A Bounding Box-based Net Partitioning Method for Double-sided Routing

Fang-Yu Hsu
Tzu-Chuan Lin
Wai-Kei Mak
Ting-Chi Wang
National Tsing Hua University
Hsinchu, Taiwan

ABSTRACT

To improve the power delivery network (PDN) efficiency under the consideration of scaling trends, back-side PDN has been proposed. To well utilize the remaining routing resources on the back side after constructing the PDN, a pioneering work [4] introduced a netlist planning flow capable of distributing a set of signal nets to both sides for routing. However, it failed to consider the routing blockages of the PG network, and the overlapping regions between bounding boxes of pins assigned to the front side and back side, respectively, leading to sub-optimal wirelength. To mitigate these drawbacks, we propose a PG-aware capacity calculation to adjust the bridging cell capacities and the routing capacities for accurate back-side routing resource estimation and a bounding box-based netlist planning approach. Compared to [4], our method resulted in 2% reduction in the average wirelength and 6.4% improvement in the average timing score for the critical nets. Compared to a netlist planning method that leveraged a commercial tool, our approach reduced the average wirelength by 7.4% and improved the average timing score for the critical nets by 22.2%.

ACM Reference Format:
Fang-Yu Hsu, Tzu-Chuan Lin, Wai-Kei Mak, and Ting-Chi Wang. 2024. A Bounding Box-based Net Partitioning Method for Double-sided Routing. In *Great Lakes Symposium on VLSI 2024 (GLSVLSI '24), June 12–14, 2024, Clearwater, FL, USA.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3649476.3658728

1 INTRODUCTION

Traditionally, a power delivery network (PDN) has been implemented on the front-side back-end-of-line (front-side BEOL) of a chip. However, due to technology node scaling, increasing complexity, and congestion of routing, the implementation of a back-side power delivery network (BSPDN) has been proposed [5]. This approach aims to enhance routing and PDN efficiency by separating the PDN from the front-side BEOL to the back-side BEOL.

This work is supported in part by the National Science and Technology Council under grant 111-2221-E-007-119-MY3.

GLSVLSI '24, June 12–14, 2024, Clearwater, FL, USA

ACM ISBN 979-8-4007-0605-9/24/06
https://doi.org/10.1145/3649476.3658728

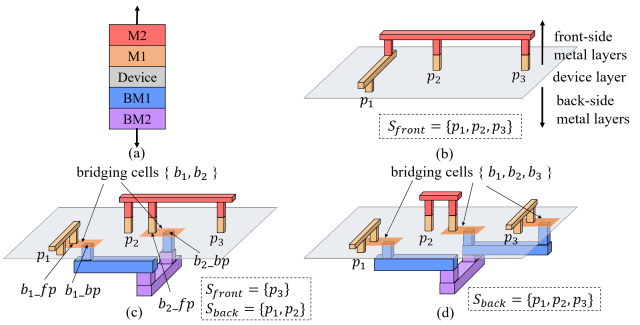

Figure 1: Routing solution of a net: (a) A double-sided layer structure. (b) A single-sided net on the front-side metal layers. (c) A double-sided net with p_1, p_2 using bridging cells. (d) A double-sided net with all pins using bridging cells.

Several studies have demonstrated the effectiveness of BSPDN. Intel has connected BSPDN to the active devices in their Intel 4 finFET process with PowerVia, boosting performance by enabling relaxed front-side metal scaling and reduction in IR drop of a fabricated E-core [3]. ARM has shown that employing BSPDN in high-performance blocks at the A14 node can improve PPA in the design [7]. Moreover, it has been revealed that the concept of BSPDN can extend beyond power delivery to include signal routing [6]. When BSPDN is not densely packed and additional routing resources are available outside the PDN, the back-side metal stacks can also be utilized for signal nets. The back-side metal layers have the potential to offer faster transmission speeds with lower resistance, allowing timing-critical nets to leverage the enhanced performance afforded by these back-side routing resources.

A TSMC patent [2] proposed a double-sided routing structure. The device layer is positioned between the front-side and back-side metal layers, serving as the layer where active devices are formed. An example of a double-sided layer structure is shown in Fig. 1(a), which contains two front-side metal layers M_1, M_2 and two back-side metal layers BM_1 and BM_2. Since the back side of a cell in a conventional cell library does not have back-side pin connections, a bridging cell is created with one front-side pin on M_1 and one back-side pin on BM_1 and is needed to be inserted on the device layer to provide the signal connection between the front-side and back-side metal layers.

Nets routed entirely on either the front or the back side are termed *single-sided nets* (e.g, Fig. 1(b)). Otherwise, they are termed *double-sided nets* (e.g, Fig. 1(c) and (d)). To complete routing, double-sided nets are effectively split into individual nets, each routed on

a single side. For a net, the subset of pins assigned to the front side is denoted as S_{front}, while the subset assigned to the back side is denoted as S_{back}. Fig. 1 illustrates three distinct routing solutions for a net with three pins, p_1, p_2, and p_3, located on M_1. In Fig. 1(b), we present a single-sided net example with $S_{front} = \{p_1, p_2, p_3\}$, where all wire segments are routed on the front-side metal layers. Fig. 1(c) showcases a double-sided net example with wire segments routed on both sides, where $S_{front} = \{p_3\}$ and $S_{back} = \{p_1, p_2\}$. Furthermore, a front-side pin p_1 connects to a front-side pin b_1_fp of a bridging cell b_1, and a back-side pin b_1_bp of b_1 is connected on the back side. Consequently, the original three-pin net is divided into three separate nets: $\{p_1, b_1\_fp\}$, $\{p_2, b_2\_fp, p_3\}$ on the front side, and $\{b_1\_bp, b_2\_bp\}$ on the back side. Fig. 1(d) shows another double-sided net with $S_{back} = \{p_1, p_2, p_3\}$, where all pins are individually connected to different bridging cells, with all back-side pins of these bridging cells connected on the back side. This configuration transforms the original three-pin net into four distinct nets: $\{p_1, b_1\_fp\}$, $\{p_2, b_2\_fp\}$, $\{p_3, b_3\_fp\}$ on the front side, and $\{b_1\_bp, b_2\_bp, b_3\_bp\}$ on the back side.

Currently, no commercial routing tools have the ability to tackle the challenges of double-sided routing. The authors of [4] presented a netlist planning methodology that preserved each net entirely on a single side or converted it into subnets for double-sided routing purpose. This approach allows for using any existing router to conduct separate routing processes on each side. They formulated the netlist planning as a 2D global routing problem to estimate the routing congestion of the netlist planning result. They employed a simple net partitioning method to provide initial partitioning solutions and repartitioned nets that are difficult to route in the modified 2D global router NTHU-Route 2.0 [1] with separate 2D routing grid graphs for the front side and the back side. Finally, separate netlists for the two sides were generated and the location for each newly added bridging cell was determined based on the routing solution. However, [4] did not consider the routing blockages of the PG network, and the overlapping regions between bounding boxes of pins assigned to the front side and back side, respectively, leading to sub-optimal wirelength. Therefore, we propose a PG-aware capacity calculation and an improved net partitioning approach in the netlist planning flow.

2 PREVIOUS WORK

The net partitioning method in [4] consists of three phases: (1) initial partitioning, (2) pin redistribution, and (3) connection point assignment. In the initial partitioning phase, if it is feasible to move all the pins of a critical net to the back side, implying that the Gcell of each front-side pin has enough white space to insert a bridging cell on the layout region, then all pins in the critical net will be assigned to the back side. Otherwise, each front-side (back-side) pin is assigned to the front side (back side) initially. In the pin redistribution phase, it constructs separate congestion maps for each side based on the RUDY value computed by each net [8] to identify potential congestion regions of each Gcell and try to balance the congestion distribution on each side. Finally, a connection point is found to connect each double-sided net between the front side and back side. However, in the pin redistribution phase, it independently reassigns the pins of a net in the congested Gcell to the opposite side, without considering the relation among the pins within the net. This may result in large increases in wirelength.

Fig. 2 shows an example of a six-pin net n. Suppose, all the pins $p_1, p_2, p_3, p_4, p_5, p_6$ in n are feasible to demote to back side (see Fig. 2(a)), which results in S_{back}=$\{p_1, p_2, p_3, p_4, p_5, p_6\}$ (see Fig. 2(b)). During the pin redistribution phase, the congestion regions on each side are identified. Moreover, p_2 and p_3 are found in the congestion regions on the back side. Consequently, p_2 and p_3 would be promoted to the front side which results in $S_{back} = \{p_1, p_4, p_5, p_6\}$ and $S_{front} = \{p_2, p_3\}$ (see Fig. 2(c)). However, its front-side bounding box and its back-side bounding box has a large overlapping region is shown in the red box, which is a cause of sub-optimal wirelength in the subsequent routing stage.

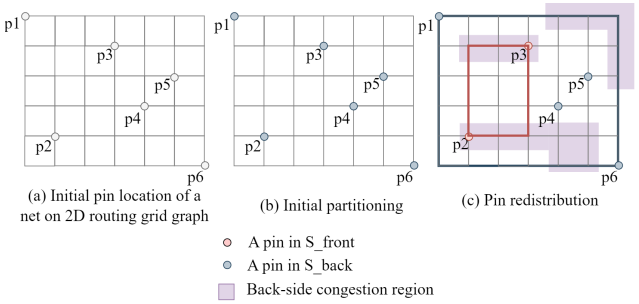

Figure 2: Example of partitioning a net by the method in [4].

3 PROBLEM FORMULATION

Here, we state our assumptions for the netlist planning problem:

- There exists a group of m metal layers on the front side, labeled as $F = \{M_1, \ldots, M_m\}$, and a group of n metal layers on the back side, denoted as $B = \{BM_1, \ldots, BM_n\}$.
- To ensure the manufacturability of a bridging cell's back-side pin on BM_1, BM_1 cannot be used for signal routing.
- IO pins are all located on BM_n [5].
- BSPDN is constructed on $\{BM_2, \ldots, BM_n\}$.
- The clock network is pre-constructed on $\{M_1, \ldots, M_m\}$.
- A specific set of preferred routing layers for the critical nets is represented as $R = B \cup F'$ with $F' \in F$.

We are given a placed netlist N, which comprises two distinct groups of signal nets, the critical nets in N_1 and the non-critical nets in N_2. The netlist planning problem asks to transform netlist N into two separate netlists, a front-side netlist and a back-side netlist, while determining a location for each newly added bridging cell. This problem expects that a high-quality double-sided routing solution can be produced by feeding the planning results into a commercial placement and routing tool, focusing on routing the critical nets in N_1 predominantly over the preferred layers. The effectiveness of routing will be evaluated based on several criteria, including the number of Design Rule Check (DRC) violations, total wirelength, via count, the average displacement of cells following placement legalization, and the average timing score of timing-critical nets in N_1, the timing score for a net n within N_1 is calculated by Eq. 1. This score is derived by considering the wire length L_i and the wire width W_i for each wire segment i in net n. The smaller the timing score the better is the timing.

$$timing\_score(n) = \sum_{i \in wire_n} \frac{L_i}{W_i} \tag{1}$$

4 PROPOSED APPROACH

To address the shortcomings in [4], we propose an efficient and effective net partitioning method, which consists of three phases 1) initial partitioning, 2) BBOX-based partitioning, and 3) connection point assignment descibed in subsections 4.2-4.4. Our objective is to assign as many pins of the critical nets as possible to the back side while mitigating the impact of overlapping regions on wirelength. Furthermore, calculation of both routing and bridging cell capacities are PG-aware as described in subsection 4.1.

4.1 PG-aware Capacity Calculation

PG-aware Bridging Cell Capacity When assigning a front-side pin to the back side (or assigning a back-side pin, such as an I/O pin, to the front side), a bridging cell is necessary to be inserted on the device layer. To estimate the placeable region during netlist planning, we divide the device layer into an array of Gcells and calculate the bridging cell capacity in each Gcell. In addition to the placeable region on the device layer, the PG network on BM_2 may also influence the placeable region of the bridging cell. Since only BM_2,...,BM_n can be used for back-side routing (BM_1 cannot be used for signal routing), vias between BM_1 and BM_2 are needed to access the back-side pins of the bridging cells on BM_1. But these vias cannot be implemented in regions where they would overlap with the PG network on BM_2. So, the PG network on BM_2 also affects the placeable region of the bridging cell. The bridging cell capacity in each Gcell v is calculated by Eq. 2 :

$$bridging\_cell\_capacity(v) = \frac{Gcell_{area}(v) - Area(cells(v) \cup PG(v))}{bridging\_cell_{area}} \quad (2)$$

where $Gcell_{area}$ is the area of a Gcell v, $Area(cells(v) \cup PG(v))$ represents the area of the union of the space occupied by cells in v and PG network in v projected onto the same plane, and $bridging\_cell_{area}$ is the area of a bridging cell. In practice, if the number of bridging cells inserted in a Gcell v exceeds the bridging cell capacity of v, it is still possible to accommodate them in neighboring Gcells by allowing more displacement during the placement legalization stage using commercial tools. Therefore, if a pin is located in Gcell v, we assess the feasibility of a bridging cell insertion based on whether there is sufficient bridging cell capacity in Gcell v or a neighboring Gcell located within a distance r from v.

PG-aware Routing Capacity Given that the PG network occupies tracks and affects the routing resources for signal routing on the back side, we deduct the tracks occupied by these routing obstacles. The remaining tracks are then considered as the available wire capacities for signal routing.

4.2 Initial Partitioning of Critical nets

During the initial partitioning phase, we establish a partial partitioning solution for the critical nets only and prioritize the critical nets with small overlapping regions in the partitioning solutions to use the bridging cells. We describe our initial partitioning in Algorithm 1.

Algorithm 1 Initial partitioning

Input: netlist N_1, threshold α, upper_bound β
Output: S_{front},S_{back}
1: Initialize $L_{remaining} \leftarrow N_1$
2: **repeat**
3: $L_{pc}, L_{sc}, L_{other}, S_{front\_temp}, S_{back\_temp} = net\_classification(L_{remaining}, \alpha)$
4: Sort L_{pc} by the size of pre-calculated overlapping region
5: **for** $i = 1, ..., min(\beta, |L_{pc}|)$ **do**
6: $n = L_{pc}[i]$
7: $S_{front}[n] = S_{front\_temp}[n]$
8: $S_{back}[n] = S_{back\_temp}[n]$
9: **end for**
10: Update the used bridging cell capacity in each Gcells
11: $L_{remaining} \leftarrow$ (unpartitioned nets in L_{pc}) $\cup$ L_{sc}
12: **until** L_{pc} is not empty
13: Recover nets with small bounding box on the back side
14: Recover congested nets

In line 3, $net\_classification$ classifies the critical nets into three categories L_{pc} (primary candidate set), L_{sc} (secondary candidate set), and L_{other} (other set). For each critical net n, we pre-calculate the size of the overlapping region (we call the pre-calculated overlapping region) that will result if pins with bridging cell capacity are assigned to the back side (stored in $S_{back\_temp}[n]$), while those without such capacity are assigned to the front-side (stored in $S_{front\_temp}[n]$). For a net n with non-empty $S_{back\_temp}[n]$, if the perimeter of its pre-calculated overlapping region is less than a fraction f of the net's original HPWL, we classify it as a primary candidate net. Otherwise, we classify it as a secondary candidate net. If a net n has empty S_{back_temp}, we classify it as an other net.

In line 4, the partitioning begins by sorting L_{pc} by the size of their pre-calculated overlapping regions in increasing order. In lines 5-9, for each primary candidate net n, pins in $S_{back\_temp}[n]$ are assigned to the back side (stored in $S_{back}[n]$), while pins in $S_{front\_temp}[n]$ are assigned to the front side (stored in $S_{front}[n]$). Since updating bridging cell capacities immediately after partitioning each net is time-consuming, we employ a lazy strategy that updates the utilized bridging cell capacities (line 10) after a certain number of nets have been partitioned (lines 5-9).

Moreover, we update the remaining nets $L_{remaining}$ in line 11. Although the nets in L_{sc} may create large pre-calculated overlapping regions at first, the size of overlapping regions might be reduced in some cases after updating the bridging cell capacities. The partitioning will be performed until no more primary candidate net exists (lines 12).

This ordering strategy ensures that nets with smaller overlapping regions are prioritized to use the bridging cells. Moreover, having no primary candidate nets left implies that assigning all the pins with sufficient bridging cell capacities of any remaining net will result in a large overlapping region, potentially leading to increased wirelength. Consequently, if there are no remaining primary candidate nets in the initial partitioning, the unpartitioned nets will go through BBOX-based partitioning in the next phase.

If the bounding box of $S_{back}[n]$ of a double-sided net n is too small, the timing benefit of routing on the back side may be offset by the cell delays of the bridging cells. Therefore, for a net with the bounding box size of S_{back} smaller than a threshold δ, we recover it back to an unpartitioned net (line 13). This net would be repartitioned in the BBOX-based partitioning stage.

To consider the routing resource utilization, we estimate the congestion for each partitioned net on each side (line 14). We construct

separate RUDY maps for each side based on the partitioning result. For each partitioned net n, we calculate $RUDY_n(v_i)$ for each Gcell v_i within the bounding box of n on side i, representing the average wirelength per unit area within that bounding box [8]. Side i refers to the front side if $i = 0$ and the back side if $i = 1$. The RUDY value for a Gcell v_i on side i is then determined by summing the RUDY values in v_i of all partitioned nets N_p as follows.

$$RUDY(v_i) = \sum_{n \in N_p} RUDY_n(v_i) \tag{3}$$

To estimate the congestion of a Gcell v_i on side i, we add its RUDY value and the number of pins located in that Gcell [8]. We then create separate congestion maps for the front side and back side. Recognizing that the available routing resources differ on the two sides, we normalize the congestion values of each Gcell in each map, which is done by dividing each value by the total routing capacities of the edges surrounding the respective Gcell v_i, as in Eq. 4.

$$cong\_map(v_i) = \frac{RUDY(v_i) + pin\_num(v_i)}{routing\_capacity(v_i)} \tag{4}$$

The average congestion value on each side for each partitioned net is given by Eq. 5.

$$avg\_cong\_value(b_n, i) = \frac{\sum_{v_i \in V_n} cong\_map(v_i)}{|V_n|} \tag{5}$$

V_n is a set of Gcells within the bounding box b_n on the side i of net n, and the average congestion value of n is calculated as the mean of the congestion values of all Gcells within V_n on the side i. If a net's congestion value exceeds a certain threshold γ, that net is recovered to an unpartitioned net and would be repartitioned in the BBOX-based partitioning stage.

4.3 BBOX-based Partitioning

In the second phase, we perform BBOX-based partitioning for all the unpartitioned nets. In the beginning, we prioritize the unpartitioned nets based on criticality and sort them based on their bounding box sizes. This net order ensures the critical nets with longer HPWL are given priority to use the bridging cells first. The flowchart of BBOX-based partitioning for a net n is depicted in Fig. 3. A net may have a single routing tree entirely on the front side, or the back side, or one subtree on the front side and another on the back side. So, we enumerate partitioning solutions that contain one or two bounding box(es) for the routing tree(s) of a net (detailed in 4.3.1). Note that the distribution of bridging cell capacity on the layout region is uneven, so it may not always be favorable to partition the pins of a net into two bounding boxes without any overlapping when a bounding box does not have enough bridging cell capacities for all pins within a bounding box. To have more flexible partitioning solutions, we allow the bounding boxes to overlap with each other. For a final partitioning solution with overlapping region between two bounding boxes, the side to which each pin within the overlapping region (called free pins) will be assigned is determined based on the availability of bridging cell capacity in the free pin allocation step. The selection of the final partitioning solution is guided by the partitioning cost (described in 4.3.2).

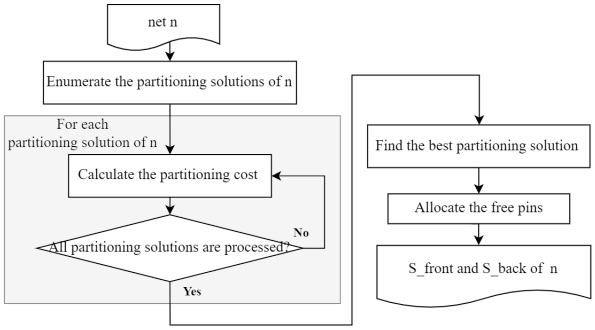

Figure 3: A flowchart of BBOX-based partitioning.

4.3.1 Enumerate the Partitioning Solutions. In this section, we detail the process for enumerating the partitioning solutions of net n with r pins. The process begins with a pin list of n sorted in increasing order based on their x-coordinates. Then, by sequentially traversing the pin list from the first to the last pin, we generate $(r-1)$ left bounding boxes through the gradual combination of pins. Similarly, $(r-1)$ right bounding boxes are created by traversing from the last to the first pin. A partitioning solution is computed whenever a left bounding box intersects with a right bounding box. Additionally, if a bounding box consists of all the pins of the net, it is considered alone to compute a partitioning solution. Fig. 4 illustrates various partitioning solutions for a six-pin net n. A bounding box colored red denotes a left bounding box, whereas an orange bounding box represents a right bounding box. For instance, the intersection of b_2_left with b_3_right at a pin location and the intersection of b_2_left with b_4_right resulting in an overlapping region are shown in Fig. 4(c) and Fig. 4(d), respectively. Similarly, we create partitioning solutions when the pin list is sorted in descending order of y-coordinates.

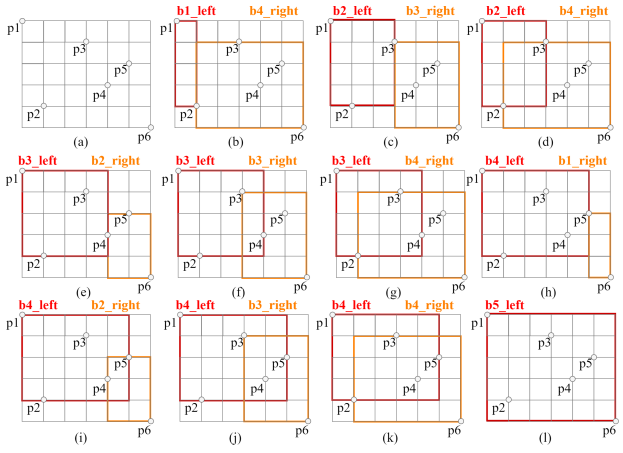

Figure 4: An example of enumerating partitioning solutions.

For those partitioning solutions without overlapping bounding boxes, the pin set of a bounding box consists of the pins located within that bounding box. Next, we detail how to create pin sets for bounding boxes in partitioning solutions with overlaps. A

boundary pin is defined as a pin located at the boundary of a bounding box that crucially determines the shape of that bounding box. A *free pin* is defined as a pin located within the overlapping region of the two bounding boxes but not positioned at the boundary of either bounding box. Regardless of which bounding box a free pin is associated with, it does not affect the shape of the bounding box. Consequently, the pin set S of a bounding box b includes all its boundary pins and the pins inside it that are not free pins or the boundary pins of the other bounding box.

For instance, the partitioning solution of the net depicted in Fig. 4(j) involves two bounding boxes labeled b_4_left and b_3_right. For b_4_left, its set of boundary pins is $\{p_1, p_2, p_5\}$, while for b_3_right, it is $\{p_3, p_6\}$. Thus, $S_{b\_left} = \{p_1, p_2, p_5\}$, $S_{b\_right} = \{p_3, p_6\}$, and the set of free pins is $\{p_4\}$. Note that the partitioning cost of this solution is calculated based on the pin sets S_{b_left} and S_{b_right}.

4.3.2 Partitioning Cost Function. To determine the final partitioning solution, we develop a partitioning cost metric for partitioning solutions, considering the congestion costs, the availability of bridging cell capacity, and the advantages it presents to critical nets when a bounding box is assigned to side i, and the total perimeter of all bounding boxes in the partitioned solution. The partitioning cost of a partitioning solution ps is calculated as in Eq. 6.

$$\begin{aligned} partitioning\_cost(ps) = \sum_{j=1}^{2} (PERI(b_j) + \\ \lambda_s \times min(side\_cost(b_j, 0, n), side\_cost(b_j, 1, n)) \end{aligned} \tag{6}$$

λ_s represents a weight. If the partitioning solution involves two bounding boxes and a large overlapping region between these two boxes, then a higher total perimeter of two bounding boxes $PERI$ cost would be incurred. We assign a bounding box to the side i with the lower side cost. The side cost of assigning the bounding box b_j for net n to side i is given by Eq. 7.

$$\begin{aligned} side\_cost(b_j, i, n) = &\lambda_c \times avg\_cong\_value(b_j, i) + \\ &\lambda_f \times TBO(S_{b_j}, i) + \\ &\lambda_p \times fpins(S_{b_j}, i, n) + \\ &SBBOX(b_j, i) \end{aligned} \tag{7}$$

where λ_c, λ_f, and λ_p denote the weights. $avg\_cong\_value(b_j, i)$ indicates congestion value if b_j is assigned to side i. $TBO(S_{b_j}, i)$ represents total bridging cell overflow if pin set S_{b_j} is assigned to side i (in Eq. 8). If $TBO(S_{b_j}, i)$ is high, it suggests that assigning all pins of S_{b_j} to side i would pose challenges in inserting the bridging cell. To encourage using more back-side resources for critical nets, $fpins(S_{b_j}, i, n)$ represents the number of pins in S_{b_j} as a penalty when n is a critical net and is assigned to the front side (in Eq. 9). To discourage a small sized b_j assigned to the back side, $SBBOX(b_j, i)$ represents a large penalty ω if b_j is assigned to the back side and the size of b_j is smaller than a threshold δ(in Eq. 10).

$$TBO(S_{b_j}, i) = \sum_{p \in S_{b_j}} BO(p, i) \tag{8a}$$

$$BO(p, i) = \begin{cases} 1 & \text{if } p \text{ requires but lacks bridging cell capacity} \\ 0 & \text{otherwise} \end{cases} \tag{8b}$$

$$fpins(S_{b_j}, i, n) = \begin{cases} \#pins\ in\ S_{b_j} & \text{if } n \text{ is a critical net} \wedge i = 0 \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

$$SBBOX(b_j, i) = \begin{cases} large\ penalty\ \omega & \text{if PERI}(b_j) < \text{a threshold } \delta \wedge i = 1 \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

4.4 Connection Point Assignment

After obtaining the final partitioning solution, a connection point is required to connect the subnets on the front side and back side for a double-sided net n. Similar to [4], if there exist an intersection region between the bounding box of the pins of n assigned to the front side and the back side, a connection point is chosen in the intersection region. If no intersection exists, it extends both bounding boxes simultaneously until an intersection is found, and the connection point is identified within the intersection region.

4.5 Location of Bridging cells

When multiple pins of a net are located within the same Gcell, a single bridging cell will be shared among them to access the back side. For each bridging cell b needed by pin p, we designate p as a target pin of b. Each bridging cell is given a location at the location of one of its target pins. The inserted bridging cells may overlap with other cells, which can be resolved by a placement legalizer.

5 EXPERIMENTAL RESULTS

5.1 Experimental Settings

The netlist planning flow, coded in C++, was run on a Linux system powered by an AMD EPYC 7282 16-Core 2.8 GHz CPU and equipped with 256GB of RAM. To evaluate our method's performance, we adapted benchmarks from the ISPD'18 detailed routing contest. However, testcases test6 and test10 were omitted from our analysis due to their excessively high placement density (above 95%), which do not have enough space for bridging cell insertion.

The original benchmarks from ISPD'18 featured nine metal layers. In our modified version, labeled as double-sided ISPD'18 benchmarks, we incorporated three additional, thicker metal layers on the back side, specifically designated as BM_1, BM_2, and BM_3, as shown in Fig. 5. To simulate the size of a bridging cell, we utilize the size of the smallest inverter cell in our benchmarks. We also identified the nets with the top 3% of bounding box sizes as the critical nets. For routing these critical nets, we preferred the uppermost two layers (M_8 and M_9) on the front side, and two layers (BM_2 and BM_3) on the back side. Furthermore, we created routing blockages on the routable back-side metal layers (BM_2 and BM_3) to simulate the PG network. The average PG density on BM_2 and BM_3 is shown in Table. 1.

To evaluate the effectiveness of our methodology, we utilized the Cadence Innovus 21.13 commercial placement and routing tool

Table 1: Comparison of different partitioning methods. ("WL", "#Vias", "TS", "Dis. (μm)", "#DRVs", and "RT (s)" denote the total wirelength, the via count, the average timing score of the critical nets, the average cell displacement after legalization, the number of DRC violations, and the runtime for the netlist planning flow, respectively.)

Case name	Benchmark information				[4]						Tool-based netlist planning method						Ours					
	#Nets	#Pins	Placement density(%)	PG density on BM_2/BM_3 (%)	WL (μm)	# Vias	TS	Dis.(μm)	# DRVs	RT(s)	WL (μm)	# Vias	TS	Dis.(μm)	# DRVs	RT(s)	WL (μm)	# Vias	TS	Dis.(μm)	# DRVs	RT(s)
test1	3153	17203	85.03	28.37/27.42	93011	41399	4963.39	1.31	0	0.90	136460	51116	11269.93	5.50	0	116	89831	40600	4339.74	1.23	0	1.07
test2	36834	159201	56.59	23.14/23.67	1690940	411340	9447.74	2.60	0	11.31	1630878	404692	8602.79	2.43	0	612	1631797	411225	8522.50	2.34	0	10.46
test3	36700	159703	45.56	24.92/25.02	1809170	389135	9320.76	4.34	0	13.72	1807478	387096	9291.43	4.75	0	805	1813774	394802	9281.43	3.22	0	12.29
test4	72410	318245	88.98	29.05/31.6	2908532	802393	6477.40	1.94	0	59.51	2937244	822506	6708.98	1.73	0	816	2862979	794088	6257.84	1.37	0	39.55
test5	72394	318195	92.32	27.86/30.64	2947137	885115	5554.76	3.31	51	85.19	2966682	884165	5597.24	35.88	107	355	2927161	881798	5416.35	4.02	0	43.75
test7	179863	793289	89.62	28.65/30.06	7234960	2259880	6661.00	1.90	79	217.48	7134375	2241636	6228.11	5.86	0	2460	6981423	2228356	6038.44	2.91	0	141.50
test8	179863	793289	90.00	28.65/30.06	7102603	2231126	6338.17	2.35	9	211.72	7149823	2239016	6207.51	6.20	0	2502	7060824	2223177	6118.75	3.73	0	143.68
test9	178858	791761	90.59	28.87/27.97	5720667	2263402	4750.23	1.29	0	180.78	5614183	2282742	4571.81	1.01	0	1186	5578762	2240662	4502.54	1.04	0	138.33
Nor. Avg.	-	-	-	-	1.020	1.006	1.064	1.036	-	1.351	1.074	1.037	1.222	2.728	-	38.067	1.000	1.000	1.000	1.000	-	1.000

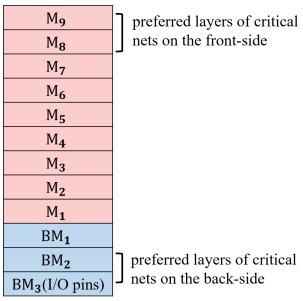

Figure 5: Double-sided routing architecture.

to perform placement legalization for all cells and generate double-sided routing solutions that route separately on each side based on separate netlists from our approach. Note that the legalization process for bridging cells also considers the PG network on BM_2, which affects the accessibility of back-side pins on BM_1.

5.2 Comparing Different Partitioning Methods

In this subsection, we provide a detailed comparison between our method and both the method described in [4] and a tool-based netlist planning method we implemented.

Since [4] did not account for the routing blockages computed by the PG network, we applied a discount to both routing and bridging cell capacities based on the average ratio of PG network occupancy in each testcase. As shown in Table 1, our method not only improved the average wirelength by 2% and achieved zero DRC violation in each testcase, but also enhanced the average timing score performance for critical nets by 6.4% and reduced the runtime by 35.1% on average. Since our method aims to reduce the sizes of the overlapping regions of double-sided nets, it effectively decreases the wirelength and results in zero DRV in all cases. Additionally, our method is much more efficient since it does not need the pin redistribution stage in [4] that computes the congestion maps in each iteration to identify the congestion regions which is time-consuming.

Furthermore, we implemented a tool-based netlist planning flow. We first created duplicate pin testcases based on the double-sided ISPD'18 benchmarks. In the duplicate pin testcases, for every pin of each critical net, we duplicated its shape and placed the duplicate on the side opposite to its original location. To ensure the signal is passed from the front side to back side (or vice versa) exclusively through the duplicate pins, we introduce a "BLK Layer" with a routing blockage across the entire design between the front-side and back-side metal layers to separate the routing. Additionally, we fill the regions on BM_1 with routing blockages, excluding the shapes of pins on BM_1 to comply with the prohibition of routing on BM_1 Moreover, we execute routing by commercial tool. Finally, the separate netlists on both sides are generated based on the routing solutions. Compared to the tool-based netlist planning approach shown in Table 1, our method not only reduced the average wirelength by 7.4% and the average via count by 3.7%, but also notably enhanced the average timing score for critical nets by 22.2%. The efficiency of our method resulted in a significant speedup of over 38 times. Moreover, we achieved 2.7 times less displacement than the baseline by integrating bridging cell capacity considerations into our netlist planning process.

6 CONCLUSION

In this work, we proposed a bounding box-based net partitioning method to improve the solution quality of netlist planning for double-sided routing and we proposed a PG-aware capacity calculation for estimating the bridging cell capacities and routing capacities affected by PDN. Beyond the comparison with [4], we established a new baseline for comparison: a tool-based netlist planning method utilizing a commercial tool.

REFERENCES

[1] Yen-Jung Chang, Yu-Ting Lee, Jhih-Rong Gao, Pei-Ci Wu, and Ting-Chi Wang. 2010. NTHU-Route 2.0: A Robust Global Router for Modern Designs. *IEEE TCAD* 29, 12 (2010), 1931–1944.

[2] Sheng-Hsiung Chen, Jerry Chang Jui Kao, Kuo-Nan Yang, and Jack Liu. 2022. System and method for back side signal routing. In *U.S. Patent 11423204.*

[3] W. Hafez et al. 2023. Intel PowerVia Technology: Backside Power Delivery for High Density and High-Performance Computing. In *IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits).*

[4] Tzu-Chuan Lin, Fang-Yu Hsu, Wai-Kei Mak, and Ting-Chi Wang. 2024. An Effective Netlist Planning Approach for Double-sided Signal Routing. *Proceedings of 2024 ASPDAC* (2024).

[5] Eric BeyneJulien Ryckaert. 2017. An integrated circuit chip with power delivery network on the backside of the chip. In *European Patent EP3324436A1.*

[6] J. Ryckaert, A. Gupta, A. Jourdain, B. Chava, G. Van der Plas, D. Verkest, and E. Beyne. 2019. Extending the roadmap beyond 3nm through system scaling boosters: A case study on Buried Power Rail and Backside Power Delivery. In *2019 Electron Devices Technology and Manufacturing Conference (EDTM).* 50–52.

[7] G. Sisto, R. Preston, R. Chen, G. Mirabelli, A. Farokhnejad, Y. Zhou, I. Ciofi, A. Jourdain, A. Veloso, M. Stucchi, O. Zografos, P. Weckx, G. Hellings, and J. Ryckaert. 2023. Block-level Evaluation and Optimization of Backside PDN for High-Performance Computing at the A14 node. In *IEEE Symposium on VLSI Technology and Circuits (VLSI Technology and Circuits).* 1–2.

[8] Peter Spindler and Frank M. Johannes. 2007. Fast and Accurate Routing Demand Estimation for Efficient Routability-driven Placement. In *2007 DATE.* 1–6.