Incorporating Cut Redistribution with Mask Assignment to Enable 1D Gridded Design

Jian Kuang, Evangeline F. Y. Young, Bei Yu
Outline

• Background
• Problem Formulation
• ILP
• Graph Model and Algorithm
• Post-processing
• Results and Conclusions
Lithography Technologies

- **DPL (double patterning lithography)**
  - One layout is decomposed into two masks
  - Litho-etch process is repeated twice
  - Resolution can be improved
  - Like 2-coloring

- **EBL (e-beam lithography)**
  - Directly creates features by electron beams w/o mask
  - Excellent resolution
Fabrication of 1D Layout

- Line-end cuts

- Simultaneous DPL and cut redistribution

- Native conflict: Even redistribution plus DPL decomposition cannot solve the conflict. Requires EBL.
Three ways to resolve a conflict

- DPL (coloring)
- Redistribute (move the location)
  cost of wire extension
- Manufacture one cut by EBL
  cost of EBL
Outline

• Background
• Problem Formulation
• ILP
• Graph Model and Algorithm
• Post-processing
• Results and Conclusions
Problem Definition

- Given a layout of \( n \) wires and \( 2n \) cuts, decide the fabrication method (using EBL or not), the mask and the location of each cut, such that
  - All design rules are satisfied.
  - \( wire_{extension} \leq \text{limit} \) for each wire
  - \( \min \sum wire_{extension} + \delta \cdot EBL_{cut}\# \)
Design Rules

- Wires can be extended but not shortened.

- No conflict between two cuts if they are merged.

Merging on the same track (overlap or abut)  Adjacent tracks (aligned)  Non-adjacent tracks ($a, b, c$ aligned)
Outline

• Background
• Problem Formulation
• ILP
• Graph Model and Algorithm
• Post-processing
• Results and Conclusions
ILP

- Existing ILP [DAC’14] solves problem for EBL plus redistribution but no DPL considerations.

- Our contributions:
  - Analyze the potential problems in [DAC’14].
  - Show how to fix the problems.
  - Consider DPL besides EBL and redistribution.

Outline

• Background
• Problem Formulation
• ILP
• **Graph Model and Algorithm**
• Post-processing
• Results and Conclusions
Flow

The conflict graph will change with cut shifting.

1. Input
2. Conflict graph construction
3. Graph splitting
4. Every component is 2-colorable?
   - Yes: Post-processing
   - No: Move selection
5. For each non-2-colorable component
   - Move selection
   - Any move is selected?
     - Yes: Perform the selected moves
     - No: EBL cut selection
What is a “Move”? 

- A move \((c_i, \pm d_i)\)
  - Cut \(c_i\)
  - Right/Left: \(\pm\)
  - Discrete moving distance \(d_i\)
  - Cost = Wire extension resulted

\[(a, -1):\ space a \ and \ b\]
\[(b, +1):\ align b \ and \ c\]
\[(c, +1):\ space b \ and \ c\]
**Move Selection**

- Select moves to change locations of cuts such that no odd cycle is created in conflict graph
  - No odd cycle $\equiv$ 2-colorable

- Select moves based on an integrated graph model
  - Obtained by integrating $G_1$, $G_2$ and $G_3$
Graph Model 1

- $G_1$: constraint graph for move operations:

  - Two moves are incompatible (have an edge) if
    - Exceed limit on wire extension, or
    - Both shift a cut in different directions, or
    - Applying both cannot resolve the targeted conflicts
Graph Model 2

- $G_2$ is a bipartite graph between conflicts and moves:

  - $e$, $d$
  - $a$, $c$
  - $b$

  (a, $-1$): $m_1$  
  (b, $+1$): $m_2$  
  (c, $+1$): $m_3$

  (a, c)  
  (a, b)  
  (b, c)  
  (c, d)

  $m_1$, $m_2$, $m_3$

**e.g.** Moving a to the left 1 step ($m_1$) can resolve the conflict between $(a, c)$ and $m_1$. Thus there is an edge between $(a, c)$ and $m_1$. 
Graph Model 3

- $G_3$: bipartite graph between conflicts and odd cycles

- The edge between $cl_0$ and $(a,c)$ means that resolving conflict $(a,c)$ can break odd cycle $cl_0$.
- All odd cycles should be broken.
- Number of odd cycles can be exponential.
  - Only consider odd cycles in a cycle basis - a set of cycles that can be combined to form every cycle in a graph.
- Constraints between moves (dash lines) are copied.
- $cl_0 - (a,c)$ in $G_3$ and $(a,c) - m_1$ in $G_2$ gives $cl_0 - m_1$ in the final graph meaning that move $m_1$ can break cycle $cl_0$. 
Move Selection by Constrained Set Cover

- Select moves to break all the identified odd cycles
- Constrained set cover problem:
- Select a set of min-cost moves to break all cycles under some constraints.
Solving Constrained Set Cover

- Use ILP to solve the constrained set cover problem:
  - Constant $a_{ij}$ indicates if an edge exists btw. cycle $i$ and move $m_j$.
  - Variable $b_j$ indicates if move $m_j$ is selected.

\[
\text{min. } \sum_{j=1}^{\#\text{move}} b_j \cdot \text{cost}(m_j),
\]

\[
\text{s.t. } \sum_{j=1}^{\#\text{move}} a_{ij} \cdot b_j \geq 1, \quad \forall \text{ cycle},
\]

\[
b_i + b_j \leq 1, \quad \forall \text{ move } m_i \text{ incompatible with } m_j
\]

- Much smaller and simpler than the ILP solving the original problem directly.

\[
\begin{align*}
\text{e.g. } & c_{l0} \\
\text{min. } & \sum_{j=1}^{3} b_j \cdot \text{cost}(m_j) \\
& b_1 + b_2 + b_3 \geq 1 \\
& b_2 + b_3 \leq 1
\end{align*}
\]
When a cut is an EBL cut, its corresponding node is deleted from the conflict graph.

**Problem:** Delete a minimum number of nodes from the conflict graph such that at least one node will be deleted from each cycle in a cycle basis.

**Solution:** Use a similar ILP without incompatible constraints.

---

**Flow (Review)**

```plaintext
do
    Select some moves.
    Perform selected moves.
    Rebuild conflict graph.
until all cuts 2-colorable or no moves are available.

Select some cuts as EBL cuts.
```
Accelerating by Potential Conflict Graph

- **Conflict graph** $G$ of cuts:
  Conflicts between cuts can change dynamically if cuts can move.

- **Potential conflict** graph $G_p$.
  An edge between two nodes iff there is a potential conflict between the two cuts with cut redistribution.

- $G_p$ is stable and can be safely split into sub-layouts to reduce problem size.
Handling Vertically Aligned Cuts

- \#cut=2: \(a\) and \(b\) never conflict
  - As we can merge them if \(\text{color}(a) = \text{color}(b)\)

- \#cut=3: no conflict if \(\text{color}(b) = \text{color}(a)\) or \(\text{color}(c)\)
  - Conflict edge \(a\rightarrow c\): unnecessary

- \# vertically aligned cuts = \(n \leq H+1\):
  \((H\) is the largest difference between two conflicting track labels.\)

**Lemma:** No conflict iff \(\exists i\) for \(2 \leq i \leq n\) s.t. \(c_1\ldots c_{i-1}\) are colored the same and \(c_i\ldots c_n\) are colored the same.

- Grouping nodes instead of adding many edges in conflict graph
DPL+EBL

• Background
• Problem Formulation
• ILP
• Graph Model and Algorithm
• Post-processing
• Results and Conclusions
Post-processing methods

- Objective: To minimize wire extensions.

- Globally: Longest-path algorithm:
  - Compact the cuts at right ends of wires to the left
  - Compact the cuts at left ends of wires to the right

- Locally: Greedily shift cuts towards their original locations
Longest Path algorithm

- For those right end cuts, construct a left compaction graph:

- Edge cost:
  - $s$ to node: Leftmost $x$ of the movable range of the cut.
  - Between nodes: Required distance between the 2 nodes.

- Edge direction:
  - $b$ to $a$ iff $x_a > x_b$

- Distance of the longest path from $s$ to $i$:
  - Leftmost $x$ to place $i$
Outline

• Background
• Problem Formulation
• ILP
• Graph Model and Algorithm
• Post-processing
• Results and Conclusions
### Comparison with ILP

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Optimal ILP</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Track#</td>
<td>EBL#</td>
<td>Extension</td>
</tr>
<tr>
<td>50</td>
<td>0</td>
<td>26</td>
</tr>
<tr>
<td>100</td>
<td>0</td>
<td>46</td>
</tr>
<tr>
<td>150</td>
<td>0</td>
<td>78</td>
</tr>
<tr>
<td>200</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>250</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>300</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>1000</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>2000</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>4000</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>8000</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

- ILP is too slow.
- Our EBL# has achieved lower bound for all datasets.
- Our quality is very close to ILP if ILP has solutions.
Comparison with Optimal Coloring + Optimal Redistribution

Our algorithm optimizes coloring and redistribution simultaneously

\[ \text{Cost} = \sum \text{wire_extension} + \hat{\partial} \cdot \text{EBL_cut\#} \quad \hat{\partial}=100 \]
Conclusion

• Co-optimization of cut redistribution and mask assignment for 1D gridded design.

• Novel graph-theoretic method that makes use of integrated graph model + longest path-based refinement

• 1D design is the future of 10nm technology node and beyond and more research can be done.