Practical Mixed-Cell-Height Legalization Considering Vertical Cell Abutment Constraint*

Teng-Ping Huang
M11007417@mail.ntust.edu.tw
National Taiwan University of Science and Technology, Taipei, Taiwan

Shao-Yun Fang
syfang@mail.ntust.edu.tw
National Taiwan University of Science and Technology, Taipei, Taiwan

ABSTRACT

Propelled by aggressive technology scaling, adopting mixed-cell-height design in VLSI circuits has made conventional single row-based cell legalization techniques obsolete. Furthermore, the vertical abutment constraint (VAC) among cells on consecutive rows emerges as an advanced design requirement, which has rarely been considered because the power/ground rails were sufficiently tall in conventional process nodes to isolate cells on different rows. Although there have been a number of studies on mixed-cell-height legalization, most of them cannot be trivially extended to well-tackle the general VAC due to the analytical optimization scheme. To address these issues, this work proposes the first mixed-cell-height legalization algorithm that addresses the general inter-row cell abutment constraint (i.e., VAC). The experimental results show that the proposed algorithm outperforms previous mixed-cell-height legalization works, even in the absence of the VAC. Upon applying the VAC, our algorithm offers superior performance and delivers promising results.

CCS CONCEPTS

• **Hardware** → **Placement**.

KEYWORDS

Mixed-cell-height legalization, vertical cell abutment constraint

ACM Reference Format:
Teng-Ping Huang and Shao-Yun Fang. 2024. Practical Mixed-Cell-Height Legalization Considering Vertical Cell Abutment Constraint. In *Proceedings of the 2024 International Symposium on Physical Design (ISPD '24), March 12–15, 2024, Taipei, Taiwan.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3626184.3633317

1 INTRODUCTION

In modern VLSI standard cell placement, two significant issues arise with aggressive technology scaling. Firstly, the use of mixed-cell-height libraries is becoming widespread to meet the demands of circuit designs in limited die spaces. Unlike conventional single-cell-height designs, where cells only interact with others in the same row, cells in mixed-cell-height designs interact with each other in multiple rows, leading to a ripple effect on the placement of multiple rows for any slight refinement on the position of a multi-row-height cell. Secondly, complicated design constraints are being introduced. For example, the edge spacing constraint specifies the required spacing between two cell edges (left/right cell boundaries), and the power/ground rail alignment constraint asks each cell must be placed correctly to ensure that its power and ground pins connect to appropriate power rails.

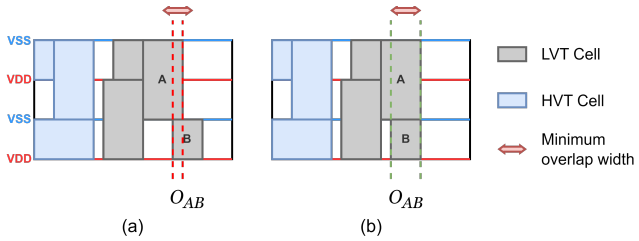

Figure 1: An example of inter-row MIA width rule. (a) An inter-row MIA width violation exists between Cells A and B. (b) The violation is resolved by shifting B such that the overlap length between them is no less than the value specified by the rule.

In addition, vertical abutment constraints (VACs) emerge as a sophisticated design requirement that prohibits the placement of two vertically adjacent cells in a specific relative position. These inter-row VACs have rarely been considered because the power/ground rails were sufficiently tall in conventional process nodes to isolate cells on different rows. One of the process-related design rules causing a VAC is due to minimum implant area (MIA). Fig. 1 illustrates an example, where the relative position of Cells A and B in Fig. 1(a) incurs an inter-row violation, because the overlap length O_{AB} between them is less than the MIA width rule. In contrast, the placement in Fig. 1(b) complies with the MIA rule. Another VAC may be induced by applying triple patterning lithography (TPL) to the fabrication of the middle-of-line (MOL) layers [4]. Fig. 2(c) gives the conflict graph of the MOL features in Fig. 2(a), which is formed due to the placement of the two cells in adjacent rows. Since a K_4 structure is generated in the conflict graph, the MOL features are not manufacturable with TPL. Figs. 2(b) and (d) show a violation-free placement and a K_4-free conflict graph produced by shifting the upper cell to the left by a site. The above examples demonstrate that, due to VACs, it is not feasible to place two vertically adjacent cells in certain relative positions.

The mixed-cell-height placement legalization problem has been extensively studied in recent years, resulting in many publications on the subject. The existing methods can be roughly classified into three categories. The first group of works solves the mixed-cell-height legalization problem by converting a library with mixed cell heights into one with a uniform cell height and reducing the problem into conventional single-cell-height legalization with additional processing [8, 9]. The idea used in these approaches is

*This work was partially supported by TSMC, Synopsys, and NSTC of Taiwan under Grant No's NSTC 110-2223-E-011-002-MY3.

ISPD '24, March 12–15, 2024, Taipei, Taiwan

ACM ISBN 979-8-4007-0417-8/24/03…$15.00
https://doi.org/10.1145/3626184.3633317

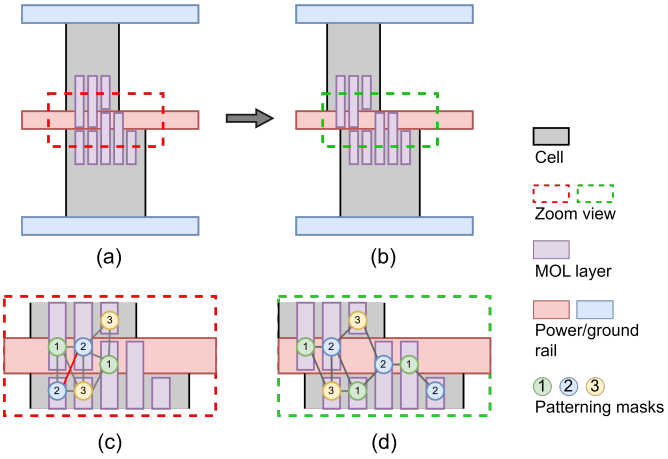

Figure 2: An example of inter-row conflict on MOL layers with TPL. (a) Two vertically adjacent cells with their MOL features. (b) The relative position of the two cells is changed by shifting the upper cell. (c) A zoom-in view of (a), where a K_4 exists in the conflict graph, causing the features unmanufacturable in TPL. (d) A zoom-in view of (b), where the K_4 is resolved, and the features are decomposable with three colors.

relatively straightforward and easy to implement. However, treating multi-row-height cells and single-row-height cells equivalently results in the loss of important circuit design characteristics, significant reduction in the available solution space, and thus severe degradation in solution quality. The second group of studies focuses on minimizing horizontal cell displacements based on a fixed row assignment result and/or horizontal cell orders, which are greedily determined by the given global placement [10, 11]. Aligning cells to the closest rows and fixing horizontal cell orders according to global placement were usually adopted to honor the global placement result as much as possible. However, as pointed out in [10], such a strategy may result in many dead spaces and cause cells to shift further from their positions in global placement, ultimately leading to a legalized result that is greatly different from the original placement. The last group of works is relatively flexible in determining the placement of each cell being legalized. Chow et al. proposed a multi-row local legalization (MLL) algorithm in which cells are placed sequentially [13]. Li et al. further improved upon the algorithm by introducing additional steps, but the underlying concepts remain similar [7]. However, the window-based optimization scheme greatly restricts the solution space since it only adjusts the placement within a window around a target cell when legalizing it. [3] proposes an efficient linear-time mixed-cell-height legalization approach, which achieves a better trade-off between the total cell displacement and the maximum cell displacement among all state-of-the-art works, to the best of our knowledge. Nevertheless, this work fixes the row assignment result and the horizontal cell orders derived from [7], making the principle of [3] similar to those in the second group. More importantly, most of the above existing legalization methods cannot be trivially extended to well-tackle the general VAC due to the analytical optimization scheme.

There are only few works considering inter-row constraints. [5] and [6] address the MIA-related rules including the inter-row MIA width rule shown in Fig. 1. However, the inter-row MIA width rule is actually different from a general VAC, because an inter-row MIA width violation can be resolved by increasing the overlap width between the two vertically adjacent cells or decreasing the overlap width to zero. As a result, [6] models inter-row MIA width rule as linear inequalities that can be integrated into their MMSIM-based approach. For example, to ensure O_{AB} is not less than the minimum overlap width, ω, a linear inequality can be designed as "x_A - x_B $\geq$ ω - W_A", where x and W are respectively the left x-coordinate and the width of a cell. In contrast, a general VAC restricts a cell not to be placed at the specific sites relative to another cell on the adjacent row. Therefore, the feasible relative placement locations of two cells are discrete instead of piecewise linear, which can hardly be modeled as linear inequalities and embedded into MMSIM-based approaches. Lin et al. [4] address the inter-row conflicts of MOL features by assuming TPL is adopted. The design constraints considered in [4] can be regarded as a general VAC because it prohibits a cell from being placed at specific sites to avoid inter-row violations. However, [4] focuses on single-cell-height detailed placement, which can be tackled row by row and is much easier than considering VACs in mixed-cell-height legalization.

This paper presents a mixed-cell-height cell legalization approach considering general VACs as well as other common constraints. The major contributions achieved by this work are summarized as follows:

- To the best of our knowledge, this is the first work of mixed-cell-height cell legalization considering general VACs.
- The proposed legalization algorithm does not modify cell shapes and is more flexible in assigning the row and determining the horizontal order for each cell. In addition, the legalization of each cell is not restricted to a window region. In this way, the proposed algorithm is able to well-tackle VAC constraints, explore a larger solution space, and generate satisfactory results.
- The proposed approach is also designed to be relatively practical and highly flexible such that it could be able to tackle any general intra-row and inter-row constraints that may arise in the future.
- Experimental results using the benchmarks in the IC/CAD 2017 CAD Contest [1] demonstrate that our algorithm achieves better cell displacement and wirelength results than state-of-the-art works even when not considering the VACs. Moreover, when considering the VACs, our algorithm incurs only 2%, 1%, and 0.5% overheads separately in average displacement, maximum displacement, and wirelength without any violations, which greatly outperforms a post-refinement approach.

The rest of the paper is organized as follows: Section 2 provides the problem definition and some preliminaries of this work. Section 3 details the proposed algorithm flow and critical steps. The experimental results are presented in Section 4. Finally, Section 5 concludes the paper.

2 PRELIMINARIES

In this section, the problem that this work aims to resolve is defined in Section 2.1. After that, the VAC is detailed in Section 2.2, and the data structure used in the proposed algorithm is introduced in Section 2.3.

2.1 Problem Definition

The objective of this work is to optimize a mixed-cell-height legalization problem that additionally considers the VAC for advanced

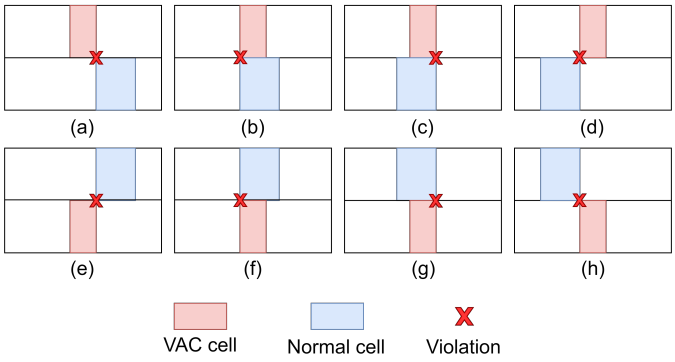

Figure 3: The VAC used in the paper and our experiments. (a)-(h) are the eight forbidden relative positions for each VAC cell.

process technologies. The overview of the problem, including the constraints and objectives, is provided below.

Given a global placement of a mixed-cell-height design, the hard constraints below have to be strictly followed:

- Power/ground rails alignment constraint [1];
- Edge spacing constraint [1];
- Vertical abutment constraint (VAC);
- Cells must be placed on manufacturing sites;
- Cells must not overlap with each other or with fixed macros.

The objective of our algorithm is to satisfy the above constraints while minimizing:

- The average cell displacement;
- The maximum cell displacement.

2.2 Vertical Abutment Constraint (VAC)

In this subsection, a general VAC, which is not addressed in [1] and any other works on mixed-cell-height legalization, is introduced. A general VAC prohibits two vertically adjacent cells to be placed in a specific relative position. To clearly present the proposed algorithm with illustrative examples, we design a VAC by using a similar approach that [1] uses to design the edge spacing constraint. A specific type of cells is first selected as the vertical-abutment-constrained cells (VAC cells), and any cell regardless of its type cannot be placed in the row directly above or below a VAC cell with the eight relative positions shown in Figs. 3(a)–(h). Specifically, each corner of a VAC cell cannot touch the corner of any other cell, as the red crosses indicate. Without loss of generality, the eight forbidden relative positions of VAC cells in Fig. 3 are also applied in our experiments, under the circumstance that the benchmarks in [1] do not contain any information relevant to inter-row constraints. This also makes our experimental results easier to be re-produced and compared with others. Note that the forbidden relative positions can always be modified to model other VACs, and the modification will not introduce any issues to the proposed mixed-cell-height legalization algorithm.

2.3 Data Structure

To detect and avoid overlaps among cells when performing the proposed algorithm flow, a directed acyclic graph (DAG) is used to represent the relative position of cells in a current placement. Fig. 4 shows the DAG of an example placement, where each node represents a cell, and each edge between two nodes represents the

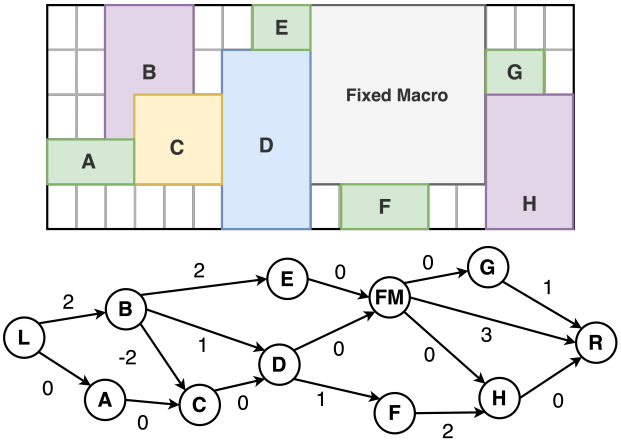

Figure 4: An example of a DAG that used to represent the relative position of each cell in the above partial placement.

distance between the two cells. In addition to cell nodes, there are two nodes respectively representing the left boundary (L) and the right boundary (R) of the placement region. An edge from node a to node b exists only if cell b is immediately to the right of cell a in some row. If two nodes overlap with each other, the weight of the edge between them is negative; if two nodes have empty space between them, the weight of the edge is positive. This data structure enables the algorithm to quickly identify overlaps among cells when shifting them, and to determine how far to shift cells to resolve overlaps and technology violations.

Two important points should be noted regarding the DAG:

(1) Although Fig. 4 shows an illegal placement example, our algorithm only inserts a cell node into the DAG when the cell has been legalized. In other words, the DAG is incrementally constructed node by node as the algorithm legalizes the placement cell by cell. This also implies that the algorithm does not fix the horizontal cell orders and the row assignments of cells from global placement, but it only fixes them once the cells have been legalized.

(2) While [3] also employs a DAG to represent the relative position of cells, our approach differs significantly from theirs. [3] starts by fixing the horizontal cell orders and the row assignment result and then constructs a complete DAG to record the initial positions of all cells. They proceed to decide the position of every cell at a time by traversing the entire DAG. In contrast, our algorithm incrementally constructs the DAG node by node while legalizing the placement cell by cell. Moreover, we only use the DAG to determine cell shifts when violations occur, instead of determining every cell position by traversing the DAG.

3 THE PROPOSED ALGORITHM

This section introduces the proposed vertical abutment-aware mixed-cell-height legalization approach. The overall flow is first presented in Section 3.1. After that, the critical steps and techniques in the flow are separately detailed in Sections 3.2–3.4. Finally, an acceleration method is introduced in Section 3.5.

3.1 Overall Flow

The overall flow of the proposed legalizer is illustrated in Fig. 5. The algorithm takes the optimized global placement result and the

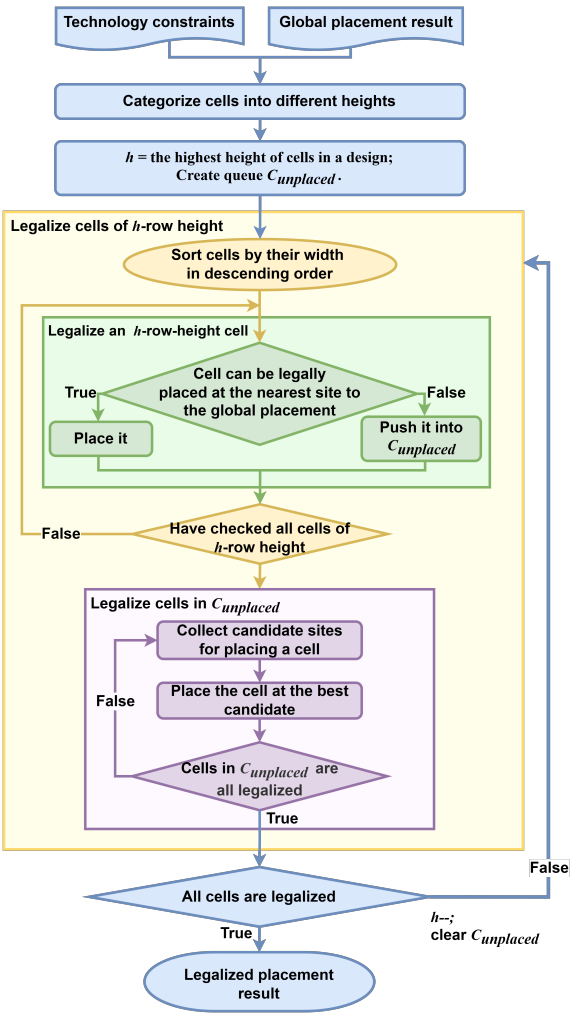

Figure 5: The overall flow of the proposed algorithm.

technology constraints as inputs. It first categorizes the cells into different groups based on their heights and proceeds to legalize the cells in the group of largest cell height first. The algorithm moves on to the next group with lower cell height only when all cells in the current group have been legalized.

The yellow window in Fig. 5 shows the flow of legalizing a group of cells of a specific cell height. Initially, the algorithm sorts the cells in the group by their widths in descending order. In the green window, it sequentially checks if a cell can be placed at the nearest site to its global position. If a cell can be placed at the site without violating any constraints, the algorithm places it directly at the site. If not, the cell is pushed into a queue, which is $C_{unplaced}$. After checking each cell in the group and placing as many cells as possible at their nearest sites, the algorithm begins to legalize the cells in $C_{unplaced}$. The flow is illustrated in the purple window in Fig. 5. To legalize a cell in $C_{unplaced}$, the algorithm first collects a set of candidate sites for placing the cell using the methods described in Section 3.3. It then selects the best candidate site from the collected candidates to place the cell, as explained in Section 3.4. The procedure of the purple window is repeated until all cells in

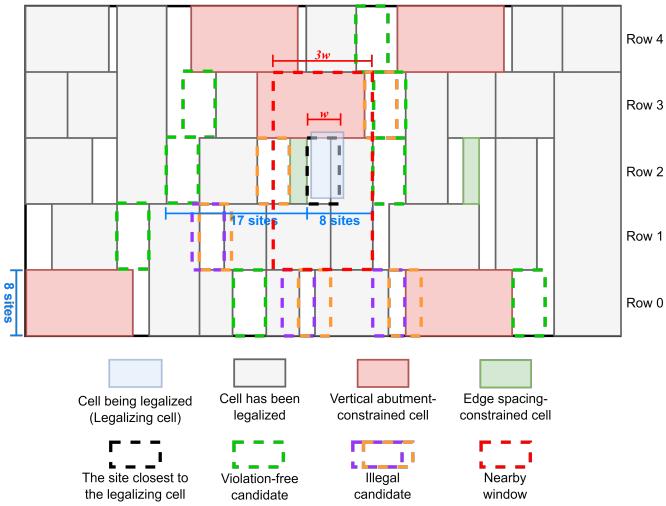

Figure 6: An example of collecting candidate sites for a cell being legalized (legalizing cell). The procedure to collect candidate sites is detailed in Section 3.3.

$C_{unplaced}$ are legalized. Finally, the entire process of the algorithm is terminated and the legalized placement result is derived once all cells in the last group have been legalized, indicating that all cells in the design have been legalized.

The crucial part of the algorithm is the purple window of the flow, and the details will be introduced in the following subsections.

3.2 Sequential Order

The proposed algorithm adopts a unique ordering approach to cell legalization. The approach is different from some conventional methods that prioritize sorting only by cell area. Sorting by area assumes that it may be difficult to find enough space to place a cell with a large area in later stages of legalization. However, our algorithm allows for the shifting of previously legalized cells, making it crucial to minimize the impact on previously legalized placement while shifting them for placing a new cell. Therefore, the proposed algorithm prioritizes placing cells with greater heights over cells with less heights, even if the former is smaller in area than the latter. This is because a cell with a greater height can potentially impact a larger number of rows at a time, leading to a greater influence on previously legalized placement. The concept of sorting cells within each group by their widths is similar to conventional methods that aim to reduce challenges in later stages of legalization. However, our approach of prioritizing higher cells first ultimately helps to reduce the influence on previously placed cells and even improves the runtime.

3.3 Candidate Sites Collection

In the following, the cell being legalized is referred to as the *legalizing cell*, and those that have been legalized and placed before the legalizing cell are defined as *legalized cells*. To demonstrate the approach to collect candidate sites for the legalizing cell, Fig. 6 is used as an example. The approach is composed of three steps: violation-free candidates collection, illegal candidates collection, and nearby window collection. Each dotted rectangle with a color in Fig. 6 is a candidate found in a step. Specifically, the green candidates are found in the first step, the purple and orange candidates

are found in the second step, and the red candidates are found in the last step.

Violation-free candidate collection is the first step to be conducted. It first searches for violation-free candidates in the same row as the closest site to the legalizing cell, and each of them can directly accommodate the legalizing cell without any violations. The determination of the closest site is based on the global placement position and the constraint of power/ground rails alignment. This step finds the first (closest) violation-free candidates (dotted green rectangles in Row 2 of Fig. 6) on both the left side and right side of the closest site. It then compares the horizontal displacements of placing the legalizing cell at each candidate, and sets the larger horizontal displacement as the maximum allowed vertical displacement for the legalizing cell (17 sites in Fig. 6). Similarly, the subroutine finds the closest violation-free candidate on both the left and right of the closest site in each row within the maximum allowed vertical displacement for the legalizing cell (dotted green rectangles in Rows 0, 1, 3, and 4 of Fig. 6).

Illegal candidates collection is the second step, and it collects candidates (dotted purple and orange rectangles in Fig. 6) that require the legalized cells to shift to accommodate the legalizing cell to expand the solution space. This step collects every empty space between the closest site to the legalizing cell and each violation-free candidate found in the previous step. For each empty space, two candidates are selected. The orange candidate aligns with the left edge of the empty space, which needs to shift the cells on its right side. The purple candidate aligns with the right edge of the empty space, which needs to shift the cells on its left side. Although other sites in each empty space can also be selected as candidates and other shift operations may be adopted, the runtime may dramatically increase due to solution space explosion. Empirically, the currently adopted operations can lead to a sufficiently good trade-off between runtime and solution quality.

The last step is nearby window collection. This step sets a window (dotted red rectangle in Fig. 6) with a width three times that of the legalizing cell and a height three times that of the row height, and centered on the closest site to the legalizing cell. Although placing the legalizing cell on each site in the window may have a greater impact on the placement, it helps to reduce the maximum displacement in the design. This reason may not be apparent from the example used because many empty spaces are near the legalizing cell. However, in later legalization stages, the empty spaces may be far from the legalizing cell due to dense placement, and placing the legalizing cell at the candidate sites in the nearby window can help to reduce the maximum displacement.

3.4 Candidate Sites Evaluation

The evaluation of placing the legalizing cell at a candidate site is performed as follows. If placing the legalizing cell at the candidate site violates any constraint, the subroutine solves the violations using the method introduced in the later subsection "Shift of Legalized Cells". The cost of placing the legalizing cell at the candidate site and shifting the legalized cells is then calculated using the function introduced in the subsection "Cost Calculation". After evaluating each candidate site for the legalizing cell, the candidate site with the minimum cost is selected to place the legalizing cell.

3.4.1 Shift of Legalized Cells. Placing the legalizing cell at an illegal candidate site may result in three types of violations, namely, cell overlap, edge spacing violation, and vertical abutment violation. Note that all illegal candidate sites found by previous subroutines comply with the power/ground rails alignment constraint.

The overlap and edge spacing violations can be easily resolved using the DAG presented in Section 2.3. Initially, the algorithm

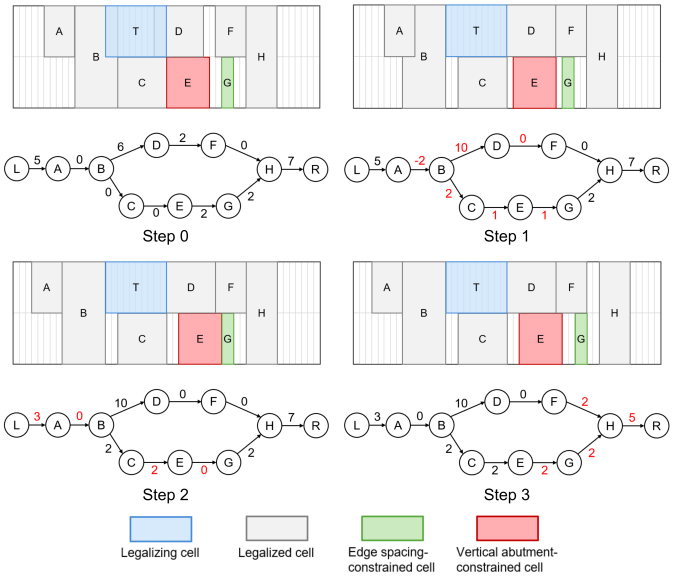

Figure 7: An example of shifting legalized cells to resolve violations resulting from placing the legalizing cell.

shifts the legalized cells that would cause an overlap or an edge spacing violation when placing the legalizing cell. The algorithm determines whether each legalized cell should be shifted to the left or right by considering which direction would result in less displacement to the legalized cell. If the left or right shift of a legalized cell causes additional overlaps, the algorithm shifts its predecessors or successors in the DAG one by one, separately, until there is no overlap. For example, in the step 0 of Fig. 7, placing the legalizing cell (Cell T, the blue cell) causes the overlaps with Cells B and D. To remove the overlaps, suppose moving Cell B to the left and cell D to the right in the step 1 of Fig. 7 causes the minimum displacements for both the cells. Once the algorithm shifts a legalized cell, it will update the weights of the edges connected to the nodes representing the legalized cells. The weights of the updated edges are marked in red in Fig. 7. The overlap resulted from the shift of cell B can then be detected by the DAG as a negative edge weight (-2). The algorithm thus keeps shifting the predecessor of cell B to the left and updating the edge weights at the same time, until there is no overlap and any negative edge weight. As for encountering an edge spacing violation, the algorithm will directly shift a cell by more sites to not only avoid the overlap but also avoid the edge spacing violation, as Cell G from the step 2 to the step 3 in Fig. 7 (the edge spacing is set as 2 sites in this example).

Solving vertical abutment violations (VAV) is more complicated than handling overlaps or edge spacing violations. The algorithm cannot easily determine the shift direction of legalized cells through the DAG to resolve VAV because the DAG only records the horizontal relation among cells. Thus, we categorize the causes of VAV into two situations and introduce how to solve them. The first situation arises from placing the legalizing cell at a site. Since the algorithm selects the site to place the legalizing cell, it must shift any cells, except the legalizing cell, to resolve the violation. If the VAV arise from the right corners of the legalizing cell, the algorithm intuitively shifts the legalized cell that causes the VAV to the right by one site to resolve the violation, and vice versa. An example is shown in the step 0 to 1 of Fig. 7, where the red cell is shifted to the right by one

site to resolve the VAV between Cells T and E. The second situation occurs when the algorithm shifts legalized cells. If the algorithm shifts a legalized cell and causes a VAV, it only shifts the cell in the same direction by one more site to resolve the violation instead of shifting another cell in another row. Consequently, spreading VAV to more rows and affecting more legalized cells can be avoided. An example is shown in the step 1 to 2 of Fig. 7, where the red cell is shifted to the right by one more site to resolve the VAV between E and F as well as that between E and D. Note that in the step 1 of Fig. 7, the algorithm can either shift Cell D or Cell E to resolve the violation, but it chooses to shift cell E. In this way, VAV spreading to the rows above cell D can be avoided.

3.4.2 Cost Calculation. The cost of placing a legalizing cell at a candidate site can be calculated as follows:

$$cost = \alpha \cdot disp_l + \sum_{i \in G} (disp_i^{after} - disp_i^{before}), \tag{1}$$

$$\alpha = \begin{cases} 8, & disp_l > \text{current maximum displacement}, \\ 1, & disp_l \leq \text{current maximum displacement}, \end{cases} \tag{2}$$

where $disp_l$ is the displacement of the legalizing cell from its global position, G is the set of legalized cells shifted due to placing the legalizing cell at the candidate site, and $disp_i^{before}/disp_i^{after}$ is the displacement of a legalized cell from its global position before/after the shift resulting from placing the legalizing cell.

To address the maximum displacement in the entire design and consider the impact on the legalized cells, we divide the cost function of a candidate site into two parts: the displacement of the legalizing cell and the incremental displacement of the legalized cells due to placing the legalizing cell at the candidate site. The displacement of the legalizing cell is assigned a separate multiplier, denoted as α. When placing the legalizing cell at the candidate site results in a new maximum displacement, α is set to eight, indicating that the algorithm will heavily penalize such a situation. Otherwise, α is set to one, implying that the displacement of each shifted cell is treated equally. Consequently, the algorithm favors candidate sites that do not cause a new maximum displacement to place the legalizing cell, even if this would result in a larger incremental displacement of the legalized cells.

It is worth noting that in some cases, shifting a legalized cell may bring the cell closer to its global position. Our cost function can recognize this as a benefit to the candidate site because ($disp_i^{after} - disp_i^{before}$) is negative in this situation.

3.5 Acceleration for the Algorithm

As mentioned in Section 3.3, the proposed algorithm collects violation-free candidates prior to illegal candidates. Since each violation-free candidate can directly accommodate the legalizing cell without the need of shifting legalized cells, the cost of placing the legalizing cell at a violation-free candidate is solely determined by the displacement of the legalizing cell. Therefore, the minimum displacement of placing the legalizing cell at a violation-free candidate can then become the cost upper bound for the latter steps of collecting illegal candidates.

Placing the legalizing cell at an illegal candidate site needs to shift legalized cells. Thus, according to Equation (1), the cost of evaluating an illegal candidate is composed of the displacement of the legalizing cell and the total displacement of the shifted legalized cells. During the cost evaluation, the algorithm shifts legalized cells one by one and accumulates the displacement of each shifted legalized cell. If the displacement of the legalizing cell plus the accumulated displacement of the legalized cells exceeds the cost upper bound at any moment, the algorithm breaks the procedure and moves on to evaluate another illegal candidate. This approach thus helps the algorithm to avoid redundant calculations.

Table 1: Benchmarks information.

Benchmark	#S. Cell	#D. Cell	#T. Cell	#Q. Cell	Density
des_perf_1	112,644	0	0	0	91%
des_perf_a_md1	103,589	4,699	0	0	55%
des_perf_a_md2	105,030	1,086	1,086	1,086	56%
des_perf_b_md1	106,782	5,862	0	0	55%
des_perf_b_md2	101,908	6,781	2,260	1,695	65%
edit_dist_1_md1	118,005	7,994	2,664	1,998	67%
edit_dist_a_md2	115,066	7,799	2,599	1,949	59%
edit_dist_a_md3	119,616	2,599	2,599	2,599	57%
fft_2_md2	28,930	2,117	705	529	83%
fft_a_md2	27,431	2,018	672	504	32%
fft_a_md3	28,609	672	672	672	31%
pci_bridge32_a_md1	26,680	1,792	597	448	50%
pci_bridge32_a_md2	25,239	2,090	1,194	994	58%
pci_bridge32_b_md1	26,134	1,756	585	439	27%
pci_bridge32_b_md2	28,038	292	292	292	18%
pci_bridge32_b_md3	27,452	292	585	585	22%

4 EXPERIMENTAL RESULTS

Our mixed-cell-height legalization algorithm is implemented in C++ programming language and executed on a personal computer with an Intel Core i5-10400 CPU running at 2.90 GHz and 16GB RAM. The operating system used is WSL (Windows Subsystem for Linux). For our experiments, we use the benchmarks from IC/CAD-2017 CAD Contest in Multi-Deck Standard Cell Legalization and Benchmarks [1], which is widely considered as a representative benchmark set in the mixed-cell-height legalization problem. The details of the benchmarks are presented in Table 1, where "#S. Cell", "#D. Cell", "#T. Cell", and "#Q. Cell" separately give the numbers of single-, double-, triple-, and quadruple-row-height cells. In addition to the power/ground rails alignment constraint and the edge spacing constraints presented in [1], the VAC introduced in Section 2.2 is also considered as a hard constraint. Two experiments were conducted to show the excellent performance of the proposed legalization approach, both when considering the VAC and when not considering it.

4.1 Results w/o VAC Consideration

In the first experiment, the proposed legalization approach is compared with the first place of IC/CAD-2017 CAD Contest and [3]. [3] is considered as the state-of-the-art mixed-cell-height legalization work that achieves the best trade-off between the average displacement and the maximum displacement. The experimental results reported in [3] demonstrate that compared to [7], although the method of [3] suffers from 1.1% larger average displacement, it greatly reduces the maximum displacement by 11.1%. The four constraints excluding the VAC listed in Section 2.1 are considered in the first experiment, which are the same as [3] for fair comparison. Note that in addition to the same hard constraints, the first place of IC/CAD-2017 CAD Contest also addresses additional soft constraints in the contest.

The experimental results are presented in Table 2, where "Avg. Disp.", "Max. Disp.", and "HPWL" separately indicate the average displacement, the maximum displacement, and the total half-parameter wirelength of each benchmark. Observing the presented results, our algorithm achieves much lower average displacement

Table 2: The experimental results without considering the VAC.

Benchmark	Avg. Disp. (sites)			Max. Disp. (sites)			HPWL (m)			Runtime (s)		
	1st	[3]	Ours	1st	[3]	Ours	1st	[3]	Ours	1st	[3]	Ours
des_perf_1	7.11	6.81	**6.45**	76.69	**38.49**	67.07	1.30	1.34	**1.29**	12.30	1.80	3.09
des_perf_a_md1	7.53	5.61	**5.52**	625.78	*__607.30__	*__607.30__	2.26	2.23	**2.21**	7.72	1.21	3.65
des_perf_a_md2	7.72	5.50	**5.41**	679.76	480.55	*__403.86__	2.28	2.25	**2.23**	7.76	1.21	3.70
des_perf_b_md1	5.41	4.58	**4.54**	90.47	**30.27**	37.53	2.16	2.16	**2.14**	6.59	1.48	0.99
des_perf_b_md2	6.16	4.97	**4.93**	199.78	**30.62**	32.87	2.19	2.19	**2.17**	6.26	1.74	1.42
edit_dist_1_md1	7.07	**5.45**	5.53	79.17	**52.84**	52.99	4.09	4.09	**4.05**	8.58	2.12	3.75
edit_dist_a_md2	6.19	5.16	**5.14**	**164.00**	**164.00**	168.00	5.17	5.17	**5.15**	7.59	1.33	2.65
edit_dist_a_md3	9.18	7.56	**6.80**	279.54	**233.00**	237.00	5.45	5.46	**5.41**	85.36	2.18	4.13
fft_2_md2	**7.72**	8.49	7.74	66.06	**45.01**	69.88	0.49	0.51	**0.48**	1.60	0.42	3.53
fft_a_md2	5.34	**4.58**	4.59	*__343.48__	*__343.48__	*__343.48__	**1.11**	**1.11**	**1.11**	1.41	0.20	1.95
fft_a_md3	5.04	**4.31**	4.32	*__109.62__	*__109.62__	*__109.62__	0.97	0.97	**0.96**	1.36	0.18	1.82
pci_bridge32_a_md1	6.90	5.30	**5.29**	425.72	**63.76**	69.95	**0.47**	0.48	**0.47**	1.43	0.20	0.42
pci_bridge32_a_md2	8.32	6.89	**6.57**	271.89	*__121.35__	*__121.35__	**0.59**	0.60	**0.59**	2.71	0.25	0.84
pci_bridge32_b_md1	7.83	5.54	**5.47**	876.62	**332.72**	338.01	**0.68**	**0.68**	**0.68**	1.71	0.28	1.10
pci_bridge32_b_md2	6.66	5.20	**5.15**	723.45	452.09	*__429.04__	**0.59**	**0.59**	**0.59**	1.55	0.17	1.00
pci_bridge32_b_md3	8.21	5.68	**5.56**	682.12	476.91	*__398.58__	0.62	0.61	**0.60**	1.86	0.19	1.30
Average	7.02	5.73	**5.56**	355.88	223.88	**217.91**	1.90	1.90	**1.88**	9.74	0.94	2.21
Norm. Avg.	1.26	1.03	**1.00**	1.63	1.03	**1.00**	1.01	1.01	**1.00**	4.41	0.42	1.00

Table 3: The experimental results with VAC consideration.

Benchmark	# VAV	Avg. Disp. (sites)		Max. Disp. (sites)		HPWL (m)		Runtime (s)	
	w/o VAC	w/o VAC	w/ VAC	w/o VAC	w/ VAC	w/o VAC	w/ VAC	w/o VAC	w/ VAC
des_perf_1	13,010	6.45	6.94	67.07	76.40	1.29	1.30	3.09	11.10
des_perf_a_md1	9,340	5.52	5.64	607.30	607.30	2.21	2.22	3.65	17.79
des_perf_a_md2	9,567	5.41	5.54	403.86	403.86	2.23	2.23	3.70	18.01
des_perf_b_md1	7,920	4.54	4.60	37.53	35.69	2.14	2.14	0.99	4.86
des_perf_b_md2	10,190	4.93	5.03	32.87	30.21	2.17	2.17	1.42	7.60
edit_dist_1_md1	8,015	5.53	5.52	52.99	52.99	4.05	4.06	3.75	10.43
edit_dist_a_md2	9,653	5.14	5.21	168.00	168.00	5.15	5.15	2.65	9.44
edit_dist_a_md3	12,677	6.80	7.11	237.00	237.00	5.41	5.42	4.13	17.96
fft_2_md2	4,751	7.74	7.86	69.88	95.09	0.48	0.49	3.53	9.31
fft_a_md2	1,671	4.59	4.62	343.48	343.48	1.11	1.11	1.95	3.77
fft_a_md3	1,686	4.32	4.35	109.62	109.62	0.96	0.96	1.82	3.39
pci_bridge32_a_md1	1,392	5.29	5.33	69.95	72.48	0.47	0.47	0.42	1.59
pci_bridge32_a_md2	2,014	6.57	6.70	121.35	121.35	0.59	0.59	0.84	3.98
pci_bridge32_b_md1	1,145	5.47	5.50	338.01	338.01	0.68	0.68	1.10	3.54
pci_bridge32_b_md2	1,186	5.15	5.18	429.04	429.04	0.59	0.59	1.00	3.31
pci_bridge32_b_md3	1,289	5.56	5.59	398.58	398.58	0.60	0.60	1.30	6.48
Average		5.56	5.67	217.91	219.94	1.88	1.89	2.21	8.29
Norm. Avg.		0.98	1.00	0.99	1.00	1.000	1.002	0.27	1.00

Table 4: The average and maximum displacements derived from our algorithm w/ and w/o VAC consideration.

Benchmark	Displacements between w/ and w/o VAC		
	Avg. (sites)	Max. (sites)	Total (sites)
des_perf_1	3.35	108	377137
des_perf_a_md1	1.44	103	156090
des_perf_a_md2	1.48	122	160613
des_perf_b_md1	0.65	56	72855
des_perf_b_md2	1.04	51	117563
edit_dist_1_md1	1.59	57	207635
edit_dist_a_md2	0.99	70	126529
edit_dist_a_md3	2.7	362	343972
fft_2_md2	4.83	100	156048
fft_a_md2	0.33	29	10039
fft_a_md3	0.33	30	10175
pci_bridge32_a_md1	0.49	76	14568
pci_bridge32_a_md2	1.37	144	40350
pci_bridge32_b_md1	0.39	51	11244
pci_bridge32_b_md2	0.32	56	9206
pci_bridge32_b_md3	0.42	58	12055
Average	1.36	92.06	114129.94

and maximum displacement compared to the first place of IC/CAD-2017 CAD Contest ("1st" in Table 2). Compared with [3], our approach also achieves slightly and averagely better results both in average displacement and maximum displacement, demonstrating the ability of the proposed legalizer to leverage the result from global placement by not fixing horizontal cell orders and a row assignment result. Each data marked by "*" in Table 2 indicates that it achieves the optimal solution, which can be ensured due to the presence of fixed macros. If a cell is placed within a fixed macro region after the global placement, to obtain a legal placement, the cell must be shifted out of the region. If the minimum displacement of shifting the cell out of the region is equivalent to the maximum displacement in the legalized placement, it can be ensured that the maximum displacement is optimal. In addition, the smaller cell displacements also contribute to better wirelength results. Table 2 shows that the proposed method slightly reduces HPWL by 1% compared to the two baseline works.

In terms of runtime, since we are unable to obtain the binaries of the two works, we directly compare our runtimes with theirs recorded in the paper/contest results. It is worth noting that the platform [3] used for their experiments is superior to ours, as their

Table 5: The results of the first place of the IC/CAD-2017 Contest with a post-refinement method to address the VAC.

Benchmark	ICCAD 2017 1st			ICCAD 2017 1st after Solving VAV			
	#VAV	Avg. Disp (sites)	Max. Disp (sites)	Avg. Disp (sites)	Max. Disp (sites)	#COFM	#COC
des_perf_1	12,556	7.11	76.69	16.17	116.76	0	541
des_perf_a_md1	10,418	7.53	625.78	8.30	625.78	7	3
des_perf_a_md2	10,109	7.72	679.76	8.44	682.76	6	9
des_perf_b_md1	8,469	5.41	90.47	5.63	90.12	0	2
des_perf_b_md2	10,804	6.16	199.78	7.02	204.78	0	4
edit_dist_1_md1	8,821	7.07	79.17	7.24	79.17	0	1
edit_dist_a_md2	10,232	6.19	164.00	6.71	164.00	5	18
edit_dist_a_md3	12,536	9.18	279.54	17.38	294.58	71	307
fft_2_md2	4,054	7.72	66.06	36.49	146.13	0	596
fft_a_md2	1,936	5.34	343.48	5.40	343.48	9	2
fft_a_md3	1,918	5.04	109.62	5.10	109.62	5	1
pci_bridge32_a_md1	1,459	6.90	425.72	7.00	425.72	43	14
pci_bridge32_a_md2	1,698	8.32	271.89	10.14	271.89	11	21
pci_bridge32_b_md1	1,176	7.83	876.62	7.90	876.62	2	0
pci_bridge32_b_md2	1,195	6.66	723.45	6.73	723.45	3	0
pci_bridge32_b_md3	1,315	8.21	682.12	8.31	682.12	2	0
Average		7.02	355.88	10.25	364.81		
Norm. Avg.		1.00	1.00	1.46	1.03		

machine is equipped with a ThreadRipper 3970X and 256GB of memory. On the other hand, [1] does not describe the platform they used. Under this circumstance, both our approach and [3] are faster than the first place of the IC/CAD-2017 Contest. In addition, although [3] claims that its algorithm is linear time, the approach uses the fixed horizontal cell orders and the row assignment result derived from [7]. Fixing these two variables can reduce time complexity while also heavily restricting the solution space. Moreover, the DAG-based linear optimization scheme of [3] lacks flexibility to address new constraints in novel technologies such as general VACs. In contrast, our algorithm can easily address other constraints within reasonable runtimes, which will be shown in the following experiment.

4.2 Results w/ VAC Consideration

The second experiment shows the flexibility of our algorithm in handling the VAC. Since this work is the first one addressing the VAC in mixed-cell-height legalization, the experiment focuses on demonstrating that our algorithm can produce a legal placement without any violations and with only small increments in the average and maximum displacements compared to our results in the first experiment. To apply the VAC, the fourth most frequently used cell type in each design is identified as the VAC cell type. Each cell instance of this type cannot be placed with others with the eight forbidden relative positions shown in Fig. 3.

Table 3 and Table 4 present the results of this experiment. They show the changes in each benchmark when our algorithm additionally addresses the VAC. The second column of Table 3 indicates the number of VAVs in the produced placement when our algorithm does not address the VAC. In contrast, there will be no VAV in the produced placement if the VAC is considered in our algorithm flow. The results in Table 3 indicate that our algorithm only respectively suffers from 2% average displacement overhead, 1% maximum displacement overhead, and 0.2% wirelength overhead to resolve all VAVs. The increased runtimes are due to much more cell shifts required to resolve VAVs during candidate evaluation. However, the runtimes still fall within an acceptable range.

Table 3 computes cell displacements based on the given global placement. In contrast, Table 4 calculates cell displacements between two placements derived from our algorithm: one incorporates the VAC consideration and the other does not. Although Table 3 only shows slight displacement overheads when comparing with the global placement, Table 4 shows the significant differences between the placements generated by "w/o VAC" and "w/ VAC," especially in dense designs such as des_perf_1 and fft_2_md2. This suggests that the VAC has a massive impact on the placement. Overall, the results show that our algorithm can resolve all VAVs while guaranteeing a high-quality placement, even if the placement needs to be greatly perturbed to address the VAC.

To show that the VAC cannot be trivially handled in the post-placement stage, we apply a placement refinement procedure to the placements from the first-place of the IC/CAD-2017 Contest. We first identify the VAVs in their placement under the same VAC and then solve them one by one in the layouts from bottom to top and left to right while maintaining the horizontal cell orders and the row assignment result. We adopt the same cell shifting approach presented in Fig. 7 to resolve each VAV. Table 5 shows the refinement results, where "#COFM" and "#COC" respectively represent the numbers of cells overlapped with fixed macros and outside the core. The results in the table show that none of the benchmarks can get a legal placement as some cells are overlapped with the fixed macros or placed outside the core area. Moreover, the average displacements seriously increase after such a post-placement refinement technique, which makes cells greatly deviate from their optimized positions derived from the global placement stage.

5 CONCLUSIONS

This work commences by examining the origins of the VAC and establishing its rule in the placement. A mixed-cell-height legalization framework is then presented that is the first in literature considering the VAC. Although the adopted techniques are simple and intuitive, they make the proposed method highly flexible in handling complicated design rules in advanced nodes such as VACs. In addition, the proposed method does not restrict row assignment and cell ordering when legalizing each cell, resulting in even better results in terms of cell displacement and wirelength compared to state-of-the-art mixed-cell-height legalization studies. More importantly, by developing more sophisticated algorithms for each step within the framework of this paper, the experimental results are certain to make further advancements, which will serve as our future research directions.

REFERENCES

[1] Nima Karimpour Darav, Ismail S. Bustany, Andrew Kennings, and Ravi Mamidi, "ICCAD-2017 CAD Contest in Multi-Deck Standard Cell Legalization and Benchmarks," in *proceedings of International Conference on Computer-Aided Design*, 2017.

[2] Z. Zhu, J. Chen, W. Zhu and Y. -W. Chang, "Mixed-Cell-Height Legalization Considering Technology and Region Constraints," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 39, no. 12, pp. 5128–5141, Dec. 2020.

[3] Chung-Hsien Wu, Wai-Kei Mak, and Chris Chu, "Linear-time Mixed-Cell-Height Legalization for Minimizing Maximum Displacement," in *proceedings of International Symposium on Physical Design*, 2022.

[4] Y. Lin, B. Yu, B. Xu and D. Z. Pan, "Triple patterning aware detailed placement toward zero cross-row middle-of-line conflict," in *proceedings of International Conference on Computer-Aided Design*, 2015.

[5] J. Chen, Y. -W. Chang and Y. -Y. Wu, "Mixed-Cell-Height Detailed Placement Considering Complex Minimum-Implant-Area Constraints," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 40, no. 10, pp. 2128–2141, Oct. 2021.

[6] J. Chen, Z. Lin, Y. Xie, W. Zhu and Y. -W. Chang, "Mixed-Cell-Height Placement With Complex Minimum-Implant-Area Constraints," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 11, pp. 4639–4652, Nov. 2022.

[7] H. Li, W. -K. Chow, G. Chen, B. Yu and E. F. Y. Young, "Pin-Accessible Legalization for Mixed-Cell-Height Circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 1, pp. 143–154, Jan. 2022.

[8] G. Wu and C. Chu, "Detailed Placement Algorithm for VLSI Design With Double-Row Height Standard Cells," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 35, no. 9, pp. 1569–1573, Sept. 2016.

[9] S. A. Dobre, A. B. Kahng and J. Li, "Design Implementation With Noninteger Multiple-Height Cells for Improved Design Quality in Advanced Nodes," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 37, no. 4, pp. 855–868, April 2018.

[10] Chao-Hung Wang, Yen-Yi Wu, Jianli Chen, Yao-Wen Chang, Sy-Yen Kuo, Wenxing Zhu, and Genghua Fan, "An effective legalization algorithm for mixed-cell-height standard cells," in *proceedings of Asia and South Pacific Design Automation Conference*, 2017.

[11] Jianli Chen, Ziran Zhu, Wenxing Zhu, and Yao-Wen Chang, "Toward optimal legalization for mixed-cell-height circuit designs," in *proceedings of Design Automation Conference*, 2017.

[12] Xingquan Li, Jianli Chen, Wenxing Zhu, and Yao-Wen Chang, "Analytical Mixed-Cell-Height Legalization Considering Average and Maximum Movement Minimization," in *proceedings of International Symposium on Physical Design*, 2019.

[13] W.-K. Chow, C.-W. Pui, and E. F.Y. Young, "Legalization algorithm for multiple-row height standard cell design," in *proceedings of Design Automation Conference*, 2016.