Timing-Aware Fill Insertions with Design-Rule and Density Constraints

ABSTRACT

Metal fill insertion has become an essential step to reduce dielectric thickness variation and improve pattern uniformity, which is important in mitigating process variations, thereby achieving better manufacturing yield. However, metal fills could induce coupling capacitance, which is not often considered in existing works that typically focus more on pattern density uniformity, incurring significant problems in timing closure. In this paper, we address the timing-aware fill insertion problem that considers the total capacitance and density constraints simultaneously. First, initial metal fill insertion and design-rule-aware legalization are used to quickly obtain an initial fill insertion solution. Second, from critical conductors to powers/grounds in a circuit, we divide conductors into different equivalent paths and then construct a capacitance graph to globally reduce the capacitance of each equivalent path. Third, we present a density-aware coupling capacitance optimization method and a fast Monte Carlo based fill selection to further reduce the coupling capacitance between any pair of conductors. Finally, we present a density-aware fill deletion method to reduce the fill amounts. We evaluate the performance of our algorithm based on the benchmarks of the 2018 CAD Contest at ICCAD and its official contest evaluator. Compared with the first place team of the contest and the state-of-the-art work, experimental results show that our algorithm achieves the lowest total capacitance and the least fill amount for each benchmark.

1 INTRODUCTION

As IC process technology evolves into the deep nanometer era, smaller feature sizes and complex requirements become pervasive in modern circuit designs. For the advanced manufacturing process, chemical-mechanical polishing (CMP) pattern uniformity and dielectric thickness variation controls are keys for improving manufacturing yield. Post-layout metal fill insertions in conductor layers is an effective technique to achieve desired pattern uniformity and dielectric thickness variation control, and is important for mitigating process variations, thereby achieving better yield [1].

However, metal fills could induce coupling capacitance which might incur significant circuit delay and thus problems in timing closure [9, 12]. Therefore, it is important to reduce the impacts of coupling capacitance during metal fill insertion. Recently, the metal fill insertion problem has been studied extensively with different objectives, such as coupling capacitance, density variation, and fill amount.

To reduce the effect of coupling capacitance on circuit performance, Xiang *et al.* [14] developed a coupling-constrained dummy-fill analysis algorithm, which identifies feasible locations for dummy fills such that the fill induced coupling capacitance can be bounded within a given threshold for each conductor. Deng *et al.* [6] presented a method for dummy metal insertion, which can optimally trade off lithography cost and coupling capacitance. To improve the layer density variation, Wu *et al.* [13] derived a linear programming formulation for dummy-fill synthesis, which also considers density gradient besides pattern density. Lin *et al.* [11] proposed an efficient dummy-fill insertion framework based on geometric properties to optimize multiple objectives simultaneously, including coupling capacitance and density variation and gradient. To minimize the fill amount, Chen *et al.* [5] presented an efficient hybrid hierarchial filling method to optimize both the density variation and the fill amount.

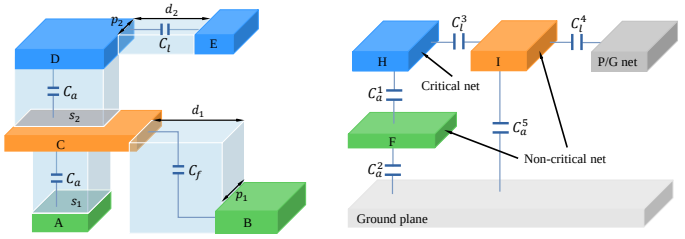
However, none of these works handles the timing-aware fill insertion problem by considering coupling capacitance, fill amount, and density constraints simultaneously. Specifically, these works optimize density gradient or the fill amount, and treat coupling capacitance as a constraint, ignoring the effect of the coupling capacitance between the conductors on the entire circuit. This insufficiency could have a significant impact on the delay of a circuit, especially on critical nets. However, calculating the total capacitance is very complicated and time-consuming. What is worse, no matter how small a buffer region we specify, the impact on timing cannot be evaluated accurately without parasitics extraction.

Therefore, it is desirable to develop an effective and efficient algorithm for timing-aware metal fill insertion, as the aim of the 2018 CAD Contest at ICCAD on Timing-Aware Fill Insertion [1]. The recent work [8] considered this contest problem and presented a window-based fill insertion method. Since the method optimizes total capacitance of critical nets within a window, which tends to be local, the solution quality may be limited.

In this paper, we propose an effective algorithm to solve the timing-aware fill insertion problem with design-rule and density constraints. The main contributions of our work are summarized as follows:

- We address the timing-aware fill insertion problem, where the total capacitance of critical nets, the number of fills inserted, and the density constraint of a circuit are considered simultaneously.
- We propose an initial fill insertion scheme and a design-rule-aware legalization method to quickly obtain an initial metal fill insertion solution.
- From critical conductors to powers/grounds in a circuit, we divide conductors into different equivalent paths and then construct a capacitance graph to globally reduce the capacitance of each equivalent path.
- We present a density-aware coupling capacitance optimization method and a fast Monte Carlo based fill selection to further reduce the coupling capacitance between any pair of conductors.
- Experimental results show that our algorithm is effective for the timing-aware fill insertion optimization. Compared with the first place team of the contest and the state-of-the-art work, our algorithm resolves all the design-rule and density violations, and achieves the lowest total capacitance and the least fill amount on each benchmark.

The rest of this paper is organized as follows. Section 2 describes the calculation of coupling capacitance and the total capacitance and gives the problem statement. Section 3 presents



(a) (b)

Figure 1: (a) Classification of coupling capacitances, where conductors A and B are on metal layer 1, conductor C is on layer 2, and conductors D and E are on layer 3. (b) Total capacitance calculation, where conductor F is on layer 1, conductors H and I are on metal layer 2.

our algorithm. Section 4 shows the experimental results. Finally, conclusions are drawn in Section 5.

2 PRELIMINARIES

In this section, we first introduce the classification, calculation, and shielding effects of coupling capacitance. Then, we describe the method of calculating the total capacitance of a net. Finally, we give the density constraints and the problem statement.

2.1 Coupling Capacitance

The coupling capacitance at each node in a layout is estimated as the sum of three components: the area capacitance, the lateral capacitance, and the fringe capacitance [3]. The area and fringe capacitances are caused by conductors being placed in different layers, and the lateral capacitance by two conductors in the same layer.

For any two conductors in different layers, they form an area capacitance if they meet three conditions as follows: (1) they are placed in different metal layers; (2) their projections on the ground plane have an overlapped region; and (3) for the overlapped region between the two conductors, no intermediate-layer metal appears. As shown in Figure 1(a), there is an area capacitance between conductors A and C, and between the conductors C and D. Suppose that the two conductors A and C have an overlapped area of s. Then, the area capacitance C_a between them is calculated by

$$C_a = U_{l_1 l_2}(s) \times s, \tag{1}$$

where $U_{l_1 l_2}(s)$ is modeled as a piecewise-linear function of the overlapped area s, which represents the area capacitance per unit area between layers l_1 and l_2.

As shown in Figure 1(a), it is also possible to form a fringe capacitance between the conductors B and C in different layers. Assume that the length of the parallel edges between the conductors is p, and the distance between the parallel edges is d. Then, the fringe capacitance C_f between the conductors B and C is calculated by

$$C_f = \begin{cases} U_{l_1 l_2}(d) \times p + U_{l_2 l_1}(d) \times p, & \text{if } d \geq 0; \\ 0, & \text{if } d < 0, \end{cases} \tag{2}$$

where $U_{l_1 l_2}(d)$ and $U_{l_2 l_1}(d)$ denote the fringe capacitance per unit distance between layers l_1 and l_2, and they are also modeled as a piecewise-linear function of distance d.

For two conductors D and E on the same layer as shown in Figure 1(a), there may be a lateral capacitance if they overlap in a certain direction. Assume that the length of the parallel edges between the two conductors is p, and the distance between the parallel edges is d, then the lateral capacitance between conductors D and E is calculated by

$$C_l = U_l(d) \times p, \tag{3}$$

where $U_l(d)$ represents the lateral capacitance per unit distance, which is a piecewise-linear function of distance d.

For any two conductors A and B, the area capacitance is formed by the overlapped area in the vertical direction in which the two conductors can directly "face" each other. The "face" means that there are no other conductors between A and B. Otherwise, the area capacitance between A and B is shielded, and the overlapped area s would be divided into two parts: the shielded area and the unshielded area. The shielding of lateral capacitance is similar to the area capacitance in the parallel direction. In addition, if two conductors A and B are placed in different layers, the fringe capacitance between A and B will be shielded when there is a conductor in the intermediate layer. A detailed description can be found in [1].

2.2 Total Capacitance Calculation

The total capacitance from a net to a power or ground net can be calculated by the series-parallel connection model. As shown in Figure 1(b), the total capacitance of the net C can be calculated by

$$C_{total} = \frac{C_a^1 \times C_a^2}{C_a^1 + C_a^2} + \frac{C_l^3 \times C_a^5}{C_l^3 + C_a^5} + \frac{C_l^3 \times C_l^4}{C_l^3 + C_l^4}. \tag{4}$$

This calculation method is suitable for a small-scale circuit.

For a large-scale circuit, the total capacitance of a net N_i can be calculated by using a matrix-based method [3]. The corresponding numerically simulated capacitance matrix is

$$\hat{C} = \begin{bmatrix} c_{11} & -c_{12} & \cdots & -c_{1n} \\ -c_{21} & c_{22} & \cdots & -c_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ -c_{n1} & -c_{n2} & \cdots & c_{nn} \end{bmatrix}, \tag{5}$$

where c_{ii} denotes the sum of all the capacitance values associated with the net N_i, and $c_{ij} (i \neq j)$ the coupling capacitance between nets N_i and N_j. Then, the total capacitance $C_{total}(N_i)$ of the net N_i is calculated by

$$C_{total}(N_i) = c_{ii} - \hat{p}^T \cdot M_{ii}^{-1} \cdot \hat{p}, \tag{6}$$

where $\hat{p} = (c_{i1}, ..., c_{i,i-1}, c_{i,i+1}, ..., c_{in})^T$ is an $(n-1)$-dimensional vector. M_{ii} is the cofactor matrix of order $(n-1) \times (n-1)$ for c_{ii} in matrix $\hat{C}$.

2.3 Problem Statement

Density Constraint: To reduce the variations in the height and thickness of the metal and achieve layer uniformity, foundries enforce a CMP design requirement called the *layer density constraint* [10].

To better measure the density, as shown in Figure 2, a sliding window-based approach is proposed to detect whether the density

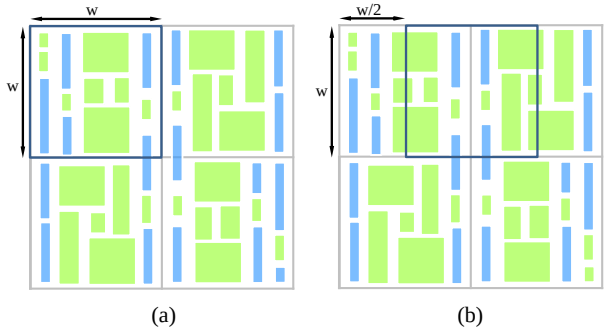


Figure 2: Density calculation based on sliding window.

constraint is violated. Suppose that the width of a window W_i is w, and a new window is created by sliding in the step of $\frac{w}{2}$. The density of the window W_i is calculated by

$$D_{l,W_i} = \sum_{S \in W_i} S/w^2, \tag{7}$$

where l and S represent the metal layer and the metal area enclosed inside the window W_i, respectively. During the insertion of fills, we must ensure that the density of each window on layer l meets the density constraint.

For the metal fill insertion problem, our objective is to minimize the total capacitance of critical nets while satisfying the density constraints. Formally, the problem can be stated as follows:

- **Timing-Aware Fill Insertion Problem:** Given a detailed routing result, the value of unit capacitance for each layer, the window size and density constraints, insert fills in each metal layer, such that the total capacitance of critical nets and fill amount are minimized, and both of the design rules (the minimum spacing, the minimum and maximum widths of conductors, etc.) and density constraints within each window are satisfied.

3 OUR ALGORITHM

Our algorithm for the timing-aware fill insertion is summarized in Figure 3. It consists of three major stages: (1) initial fill insertion and legalization, (2) total capacitance transformation, and (3) reducible edge capacitance reduction.

In the initial fill insertion and legalization stage, we first construct slot and track structures for each half-window to store the information about the metal conductors. Then, an inequality is derived for quickly finding free tracks in each slot. And then, in each slot, with a free track as the center, fills of the same height are inserted into the free tracks. Finally, a series of legalization operations are used to modify the initial fill insertion result to satisfy all the constraints.

In the total capacitance transformation stage, we first present a concept of equivalent path for estimating the total capacitance. Then, to reduce the capacitance of each equivalent path globally, we propose a capacitance graph for selecting reducible edges of each equivalent path. Finally, reducing the capacitance problem in an equivalent path is transformed into a problem of reducing the coupling capacitance between the conductors.

In the reducible edge capacitance reduction stage, we first establish a density assignment model to obtain the largest deletable area for each half-window. Then, we present a fast Monte Carlo based fill selection algorithm to select the fills which are responsible for reducing the coupling capacitances of the corresponding reducible edges. Further, we design a coupling capacitance reduction strategy to improve the coupling capacitance on the capacitance edge. Finally, a density-aware fill deletion method is applied to reduce the fill amounts. The details of each stage are given in the following subsections.

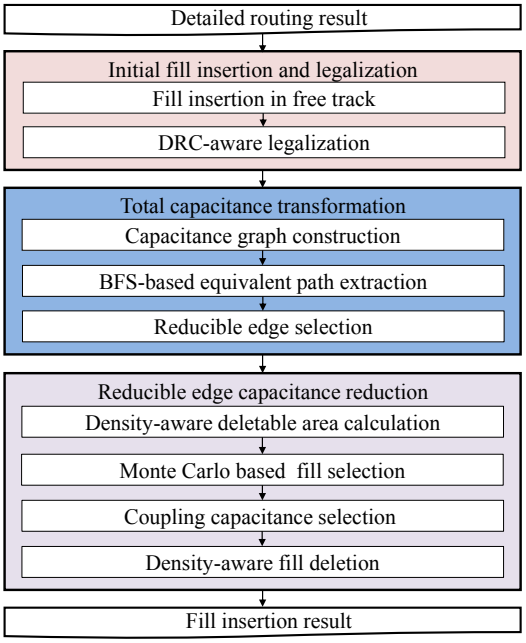


Figure 3: Framework of our algorithm.

3.1 Initial Fill Insertion and Legalization

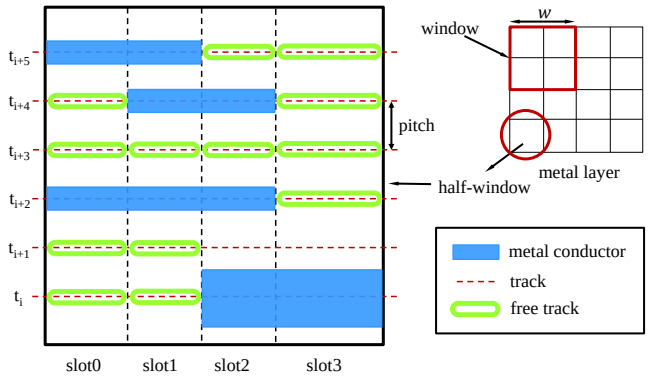


Figure 4: Construction of the track and slot structures for a half-window.

Based on the previous works [4, 6, 10], we first create a track structure based on the input positions of metal conductors of each metal layer as in Figure 4. Then, windows generated by sliding window are divided into a series of smaller half-windows. A half-window is a square region of width $\frac{w}{2}$.

Obviously, a window consists of four half-windows. Based on the previous work [14], we first construct the slot structure for every half-window. Then, for any slot in a half-window, suppose that there is a metal conductor $c_i(x_i^l, y_i^l, w_i, h_i)$ in this slot, where $(x_i^l,\ y_i^l)$ is the bottom-left coordinate of the metal conductor on layer l, and w_i and h_i are the width and height of the conductor, respectively. For the conductor c_i, if it satisfies the following two conditions, then track t_k is marked as occupied:

$$\begin{array}{l}(a) \text{ if metal layer } l \text{ is horizontal,}\\ \quad y_i^l < \frac{P_l(t_k)+P_l(t_{k+1})}{2} \text{ or } y_i^l + h_i > \frac{P_l(t_{k-1})+P_l(t_k)}{2};\\ (b) \text{ if metal layer } l \text{ is vertical,}\\ \quad x_i^l < \frac{P_l(t_k)+P_l(t_{k+1})}{2} \text{ or } x_i^l + w_i > \frac{P_l(t_{k-1})+P_l(t_k)}{2},\end{array} \tag{8}$$

where $P_l(t_k)$ represents the y-position of track t_k if metal layer l is horizontal; otherwise, it denotes the x-position of track t_k. Thus, we can quickly find free tracks by traversing the metal conductors in each slot. When all free tracks in the slots are determined, a series of fills of the same height are generated for each slot with centers on the free tracks. Meanwhile, during insertion of these fills, the minimum spacing constraint in the direction perpendicular to layer l must be guaranteed. Finally, we can get an initial fill insertion result for each metal layer.

In this result, the minimum and maximum widths and the minimum spacing constraint in the direction parallel to the metal layer may be violated. Therefore, to obtain a legal solution and improve the density of each half-window, we first merge the inserted fills as many as possible in both the horizontal and vertical directions. Then, we propose a legalization method to satisfy the minimum and maximum widths and the minimum spacing constraints in the direction parallel to the metal layer.

Algorithm 1 gives an overview of our legalization method. In Line 1, we generate a temporary fill $c_t(x_t^l, y_t^l, w_t, h_t)$ for updating the inserted fills. Then, we split the fills that do not meet the maximum width constraint in the direction parallel to the metal layer (Lines 2–18). In Line 19, since the minimum spacing constraint may not be satisfied in the direction parallel to the metal layer, we need to traverse adjacent conductors on the same track. If there is a spacing violation between two adjacent conductors, we need to adjust the distance between them. In Line 20, we traverse the fills in F and then remove the fills that do not satisfy the minimum width constraint of fills.

Algorithm 1 Legalization

Input: F_0: initial fill result, M^l: maximum conductor width on metal layer l; $\hat{L}, \hat{B}$: left and bottom boundaries of circuit;
Output: F_l: legalization result;

```
1: generate a temporary fill c_t(x_t^l, y_t^l, w_t, h_t), F_l := ∅;
2: for each fill c_i(x_i^l, y_i^l, w_i, h_i) ∈ F_0 do
3:    if x_i^l + w_i > M^l then
4:       do
5:          x_t^l := x_i^l, y_t^l := y_i^l, h_t := h_i, w_t := ⌈(x_i^l − L̂)/M^l⌉ × M^l − x_i^l;
6:          x_i^l := x_i^l + w_t, w_i = w_i − w_t;
7:          F_l := F_l ∪ {c_t};
8:       until w_i < 0
9:    else if y_i^l + h_i > M^l then
10:      do
11:         x_t^l := x_i^l, y_t^l := y_i^l, w_t := w_i, h_t := ⌈(y_i^l − B̂)/M^l⌉ × M^l − y_i^l;
12:         y_i^l := y_i^l + h_t, h_i = h_i − h_t;
13:         F_l := F_l ∪ {c_t};
14:      until h_i < 0
15:   else
16:      F_l := F_l ∪ {c_i};
17:   end if
18: end for
19: adjust the distance between any two adjacent metals such
    that they meet the minimum spacing constraint;
20: delete all fills that do not meet the minimum width constraint.
```

3.2 Total Capacitance Transformation

After the legalization stage, we can get an initial metal fill insertion result with all the constraints being satisfied. From Equation (6), we know that calculating the total capacitance of a circuit is time-consuming, especially for a large-scale problem, since it includes inversion of a matrix. To address this challenge, we propose an equivalent path based approach to approximate the total capacitance, which are based on the following concepts of capacitance edge and equivalent path.

Definition 1 (Capacitance Edge). *For any two conductors A and B, if there is a coupling capacitance between them, we use an edge to connect the two conductors. This edge is called a* capacitance edge.

From Section 2.1, we know that three coupling capacitances may be formed between any two metal conductors. Therefore, coupling capacitance edges can be divided into three categories: the area capacitance edge (C_a), the fringe capacitance edge (C_f), and the lateral capacitance edge (C_l).

Definition 2 (Equivalent Path). *If there is a path P formed by a continuous capacitance from conductor A to ground (P/G nets), then the path P is called an equivalent path from conductor A to ground (P/G nets).*

To reduce the total capacitance of a critical net c_i, we should reduce the total capacitance of the equivalent path from c_i to ground (P/G nets) corresponding to the critical net c_i. From Definition 2, we know that an equivalent path is a series circuit. For the calculation of the total capacitance C_{total} of a series circuit, we have

$$\frac{1}{C_{total}} = \frac{1}{C_1} + \frac{1}{C_2} + \cdots + \frac{1}{C_n}, \tag{9}$$

where $C_1, ..., C_n$ are the values of coupling capacitance between any pair of conductors. Obviously,

$$C_{total} \leq \min\{C_1, C_2, ...C_n\}. \tag{10}$$

From Inequality (10), C_{total} is dominated by the smallest capacitance in an equivalent path. Therefore, to reduce the capacitance of an equivalent path, we only need to focus on reducing the capacitance of an edge in this path. When two conductors which make up the capacitance edge are from P/G nets, critical nets, or non-critical nets, we know that the capacitance of this capacitance edge is difficult to reduce. To distinguish these capacitance edges, we introduce the concept of *reducible edge* as follows:

Definition 3 (Reducible Edge). *If one of two conductors that make up the capacitance edge c_e is a fill, then c_e is called a* reducible edge.

Note that there may be many equivalent paths from a conductor to ground or P/G nets. In order to reduce the total capacitance from a critical net to ground or P/G nets globally, we propose the *capacitance graph* as follows.

Definition 4 (Capacitance Graph). *A capacitance graph $CG = (V, E, W)$ from conductor A to B is an undirected and weighted graph, where vertex $v_i \in V$ represents a conductor. If there is a capacitance edge between vertices v_i and v_j, then there exists an edge $e_{ij} \in E$ between them. $w_{ij} \in W$ represents the weight of edge $e_{ij} \in E$, which is the number of equivalent paths from conductor A to B passing through the edge e_{ij}.*

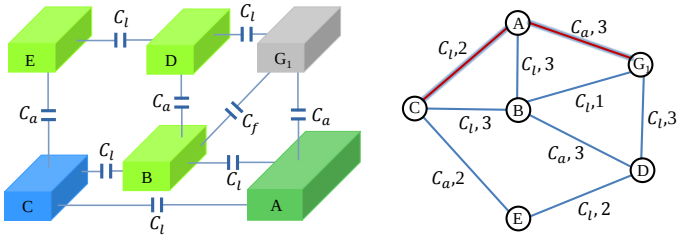


Figure 5: (a) An actual filled example. (b) Constructed capacitance graph: each edge is marked with the type of the capacitance edge and the value of the weight.

Figure 5 depicts a capacitance graph example from the critical conductor C to P/G net G_1. In this figure, conductor A is a non-critical net and conductors B, D, and E are fills. Therefore, the capacitance edges C_{AC} and C_{AG_1} are non-reducible edges. In Figure 5(a), we can find seven equivalent paths from conductor C to G_1, which are paths $CBAG_1$, CBG_1, $CBDG_1$, $CABDG_1$, CAG_1, $CEDBAG_1$, and $CEDG_1$, respectively. According to Definition 4, we can quickly construct the capacitance graph of Figure 5(a) as shown in Figure 5(b). Then, we can quickly assign the weight of each capacitance edge in Figure 5(b) by the obtained equivalent paths. From Figure 5(b), if we can reduce the capacitance of the capacitance edge e_{CB}, the capacitance of the three equivalent paths $CBAG_1$, CBG_1, and $CBDG_1$ can be reduced.

Based on the above observation, we devise a scheme to find a reducible edge in a capacitance graph so that the selected reducible edges can completely cover these paths, while minimizing the number of selected reducible edges. Algorithm 2 details the reducible edge selection.

In Line 2 of Algorithm 2, we first construct a cube centered at the critical conductor c with the width and length of w, and the depth of the largest layer. Then, we pick all P/G conductors or grounds in this cube. Further, we construct a capacitance graph CG from conductor c to each picked P/G conductors or ground in the cube, and find the set EP of all equivalent paths from pg to c by using the Breadth-First Search (BFS) algorithm (Lines 4–5) in the constructed capacitance graph CG. In Line 6, we assign weights to the edges of CG when EP is determined. Furthermore, in Lines 7-12, we adopt a greedy method to select reducible edges in the constructed graph CG. When a reducible edge is selected, all equivalent paths passing through this edge

Algorithm 2 Reducible Edge Selection

Input: C: a set of critical conductors;
Output: RC: the set of reducible edges needing reduction of coupling capacitance and the corresponding weights;
1: **for** each critical conductor $c \in C$ **do**
2: find the set PG of ground or P/G conductors in a given cube;
3: **for** each $pg \in PG$ **do**
4: construct a capacitance graph CG from c to pg;
5: find the set EP of all equivalent paths from c to pg;
6: calculate the weight of each edge in CG;
7: **for** each path $ep \in EP$ **do**
8: find a reducible edge re with the largest weight in ep;
9: find a set P_t of all paths in EP passing through edge re;
10: $RC := RC \bigcup \{re\}$, $EP := EP \setminus P_t$;
11: update the weights of graph CG;
12: **end for**
13: **end for**
14: **end for**

will be removed from EP, and then the weight of each edge of CG is recalculated based on the existing equivalent paths.

Then, by reducing the coupling capacitances of these selected reducible edges, we can reduce the capacitance of each equivalent path from each critical conductor to each P/G conductor or ground, and finally, we can reduce the total capacitance of all critical nets. The next subsection will introduce our method of reducing the coupling capacitance of a reducible edge.

3.3 Reducible Edge Capacitance Selection

After all reducible edges having been selected in the above subsection, we proposed a linear programming (LP) model to obtain the largest deletable area of each half-window. Then, a Monte Carlo based fill selection algorithm is presented to select the fills which are responsible for reducing the coupling capacitances of the corresponding reducible edges. And then, a coupling capacitance reduction strategy is proposed to improve the coupling capacitance on the capacitance edge. Finally, a density-aware fill deletion method is introduced to reduce the number of fills. The details are as follows.

3.3.1 Density-aware Deletable Area Calculation. To obtain as much deletable area as possible for each half-window, that is, to make the remaining density of each half-window as small as possible after deletion, we formulate it to a density assignment problem. In each layer l, this problem can be modeled as a linear programming (LP) problem as follows:

$$\max \quad \sum_{i=1}^{m}\sum_{j=1}^{n}(\tilde{\rho_{ij}} - \rho_{ij}) \tag{11}$$

$$\text{s.t.} \quad \rho_{min} \le \frac{1}{4}\sum_{s=i}^{i+1}\sum_{t=j}^{j+1}\rho_{st} \le \rho_{max}, \quad 1 \le i < m, 1 \le j < n; \tag{11a}$$

$$0 \le \rho_{ij} \le \tilde{\rho_{ij}}, \qquad 1 \le i \le m, 1 \le j \le n, \tag{11b}$$

where m and n represent the number of half-windows in the horizontal and vertical directions, respectively. ρ_{ij} and $\tilde{\rho_{ij}}$ represent the deleted density and the initial density in the window W_{ij} of metal layer l, respectively. In the above LP, Constraint (11a) guarantees that window W_{ij} does not violate density constraints. ρ_{min} and ρ_{max} denote the minimum and maximum density constraints of metal layer l, respectively. Constraint (11b) guarantees that the final deleted density of each window will not exceed its initial density.

Since the number of half-windows within a metal layer is very small, we can quickly solve Problem (11) by calling an LP solver [2]. After solving Problem (11), we can obtain the area which can be deleted for each half-window.

3.3.2 Monte Carlo Based Fill Selection. When all reducible edges haven been selected, these edges will form a graph G consisting of many connected components. In graph G, each vertex represents a conductor (fill, critical net, non-critical net, or P/G net), and the weight of an edge is from Algorithm 2, which denotes the importance of the reducible edge.

For each reducible edge in graph G, we aim to minimize the reducible edge capacitance by adjusting the corresponding fill on this edge. For this purpose, we select reducible edges of graph G as many as possible based on the definition of reducible edge graph (REG) as follows.

Definition 5 (Reducible Edge Graph). *Suppose that graph G is a subgraph of a capacitance graph (CG) induced from the selected reducible edges (RC). We replace every undirected reducible edge e_{ij} by two directed reducible edges e^d_{ij} and e^d_{ji}. The weights of e^d_{ij} and e^d_{ji} equal the weight of e_{ij}. If vertex i is not a fill, then we delete all existing edges e^d_{ij}. As a result, we obtain a reducible edge graph $REG = (V, E^d, W)$.*

Then, the problem of selecting reducible edges of the graph G as many as possible is modelled as the integer linear programming (ILP) problem as follows:

$$\max \quad \sum_{i\in V}\sum_{j\in V} e^d_{ij} w_{ij} \tag{12}$$

$$\text{s.t.} \quad \sum_{j\in V} e^d_{ij} \le 1, \qquad \forall\, i \in V; \tag{12a}$$

$$e^d_{ij} + e^d_{ji} \le 1, \qquad \forall\, i, j \in V; \tag{12b}$$

$$e^d_{ij} \in \{0, 1\}, \qquad \forall\, i, j \in V, \tag{12c}$$

where $e^d_{ij} = 1$ means that, the fill corresponding to vertex i is responsible for reducing the coupling capacitance of the reducible edge e_{ij}; otherwise, the fill corresponding to vertex i cannot be used to reduce the coupling capacitance of edge e_{ij}. w_{ij} represents the weight between vertices i and j. In Problem (12), Constraint (12a) guarantees that the fill corresponding to vertex i can only be used to reduce the capacitance of the edge associated with it. Constraint (12b) ensures that the capacitance of the edge can only be reduced by the two conductors on this edge. Since the size of each component of REG is not too large, we can quickly solve Problem (12) by calling an ILP solver.

Although Problem (12) can be solved optimally, the total capacitance of a circuit after fill insertion corresponding to an optimal solution of Problem (12) may not be minimum. Given an optimal solution of Problem (12), to further improve the total capacitance after the reduction based on the solution, we use Equation (6) to estimate the total local capacitance change, and select the reducible edge with the largest total local capacitance reduction.

However, calculation of the local capacitance is based on the method described in Section 2.2. In Equation (6), the calculation of inverse M_{ii} is time-consuming; for example, the Gaussian or the Gauss-Jordan elimination takes $T(n) = O(n^3)$ time. To speed up computation of the inverse matrix, we use the Monte Carlo approach proposed in [7]. Define an iteration of order i as a function of the following form

$$u^{(k+l)} = F_k(A, I, u^{(k)}, u^{(k-1)}, \ldots, u^{(k-i+1)}),$$

where $u^{(k)}$ is the m-component vector obtained from the k-th iteration. It is desired that

$$u^{(k)} \longrightarrow u = A^{-1} \; as \; k \longrightarrow \infty.$$

In the algorithm, we compute the iterations $u^{(q)}$, $1 \le q \le k$ using the Monte Carlo approach with an additional statistical error. In practice, the truncation parameter k is obtained from the condition that the difference between the stochastic approximations of two successive approximations is smaller than a given sufficiently small parameter ε [7]. The time complexity of this algorithm is $O(km^2)$, where m is the dimension of the matrix. Since we only use the inverse matrix to determine whether the capacitance has been improved, it does not need to be very accurate for the results to further speed up the algorithm. For sparse matrices, the algorithm can be more efficient.

Finally, we first put these conductors into their corresponding half-windows according to their positions. Then, the deletable area of each half-window is evenly distributed to each conductor according to the areas of all selected conductors in the half-window. At last, by analyzing unit capacitances scaled from the industrial process, we propose different density-aware approaches to reduce capacitances of selected reducible edges of different coupling capacitance types in the next subsection.

3.3.3 Coupling Capacitance Reduction. As mentioned in Section 2.1, the coupling capacitance between conductors can be mainly divided into three types: (1) area capacitance; (2) lateral capacitance; (3) fringe capacitance. Since both the lateral and fringe capacitances are determined by the length of the parallel edges and the distance between the parallel edges, our coupling capacitance reduction scheme is mainly divided into the following two types.

Area capacitance reduction: Any two metal conductors located in different layers may form an area capacitance as shown in Figure 6(a). In this figure, assume that fill A placed on layer l is responsible for reducing the area capacitance between fill A and conductor B. Suppose that S^l is the minimum spacing constraint of layer l, and suppose that the width and length of the overlap region are $\hat{w}$ and $\hat{h}$, respectively.

According to Equation (1), if we want to reduce the area capacitance between fill A and conductor B, we should reduce the overlapping area between them by splitting fill A into A_1 and A_2. However, when the overlapping area between fills A_1 (or A_2) and B is reduced to zero, the area capacitance between A_1 (or A_2) and B will be changed to the fringe capacitance. See Figure 6 for an illustration. Therefore, to deal with this issue effectively, we propose two rules for reducing the area capacitance by studying the capacitance table of per unit area as follows:

(1) *Rule 1.* If $\hat{w} \geq S^l + 2$, then we separate fill A into two parts satisfying the minimum width and spacing constraints. Thus, a large area capacitance between A and B is reduced to two smaller area capacitances, as shown in Figure 6(b).

(2) *Rule 2.* If $\hat{w} < S^l + 2$, then we separate fill A into two parts satisfying the minimum width and spacing constraints. Thus, a large area capacitance between A and B is reduced to one smaller area capacitance and one fringe capacitance, as shown in Figure 6(c).

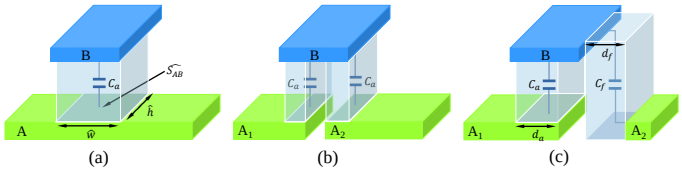


Figure 6: Introduction of area capacitance and rules for area capacitance reduction. (a) Area capacitance. (b) *Rule 1*. (c) *Rule 2*.

By using *Rule 1* and *Rule 2*, the coupling capacitance between fill A and conductor B is reduced by considering the assigned deletable area. In the reduction process, if any constraint is violated, the reduction operation is stopped.

Lateral and fringe capacitance reduction: From Section 2.1, the value of lateral or fringe capacitance is determined by the length p of the parallel edges and the distance d between the parallel edges. As shown in Figures 7 (a) and (b), assume that fill B is responsible for reducing the lateral (or fringe) capacitance between conductor A and fill B. From Equations (2) and (3), the capacitance per unit distance $U(d)$ is a piecewise-linear function of distance d. Thus, we have

$$U(d) = a_i \times d + b_i, d \in [L_i, U_i], \tag{13}$$

where L_i and U_i are the lower and upper bounds of the i-th distance interval, respectively, a_i and b_i are the linear coefficients of the i-th distance interval $[L_i, U_i]$.

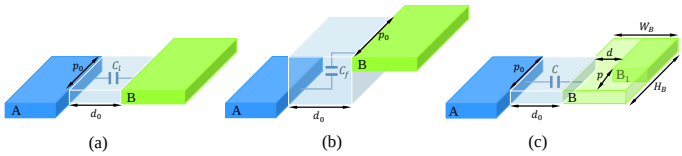


Figure 7: An example of lateral/fringe capacitance reduction. (a) Lateral capacitance. (b) Fringe capacitance. (c) Top view of (a) or (b): lateral/fringe capacitance reduction.

According to the input information of unit capacitance, we can divide the distance between a conductor A and a fill B into a series of consecutive intervals. In each interval, both of the lateral and fringe capacitance can be reduced by adjusting the length p of parallel edges and the distance d between the parallel edges, as shown in Figure 7(c). Therefore, for the i-th distance interval $[L_i, U_i]$, we can model the lateral or fringe capacitance reduction problem as a nonlinear programming (NLP) problem as follows:

$$\begin{aligned}
\min \quad & C(d,p) = [a_i \times (d_0 + d) + b_i] \times (p_0 - p) && (14)\\
\text{s.t.} \quad & (H_B - p) \times d + W_B \times p \leq \tilde{D}_B; && (14a)\\
& d + m_w \leq W_B; && (14b)\\
& p + m_w \leq H_B; && (14c)\\
& L_i \leq d + d_0 \leq U_i; && (14d)\\
& d, p \geq 0; && (14e)
\end{aligned}$$

where W_B and H_B denote the width and height of fill B, d_0 and p_0 represent the original distance and the original length of parallel edges between conductor A and fill B, m_w denotes the minimum width constraint of the layer where fill B is located, and $\tilde{D}_B$ is the available deletable area of B.

In the NLP (14), constraint (14a) ensures that the area deleted by fill B does not exceed its available amount. Constraints (14b) and (14c) guarantee that the adjusted distance d and the length p of parallel edges would not violate the minimum width constraint. By solving Problem (14), we can obtain the optimal distance d and the length p of parallel edges between conductor A and fill B. However, since the objective of Problem (14) is not a convex function and the constraint (14a) is not a linear constraint, the solution (d^*, p^*) of Problem (14) cannot be easily obtained. To optimize Problem (14), we first fix p to 0, and then Problem (14) can be solved by the following formula:

$$d^* = \begin{cases} L_i - d_0, \text{ if } a_i \geq 0; \\ \min\{U_i - d_0, \frac{\tilde{D}_B}{H_B}, W_B - m_w\}, \text{ if } a_i < 0. \end{cases} \tag{15}$$

Then in (15), fixing the value of d to d^*, we can obtain an optimal value of p as

$$p^* = \min\{\frac{\tilde{D}_B - d^* \times H_B}{W_B - d^*}, H_B - m_w\}. \tag{16}$$

By using formulas (15) and (16), we obtain an approximate solution of Problem (14) in the interval $[L_i, U_i]$. Then, we can establish a series of the NLP model for all feasible distance intervals between conductor A and fill B, and then we can obtain a series of coupling capacitance values $C(d,p)$. Finally, we choose the lowest one.

3.3.4 Density-aware Fill Deletion. After the reduction of the coupling capacitances of reducible edges, the windows might still have large deletable areas. To further reduce the total capacitance of the critical nets and the fill amount, we derive a fill deletion algorithm considering the density of a half-window.

Algorithm 3 gives an overview of our fill deletion algorithm. Firstly, we sort all fills in ascending order according to their respective areas in Line 1. Then, for each fill, we calculate the half-window occupied by the fill in Line 3. Further, we check whether the fill will violate the minimum density constraint when it is deleted. If the density constraint is violated, the deletion operation is skipped for this fill in Lines 5-11. Finally, the extra fills will be deleted, and the corresponding deletable area in each half-window will be updated.

4 EXPERIMENTAL RESULTS

We implemented our algorithm for the timing-aware fill insertion with density constraints using the C++ programming language, and tested it on the benchmarks provided by the 2018 CAD

Algorithm 3 Density-Aware Fill Deletion

Input: $D_A(W)$: deletable area in a half-window W;
F_1: set of fills;
Output: F_f: final fill insertion result;

```
sort the elements in F_1 in ascending order by area;
for each fill f ∈ F_1 do
    compute the set W̄ of half-windows occupied by fill f;
    set flag := true;
    for each half-window W_i ∈ W̄ do
        calculate the overlap area A(f, W_i) of f and W;
        if A(f, W_i) > D_A(W_i) then
            set flag := false;
            break;
        end if
    end for
    if flag == false then
        set F_f := F_f ∪ {f};
    else
        update D_A(W_i) in each half-window W_i ∈ W̄;
    end if
end for
```

Table 1: Statistic of the Contest Benchmarks.

Benchmark	Size	#Cd	#Nets	#CNets	#CCd	#Ly
Circuit1	600000 × 540000	305667	30265	105	12897	9
Circuit2	1050000 × 1110000	750166	64673	185	33325	9
Circuit3	270000 × 170000	64903	6682	55	5307	9
Circuit4	420000 × 210000	149464	16483	100	11896	9
Circuit5	480000 × 350000	275425	29725	171	22813	9

Table 2: Design-Rule and Density Constraints for All Benchmarks.

Layer	minW	maxW	minS	minD	maxD
1	65	1300	65	0.4	1
2	65	1300	65	0.4	1
3	65	1300	65	0.4	1
4	65	1300	65	0.4	1
5	65	1300	65	0.4	1
6	65	1300	65	0.4	1
7	130	1300	130	0.4	1
8	130	1300	130	0.4	1
9	360	3600	360	0.4	1

Contest at ICCAD on Timing-Aware Fill Insertion [1]. Table 1 show the benchmarks, in which the number of conductors ranges from 64K to 750K. In Table 1, the size of the fill region, the numbers of conductors, nets, critical nets, critical conductors, and layers of each benchmark are denoted by "Size", "#Cd", "#Nets", "#CNets", "#CCd" and "#Ly", respectively. Table 2 lists the design-rule and density constraints on different layers for all benchmarks, where "minW" and "maxW" denote the respective minimum and maximum widths of insertion fills, "minS" the minimum spacing between any two conductors, and "minD" and "maxD" the respective minimum and maximum densities of each window.

To verify the effectiveness of our timing-aware fill insertions algorithm, we compared it with the first place team of the contest and the state-of-the-art work [8] on the benchmarks. Thanks to the codes provided by the authors of the first place team and [8], all the experiments were run on a Linux machine with 2.4GHz Intel Xeon CPU and 64GB memory. Table 3 gives the comparisons of the fill insertion results among "1st place team", "FIT [8]", and our algorithm ("Ours"). In this table, "#Fill" gives the number of fills, "Cap" the total capacitance of critical nets, "Avg.D" the average density of all windows, and "CPU(s)" the runtime in seconds. "Normalized" gives the normalized fills, the total capacitance, the average density, and runtime ratios based on the results of the first place team. All the experimental results have passed the official checker ("layoutchecker"). In other words, the fill insertion results for each benchmark do not violate any design-rule and density constraints. In addition, the total capacitance is calculated by the official contest evaluator ("totalcap") [1].

It can be seen from Table 3 that, our algorithm consistently achieves the lowest total capacitance and the least fill amount for every benchmark. On average, compared with "1st place team", our algorithm achieves 12.2% lower total capacitance, 64.4% less fill amount, and 12.8% shorter runtime. In addition, our algorithm also achieves 4.2% lower total capacitance and 19.4% less fill amount than the state-of-the-art work "FIT [8]". We further compared the final results with the initial ones without fill insertions (denoted by "Original"). On average, our algorithm only increases 1.578× total capacitance to meet the density constraints, while the work "FIT [8]" needs to increase 1.649× total capacitance to meet the constraints. The experimental results show that our algorithm is effective for the timing-aware fill insertion optimization.

Figure 8 depicts the final layouts of Circuit3 on layer 2 by our algorithm. Figure 8(a) is the layout of the final result generated by our algorithm. Figure 8(b) gives a partial layout of Figure 8(a). Further, Figure 8(c) gives the layout of a window to better examine the fill insertion. Here, critical nets, non-critical nets, and fills are denoted by red, gray, and green rectangles, respectively.

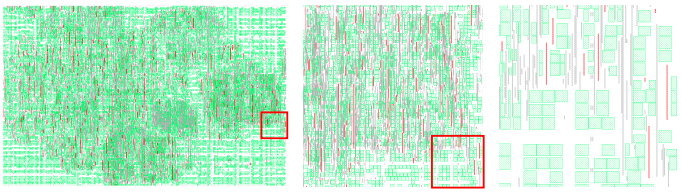


Figure 8: The layouts of Circuit3 on layer 2. Critical nets, non-critical nets, and fills are denoted by red, gray, and green rectangles, respectively. (a) The layout of the final result generated by our algorithm. (b) A partial layout of Figure 8(a). (c) The layout of a window.

5 CONCLUSIONS

In this paper, we have presented an efficient and effective method to address the timing-aware fill insertion problem with total capacitance and density constraints. We have developed a scheme for fast fill insertion and legalization to generate a legal initial solution, and have divided conductors in a circuit into different

Table 3: Experimental Results.

Benchmarks	Original		1st place team					FIT [8]					Ours				
	Avg.D	Cap (pF)	Vio*	#Fill	Avg.D	Cap (pF)	CPU	Vio*	#Fill	Avg.D	Cap (pF)	CPU	Vio*	#Fill	Avg.D	Cap (pF)	CPU
Circuit1	0.09	17.50	N	2058575	0.44	33.11	22.50	N	852246	0.44	30.94	13.47	N	748780	0.41	29.50	21.23
Circuit2	0.09	40.70	N	6797837	0.46	79.41	72.37	N	2805104	0.46	73.53	42.36	N	2587613	0.42	67.56	57.81
Circuit3	0.09	7.45	N	333151	0.44	13.01	4.12	N	153521	0.44	11.80	4.16	N	114854	0.41	11.15	3.80
Circuit4	0.10	15.20	N	673314	0.42	25.46	8.96	N	312561	0.44	23.25	5.08	N	239989	0.41	22.88	8.56
Circuit5	0.10	29.50	N	1281907	0.43	50.89	17.45	N	597433	0.44	45.97	9.34	N	426217	0.41	45.47	12.91
Normalized	-	-	-	1	1	1	1	-	0.444	1.014	0.917	0.659	-	0.356	0.941	0.878	0.872

Vio* denotes whether design-rule and density constraints are violated and "N" represents no.

equivalent paths from critical conductors to powers/grounds. Furthermore, with the help of equivalent paths, we have derived the capacitance graph with reducible edges. According to our analysis, reducing the capacitances between two conductors on a reducible edge can substantially reduce the total capacitance. We have finally also performed density-aware coupling capacitance optimization between every two conductors on reducible edges. We have evaluated the performance of our algorithm based on the 2018 CAD Contest at ICCAD benchmarks and the official contest evaluator. Compared with the first place team of the contest and the state-of-the-art work, the experimental results have shown that our algorithm resolves all the design-rule and density violations, and achieves the lowest total capacitance and the least fill amount for every benchmark.

REFERENCES

[1] 2018 CAD Contest at ICCAD on Timing-aware fill insertion. http://iccad-contest.org/2018/problems.html.

[2] IBM Inc. CPLEX: High-performance mathematical programming solver for linear programming, mixed integer programming, and quadratic programming, version 12.70, https://www.ibm.com/analytics/cplex-optimizer.

[3] N. D. Arora, K. V. Raol, R. Schumann, and L. M. Richardson. Modeling and extraction of interconnect capacitances for multilayer VLSI circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 15(1):58–67, January 1996.

[4] H.-Y. Chen, S.-J. Chou, S.-L. Wang, and Y.-W. Chang. A novel wire-density-driven full-chip routing system for cmp variation control. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 28(2):193–206, February 2009.

[5] Y. Chen, A. B. Kahng, G. Robins, and A. Zelikovsky. Closing the smoothness and uniformity gap in area fill synthesis. In *Proceedings of the International Symposium on Physical Design*, pages 137–142, 2002.

[6] L. Deng, M. D. F. Wong, K.-Y. Chao, and H. Xiang. Coupling-aware dummy metal insertion for lithography. In *Proceedings of IEEE/ACM Asia and South Pacific Design Automation Conference*, pages 13–18, 2007.

[7] I. T. Dimov, T. T. Dimov, and T. V. Gurov. A new iterative Monte Carlo approach for inverse matrix problem. *Journal of Computational and Applied Mathematics*, 92(10):15–35, February 1998.

[8] B. T. Jiang, X. P. Zhang, R. Chen, G. J. Chen, P. S. Tu, W. Li, E. F. Y. Young, and B. Yu. Fit: Fill insertion considering timing. In *Proceedings of ACM/IEEE Design Automation Conference*, 2019.

[9] A. B. Kahng, K. Samadi, and P. Sharma. Study of floating fill impact on interconnect capacitance. In *Proceedings of International Symposium on Quality Electronic Design*, pages 691–696, 2006.

[10] J. Y. Y. Kiat, K. Nyunt, and W. H. Yong. Parasitic capacitance and density optimization modeling fill synthesis for VLSI interconnect. In *Proceedings of the Quality Electronic Design*, pages 16–22, 2012.

[11] Y. Lin, B. Yu, and D. Z. Pan. High performance dummy fill insertion with coupling and uniformity constraints. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 36(9):1532–1544, September 2017.

[12] V. S. Shilimkar and A. Weisshaar. Modeling of metal-fill parasitic capacitance and application to on-chip slow-wave structures. *IEEE Transactions on Microwave Theory and Techniques*, 65(5):1456–1464, 2017.

[13] P. Wu, H. Zhou, C. Yan, J. Tao, and X. Zeng. An efficient method for gradient-aware dummy fill synthesis. *Integration, the VLSI Journal*, 46(3):301–309, 2013.

[14] H. Xiang, L. Deng, R. Puri, K.-Y. Chao, and M. D. F. Wong. Dummy fill density analysis with coupling constraints. In *Proceedings of the International Symposium on Physical Design*, pages 3–10, 2007.