Differentiable Net-Moving and Local Congestion Mitigation for Routability-Driven Global Placement

Wenchao Li[1], Hongxi Wu[1], Duanxiang Liu[2], Xingquan Li[3], Wenxing Zhu[1*]
[1]Fuzhou University, China; [2]Shanghai UniVista Industrial Software Group Co., Ltd., China; [3]Pengcheng Laboratory, China

***Abstract*—Routability-driven global placement is a major challenge in modern VLSI physical design, for which mitigating routing congestion is a critical approach. Cell inflation can effectively address local routing congestion and is widely adopted, but with the issue of over-inflating or moving cells back into congested areas. Minimizing the congestion within a net bounding box is effective for alleviating global routing congestion, but the bounding box may be too large and contain congestion not contributed by the net. To address the first issue, we propose a momentum-based cell inflation technique that considers historical inflation ratios for mitigating local routing congestion. Then, we construct a differentiable global congestion function, developed from Poisson's equation, and introduce virtual standard cells onto two-pin nets to accurately guide net movements for mitigating global routing congestion. Furthermore, to improve pin accessibility, we adjust placement density around power and ground rails according to the routing congestion in global placement. The proposed techniques are integrated into an electrostatic-based global placement framework. Experiments on the ISPD 2015 contest benchmarks show that our framework achieves better routability results, with an average of 40% DRVs reduction and comparable wirelength and via count, compared to the leading routability-driven placer.**

***Index Terms*—routability-driven global placement, eliminating congestion, cell movement.**

I. Introduction

Ideally, placement and routing should be performed simultaneously in VLSI physical design. However, this is highly challenging due to their different problem nature and high computational complexity. Therefore, most studies follow the placement-then-routing flow and investigate ways to balance placement and routing to improve overall design efficiency [1, 2, 3, 4, 5]. With the rapid growth of design complexity and advances in process technology, routability has become a big issue in modern circuit design. Separation of placement and routing often leads to a large number of design rule violations after detailed routing. Therefore, routability-driven placement becomes critical. Optimizing a placement by considering routability can effectively address the routability and reduce the complexity in the later routing stage [3, 6, 7]. This work focuses on routability-driven global placement problem.

During global placement, the routability issue is complicated by diverse sources of routing congestion. As noted in [2], congestion can be divided into local routing congestion and global routing congestion. Fig. 1(a) shows both types of congestion. Local routing congestion occurs when there are too many cells in a given placement region, this congestion can be alleviated by relocating cells from the congested placement region. Global congestion occurs when too many nets traverse a G-cell, this congestion cannot be alleviated by moving cells out of the congested placement area, since there may be no cells in the congested region. Mitigating both types of congestion is a critical issue in routability-driven global placement.

Many strategies have been proposed to consider routability during the global placement stage. Cell inflation is an effective approach for mitigating local routing congestion. This approach usually utilizes a global router to create a routing congestion map and then applies cell inflation to artificially increase the density of congested areas, encouraging cell displacement from these regions [3, 5, 8]. One approach of cell inflation [3, 5] only considers the present congestion, when cells are moved to an uncongested area, these cells are not inflated, causing the cells to return to the previously congested areas inadvertently, and these areas may become congested again [9]. Another approach is inflating cells monotonically increasingly by considering historical inflation ratios [8]. This technique may lead to over-inflation even when cells have been moved away from the congested area. Therefore, to maximize the efficacy of cell inflation, it is crucial to design a dynamic cell inflation scheme by incorporating historical inflation ratios.

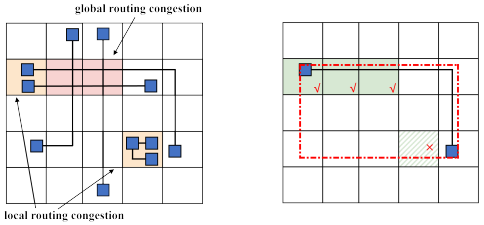


Fig. 1: (a) Local routing congestion and global routing congestion. (b) A net from (a); the red dashed box represents the bounding box (BB) of the net, the green area indicates routing congestion, and the congestion in the lower right corner is unrelated to the net.

To alleviate global routing congestion, it is an effective approach that adds a congestion penalty based on the bounding box (BB) of a net to the objective function of the global placement mathematical model. Within the deep learning framework, [4] leverages RUDY [10], PinRUDY and MacroRegion features, which are calculated based on BB, for forward propagation to generate a congestion map, followed by backpropagation to obtain routability-driven cell gradients. Nevertheless, RUDY treats all regions within the BB equally, overlooking the specific congestion situation. [6] introduces a congestion penalty for G-cell edges, minimizing the usage of G-cell edges based on nets' locations. Specifically, the congestion penalty involves every edge of G-cells surrounding the BB of a net, and the achieved effect is a reduction in the cells' distance within the net rather than direct congestion mitigation. [2] gives a BB-based congestion-aware net penalty model to reduce the net's overlap with congested areas. However, the BB may be too large and contain congestion that is not contributed by the net. For instance, as Fig. 1(b) illustrates, congestion within a net's BB may not be caused by the net itself, the lower right congested region stems from excess cell clustering, but [2] penalizes the net for this unrelated congestion.

In addition to local and global routing congestion, pin accessibility is another critical issue closely tied to routability. Existing methods

*Corresponding author: Wenxing Zhu, email: wxzhu@fzu.edu.cn.

[11, 12, 13] primarily optimize pin accessibility at the detailed placement stage, using local perturbations such as cell shifting, cell flipping, cell swapping, and cell replacement. However, pin accessibility has not been addressed at the global placement stage, which would be helpful for routability.

To address the above issues, we propose an analytical routability-driven global placement framework that includes net and multi-pin cell moving strategies, and a momentum-based cell inflation technique, which can simultaneously alleviate local and global routing congestion. Additionally, we consider optimizing pin accessibility during global placement to enhance routability further. The main contributions of this work are summarized as follows:

- We propose a differentiable congestion function developed from Poisson's equation and introduce virtual cells onto two-pin nets. The function is penalized into the global placement model for effectively guiding net movements to mitigate global routing congestion.
- A momentum-based cell inflation technique is developed to incorporate historical inflation ratios in calculating cells' current inflation ratios to mitigate local routing congestion more effectively.
- To improve pin accessibility, we adjust the placement density around power and ground rails during global placement, according to the routing congestion situation of the related region.
- We integrate the proposed techniques into an electrostatic global placement framework. Experiments on the ISPD 2015 detailed-routing-driven placement contest benchmarks demonstrate that our algorithm improves routability, achieving an average of 40% reduction in DRVs while maintaining comparable wirelength and via count, compared to the leading routability-driven placer Xplace-Route.

II. Analytical Global Placement

This section introduces the wirelength-driven analytical global placement problem, and presents our formulation of analytical routability-driven global placement problem.

A. Wirelength-Driven Global Placement

A circuit can be represented by a hypergraph $H = (V, E)$, where $V = \{v_1, v_2, \ldots, v_n\}$ denotes the set of cells, and $E = \{e_1, e_2, \ldots, e_r\}$ is the set of nets. The placement area is divided into a uniform grid of $m \times n$ bins $\mathbb{B}$. Then the wirelength-driven global placement problem can be formulated as

$$\begin{aligned} \min_{x,y} \quad & \sum_{e \in E} WL(e; x, y) \\ \text{s.t.} \quad & D_b(x, y) \le D_b, \quad \forall b \in \mathbb{B}, \end{aligned}$$

where $WL(e; x, y)$ is net e's wirelength, $D_b(x, y)$ is the density of bin b, and D_b represents the maximum allowable density of the bin.

The wirelength function $W(e; x, y)$ is often defined as the half-perimeter wirelength (HPWL). Direct minimization of HPWL in global placement is challenging due to its non-smoothness. In this work, we adopt the weighted average (WA) wirelength model [14] to approximate HPWL, the WA model in the x-direction is:

$$WA_{e_x} = \left(\sum_{i \in e} x_i e^{\frac{x_i}{\gamma}} \Big/ \sum_{i \in e} e^{\frac{x_i}{\gamma}} \right) - \left(\sum_{i \in e} x_i e^{-\frac{x_i}{\gamma}} \Big/ \sum_{i \in e} e^{-\frac{x_i}{\gamma}} \right).$$

The WA model iterates through the horizontal and vertical coordinates of each pin i in net e, producing a smooth wirelength function to approximate the HPWL, with γ controlling the degree of smoothness.

Electrostatics-based density [15] has inspired many state-of-the-art global placement algorithms [3, 16, 17]. This approach models each cell as a charged particle interacting within an electric field, where the field strength reflects cell density. High-density region creates a repulsive force, pushing cells toward lower-density region to achieve a balanced placement. Specifically, based on Gauss's law, ePlace [15] utilizes Poisson's equation with boundary and compatibility conditions to address cell density in placement optimization:

$$\begin{cases} \nabla \cdot \nabla \psi(x, y) = -\rho(x, y), & (x, y) \in R, \\ \mathbf{n} \cdot \nabla \psi(x, y) = 0, & (x, y) \in \partial R, \\ \iint_R \rho(x, y) = \iint_R \psi(x, y) = 0, \end{cases} \tag{1}$$

where R is the rectangular placement area, ∂R denotes its boundary, $\mathbf{n}$ is the outward normal vector of the region, ρ is the charge density, and ψ stands for the electric potential. The numerical solution of Poisson's Equation (1) can be found in [15].

Then, the wirelength-driven analytical global placement problem can be formulated as [3, 15, 16]:

$$\min_{x,y} \sum_{e \in E} WA_e(x, y) + \lambda_1 D(x, y), \tag{2}$$

where $D(x, y)$ is the electrostatic-based density penalty term, calculated by: $D(x, y) = \frac{1}{2} \sum_{i \in V} A_i \psi_i$, where A_i is the area of cell i, ψ_i is the electric potential of cell i. λ_1 is the weighting factor to balance the total wirelength and density penalty. Calculation of the electrostatic-based density gradient $\nabla D(x, y)$ can be found in [15].

B. Analytical Routability-Driven Global Placement

To address routability in global placement, we first introduce the definition of routing congestion. The 3D routing region is divided into a set of global routing cells (G-cells), denoted by $G_R \in \mathbb{R}^{R_r \times R_c \times L}$, here, R_r and R_c represent the row and column numbers of the routing grid, while L denotes the number of routing layers. Based on the G-cells, a congestion map C can be defined as [8]:

$$C_{m,n} = \max \left(\frac{\sum_{l=1}^{L} Dmd_{m,n,l}}{\sum_{l=1}^{L} Cap_{m,n,l}} - 1, 0 \right), \quad \forall (m, n) \in \mathbb{R}^{R_r \times R_c}, \tag{3}$$

where $C_{m,n}$ is the congestion value of the m-th row and n-th column of congestion map C. Dmd represents the routing demand map, which represents the total of wire demand and via demand, Cap refers to the routing capacity map.

To facilitate the establishment of the congestion model, we create a mapping of G-cells onto the 2D plane, which has the same dimension as the bins division. Let $Dmd_{m,n} = \sum_{l=1}^{L} Dmd_{m,n,l}$ and $Cap_{m,n} = \sum_{l=1}^{L} Cap_{m,n,l}$ represent the total routing demand and the total routing capacity of all layers' G-cells located at the m-th row and n-th column, respectively. Hereafter, any mention of G-cell refers to the 2D mapping. For routability-driven global placement, constraints are imposed into Equation (2) to ensure that the routing demand does not exceed its given routing capacity, and the routability-driven global placement problem is:

$$\begin{aligned} \min_{x,y} \quad & \sum_{e \in E} WA_e(x, y) + \lambda_1 D(x, y) \\ \text{s.t.} \quad & Dmd_{m,n} \le Cap_{m,n}, \quad \forall (m, n) \in \mathbb{R}^{R_r \times R_c}. \end{aligned} \tag{4}$$

Meeting the routing congestion constraints of Equation (4) is challenging, as routing demands frequently exceed available resources in certain regions, making routing congestion hard to alleviate. To handle this issue, we adopt the electric field model of ePlace [15], which enables moving cells away from high-density placement areas.

Analogous to [15], we apply the electric field model for routing congestion mitigation, using Poisson's equation to obtain an analytical solution for guiding the movement of cells away from high-congestion areas. To this aim, we take ρ in Equation (1) as

$$\rho_{m,n} = \frac{Dmd_{m,n}}{Cap_{m,n}}, \quad \forall (m, n) \in \mathbb{R}^{R_r \times R_c},$$

where $Dmd_{m,n}$ and $Cap_{m,n}$ are obtained by invoking the global router of [18]. Then, we solve Poisson's Equation (1) to obtain a solution, which is the electric potential for congestion, and calculate the related electrostatic-based congestion penalty term $C(x,y)$ by: $C(x,y) = \frac{1}{2}\sum_{i\in V'} A_i\psi_i$, where V' represents the set of multi-pin cells and virtual cells (for moving nets, detailed in Subsection III-A), A_i and ψ_i denote the area and electric potential of cell i, respectively. Then similar to Equation (2), the analytical routability-driven placement model can be established:

$$\min_{x,y} \quad \sum_{e\in E} WA_e(x,y) + \lambda_1 D(x,y) + \lambda_2 C(x,y), \tag{5}$$

where λ_2 is the weight for the congestion penalty term, which will be elaborated on in Subsection III-A. **The electrostatic-based congestion gradient $\nabla C(x,y)$ is computed with the method in [15]. However, it is not used directly as that of cell density, since the aim is mainly for moving nets to mitigate global routing congestion.** Updating $\nabla C(x,y)$ is detailed in Subsection III-A.

III. The Proposed Algorithm

This section details our proposed analytical routability-driven global placement framework for problem (5). Fig. 2 outlines the overall flow of the proposed algorithm. The red boxes indicate the technical points that we contribute.

In Fig. 2, we select power and ground (PG) rails based on the positions of macro blocks to prepare for the subsequent pin accessibility optimization. Then, we use the wirelength-diven global placer Xplace [16] to obtain an initial global placement result. Next, we proceed to the routability-driven global placement stage. We adopt the GPU-accelerated 3D Z-shape routing algorithm [18] to estimate routing congestion and generate a congestion map. This map is used for adjusting the cell inflation rate through our momentum-based cell inflation technique, and then we dynamically modify the local density using our pin accessibility adjustment technique. Next, the updated placement information is incorporated into the objective function of problem (5), which is subsequently solved using the Nesterov solver proposed in [15]. The routability-driven global placement is repeated until the $C(x,y)$ no longer decreases or the given number of iterations is reached. Lastly, the routability-driven legalization and detailed placement of Xplace-route [8] is adopted for the final placement.

In the following content of this section, we first introduce the congestion gradient update for mitigating global routing congestion in Subsection III-A. Then in Subsection III-B, we present the momentum-based cell inflation technique for mitigating local routing congestion. Finally, Subsection III-C discusses the pin accessibility optimization during the global placement stage.

A. Congestion Gradient Update

This subsection presents the congestion gradient update technique for the problem (5). As described earlier, congestion is generally categorized into local routing congestion, due to an excessive number of cells within an area covered by a routing grid, and global routing congestion, caused by high wiring density within a routing grid.

After adopting the GPU-accelerated 3D Z-shape routing algorithm [18] to estimate routing congestion and generate a congestion map, we can obtain the congestion penalty and cells' congestion gradient in problem (5). However, it would be insufficient to move the cells directly according to the congestion gradient generated by the electric field model as we do for the cell density, because a high congestion value is not only due to the excessive clustering of cells (local routing congestion, handled in Subsection III-B), but may also result

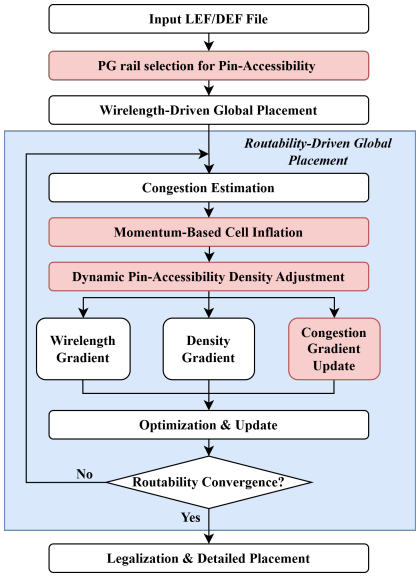


Fig. 2: Overview of our routability-driven global placement flow.

from a large number of nets covering the region (global routing congestion), as demonstrated in Fig. 1. In this subsection, we propose two strategies to mitigate global routing congestion.

Initially, We set the congestion gradient of all cells to 0, and then adjust them via the following approach. Firstly we update the congestion gradient for the two cells connected by a two-pin net, aiming to move the net. Then we update the congestion gradient for selected multi-pin cells.

1) Congestion Gradient Update for Two-Pin Net Moving: To mitigate global routing congestion, we propose a congestion gradient update algorithm for two-pin net moving. Since there is no routing information in the placement stage, the electric field force model only affects cells within the placement region, while generally ignoring net information. Therefore, to optimize routability for the nets, we introduce a virtual cell in every two-pin net, serving as a pivot point to guide the movement of the net.

For a two-pin net e with pins p_1 and p_2, as shown in Fig. 3(a), (x_1,y_1) and (x_2,y_2) are the positions of p_1 and p_2, respectively. Let c_1 and c_2 represent the cells to which these two pins belong. By connecting the pins p_1 and p_2, we get the segment $\widehat{p_1p_2}$. Then we determine the location of a virtual cell c_v with the same size as a standard cell for the two-pin net in a congested region.

Let l_x and l_y denote the width and height of a G-cell, respectively. We calculate (x_v,y_v) for c_v by the following equations:

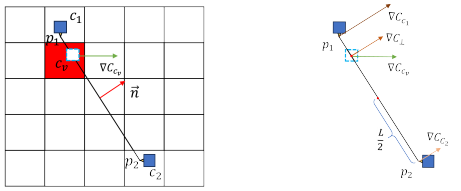
$$k = \max\left(\left\lfloor\frac{|x_1-x_2|}{l_x}\right\rfloor, \left\lfloor\frac{|y_1-y_2|}{l_y}\right\rfloor\right), \tag{6}$$

$$(x_i,y_i) = (x_1,y_1) + \frac{i}{k+1}(x_2-x_1, y_2-y_1), \quad i\in[1,k], \tag{7}$$

$$(x_v,y_v) = \arg\max_{(x_i,y_i)} C_{m,n}, \quad m = \left\lfloor\frac{x_i}{l_x}\right\rfloor, n = \left\lfloor\frac{y_i}{l_y}\right\rfloor. \tag{8}$$

In Equation (6), we approximate the number k of G-cells traversed by the segment $\widehat{p_1p_2}$ by taking the maximum value between $\left\lfloor\frac{|x_1-x_2|}{l_x}\right\rfloor$ and $\left\lfloor\frac{|y_1-y_2|}{l_y}\right\rfloor$. Next we proportionally select k candidate points (x_i,y_i) on the segment in Equation (7). Then, we substitute the candidate points into the congestion map to select the candidate point with the maximum congestion value as (x_v,y_v) in Equation (8), where $C_{m,n}$ is given in Equation (3). At (x_v,y_v), a virtual cell c_v of the same size as a standard cell is created (shown in Fig. 3(a)).

Subsequently, we utilize the virtual cell c_v to calculate the con-



(a) (b)

Fig. 3: Congestion gradient for cells on a two-pin net. (a): c_1 and c_2 are two cells connected by the net with pins p_1 and p_2. $\vec{n}$ is the unit normal direction of the segment $\widehat{p_1p_2}$, oriented to form an acute angle with ∇C_{c_v}. The red region indicates the most congested region, where a virtual cell c_v is generated with the congestion gradient ∇C_{c_v}. (b): $\nabla C_\perp$ is ∇C_{c_v}'s projection onto $\vec{n}$. Then, ∇C_{c_1} and ∇C_{c_2} are updated so that they have the same direction as $\nabla C_\perp$.

gestion gradients for the cells c_1 and c_2, ensuring their connection net can be moved away from the high congestion region. Let $\vec{n}$ be the unit normal direction of the segment $\widehat{p_1p_2}$, and oriented to the direction with an acute angle with ∇C_{c_v}. Here, ∇C_{c_v} is the congestion electric field gradient experienced by the virtual cell c_v. We project ∇C_{c_v} onto $\vec{n}$ and get $\nabla C_\perp$, as shown in Fig. 3(b). This is the most efficient direction for the connecting net to leave the congested region. The final congestion gradient for the cells c_1 and c_2 is calculated as follows:

$$d_{iv} = \sqrt{(x_v - x_i)^2 + (y_v - y_i)^2},$$
$$\nabla C_{c_i} = \frac{L}{2d_{iv}} \nabla C_\perp, \tag{9}$$

where d_{iv} is the Euclidean distance between pin p_i and virtual cell c_v. $L = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$ is the segment length of $\widehat{p_1p_2}$.

In Equation (9), we use $\frac{L}{2}$ as a reference to control a cell's congestion gradient based on the distance between the cell and the virtual cell. A cell near the congested region receives a larger congestion gradient, while a cell far away from the congested region receives a smaller congestion gradient. This allows the cells to move away from highly congested routing regions more efficiently.

Algorithm 1 summarizes the pseudocode for the proposed net-driven cell congestion gradient calculation. First, we identify the maximum congestion point (x_v, y_v) along the segment connecting two pins (line 1) and create a virtual cell c_v (line 2). The congestion gradient ∇C_{c_v} of c_v is then obtained using the electric field model (line 3). Next, the straight-line distance L between the two pins is calculated, and a perpendicular vector $\vec{n}$ that forms an acute angle with the gradient ∇C_{c_v} is derived (lines 4-5). Finally, the congestion gradient of c_1 and c_2 is computed (lines 6-9).

2) Congestion Gradient Update for Multi-pin Cell: A multi-pin cell is connected to several other cells. A large number of nets are connected to multi-pin cells, and these cells often increase global routing congestion in areas where routing resources are already limited. To address this issue, we compute by Equation (1) the congestion gradient of **selected multi-pin cells**, and then update the gradient for moving these cells away from congested regions, thereby alleviating global routing congestion. The specific approach is presented as Algorithm 2.

In lines 3-6 of Algorithm 2, Algorithm 1 is executed to obtain the congestion gradient for each cell of the two-pin net. Lines 7-15 perform congestion gradient adjustments for selected multi-pin cells. Specifically, when the number of pins of a cell i is greater than the average number of pins $\bar{n}$ (line 1) and the congestion value of the G-cell under which the cell i's center position is located exceeds 0.7 (line 11), we adopt the congestion gradient of the cell by Equation (1) (line 12). Finally, the congestion gradient of all cells in net e is obtained by combining the congestion gradients produced by the two approaches (lines 16).

Algorithm 1: Cell Congestion Gradient Update for Two-Pin Net Moving

Input: Cells c_1, c_2, Pin positions p_1, p_2, Congestion map C;
Output: $CGrad^{net}$: Congestion gradient of cells c_1 and c_2.
1 $(x_v, y_v) \leftarrow$ find max congestion position by Equation (8);
2 $c_v \leftarrow$ create a virtual cell;
3 $\nabla C_{c_v} \leftarrow$ congestion gradient of c_v by Equation (1);
4 $L \leftarrow$ calculate distance between p_1 and p_2;
5 $\vec{n} \leftarrow$ the unit vector perpendicular to the segment connecting p_1 and p_2 and forming an acute angle with ∇C_{c_v};
6 **for** $i \in \{1, 2\}$ **do**
7 $\quad d_{iv} \leftarrow$ calculate distance between p_i and c_v;
8 $\quad \nabla C_\perp \leftarrow (\nabla C_{c_v} \cdot \vec{n}) \cdot \vec{n}$;
9 $\quad CGrad^{net}_{c_i} \leftarrow \frac{L}{2d_{iv}} \nabla C_\perp$;
10 **end**
11 **return** $CGrad^{net}$;

With Algorithm 2, the congestion gradient of all cells is obtained by traversing all nets of a circuit, and then the congestion gradient of each cell is the superposition of the congestion gradients returned to the cell by each of these nets.

Algorithm 2: Congestion Gradient Update for Cells on a Net

Input: Net e, the set of cells c_i of the net, pins $p_{i,j}$ of cell i, congestion map C;
Output: $CGrad$: Congestion gradient of cells c_i in net e.
1 $\bar{n} \leftarrow$ average number of pins of all cells;
2 $k \leftarrow$ number of pins in the net e;
3 $CGrad^{net} \leftarrow 0$;
4 **if** $k = 2$ **then**
5 $\quad$ Use Algorithm 1 to obtain $CGrad^{net}$;
6 **end**
7 $CGrad^{cell} \leftarrow 0$;
8 **foreach** c_i **do**
9 $\quad n \leftarrow$ the number of pins of cell c_i;
10 $\quad C_{c_i} \leftarrow$ the congestion value of the G-cell under which the cell c_i's center position is located;
11 $\quad$ **if** $n > \bar{n}$ *and* $C_{c_i} > 0.7$ **then**
12 $\quad\quad CGrad^{cell}_{c_i} \leftarrow$ congestion gradient of c_i by Equation (1);
13 $\quad$ **end**
14 $\quad CGrad^{cell} \leftarrow CGrad^{cell}_{c_i}$;
15 **end**
16 $CGrad \leftarrow CGrad^{net} + CGrad^{cell}$;
17 **return** $CGrad$;

3) The Weight of Congestion Penalty Term: In Section II-B, we established the analytical routability-driven global placement model (5), in which λ_2 is the congestion penalty weight. Here, it is set as:

$$\lambda_2 = \frac{2N_C}{N} \cdot \frac{||\nabla W(x,y)||_1}{||\nabla C(x,y)||_1}. \tag{10}$$

In Equation (10), λ_2 is obtained by the ratio of the L_1 norm of the wirelength gradient $\nabla W(x,y)$ to that of the congestion gradient $\nabla C(x,y)$, multiplied by an adjustment coefficient. Specifically, N_C is the total number of cells in congested regions, and N is the total number of cells. Using Equation (10), λ_2 is adjusted based on the current overall congestion during the iteration. When the number of cells in a congested area is big, λ_2 will be large, prioritizing con-

gestion optimization. Conversely, if congestion is low, λ_2 decreases, allowing for a greater emphasis on wirelength optimization.

B. Momentum-Based Cell Inflation

This subsection presents the momentum-based cell inflation technique for mitigating local routing congestion. Most cell inflation techniques primarily adjust cell's inflation rate based on the congestion information from the current iteration [3, 5]. They encounter the issue of moving a cell back into the original congested area when the cell is transited from the inflated state to the original state or over-inflated during the iteration. To address this issue, we incorporate the momentum technique into calculating the inflation rate. Our approach considers the historical inflation rate of a cell similar to NTUplace4dr [9] and a cell deflation mechanism to utilize available resources better. Once the inflation rate is determined, the cell size is proportionally inflated during density calculation, increasing the density of congested region. This provides a higher gradient to the cells in congested regions, prompting them to spread out the regions.

Specifically, the core formula is as follows:

$$\begin{aligned} r_i^t &= \min\left\{\max\left(r_i^{t-1}+\Delta r_i^t, r_{\min}\right), r_{\max}\right\}, \\ \Delta r_i^t &= \alpha \Delta r_i^{t-1} + (1-\alpha) s_i^t, \\ s_i^t &= \delta_i^t C_i^t, \end{aligned} \tag{11}$$

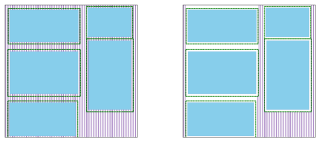
where r_i^t represents the inflation rate of cell i at the t-th ($t = 1, 2, 3, \dots$) iteration of cell inflation, and $r_i^0 = 1$. The r_{min}, $r_{\max}$, and α are user-specified values that provide an appropriate inflation rate for each cell, which are set to 0.9, 2.0, and 0.4 in this work respectively. Δr_i^t denotes the change value in the inflation rate, which is mainly affected by the inflation rate of all previous iterations, and $\Delta r_i^1 = C_i^1$. C_i^t is the congestion value of the G-cell under which cell i's center position is located at the t-th iteration of cell inflation. Moreover, in Equation (11), s_i^t is the congestion correction factor, which is determined by a decision parameter δ_i^t and the congestion value. δ_i^t is calculated by:

$$\delta_i^t = \begin{cases} -\left|\frac{C_i^{t-1}\bar{C}^t - C_i^t \bar{C}^{t-1}}{\bar{C}^{t-1}\bar{C}^t}\right|, & \text{if } C_i^t < \bar{C}^t \text{ and } C_i^{t-1} > \bar{C}^{t-1}; \\ 1, & \text{otherwise.} \end{cases} \tag{12}$$

In Equation (12), $\bar{C}^t$ is the average congestion value of all G-cells at the t-th cell inflation. If δ_i^t remains constant to 1, r_i^t will be monotonically increasing, meaning the inflation ratio of the cell will continue expanding even after it has been moved away from the congested region. This is unreasonable, so we introduce a cell deflation mechanism. When the congestion value C_i^t of cell i at the t-th iteration is lower than the average congestion value $\bar{C}^t$ of the t-th iteration, and the congestion value C_i^{t-1} of the $(t-1)$-th iteration is higher than the average value $\bar{C}^{t-1}$, we think that the cell has been moved away from a high congestion area to a low congestion area. At this time, a negative congestion correction factor is applied to the cell. We quantify the degree of cell deflation through the following approach. We calculate the ratio of $\bar{C}^t - C_i^t$ to the average congestion value $\bar{C}^t$, and then add the term $\frac{C_i^{t-1}-\bar{C}^{t-1}}{\bar{C}^{t-1}}$ to $\frac{\bar{C}^t - C_i^t}{\bar{C}^t}$ to get the absolute value $\left|\frac{C_i^{t-1}\bar{C}^t - C_i^t\bar{C}^{t-1}}{\bar{C}^{t-1}\bar{C}^t}\right|$, which represents the deflation strength of cell i, as shown in Equation (12).

C. Dynamic Pin-Accessibility Density Optimization

This subsection presents the dynamic pin-accessibility density optimization technique. Pin-accessibility is closely related to routability. Most work performs pin-accessibility optimization at detailed placement, in which local perturbations such as cell shifting, flipping, and swapping are used to optimize pin-accessibility [11, 12, 13]. These methods limit exploration space for pin-accessibility optimization. To improve solution quality, we consider pin accessibility optimization during global placement. As mentioned in Xplace-Route [8], during the placement process, cells on the M1 layer may move beneath the power grid rails on the M2 layer, making it difficult to connect the pins on these cells due to the constrained routing resources on M1. Xplace-Route [8] only adjusts cell density around PG rails before placement, while we focus on addressing pin accessibility throughout the entire global placement process. Our pin-accessibility algorithm is mainly divided into the following two steps:



(a) (b)

Fig. 4: PG rail selection for density adjustment in design matrix_mult_a [1], where the purple lines represent PG rails, the blue areas denote macro blocks and the green dashed line represents the expanded macro's bounding box. (a) presents all PG rails before PG rail selection. (b) gives the selected PG rails, the cell density of the related area will be adjusted.

1) PG rail selection for Pin-Accessibility: We find experimentally that, indiscriminately increasing the cell density of all regions covered by PG rails on the M2 layer is unreasonable. For example, the narrow region between macros are already highly congested, increasing the cell density around PG rails in the region would hinder cell spreading. Therefore, we perform a pre-processing step to identify regions where density can be adjusted effectively.

Specifically, we expand each macro's bounding box by 10%, and then project all bounding boxes and the PG rails onto the 2D plane, as shown in Fig. 4(a). These PG rails will be cut by the bounding boxes into some new PG rails. Then, we choose those PG rails whose length are at least 0.2 times the placement region's width (or height) in the horizontal (or vertical) direction, as shown in Fig. 4(b). The chosen PG rails are used for the next step of cell density adjustment.

2) Dynamic Pin-Accessibility Density Adjustment: Since multiple factors can affect congestion, it is impractical to focus only on the PG rails and ignore other factors. To reserve space for optimizing other factors and improve pin accessibility, we propose a precise density adjustment strategy concerning PG rails, aiming at enlarging density in congestion regions covered by the selected PG rails.

First, we evaluate the routing congestion on the area of the selected PG rails using the global router of [18]. Since we predefine the G-cell and bin to have the same dimension in Subsection II-B, we can map the congestion value of each G-cell to the corresponding bin at the same position. Then, **we selectively increase density only in highly congested bins**. This density adjustment dynamically adapts to congestion changes, preventing density increases across all selected PG rail areas and reserving space for further optimization.

Dynamic pin-accessibility-aware density primarily contains the original density D^{ori} and the M2-layer PG rails density D^{PG} (defined in Equation (13) and (14)). Cells located in the congested region covered by the selected PG rails will get a greater density gradient. This causes the cells at the region covered by congested PG rails to diffuse, reserving connection space for the pins in this area. The modified density D_b in bin b is: $D_b = D_b^{\text{ori}} + D_b^{PG}$, where

$$D_b^{\text{ori}} = \frac{1}{A_b}\sum_{i\in V} A_{c_i \cap b}, \qquad \forall b \in B, \tag{13}$$

$$D_b^{PG} = \frac{\eta_b(1+C_b)}{A_b}\sum_{i\in V_{PG}} A_{PG_i \cap b}, \qquad \forall b \in B. \tag{14}$$

TABLE I: Average results on the ISPD 2015 Contest Benchmarks [1]. The benchmarks removing the fence region constraints are marked by †.

Design	Xplace [16]					Xplace-Route [8]					Ours				
	DRWL/um	#DRVias	#DRVs	PT/s	RT/s	DRWL/um	#DRVias	#DRVs	PT/s	RT/s	DRWL/um	#DRVias	#DRVs	PT/s	RT/s
des_perf_1	1452606	569989	24977	5	869	1446525	566434	12768	7	657	1429056	566581	**10652**	30	568
des_perf_a†	2344941	560801	29875	5	180	2474885	578102	40701	20	200	2360122	569940	**10094**	20	3430
des_perf_b†	1817040	554677	19580	5	648	1807464	541944	1557	7	123	1783673	541130	**849**	21	89
edit_dist_a†	5670681	1006803	405858	6	748	5745638	1015369	424887	23	779	5737250	1034387	**359904**	19	801
fft_1	516059	186603	9249	3	415	515345	186516	4077	11	293	513993	185309	**3661**	26	299
fft_2	598828	187925	9334	3	117	627383	190944	1197	10	248	612371	190620	**1070**	14	328
fft_a	1090612	193181	5650	3	380	1144134	192291	925	11	296	1133734	192225	**790**	9	273
fft_b	1256346	205052	33875	3	191	1318002	217100	**14603**	12	255	1310187	216838	14710	9	263
matrix_mult_1	2708706	809962	80816	6	378	2656679	826886	15371	21	2236	2638437	820336	**10717**	25	2180
matrix_mult_2	2719259	840175	72311	6	419	2681555	860246	14422	22	2479	2676434	853057	**10651**	26	2975
matrix_mult_a	3892797	865702	34618	7	1590	3928555	849064	9380	10	2056	3938218	849468	**9304**	18	1704
matrix_mult_b†	3650328	790195	68415	6	338	3655611	783526	47964	8	374	3680673	789855	**37698**	18	323
matrix_mult_c†	3713674	813859	34226	6	1559	3681892	793568	9119	8	1205	3679849	795182	**8907**	19	686
pci_bridge32_a†	642272	148390	6553	3	631	656083	146580	4288	4	803	649379	146588	**3660**	9	1035
pci_bridge32_b†	978132	149169	2828	3	84	1008959	148146	339	4	53	998056	148052	**198**	9	38
superblue11_a†	40395036	5670612	866	33	1836	40373818	5716757	1047	49	1901	40209334	5652221	**669**	134	1670
superblue12	42759984	10825565	3276003	45	7513	43155488	10531913	22263	279	6091	49383559	12779668	**15113**	260	6796
superblue14	28028276	4330996	344	23	2555	28067336	4332033	367	43	2601	28097546	4289310	**310**	147	1511
superblue16_a†	31543711	4648757	4486	27	4163	31597720	4668672	4361	50	4303	31267452	4653129	**3387**	157	4004
superblue19	20830885	3637145	10097	18	2574	20849011	3618684	**6599**	54	1936	21126469	3647126	8280	109	1781
Avg. Ratio	1.00	1.00	5.00	0.25	1.37	1.00	0.99	1.40	0.63	1.07	1.00	1.00	1.00	1.00	1.00

* The DRV result of Xplace on superblue12 is not good, we exclude it from calculating its mean ratio to ensure a fair comparison of average ratios.
* Routing runtime may not reflect placement routability, as Innovus will end the detailed routing early if the placement has poor routability [8].

$$\eta_b = \begin{cases} 1, & \text{if } C_b > \bar{C}; \\ 0, & \text{otherwise.} \end{cases} \tag{15}$$

After adjusting the density map, we can calculate the density penalty in problem (5).

In the above Equations (13) and (14), $A_{c_i \cap b}$ represents the overlapping area between cell c_i and bin b, $A_{PG_i \cap b}$ represents the overlapping area between PG rail PG_i and bin b, and A_b represents the area of bin b. V is the set of cells, and V_{PG} is the set of selected PG rails during the PG rail selection process. C_b denotes the congestion value mapped at bin b, and $\bar{C}$ denotes the average congestion value across all bins. The PG rail density D_b^{PG} of bin b is controlled by the congestion value of bin b. As in Equation (15), if the congestion value C_b is higher than the average congestion value $\bar{C}$, the density increases with a weight of $1 + C_b$. This approach selectively increases the density in highly congested areas rather than adjusting the density in all regions where the selected PG rails cover, leaving additional space for other optimizations.

IV. Experimental Results

Our proposed framework is developed with PyTorch and CUDA, and deployed on the wirelength-driven global placer Xplace [16]. The experiments are performed on a Linux server with a 2.90GHz Intel Xeon CPU and a single Nvidia A800 GPU. The ISPD 2015 contest benchmarks [1] without fence-region constraints are used for the comprehensive evaluation and comparison.

We compare our analytical routability-driven global placement algorithm with Xplace [8] (without routability) and its routability-enhanced version Xplace-Route [8], a leading routability-driven placer. To ensure testing accuracy and maintaining the fairness of comparisons, the same version of the commercial tool Innovus [19] is used for global and detailed routing of the placement result by each compared placer. Specifically, we run each compared placer first and record the placement runtime (PT/s). Next, the placement result is input into Innovus to complete the global and detailed routing, and the post-routing results are recorded to measure routability. These results include detailed routing wirelength (DRWL/um), the number of DR vias (#DRVias), and the number of detailed routing violations (#DRVs). Moreover, we report the total routing runtime (RT/s). Table I lists the experimental results.

Table I shows that, our algorithm gains a significant improvement in routability compared to Xplace [8] and Xplace-Route [8]. Specifically, our algorithm achieves a 400% reduction in #DRVs compared to Xplace, meanwhile maintaining the same average DRWL and #DRVias. Compared to the leading routability-driven placer, Xplace-Route, our algorithm achieves a 40% reduction in #DRVs, while maintaining almost the same average DRWL and #DRVias.

TABLE II: Ablation Experiment.

Methods			DRWL/um	#DRVias	#DRVs
MCI	DC	DPA	Avg. Ratio	Avg. Ratio	Avg. Ratio
-	-	-	1.00	0.99	1.40
✓	-	-	0.99	0.99	1.27
✓	✓	-	1.00	1.00	1.12
✓	✓	✓	1.00	1.00	1.00

For the runtime, our algorithm spends $4.00\times$ and $1.59\times$ the total placement runtimes compared to Xplace and Xplace-Route, respectively. However, our algorithm's total global and detailed routing runtime is sped up by $1.37\times$ and $1.07\times$, compared to Xplace and Xplace-Route. These comparisons show that while our algorithm spends more time in placement to integrate routability techniques and reduce congestion, it ultimately reduces routing costs.

Further, we conduct ablation experiments to verify the effectiveness of our three techniques for alleviating routing congestion, with results shown in Table II. MCI refers to our algorithm with momentum-based cell inflation, DC to the algorithm with a differentiable congestion function, and DPA to the one with dynamic pin accessibility density adjustment. From the table, we observe that #DRVs decreases more as more techniques are applied, while the average DRWL and #DRVias remain nearly unchanged. This demonstrates that each technique improves routability without degrading wirelength or via count.

V. Conclusion

We have proposed an analytical framework for addressing routability in global placement. In this framework, a differentiable congestion function is developed by considering global routing congestion. Moreover, virtual cells in two-pin nets and selected multi-pin cells are introduced for guiding nets moving to mitigate global routing congestion. Furthermore, momentum-based cell inflation and dynamic pin-accessibility density optimization techniques are proposed for adjusting cell density to mitigate local routing congestion. Experimental results show that, our algorithm can effectively alleviate local and global routing congestion, improve routability significantly meanwhile maintaining comparable other metrics, compared to the leading routability-driven placer, Xplace-Route [8].

ACKNOWLEDGMENTS

This work was supported in part by the National Natural Science Foundation of China under Grant 62174033, the National Key Research and Development Program of China under Grant 2023YFA1011302, the Major Key Project of PCL (No. PCL2023A03), and the Natural Science Foundation of Fujian Province under Grants (No. 2024J09045). We thank Dr. Lixin Liu for helpful discussions and thank the authors of Xplace [8, 16] for their great contribution of the placer.

REFERENCES

[1] I. S. Bustany, D. Chinnery, J. R. Shinnerl, and V. Yutsis, "ISPD 2015 benchmarks with fence regions and routing blockages for detailed-routing-driven placement," in *Proc. ISPD*, 2015, pp. 157–164.

[2] J.-M. Lin, C.-W. Huang, L.-C. Zane, M.-C. Tsai, C.-L. Lin, and C.-F. Tsai, "Routability-driven global placer target on removing global and local congestion for VLSI designs," in *Proc. ICCAD*, 2021, pp. 1–8.

[3] Y. Lin, S. Dhar, W. Li, H. Ren, B. Khailany, and D. Z. Pan, "Dreamplace: Deep learning toolkit-enabled GPU acceleration for modern VLSI placement," in *Proc. DAC*, 2019, pp. 1–6.

[4] S. Liu, Q. Sun, P. Liao, Y. Lin, and B. Yu, "Global placement with deep learning-enabled explicit routability optimization," in *Proc. DATE*, 2021, pp. 1821–1824.

[5] C.-K. Cheng, A. B. Kahng, I. Kang, and L. Wang, "Replace: Advancing solution quality and routability validation in global placement," *IEEE TCAD*, vol. 38, no. 9, pp. 1717–1730, 2018.

[6] M. k Hsu, Y.-F. Chen, C.-C. Huang, S. Chou, T.-H. Lin, T.-C. Chen, and Y.-W. Chang, "Ntuplace4h: A novel routability-driven placement algorithm for hierarchical mixed-size circuit designs," *IEEE TCAD*, vol. 33, no. 12, pp. 1914–1927, 2014.

[7] J.-M. Lin, Y.-Y. Chang, and W.-L. Huang, "Timing-driven analytical placement according to expected cell distribution range," in *Proc. ISPD*, 2024, pp. 177–184.

[8] L. Liu, B. Fu, S. Lin, J. Liu, M. D. F. Wong, and E. F. Y. Young, "Xplace: An extremely fast and extensible placement framework," *IEEE TCAD*, vol. 43, no. 6, pp. 1872–1885, 2024.

[9] C.-C. Huang, H.-Y. Lee, B.-Q. Lin, S.-W. Yang, C.-H. Chang, S.-T. Chen, Y.-W. Chang, T.-C. Chen, and I. Bustany, "Ntuplace4dr: A detailed-routing-driven placer for mixed-size circuit designs with technology and region constraints," *IEEE TCAD*, vol. 37, no. 3, pp. 669–681, 2017.

[10] P. Spindler and F. M. Johannes, "Fast and accurate routing demand estimation for efficient routability-driven placement," in *Proc. DATE*, 2007, pp. 1–6.

[11] S. Kim and T. kim, "Pin accessibility-driven placement optimization with accurate and comprehensive prediction model," in *Proc. DATE*, 2022, pp. 778–783.

[12] J. Seo, J. Jung, S. Kim, and Y. Shin, "Pin accessibility-driven cell layout redesign and placement optimization," in *Proc. DAC*, 2017, pp. 1–6.

[13] Y. Ding, C. Chu, and W.-K. Mak, "Pin accessibility-driven detailed placement refinement," in *Proc. ISPD*, 2017, pp. 133–140.

[14] M.-K. Hsu, Y.-W. Chang, and V. Balabanov, "TSV-aware analytical placement for 3D IC designs," in *Proc. DAC*, 2011, pp. 664–669.

[15] J. Lu, P. Chen, C.-C. Chang, L. Sha, D. J.-H. Huang, C.-C. Teng, and C.-K. Cheng, "eplace: Electrostatics-based placement using Fast Fourier Transform and Nesterov's method," *ACM TODAES*, vol. 20, no. 2, pp. 1–34, 2015.

[16] L. Liu, B. Fu, M. D. F. Wong, and E. F. Y. Young, "Xplace: An extremely fast and extensible global placement framework," in *Proc. DAC*, 2022, pp. 1309–1314.

[17] W. Zhu, Z. Huang, J. Chen, and Y. C, "Analytical solution of Poisson's equation and its application to VLSI global placement," in *Proc. ICCAD*, 2018, pp. 1–8.

[18] S. Lin and M. D. F. Wong, "Superfast full-scale GPU-accelerated global routing," in *Proc. ICCAD*, 2022, pp. 1–8.

[19] Cadence Design Systems, "Cadence Innovus Implementation System," n.d. [Online]. Available: https://www.cadence.com.