A Novel Wire-Density-Driven Full-Chip Routing System for CMP Variation Control

Huang-Yu Chen, Student Member, IEEE, Szu-Jui Chou, Sheng-Lung Wang, and Yao-Wen Chang, Member, IEEE

Abstract—As nanometer technology advances, the post chemical–mechanical polishing (CMP) topography variation control becomes crucial for manufacturing closure. To improve the CMP quality, dummy-feature filling is typically performed by foundries after the routing stage. However, filling dummy features may greatly degrade the interconnect performance and significantly increase the input data in the following time-consuming reticle enhancement techniques. It is, thus, desirable to consider wire-density uniformity during routing to minimize the side effects from aggressive post-layout dummy filling. In this paper, we present a new full-chip grid-based routing system considering wire density for reticle planarization enhancement.

To fully consider a wire distribution, the router applies a novel two-pass top-down planarity-driven routing framework, which employs new density critical area analysis based on Voronoi diagrams and incorporates an intermediate stage of a density-driven layer/track assignment based on incremental Delaunay triangulation. Experimental results show that our methods can achieve a more balanced wire distribution than state-of-the-art works.

Index Terms—Design for manufacturability, layout, physical design, routing.

I. INTRODUCTION

As IC process geometry shrinks to 65 nm and below, one important yield loss of interconnects comes from the chemical–mechanical polishing (CMP), which consists of electroplating followed by CMP [29].

As the denotation, CMP contains chemical and mechanical parts. A typical schematic of the CMP process is illustrated in Fig. 1, where abrasive slurry, which chemically dissolves the wafer layer, is distributed over the surface of the polishing pad, and a dynamic polishing head mechanically presses the polishing pad and the wafer. After CMP, the surface of the wafer would become smooth and flat.

Because of the difference in hardness between copper and dielectric materials, the CMP planarizing process might generate topography irregularities. A non-uniform feature density distribution on each layer causes CMP to over polish or under polish, generating metal dishing and dielectric erosion [31]. These thickness variations have to be controlled carefully since the variation in one interconnect level is progressively transferred to subsequent levels during manufacturing, and, finally, the compounding variation can be significant on an upper level, which is often called the multilayer accumulative effect [32].

Two key problems arise from the post-CMP thickness variation.

1) The layout surface fluctuates inside or outside the depth of focus of the photolithography system, such that the exposed patterns do not appear acceptably sharp, and open/short defects may even occur.

2) These irregular variations greatly change the electrical characteristics of interconnects, particularly for resistance and capacitance, degrading the accuracy of timing analysis and worsening the electromigration.

These issues become critical in the nanometer designs and have recently attracted much attention [23]. As a result, to improve chip thickness uniformity, the Taiwan Semiconductor Manufacturing Company (TSMC) recommends performing virtual CMP analysis to identify the metal and dielectric thickness variation hot spot before chip fabrication for 65-nm manufacturing processes (see [33]).

To improve the CMP quality, modern foundries often impose recommended layout density rules and fill dummy features into layouts to restrict the variations on each layer. For example, on each metal layer, local pattern density within every predefined window must be within a specified range, for example, between 10% and 40%. These density bounds can also help in minimizing the multilayer accumulative effect, as reported in [26].

Digital Object Identifier 10.1109/TCAD.2008.2009156
Dummy features may either be connected to power/ground (P/G; tied fills) or left floating (floating fills) [27]. The tied fill has predictable but higher capacitance, whereas the floating fill has lower but unpredictable one due to the floating nature. Traditionally, electrical impacts of dummy fills can be negligible, and dummy features are inserted during the post routing stage. Filling algorithms have been proposed to satisfy density bounds and reduce the density variation [22], [34]. However, as reported in [35], these filled dummy features may incur troublesome problems at 65 nm and successive technology nodes. The tied fill may induce crosstalk for its high coupling capacitance to nearby interconnects and would place a heavy burden for the P/G network. On the other hand, the capacitance of floating fills is usually uncertain, and, thus, its capacitance extraction requires additional efforts to solve the same matrix system of nonfloating capacitance with more unknowns and some transformation [36].

Moreover, dummy fills also purely increase the data volume of mask, lengthening the time of mask-making processes such as mask synthesis, writing, and inspection verification. In particular, these filled features would significantly increase the input data in the following time-consuming reticle enhancement techniques, such as optical proximity correction and phase shift mask. Therefore, much research focuses on impact-limited dummy-feature-filling algorithms [8], [25].

In the nanometer technology, routing has become a decisive factor for determining chip manufacturability since it presides over most of the layout geometry in the back-end design process. To tackle these manufacturing challenges, routing techniques must handle the increasing complexity. The routing approaches applying the bottom-up coarsening and top-down uncoarsening techniques have demonstrated the superior capability of handling large-scale routing problems, such as the A-shaped multilevel [4], [5], [18], the V-shaped multilevel [6], and the two-pass bottom-up [7] routing frameworks.

Recently, routing considering a wire distribution has attracted much attention in the literature. The earlier studies for CMP processes have indicated that the post-CMP topography is highly correlated to the layout pattern density because, during the polishing step, interlevel dielectric removal rates are varied with the pattern density [32]. Furthermore, the layout pattern (consisting of wires and dummy features) density can be systematically determined by the wire-density distribution, as reported in [13]. Specifically, as shown in Fig. 2, for a given global cell $v_i$ with a metal density $m_i$, the Cu thickness of $v_i$, i.e., $t_i$, can be expressed as follows [13]:

$$t_i = \alpha \left(1 - \frac{m_i^2}{\beta}\right)$$

(1)

which shows that the post-CMP copper thickness is determined by the wire density.

Therefore, managing wire density at the routing stage has great potential for alleviating the aggressive dummy-feature-filling-induced problems.

Li et al. [28] presented the first routing system in the literature addressing the CMP-induced variation. By setting the desired density in the cost function of global routing, the routing results have a more balanced interconnect distribution.

Cho et al. [13] proposed a pioneering work to consider CMP variation during global routing. They empirically developed a predictive CMP density model and showed that the number of inserted dummy features can be predicted by the wire density. Therefore, they proposed a minimum-pin density global routing algorithm to reduce the maximum wire density in each global tile. However, both approaches only consider the wire density inside a routing tile. Since the topographic variation is a long-range effect, focusing the density value inside each routing tile may incur larger intertile density difference and result in more irregular post-CMP thickness [see Fig. 3(a)]. Therefore, optimizing wire-density uniformity inside a routing tile is obviously not a right metric and a common pitfall for CMP control. For better CMP control, it is more desirable to minimize the global variation of the wire density, i.e., the density gradient, which is a major objective for dummy fills [10]. As shown by the example in Fig. 3, if the density lower and upper bounds are 20% and 80%, respectively, then the three adjacent routing tiles in Fig. 3(b) all satisfy these rules. However, Fig. 3(c) is a better choice for CMP control because it has the minimum wire-density gradient.

In this paper, we present a new full-chip grid-based routing system, named the two-pass top-down grid-based router (TTR), considering wire-distribution uniformity for density variation minimization. To fully consider the wire distribution, the router is based on a novel two-pass top-down planarization-driven routing framework, which consists of four major stages (see Fig. 4 for an illustration):

1) prerouting: identify the potential density hot spots based on the pin distribution and the wire connection to guide the following global routing;
2) global routing: apply prerouting-guided planarization-aware global pattern routing for local nets and iteratively refine the solution;
3) layer/track assignment: perform density-driven layer/track assignment for long segments panel by panel;
4) detailed routing: use segment-to-segment detailed maze routing to route short segments and reroute failed nets level by level.

Different from the aforementioned works, the TTR has the following distinguished features:

- A new routing framework of performing density prediction in the prerouting stage, followed by planarization-aware global routing at the first uncoarsening stage, an
Fig. 3. Density variation among neighboring subregions impacts topography. (a) Different wire distribution in a subregion exists even under the same density. Large density variation among neighboring subregions leads to post-CMP thickness irregularities. (b) Three adjacent routing tiles satisfy density rules but result in an unbalanced wire distribution. (c) Better result for minimizing the density gradient among tiles.

Fig. 4. New two-pass top-down planarity-driven routing framework.

intermediate stage of density-driven layer/track assignment, and then detailed routing at the second uncoarsening stage.

- An efficient density critical area analysis (CAA) algorithm based on Voronoi diagrams is performed offline in the prerouting stage, which considers both topological information of pins and wire connection to complement the density analysis. As shown in Section IV, the Voronoi-diagram-based CAA algorithm leads to a 3%–5% faster overall routing process due to easier density control for later detailed routing. Furthermore, it can substantially improve the resulting wire-density uniformity.

- A planarization-aware global router is employed to consider the density lower and upper bounds while minimizing the density gradient among global tiles.

- A layer assigner for panel-density minimization and a density-driven track assignment (DTA) algorithm based on the incremental Delaunay triangulation (DT) are performed before detailed routing to preserve more flexibility for wire-density arrangement.

Compared with the density-driven routing system [28], experimental results show that the TTR can achieve 43% reduction on the maximum number of nets crossing in tiles and obtain at least 35% smaller standard deviations of the wire distribution.

The rest of this paper is organized as follows. Section II describes the routing model and the routing framework. Section III presents our density-driven routing algorithms. Experimental results are reported in Section IV, and conclusions are given in Section V.

II. ROUTING MODEL

We first explain the routing model. As illustrated in Fig. 4, \( G_k \) corresponds to the routing graph of level \( k \). Each level contains a number of global cells (GCs), and the GCs belonging to different levels have different sizes. We denote \( GC_k \) as the GC of level \( k \).

The first top-down routing pass is for global routing, which starts uncoarsening from the coarsest level to the finest level (level 0). At each level \( k \), our global router finds routing paths
Therefore, it is necessary to consider both the topological information and the wire connections of each pin to complement the density analysis. To remedy the deficiency, we develop a new enhanced analysis model based on Voronoi diagrams. The Voronoi diagram of a point set $P$ partitions the plane into regions, called Voronoi cells, each of which is associated with a point of $P$. If a point in the plane is closer to the point $p_t \in P$ than to any other point of $P$, then this point will be in the interior of the Voronoi cell that is associated with $p_t$. The boundary segments of a Voronoi cell are called the Voronoi edges. A Voronoi diagram can efficiently compute the physical proximity and has been well studied in computational geometry [19]. Papadopoulou and Lee [30] used Voronoi diagrams of rectilinear polygons to compute the critical areas for short defects in a circuit layout.

The motivation for the Voronoi diagram approach lies in the following observation.

1) Observation 1: Given the Voronoi diagram of points, the standard deviation for the size of Voronoi cells strongly depends on the distribution of these points.

As illustrated in Fig. 6(a), the Voronoi cells for points with a non-uniform distribution have large variation in sizes; in contrast, as shown in Fig. 6(b), for points with a uniform distribution, the sizes of Voronoi cells are almost the same.

Another observation can quantify the proximity relation to indicate whether a point lies in the dense area.

2) Observation 2: For a point, the number of adjacent Voronoi cells that entirely sit within a specified distance from this point reflects the dense quantity of the region where this point lies.

As shown in Fig. 7(a), the point in the dense area has more Voronoi cells around it within a given circle with its center at this point.

Base on these observations, we specify a range $r$ and associate each pin $p$ with a density cost $d_p$, which is defined as

$$d_p = \alpha \nu_p + (1 - \alpha) \omega_p$$

(2)

where $\nu_p$ is the number of Voronoi cells around $p$ (excluding the Voronoi cell that is associated with $p$ itself) that entirely sit inside the circle with a center at $p$ and radius $r$, $\omega_p$ is the number of wires connecting to $p$, and $\alpha$, $0 \leq \alpha \leq 1$, is a user-defined parameter. For the example shown in Fig. 7(b), there are three Voronoi cells around $p$ that entirely sit inside the circle, and four wires are connected to $p$. Therefore, $\nu_p$ and $\omega_p$ are equal to 3 and 4, respectively.

In the current implementation, we set radius $r$ as the average distance among the pins of adjacent Voronoi cells. This way, the expected value for $\nu_p$ would be zero if $p$ lies in a uniformly distributed region; otherwise, $\nu_p$ would increase as a penalty to reflect the density hot spot where $p$ lies. Additionally, since two-pin nets practically dominate the netlist in most designs, the expected value of $\omega_p$ would be equal to 1. Therefore, the ranges of $\nu_p$ and $\omega_p$ in (2) are similar and can be reasonably combined together through the $\alpha$ parameter.

After all density costs of pins have been computed, we transform these costs into the cost of global tiles. For each global tile $t$, we set its predicted density cost $d_t = \max\{d_p \mid p \text{ is inside } t\}$ in the prerouting stage. Then, the TTR feeds the...
pre-estimated density information to the following routing stages. The density CAA can be performed efficiently. We have the following theorem.

**Theorem 1:** The Voronoi-diagram-based density CAA runs in \(O(|P| \lg |P|)\) time, where \(|P|\) is the number of pins.

**Proof:** The construction for a Voronoi diagram of \(|P|\) pins takes \(O(|P| \lg |P|)\) time. A Voronoi diagram contains \(O(|P|)\) Voronoi edges; therefore, the computation for the average distance among the pins of adjacent Voronoi cells needs \(O(|P|)\) time. Moreover, the transformation from the pin costs into the costs of global tiles takes \(O(|P|)\) time. Therefore, the overall time complexity is \(O(|P| \lg |P|)\).

Note that the Voronoi-diagram-based CAA algorithm is performed only once, and its running time overhead is very small (about 3% of the total running time in our experiment). Furthermore, it even leads to a 3%–5% faster overall routing process due to easier density control for later detailed routing, and it can substantially improve the resulting wire-density uniformity.

### B. Planarization-Aware Global Routing

The global routing plans tile-to-tile routing paths for all nets and thereby is an important step to decide the wire distribution and maintain a uniform metal density across the chip. As mentioned in Section I, both previous works [13], [28] consider only the wire density inside each global tile, which might incur a larger intertile density gradient and, thus, more irregular post-CMP thickness. As a result, for better CMP control, a global router has to consider the density variation (gradient) among global tiles in addition to the wire density inside each tile.

In our TTR, the global routing performed in the first top-down uncoarsening pass is based on pattern routing [24]. Pattern routing uses an L-shaped (one-bend) or Z-shaped (two-bend) route to make the connection, which gives the shortest path length between two points while reducing the routing bends. Therefore, the obtained routing path is the shortest, and we, thus, can focus on the objectives that we are most concerned.

We define planarization-aware cost \(\Phi_t\) for each global tile \(t\) as follows:

\[
\Phi_t = \bar{d}_t + \begin{cases} 
\kappa_p - \beta(2d_t - 1) & \text{if } d_t \geq B_u \\
\beta(2d_t - 1) - \beta(d_t - \bar{d}_t)^2 & \text{if } B_l \leq d_t < B_u \\
\kappa_n & \text{if } d_t < B_l
\end{cases}
\]

where \(d_t\) is the wire density of \(t\), \(\bar{d}_t\) is the predicted hot-spot cost calculated in the prerouting stage, \(\kappa_p\) is the average wire density of tiles adjacent to \(t\) (including \(t\)), \(B_l\) and \(B_u\) are the density lower and upper bounds specified in foundry density rules, respectively, and \(\beta\), where \(0 \leq \beta \leq 1\), is a user-defined parameter [note that both the values of \(2d_t - 1\) and \((d_t - \bar{d}_t)^2\) are between 0 and 1]. \(\kappa_p\) and \(\kappa_n\) are constants, where \(\kappa_p\) is a positive penalty that hinders the overdenseness in the global tile, and \(\kappa_n\) is a negative reward that encourages paths to go through sparse tiles. (Note that, unlike maze routing, the global pattern routing allows such negative rewards.) The second equation simultaneously considers the local density and minimizes the density difference among adjacent regions.

For a more balanced wire distribution, cost function \(\Phi_p\) of global routing path \(g_p\) is defined as follows:

\[
\Phi_p = \operatorname{avg}\{\Phi_t \mid \text{tile } t \text{ is on path } g_p\}
\]
in which the average manner can represent the consciousness of an even wire distribution.

Note that, as shown in Fig. 4, the TTR iteratively refines the solution during the first-pass stage. Specifically, a rip-up and reroute procedure applying (3) is performed at the end of each level to progressively fine-tune the global routing solution at that level, which not only helps reduce the net-ordering problem of sequential routing but also facilitates the transitions between levels.

C. Density-Driven Layer/Track Assignment

A recent routing system considering the balanced wire distribution often performs density-driven global routing that is directly followed by detailed maze routing [28]. However, even if the detailed router finds routing paths inside the global tiles that are determined by the global router, an uneven wire distribution may still occur in each global tile. For example, Fig. 8(a) shows a detailed routing result where an unbalanced wire distribution occurs in the tiles. In contrast, if wire planning is performed first after global routing, the detailed router could make wiser decisions and obtain better solutions, as illustrated in Fig. 8(b). Therefore, it is desirable to perform wire planning before detailed routing.

Recently, Cong et al. [16] have proposed the first wire-planning scheme between global and detailed routers to reduce congestion. Barytawala et al. [2] also suggested adding a track assignment stage between global and detailed routing to improve the routing quality. Ho et al. [20] developed a layer/track assignment heuristic in the intermediate stage for crosstalk optimization. Later, Ho et al. [21] further extended their track assigner for the wirelength reduction in X-architecture routing. Cho et al. [15] also performed wire planning in track assignment using second-order cone programming for yield (critical area) optimization. However, the wire density is not addressed in these works.

1) Density-Driven Layer Assignment (DLA): In this paper, we propose a new layer/track assignment algorithm for wire-density optimization. To our best knowledge, this is the first work of wire planning that addresses the wire-density optimization in the literature.

We handle long horizontal (vertical) segments that span more than one complete global tile in a row (column) in the middle layer/track assignment stage and delegate short segments to the detailed router. The full row (or column) of a global tile array is called a row (column) panel. We will refer to a row panel as a panel throughout this paper for brevity, unless specified otherwise.

In a panel, the local density of a column is defined as the total number of segments and obstacles at that column, and the panel density is the maximum local density among all columns in all layers. For example, Fig. 9(a) gives a row panel with 11 columns, i.e., c₁–c₁₁. There are six segments s₁–s₆ in the panel and two obstacles o₁ and o₂ in layers, and its panel density is equal to 4. We intend to evenly arrange these segments to two horizontal layers (for example, layers 1 and 3) while minimizing the panel density at each layer. The density-driven layer assignment (DLA) problem is defined as follows.

- **The DLA problem.** Given a set L of layers, a set S of disjoint segments in a panel, and a set O of fixed obstacles in layers, assign each segment of S to a layer such that, for each layer, the maximum local density is minimized, and the panel density is minimized.

Note that, for practical concern, in addition to the objectives of the DLA, a good/practical layer assigner shall also assign layers with more segments of the same nets closer to each other to minimize the stacked-via usage (i.e., minimize the layer transition of signal wires to reduce the routing resource and congestion).

To solve the DLA problem, we adopt a two-phase technique of 1) density-driven layer partitioning followed by 2) stacked-via aware layer assignment. The layer-partitioning step partitions the segments and the obstacles in each panel into |L| layer groups such that the main objective of the DLA is achieved, whereas the stacked-via-aware-layer-assignment step decides the actual layer for each layer group in panels to make layers with more segments of the same nets closer to each other to minimize the stacked-via usage.

In the first layer-partitioning phase, we build the horizontal constraint graph HCG(V,E) for S and O in the panel. Each vertex v ∈ V corresponds to a segment or an obstacle, and two vertices v₁ and v₂ are connected by an edge e ∈ E if their spans overlap. The cost of edge e(v₁, v₂) is defined as the maximal local density among the overlapping columns between v₁ and v₂. With this weighting policy, if two vertices are connected by an edge with a high cost, they should be separated into different layers. Fig. 9(b) shows the HCG of the panel in Fig. 9(a). Here, obstacle o₂ and segment s₃ overlap in columns c₃ and c₄, and the maximal local density of c₃ and c₄ is 3. Therefore, the cost of edge (o₂, s₃) is equal to 3.

Consequently, we can formulate the DLA problem as a max-cut k-coloring problem (MCP) [14] on the HCG graph, where k is equal to |L|. This way, we can guarantee that the partitioning result can evenly distribute the segments of the maximal local density to different layer groups. However, the MCP is NP-complete [14]. Thus, we resort to a simple, yet efficient, heuristic by constructing a maximum spanning tree on the HCG and applying a k-coloring algorithm on this tree. Note that the k-coloring algorithm on a tree can be solved in linear time. Fig. 9(c) shows a layer-partitioning result of Fig. 9(a), where s₁, s₂, s₃, and s₆ are partitioned as one layer group, and o₂, s₄, s₅, and o₁ are partitioned as another one. Note that the objects o₁, s₃, s₅, and s₆ at columns c₉ and c₁₀ that induce the maximum local density are separated into two different layer groups.
In the second stacked-via-aware-layer-assignment phase, we determine the actual layer for each layer group in panels to minimize the layer transitions and, thus, the stacked-via usage. We build the connection graph \( C(V, E) \) for all layer groups that are obtained in the first step. Each vertex \( v \in V \) corresponds to a layer group, and the weight of an edge \( e(v_i, v_j) \in E \) is equal to the connectivity between \( v_i \) and \( v_j \). The connectivity between two layer groups is defined as the number of nets with segments in both groups. For example, if a net \( n \) has one segment \( s_i \) in layer group \( L_i \) and another segment \( s_j \) in layer group \( L_j \), the connectivity between \( L_i \) and \( L_j \) is increased by 1.

Fig. 9(d) depicts an example connection graph consisting of six layer groups, i.e., \( L_1, L_2, \ldots, L_6 \), in the three panels \( A_{c1}, A \), and \( A_{c2} \), where \( A \) is a row panel, and \( A_{c1} \) and \( A_{c2} \) are the column panels. (Note that, to simplify the presentation, the edges with zero weight are not shown in this graph). Row panel \( A \) denotes the panel described in Fig. 9(a), where layer group \( L_3 \) contains \( s_1, s_2, s_3 \), and \( s_6 \), and \( L_4 \) contains \( o_1, o_2, s_4 \), and \( s_5 \) after the layer-partitioning step in Fig. 9(c). Fig. 9(e) shows a connection among segments \( s_7, s_4 \), and \( s_8 \) that increases the connectivity by 1 between layer groups \( L_2 \) and \( L_4 \), and \( L_4 \) and \( L_6 \) of Fig. 9(d).

To minimize the stacked-via usage, we shall make layers with more segments of the same nets closer to each other. In other words, those layer groups with larger connectivity are assigned to closer layers. This problem can be solved by first computing the maximum-weighted Hamiltonian path (MWHP) on \( C(V, E) \) and then assigning layer groups with the largest connectivity closer to each other. For the example in Fig. 9(d), the MWHP is \( (L_1, L_2, L_3, L_6, L_4, L_5) \). Let \( f(l) \) be the layer assignment for layer groups \( l = (l_1, l_2, \ldots) \). Suppose that we have four metal layers, and layers 1, 2, 3, and 4 are dedicated for horizontal, vertical, horizontal, and vertical wires, respectively. Both the two layer assignment solutions \( f(L_1, L_2, L_3, L_6, L_4, L_5) = (4, 2, 1, 2, 3, 4) \) or \( (2, 4, 3, 4, 1, 2) \) can achieve the minimum layer transitions and, thus, the stacked-via usage. (Note that \( A \) is a row panel, and \( A_{c1} \) and \( A_{c2} \) are column panels.)

Since the MWHP problem is NP-hard, we resort to a greedy algorithm that is similar to Kruskal’s minimum spanning tree
algorithm to handle the MWHP problem. We first sort edges by their weights and then add edges in a nonincreasing weight order if they form a path. Fig. 9(f) shows the layer group assignment after assigning $L_3$ to layer 1 and $L_4$ to layer 3 in Fig. 9(d).

At the last step, since obstacles are already in fixed layers, we applied a minimum-impact repair procedure for obstacles. If an obstacle is not placed in the right layer (e.g., $o_1$ in Fig. 9(f)), the layer of a vertex $v_o$ of an obstacle is exchanged with that of a vertex $v_s$ of a segment such that the edge cost $(v_o, v_s)$ is the maximum among the edges connected with $v_o$ in the maximum spanning tree. If there does not exist such a vertex $v_s$, we can just assign $v_o$ to the correct layer since there is no segment there (otherwise, there must be an edge connected with $v_o$). The final assignment result after the repair procedure for exchanging the layer of vertex $o_1$ with that of vertex $s_b$ is shown in Fig. 9(g).

As a result, the final assignment has a very balanced density distribution that the average local density of layer 1 is 1.18 and that of layer 3 is 1.27, while the panel densities in both layers are equal to 2. See Fig. 9(h) and (i) for the resulting segment assignments for layers 1 and 3, respectively.

2) Density-Driven Track Assignment (DTA): After the layer assignment, we intend to uniformly spread the segments in each layer of panels and balance the segment distribution among neighboring panels. For convenience, we hereafter refer to a layer of a panel as a panel since the layer assignment has already been performed. Let $T$ be the set of tracks inside a panel. Each track $\tau \in T$ can be represented by the set of its constituent contiguous intervals. Denoting these intervals by $x_i$, we have $\tau \equiv \bigcup x_i$. Each $x_i$ is either

1) a blocked interval, where no segment from $S$ can be assigned;
2) an occupied interval, where a segment from $S$ has been assigned;
3) a free interval, where no segment from $S$ has yet been assigned.

A segment $s \in S$ is said to be assignable to $\tau \in T$, $\tau \equiv \bigcup x_i$, iff $x_i \cap s \neq \emptyset$, which implies that either $x_i$ is a free interval or is an interval that is occupied by a segment of the same net. Thus, the DTA problem is defined as follows.

- **The DTA problem.** Given a panel $A$ and its two neighboring panels $A_u$ and $A_b$, a set of tracks $T \in A$, a set of segments $S \in A$, and a set of fixed obstacles $O \in A$, for a given cost function, $\Psi : S \times T \rightarrow \mathbb{R}$, which represents the density cost of the assignment of $S$ to $T$ that minimizes $\Psi$.

To solve this problem, we propose an incremental Delaunay-triangulation-based track assignment (IDTA) algorithm. In Observation 1, we have discovered the relation between the density uniformity and the Voronoi diagram. Instead of using the Voronoi diagram, we can leverage the good properties of its dual graph, called the Delaunay Triangulation (DT), to evaluate the segment distribution. The DT for a point set is triangulation that minimizes the standard deviations of angles among all triangles, and the circumscribed circle of every triangle will not contain any other point in its interior [19].

In hypsography, the DT has been successfully applied for modeling terrain to study the distribution of elevations on the surface by given a set of sample points since the DT tends to make equilateral triangles (with smaller circumcircles compared to their areas) and, thus, gives a nice set of triangles to use as polygons in the model [3]. Therefore, the DT is a suitable metric of the track-assignment problem to reflect the segment distribution since, for CMP planarization, the existence of segments in the panel is analogous to elevations on the surface. Similar to the Voronoi diagram, the standard deviation for the size of triangles in the DT can reflect the distribution of these points. Thus, we can represent each segment by three points—two endpoints and one center point—and analyze the corresponding DT of these points. (Note that, if a segment cannot be assigned as a single entity, we can separate it into subnets and then use three points to model each subnet. It is also possible to use more points for each segment, but its time complexity will be higher, which is a tradeoff between accuracy and efficiency.) Therefore, we define $\Psi$ in the DTA problem as the standard deviation of the triangle sizes in the DT resulting from the track assignment.

As shown in the example in Fig. 10(a) and (b), if the segment distribution is non-uniform, it would result in a larger area difference among all triangles (i.e., the larger area gap between the minimum and maximum triangles); on the other hand, if the segment distribution is uniform, then the area difference would be small, as shown in Fig. 10(c). This way, the best assignment configuration in $A$ would be the one with the minimum area difference among all triangles in the DT.

Before performing the IDTA algorithm, we first model the distribution of segments and obstacles in each neighboring panel into an artificial segment lying on the boundary of $A$ to consider the boundary interaction [15]. To reflect the distribution of objects in a neighboring panel $A_n$ of $A$, we set the length of an artificial segment as the average occupied length per track in $A_n$, and the center of this artificial segment is determined by the center of gravity of all segments and obstacles in $A_n$. For example, given a panel $A$ [layer 1 after the layer assignment in Fig. 9(a)] and its two neighboring panels $A_u$ and $A_b$ as shown in Fig. 11(a), the length of artificial segment $s_u$ for $A_u$ is equal to $(3 + 6 + 3 + 4)/4 = 4$, and its center lies in $(3 \times 2.5 + 6 \times 5 + 3 \times 7.5 + 4 \times 9)/(3 + 6 + 3 + 4) = 6$. Similarly, the length of artificial segment $s_b$ for $A_b$ is equal to 3, and its center is at 4.5. Fig. 11(b) shows the two artificial segments $s_u$ and $s_b$ on the boundaries of $A$ for modeling $A_u$ and $A_b$ in Fig. 11(a), respectively.

Fig. 12 shows the IDTA algorithm. Without loss of generality, we discuss the track assignment at a row panel, and the case for a column panel is similar. For the track assignment problem,
The $x$-coordinates of segments are fixed (i.e., the segments in row panels can only move in the vertical direction); therefore, we can focus on the $y$-direction. At the beginning, we define the flexibility of a segment $s_i$ as

$$\xi(s_i) = t_i + \frac{1}{\ell_i}$$

where $t_i$ is the number of assignable tracks of $s_i$, and $\ell_i$ is the length of $s_i$. Since the $x$-coordinate of $s_i$ is fixed, $t_i$ can easily be computed. If the flexibility of $s_i$ is smaller, which means that $s_i$ might have longer length or less space to insert, then $s_i$ should be assigned first.

After the flexibility computation, we construct initial DT that includes only the obstacles and two artificial segments. Each segment or obstacle is represented by three points—its left endpoint, center point, and right endpoint. Fig. 13(a) shows the initial DT. The construction of the DT takes $O(|P| \log |P|)$ time, where $|P|$ is the number of points. Note that the DT can be updated incrementally; if a new point is added into the existing DT, we only need to update the triangles that are introduced by this new point. Therefore, the process can be performed very efficiently. The update will be frequently used in the following steps.

**Lemma 1**: Adding a new point into the existing DT of $|P|$ points takes $O(\log |P|)$ time.

Segments are sequentially assigned in the nondecreasing order of their flexibility. Suppose that segment $s_j$ has the smallest flexibility among all unassigned segments. Then, we assign $s_j$ to a proper track. To minimize the area difference among all triangles, the track that results in the DT with a smaller area difference is preferred. Note that, as mentioned in Section II, since the uncoarsening stage recursively partitions the chip until the size of global cells at a level is below a threshold regardless of the design size, the number of tracks in a panel is a constant. Therefore, it requires only constant time to find the best track.
After assigning \( s_j \) to track \( \text{track}(s_j) \), we need to update the DT and the flexibility of segments. Since we can incrementally update the DT, only the new triangles that are introduced by \( s_j \) need to be regenerated. Only the segments that overlap \( s_j \) and are originally assignable to \( \text{track}(s_j) \) need to update their values of flexibility. For those segments, the new flexibility would be the original flexibility minus 1. The number of segments overlapping with \( s_j \) is bounded by \( \ell_j \times t_j \), which is bounded by the constant size of the panel; here, \( \ell_j \) is a value, and \( t_j \) is bounded by the number of tracks in a panel, which is predetermined before the routing and is around 10–20 in our implementation. Therefore, the total time complexity of updating the DT and the flexibility of segments is \( O(|S|) \), and we have the following theorem for the overall time complexity of the IDTA algorithm.

**Theorem 2:** The IDTA algorithm runs in \( O(|S| \log |S|) \) time, where \(|S|\) is the number of segments in a panel.

**Proof:** For the assignment of a segment \( s_j \) of the length \( \ell_j \) in the DT with \(|P|\) points, it takes \( O(\ell_j \log |P|) \) time to choose the best track location (where \( t_j \) is the number of assignable \( s_j \)). \( O(|P|) \) time to update the DT, and \( O(\ell_j t_j) \) time to recompute the flexibility. Since \( \ell_j \) and \( t_j \) are constants and \(|P|\) is bounded by \( |S| \), the overall time complexity for assigning \( s_j \) is \( O(|S| \log |S|) \), and, thus, the overall time complexity for the IDTA in a panel is \( O(|S| \log |S|) \).

Fig. 13 shows the track assignment results coming from Fig. 11(b). Fig. 13(a) shows the initial DT, including only the best track location (where \( s_j \) is originally assignable to \( \text{track}(s_j) \)). Fig. 13(b)–(d) shows the IDTA in a panel is \( O(|S| \log |S|) \). We compared the proposed two-pass top-down routing framework of the TTR with the grid-based full-chip multilevel router considering balanced routing density in [28] (named MROR). The MROR program was provided by Li et al. [28] and was run on the same machine. For fair comparison, the TTR used the same setting for the size of routing tiles in all benchmarks as the MROR. Note that, as reported in [28], the MROR achieves better solutions than the previous work [4], and, thus, we shall directly compare the TTR with the MROR.

In addition, we also examined the effects of the Voronoi-diagram-based density CAA in the TTR by comparing with the minimum-pin density routing algorithm presented in [13]. Note that Cho et al. [13] applied their algorithm in an ILP-based global router called the BoxRouter [12]. Therefore, to focus on the comparison of the two CAA algorithms, we integrated the minimum-pin density routing algorithm into the TTR. In other words, we removed the prerouting of the TTR and replaced the cost function of the global router in (3) by the minimum-pin density routing algorithm. Note that we did not intend to directly compare with that of Cho et al. [13] since that work and ours have different objectives—[13] works on global routing alone, whereas ours is for the complete routing solution.

### IV. Experimental Results

The TTR routing system was implemented in the C++ programming language on a 1.2-GHz SUN Blade-2000 workstation with 8-GB memory. We used the LEDA packages to compute the Voronoi diagrams and the DT. We conducted the experiments based on the 11 MCNC routing benchmarks (provided by Cong et al. [17]) and five real industrial Faraday benchmarks (introduced in [1]). Tables I and II list the two sets of benchmark circuits. Note that, in the tables, “Tile Size” gives the tile size of each circuit that the TTR used, and \( r \) reports the radius of the Voronoi-diagram-based CAA of each circuit, i.e., the average distance among pins of adjacent Voronoi cells.

In our implementation, parameter \( \alpha \) in (2) was set to 0.5, and parameters \( \beta, \kappa_1, \kappa_2, B_1, \) and \( B_2 \) in (3) for all benchmarks were given as 0.5, 2, –2, 10%, and 40%, respectively. (Note that, if the wire width is equal to the wire spacing, the maximum wire density for a fully wire-occupied routing tile is equal to 50%; therefore, we set the respective density lower and upper bounds as 10% and 40%. Also, 0.5 means the same weighting for the two costs.)

We compared the proposed two-pass top-down routing framework of the TTR with the grid-based full-chip multilevel router considering balanced routing density in [28] (named MROR). The MROR program was provided by Li et al. [28] and was run on the same machine. For fair comparison, the TTR used the same setting for the size of routing tiles in all benchmarks as the MROR. Note that, as reported in [28], the MROR achieves better solutions than the previous work [4], and, thus, we shall directly compare the TTR with the MROR.

### IV. Experimental Results

The TTR routing system was implemented in the C++ programming language on a 1.2-GHz SUN Blade-2000 workstation with 8-GB memory. We used the LEDA packages to compute the Voronoi diagrams and the DT. We conducted the experiments based on the 11 MCNC routing benchmarks (provided by Cong et al. [17]) and five real industrial Faraday benchmarks (introduced in [1]). Tables I and II list the two sets of benchmark circuits. Note that, in the tables, “Tile Size” gives the tile size of each circuit that the TTR used, and \( r \) reports the radius of the Voronoi-diagram-based CAA of each circuit, i.e., the average distance among pins of adjacent Voronoi cells.

In our implementation, parameter \( \alpha \) in (2) was set to 0.5, and parameters \( \beta, \kappa_1, \kappa_2, B_1, \) and \( B_2 \) in (3) for all benchmarks were given as 0.5, 2, –2, 10%, and 40%, respectively. (Note that, if the wire width is equal to the wire spacing, the maximum wire density for a fully wire-occupied routing tile is equal to 50%; therefore, we set the respective density lower and upper bounds as 10% and 40%. Also, 0.5 means the same weighting for the two costs.)

We compared the proposed two-pass top-down routing framework of the TTR with the grid-based full-chip multilevel router considering balanced routing density in [28] (named MROR). The MROR program was provided by Li et al. [28] and was run on the same machine. For fair comparison, the TTR used the same setting for the size of routing tiles in all benchmarks as the MROR. Note that, as reported in [28], the MROR achieves better solutions than the previous work [4], and, thus, we shall directly compare the TTR with the MROR.

In addition, we also examined the effects of the Voronoi-diagram-based density CAA in the TTR by comparing with the minimum-pin density routing algorithm presented in [13]. Note that Cho et al. [13] applied their algorithm in an ILP-based global router called the BoxRouter [12]. Therefore, to focus on the comparison of the two CAA algorithms, we integrated the minimum-pin density routing algorithm into the TTR. In other words, we removed the prerouting of the TTR and replaced the cost function of the global router in (3) by the minimum-pin density routing algorithm. Note that we did not intend to directly compare with that of Cho et al. [13] since that work and ours have different objectives—[13] works on global routing alone, whereas ours is for the complete routing solution.

Tables III and IV show the comparison results on the MCNC and Faraday benchmarks, respectively. Note that, since the MROR program can only handle the designs with all pins lying in layer 1 (as in the MCNC benchmarks), we did not conduct the experiments on the Faraday benchmarks (where pins are distributed between layers 1 and 3) for the MROR.

Note that, according to the wire-density predictive CMP model characterized in [13], the post-CMP copper thickness is proportional to the square of the wire density. Therefore, it is desirable to minimize the variation of the wire density for CMP control, which is the main objective of this paper. As a result,
TABLE III
COMPARISON FOR THE WIRE DENSITY CONTROL ON THE MCNC BENCHMARKS

<table>
<thead>
<tr>
<th>Circuit</th>
<th>#Netmax</th>
<th>#Netseg</th>
<th>#Netdgk</th>
<th>σr</th>
<th>σp</th>
<th>CPU (sec)</th>
<th>Minimum density global routing [13] + TTR’s routing framework</th>
<th>TTR (Ours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mcc1</td>
<td>45</td>
<td>9.9</td>
<td>11.3</td>
<td>7.6</td>
<td>7.3</td>
<td>124 2600</td>
<td>41 10.3 11.1 5.1 7.6 36.1 124 2639 30 10.3 11.0 5.9 6.4 33.4</td>
<td></td>
</tr>
<tr>
<td>Mcc2</td>
<td>96</td>
<td>18.7</td>
<td>20.9</td>
<td>17.3</td>
<td>18.5</td>
<td>2714.9 256 15814 119 20.6 22.2 14.4 19.6 798.0 256 16644 87 20.5 22.2 13.9 16.0 645.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Struct</td>
<td>7</td>
<td>1.4</td>
<td>1.4</td>
<td>1.1</td>
<td>1.6</td>
<td>61.4 193 2128 5 1.2 0.8 0.9 0.8 66.8 167 2124 6 1.1 0.8 1.1 1.0 58.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Primary1</td>
<td>15</td>
<td>0.7</td>
<td>0.6</td>
<td>1.2</td>
<td>1.8</td>
<td>69.1 328 2423 12 0.8 0.7 0.9 1.4 27.0 215 2207 6 0.7 0.3 0.9 0.8 24.3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Primary2</td>
<td>25</td>
<td>2.1</td>
<td>1.9</td>
<td>1.6</td>
<td>2.8</td>
<td>322.2 387 8338 22 2.5 1.9 1.3 1.8 144.0 303 7693 8 1.8 0.9 1.3 1.6 131.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S373</td>
<td>15</td>
<td>4.4</td>
<td>3.5</td>
<td>3.2</td>
<td>1.6</td>
<td>4.5 87 1091 8 2.5 2.4 1.6 1.5 8.1 91 1193 9 2.5 2.4 1.8 1.5 8.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S9234</td>
<td>14</td>
<td>4.0</td>
<td>2.6</td>
<td>3.2</td>
<td>1.6</td>
<td>3.2 95 912 7 1.7 1.6 1.4 1.3 5.2 95 1003 9 1.7 1.6 1.2 1.6 5.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S13207</td>
<td>27</td>
<td>9.3</td>
<td>5.9</td>
<td>5.2</td>
<td>2.8</td>
<td>15.8 97 1727 13 3.4 3.0 2.1 1.8 24.8 97 1821 11 3.3 3.0 2.3 1.7 24.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S15850</td>
<td>26</td>
<td>10.3</td>
<td>7.4</td>
<td>5.4</td>
<td>2.9</td>
<td>23.8 97 1834 12 4.0 3.8 2.3 1.9 34.2 97 1915 13 3.9 3.8 2.4 1.9 33.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S38417</td>
<td>23</td>
<td>7.5</td>
<td>4.3</td>
<td>4.2</td>
<td>1.3</td>
<td>54.2 188 5043 16 10 3.0 2.4 1.8 1.4 62.5 188 5462 11 2.9 2.4 2.0 1.4 62.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>S38584</td>
<td>29</td>
<td>9.1</td>
<td>5.8</td>
<td>5.4</td>
<td>2.9</td>
<td>137.7 189 6004 16 3.3 3.1 2.3 1.6 112.0 189 6338 15 3.3 3.1 2.3 1.6 112.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Comp</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>- - 0.68 0.72 0.74 0.59 0.65 1.01 - - 0.57 0.66 0.64 0.65 0.98</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TABLE IV
COMPARISON FOR THE WIRE DENSITY CONTROL ON THE INDUSTRIAL FARADAY BENCHMARKS

<table>
<thead>
<tr>
<th>Circuit</th>
<th>#Netmax</th>
<th>#Netseg</th>
<th>#Netdgk</th>
<th>σr</th>
<th>σp</th>
<th>CPU (sec)</th>
<th>Minimum density global routing [13] + TTR’s routing framework</th>
<th>TTR (Ours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMA</td>
<td>99.19%</td>
<td>114</td>
<td>272</td>
<td>5168</td>
<td>14</td>
<td>3.14 2.77 1.70 1.77 48.8</td>
<td>99.29% 101 272 5525 10 3.08 2.70 1.75 1.64 47.0</td>
<td></td>
</tr>
<tr>
<td>DSP1</td>
<td>99.11%</td>
<td>157</td>
<td>264</td>
<td>4241</td>
<td>11</td>
<td>2.91 2.50 1.95 1.89 124.2</td>
<td>99.18% 145 263 4529 10 2.85 2.44 2.24 1.95 117.3</td>
<td></td>
</tr>
<tr>
<td>DSP2</td>
<td>99.10%</td>
<td>158</td>
<td>268</td>
<td>4674</td>
<td>11</td>
<td>2.78 2.78 1.71 1.92 87.2</td>
<td>99.06% 164 268 4892 10 2.72 2.70 1.90 1.91 82.3</td>
<td></td>
</tr>
<tr>
<td>RISC1</td>
<td>99.16%</td>
<td>223</td>
<td>265</td>
<td>5864</td>
<td>21</td>
<td>3.63 3.79 2.95 3.78 355.3</td>
<td>99.16% 221 265 6226 17 3.59 3.73 3.08 3.29 333.4</td>
<td></td>
</tr>
<tr>
<td>RISC2</td>
<td>99.23%</td>
<td>199</td>
<td>260</td>
<td>6141</td>
<td>21</td>
<td>3.64 3.70 2.55 3.08 297.4</td>
<td>99.19% 209 260 6533 13 3.59 3.62 2.77 2.89 280.0</td>
<td></td>
</tr>
<tr>
<td>Comp</td>
<td>99.10%</td>
<td>- - -</td>
<td>1.00</td>
<td>1.00</td>
<td>1.00</td>
<td>- - 1.00 1.00 1.00 1.00</td>
<td>99.18% - - - - 0.75 0.98 0.98 1.08 0.95 0.95</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 14. Routing result and the vertical wire-crossing map in tiles for “S13207.” (The red, green, and blue lines represent metals 1, 2, and 3, respectively.) (a) and (b) Routing layout and its vertical wire-crossing of the MROR [28]. The maximum vertical wire crossing is 27. (c) and (d) Routing layout and its vertical wire crossing obtained from the minimum-pin density global routing [13] + TTR’s routing framework. The maximum vertical wire crossing is 13. (e) and (f) Routing layout and its vertical wire crossing of the TTR (ours). The maximum vertical wire crossing is only 11.
Fig. 15. Routing result and the wire-crossing map in tiles for “RISC1.” (The red, green, blue, magenta, coffee, and aqua blue lines represent metals 1, 2, 3, 4, 5, and 6, respectively, and the white space is allocated by seven macros.) (a)–(c) Routing layout and its horizontal and vertical wire crossings obtained from the minimum-pin density global routing [13] + TTR’s routing framework. Maximum horizontal and vertical wire crossings are 21 and 14, respectively. (d)–(f) Routing layout and its horizontal and vertical wire crossings of the TTR (ours). The maximum horizontal and vertical wire crossings are 17 and 14, respectively.

we adopted the same metrics as those used in [28], which evaluate the uniformity of the wire distribution in the routing stage.

In the tables, “Rout.” stands for routability, “#Fail” gives the number of failed two-pin nets, “#Net_{max}” denotes the maximum number of nets crossing a level-0 tile, “#Net_{avg,h}” represents the average number of nets horizontally crossing a tile (“σ_h” gives its standard deviation), and “#Net_{avg,v}” gives the average number of nets vertically crossing a tile (“σ_v” gives its standard deviation). For the TTR routing systems, “#LG” denotes the total number of layer groups for the layer assignment, and “#Seg” shows the total number of segments. As shown in the tables, all routers obtain 100% routing completion on the MCNC benchmarks, and both routers applying the new framework of the TTR outperform the multilevel router MROR in wire uniformity. Compared with the MROR, the TTR incorporated with the minimum-pin density global routing algorithm reduces #Net_{max}, #Net_{avg,v}, and #Net_{avg,h} by 32%, 28%, and 26%, respectively, and the TTR with the Voronoi-diagram-based CAA can achieve 43%, 34%, and 36% reductions on #Net_{max}, #Net_{avg,v}, and #Net_{avg,h}, respectively. Moreover, the routers using the TTR framework also result in at least 35% smaller standard deviations of the wire distribution in both directions (which implies better density smoothness) than the MROR. (Note that the measure is on the statistics of wire crossings, not for the CMP thickness directly.) From the results reported in Tables III and IV, the global routing guided by the Voronoi-diagram-based CAA can achieve better wire uniformity than the minimum-pin density global router; in particular, it can reduce the running time of the overall routing process, although the Voronoi-diagram-based CAA itself has higher time complexity than the minimum-pin density scheme alone. The results show the effectiveness of the Voronoi-diagram-based CAA. Fig. 14 shows the routing layouts of “S13207” and the corresponding wire-crossing maps in the vertical direction for the aforementioned three routers, and Fig. 15 shows the results for the Faraday circuit “RISC1” and the horizontal and vertical wire-crossing maps. The experimental results consistently show the superior effectiveness and efficiency of our routing algorithm and framework in wire-density control.

Tables V and VI show the detailed running times of the TTR routing system on the MCNC and Faraday benchmarks. In the two tables, “PR,” “GR,” “LA,” “TA,” and “DR” stand for prerouting, global routing, layer assignment, track assignment, and detailed routing, respectively. As shown in the tables, the Voronoi-diagram-based CAA incurs only about 3% of the total running time. In particular, it even leads to a 5% faster overall routing process than the minimum-pin density global routing algorithm; the reason for the better efficiency is due to easier density control for later detailed routing for the Voronoi-diagram-based CAA.
TABLE V

<table>
<thead>
<tr>
<th>Circuit</th>
<th>Minimum pin density routing [13] + TTR’s routing framework</th>
<th>TTR (Ours)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Minimum pin density cost</td>
<td>Global routing</td>
</tr>
<tr>
<td></td>
<td>GR</td>
<td>LA</td>
</tr>
<tr>
<td>Mcc1</td>
<td>0.09</td>
<td>0.11</td>
</tr>
<tr>
<td>Mcc2</td>
<td>1.03</td>
<td>2.20</td>
</tr>
<tr>
<td>Struct</td>
<td>0.02</td>
<td>0.03</td>
</tr>
<tr>
<td>Primary1</td>
<td>0.06</td>
<td>0.04</td>
</tr>
<tr>
<td>Primary2</td>
<td>0.16</td>
<td>0.35</td>
</tr>
<tr>
<td>S5378</td>
<td>0.02</td>
<td>0.01</td>
</tr>
<tr>
<td>S9234</td>
<td>0.02</td>
<td>0.01</td>
</tr>
</tbody>
</table>

V. CONCLUSION

We have presented a new two-pass top-down full-chip grid-based router, named the TTR, considering the wire density for CMP variation control. The TTR features a new Voronoi-diagram-based density critical area analyzer, a planarization-aware global router, a layer assigner for panel-density minimization, and an effective track assigner based on the incremental DT. Experimental results have shown the effectiveness and the efficiency of the proposed methods.

In this paper, we have developed a metric for the DTA problem by taking three sampling points per segment and evaluating the resulting DT. Future work shall include the development of a more sophisticated model/metric for the DTA problem, e.g., the investigation of the tradeoff between the number of sample points and the effectiveness/accuracy of the metric.

In this paper, we have focused on signal nets. Further work shall include the routing for special nets such as power/ground/clock nets. Power/ground/clock nets are typically routed in an earlier stage than the signal nets. Therefore, during the routing of signal nets, we can treat these routed nets as obstacles and consider their metal density by adding this preoccupied density to the planarization-aware cost \( d_i \) in (2)) for further processing. Furthermore, large-scale designs with millions of nets are already in production, and modern router designs often need to consider multiple objectives such as wiring congestion, timing, area, reliability, and manufacturability; these requirements have reshaped the routing problem and impose significant challenges to the development of modern routers.

REFERENCES
