FIT: Fill Insertion Considering Timing

Bentian Jiang
CSE Department, CUHK
btjiang@cse.cuhk.edu.hk

Xiaopeng Zhang
CSE Department, CUHK
xpzhang@cse.cuhk.edu.hk

Ran Chen
CSE Department, CUHK
rchen@cse.cuhk.edu.hk

Gengjie Chen
CSE Department, CUHK
gjchen@cse.cuhk.edu.hk

Peishan Tu
CSE Department, CUHK
pstu@cse.cuhk.edu.hk

Wei Li
CSE Department, CUHK
wli@cse.cuhk.edu.hk

Evangeline F. Y. Young
CSE Department, CUHK
fyyoung@cse.cuhk.edu.hk

Bei Yu
CSE Department, CUHK
byu@cse.cuhk.edu.hk

ABSTRACT

Dummy fill insertion is a mandatory step in modern semiconductor manufacturing process to reduce dielectric thickness variation, and provide nearly uniform pattern density for the chemical mechanical planarization (CMP) process. However, with the continuous shrinking of the VLSI technology nodes, the coupling effects between the inserted metal fills and signal tracks can severely affect the original timing closure of the layout design. In this paper, we propose a robust, efficient and high-performance framework for timing-aware dummy fill insertion, which simultaneously minimizes the coupling capacitance of critical signal wires and other wires. The experimental results on IC/CAD 2018 contest benchmarks shows that our proposed framework outperforms contest winner by 8% on critical coupling capacitance with 3.3× runtime speedup.

1 INTRODUCTION

Nowadays, analysis and optimizations of multilevel interconnect have become extremely challenging in leading-edge VLSI manufacturing process due to the scale miniaturization. Various modern design for manufacturability (DFM) technologies have been proposed to tackle those challenges so as to achieve high-yielding designs. As the predominant planarization technique for multilevel metallization, chemical-mechanical polishing (CMP), has been widely used to attain high level of planarization [1]. However, the quality of CMP patterns is highly related to the uniformity of the density distribution, significant surface topography variation will affect the depth of focus in lithography [1–3]. Moreover, variations in dielectric thickness also reduce yield and affect the performance of circuits. In order to reduce those potential defects, floating dummy fill insertion is commonly performed after the physical design stage. When a fill is inserted, it reduces the dielectric thickness variation, increases planarity, and provides nearly uniform pattern density, all of which are important to mitigate the process variability thereby achieving better yield [4]. However, the coupling effects between the inserted metal fills and signal tracks may severely affect the original timing closure of the layout design. With continuous shrinking of VLSI technology nodes, it is no longer acceptable to ignore interactions between various components of the interconnect in the design-to-manufacturing flow. It is imperative to develop a powerful CAD tool that can significantly reduce the coupling capacitance impact during the metal fill insertion.

In the past decades, extensive works on CMP dummy fill insertion have been proposed based on different techniques and objectives. To reduce the impact of CMP fills on circuits performance, Kahng *et al.* [5, 6] first studied the impact of various floating fill configuration parameters on coupling capacitance, which provided important guidelines for insertion strategies. Based on the guidelines proposed in [5], paper [7] developed a grid-based fill insertion method that heuristically reduced the impact on circuits performance. As for minimizing the fill amount, paper [8] proposed an efficient hybrid hierarchical filling approach with objectives of minimizing the density variation (*Min-Var*) and the fill amount (*Min-Fill*). A recent work [9] presented a fully polynomial time approximation scheme for linear programming formulation with a *Min-Fill* objective. Furthermore, ICCAD 2014 held a dummy fill insertion contest [10] that modeled conventional issues and considered *layer overlay, density variation, line hotspots* and *outlier hotspots*. In paper [11], Liu *et al.* focused on finding a decomposition algorithm that could generate fillable rectangles with better flexibility, and they also proposed an ultra-fast fill insertion engine with the objectives of *Min-Var* and *Min-Fill*. In paper [3, 12], Lin *et al.* proposed an efficient dummy fill insertion flow that further took density gradient into consideration, followed by a min-cost flow acceleration for solving an integer linear programing.

It is notable that the metrics and objectives mentioned above mainly focus on optimization for metal density uniformity, and, none of these metrics can explicitly model or accurately quantify the parasitic capacitance on signal wires. Thus, current mainstream solutions are still far from satisfaction in terms of timing performance. Besides, fill insertion considering timing becomes more and more challenging due to: (1) the shrinkage of technology nodes continuously tightens the design rules, which makes a violation-free solution much easier to fall into sub-optimality in both timing and density uniformity; (2) the large computational-overhead for parasitic extraction may cause an incremental optimization framework with capacitance evaluation not affordable in terms of runtime. An efficient and robust algorithm to insert fills is highly demanded.

Motivated by the challenges mentioned above, in this paper, we develop a fast and high-performance framework for timing-aware dummy fill insertion. Our main contributions can be summarized as follows.

DAC '19, June 2–6, 2019, Las Vegas, NV, USA

ACM ISBN 978-1-4503-6725-7/19/06…$15.00
https://doi.org/10.1145/3316781.3317826

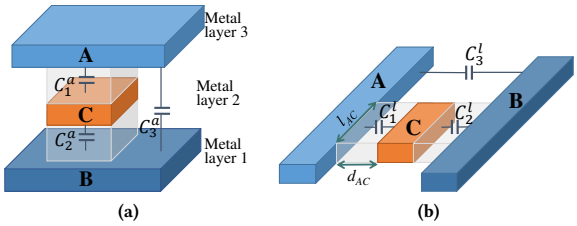

Figure 1: (a) Illustration for area capacitance; (b) Illustration for lateral capacitance.

- A timing-aware target density planning is proposed, which guides an ultra-fast global fill synthesis algorithm to generate high-quality fill results.
- A timing-aware detailed post refinement algorithm is developed to relocate fills with negative impacts on timing, and to shift fills to optimal locations using an analytical method.
- Experimental results on IC/CAD 2018 contest benchmarks [4] show that our framework outperforms the contest winning team by 8% reduction on critical net capacitance, 2% reduction on total net capacitance, with 3.3× runtime speedup.

The rest of the paper is organized as follows. Section 2 lists some preliminaries. Section 3 discusses the details of the framework and algorithms. Section 4 presents experimental results, followed by a conclusion in Section 5.

2 PRELIMINARIES

For timing-aware dummy fill insertion, metal fills are to be inserted subject to the density criteria and design rules, while minimizing the total equivalent capacitance of the given critical nets [4].

2.1 Capacitance Evaluation

The metal fill insertion algorithm needs to account for the impact of capacitance on the signal nets. There are three main types of capacitances to be considered when evaluating the impact of metal fill: area capacitance, lateral capacitance and fringe capacitance, and performance is measured by the total equivalent capacitance of the given critical nets to the ground [4].

2.1.1 Area Capacitance. Two conductor pieces will form an area capacitance when (1) they are on different metal layers, and (2) their projections on the ground plane overlap. For example, in Figure 1(a), metal C forms C_1^a and C_2^a with conductors A and B respectively, and the unshielded overlapping area between conductors A and B forms C_3^a. For an overlapped area s formed by conductors on different layers l_1 and l_2, the area capacitance C^a is calculated by $C^a = P_{l_1,l_2}(s) \times s$, where P_{l_1,l_2} is the area capacitance per unit area and is a function of the overlapped area s.

2.1.2 Lateral Capacitance. Two conductor pieces will form a lateral capacitance when (1) they are on same layer, and (2) they have horizontal overlap. For example, in Figure 1(b), metal C forms C_1^l and C_2^l with conductors A and B, and the unshielded overlapped length between A and B forms C_3^l. The lateral capacitance is calculated by $C^l = P_l(d) \times l$, where l is the length of the parallel overlapped edges of the conductors and $P_l(d)$ is the lateral capacitance per unit length, which is a function of the distance d between the parallel edges.

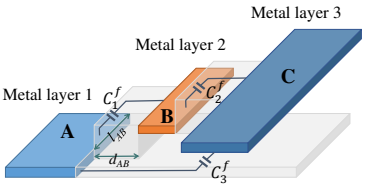

Figure 2: Illustration for fringe capacitance.

2.1.3 Fringe Capacitance. Two conductor pieces will form a fringe capacitance when (1) they are on different layers, and (2) they have parallel edge overlap but do not have area overlap. For example, in Figure 2, metal B forms C_1^f and C_2^f with conductors A and C, and the unshielded overlapped length between A and C forms C_3^f. The fringe capacitance is calculated by $C^f = P_{l_1,l_2}(d) \times l + P_{l_2,l_1}(d) \times l$, where $P_{l_1,l_2}(d)$, $P_{l_2,l_1}(d)$ are the fringe capacitance per unit length, which are functions of the distance d between the parallel edges.

2.1.4 Equivalent Capacitance. The total capacitance of a given set of nets is computed as the summation of their equivalent capacitances to the ground, and it can be obtained by performing conductance matrix based network analysis [4].

2.2 Metal Density and Design Rules

A layout with metal fill must meet the hard constraints on density criteria and design rules [4]. Density is calculated in a window based manner, where a running window of size $w \times w$ and a step size of $\frac{w}{2}$ is considered on each layer l to calculate the metal density D_{l,W_i} in window W_i. The density in each window must lie between a given minimum density D_l^{min} and a given maximum density D_l^{max}. Meanwhile, the design rules for each layer are specified by the constraints on *minimum spacing*, *minimum fill width*, and *maximum fill width*.

2.3 Problem formulation

Given a design layout, the dummy fill insertion problem is to insert metal fills to the layout to satisfy the density criteria and the design rules, while minimizing the total equivalent capacitance of the given critical nets and the overall runtime. No design rule and density violation is allowed [4].

The total parasitic capacitance of all the signal nets (critical and non-critical) is also considered, since it will affect performance like power consumption, timing, etc. Besides, critical capacitance can usually be optimized at the expenses of the total capacitance when targeting at meeting certain density threshold. Hence a good solution should achieve good critical capacitance without sacrificing much on the total capacitance.

3 ALGORITHMS

In this work, we develop a complete framework, which can generate high-quality timing-aware fill insertion for an input layout. A built-in parasitic extraction tool helps us to quantify the equivalent coupling capacitance on signal tracks. The overall flow of our

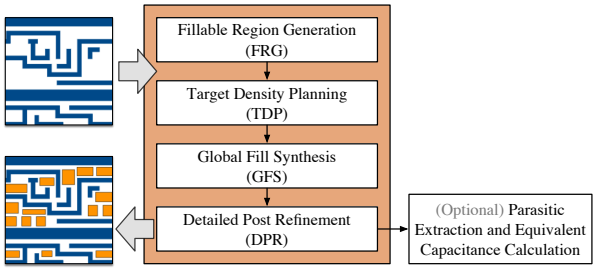

Figure 3: Overall dummy fill insertion flow.

framework is summarized in Figure 3. After parsing the original design layout, we first identify a set of complicated non-overlapping rectilinear polygons as fillable regions according to the layout topography, followed by our fillable region decomposition. The wires and fillable rectangles are then assigned into windows. During this step, the information of *wire density* and *density slack* (gap between density upper bound and target density) are collected. With the wires and fillable rectangles assigned into corresponding windows, we can perform our timing-aware fill insertion algorithm, which can be divided into 3 phases: target density planning, global fill synthesis, and detailed post refinement. Target density planning distributes the target density of each window to reduce the coupling capacitance on critical nets, as well as to maintain a good density uniformity. Global fill synthesis generates high-quality initial fill solution according to the target density in each window and some insertion guidelines. Finally, detailed post refinement is performed to further reduce the coupling capacitance on critical nets.

3.1 Fillable Region Generation

Since the input files only contain the locations of the original wire layout, we need to identify a set of rectilinear polygons as fillable regions according to the wire locations. Fillable region is a region that inserting any fill inside will not violate the *minimum spacing* rule with wires. We first extend each wire in all directions by a distance equal to the *minimum spacing*, and then extract a set of fillable polygons by taking the area complementary to the extended wires. Finally, we will store the vertices of each fillable polygon in a counter-clockwise manner.

The fillable polygons extracted from a layout are usually very complicated with thousands of vertices and possibly with holes inside, thus it is essential to perform polygon decomposition to do a polygon-to-rectangle conversion [3, 11]. Meanwhile, the quality of metal fill optimization is highly related to the conversion results, since it determines fill shape and a density upper bound in each window. Our decomposition algorithm is extended from *I-PTR* (Improved Polygon-To-Rectangle) proposed in paper [11] which traverses the entire vertex list and iteratively extract fillable rectangle efficiently.

Table 1: Comparison of density upper bound of windows

	Case 1	Case 2	Case 3	Case 4	Case 5
I-PTR	0.7196	0.7339	0.7196	0.6990	0.6940
D-PTR	0.7866	0.8009	0.7725	0.7632	0.7642
Improvement	9.30%	9.13%	7.34%	9.18%	10.13%

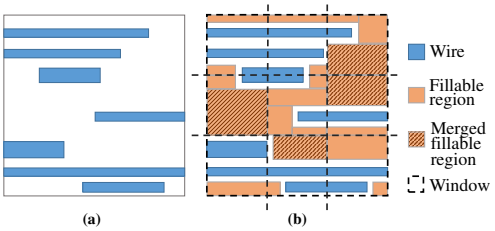

Figure 4: (a) Input layout; (b) Output fillable regions.

We further extend *I-PTR* into *D-PTR* (Direction-Aware Polygon-To-Rectangle), where the shapes of the output rectangles ("fat" or "tall") will automatically match with the directions (horizontal or vertical respectively) of the wires on the current metal layer. Meanwhile, with fillable rectangles and wires being assigned to corresponding windows, we will conduct horizontal and vertical sweep line algorithms in each window to merge all enclosed fillable rectangles that share common edge (see Figure 4). Experimental results in Table 1 show that, compared with *I-PTR*, our *D-PTR* with merge operations significantly improves the density upper bound for windows (which is essential especially for windows on lower metal layers) and expands the solution spaces for later procedures.

3.2 Target Density Planning

As mentioned in Section 2.2, the density check on each layer is performed by a running window with size of $w \times w$ and step length $\frac{w}{2}$. To handle this effectively, we can divide the entire layout into sub-windows with size of $\frac{w}{2} \times \frac{w}{2}$, and require every sub-window to satisfy the original density requirements. However, from the perspective of fill synthesis, larger windows always have higher flexibility for local fill distribution, while the sub-window division may cause sub-optimality in local fill distribution, Therefore, instead of directly applying the above sub-window division, we further propose a Target Density Planning (TDP) algorithm to distribute the target density for each sub-window with constraints on density criteria, and simultaneously to consider the following objectives: (1) reduce the coupling capacitance with critical nets, (2) reduce total wire capacitance. In the remaining parts of the paper, a window refers to a sub-window (of size $\frac{w}{2} \times \frac{w}{2}$) as mentioned above.

$$\min \sum_{i,j} \Omega_{i,j} D_{i,j} - \min_{i,j} \{D^{max}_{i,j} - D_{i,j}\} \tag{1a}$$

$$\text{s.t. } D^{wire}_{i,j} \le D_{i,j} < D^{max}_{i,j}, \ \forall\ i,\ j \tag{1b}$$

$$D_{i,j} + D_{i,j+1} + D_{i+1,j} + D_{i+1,j+1} \ge 4 \cdot D_{min},, \ \forall\ i,\ j. \tag{1c}$$

Let $\mathbf{W}$ be the window set of a layer with dimension $m \times n$, where W_{ij} is the window in ith row and jth column, and $1 \le i \le m,\ 1 \le j \le n$. For each layer, our target density planning is formulated as in Equation (1), where $D^{wire}_{i,j}$, $D^{max}_{i,j}$, $D_{i,j}$ are the wire density (without fill), the density upper bound (which can be computed by a greedy insertion until no fill can be inserted) and the target density (variable) of window W_{ij} respectively. D_{min} is the given minimum density requirement, and $\Omega_{i,j}$ is a weight to measure the

criticality of W_{ij} defined as

$$\Omega_{i,j} = \begin{cases} \epsilon, & \text{if} \quad a_{ij}^{c} = 0,\ a_{ij}^{nc} = 0, \\ \omega^{c} \cdot a_{ij}^{c} + \omega^{nc} \cdot a_{ij}^{nc}, & \text{else.} \end{cases} \tag{2}$$

where a_{ij}^{c} denotes the summation of critical wire area in window W_{ij} and its corresponding windows in the upper and lower layers, a_{ij}^{nc} represent the non-critical wire area in window W_{ij}. Parameters ω^{c} and ω^{nc} are the weights of a_{ij}^{c} and a_{ij}^{nc}, and ϵ is an extremely small constant. Equation (1a) consists of two parts. The first weighted summation term tries to minimize the target densities of critical windows (windows which enclose critical wires), meanwhile, it also helps to reduce the target densities for windows with more wires. In practice, ω^{c} should be much larger than ω^{nc} to ensure the highest priority for the critical windows to get less fills. Besides, a small ϵ helps to attract more density assignment for those empty windows located close to the critical windows or windows with higher wire density. The second term considers the smallest density margin between the target density and the density upper bound among all the windows. With the help of the second term and the density constraints (1c)-(1b), minimization of the objective function reduces the density variation and maintains the flexibility for the ease of insertion. Note that Equation (1a) is not a standard LP formulation, we linearize it by introducing an auxiliary variable M and an auxiliary constraints (Equation (3b)). Then Equation (3) can be efficiently solved by the off-the-shelf solver [13].

$$\min \sum_{i,j} \Omega_{i,j} D_{i,j} - M \tag{3a}$$

$$\text{s.t. } M \le D_{i,j}^{max} - D_{i,j},\ \forall\ i,\ j \tag{3b}$$

$$D_{i,j}^{wire} \le D_{i,j} < D_{i,j}^{max},\ \forall\ i,\ j \tag{3c}$$

$$D_{i,j} + D_{i,j+1} + D_{i+1,j} + D_{i+1,j+1} \ge 4 \cdot D_{min},\ \forall\ i,\ j. \tag{3d}$$

3.3 Global Fill Synthesis

In this section, we will present our fill synthesis algorithm for each window under the guidance of the target density computed in the previous step. The quality of fill synthesis plays the most important role in dummy fill insertion. More preciously, for timing-aware fill insertion, the fill shapes and fill locations directly affect the coupling capacitance. We propose a fast and high-performance fill synthesis algorithm partially based on the guidelines provided in previous works [5, 6, 14] to minimize the total capacitance. Our major insertion policies can be described as follows (with decreasing importance): (1) avoid area overlap between fills and the critical wires on upper/lower layers, (2) increase the spacing between fills and other conductors, (3) avoid insertion in the region between two adjacent parallel wires (high impact region [5]), (4) reduce the parallel overlap lengths between fills and other conductors.

The main steps of our global fill synthesis flow are summarized in Algorithm 1 and 2. After initialization, we first sort the windows in an increasing order of their density gaps $D_{max} - D_t$, where D_{max} and D_t are the density upper bound and the target density of a window respectively. Since the filling order of adjacent windows may affect the fill results of each other, sorting by density gap can ensure the highest filling priorities for windows which have less budget to achieve their target density (D_t). We then iteratively

Algorithm 1 Layer-based global fill synthesis

Input: Window set W of layer l;
1: Initialize layer violation checker and space S;
2: Sort window set W by the density gap $D_{max} - D_t$;
3: **for** window $w \in W$ **do** ▷ D_t is the target density of w
4: **while** $S \ge minSpace$ **do**
5: $success \leftarrow$ windowFiller(w, S, D_t);
6: **if** success **then** break; ▷ Reset space S
7: $S = S - steplength$;
8: Detect violation for w, roll back if needed;
9: Update layer violation checker with fill result for w;
10: Perform design rule and density checks for l, roll back if needed;
11: **return** fill results for layer l

Algorithm 2 Window filler

Input: Window w, space S and target density D_t;
1: Define D as the current density of the input window w;
2: Define R as the fillable rectangle set of the input window w;
3: Define F as the fill candidate set inside a fillable rectangle;
4: Define U as the default fill size and w_{min} as the min fill width;
5: Sort the fillable rectangle set R by criterion in eq. (4);
6: **for** fillable rectangle $r \in R$ **do**
7: **if** $D \ge D_t$ **then** break;
8: $U = \max(\theta \cdot (D_t - D)^2,\ {w_{min}}^2)$;
9: $F \leftarrow$ getFillCandiates (r, S, U); ▷ Generate fill candidates
10: Rearrange the positions of F to move them away from wires;
11: **for** fill candidate $f \in F$ **do**
12: **if** $D \ge D_t$ **then** break;
13: **if** f has area overlaps with critical wires $\notin$ layer l **then**
14: Eliminate the overlapped area of f;
15: Detect violation with violation checker;
16: Legalize violation in f if needed;
17: Update D and violation checker;
18: **if** $D \ge D_t$ **then return** success
19: **else return** failure

perform *window filler* (Algorithm 2) on each window to search for a fill solution with the largest spacing (Algorithm 1 line 4-7) that can simultaneously satisfy the target density and the design rules. For each result, design rule check and density check will be performed to ensure the correctness.

Significant quality improvement comes from the detailed insertion strategies described in the window filler algorithm. When a window is being filled, we first sort the fillable rectangles of the window in a non-increasing order of the criterion:

$$\alpha \cdot h + \beta \cdot A + \gamma \cdot \sqrt{d_e} + \eta \cdot \frac{1}{l}, \tag{4}$$

Where h, A, d_e, l are the height of the fillable rectangle (width for horizontal layer), the area of the fillable rectangle, the centroid Euclidean distance and the parallel overlap length ($l = \max(1, l)$) between the fillable rectangle and its nearest critical wire, respectively. The weights of h, A, d_e and l are in decreasing order of their values, where h and A help to identify and avoid insertion in high impact regions [5], d_e and l consider the potential impacts on critical wires.

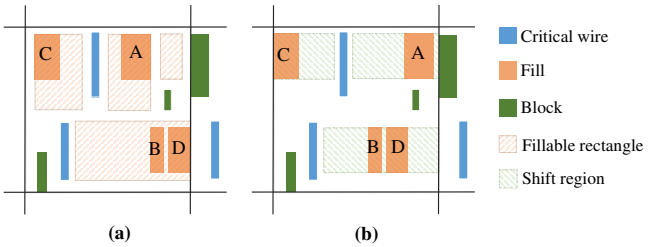

Figure 5: (a) Before shift refinement, the positions of fills A and C are limited by the fillable rectangles, and fills A, B, C, D are not in the optimal locations; (b) After shift refinement, fills A and C jump out of the fillable rectangles to find their optimal locations, while fills B and D also find their optimal locations.

Secondly, for each fillable rectangle inside the window, a set of fill candidates will be generated according to the given spacing S and default fill size U, where U is dynamically determined by the margin between the target and the current density. We then rearrange the positions of the fill candidates to reduce the parallel overlap between the fill candidates and to insert the fills away from the wires. Violation detection is performed on each candidate, followed by a legalization if needed. Finally, a qualified fill result of the window will be returned to Algorithm 1.

3.4 Detailed Post Refinement

Since the objective of the Global Fill Synthesis (GFS) flow is to minimize the total fill coupling impact on wires in general, some patterns may not be very friendly to critical wires. Detailed post refinement is needed to eliminate or adjust the fills with high impact to critical wires as well as to honor the global filling results. We thus propose a timing-aware Detailed Post Refinement (DPR) which consists of a fill relocation step, and a fill shifting step.

3.4.1 Timing-aware Fill Relocation. For each critical window with target density D_t, let A_i be the fill area (variable) of the ith fillable rectangle inside this critical window. Now, we want to relocate fills (obtained from GFS) with high-impact on timing to those fillable rectangles that have less impact to timing. Then we have,

$$\min \sum_i \gamma_i A_i \tag{5a}$$

$$\text{s.t. } 0 \le A_i \le A_i^{max},\ A_i \in \mathbb{Z}, \tag{5b}$$

$$\sum_i A_i \ge D_t \cdot (\frac{w}{2})^2,\ \forall i, \tag{5c}$$

where A_i^{max} is the fill area upper bound of the ith fillable rectangle, $\gamma_i = \sum_k \frac{l_{ik}}{d_{ik}^2}$ is the weight to estimate the timing-impact of the fill insertion in ith fillable rectangle, d_{ik} is the Manhattan distance from the ith fillable rectangle to its k^{th} ($k \in \{1, 2, 3\}$) closest critical wire, and l_{ik} is their corresponding parallel overlap length. Equation (5) re-assigns the fill area of each fillable rectangle, with an objective of minimizing the potential coupling capacitance to critical wires. Note that this problem can be solved optimally with a greedy method.

After the planning, for each fillable rectangle i inside the critical window, the fill area A_i will be compared with the actual filled area A_i' obtained from the previous global fill synthesis (GFS) stage. The fills inside this fillable rectangle will be relocated only if the area difference $|A_i - A_i'|$ is larger than a certain threshold. Otherwise the GFS fill result inside this fillable rectangle will be kept.

3.4.2 Timing-aware Fill Shifting. In order to further improve the timing performance of our fill solution, we will to find out the most suitable location for each fill to minimize the critical capacitance. An alternating one dimensional timing-aware fill shifting (x-dimension or y-dimension) is proposed. Take a vertical layer as example (the directions of wires are all vertical), our timing-aware fill shifting (in x-dimension) is formulated as

$$\min D = \sum_{f \in F} \sum_{c \in C} \frac{l_{fc}}{d^2(f,c)}, \tag{6a}$$

$$\text{s.t. } d(f,c) = |x_f - x_c| - \frac{1}{2}w_f - \frac{1}{2}w_c, \tag{6b}$$

$$L + \frac{1}{2}w_f \le x_f \le R - \frac{1}{2}w_f, \tag{6c}$$

$$|x_f - x_{f'}| \ge \frac{1}{2}w_f + \frac{1}{2}w_{f'} + S_{min}, \tag{6d}$$

$$|x_f - x_b| \ge \frac{1}{2}w_f + \frac{1}{2}w_b + S_{min}, \tag{6e}$$

$$\forall\ f,\ f' \in F, \text{ and } f \ne f',\ c \in C,\ b \in B, \tag{6f}$$

where in each critical window, F is a fill set which contains all fills f inside the current window, C is a critical wire set which contains all critical wires c inside the current window and the neighboring windows, B is the block set which contains wires in the current window and fixed conductors (wires and fills) in the neighboring windows. l_{fc} measures the parallel overlap between the fill f and the critical wire c, and $d(f,c)$ (variable) measures the horizontal distance between f and c. Besides, x_f, x_c and x_b denote the x-coordinates of the center points of the fill f (movable), the critical wire c, and the block b, while w_f, w_c and w_b represent the width of f, c and b respectively. S_{min} is the minimum spacing requirement, while L and R are the left and right boundaries (x-coordinates) of the current window.

We observed that the lateral capacitance between the fill f and the critical wire c (described in Section 2.1) is roughly proportional to $\frac{l_{fc}}{d^2(f,c)}$. Thus, we try to minimize the summation of all $\frac{l_{fc}}{d^2(f,c)}$ in a critical window. By minimizing the Equation (6a), the fills in the window can shift to better position to reduce the lateral capacitance, as shown in Figure 5. To simplify Equation (6a), we additionally impose the horizontal fixed order constraint to it, where the relative horizontal order among each conductor pair is fixed. As the result, the movable region of a fill is restricted by other conductors, and it is no longer related to the original fillable rectangle (see in Figure 5). Moreover, Equation (6a) becomes differentiable due to the fixed order constraint, and we can optimize it using gradient descent. For each fill, its position x_f can be updated by the Equation (7). Here $x_f^{(t)}$ represents the center point location (x-coordinate) of fill f in the t^{th} iteration, and α is the step size of the gradient descent. In each iteration, if $D^{(t+1)} > D^{(t)}$, α should be replaced by $\frac{\alpha}{2}$. After convergence, each fill can get an appropriate position in the x-dimension, as illustrated by the example in Figure 5.

Table 2: Experimental Results on IC/CAD 2018 Benchmark

case	# wires	# critical wires	1st place team				FIT			
			RT-s (s)	RT-m (s)	$C_{critical}$ (pF)	C_{total} (pF)	RT-s (s)	RT-m (s)	$C_{critical}$ (pF)	C_{total} (pF)
case1	305667	12897	19.10	19.10	33.11	11313.69	8.80	5.16	30.94	10883.66
case2	750166	33325	61.83	61.83	79.41	39612.26	31.44	18.41	73.53	39523.68
case3	64903	5307	3.51	3.51	13.01	1669.72	1.59	1.09	11.80	1558.17
case4	149464	11896	7.52	7.52	25.46	3136.48	3.55	2.43	23.25	2969.20
case5	275425	22813	15.14	15.14	50.89	6150.53	6.97	4.62	45.97	5705.39
Total	-	-	107.10	107.10	201.88	61882.68	52.34	31.71	185.48	60640.10
Ratio	-	-	1.000	1.000	1.000	1.000	**0.489**	**0.296**	**0.919**	**0.980**

* RT-s denotes overall runtime in single thread mode, RT-m denotes overall runtime in 8-threads.

$$\frac{\partial D}{\partial x_f} = \begin{cases} \sum_{c \in C} \dfrac{-2 \cdot l_{fc}}{(x_f - x_c - \frac{1}{2}w_f - \frac{1}{2}w_c)^3}, & \text{if } x_f \geq x_c, \\ \sum_{c \in C} \dfrac{-2 \cdot l_{fc}}{(x_f - x_c + \frac{1}{2}w_f + \frac{1}{2}w_c)^3}, & \text{if } x_f < x_c, \end{cases} \tag{7a}$$

$$x_f^{(t+1)} \leftarrow x_f^{(t)} - \alpha \frac{\partial D}{\partial x_f^{(t)}}, \quad f \in F, \tag{7b}$$

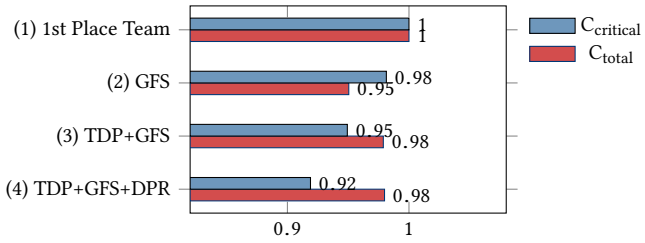

Figure 6: Comparison of normalized performance.

4 EXPERIMENTAL RESULTS

We implemented our proposed FIT framework in C++ language, CPLEX [13] is used as the linear programming solver. The benchmarks are released by IC/CAD 2018 contest [4], of which the statistics are shown in Table 2. All the tests and evaluations are conducted on a 4-cores 3.4GHz Linux machine with 32GB memory, including the experimental results from the binaries of the first place team in the contest, and multithreading is enabled for all approaches. The official capacitance evaluation tool is released by the organizers of IC/CAD 2018 contest[4].

Quantitative results are listed in Table 2, where columns "RT-s", "RT-m", "$C_{critical}$", "C_{total}" list the single thread runtime, the multi-thread runtime, the total equivalent capacitance of the given critical nets, and the total parasitic capacitance of all the nets, respectively. It is shown that our FIT framework outperforms the first place team in all metrics. More specifically, the proposed FIT framework get 8% reduction on equivalent capacitance of critical nets, and 2% reduction on patristic capacitance of all nets in comparison with the first place team. Besides, our FIT achieves more than 2× runtime speedup in single-thread execution, and 3.37× runtime speedup in multi-thread execution. Moreover, we study the impact of each stage on timing. As shown in Figure 6, with the help of the target density planning, we obtained about 3% reduction on $\mathbf{C_{critical}}$, while $\mathbf{C_{total}}$ increases by about 2.8%. After enabling our detailed post refinement, $\mathbf{C_{critical}}$ further reduces about 3%, and there is nearly no increase in $\mathbf{C_{total}}$. Above results verify that our algorithm can generate better solutions which achieves good critical capacitance without sacrificing much on the total capacitance.

5 CONCLUSION

In this work, we propose a fast and high-performance methodology for the timing-aware fill insertion problem, and experimental results have verified the effectiveness of our algorithms. Future work would include ultra-fast parasitic extraction combined with incremental analytical optimization for timing-aware fill insertion.

ACKNOWLEDGMENT

The work described in this paper was partially supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CUHK14208914).

REFERENCES

[1] A. B. Kahng and K. Samadi, "CMP fill synthesis: A survey of recent studies," *IEEE TCAD*, vol. 27, no. 1, pp. 3–19, 2008.
[2] P. Gupta, A. B. Kahng, O. S. Nakagawa, and K. Samadi, "Closing the loop in interconnect analyses and optimization: CMP fill, lithography and timing," in *Proc. VMIC*, 2005, pp. 352–363.
[3] Y. Lin, B. Yu, and D. Z. Pan, "High performance dummy fill insertion with coupling and uniformity constraints," *IEEE TCAD*, vol. 36, no. 9, pp. 1532–1544, 2017.
[4] Y. Bo and S. Sriraaman, "ICCAD-2018 CAD contest in timing-aware fill insertion," in *Proc. ICCAD*, 2018.
[5] A. B. Kahng, K. Samadi, and P. Sharma, "Study of floating fill impact on interconnect capacitance," in *Proc. ISQED*, 2006.
[6] A. B. Kahng and R. O. Topaloglu, "A DOE set for normalization-based extraction of fill impact on capacitances," in *Proc. ISQED*, 2007, pp. 467–474.
[7] ——, "Performance-aware CMP fill pattern optimization," in *Proc. VMIC*, 2007.
[8] Y. Chen, A. B. Kahng, G. Robins, and A. Zelikovsky, "Closing the smoothness and uniformity gap in area fill synthesis," in *Proc. ISPD*, 2002, pp. 137–142.
[9] C. Feng, H. Zhou, C. Yan, J. Tao, and X. Zeng, "Efficient approximation algorithms for chemical mechanical polishing dummy fill," *IEEE TCAD*, vol. 30, no. 3, pp. 402–415, 2011.
[10] R. O. Topaloglu, "ICCAD-2014 CAD contest in design for manufacturability flow for advanced semiconductor nodes and benchmark suite," in *Proc. ICCAD*, 2014, pp. 367–368.
[11] C. Liu, P. Tu, P. Wu, H. Tang, Y. Jiang, J. Kuang, and E. F. Y. Young, "An effective chemical mechanical polishing filling approach," in *Proc. ISVLSI*, 2015, pp. 44–49.
[12] Y. Lin, B. Yu, and D. Z. Pan, "High performance dummy fill insertion with coupling and uniformity constraints," in *Proc. DAC*, 2015, pp. 71:1–71:6.
[13] 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.
[14] A. Kurokawa, T. Kanamoto, T. Ibe, A. Kasebe, C. W. Fong, T. Kage, Y. Inoue, and H. Masuda, "Dummy filling methods for reducing interconnect capacitance and number of fills," in *Proc. ISQED*, 2005, pp. 586–591.