A Novel Blockage-avoiding Macro Placement Approach for 3D ICs based on POCS

Jai-Ming Lin, Po-Chen Lu, Heng-Yu Lin, Jia-Ting Tsai
jmlin@ee.ncku.edu.tw,borson432@gmail.com,n26090059@gs.ncku.edu.tw,n26104971@gs.ncku.edu.tw
National Cheng Kung University, Tainan, Tawian

ABSTRACT

Although the 3D integrated circuit (IC) placement problem has been studied for many years, few publications devoted to the macro legalization. Due to large sizes of macros, the macro placement problem is harder than cell placement , especially when preplaced macros exist in a multi-tier structure. In order to have a more global view, this paper proposes the partitioning-last macro-first flow to handle 3D placement for mixed-size designs, which performs tier partitioning after placement prototyping and then legalizes macros before cell placement. A novel two-step approach is proposed to handle 3D macro placement. The first step determines locations of macros in a projection plane based on a new representation, named K-tier Partially Occupied Corner Stitching. It not only can keep the prototyping result but also guarantees a legal placement after tier assignment of macros. Next, macros are assigned to respective tiers by Integer Linear Programming (ILP) algorithm. Experimental results show that our design flow can obtain better solutions than other flows especially in the cases with more preplaced macros.

CCS CONCEPTS

• **Hardware → Placement**.

KEYWORDS

physical design, macro placement, 3D IC

ACM Reference Format:
Jai-Ming Lin, Po-Chen Lu, Heng-Yu Lin, Jia-Ting Tsai. 2022. A Novel Blockage-avoiding Macro Placement Approach for 3D ICs based on POCS. In *Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD'22)*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3508352.3549352

1 INTRODUCTION

3D ICs are considered as a promising solution to extend the 2D scaling trajectory predicted by Moore's Law. 3D ICs have attracted more attention in recent years because they have several advantages such as short wirelength, low power, and a small die area. Since logic devices connected by one net can be allocated to different tiers, a 3D IC obtains shorter wirelength than a 2D IC, which causes it to have better performance.

As design complexity continuously increases, a modern Very Large Scale Integration (VLSI) chip contains more and more macros. Since the locations of macros will determine the distribution of standard cells, it is necessary to find proper locations of macros in order to get short wirelength and small routing congestion. However, macro legalization is much harder than standard cells because of large areas of macros. Moreover, the problem becomes even more difficult due to an irregular placement region caused by many pre-placed macros. Therefore, the industry is desperate to have an efficient and effective macro placer.

Several researches discussing 3D placement have been proposed, and they can be roughly divided into two categories. One is the true-3D approach [10], [16], [17] and the other is the pseudo-3D approach [4], [6], [12], [19]. True-3D approach extends existing 2D analytical placers to handle the 3D placement problem. They first perform 3D global placement to spread components over three-dimensional space while optimizing wirelength. Then, components in different tiers are legalized separately, where Cong *et al.* [10] and Luo *et al.* [17] legalize macros tier-by-tier by solving linear programming (LP) according to constraint graphs. ePlace-3D [16] applies the simulated annealing (SA) algorithm to legalize macros in multiple tiers at the same time. However, the global optimization result may be violated since a macro can be moved from a tier to another by a perturbation operation of SA.

Pseudo-3D approach utilizes 2D commercial tools to generate 3D IC layouts, which can be further classified into two types according to their design flows. The partitioning-first flow (PF flow) [6] first allocates macros and standard cells to different tiers by a partitioning algorithm before placement of components. Design complexity is reduced instantly after a netlist is divided into several tiers; however, it may induce relatively poor solution quality because of the lack of placement information during tier partitioning. After partitioning components into different tiers, Cascade-2D [6] merges all tiers into one plane in order to perform place and route (P&R) by a 2D commercial tool. On the contrary, the partitioning-last flow (PL flow) [12], [19] performs tier partitioning after distributing macros and standard cells over a 2D placement region. This makes it have a more global view than the PF flow so that it can get small wirelength and power consumption in the resulting placement. In order to place components over multi-tiers in one 2D plane, Compact-2D [12] first enlarges a chip area before applying a 2D P&R tool. Unlike Compact-2D, Shrunk-2D [19] shrinks components and wires so that it is possible to perform the 2D P&R on an original chip area. Recently, Bamberg *et al.* [4] proposed a 3D IC design flow in the face-to-face 3D IC structure, named Macro-3D. A chip is divided into a macro tier and a logic tier, respectively. After components in respective tiers are placed, signal nets are routed into the contiguous routing layers between the two stacked tiers by a commercial tool.

ICCAD '22, October 30-November 3, 2022, San Diego, CA, USA
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9217-4/22/10…$15.00
https://doi.org/10.1145/3508352.3549352

In addition to determining relative locations of macros in a 2D plane, we need to assign macros to different tiers in 3D macro placement, which makes the problem harder than 2D macro placement. Most of 3D placement researches focus on the global placement which determines the distribution of cells and macros in a 3D placement region. There exist limited works devoted to macro legalization in the multi-tier structure of a 3D IC. In addition, existing macro legalization researches focus on 2D IC designs [5], [7], [8], [9], and they are based on the SA algorithm. Recently, Lin *et al.* [14] [15] apply the simulated evolution (SE) algorithm and Corner Stitching (CS) [18] representation to implement a fast macro placement in order to speed up the procedure.

1.1 Our Contribution

No matter in the true-3D or in the pseudo-3D, macros are legalized tier-by-tier after they are assigned to respective tiers. Even though macros are allocated to tiers according to a global placement result in the true-3D or the PL flow of the pseudo-3D, they may overlap each other after tier partitioning. In order to get a legal solution, they have to push macros away from their original locations, which cause the global optimization result to be deteriorated. In the worst condition, they may fail to get a feasible solution, especially when a 3D IC contains many preplaced macros in a tier.

To resolve the problems mentioned in the above, this paper proposes the first work to focus on getting a better macro placement result in a 3D IC with preplaced macros. The characteristics of this paper are summarized as follows:

1.1.1 Partitioning-last macro-first Flow. we propose the partitioning-last macro-first flow (PL-MF) to handle blockage-aware 3D placement for a mixed-size design. In order to have a global view, we first perform the placement prototyping to find a better distribution of components in a projection 2D plane before partitioning. To avoid destroying the optimization result, we will place macros to respective tiers before tier assignment and legalization of standard cells.

1.1.2 K-tier Partially Occupied Corner Stitching Representation. we propose a novel K-tier Partially Occupied Corner Stitching (K-POCS) representation and define three feasibility conditions. As the result of K-POCS, macros can be directly shifted to respective tiers without needing further legalization in a 2D plane even though there exist preplaced macros.

1.1.3 Greedy algorithm to Check the Feasibility of K-POCS. we propose a greedy algorithm to speed up the runtime of checking a feasibility condition of K-POCS.

1.1.4 Consideration of Preplaced Macros in 3D ICs. our methodology is feasible, even though 3D ICs contain many preplaced macros.

2 PROBLEM FORMULATION AND PRELIMINARY

2.1 Problem Formulation

Given a K-tier 3D IC with preplaced macros and a netlist, our methodology determines the locations and tiers of movable macros in a fixed-outline 3D IC so that no macros will overlap in any tier. Let M_p denote a set $\{\widehat{m}_1, \widehat{m}_2, \dots\}$ of preplaced macros while M_m denote a set $\{m_1, m_2, \dots\}$ of movable macros. The coordinate and tier of $m_i, \forall m_i \in M_m$, are represented by (x_i, y_i) and t_i, respectively. Our objective is to minimize total wirelength.

2.2 Review of Macro Placement based on Corner Stitching

Lin *et al.* [14] proposed an iterative 2D macro placement algorithm based on CS [18], where a two-dimensional plane is divided into a set of rectangular regions by placed macros. A solid tile will be generated after a region is occupied by a macro. By extending the upper and lower edges of the solid tile horizontally, a set of space tiles is formed, where the extension lines cannot pass through solid tiles. Fig. 1 (a) shows an example of a macro placement recorded by CS. The four corners of a space tile are candidate locations for a new placement. Each time we will place a new macro at the corner with the best cost, where the macro cannot overlap with other solid tiles.

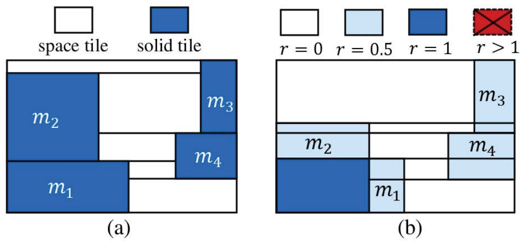

Figure 1: **(a) A macro placement based on CS. (b) A macro placement based on 2-POCS.**

3 OUR METHODOLOGY

This section overviews our methodology. Fig. 2 (a) shows the flowchart of our methodology. We first perform placement prototyping to obtain an initial placement in a projection plane, denoted by $\mathcal{P}$, by projecting 3D placement resource in a 2D plane and distributing standard cells and macros in $\mathcal{P}$. Then, 3D macro placement determines legal locations of macros in 3D plane and the detailed procedure is illustrated in Sec. 5. After the locations of macros are determined, standard cells in $\mathcal{P}$ are allocated to respective tiers. Since the number of through silicon vias (TSVs) in each tier will be determined after this stage, we assign the locations of TSVs according to the placement in each tier. Finally, standard cell placement and signal net routing are completed by a commercial tool.

3.1 Placement Prototyping in a Projection Plane

Cells and macros are spread over a 2D projection plane $\mathcal{P}$ by the analytical placer [13]. To distribute objects over a placement region evenly, we first divide $\mathcal{P}$ into uniform bins, denoted by b's, and estimate placement utilization in each bin. Let M_b denote the maximal allowable placement area in b, which is estimated by the following equation:

$$M_b = t_{density}(Kw_bh_b - \textstyle\sum_k P_{b,k}) \quad (1)$$

where K is the number of tiers in a 3D IC, and w_b (h_b) denotes the width (height) of b. Since our approach does not shrink macro areas, we have to enlarge placement area in each bin to consider placement resource in K tiers. $P_{b,k}$ denotes the occupied area of b in

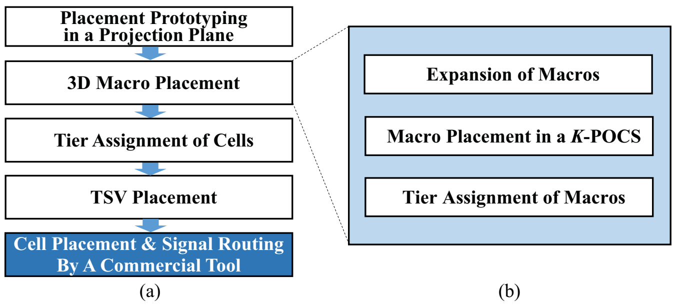
Placement Prototyping in a Projection Plane
3D Macro Placement
Tier Assignment of Cells
TSV Placement
Cell Placement & Signal Routing By A Commercial Tool
Expansion of Macros
Macro Placement in a K-POCS
Tier Assignment of Macros
(a)
(b)

Figure 2: Flowchart of our methodology.

the k-th tier by preplaced macros. Let $t_{density}$ denote a placement density in each bin.

3.2 Tier Assignment of Standard Cells

After 3D macro placement, we apply hMETIS [11] to allocate standard cells to different tiers. hMETIS is a K-way partitioning algorithm, which aims to minimize total cutsize while balancing areas in all partitions. Since macros have been assigned in the previous stage, we will fix their pins in the specified tiers before partitioning. Moreover, their areas must be subtracted from the target area of the corresponding tier to ensure area balance in all tiers.

3.3 TSV Placement

Since a cell library does not include a TSV cell, we have to determine the locations of TSVs before applying a tool to place standard cells. We first set each TSV to the location at the geometric center of pins of the net connecting to the TSV. If the location is occupied by placed macros, we move it to the closest legal location according to the breadth-first algorithm.

4 K-TIER PARTIALLY OCCUPIED CORNER STITCHING

This section illustrates the K-tier Partially Occupied Corner Stitching and its feasibility conditions.

4.1 K-tier Partially Occupied Corner Stitching

To optimize 3D macro placement, we first perform macro placement in a 2D plane $\mathcal{P}$, according to the placement prototyping result. Our target is to keep the optimization result while considering preplaced macros in different tiers so that the placement result can be transformed back to a legal placement in a 3D IC.

Since each location in $\mathcal{P}$ may have more than one placement resource, macros are allowed to overlap in some conditions. Thus, we create a new representation called K-POCS, K-tier Partially Occupied Corner Stitching, which records a macro placement result by extending CS [18]. $\mathcal{P}$ will be divided into tiles, denoted by τ_i's, by placed macros. Unlike CS, each tile in K-POCS has a different occupied ratio. Let r_i in (2) denote the occupied ratio of τ_i.

$$r_i = \frac{\#\ of\ macros\ in\ \tau_i}{K}, \tag{2}$$

where K is the number of tiers in a 3D IC. A tile whose occupied ratio is larger or equal to 1 is called a fully occupied tile (FOT); otherwise, it is called a non-fully occupied tile (N-FOT). Fig. 1 (b) shows a 2-POCS for a 2 tier 3D IC, where a tile with white or sky color denotes a N-FOT while a tile with red or blue color denotes a FOT.

When we place a new macro m_i into K-POCS, only four corners of a NFOT in the K-POCS are candidates for placement. In addition, it has to meet the three conditions, which will be introduced in Sec. 4.2. After m_i is placed, the placed region of m_i may be divided into a set $\mathcal{T}$ of tiles τ_k's with different occupied ratios by existing N-FOTs in $\mathcal{P}$. To facilitate it to estimate the occupied ratio of a tile in the future, m_i will be recorded into each τ_k in $\mathcal{T}$. Next, we extend the upper and lower edges of m_i horizontally to form new tiles. Note that the extension lines cannot pass FOTs in $\mathcal{P}$.

4.2 Feasibility Conditions of K-POCS

The three feasibility conditions for placing a new macro m_i at a corner of a N-FOT in a K-POCS are shown as follows:

- Condition 1: do not result in a tile whose occupied ratio is larger than 1.
- Condition 2: find a K-partite graph in the graph G associated to the K-POCS.
- Condition 3: ensure that preplaced macros in the same tier whose vertices have the same color in the resulting K-partite graph.

It is obvious that a K-POCS cannot be transformed back to a legal macro placement in a K-tier 3D IC if Condition 1 is violated.

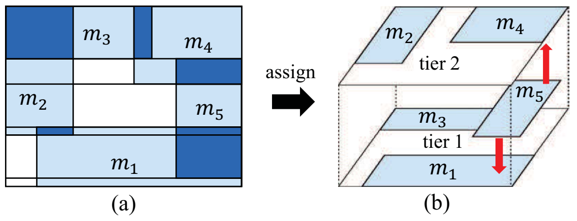

Figure 3: (a) A placement which meets Condition 1 of 2-POCS. (b) An illegal tier assignment of macros for the 2-POCS in (a).

Even though a K-POCS meets Condition 1, it still has a chance to induce an illegal 3D macro placement. For example, Fig. 3 (a) shows a 2-POCS for a 2-tier 3D IC, which meets condition 1. However, m_5 will overlap other macros no matter whether it is assigned to tier 1 or tier 2 in a possible tier assignment of macros as shown in Fig. 3 (b). Hence, condition 2 is used to exclude the above situation. To check Condition 2, we first construct a graph G according to a K-POCS as follows:

- First, we initialize a vertex n_i for each placed macro m_i.
- Second, we add an edge (n_i, n_j) between n_i and n_j if any two macros m_i and m_j overlap in the K-POCS.

Next, we check whether G is a K-partite graph, where a graph is a K-partite graph if it is possible to use K colors to paint the graph, where no two vertices on an edge are painted in the same color.

The top and bottom of Fig. 4 (a) respectively show a 2-POCS and the associated graph G. The 2-POCS meets Condition 2 since G is a 2-partite graph (i.e, it can be painted with two colors). Fig. 4 (b) shows a new 2-POCS after m_5 is placed at a N-FOT. Condition 2 is met since the associated graph G is still a 2-partite graph. However, Condition 2 is violated in the 2-POCS of Fig. 4 (c) when m_5 is placed at another N-FOT since we cannot find a 2-partite graph in the associated graph G.

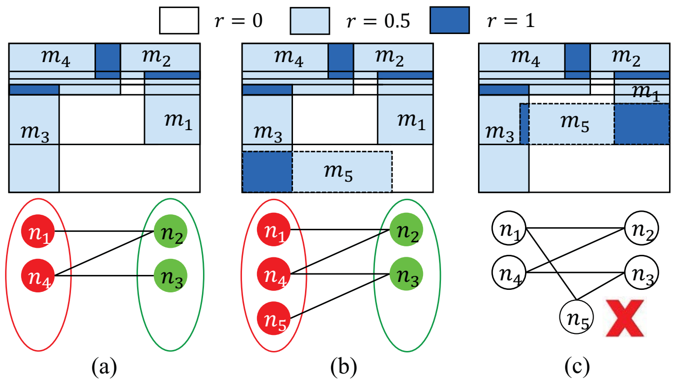

Figure 4: **(a) A 2-POCS meets Condition 2 according to the associate graph G. (b) Condition 2 is met after placement of m_5. (c) Condition 2 is violated after placement of m_5.**

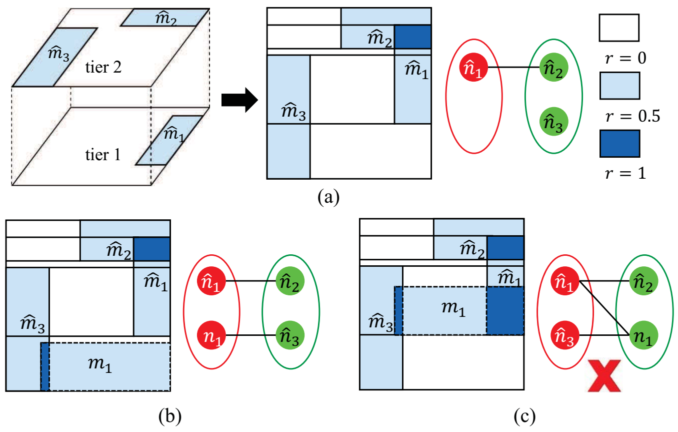

Figure 5: **(a) A 2-POCS with preplaced macros and its transformed graph G. (b) Condition 3 is met after placement of m_1. (c) Condition 3 is violated after placement of m_1.**

K-POCS still has to meet Condition 3 if a 3D IC contains preplaced macros. For example, the left of Fig. 5 (a) shows a 3D IC with three preplaced macros, where $\widehat{m}_1$ is placed in tier 1 while $\widehat{m}_2$ and $\widehat{m}_3$ both are placed in tier 2. The middle and right of the figure are the 2-POCS and the associated graph G, where G is a 2-partite graph and $\widehat{n}_2$ and $\widehat{n}_3$ can be painted in the same color. Fig. 5 (b) shows a legal 2-POCS after m_1 is placed at a N-FOT. However, the 2-POCS of Fig. 5 (c) is illegal since $\widehat{n}_2$ and $\widehat{n}_3$ cannot be painted with the same color in the resulting 2-partite graph.

4.3 Multi-partite Graph Checking

This subsection introduces two approaches to check if the graph G of a K-POCS is a K-partite graph.

4.3.1 ILP-based Approach. G is a K-partite graph if the following ILP formulation has a solution; otherwise, it is not a K-partite graph.

$$\begin{aligned} \min \quad & 0 & \text{(3a)} \\ \text{s.t.} \quad & 1 \le c_i \le K,\ \forall n_i & \text{(3b)} \\ & \widehat{c}_j = T(\widehat{m}_j),\ \forall \widehat{n}_j & \text{(3c)} \\ & c_i \ne c_j,\ \forall (n_i, n_j) & \text{(3d)} \\ & c_i \ne \widehat{c}_j,\ \forall (n_i, \widehat{n}_j) & \text{(3e)} \end{aligned}$$

The objective function is 0. Let c_i denote the color of a vertex n_i. Since each color corresponds to a specific tier number, c_i is an integer whose range is $[1, K]$ in (3b). Let $T(\widehat{m}_j)$ represent the tier number of a preplaced macro $\widehat{m}_j$. $\widehat{c}_j$ must be equal to $T(\widehat{m}_j)$ in (3c), where $\widehat{c}_j$ denotes the color of a vertex $\widehat{n}_j$. Two variables c_i and c_j cannot have the same value if they are terminals of an edge (n_i, n_j) in (3d). Similarly, two variables c_i and $\widehat{c}_j$ cannot have the same value for each edge $(n_i, \widehat{n}_j)$ in (3e). Since (3d) is not a linear constraint, it is transformed into the inequality $|c_i - c_j| > 0$. Because an absolute value is also forbidden, $|c_i - c_j|$ is replaced by a variable $s_{i,j}$. Moreover, three additional constraints are added into the ILP formulation as follows:

$$s_{i,j} > c_i - c_j \quad (4)$$

$$s_{i,j} > c_j - c_i \quad (5)$$

$$s_{i,j} > 0 \quad (6)$$

(3e) is handled by the same technique.

4.3.2 Greedy Algorithm. Since we need to check whether the associated graph G of a K-POCS is a K-partite graph whenever a new macro is placed, it may consume more runtime if we apply the ILP algorithm. Hence, we propose a greedy algorithm to speed up the procedure.

Given a 3D IC with preplaced macros, we first construct an initial K-POCS and the associated graph G. The vertices of preplaced macros are painted in advance, where those vertices associated to preplaced macros in the same tier are painted by an identical color; otherwise, they are painted by different colors. To ensure that the resulting graph is K-partite, our greedy algorithm follows two rules: 1) two connected vertices must be painted with different colors, and 2) at most K colors can be used to paint all vertices. Assume G denotes the graph before a new macro m_i is placed. After m_i is placed at a corner of a N-FOT, G is updated according to the new K-POCS. The operation is rejected if the vertices connecting to n_i already have K colors; otherwise, we determine a new color of n_i according to the remaining colors. Since choosing a different color will lead to a distinct solution when a vertex has multiple choices, the available color with the smallest number is selected to increase chances to get a feasible solution. Although the greedy algorithm will limit the solution space, the heuristic effectively increases opportunities to obtain a feasible solution.

Fig. 6 shows an example of placing macros in a 3-tire 3D IC. In the left of Fig. 6 (a) shows a 3-POCS for three placed macros m_1 to m_3. If the colors of three vertices are painted with different colors, there exists no choice for n_4 as shown in the the right of Fig. 6 (a). However, if we can paint n_2 and n_3 with the same color in the left of Fig. 6 (b), it still has a choice to paint n_4 in the right of Fig. 6 (b).

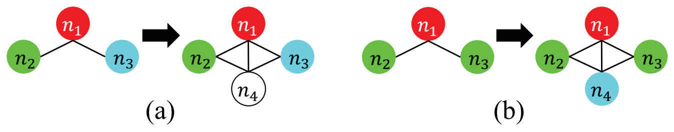

Figure 6: **(a) Coloring without heuristic. (b) Coloring according to heuristic.**

5 3D MACRO PLACEMENT

This section introduces the 3D macro placement algorithm, which is composed of three stages as shown in Fig. 2 (b).

5.1 Expansion of Macros

Each macro m_i is enlarged by some area to preserve routing resources between macros, and a macro with more pins will be expanded by a larger area since it may have more routing nets around it. Let Δ_i denote the increased area of m_i which is estimated by the following equation:

$$\Delta_i \quad = \quad \delta \frac{n_{p_i}}{n_{p_{total}}}, \tag{7}$$

where n_{p_i} denotes the number of pins of m_i and $n_{p_{total}}$ is the total number of pins for all macros. δ is a user-specified parameter.

5.2 Macro Placement Algorithm based on K-POCS

The macro placement algorithm based on K-POCS is shown as follows: first, we construct an initial K-POCS according to preplaced macros. Then, movable macros are sorted according to their areas in the non-increasing order and stored in a queue Q. Each time, we remove a macro m_i from Q and try to place it at all corners in the K-POCS. Finally, m_i will be placed at the legal corner (i.e., meet Condition 1 to 3) with the best cost by (8) and the K-POCS is updated. The procedure repeats until Q is empty.

When m_i is a legal corner, the placement quality is evaluated by the cost function Φ as follows:

$$\Phi \quad = \quad \alpha WL + \beta Disp + \gamma \frac{1}{D_{GC}} \tag{8}$$

where WL is the total wirelength according to half-perimeter model, and $Disp$ denotes the displacement of m_i from its initial location determined in the placement prototyping. D_{GC} is the distance between the current location and GC, where GC is the geometric center of N-FOTs in an initial K-POCS. We try to place macros away from GC to leave a regular region for placing standard cells. α, β and γ are user-specified parameters (α, β and γ are set as 0.1, 0.04 and 0.08 respectively, in our experiment).

5.3 Tier Assignment of Macros

After all macros are placed, we have to assign macros to different tiers from a K-POCS. Since the procedure only needs to be performed once, we apply ILP to get an optimal solution. The ILP formulation is as follows:

$$\begin{aligned}
\text{min} \quad & \sum_{i,j} net_{i,j} \times |t_i - t_j| && \text{(9a)}\\
& + \sum_{i,j} \widehat{net}_{i,j} \times |t_i - \widehat{t}_j|, \ \forall m_i, m_j, \widehat{m}_j \\
\text{s.t.} \quad & 1 \le t_i \le K, \ \forall m_i && \text{(9b)}\\
& \widehat{t}_j = T(\widehat{m}_j), \ \forall \widehat{m}_j && \text{(9c)}\\
& t_i \ne t_j, \forall (n_i, n_j) && \text{(9d)}\\
& t_i \ne \widehat{t}_j, \forall (n_i, \widehat{n}_j) && \text{(9e)}
\end{aligned}$$

Since a cutsize between two tiers implies a TSV, the objective function (9a) is to minimize the total TSVs induced by tier assignment of macros, where $net_{i,j}$ denotes the number of nets connecting to m_i and m_j and t_i denotes the assigned tier of a movable macro m_i. The value of t_i is in the range [1, K] in (9b). Similarly, $\widehat{net}_{i,j}$ denotes the number of nets connecting to m_i and $\widehat{m}_j$. A preplaced macro $\widehat{m}_j$ is set to its tier number $T(\widehat{m}_j)$ in (9c). Besides, m_i and m_j (m_i and $\widehat{m}_j$) cannot be assigned to the same tier if there exists an edge (n_i, n_j) ($(n_i, \widehat{n}_j)$) in G in (9d) ((9e)), where $|t_i - t_j|$ is replaced by a variable $r_{i,j}$. Moreover, two additional constraints are added into the formulation as follows:

$$r_{i,j} > t_i - t_j \tag{10}$$

$$r_{i,j} > t_j - t_i \tag{11}$$

The same technique is applied to $|t_i - \widehat{t}_j|$. Besides, (9d) and (9e) are handled by the same technique shown in Sec. 4.3.1.

6 EXPERIMENTAL RESULTS

Our methodology was implemented in the C++ programming language and ran on Linux workstation with 32-core Intel® Xeon® Silver 4110 2.1 GHz CPU and 400 GB memory. The ILP formulations were solved by CPLEX [2]. We tested our methodology on the circuits designed by Himax Technologies Inc. [3], and the information of the circuits is shown in Table 1. Since the original circuit was implemented in a 2D IC, we allocated preplaced macros in a 2D plane to tiers of a 3D IC in advance, where the relative locations of preplaced macros were maintained identical to that in a 2D IC if possible.

Table 1: **Our Benchmarks**

Cir.	Movable Macros	Preplaced Macros	Standard Cells	Nets
Cir1	18	13	175K	178K
Cir2	71	47	1098K	1126K
Cir3	55	15	233K	235K
Cir4	38	15	321K	327K
Cir5	32	12	347K	351K
Cir6	66	3	209K	217K
Cir7	184	0	3843K	4108K
Cir8	549	0	3746K	4481K
Cir9	260	21	1750K	2117K

For comparison with our PL-MF flow, we implemented two other design flows including PF flow and PL flow. Fig. 7 (a), (b) and (c) show the flowcharts of PF flow, PL flow and PL-MF flow, respectively, where PF flow directly partitions netlist for tier assignment while PL flow applies the bin-based approach [12] [19] according to a placement result. For fair comparison, all methods were tested in the same condition in a 3D IC. In addition, the global distribution result was obtained by an identical placement prototyping approach [13]. Moreover, we applied the legalization approach based on CS and simulated evolution algorithm [14] to legalize macros tier-by-tier in the PF flow and PL flow. After the legal locations and tiers of macros and the distribution of standard cells and TSVs were obtained, the result was fed back to IC compiler (ICC) [1] through TCL or DEF files. Finally, we used ICC to legalize standard cells and complete signal net routing in each tier. Note that the locations of macros and TSVs are fixed during legalization. Wirelength and routing overflow are measured by ICC in this experiment.

Table 2 shows the experimental results when we perform three design flows in 3D ICs with 3 tiers. The results of PF flow, PL flow, and PL-MF flows are shown in columns 2-5, 6-9, and 10-17, respectively, where columns 10-13 and 14-17 display the results when we use the ILP-based approach and greedy algorithm to implement multi-partite graph checking, respectively. Although PF flow uses less number of TSVs than PL and PL-MF flows, its

Table 2: Comparisons of PF Flow, PL Flow, and PL-MF Flow in 3-tier 3D ICs

	PF Flow				PL Flow				PL-MF Flow							
									ILP-based Approach				Greedy Algorithm			
Cir.	# of TSVs	GR. O.F.	GR. WL 10^7 μm	T (s)	# of TSVs	GR. O.F.	GR. WL 10^7 μm	T (s)	# of TSVs	GR. O.F.	GR. WL 10^7 μm	T (s)	# of TSVs	GR. O.F.	GR. WL 10^7 μm	T (s)
Cir1	1188	11813	1.19	117.72	3244	41143	1.334	84.17	1812	14379	1.194	605.66	1583	43,158	1.277	115.34
Cir2	6356	186152	7.385	911.2	11560	215539	7.106	803.18	12455	27027	7.268	4059.68	12462	31516	7.288	671.13
Cir3	2644	932	1.403	158.85	4909	4306	1.457	154.26	4650	1681	1.429	909.43	4452	1706	1.439	142.99
Cir4	2656	6298	1.858	207.63	5912	8060	1.91	175.19	4065	9933	1.929	980.41	3971	10,223	1.921	130.1
Cir5	3264	1470	0.907	222.46	9648	6316	0.978	269.79	4247	3376	0.925	1315.65	4277	1997	0.904	156.62
Cir6	3569	1227	0.987	146.31	6414	2290	1.01	150.45	5920	2213	0.97	75117.6	5474	2044	1.006	112.77
Cir7	34692	23255	17.793	3378.68	30501	524114	23.161	5622.84	59718	45917	20.9	81592.9	46825	40,906	20.8	3745.8
Cir8	9809	6866	12.387	6557.54	12720	8632	11.7	11197	27950	15545	12.24	22820.7	28894	16064	11.912	2876.23
Cir9	8614	567482	5.736	5508.43	17243	119220	5.251	6260.59	23110	61807	5.32	53182.7	16301	84787	5.1	3034.66
Nor.	0.616	1.819	0.988	1.422	1.238	3.349	1.026	1.652	1.105	0.971	0.999	82.995	1	1	1	1

Table 3: Comparisons of PF Flow, PL Flow, and PL-MF Flow in 4-tier 3D ICs

	PF Flow				PL Flow				PL-MF Flow							
									ILP-based Approach				Greedy Algorithm			
Cir.	# of TSVs	GR. O.F.	GR. WL 10^7 μm	T (s)	# of TSVs	GR. O.F.	GR. WL 10^7 μm	T (s)	# of TSVs	GR. O.F.	GR. WL 10^7 μm	T (s)	# of TSVs	GR. O.F.	GR. WL 10^7 μm	T (s)
Cir1	NA	NA	NA	NA	4942	63439	1.392	102.25	2315	23719	1.219	392.2	2273	92584	1.221	67.33
Cir2	7205	68288	6.882	1168.9	15332	110105	7.01	913.6	15890	8938	6.737	5412.92	16064	10803	7.18	749.84
Cir3	3987	1309	1.383	210.79	9812	3956	1.447	211.02	5956	2244	1.407	1227.74	6457	3302	1.45	131.99
Cir4	4495	74504	2.014	219.18	9807	15092	1.95	235.16	6489	3537	1.89	735.26	7927	3032	1.847	138.5
Cir5	4174	3929	0.927	236.67	12782	52884	1.062	403.81	7815	5129	0.939	840.88	6868	34959	0.96	160.37
Cir6	4330	1797	0.931	184.98	NA	NA	NA	NA	8696	2869	0.942	842.72	8687	2925	0.93	120.6
Cir7	62835	42629	17.444	4965.83	60888	100275	18.506	6027.54	56027	42552	18.47	41220.4	50437	32239	16.704	4678.14
Cir8	25167	10899	11.971	27045.6	27290	15164	11.57	16339.8	54955	28757	11.42	23169.3	42011	22496	11.65	2955.7
Cir9	NA	NA	NA	NA	31192	167168	5.863	10071.9	27822	99241	5.268	57494.3	25405	56781	5.12	3461.12
Nor.	0.655*	4.832*	1.006*	2.566*	1.354*	3.162*	1.065*	2.285*	1.045	0.934	1.004	8.127	1	1	1	1

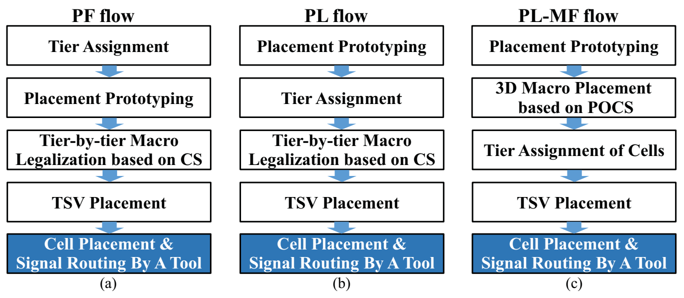
PF flow
Tier Assignment
Placement Prototyping
Tier-by-tier Macro Legalization based on CS
TSV Placement
Cell Placement & Signal Routing By A Tool
(a)
PL flow
Placement Prototyping
Tier Assignment
Tier-by-tier Macro Legalization based on CS
TSV Placement
Cell Placement & Signal Routing By A Tool
(b)
PL-MF flow
Placement Prototyping
3D Macro Placement based on POCS
Tier Assignment of Cells
TSV Placement
Cell Placement & Signal Routing By A Tool
(c)

Figure 7: Design flows for 3D ICs. (a) PF flow. (b) PL flow. (c) PL-MF flow.

routing overflow (denoted by O.F.) is significantly worse than PL-MF flow. Moreover, the PL flow gets worse results than PL-MF flow including the number of TSVs and routing overflow. In order to get legal locations of macros, the locations of macros may be changed significantly with respect to placement prototyping in the PF and PL flows, which will cause large movements of standard cells. However, placement prototyping result can be better preserved in the PL-MF flow because it finds legal locations of macros before partitioning of standard cells. Moreover, runtime of the PL-MF flow can be significantly reduced if it adopts the greedy approach although it gets a little worse results than the ILP-based approach. Note that runtimes of prototyping, cell placement, and signal routing are excluded in the table for clarity.

Table 3 shows the comparison results of different design flows when a 3D IC has four tiers, and the impreovement by PL-MF flow becomes more obvious as the number of tiers increases. Similar to the last experiment, PF-flow requires the least number of TSVs. But its routing overflow is significantly worse than the PL and PL-MF flows. Besides, PL-MF flow gets better results than PL flow no matter in number of TSVs and routing overflow. Moreover, because macros are legalized after a circuit is partitioned, PF and PL fail to obtain a legal placement for some circuit, which is denoted by NA in the table. The normalized value of the column with NA is computed according to the available values and is marked with the symbol "*".

Fig. 8 (a) and (b) show the runtime breakdowns while the ILP-based approach and the greedy algorithm are applied in our flow. The stages of PL-MF flow (Fig. 7 (c)) are denoted by PP, MP, TAC, TSVP, and CP&R in order. The runtime of the 3D macro placement stage can be greatly reduced when the greedy algorithm is applied.

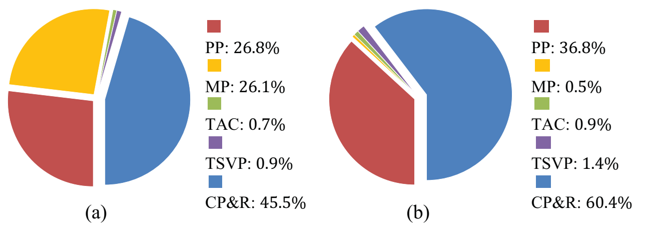

Figure 8: Runtime breakdowns of PL-MF flow while different approaches are used to to check feasibility of K-POCS. (a) ILP-based algorithm. (b) Greedy algorithm.

Fig. 9 shows the resulting macro placement of Cir.2 in a 3-tier 3D IC. Fig. 9 (a) shows the distribution of macros in a projection plane, and the corresponding 3-POCS is shown in Fig. 9 (b). The placement of macros in each tier after tier assignment is shown in Fig. 9 (c). Fig. 9 demonstrates that our method can consider preplaced macros before the tier assignment which avoids overlaps between movable macros and preplaced macros after tier assignment.

Fig. 10 shows the congestion maps of layouts in a 3-tier 3D IC for Cir.2 which are implemented by PF, PL, and PL-MF flows, respectively. PF and PL flows have severe routing congestion in some placement regions (please see tier 2 in Fig. 10 (a) and tier 3 in Fig. 10 (b)) since they may assign more macros with stronger connectivity into the same tier. On the contrary, the PL-MF flow has small routing congestion as shown in Fig. 10 (c) because it will

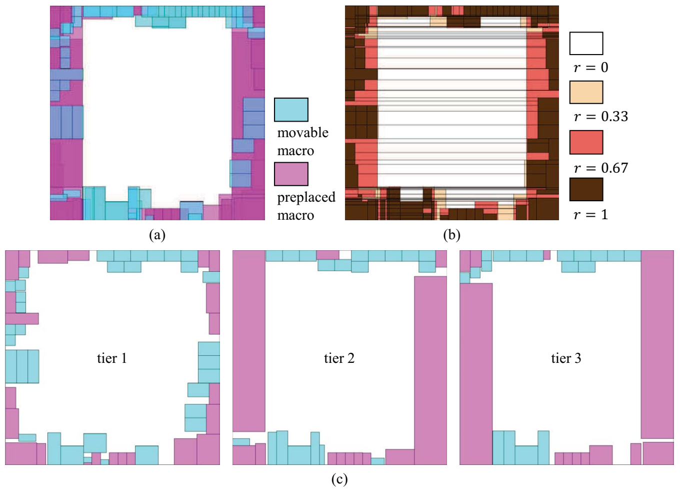

Figure 9: Resulting macro placement of Cir.2 in a 3-tier 3D IC. (a) Macro placement in a projection 2D plane. (b) The corresponding 3-POCS of (a). (c) Resulting macro placement in three tiers after tier assignment.

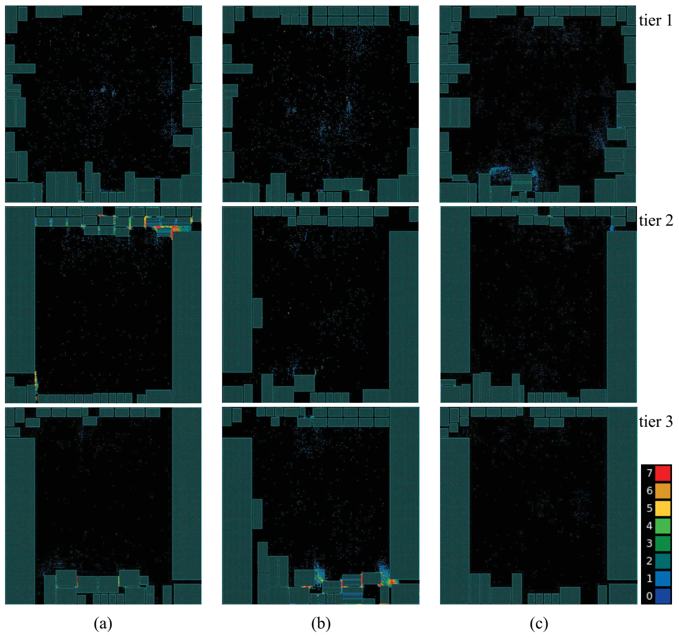

Figure 10: Congestion maps of layouts for Cir.2 in a 3-tiers 3D IC by (a) PF flow, (b) PL flow, and (c) PL-MF flow.

consider the remaining space during 3D macro placement before tier assignment. Although it may generate more TSVs between tiers, macros can be distributed over multiple tiers more evenly which effectively reduces routing congestion in a tier caused by large connection between macros and standard cells.

7 CONCLUSION

This paper has proposed a new PL-MF flow to handle blockage-aware 3D placement for a mixed-sized design. We have introduced a 3D macro placement algorithm, which can keep a better prototyping result while considering preplaced macros in a 3D IC. The experimental results have shown that PL-MF flow can get better results than other design flows.

REFERENCES

[1] Synopsys Inc. https://www.synopsys.com/company.html.
[2] IBM Inc. https://www.ibm.com/analytics/cplex-optimizer.
[3] Himax Technologies, Inc. https://www.himax.com.tw/zh/company/about-himax/.
[4] L. Bamberg, A. Garcia-Ortiz, L. Zhu, S. Pentapati, D. E. Shim, and S. K. Lim, "Macro-3D: A Physical Design Methodology for Face-to-face-stacked Heterogeneous 3D ICs," in Proc. of DATE, pp. 37-42, Mar. 2020.
[5] C.-H. Chang, Y.-W. Chang, and T.-C. Cheng, "A Novel Damped-wave Framework for Macro Placement," in Proc. of ICCAD, pp. 504-511, 2017.
[6] K. Chang, S. Sinha, B. Cline, R. Southerland, M. Doherty, G. Yeric, and S. K. Lim, "Cascade2D: A Design-aware Partitioning Approach to Monolithic 3D IC with 2D Commercial Tools," in Proc. of ICCAD, pp. 1-8, 2016.
[7] Y.-F. Chen, C.-C. Huang, C.-H. Chiou, Y.-W. Chang, and C.-J. Wang, "Routability-driven Blockage-aware Macro Placement," in Proc. of DAC, pp. 1-6, 2014.
[8] T.-C. Chen, P.-H. Yuh, Y.-W. Chang, F.-J. Huang, and T.-Y. Liu, "MP-trees: A Packing-based Macro Placement Algorithm for Modern Mixed-size Designs," IEEE TCAD, vol. 27, no. 9, pp. 1621-1634, 2008.
[9] C.-H Chiou, C.-H Chang, S.-T Chen, and Y.-W Chang, "Circular-contour-based Obstacle-aware Macro Placement," in Proc. of ASP-DAC, pp. 172-177, 2016.
[10] J. Cong and G. Luo, "An Analytical Placer for Mixed-size 3D Placement," in Proc. of ISPD, pp. 61-66, 2010.
[11] G. Karypis and V. Kumar, "hMETIS, a hypergraph partitioning package version 1.5.3," http://glaros.dtc.umn.edu/gkhome/metis/hmetis/download.
[12] B. W. Ku, K. Chang, and S. K. Lim, "Compact-2D: A Physical Design Methodology to Build Commercial-Quality Face-to-face-bonded 3D ICs," in Proc. of ISPD, pp. 90-97, 2018.
[13] J.-M. Lin, S.-T. Li, and Y.-T. Wang, "Routability-driven Mixed-size Placement Prototyping Approach Considering Design Hierarchy and Indirect Connectivity between Macros," in Proc. of DAC, pp. 1-6, Jun. 2019.
[14] J.-M. Lin, Y.-L. Deng, Y.-C. Yang, J.-J. Chen and Y.-C. Chen, "A Novel Macro Placement Approach based on Simulated Evolution Algorithm," in Proc. of ICCAD, pp. 1-7, Nov. 2019.
[15] J.-M. Lin, Y.-L. Deng, Y.-C. Yang, and J.-J. Chen, P.-C. Lu, "Dataflow-aware Macro Placement based on Simulated Evolution Algorithm for Mixed-Size Designs," IEEE TVLS, vol. 29, No. 5, pp. 973-984, May. 2021.
[16] J. Lu, H. Zhuang, I. Kang, P. Chen, and C.-K. Cheng, "ePlace-3D: Electrostatics Based Placement for 3D-ICs," in Proc. of ISPD, pp. 11-18, 2016.
[17] G. Luo, Y. Shi, and J. Cong, "An Analytical Placement Framework for 3-D ICs and Its Extension on Thermal Awareness," IEEE TCAD, vol. 32, no. 4, pp. 510-523, 2013.
[18] J. K. Ousterhout, "Corner Stitching: A Data-structuring Technique for VLSI Layout Tools," IEEE TCAD, vol. 3, no. 1, pp. 87-100, 1984.
[19] S. Panth, K. Samadi, Y. Du and S. K. Lim, "Shrunk-2D: A Physical Design Methodology to Build Commercial-Quality Monolithic 3D ICs," IEEE TCAD, vol. 36, no. 10, pp. 1716-1724, 2017.