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

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

The Chinese University of Hong Kong

香港中文大學

# 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

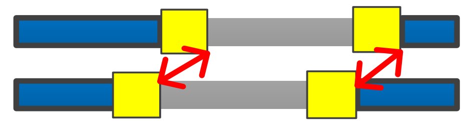
- EBL (e-beam lithography)
  - Directly creates features by electron beams w/o mask
  - Excellent resolution

# Fabrication of 1D Layout

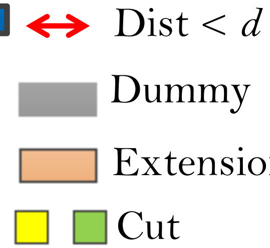
- Line-end cuts

Target

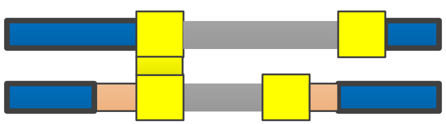
Dense lines



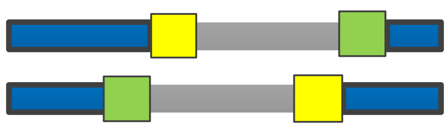
Line-end cuts



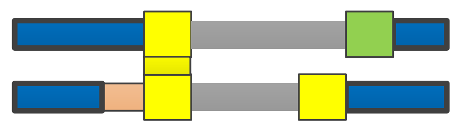

- Simultaneous DPL and cut redistribution



Redistribution

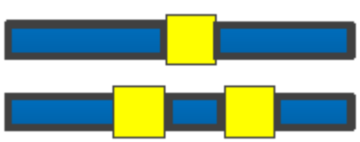


DPL of cuts



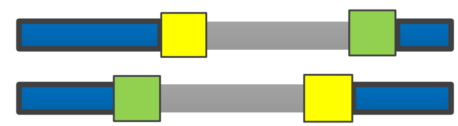
Redistribution+DPL

- Native conflict: Even redistribution plus DPL decomposition cannot solve the conflict. **Requires EBL.**



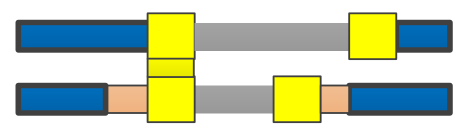
# Resolving Conflicts between Cuts

- Three ways to resolve a conflict
  - DPL (coloring)



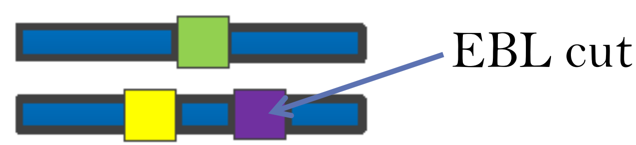
  - Redistribute (move the location)

    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

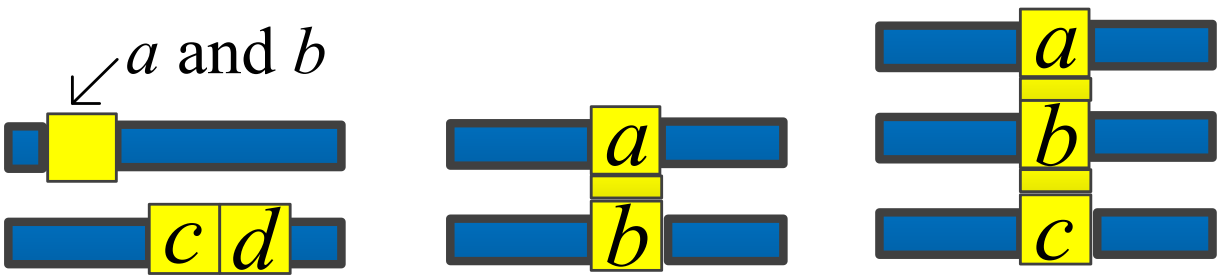
# Problem Definition

- 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.
  - $wire\_extension \leq$ limit for each wire
  - min. $\sum wire\_extension + \partial \cdot EBL\_cut\#$

# 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

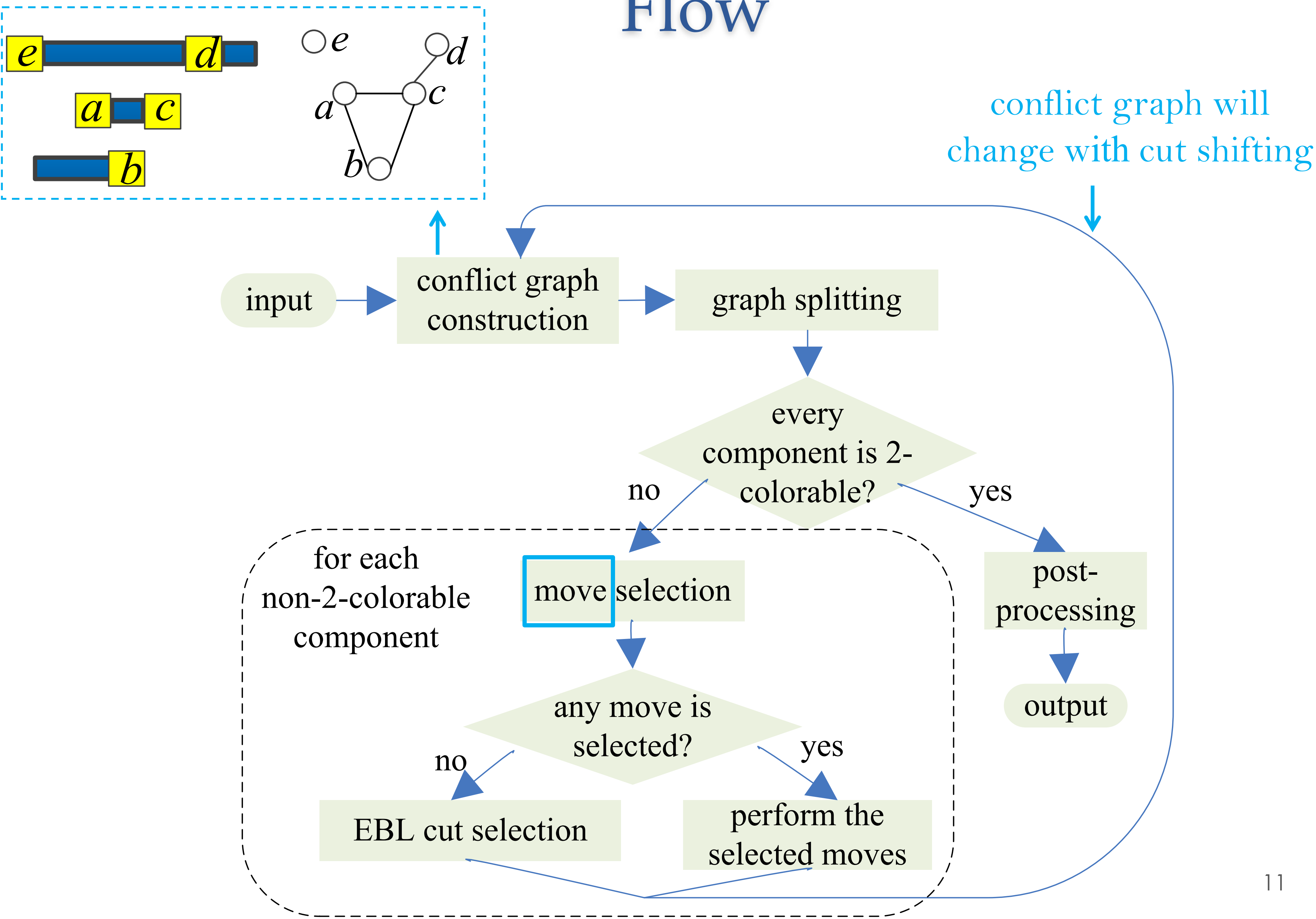
- Existing ILP [DAC'14] solves problem for EBL plus redistribution but no DPL considerations.
- Our contributions:
  - Analyze the potential problems in [DAC'14].
  - Show how to fix the problems.
  - Consider DPL besides EBL and redistribution.

[DAC'14] Ding et al. "Throughput optimization for SADP and e-beam based manufacturing of 1D layout", In Proc. DAC, 2014

# Outline

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

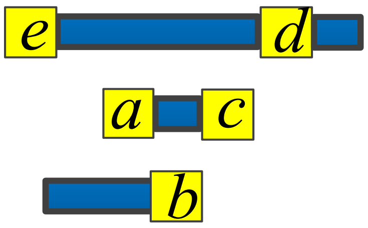
# Flow



e
d
a
c
b
e
d
c
a
b
conflict graph will change with cut shifting
input
conflict graph construction
graph splitting
every component is 2-colorable?
no
yes
for each non-2-colorable component
move selection
post-processing
output
any move is selected?
no
yes
EBL cut selection
perform the selected moves

# 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 $c$

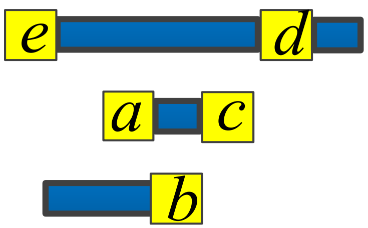
# Move Selection

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

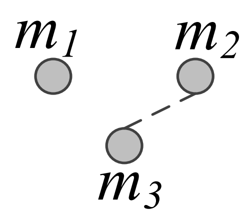

- Select moves based on an integrated graph model
  - Obtained by integrating $G_1$, $G_2$ and $G_3$

# Graph Model 1

- $G_1$: constraint graph for move operations:



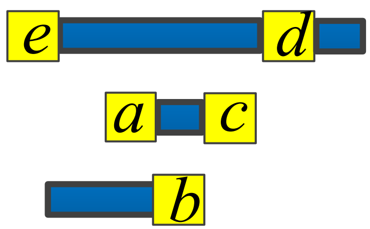

$$(a,-1):\ m_1$$
$$(b,+1):\ m_2$$
$$(c,+1):\ m_3$$



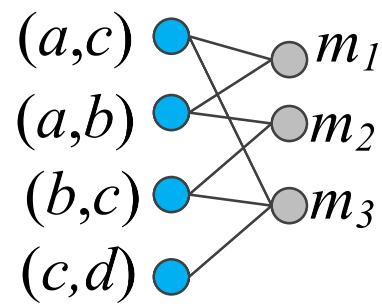

- Two moves are incompatible (have an edge) if
  - Exceed limit on wire extension, or
  - Both shift a cut in different directions, or
  - Applying both cannot resolve the targeted conflicts

# Graph Model 2

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



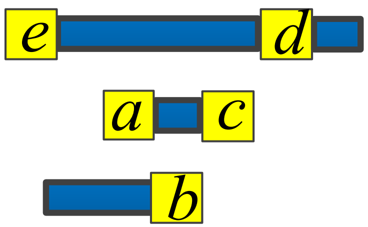

$$(a,-1): \quad m_1$$
$$(b,+1): \quad m_2$$
$$(c,+1): \quad m_3$$

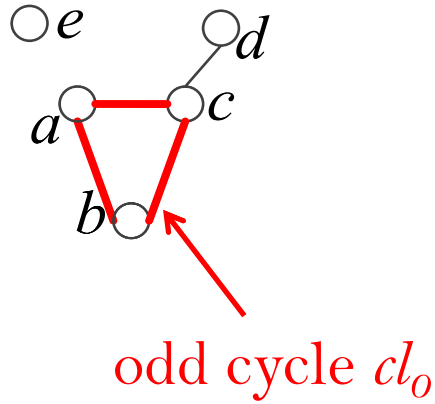


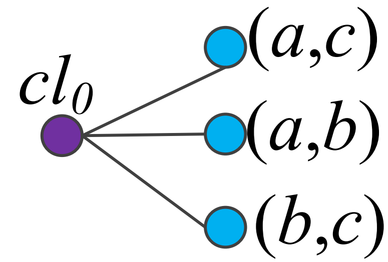

**e.g**. Moving a to the left 1 step ($m_1$) can resolve the conflict between $(a,c)$ and $m_1$. Thus there is an edge between $(a,c)$ and $m_1$.

# Graph Model 3

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

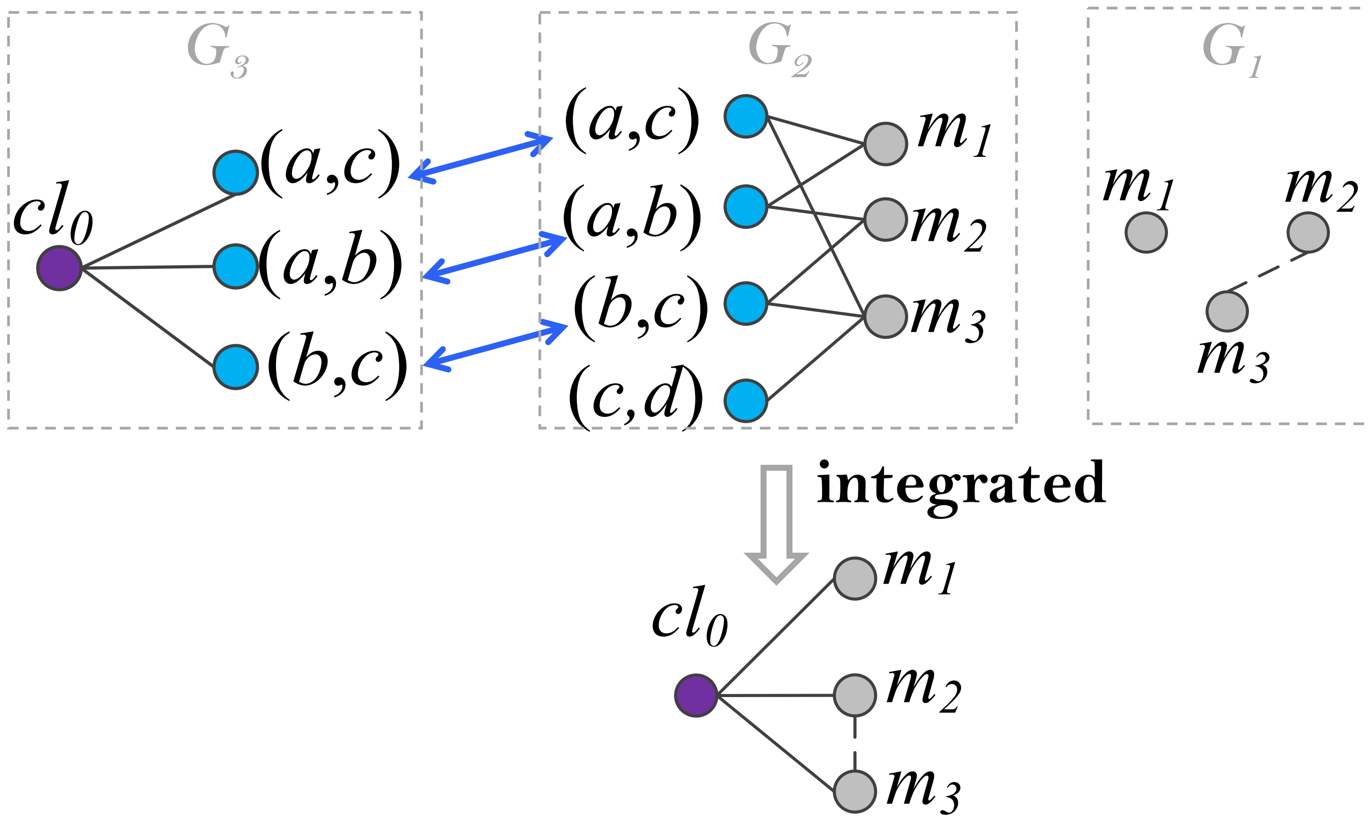



- The edge between $cl_0$ and $(a,c)$ means that resolving conflict $(a,c)$ can break odd cycle $cl_0$.
- All odd cycles should be broken.
- Number of odd cycles can be exponential.
  - Only consider odd cycles in a cycle basis - a set of cycles that can be combined to form every cycle in a graph.

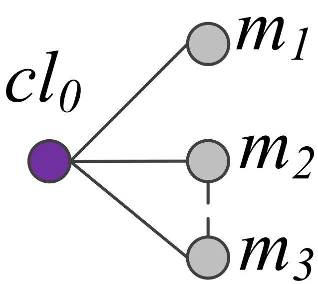
# Integrated Graph Model




- 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

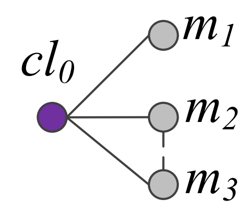
- Use ILP to solve the constrained set cover problem:
  - Constant $a_{ij}$ indicates if an edge exists btw. cycle $i$ and move $m_j$
  - Variable $b_j$ indicates if move $m_j$ is selected

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

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

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

**e.g.**




$$\min. \sum_{j=1}^{3} b_j \cdot cost(m_j)$$

$$b_1 + b_2 + b_3 \geq 1$$

$$b_2 + b_3 \leq 1$$

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

# EBL Cut Selection

| Flow (Review) |
|---|
| **do**<br>Select some moves.<br>Perform selected moves.<br>Rebuild conflict graph.<br>**until** all cuts 2-colorable **or** no moves are available.<br><br>Select some cuts as EBL cuts. |

- 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

# Accelerating by Potential Conflict Graph

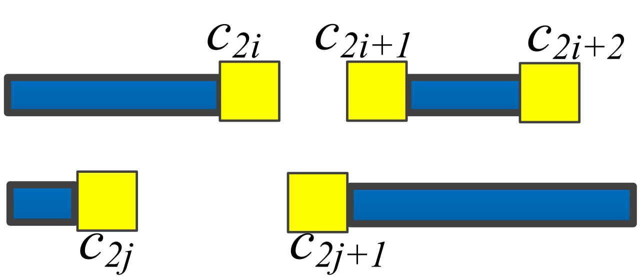
- Conflict graph $G$ of cuts:

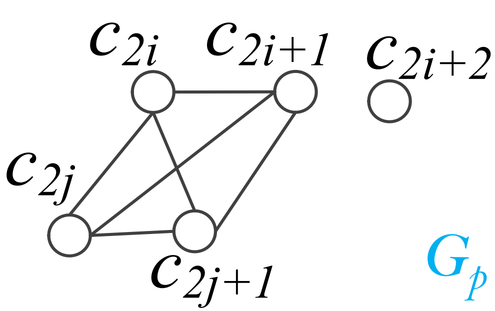
  Conflicts between cuts can change dynamically if cuts can move.

- Potential conflict graph $G_p$.

  An edge between two nodes iff there is a potential conflict between the two cuts with cut redistribution

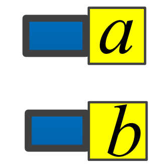
- $G_p$ is stable and can be safely split into sub-layouts to reduce problem size



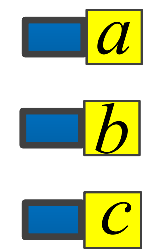



# Handling Vertically Aligned Cuts

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

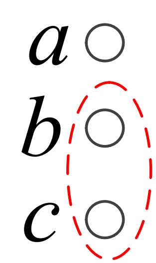



- #cut=3: no conflict if color($b$) = color($a$) or color($c$)
  - Conflict edge $a$—$c$ : unnecessary




- # vertically aligned cuts = $n \le H+1$:

  ($H$ is the largest difference between two conflicting track labels.)
- **Lemma**: No conflict iff $\exists i$ for $2 \le i \le n$ 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

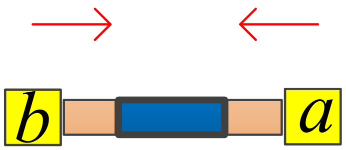


# DPL+EBL

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

# Post-processing methods

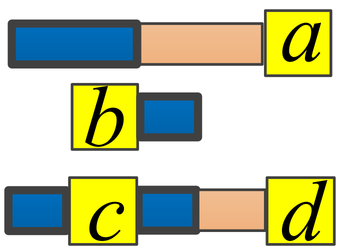
- Objective: To minimize wire extensions.
- Globally: Longest-path algorithm:
  - Compact the cuts at right ends of wires to the left
  - Compact the cuts at left ends of wires to the right

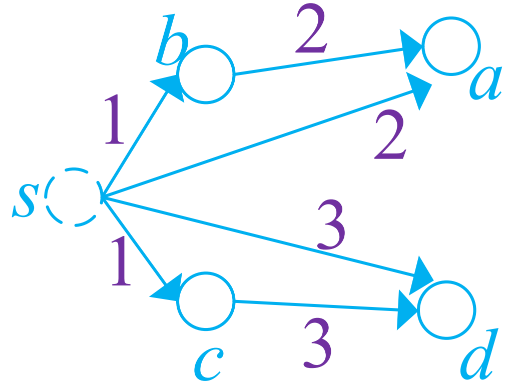



- Locally: Greedily shift cuts towards their original locations

# 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

| Dataset | Optimal ILP | | | Ours | | |
|---|---|---|---|---|---|---|
| Track# | EBL# | Extension | Time(s) | EBL# | Extension | Time(s) |
| 50 | 0 | 26 | 18.0 | 0 | 26 | 0.1 |
| 100 | 0 | 46 | 180.4 | 0 | 46 | 0.2 |
| 150 | 0 | 78 | 6446.9 | 0 | 79 | 0.4 |
| 200 | - | - | >36000 | 0 | 104 | 0.5 |
| 250 | - | - | >36000 | 0 | 129 | 0.6 |
| 300 | - | - | >36000 | 0 | 164 | 0.7 |
| 1000 | - | - | >36000 | 0 | 583 | 2.2 |
| 2000 | - | - | >36000 | 0 | 1230 | 4.5 |
| 4000 | - | - | >36000 | 1 | 2500 | 9.4 |
| 8000 | - | - | >36000 | 1 | 5178 | 18.8 |

Caused by native conflict

- 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

| Dataset | Opt. color + redistribute | | Ours | |
|---|---|---|---|---|
| Track# | Cost | Time (s) | Cost | Time (s) |
| 50 | 4029 | 0.7 | 26 | 0.1 |
| 100 | 9050 | 1.2 | 46 | 0.2 |
| 150 | 14091 | 2.4 | 79 | 0.4 |
| 200 | 17109 | 2.8 | 104 | 0.5 |
| 250 | 19135 | 3.4 | 129 | 0.6 |
| 300 | 23174 | 4.7 | 164 | 0.7 |
| 1000 | 69670 | 35.6 | 583 | 2.2 |
| 2000 | 132380 | 91.4 | 1230 | 4.5 |
| 4000 | 280764 | 245.3 | 3500 | 9.4 |
| 8000 | 573740 | 2784.9 | 6178 | 18.8 |
| Ratio | 95.0 | 84.8 | 1 | 1 |

- Our algorithm optimizes coloring and redistribution simultaneously
- Cost $= \sum \mathit{wire\_extension} + \partial \cdot \text{EBL\_cut\#}$ $\quad \partial = 100$

# 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.