]$. There is no guarantee that TPL is minimized when the trunk of a net is assigned to a track in the relief-zone, but the occurrence of parallel wires is expected to be suppressed in terms of at least one pin of the net.

In Figs. 2, the relief-zones of pins p and p' are $R(p) = [y(p'), \infty]$ (shaded blue) and $R(p') = [-\infty, y(p')]$ (shaded



(a) Within the relief-zone (b) Outside the relief-zone

Fig. 3 Relief-zone and track assignments.



(a) Two parallel wires (b) One parallel wire

Fig. 4 Relief-zone and track assignments.

orange), respectively. In Fig. 2(b), the trunks of nets $n(p)$ and $n(p')$ are assigned to tracks t_3 and t_1, respectively, where $y(t_3) \in R(p)$ and $y(t_1) \in R(p')$, and no parallel wire occurs. While, in Figs. 2(a) and (d), and (e), the trunk of $n(p')$ is assigned to a track whose y-coordinate is not in $R(p')$, and a parallel wire occurs. Similarly, in Figs. 2(c) and (f), a parallel wire occurs.

In Figs. 3, the relief-zones of nets n and n' are $R(n) = [3, \infty]$ (blue arrow) and $R(n') = [-\infty, 2]$ (orange arrow), respectively. They are the intersections of the relief-zones of pins of each net. In Fig. 3(a), the trunks of nets n and n' are assigned to tracks in the relief-zone of each net, and no parallel wire occurs. While, in Fig. 3(b), the trunks of nets n and n' are assigned to tracks outside their relief-zones, and parallel wires occur.

In Figs. 4, the relief-zones of nets n and n' are $R(n) = [-\infty, 1]$ (blue arrow) and $R(n') = [-\infty, 2]$ (orange arrow), respectively. They are set to the relief-zone of the pin whose upper boundary is the minimum among the relief-zones of pins of each net since the intersection of them is empty. In such cases, a parallel wire always occurs.

3.2 UEO Net Priority

Algorithms designed under the Greedy Routing Framework with Guarantee (GRFG) [5] assign trunks of nets according to a net priority. Here, the net priority UEO (Under, Enclose, Over) that is intended to achieve small TPL under the condition that trunks are assigned one by one from the bottom track to the top track is defined.

UEO net priority is defined in terms of a track to which the trunks of nets are assigned. Nets are categorized into three types, type-U, type-E, and type-O, and nets belonging to type-U are given the highest priority, followed by type-E and then type-O. In addition, in order to suppress TPL and the total y-length as much as possible, the priority among nets in the same type is defined.

The type of a net is determined based on the relative location of the relief-zone of the net to the track to which it is assigned. A net n is classified for track t as

1. Type-U if $R_{\max}(n) \leq y(t)$,
2. Type-E if $R_{\max}(n) > y(t) \geq R_{\min}(n)$, and
3. Type-O otherwise (if $R_{\min}(n) > y(t)$).

A type-U net has the highest priority. The current track processed is above the relief-zone of the net, and the trunk of the net will be assigned to a track above its relief-zone. That is, it is determined that the parallel wire occurs in terms of the net, and the parallel length in terms of the vertical wire of a pin that defines the relief-zone increases if it is not assigned to the current track. It is better to assign to a track as early as possible to suppress the increase of parallel wire in terms of the relief-zone.

Among type-U nets, nets are prioritized in descending order of the number of pins of the net whose relief-zones are below track t, which is given by:

$$\mathrm{RB}(n, t) = |\{p \in n \mid R_{\max}(p) < y(t)\}|.$$

Then, ties are broken by the symmetric difference (SD) of the net in terms of the current track which is defined as the difference of the number of pins above the track and below the track [5]. The SD for a net n on track t is given by:

$$\mathrm{SD}(n, t) = |\{p \in n \mid y(p) < y(t)\}| - |\{p \in n \mid y(p) > y(t)\}|.$$

Then, ties are broken by the leftmost principal. The second and the third properties are expected to suppress the increase of TPL and the total y-length, respectively.

A type-E net has the second priority. The occurrence of parallel wire in terms of the vertical wire of a pin that defines the relief-zone can be avoided if the trunk is assigned to the current track. However, it is a chance to avoid the occurrence of parallel wire even if the trunk is not assigned to the current track. Ties among type-E nets are broken by the priority by SD, and then by the leftmost principal.

A type-O net has the lowest priority. The current track processed is below the relief-zone of the net, and the parallel wire occurs if the trunk is assigned to the current track. There is a chance to assign the trunk to a track in the relief-zone if it is not assigned to the current track. Among type-O nets, nets are prioritized in ascending order of the number of pins of the net whose relief-zones are above track t, which is given by:

$$\mathrm{RA}(n, t) = |\{p \in n \mid R_{\min}(p) > y(t)\}|.$$

Then, ties are broken by the priority by SD, and then by the leftmost principal.

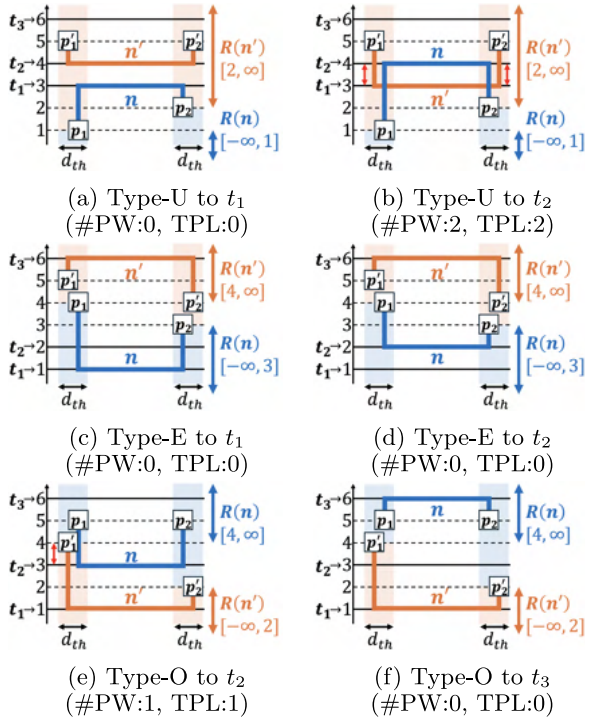
In Figs. 5(a) and (b), the relief-zones of nets n and n' are $R(n) = [-\infty, 1]$ (blue arrow) and $R(n') = [2, \infty]$ (orange arrow), respectively. Nets n and n' are type-U and type-E in terms of tracks t_1 and t_2, respectively. In Fig. 5(a), the trunks

of n and n' are assigned to lower t_1 and upper t_2, respectively, and no parallel wire occurs (#PW=0) and the total parallel length is zero (TPL=0). While, in Fig. 5(b), the trunks of n and n' are assigned to upper t_2 and lower t_1, respectively, and two parallel wires occur (#PW=2) and the total parallel length is 2 (TPL=2). It implies that the trunk of the type-U net is assigned to a lower track is better.

In Figs. 5(c) and (d), the relief-zones of nets n and n' are $R(n) = [-\infty, 3]$ and $R(n') = [4, \infty]$, respectively. Nets n and n' are type-E and type-O in terms of tracks t_1 and t_2, respectively. In Figs. 5(c) and (d), the trunks of n and n' are assigned to tracks in the relief-zone of each net, and no parallel wire occurs. It implies that even if the trunk is not assigned to a lower track, there remains a chance to avoid parallel wires.

In Figs. 5(e) and (f), the relief-zones of nets n and n' are $R(n) = [4, \infty]$ and $R(n') = [-\infty, 2]$, respectively. Nets n and n' are type-O and type-E in terms of tracks t_2 and t_1, respectively. In Fig. 5(e), the trunks of n and n' are assigned to upper t_2 and lower t_1, respectively, and parallel wire occurs. While, in Fig. 5(f), the trunks of n and n' are assigned to upper t_3 and lower t_1, respectively, and no parallel wire occurs. It implies that the trunk of the type-O net is assigned to a higher track is better.

In Figs. 6, the relief-zones of six nets n_1, n_2, n_3, n_4, n_5, and n_6 are represented by arrows in corresponding colors. UEO net priority for track t_1 is $(n_5, n_1, n_4, n_2, n_6, n_3)$ where n_1, n_4, and n_5 are type-U, n_2 and n_6 are type-E, and n_3 is type-O. Among type-U nets, n_5 has a higher priority than others since $\mathtt{RB}(n_5, t_1) = 2 > \mathtt{RB}(n_1, t_1) = \mathtt{RB}(n_4, t_1) = 1$. n_1 has a higher priority than n_4 since $\mathtt{SD}(n_1, t_1) = 2 > \mathtt{SD}(n_4, t_1) = 1$. Among type-E nets, n_2 has a higher priority than n_6 since $x_{\min}(n_2) < x_{\min}(n_6)$ while $\mathtt{SD}(n_2, t_1) = \mathtt{SD}(n_6, t_1) = -2$. UEO net priority for t_2 and t_3 are $(n_4, n_5, n_1, n_6, n_2, n_3)$ and $(n_4, n_5, n_1, n_2, n_6, n_3)$, respectively.



(a) Type-U to t_1 (#PW:0, TPL:0)
(b) Type-U to t_2 (#PW:2, TPL:2)
(c) Type-E to t_1 (#PW:0, TPL:0)
(d) Type-E to t_2 (#PW:0, TPL:0)
(e) Type-O to t_2 (#PW:1, TPL:1)
(f) Type-O to t_3 (#PW:0, TPL:0)

Fig. 5 The type of net n in terms of t_2.

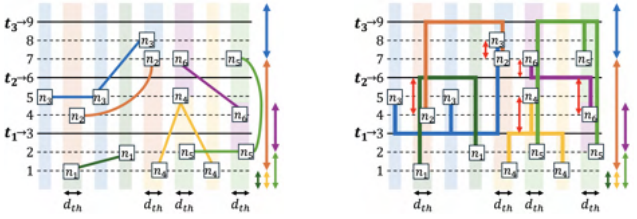
3.3 UEO Channel Routing Algorithm

UEO channel routing algorithm given in Algorithm 1 assigns the trunks of nets to tracks under GRFG to minimize TPL using UEO net priority.

Given a set of nets N_{in} and a set of tracks T_{in}, UEO determines UEO net priority for each track in T_{in}. In Step 5, the function `Top_UEO_Priority` selects the top priority unassigned net for track t_i. In UEO, type-U and type-E nets are assigned according to UEO net priority under GRFG,

Algorithm 1 UEO Channel Routing Algorithm

Require: set of nets N_{in}, set of tracks $T_{\text{in}} = \{t_1, t_2, \ldots, t_m\}$ where $y(t_1) < y(t_2) < \ldots < y(t_m)$
Ensure: track assignment $\mathcal{A}(n)$ for all $n \in N_{\text{in}}$
1: $i \leftarrow 0$, $N \leftarrow N_{\text{in}}$, $T \leftarrow T_{\text{in}}$
2: **while** $N \neq \emptyset$ **do**
3: $T \leftarrow T \setminus \{t_i\}$, $i \leftarrow i + 1$, $x \leftarrow -\infty$, $U \leftarrow N$
4: **while** $U \neq \emptyset$ **do**
5: $n \leftarrow$ `Top_UEO_Priority`(U, t_i), $U \leftarrow U \setminus \{n\}$
6: **if** $(x, \infty) \cap Z(T, N) = \emptyset$ **and** $R_{\min}(n) > y(t_i)$ **then**
7: **break**
8: **end if**
9: **if** $x \leq x_{\min}(n)$ **and** $(x, x_{\min}(n)) \cap Z(T, N) = \emptyset$ **then**
10: $\mathcal{A}(n) \leftarrow t_i$, $x \leftarrow x_{\max}(n)$, $N \leftarrow N \setminus \{n\}$, $U \leftarrow N$
11: **end if**
12: **end while**
13: **end while**



(a) Input with Relief-Zone
(b) Left-Edge (w^y:50, #PW:5, TPL:8)
(c) SDG (w^y:36, #PW:4, TPL:6)
(d) UEO (w^y:39, #PW:1, TPL:3)

Fig. 6 Track assignments.

as earlier assignments generally result in shorter parallel lengths. While, UEO may not assign type-O nets to prevent the occurrence of parallel wire in terms of the net as much as possible. The net assignment process for a given track is terminated in Step 6 when the critical zone (CZ) of the track is fully covered by the nets assigned so far, and no net except type-O can be assigned without violating HC.

Here, the time complexity of UEO is discussed under the assumption that the number of pins of a net is bounded by a constant and that $|T_{\text{in}}| = D(N_{\text{in}})$. Let m be the number of nets.

Step 1 is executed once, and the total time complexity of step 1 is $O(m)$. In step 1, N is set to N_{in}, and nets are removed one by one from N while steps 2 to 13 are executed repeatedly until N becomes empty. The time complexity of the one-time execution of each step, except step 5, is $O(m)$. The time complexity of one-time execution of step 5 in which the top priority net is selected is $O(m \log m)$ since the values used to determine UEO net priority of each net are obtained in $O(1)$, and a sort of nets according to the priority is $O(m \log m)$. A sorting of nets is executed once for each N, that is, $O(m)$ times. Therefore, the total time complexity of step 5 is $O(m^2 \log m)$. During one time execution of the while loop of steps 4 to 12, at least one net is assigned to a track since $|T_{\text{in}}| = D(N_{\text{in}})$ and UEO follows GRFP, and is removed from N. Therefore, the numbers of executions of steps 2 to 13 are $O(m^2)$. Therefore, the total time complexity except step 5 is $O(m^2) \times O(m)$ which is $O(m^3)$. As the total, the time complexity of UEO is $O(m^3)$.

The assignment result using UEO net priority is shown in Fig. 6(d), which reduces TPL to 3, compared to Left-Edge (TPL:8) and SDG (TPL:6). This demonstrates the effectiveness of UEO net priority in minimizing TPL.

4. Experimental Results

The proposed routing algorithm UEO, as well as Left-Edge [8] and SDG [5] for comparisons, is implemented in Java, utilizing OpenJDK 22.0.2. All experiments are conducted on Mac OS, running on a 10-core Apple M1 Pro CPU with 16GB of memory where algorithms operate using a single CPU thread throughout all experiments. The evaluations of the largest benchmark, comprising 10,000 nets, are completed in ten seconds.

The benchmarks used for evaluation were randomly generated due to the lack of available benchmarks from real designs. The benchmark for evaluation consists of eight groups with different numbers of nets with "gm-" prefix to indicate multi-pin configurations, each group consisting of 10 instances distinguished by suffix. Pins are uniformly distributed such that $x(p), y(p) \sim \mathcal{U}(0, 1)$, for all pin p in net n. The number of pins of a net is randomly and evenly distributed between 2 and 10. The number of tracks in T_{in} is equal to $D(N_{\text{in}})$ where y-coordinates of tracks are uniformly distributed within the same range, that is, $\sim \mathcal{U}(0, 1)$. The threshold d_{th} for the benchmark is set as $d_{\text{th}} = 0.012/\#\text{net}$.

In Table 1, the total y-lengths, and in Table 2, the total parallel lengths (TPL) and the average computation times are shown.

In the tables, "LE," "SDG," and "UEO" stand for Left-Edge [8], SDG [5], and UEO proposed in this paper, respectively. "Total Distance" represents the total x-distance $\sum_{n \in N_{\text{in}}} d^x(n)$ and the total y-distance $\sum_{n \in N_{\text{in}}} d^y(n)$ where $d^x(n) = x_{\max}(n) - x_{\min}(n)$ and $d^y(n) = \sum_{p \in n} |y(p) - y(p_{\text{mid}})|$ where p_{mid} is a pin which has $\lceil |n|/2 \rceil$-th largest y-coordinate among pins of net n. "Ratio to Total y-Distance" refers to the ratio of the total y-length of the nets to the total y-distance of the nets. "Total Parallel Length (TPL)" refers to the total parallel length of nets where the average values by Left-Edge are used as a reference. TPL is used to assess local congestion in this paper. These metrics include three values: the average (ave.), the minimum (min), and the maximum (max) for instances in each group.

The results shown in Table 1 indicate that UEO reduces the total y-length, w^y, by approximately 20% compared to Left-Edge and increases it by 16% compared to SDG. The results shown in Table 2 indicate that UEO reduces TPL by about 85% compared to Left-Edge and by 68% compared to SDG, and the computation times of SDG and UEO are larger than LE.

The results given in Tables 1 and 2 are summarized by the graphs shown in Fig. 7(a), (b), and (c). It is confirmed that the variations of the total y-length, TPL, and the computation time decrease as the number of nets increases. In Fig. 7(c), it is observed that the computation times less than one second are dominated by low-order terms, and times where #net 50,000 and 100,000 are added. The computation times of Left-Edge, SDG, and UEO when #net 100,000 are about 13 seconds, 1,900 seconds, and 1,800 seconds, respectively. The time complexities of Left-Edge, SDG, and UEO are estimated based on the differences in average computation times between #net(= m) 50,000 and 100,000, yielding $O(m^{2.37})$, $O(m^{2.29})$, and $O(m^{2.29})$, respectively.

Figures 8 and 9 depict vertical wires of benchmark "gm-6-0" (1,000 nets) obtained by Left-Edge and UEO in red lines. In Figs. 8(a) and (b), all vertical wires are shown. UEO achieves a shorter vertical wire length (w^y:1,653.3) compared to Left-Edge (w^y: 2,045.1). In Figs. 9(a) and (b), vertical wires which form parallel wires are shown. The numbers of parallel wires (#PW) by Left-Edge and UEO are 319 and 120, respectively. The total parallel length (TPL) by UEO is 9.1 while it is 63.8 by Left-Edge. It is confirmed that TPL as well as the number of parallel wires is reduced by UEO effectively.

5. Conclusion

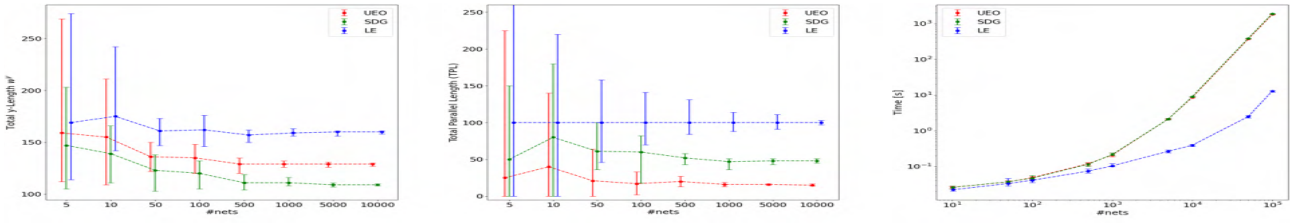
In this paper, we present UEO routing algorithm, specifically designed for routing layers with tight horizontal capacity, which is modeled by generalized channels. UEO provides routing solutions by connecting nets using a single-trunk Steiner tree structure in generalized channels. Experimental results indicate that the total parallel length (TPL) which is used to evaluate the local congestion is reduced by UEO

Table 1 Comparison of Total y-Length (w^y).

Benchmark (#net)	Total Distance x	Total Distance y (%)	Ratio to Total y-Distance: ave.(%) (min(%), max(%)) LE	SDG	UEO
gm-1 (5)	1.5	5.8 (100)	169 (114, 274)	147 (105, 203)	159 (112, 269)
gm-2 (10)	3.2	11.4 (100)	175 (142, 242)	139 (111, 166)	155 (109, 211)
gm-3 (50)	17.5	63.7 (100)	161 (147, 173)	123 (103, 138)	136 (122, 150)
gm-4 (100)	34.0	127.5 (100)	162 (146, 176)	120 (105, 132)	135 (120, 148)
gm-5 (500)	166.6	642.8 (100)	157 (150, 162)	111 (104, 119)	129 (120, 135)
gm-6 (1,000)	333.5	1,285.7 (100)	159 (156, 163)	111 (108, 116)	129 (126, 132)
gm-7 (5,000)	1,665.1	6,390.4 (100)	160 (156, 161)	109 (107, 111)	129 (126, 131)
gm-8 (10,000)	3,331.2	12,765.9 (100)	160 (158, 161)	109 (108, 110)	129 (127, 130)

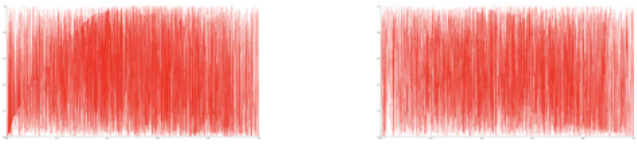
Table 2 Comparisons of Total Parallel Length (TPL) and Computation Time.

Benchmark (#net)	TPL ave. (%) LE	Ratio to TPL ave. by LE: ave.(%) (min(%), max(%)) LE	SDG	UEO	Average Computation Time [s] LE	SDG	UEO
gm-1 (5)	0.4 (100)	100 (0, 360)	50 (0, 150)	25 (0, 225)	0.02	0.03	0.02
gm-2 (10)	0.5 (100)	100 (0, 220)	80 (0, 180)	40 (0, 140)	0.02	0.03	0.03
gm-3 (50)	2.8 (100)	100 (46, 158)	61 (36, 100)	21 (0, 64)	0.03	0.04	0.04
gm-4 (100)	6.0 (100)	100 (70, 141)	60 (18, 82)	17 (2, 33)	0.04	0.05	0.05
gm-5 (500)	30.6 (100)	100 (84, 131)	52 (43, 58)	20 (13, 27)	0.07	0.12	0.12
gm-6 (1,000)	67.3 (100)	100 (88, 114)	47 (36, 51)	16 (13, 19)	0.10	0.21	0.21
gm-7 (5,000)	319.0 (100)	100 (91, 111)	48 (43, 51)	16 (15, 17)	0.26	2.11	2.11
gm-8 (10,000)	634.8 (100)	100 (97, 103)	48 (45, 51)	15 (14, 17)	0.39	8.90	8.59



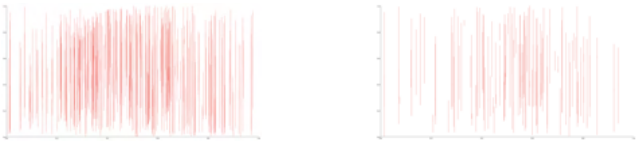
(a) Total y-Length (w^y) (b) Total Parallel Length (TPL) (c) Computation time

Fig. 7 Experimental results.



(a) Left-Edge (w^y:2,045.1) (b) UEO (w^y:1,653.3)

Fig. 8 Vertical wires in gm-6-0 (#net:1,000).



(a) Left-Edge (#PW:319, TPL:63.8) (b) UEO (#PW:120, TPL:9.3)

Fig. 9 Parallel wires in gm-6-0 (#net:1,000).

about 85% from Left-Edge, and about 68% from SDG. However, further improvements of UEO are required to alleviate the actual local congestion and to adapt to a broader range of design constraints. Future work will focus on selecting more diverse benchmarks to better represent real-world data distributions and incorporating wire and pin width variations, such as gap channel optimization [23]–[25], to enhance the algorithm's practical applicability. Ultimately, UEO holds great potential as a fundamental tool for advanced chip designs in a recent 3D integration of VLSI.

Acknowledgments

This work was partially supported by MEXT Initiative to Establish Next-generation Novel Integrated Circuits Centers (X-NICS) Grant Number JPJ011438.

References

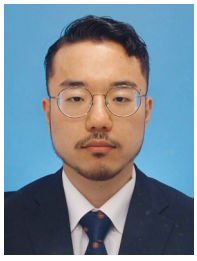
[1] G. Posser, E.F. Young, S. Held, Y.L. Li, and D.Z. Pan, "Challenges and approaches in VLSI routing," Proc. International Symposium on Physical Design (ISPD), pp.185–192, 2022.

[2] M. Sako, T. Nakajima, F. Kono, T. Nakano, M. Fujiu, J. Musha, D. Nakamura, N. Kanagawa, Y. Shimizu, K. Yanagidaira, T. Utsumi, T. Kawano, Y. Hosomura, H. Yabe, M. Kano, H. Sugawara, A.H. Sravan, K. Hayashi, T. Kouchi, and Y. Watanabe, "A 1 Tb 3b/cell 3D-flash memory of more than 17 Gb/mm^2 bit density with 3.2 Gbps interface and 205 MB/s program throughput," 2023 IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits), pp.1–2, June 2023.

[3] S. Kobayashi, K. Tashiro, Y. Minemura, K. Nakagami, K. Arita,

T. Oohashi, K. Funayama, H. Sakai, M. Mushiga, K. Okabe, Y. Kanno, S. Shimizu, E. Fujikura, A. Nakae, K. Yamaguchi, H. Yamawaki, K. Nakajima, and M. Sato, "High peformance 3D flash memory with 3.2 Gbps interface and 205 MB/s program throughput based on CBA (CMOS directly bonded to array) technology," 2023 International Electron Devices Meeting (IEDM), pp.1–4, 2023.
[4] M. Tagami, "CMOS directly bonded to array (CBA) technology for future 3D flash memory," 2023 International Electron Devices Meeting (IEDM), pp.1–4, 2023.
[5] Z. Wang, M. Shimoda, and A. Takahashi, "SDG channel routing to minimize wirelength for generalized channel," IEICE Trans. Fundamentals, vol.E108-A, no.3, pp.500–508, March 2025.
[6] Z. Wang, M. Shimoda, and A. Takahashi, "BCA channel routing to minimize wirelength for generalized channel problem," 2024 IEEE International Symposium on Circuits and Systems (ISCAS), pp.1–5, 2024.
[7] Z. Wang, M. Shimoda, and A. Takahashi, "Single trunk routing problem for generalized channel," IEICE Technical Report, VLD2023-104, 2024.
[8] A. Hashimoto and J. Stevens, "Wire routing by optimizing channel assignment within large apertures," Proc. 8th Design Automation Workshop (DAC), pp.155–169, 1971.
[9] D.N. Deutsch, "A "Dogleg" channel router," Proc. 13th Design Automation Conference (DAC), pp.425–433, 1976.
[10] A.S. LaPaugh, "Algorithms for integrated cricuit layout: An analytic approach," Ph.D. thesis, Massachusetts Institute of Technology, USA, 1980.
[11] T. Yoshimura and E. Kuh, "Efficient algorithms for channel routing," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol.1, no.1, pp.25–35, 1982.
[12] T. Szymanski, "Dogleg channel routing is NP-complete," IEEE Trans. Comput.-Aided Design Integrated Circuits and Systems, vol.4, no.1, pp.31–41, 1985.
[13] T.T. Ho, S. Iyengar, and S.Q. Zheng, "A general greedy channel routing algorithm," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol.10, no.2, pp.204–211, 1991.
[14] A. Takahashi and H. Murata, "Three-layer L-shaped channel routing algorithm," IPSJ Journal, vol.40, no.4, pp.1618–1625, 1999 (in Japanese).
[15] S.S. Sau, A. Pal, T.N. Mandal, A.K. Datta, R.K. Pal, and A. Chaudhuri, "A graph based algorithm to minimize total wire length in vlsi channel routing," 2011 IEEE International Conference on Computer Science and Automation Engineering, pp.61–65, 2011.
[16] K. Taniguchi, S. Tayu, A. Takahashi, Y. Todoroki, and M. Minami, "Bottleneck channel routing to reduce the area of analog VLSI," Proc. 24th Workshop on Synthesis And System Integration of Mixed Information technologies (SASIMI), pp.26–31, 2022.
[17] K. Taniguchi, S. Tayu, A. Takahashi, M. Molongo, M. Minami, and K. Nishioka, "A fast three-layer bottleneck channel track assignment with layout constraints using ILP," Proc. the 25th Workshop on Synthesis And System Integration of Mixed Information Technologies (SASIMI), pp.50–55, 2024.
[18] K. Taniguchi, S. Tayu, A. Takahashi, M. Molongo, M. Minami, and K. Nishioka, "Two-layer bottleneck channel track assignment for analog VLSI," IPSJ Trans. System LSI Design Methodology, vol.17, pp.67–76, 2024.
[19] K. Taniguchi, S. Tayu, A. Takahashi, M. Molongo, M. Minami, and K. Nishioka, "A fast three-layer one-side bottleneck channel routing with layout constraints using ILP," IEICE Trans. Fundamentals, vol.E108-A, no.3, pp.509–516, 2025.
[20] M. Borah, R. Owens, and M. Irwin, "An edge-based heuristic for Steiner routing," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol.13, no.12, pp.1563–1568, 1994.
[21] H. Chen, C. Qiao, F. Zhou, and C.K. Cheng, "Refined single trunk tree: A rectilinear Steiner tree generator for interconnect prediction," Proc. International Workshop on System-Level Interconnect Prediction (SLIP), pp.85–89, 2002.
[22] S.E.D. Lin and D.H. Kim, "Construction of all rectilinear Steiner minimum trees on the Hanan grid and its applications to VLSI design," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol.39, no.6, pp.1165–1176, 2020.
[23] M. Shimoda and A. Takahashi, "Gridless gap channel routing with variable-width wires," IEICE Trans. Fundamentals, vol.E108-A, no.3, pp.517–524, March 2025.
[24] M. Shimoda and A. Takahashi, "Wirelength minimization by gap swap-flip in gridless gap channel routing," Proc. IEEE International SoC Design Conference (ISOCC), 2024.
[25] M. Shimoda and A. Takahashi, "Generalized gap channel routing to minimize wirelength," IPSJ Trans. System LSI Design Methodology, vol.18, pp.2–9, 2025.
[26] H. Szentimrey, A. Al-Hyari, J. Foxcroft, T. Martin, D. Noel, G. Grewal, and S. Areibi, "Machine learning for congestion management and routability prediction within FPGA placement," ACM Trans. Des. Autom. Electron. Syst. (TODAES), vol.25, no.5, pp.1–25, 2020.
[27] K. Lakshmanna, F. Shaik, V.K. Gunjan, N. Singh, G. Kumar, R.M. Shafi, and S. Sarfraz, "Perimeter degree technique for the reduction of routing congestion during placement in physical design of VLSI circuits," Complex., vol.2022, no.11, 8658770, 2022.
[28] M.P. Wong, W.H. Liu, and T.C. Wang, "Negotiation-based track assignment considering local nets," 2016 21st Asia and South Pacific Design Automation Conference (ASP-DAC), pp.378–383, 2016.
[29] F.K. Sun, H. Chen, C.Y. Chen, C.H. Hsu, and Y.W. Chang, "A multi-threaded initial detailed routing algorithm considering global routing guides," 2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), pp.1–7, 2018.
[30] L. McMurchie and C. Ebeling, "Pathfinder: A negotiation-based performance-driven router for FPGAs," Third International ACM Symposium on Field-Programmable Gate Arrays, pp.111–117, 1995.
[31] J. Lou, S. Krishnamoorthy, and H.S. Sheng, "Estimating routing congestion using probabilistic analysis," Proc. 2001 International Symposium on Physical Design, pp.112–117, 2001.
[32] T. Terlaky, A. Vannelli, and H. Zhang, "On routing in VLSI design and communication networks," Discrete Applied Mathematics, vol.156, no.11, pp.2178–2194, 2008.

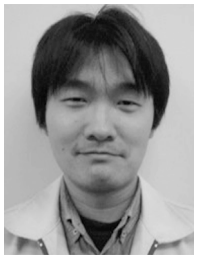
Zezhong Wang received the B.E. degree in Electrical Engineering from North Carolina Agricultural and Technical State University in 2018 and the M.E. degree in Electrical and Computer engineering from Boston University, Boston, USA in 2020. He is currently a Doctoral student with the Department of Information and Communications Engineering of Institute of Science Tokyo. His current research focus on VLSI physical design.



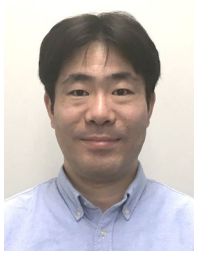
Hiroto Nakayama received the B.E. degree in information and communications engineering from Tokyo Institute of Technology, Tokyo, Japan, in 2024. He is currently with Jitto Inc. from 2024.

Masayuki Shimoda received the B.E., M.E., and D.E. degrees in engineering from Institute of Science Tokyo (formerly, Tokyo Institute of Technology), Tokyo, Japan, in 2018, 2020, and 2024, respectively. He is currently an Assistant Professor with the Department of Information and Communications Engineering of Institute of Science Tokyo. His current research interests include machine learning and VLSI physical design.

Atsushi Takahashi received the B.E., M.E., and D.E. degrees in electrical and electronic engineering from Tokyo Institute of Technology, Tokyo, Japan, in 1989, 1991, and 1996, respectively. He was at the Tokyo Institute of Technology as a Research Associate from 1991 to 1997, and as an Associate Professor from 1997 to 2009 and from 2012 to 2015. He is currently a Professor with Department of Information and Communications Engineering, School of Engineering, Institute of Science Tokyo. His current research interests include VLSI layout design and combinational algorithms. He is a senior member of IEEE and IPSJ, and a member of ACM.



Kosuke Yanagidaira was born in Niigata, Japan, in 1979. He received the B.E. and M.E. degrees in electronic engineering from Chuo University, Tokyo, Japan, in 2003 and 2005, respectively., In 2005, he joined the SoC Research and Development Center, Toshiba Corporation, Yokohama, Japan, where he was engaged in the development of nand flash memories. Since 2019, he has been with Kioxia Corporation, Yokohama, where he has been working on the development of 3-D flash memories.



Chikaaki Kodama received the B.E., M.E., and D.E. degrees in electronic and information engineering from Tokyo University of Agriculture and Technology, Tokyo, Japan, in 1999, 2001, and 2006, respectively. He was with the Development of Custom Computer-Aided Design for SPARC64 Processor, Fujitsu Ltd., Kawasaki, Japan, from 2001 to 2003. He joined Toshiba Microelectronics Corporation, in 2006, and was transferred to Toshiba Corporation, in 2011. He is currently with Kioxia Corporation, Yokohama, Japan. His current research interests include design for manufacturability, especially mask optimization for optical lithography, and very large scale integration layout design. He is a Senior Member of IEEE.