Journal of
Micro/Nanolithography,
MEMS, and MOEMS

Nanolithography.SPIEDigitalLibrary.org

Systematic framework for evaluating standard cell middle-of-line robustness for multiple patterning lithography

Xiaoqing Xu
Brian Cline
Greg Yeric
Bei Yu
David Z. Pan

SPIE.

Systematic framework for evaluating standard cell middle-of-line robustness for multiple patterning lithography

Xiaoqing Xu,[a,*] Brian Cline,[b] Greg Yeric,[b] Bei Yu,[c] and David Z. Pan[a]
[a]The University of Texas at Austin, ECE Department, 201 East 24th Street, Stop C8800, Austin, Texas 78712, United States
[b]ARM Inc., 5707 Southwest Parkway #100, Austin, Texas 78735, United States
[c]Chinese University of Hong Kong, CSE Department, Shatin, NT, Hong Kong

Abstract. For robust standard cell design, designers need to improve the intercell compatibility for all combinations of cells and cell placements. Multiple patterning lithography colorability check breaks the locality of traditional rule check, and *N*-wise checks are strongly needed to verify the colorability for layout interactions across cell boundaries. A systematic framework is proposed to evaluate the library-level robustness over multiple patterning lithography from two perspectives, including complete checks on two-row combinations of cells and long-range interactions. With complete checks on two-row combinations of cells, the vertical and horizontal boundary checks are explored to predict illegal cell combinations. For long-range interactions, random benchmarks are generated by cell shifting and tested to evaluate the placement-level efforts needed to reduce the manufacturing complexity from quadruple patterning lithography to triple patterning lithography for the middle-of-line (MOL) layers. Our framework is tested on the MOL layers but can be easily adapted to other critical layers with multiple patterning lithography constraints. © *2016 Society of Photo-Optical Instrumentation Engineers (SPIE)* [DOI: 10.1117/1.JMM.15.2.021202]

Keywords: multiple patterning lithography; middle-of-line; cell compatibility check.

Paper 15089SSP received May 23, 2015; accepted for publication Oct. 19, 2015; published online Feb. 3, 2016.

1 Introduction

Multiple patterning lithography (MPL) has been widely adopted in industry for technology scaling due to the resolution limits of 193 nm lithography tools.[1–4] To enable MPL, restrictive design rules or constraints have been introduced into the back-end design flow. To deal with these complex constraints, a wide range of research efforts have been focusing on standard cell (SC) design,[1,5] placement,[1,6,7] routing,[8–12] layout migration, and decomposition.[13–17] As the foundation of the entire back-end design flow, the SC library plays a significant role in the design closure, including both the final sign-off performance of the design as well as the overall time-to-market. Therefore, huge amounts of efforts are spent on the SC library design and evaluation for each technology node. In particular, SC designers not only need to improve the layout design of each individual cell, but also need to check the layout interactions across the cell boundaries when cells are placed next to each other. Ideally, a robust SC library can be used in any arbitrary placement, which means that all combinations of individual cell placement should be legal. During the library design phase, it is important for the SC designers to quickly track those cells or combinations of cells that penalize the library robustness. Therefore, an efficient SC library evaluation framework is strongly needed so that SC designers can improve the cell layout design accordingly. However, there have been few works related to efficient SC library evaluation, which becomes particularly important when considering MPL design constraints. For SC library design, intercell compatibility is essential for achieving a robust library that can be used in any design implementation, no matter what kind of placement is implemented for that design. Historically, intercell placement compatibility could be guaranteed by pair-wise checks, i.e., checking all two-cell combinations, because the lithographic interactions were small in comparison to the cell sizes and limited to single mask layers.

With the industry extending 193 nm lithography via MPL, the lithographic interactions now include complex layout design constraints that extend beyond two-cell interactions. MPL coloring and decomposition—especially on middle-of-line (MOL) layers, like the contact-to-active (CA) layer,[18] that routinely introduces interactions across cell boundaries—breaks the locality of traditional rule check and makes historical pair-wise cell compatibility check obsolete. Figure 1 illustrates the cell interactions introduced by MPL, where Fig. 1(a) shows a three-cell interaction that requires triple patterning lithography (TPL) to decompose the pattern at the boundary, while Fig. 1(b) illustrates a three-cell interaction example that requires quadruple patterning lithography (QPL). Specifically, for the dimensions in the representative 10/15 nm technologies used in this work, TPL or even QPL is needed on the CA layer, depending on the SC boundary conditions and the strength of color-aware placement tools. In cutting-edge technology nodes like 10 nm and beyond, *N*-wise checks are now strongly needed to verify the MPL colorability for cell interactions across boundaries, as demonstrated in Fig. 1. An *N*-wise check denotes the design rule check (DRC) for a cell combination consisting of *N* cells. This means that what once was

*Address all correspondence to: Xiaoqing Xu, E-mail: xiaoqingxu.austin@utexas.edu

1932-5150/2016/$25.00 © 2016 SPIE

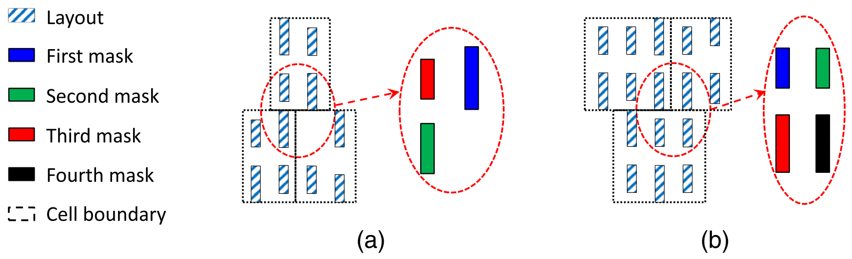


Fig. 1 Three-cell interactions: (a) triple patterning lithography and (b) quadruple patterning lithography.

a tractable problem in pair-wise cell check now becomes an exponential problem in the exhaustive N-wise check. Fortunately, the extremely regular layout structure of layers that interact at the cell boundary leads to large amounts of redundancy among exhaustive intercell compatibility checks. For a typical SC library, it becomes worthwhile to explore and remove the redundancy among these intercell compatibility checks to make N-wise checks more tractable.

For the 10 nm technology, the MOL layers, including CA and contact-to-poly (CB), are essential for the intracell routing,[19,20] improving cell routability and pin accessibility.[21] This work mainly focuses on the library robustness evaluation on the CA layer due to its complex MPL design constraints. Specifically, for the representative 10/15 nm technologies used in this study, the CA layer has small dimensions/pitches that require TPL or even QPL. Compared to other MPL layers, such as Metal-1, stitching is not allowed for the CA layer (which is enforced by a technology's DRCs). The CA layer routinely introduces interactions across rows under MPL constraints, which makes N-wise checks particularly important in the library evaluation stage. In particular, our evaluation framework can be easily adapted to other MPL layers, such as Metal-1.

For library robustness evaluation, the intercell compatibility checks should report all the illegal cell combinations to the designers. Then, designers can improve the library cell layout and reduce the number of illegal cell combinations as much as possible. For the representative 10/15 nm libraries we used, the intercell compatibility checks on the CA layer reveal that the TPL DRC violations originate from the local coloring conflicts when special constructs are forbidden to be used for SC layout design. Therefore, we further evaluate the robustness of the library over TPL from the perspective of long-range interactions. In this work, the concepts of cell sequence and cell compatibility check are proposed to enable efficient N-wise intercell compatibility checks. We manage the exponential growth of the N-wise checks by exploiting the pattern regularity of the CA layer and remove redundancy among various cell sequences. Vertical and horizontal boundary checks are explored in order to predict illegal cell combinations. Then, multiple rows of independent cell sequences are randomly combined to evaluate the long-range interactions on the CA layer over MPL. Our main contributions are summarized as follows:

- A systematic framework is proposed to evaluate the library-level MOL robustness for MPL.
- To enable efficient N-wise checks, the cell sequence size reduction problem is proposed and solved efficiently with a graph-based approach and divide-and-conquer technique.
- The illegal cell combinations are predicted with complete checks on two-row combinations of cells for a given SC library.
- For specific SC libraries, the long-range interactions are explored to evaluate the placement-level efforts in reducing QPL to TPL for the MOL layers.

The rest of this paper is organized as follows. Section 2 introduces the related background information, basic definitions, and overall framework. Section 3 presents the formulation and solution of the cell sequence size reduction problem. Section 4 introduces the cell compatibility check to enable efficient horizontal and vertical boundary checks with the elementary checker. Section 5 demonstrates the effectiveness of our framework with the results on two sets of benchmarks, including an ARM 10 nm representative library and a modified Open NanGate 15 nm library.[22] Several key observations are discussed in detail.

2 Preliminaries and Overall Flow

2.1 *Middle-of-Line and Row Structure*

In advanced technology nodes, the complex lithography and process constraints prevent simply using Metal-1 layer for intracell routing. Hence, the MOL layers have been adopted for intracell signal-wiring in the 20 nm technology node and beyond.[19,20] As shown in Fig. 2(a), typical MOL layers include CA and CB.[18] In particular, there may be cross-coupled connections, i.e., special constructs,[18] in the CA layer depending on the technological choices. In the placement row structure shown in Fig. 2(b), SCs are aligned horizontally and share the same height. In this work, we primarily use single-row-height cells to prove the proposed framework, since single-row cells are the predominant cells in an SC library. However, this methodology is not exclusive to single-row cells and can be similarly used to check the robustness of multiple-row-height cells or to verify combinations of single-row cells and multiple-row cells.

In a typical SC row, the mirrored structure of the power and ground rails goes from the left to right of the design. For practical placement, there might be white spaces among neighboring cells on the same row. However, in the library design or evaluation stage, the exact amount of white spaces between any cell pair is unknown without placement information. A typical way to enable robust SC design is to estimate the white spaces conservatively. As illustrated in Fig. 2(c), the white spaces among cells on a single row are removed and cells are abutted horizontally. Thus, we have the following definition.

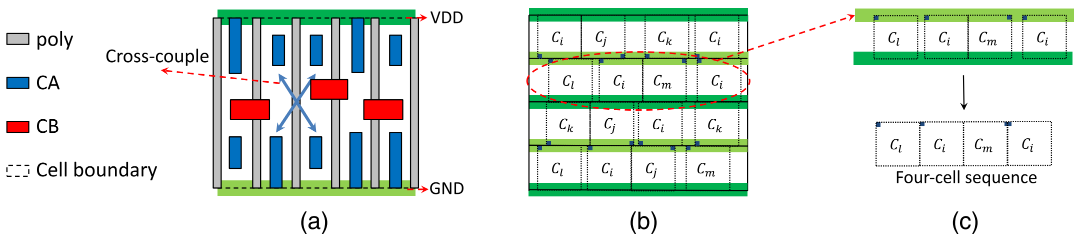


Fig. 2 Standard cell and row structure: (a) middle-of-line structures, (b) multiple rows, and (c) one single row and four-cell sequence.

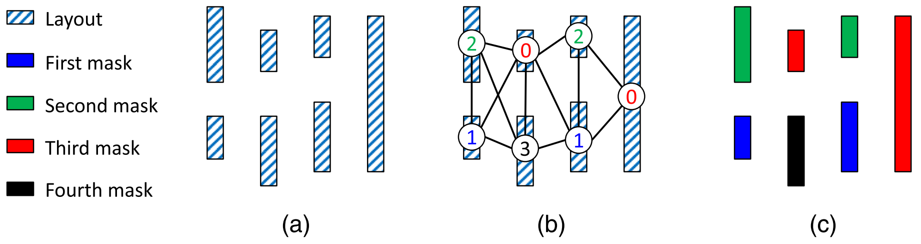


Fig. 3 Layout graph for multiple patterning lithography layout decomposition: (a) target layout, (b) graph and coloring, and (c) mask assignment.

Definition 1 (cell sequence). A set of cells abutting each other in the horizontal direction without overlapping or white space is called a cell sequence. If the sequence consists of k SCs, it is defined as a k-cell sequence.

An example of the four-cell sequence is shown in Fig. 2(c).

2.2 *Conflict Graph*

To determine whether the target layout is free of MPL DRC violations, i.e., k-patterning friendly, the first step is to build the conflict graph.[14] Figures 3(a) and 3(b) show a typical layout in the CA layer and the corresponding conflict graph, respectively. Then, the result of the k-patterning DRC is determined by the k-colorability of the conflict graph. For example, the target layout passes the quadruple ($k = 4$) patterning coloring check in Fig. 3(b) and the mask assignment is demonstrated in Fig. 3(c). The k-patterning DRC is similar to the problem of layout decomposition. The MPL layout decomposition problem is well studied for double patterning,[14,23–25] triple patterning,[15,26–30] and quadruple patterning.[17] Due to the timing criticality of the MOL layers for the SC design, we assume no stitches are allowed for the feasible coloring solution. In addition, MPL DRC aims at deciding the existence of some coloring solution instead of finding an optimal one in layout decomposition. Then, an elementary DRC checker is presented to determine the k-colorability of the conflict graph for the CA layer.

2.3 *Overall Flow*

The overall flow for our framework is demonstrated in Fig. 4. The main target of this flow is to achieve N-wise checks, namely cell compatibility checks, beyond traditional pair-wise checks, to evaluate library robustness over multiple patterning lithography constraints. Traditional pair-wise checks only include two cells abutted in the same row or placed in the neighboring rows, which means the three-cell interactions shown in Fig. 1 cannot be systematically detected. Given an SC library, our proposed framework starts with the cell sequence construction, which produces a set of independent k-cell sequences by removing the redundancy among exhaustively constructed sequences. Independent cell sequence construction efficiently reduces the number of complete N-wise checks while increasing the preparation runtime. At the placement level, the TPL or QPL DRC involves three-cell interactions as shown in Fig. 1. Thus, we need to enable N-wise checks across two rows to predict illegal cell combinations. These checks are independent from each other and can be easily parallelized based on OpenMP framework[31] for speedup. Random benchmarks are constructed to explore long-range interactions. This further produces the cell shift histogram to evaluate the placement-level efforts needed to reduce QPL to TPL. Reducing from QPL to TPL is potentially attractive because it may allow us to save one mask. The output of our framework consists of independent k-cell sequences, illegal cell combinations, and long-range interactions. From the set of independent cell sequences, the designers can find out those critical cells with complex layout patterns within the library. Furthermore, the illegal cell combinations computed give the information on the cell interactions that introduce the MPL conflicts.

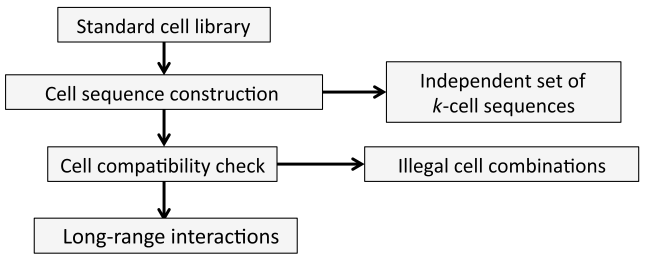

Standard cell library
Cell sequence construction
Independent set of k-cell sequences
Cell compatibility check
Illegal cell combinations
Long-range interactions

Fig. 4 The overall flow for our framework.

In general, a robust library over MPL constraints should introduce a minimum number of illegal cell combinations. The designers can possibly modify the layout of the related critical cells to improve the robustness of the library over MPL constraints. Although the current framework aims at the CA layer with regular layout patterns, it can be easily extended to other MPL layers. In particular, the MPL DRC on a specific layer for each cell combination can be implemented independently, which makes our framework easily adaptable to other layers once the related MPL DRC tools are available.

3 Cell Sequence Construction

In this section, we first define the cell sequence redundancy given a pair of cell sequences. Then, the redundancy graph and divide-and-conquer technique are presented to remove the redundancy and compute the independent set of cell sequences. In the end, we demonstrate the efficient cell sequence construction algorithm.

3.1 *Cell Sequence Size Reduction*

For the k-patterning DRC, a pass or violation is decided by the k-colorability of the conflict graph constructed from the layout.[14,15,17] For the k-colorability of a graph, we have the following theorem.

Theorem 1. If a graph is k-colorable, a subgraph of the graph is also k-colorable. The subgraph of a graph is achieved by removing some nodes or edges of the given graph.

Proof. A k-colorable graph means a feasible coloring solution exists for the graph. Pick one feasible solution and assign color to each node. Then, we have legal color assigned to each node of the subgraph of graph, which means the subgraph of the graph is also k-colorable.

With the regular layout structure on the CA layer, the conflict graph is determined by a set of rectangles and the relative positions among them. Figure 5(a) illustrates the layout of two one-cell sequences, namely C_i and C_j, on the CA layer. As shown in Fig. 5(b), the layout of C_j is a superset of the layout of C_i if we overlay C_i over C_j by aligning the left or right boundaries. Specifically, the layout C_j on the left boundary exactly matches the layout of C_i. Moreover, the line ends of the right boundary of layout C_j are extended vertically compared to the layout of C_i, which may induce extra conflict edges during conflict graph construction. This means that the conflict graph for C_i is the subgraph of the conflict graph for C_j on both the left and right boundaries. According to Theorem 1, the k-colorability of C_j guarantees that of C_i. Then, the k-patterning check for C_i will be redundant to that for C_j. The redundancy check procedure is further discussed in Algorithm 1. Lines 1 to 3 check whether the layout of C_j is a superset of C_i by overlaying cells on the left and right boundaries [as shown in Fig. 5(b), where C_i is overlaid on both the right and the left boundaries of C_j]. The horizontal cell flipping is considered in lines 4 to 7. In particular, this simple redundancy check applies to the MOL layers in the 10/15 nm technology nodes we used. General redundancy check on different layout layers or technology nodes depends on the related design rules and can be more complicated. As mentioned earlier, the MPL DRC involves N-wise checks, which brings the necessity of k-cell sequence construction. Then, the redundancy of C_i to C_j is demonstrated in Fig. 5(c). When we build the two-cell sequences introducing another cell C_k, the layout of C_kC_j is a superset of the layout of C_kC_i and a similar relationship exists between C_jC_k and C_iC_k. If C_j is included in the independent set of cell sequences, C_i should be excluded from that independent set for both MPL DRC and cell sequence construction, since including C_i would be redundant.

Redundancy graph

The ultimate goal for the cell sequence size reduction is to reduce the number of N-wise checks for MPL. Given a set of cell sequences, we introduce the redundancy graph for

Algorithm 1 Redundancy check.

Require: Layout of the CA layer of cell C_i and cell C_j

Output: True if C_i is redundant to C_j, otherwise False;

```
1: if Layout of $C_j$ is a superset of $C_i$ by overlaying $C_i$ over $C_j$ on
   the left and right boundaries then
2:     return True;
3: end if
4: Flip the layout of $C_i$ horizontally;
5: if Layout of $C_j$ is a superset of $C_i$ by overlaying $C_i$ over $C_j$ on
   the left and right boundaries then
6:     return True;
7: end if
8: return False;
```

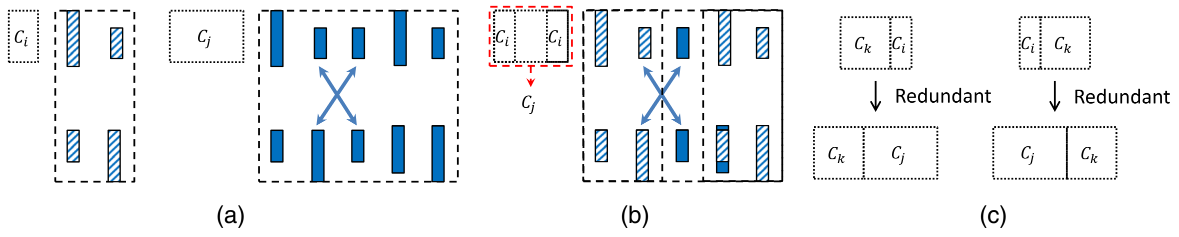


Fig. 5 Sequence cell redundancy: (a) cell C_i and C_j, (b) C_i redundant to C_j, (c) C_kC_i redundant to C_kC_j, C_iC_k redundant to C_jC_k.

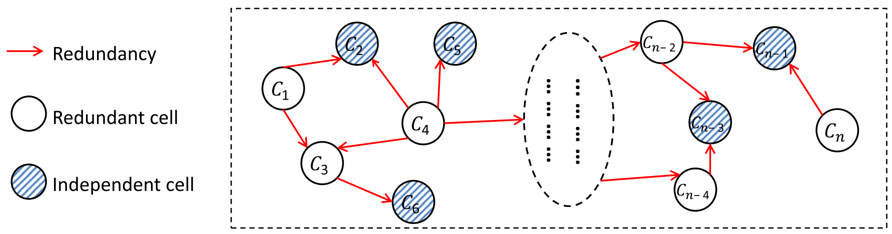


Fig. 6 Cell sequence redundancy graph.

removing redundancy and compute an independent set of cell sequences. The redundancy graph is a directed graph with each node representing one cell sequence. As shown in Fig. 6, there are n nodes and node C_i represents the cell sequence C_i. Now, we may use C_i to represent the cell sequence and the corresponding graph node interchangeably. In addition, an edge from node C_i to node C_j denotes that cell sequence C_i is redundant to cell sequence C_j. For example, in Fig. 6, an edge from C_3 to C_6 means cell sequence C_3 is redundant to C_6. In the end, with the redundancy graph constructed from the input set of cell sequences, the independent set consists of those nodes without successors in the graph. For instance, the dashed nodes $(C_2, C_5, C_6, C_{n-3}, C_{n-1})$ in Fig. 6 belong to the independent cell sequence set.

Algorithm 2 Divide and conquer.

```
Require: A set of cell sequences (CS), graph node size bound ($n_0$);
Output: cell sequence set (ICS)
 1: Define ICS as the independent cell sequence set;
 2: Define RG(CS) as the redundancy graph for set CS;
 3: function DIVIDECONQUER CS
 4:     Compute size = size of CS;
 5:     if size $\le n_0$ then
 6:         ICS = set of nodes without successors in RG(CS)
 7:         return ICS;
 8:     else
 9:         Divide CS into $\sqrt{\text{size}}$ subsets;
10:         for each subset for CS do
11:             DIVIDECONQUER subset;
12:         end for
13:     end if
14: end function
```

Given n cell sequences, the time complexity to fully build a redundancy graph is $O(n^2)$ because in the worst case, we need to determine the redundancy between each pair of cells. For practical implementations, the input size can be as large as $n \ge 10^6$, which would incur large runtime. Instead of fully constructing the redundancy graph, an iterative divide-and-conquer technique is deployed to optimize the set within an affordable runtime. Algorithm 2 illustrates the details on the divide-and-conquer technique with the redundancy graph construction. Lines 5 to 10 explain the redundancy graph construction and independent set computation if the input size is within the graph node size bound (n_0). Otherwise, the input cell sequence set is recursively segmented into various subsets until each segment is less than n_0. We see that the intersubset redundancy among cell sequences on different subsets is ignored in Algorithm 2. To account for that, Algorithm 3 demonstrates the iterative approach to implement the cell sequence size reduction. For each iteration from line 4 to line 9, the cell sequence set is optimized

Algorithm 3 Cell sequence size reduction.

```
Require: A set of cell sequences (CS), iteration bound ($m_0$);
Output: Independent cell sequences (ICS)
 1: Define iteration count $m = 0$;
 2: Define ICS as the independent cell sequence set;
 3: while $m \le m_0$ do
 4:     ICS = DIVIDECONQUER CS;
 5:     if size of ICS< size of CS then
 6:         CS = random-shuffle(ICS);
 7:     else
 8:         break;
 9:     end if
10:     $m = m + 1$;
11: end while
12: return ICS
```

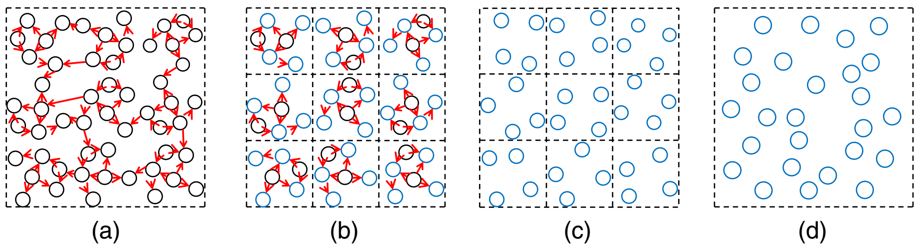


Fig. 7 Divide and conquer: (a) divide the cell sequences into several subsets, (b) remove redundancy in each subset, (c) random shuffle, and (d) for next iteration.

using the divide-and-conquer technique, and the set is randomly shuffled for the next iteration. The exit condition is that no redundant cell is found or the iteration count exceeds the preset bound (m_0). A more intuitive interpretation of Algorithms 2 and 3 is illustrated in Fig. 7. The input set is segmented into several subsets in Fig. 7(a), each subset is optimized based on redundancy graph in Fig. 7(b), and the results are shown in Fig. 7(c). To prepare for the next iteration, the optimized set is randomly shuffled in Fig. 7(d).

3.2 *Cell Sequence Construction*

Given an SC library consisting of m cells, i.e., m one-cell sequences, the exhaustive construction scheme gives the number of k-cell sequences as $(2 \times m)^n$ (the factor of 2 accounts for horizontal cell flipping). A simple calculation with $m = 100$ yields that the number of four-cell sequences will be 1.6 billion. To manage the size to an acceptable level, cell sequences are built in a bottom-up manner as shown in Algorithm 4. We iterate through the number of cells, denoted as k, in one sequence in line 4. The independent k-cell sequences are computed in line 5. Then, in lines 6 to 9, the $(k+1)$-cell sequences are constructed from the independent sets, including the one-cell sequence set to the k-cell sequence set. The iteration ends with the preset maximum number of cells (k_0) in one sequence. With Algorithm 4, the k-cell sequences are constructed from the independent cell sequences in a bottom-up manner. For practical implementations, this method helps to achieve the complete set of up to three-cell sequences in a reasonable amount of runtime despite the exponential growth of the cell sequences.

Algorithm 4 Cell sequence construction.

```
Require: The set of one-cell sequences ($CS_1$) in the library and
         the maximum number of cells in one sequence ($k_0$);
Output: Independent cell sequences $ICS_{k_0}$
 1: Define $CS_k$ as the set of $k$-cell sequences;
 2: Define $ICS_k$ as the set of independent $k$-cell sequences;
 3: Define initial number of cells in one sequence as $k = 1$;
 4: while $k \& lt; = k_0$ do
 5:     $ICS_k = CSSR(CS_k)$);
 6:     for $m = 1$; $m \le \lfloor (k+1)/2 \rfloor$; $m = m + 1$ do
 7:         Construct the $(k+1)$-cell sequence set ($set_{k+1}$) from
            $ICS_m$ and $ICS_{k+1-m}$;
 8:         $CS_{k+1} = CS_{k+1} \cup set_{k+1}$;
 9:     end for
10:     $k = k + 1$;
11: end while
12: return $ICS_{k_0}$
```

4 Cell Compatibility Check

In this section, we build two-row combinations of cells and multirow combinations of cells, to predict illegal cell combinations and explore long-range interactions, respectively. An elementary checker is presented to predict the local conflicts and nonlocal conflicts for TPL. Since our checks are built from cell sequences, intrarow checks are only subsets of the checks on two-row combinations of cells and are not built explicitly here.

4.1 *Two-Row Combinations of Cells*

Ideally, we need to exhaustively build cell combinations across as many rows as possible to explore long-range interactions for the MPL DRC. However, exhaustive checks on two-row combinations of cells are sufficient for the local conflict detections for QPL and TPL. In addition, the exponential growth of the cell combinations beyond two rows makes it unaffordable in practice. Hence, we only exhaustively build two-row combinations of cells from the independent cell sequences. In addition, we also enable the cell sequence shifting to account for the relative position changes of two cell sequences on neighboring rows. As shown in Fig. 8(a), one type of two-row combination of cells consists of an independent one-cell sequence and an independent three-cell sequence on top of it, which is denoted as ics_1 on ics_3. For practical placement, the one-cell sequence in

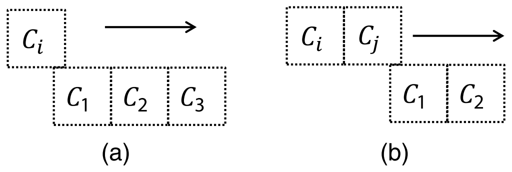


Fig. 8 Two-row combinations of cells: (a) independent one-cell sequence on independent three-cell sequence (ics_1 on ics_3) and (b) independent two-cell sequence on independent two-cell sequence (ics_2 on ics_2).

Fig. 8(a) can be anywhere on top of the three-cell sequence cell. Therefore, we allow for the cell sequence shifting from left to right in the unit of placement pitch and the k-patterning colorability is checked for each placement pitch. Moreover, Fig. 8(b) shows another type of cell combination denoted as ics_2 on ics_2, and the cell shifting is also evaluated by the DRC. For practical implementations, we build three types of cell combinations, including ics_1 on ics_2, ics_1 on ics_3, and ics_2 on ics_2. Huge amounts of MPL checks are needed if the cell shifting is allowed for the exhaustively constructed two-row combinations of cells. Since these MPL checks are independent from each other, parallel implementations are adopted to accelerate the comprehensive checks on two-row combinations of cells.

4.2 *Multirow Combinations of Cells*

Next, we randomly build cell combinations in multiple rows, i.e., multirow combinations of cells, to explore the long-range interactions. Specifically, benchmarks with multiple rows are randomly generated from the independent one-cell sequences as illustrated in Fig. 9. We preset a bound for the design width and height for placement. Independent one-cell sequences are randomly chosen and abutted horizontally and vertically to fill in the placement region. In Fig. 9(a), one-cell sequences are abutted horizontally without gaps among their placement and routing boundaries. However, for Fig. 9(b), the cell shifting is enabled to greedily remove the local conflicts for MPL, which creates white spaces among cells in the same row. As discussed in Sec. 5, these types of multirow combinations of cells help to evaluate the placement-level efforts required to reduce QPL to TPL. In particular, there exist white spaces at the end of each placement row when the random cell cannot be inserted due to the fixed design width.

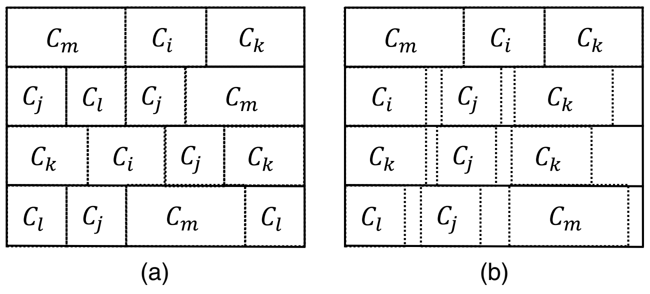


Fig. 9 Multirow combinations of cells: (a) multiple rows of one-cell sequences and (b) multiple rows of one-cell sequences with local conflicts removed.

4.3 *Multiple Patterning Lithography Compatibility Check*

Given a cell combination, our elementary k-patterning checker starts with the conflict graph construction. With the conflict graph, the k-patterning DRC consists of three stages, including iterative vertex removal, clique detection, and backtrack coloring. The iterative vertex removal means that any conflict graph node with degree $<k$ can be removed without impacting the colorability of the graph.[17,26] The clique detection searches for a $(k+1)$-clique, which is not k-colorable. For the complex conflict graph without $(k+1)$-cliques, the backtrack coloring is deployed to determine the k-colorability of the conflict graph in a brute-force manner. Our elementary checker works for small-scale conflict graph within reasonable runtime.

Local conflict and nonlocal conflict

Next, we discuss why QPL, instead of TPL, is needed for the CA layer in the representative technologies. With the cell combinations constructed and our elementary checker, we can enable efficient triple patterning checks. In general, we find that there are two types of conflicts, local conflicts and nonlocal conflicts, which lead to the TPL DRC violations. As shown in Figs. 10(a) and 10(b), the local conflict originates from the local four-clique involving neighboring rows. The nonlocal conflict is due to the coloring failure of the layout across three rows or beyond. An example is illustrated in Figs. 10(a), 10(c), and 10(d). For a three-coloring of the graph, if we have adjacent three circles sharing one edge, the two nodes without edge connection should share the same color. As shown in the dashed rectangle in Fig. 10(d), the two nodes without edge connection share the same color, denoted as 1. The same color constraint propagates across the entire graph and introduces a coloring conflict in the dash ellipse in Fig. 10(d).

Furthermore, the graph nodes due to the special constructs, as shown in Fig. 10(c), play a critical role in the nonlocal conflicts. If we forbid or remove the special construct in the library design, nonlocal conflicts never happen in our three-patterning checks. This means the TPL DRC violations are simply due to the local conflicts, i.e., local four-cliques for TPL. This also leads to the possibility of removing local four-cliques during placement and reducing QPL to TPL for the CA layer. A local four-clique consists of four rectangles, which at most involves four standard cells neighboring each other, and greedy local four-clique removal for multirow combinations of cells can be achieved by shifting cells and adding white spaces, as shown in Fig. 9(b).

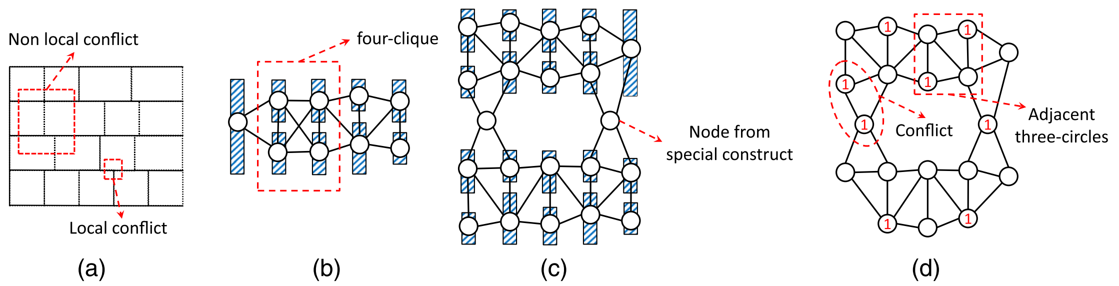


Fig. 10 Violations of triple patterning lithography design rule checks: (a) the local and nonlocal conflicts, (b) the local conflict, (c) the nonlocal conflict, and (d) the coloring failure for the nonlocal conflict.

5 Experimental Results

We have implemented our algorithms in C++ and tested them with two sets of benchmarks. One is an ARM representative 10 nm standard cell library consisting of 116 logic cells. The other is the NanGate 15 nm open cell library[22] consisting of 76 logic cells. All experiments on the ARM representative 10 nm library are performed on a Linux machine with 3.33 GHz Intel Xeon CPU X5680. For experiments on the modified NanGate 15 nm library, they are performed on a Linux machine with 3.4 GHz Intel Core and 32 GB memory. For the NanGate 15 nm open cell library, there exist three MOL layers for intracell routing. The CA layer is predesigned as being explicitly decomposed into two masks on separate layers. In order to test our methodology, we recombine these two MOL layers into one single layer and then run our algorithms on the modified layout using the combined MOL layer. Industrial design rules are applied to the ARM 10 nm library during conflict graph construction. The minimum single color spacing is set as 80 nm for the NanGate 15 nm library. In Algorithm 2, the graph size bound (n_0) is set as 100, and in Algorithm 3, the iteration bound (m_0) is set as 10. We first show the benefit of our cell sequence size reduction. After identifying independent cell sequences, the size reduction of two-row combinations of cells is demonstrated. To accelerate the MPL checks on two-row combinations of cells, we employ parallelization based on the OpenMP framework.[31] Then, we discuss a set of observations from the cell compatibility check. In particular, the cell shifting histogram is introduced to demonstrate the placement-level efforts required to reduce QPL to TPL for the CA layer with the representative libraries we used. The nonlocal conflicts for TPL are inevitable if special constructs are included in the SC design as discussed in Sec. 4.3. If the special constructs are excluded, it is possible to resolve the local conflicts in the placement level and reduce QPL to TPL. Therefore, we build two sets of multirow combinations of cells, one is from the NanGate library and the other is obtained by modifying the ARM library and removing the special constructs within the SCs.

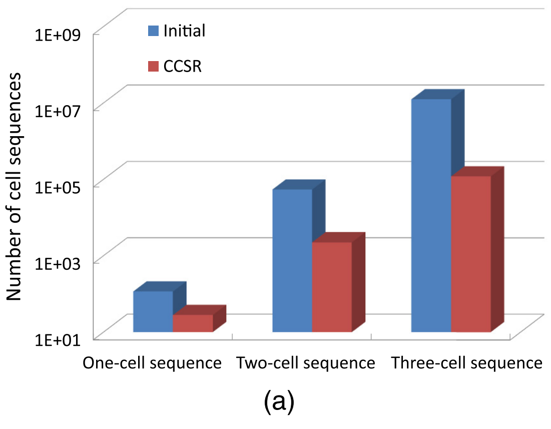
5.1 *Cell Sequence Size Reduction*

The cell sequence size reduction helps remove the redundancy among cell sequences and enables efficient cell sequence construction as discussed in Sec. 3. However, from the theoretical perspective, the number of cell sequences still grows exponentially with the length of the sequence. As shown in Fig. 11, the maximum number of cells we can achieve in one sequence is only 3. However, the three-cell sequences are sufficient to detect local conflicts by exploring cell interactions in two rows for TPL and QPL. Meanwhile, we see orders of magnitude reduction on the number of cell sequences from the brute-force construction for both the ARM and NanGate representative libraries. In Fig. 11, Initial denotes the brute-force construction, while CSSR denotes the cell sequence size reduction. Meanwhile, for each three-cell sequence built from the library, it is either included in the independent set or redundant to some sequence within the independent set. Moreover, we discuss the necessity of the divide-and-conquer approximation and speedup. As mentioned earlier, the redundancy graph construction has the time complexity as $O(n^2)$, where n denotes the graph node size. Here, we give an example on the runtime for practical implementations. For the ARM representative library, the number of three-cell sequences before redundancy removal is $n = 120{,}960$, and the average amount of time to compute the redundancy between a pair of three-cell sequences is $t_0 = 0.36$ s from our implementation. Then, we can estimate the amount of runtime to compute the redundancy graph without the speedup as $t = 0.5 \times n^2 \times t_0$, which is approximately 83 years. With the divide-and-conquer speedup, we achieve the results in Fig. 11 within several hours. Specifically, the one-cell sequence construction takes 0.22 s, two-cell sequence construction takes 16.6 s, and three-cell sequence construction takes 5210.5 s.

5.2 *Checks on Two-Row Combinations of Cells*

5.2.1 *Size reduction*

Figure 12 demonstrates the size reduction of two-row combinations of cells. For the horizontal axis, ics_i on ics_i denotes one type of two-row combination of cells, which has an i-cell sequence on top of a j-cell sequence. As discussed in Sec. 4, the two-row combinations of cells are only exhaustively built from the independent cell sequences based on the cell sequence size reduction. Then, the size of ics_i on $ics_j = 2\times$ the size of i-cell sequences $\times$ the size of j-cell sequences, where the factor of 2 accounts for the vertical flipping of cell sequences. Actually, the orders of magnitude



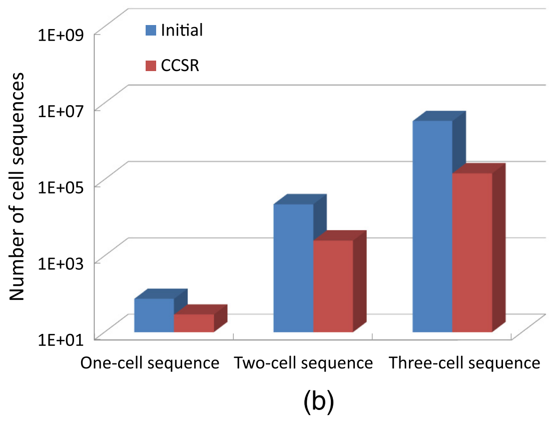


Fig. 11 Cell sequence size reduction: (a) ARM 10 nm representative library and (b) modified NanGate 15 nm library.

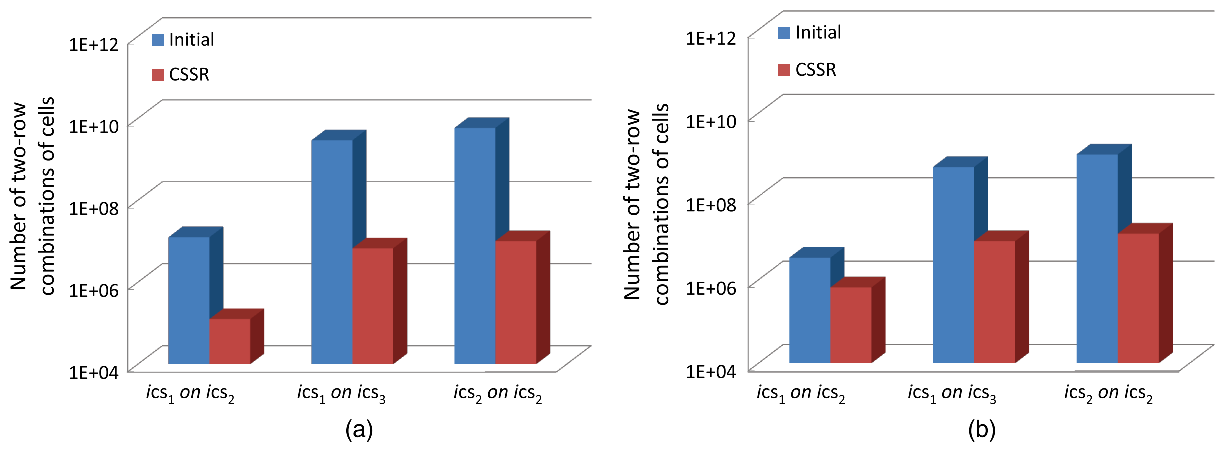


Fig. 12 Size reduction of two-row combinations of cells from the cell sequence size reduction: (a) ARM 10 nm representative library and (b) modified NanGate 15 nm library.

reduction for two-row combinations of cells are due to the cell sequence size reduction in Fig. 11. For example, in Fig. 12(a), we observe that billions of ics_1 on ics_3 are reduced to millions of them. In addition, any two-row combination of cells constructed from the cell library is either included in the reduced sets or redundant to a cell combination in the reduced sets.

5.2.2 *Complete checks*

The two-row combinations of cells have been explicitly constructed from the independent k-cell sequences in Sec. 5.2.1. Millions of cell combinations are constructed for both ics_1 on ics_3 and ics_2 on ics_2. The number of MPL checks will be linearly proportional to the size of two-row combinations of cells if we allow for cell shifting. In practice, the serial implementation needs several hours to finish all the checks on two-row combinations of cells. The parallel implementation based on OpenMP[31] reduces the overall runtime to <1 h. All two-row combinations of cells pass the QPL checks, and the results from TPL checks of ics_1 on ics_1, ics_1 on ics_2, ics_1 on ics_3, and ics_2 on ics_3 for NanGate 15 nm library are demonstrated in Figs. 13(a)–13(d), respectively. For two-row combinations of cells, the root cause of the TPL DRC failure is the local four-clique from the row-to-row interaction as shown in Fig. 10(b). We allow for the cell shifting to explore the cell interactions at different placement pitches for each cell combination. In Fig. 13, the horizontal axis is the percentage of placement pitches leading to local four-cliques over total number of placement pitches shifted for each two-row combination of cells. The vertical axis is the frequency, meaning the number of cell combinations sharing the same percentage of placement pitches leading to local four-cliques. For ics_1 on ics_1 in Fig. 13(a), the percentages of placement pitches leading to local four-cliques for all two-row combinations of cells are within 60%. The designers can track those cell combinations that have percentages over 50% and improve the cell layout design for related standard cells. Fifty percent is an arbitrary number and can be changed based on the designers' preference. Our framework assumes conservative horizontal and vertical boundary checks, which means white space is forbidden for neighboring cells within each combination. As the cell combination complexity increases from Figs. 13(a)–13(d), the percentage of placement pitches leading to local four-cliques also increases. Under extreme cases in Figs. 13(c) and 13(d), over 80% of the placement pitches introduce local four-cliques for some two-row combinations of cells. From these checks on two-row combinations of cells, we observe that some two- or three-cell sequences abutting each other without white space are not desired for practical placement. Since our checks on two-row combinations of cells are complete, traditional pair-wise checks are a subset of the checks for ics_2 on ics_2. Our complete checks guarantee to find all the illegal cell combinations leading to TPL local conflicts for the libraries we used.

5.3 *Long-Range Interactions*

Long-range interaction is introduced by multiple patterning lithography coloring constraints, which is beyond the traditional pair-wise checks. Table 1 demonstrates the multirow combinations of cells randomly generated from the set of independent one-cell sequences for the NanGate 15 nm library. Within the tables, Bench denotes the benchmark name, Size is the area for placement, Util (%) is the utilization rate representing the percentage of cell area over total area of the design, Cell# is the number of cells, Rect# is the number of rectangles, Pass denotes the DRC result, and CPU denotes the amount of runtime. We have the utilization rate [Util(%)] <100% because white space is left at the end of a row when there is not enough to insert another random one-cell sequence. As shown in both tables, all the cell combinations pass the QPL DRCs. In contrast, TPL DRCs lead to varying numbers of violations. As mentioned earlier, the violations of TPL DRCs originate from two types of conflicts, including local conflicts and nonlocal conflicts. The nonlocal conflicts are due to the propagation of same color constraints across three rows or beyond as shown in Fig. 10(d). The conflict graph nodes from the special constructs may prevent simplifying the graph into small components. Actually, the use of special constructs varies from technology to technology, and the NanGate 15 nm library does not have this cross-coupled structure. Thus, the three-patterning checks for all the multirow combinations of cells from the NanGate library can be achieved in Table 1 (without the special construct that merges the graph across a row, the graph naturally partitions

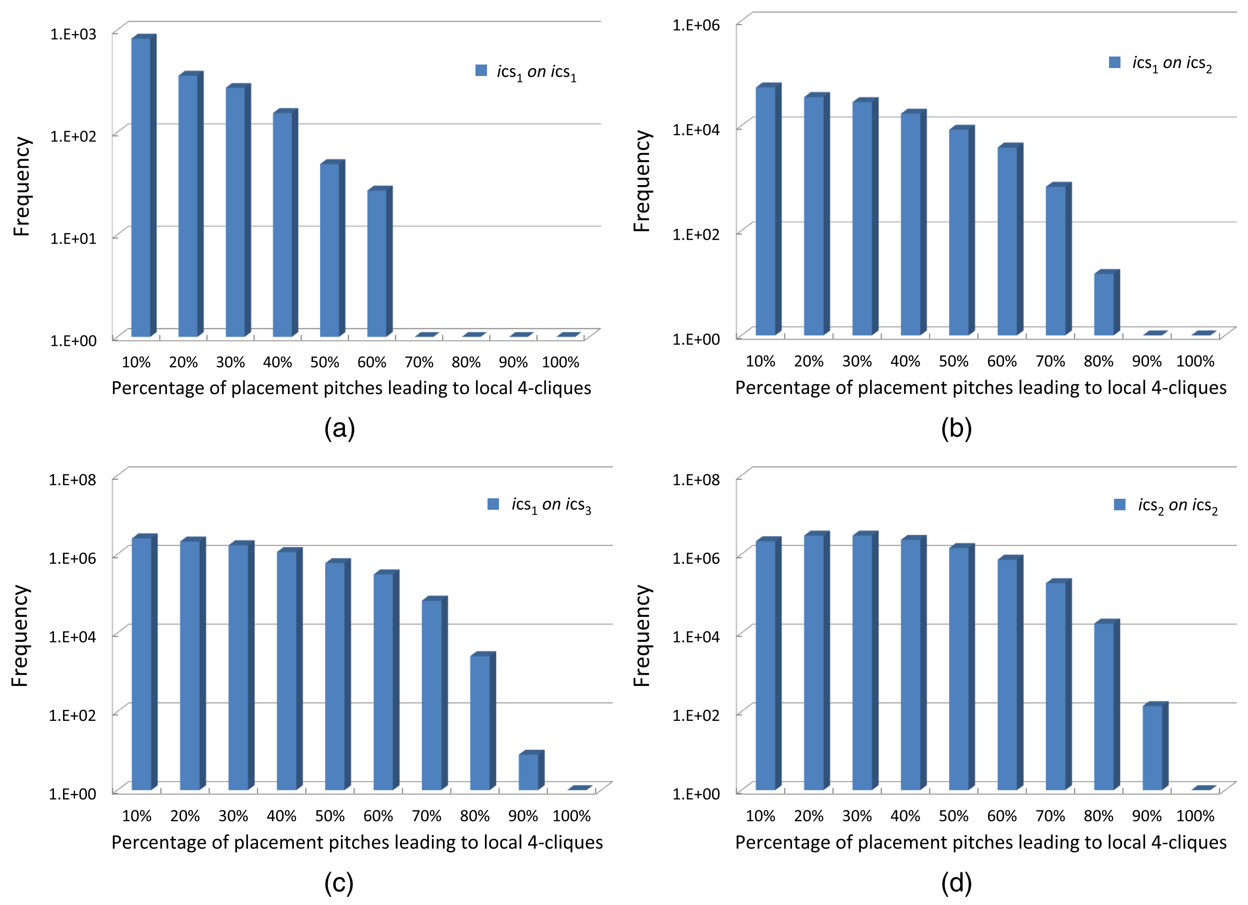


Fig. 13 Checks on two-row combinations of cells for the NanGate 15 nm library: (a) ics_1 on ics_1, (b) ics_1 on ics_2, (c) ics_1 on ics_3, and (d) ics_2 on ics_2.

itself by row, which leads to much more manageable graph sizes and runtimes).

For the NanGate library without special constructs, the TPL DRC violations of multirow combinations of cells are simply due to the local four-cliques. As shown in Fig. 14(a), the conflict graph on the CA layer is split into independent components in a row-by-row manner. The local four-cliques come from the cell interactions between neighboring rows. The multirow combinations of cells without local four-cliques are still built horizontally and vertically from the random one-cell sequences. When local four-cliques are detected, the current cell being inserted is shifted from left to right by one placement pitch until all local four-cliques are eliminated. The nonlocal conflicts for TPL are inevitable if special constructs are included in the SC design as discussed in Sec. 4.3. If the special constructs

Table 1 NanGate 15 nm multirow combinations of cells with local 4-cliques.

					TPL checks		QPL checks	
Bench	Size (um^2)	Util (%)	Cell#	Rect#	Pass	CPU (s)	Pass	CPU (s)
NanGate_1	0.8×0.8	91.0	77	1870	No	0.07	Yes	0.04
NanGate_2	1.5×1.5	96.5	288	7875	No	0.35	Yes	0.29
NanGate_3	31×31	98.0	1257	31,783	No	3.37	Yes	2.83
NanGate_4	61×61	99.0	4948	12,864	No	52.7	Yes	39.6
NanGate_5	115×115	99.5	17,382	454,671	No	584.9	Yes	449.1
NanGate_6	154×154	99.6	31,057	809,015	No	1823.9	Yes	1403.4

Note: TPL, triple patterning lithography; QPL, quadruple patterning lithography.

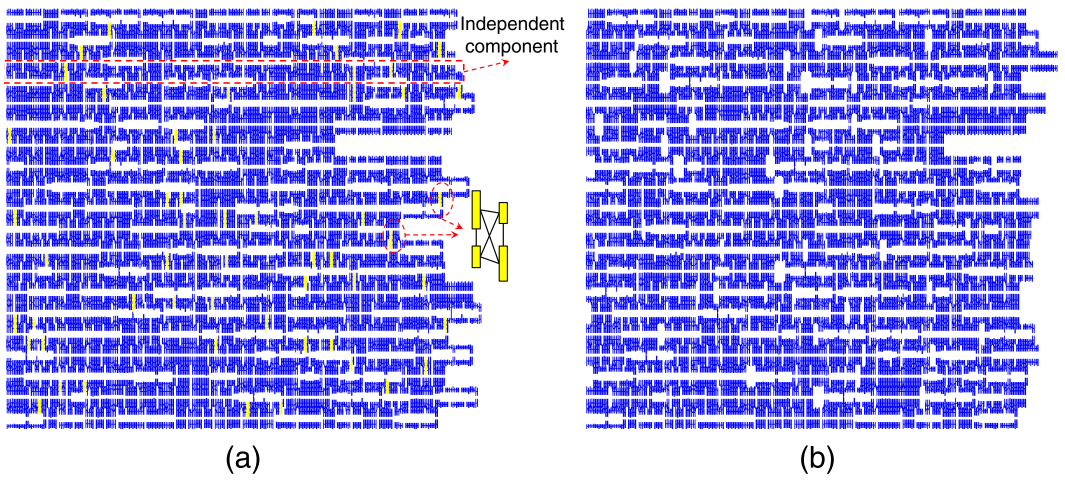


Fig. 14 Benchmarks with multiple rows: (a) with local 4-cliques and (b) without local 4-cliques.

are excluded, it is possible to resolve the local conflicts in the placement level and reduce QPL to TPL. Therefore, we build two sets of multirow combinations of cells, one is from the NanGate library in Table 2 and the other is obtained by modifying the ARM library and removing the special constructs within the standard cells in Table 3.

We further discuss the cell shifting histogram in Fig. 15. In the cell shifting histogram, the horizontal axis is the number of placement pitches each inserted cell is shifted before achieving a placement site that is free of local four-cliques and the vertical axis is the number of cells. In general, the cell shifting histogram is an initial attempt to capture the effort that might be needed by a placement algorithm in order to achieve a legal placement given the same scenario of cells. For most benchmarks, the number of cells is a nonincreasing function of the number of pitches shifted. To the first order, we prefer that cells are shifted less than three placement pitches, as shown in the shadowed region marked Preferred region in Fig. 15. In the representative libraries we used, a minimum-sized inverter occupied three placement pitches (in width), so we chose to use the same width as the upper bound of our Preferred region. For each benchmark, we further calculate the percentage of cells in the preferred region (C.i.p) and the utilization rate (Util), which are two metrics to evaluate the placement-level efforts to reduce QPL to TPL. If TPL on the CA layer for practical designs is desired, we expect to require that a high percentage of cells would be in the preferred region while the utilization rate of the placement is also high. A high percentage of cells in the preferred region means that minimum placement-level efforts will be needed to remove the local four-cliques and achieve TPL-friendly design. The high utilization of each benchmark means that the cell shifting in the placement stage will introduce less area penalty. To improve the robustness at the SC library level, SC designers need to increase the utilization rate and the percentage of cells in the preferred region for the multirow combinations of cells.

In the end, the cell compatibility check yields a unique set of observations for the representative libraries used: (1) the four-clique exists due to the row-to-row interactions, which means QPL needs to be deployed to enable implicit coloring for standard cells; (2) there is no local five-clique for the CA layer in the representative technology used; and (3) for all the cell combinations generated, the colorability for QPL can be easily proved with our elementary checker.

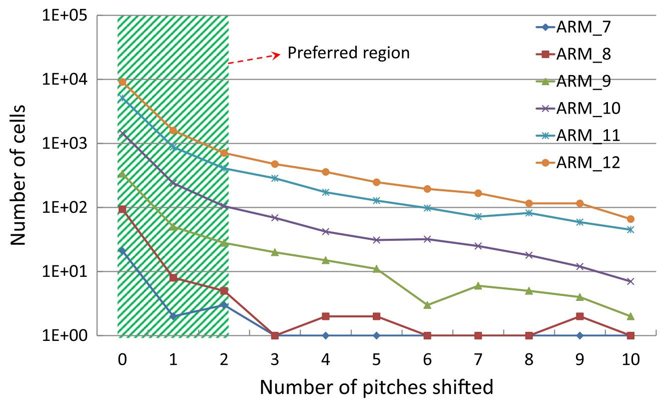
Table 2 NanGate 15 nm multirow combinations of cells without local 4-cliques.

						TPL checks	
Bench	Size (um^2)	Util (%)	Cell#	Rect#	C.i.p (%)	Pass	CPU (s)
NanGate_7	0.8×0.8	91.3	72	1865	91.7	Yes	0.02
NanGate_8	1.5×1.5	92.8	295	7499	92.5	Yes	0.19
NanGate_9	31×31	94.6	1191	30,561	90.3	Yes	2.53
NanGate_10	61×61	95.6	4800	123,621	90.4	Yes	37.9
NanGate_11	115×115	95.5	16,757	433,639	90.0	Yes	461.8
NanGate_12	154×154	95.7	29,870	771,964	89.9	Yes	1475.8

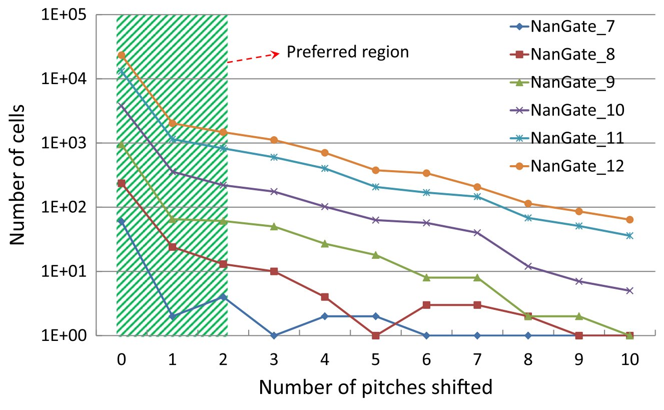
C.i.p, percentage of cells in the preferred region.

Table 3 ARM 10 nm representative multirow combinations of cells (special constructs removed) without local 4-cliques.

						TPL checks	
Bench	Size (um^2)	Util (%)	Cell#	Rect#	C.i.p (%)	Pass	CPU (s)
ARM_7	17.2×17.2	89.7	29	913	89.7	Yes	0.09
ARM_8	34.6×34.6	86.9	117	4443	91.5	Yes	1.00
ARM_9	69.1×69.1	84.7	480	17,300	86.3	Yes	2.23
ARM_10	138×138	92.3	2028	75,405	88.4	Yes	1237.6
ARM_11	259×259	93.2	7338	267,291	87.1	Yes	4904.2
ARM_12	346×346	93.5	13,162	476,000	86.8	Yes	1502.1



(a)



(b)

Fig. 15 The cell shifting histogram denoting the number of pitches shifted to remove local 4-cliques: (a) ARM 10 nm representative library (special constructs removed) and (b) modified NanGate 15 nm library.

6 Conclusion and Discussions

This paper proposes a comprehensive framework to evaluate the MOL robustness for a standard cell library over multiple patterning lithography constraints. To reduce the number of N-wise checks, the issues of cell sequence construction and size reduction are proposed and solved efficiently. The cell compatibility check and an elementary checker are presented to enable efficient N-wise checks. For the representative 10/15 nm libraries we used, the cell shifting histogram is proposed to quantify the placement-level efforts required to reduce QPL to TPL for the CA layer. Our framework applies to the CA layer and can be easily extended to other MPL layers in future technology nodes. Our future work includes advanced approaches to explore long-range interactions to detect and resolve nonlocal TPL conflicts in a systematic way.

References

1. L. Liebmann, D. Pietromonaco, and M. Graf, "Decomposition-aware standard cell design flows to enable double-patterning technology," *Proc. SPIE* **7974**, 79740K (2011).
2. D. Z. Pan, B. Yu, and J.-R. Gao, "Design for manufacturing with emerging nanolithography," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* **32**(10), 1453–1472 (2013).
3. K. Lucas et al., "Implications of triple patterning for 14 nm node design and patterning," *Proc. SPIE* **8327**, 832703 (2012).
4. R. Merritt, "Intel sees quad-patterned path to 10-nm chips," September 2012, http://www.eetimes.com/document.asp?doc_id=1262512 (13 September 2012).
5. X. Xu et al., "Self-aligned double patterning aware pin access and standard cell layout co-optimization," in *ACM Int. Symp. on Physical Design*, pp. 101–108 (2014).
6. K. B. Agarwal et al., "Multi-patterning lithography aware cell placement in integrated circuit design," U.S. Patent 8495548 (2013).
7. B. Yu et al., "Methodology for standard cell compliance and detailed placement for triple patterning lithography," in *IEEE/ACM Int. Conf. on Computer-Aided Design*, pp. 349–356 (2013).
8. M. Cho, Y. Ban, and D. Z. Pan, "Double patterning technology friendly detailed routing," in *IEEE/ACM Int. Conf. on Computer-Aided Design*, pp. 506–511 (2008).
9. M. Mirsaeedi, J. A. Torres, and M. Anis, "Self-aligned double-patterning (SADP) friendly detailed routing," *Proc. SPIE* **7974**, 79740O (2011).
10. Q. Ma, H. Zhang, and M. D. F. Wong, "Triple patterning aware routing and its comparison with double patterning aware routing in 14nm technology," in *ACM/IEEE Design Automation Conf.*, pp. 591–596 (2012).
11. Y.-H. Lin et al., "TRIAD: a triple patterning lithography aware detailed router," in *IEEE/ACM Int. Conf. on Computer-Aided Design*, pp. 123–129 (2012).
12. X. Xu et al., "PARR: pin access planning and regular routing for self-aligned double patterning," in *ACM/IEEE Design Automation Conf.*, Vol. 6, pp. 1–28 (2015).
13. C.-H. Hsu, Y.-W. Chang, and S. R. Nassif, "Simultaneous layout migration and decomposition for double patterning technology," in *IEEE/ACM Int. Conf. on Computer-Aided Design*, pp. 595–600 (2009).
14. A. B. Kahng et al., "Layout decomposition for double patterning lithography," in *IEEE/ACM Int. Conf. on Computer-Aided Design*, pp. 465–472 (2008).

15. B. Yu et al., "Layout decomposition for triple patterning lithography," in *IEEE/ACM Int. Conf. on Computer-Aided Design*, pp. 1–8 (2011).
16. B. Yu, J.-R. Gao, and D. Z. Pan, "Triple patterning lithography (TPL) layout decomposition using end-cutting," *Proc. SPIE* **8684**, 86840G (2013).
17. B. Yu and D. Z. Pan, "Layout decomposition for quadruple patterning lithography and beyond," in *ACM/IEEE Design Automation Conf.*, pp. 1–6 (2014).
18. M. Rashed et al., "Innovations in special constructs for standard cell libraries in sub 28nm technologies," in *IEEE Int. Electron Devices Meeting*, pp. 248–251 (2013).
19. G. Northrop, "Design technology co-optimization in technology definition for 22nm and beyond," in *IEEE Symp. on VLSI Technology*, pp. 112–113 (2011).
20. M. Rashed et al., "Semiconductor device with transistor local interconnects," U.S. Patent 8581348 (2013).
21. W. Ye et al., "Standard cell layout regularity and pin access optimization considering middle-of-line," in *ACM Great Lakes Symp. on VLSI*, pp. 289–294, ACM (2015).
22. NanGate, "NanGate FreePDK15 Open Cell Library," 2014, http://www.nangate.com/?page_id=2328 (29 May 2014).
23. K. Yuan, J.-S. Yang, and D. Z. Pan, "Double patterning layout decomposition for simultaneous conflict and stitch minimization," in *ACM Int. Symp. on Physical Design*, pp. 185–196 (2009).
24. Y. Xu and C. Chu, "A matching based decomposer for double patterning lithography," in *ACM Int. Symp. on Physical Design*, pp. 121–126 (2010).
25. X. Tang and M. Cho, "Optimal layout decomposition for double patterning technology," in *IEEE/ACM Int. Conf. on Computer-Aided Design*, pp. 9–13 (2011).
26. S.-Y. Fang, W.-Y. Chen, and Y.-W. Chang, "A novel layout decomposition algorithm for triple patterning lithography," in *ACM/IEEE Design Automation Conf.*, pp. 1185–1190 (2012).
27. H. Tian et al., "A polynomial time triple patterning algorithm for cell based row-structure layout," in *IEEE/ACM Int. Conf. on Computer-Aided Design*, pp. 57–64 (2012).
28. J. Kuang and E. F. Young, "An efficient layout decomposition approach for triple patterning lithography," in *ACM/IEEE Design Automation Conf.*, Vol. 6, pp. 1–19 (2013).
29. B. Yu et al., "A high-performance triple patterning layout decomposer with balanced density," in *IEEE/ACM Int. Conf. on Computer-Aided Design*, pp. 163–169 (2013).
30. Y. Zhang et al., "Layout decomposition with pairwise coloring for multiple patterning lithography," in *IEEE/ACM Int. Conf. on Computer-Aided Design*, pp. 170–177 (2013).
31. OpenMP Architecture Review Board, "OpenMP application program interface," Version 3.1, http://openmp.org/wp/ (July 2011).

Xiaoqing Xu is a PhD student in electrical and computer engineering at the University of Texas, Austin, under the supervision of Professor David Z. Pan. His research interests include robust standard cell design, design for manufacturability, and physical design. He received the MCD Fellowship from the University of Texas at Austin in 2012 and the Excellent Graduate from Peking University in 2012.

Brian Cline is a principal design engineer at ARM Holdings, focusing on logic, memory, and processor design and performance at advanced process nodes.

Greg Yeric is a senior principal R&D engineer at ARM Holdings, focusing on design-technology co-optimization and predictive technology. He began his career at Motorola's Advanced Products Research and Development Laboratories in embedded nonvolatile memory process integration, subsequently working in technology development roles at TestChip Technologies, HPL Technologies, and Synopsys.

Bei Yu is an assistant professor in the Department of Computer Science and Engineering, Chinese University of Hong Kong. He received his PhD degree from the University of Texas at Austin in 2014. His research interests include design for manufacturing, hardware security, machine learning, and combinatorial algorithm with applications in VLSI computer-aided design (CAD). He received EDAA Outstanding Dissertation Award, Best Paper Awards at ICCAD13 and ASPDAC12, and three other Best Paper Award Nominations at DAC'14, ASPDAC'13, and ICCAD'11.

David Z. Pan is a professor in the Department of Electrical and Computer Engineering, University of Texas at Austin. His research interests include nanometer IC design for manufacturability/reliability, new frontiers of physical design, and CAD for emerging technologies. He has published over 200 technical papers and is the holder of eight U.S. patents. He has received many awards, including SRC Technical Excellence Award and 11 best paper awards, among others. He is an IEEE fellow.