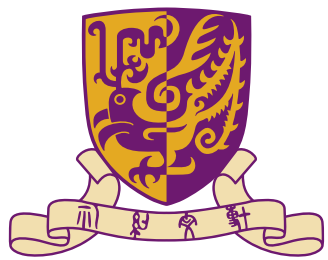
MrDP: Multiple-row Detailed Placement of Heterogeneous-sized Cells for Advanced Nodes

Yibo Lin[1] **Bei Yu**[2] Xiaoqing Xu[1] Jhih-Rong Gao[3]
Natarajan Viswanathan[3] Wen-Hao Liu[3] Zhuo Li[3]
Charles J. Alpert[3] David Z. Pan[1]

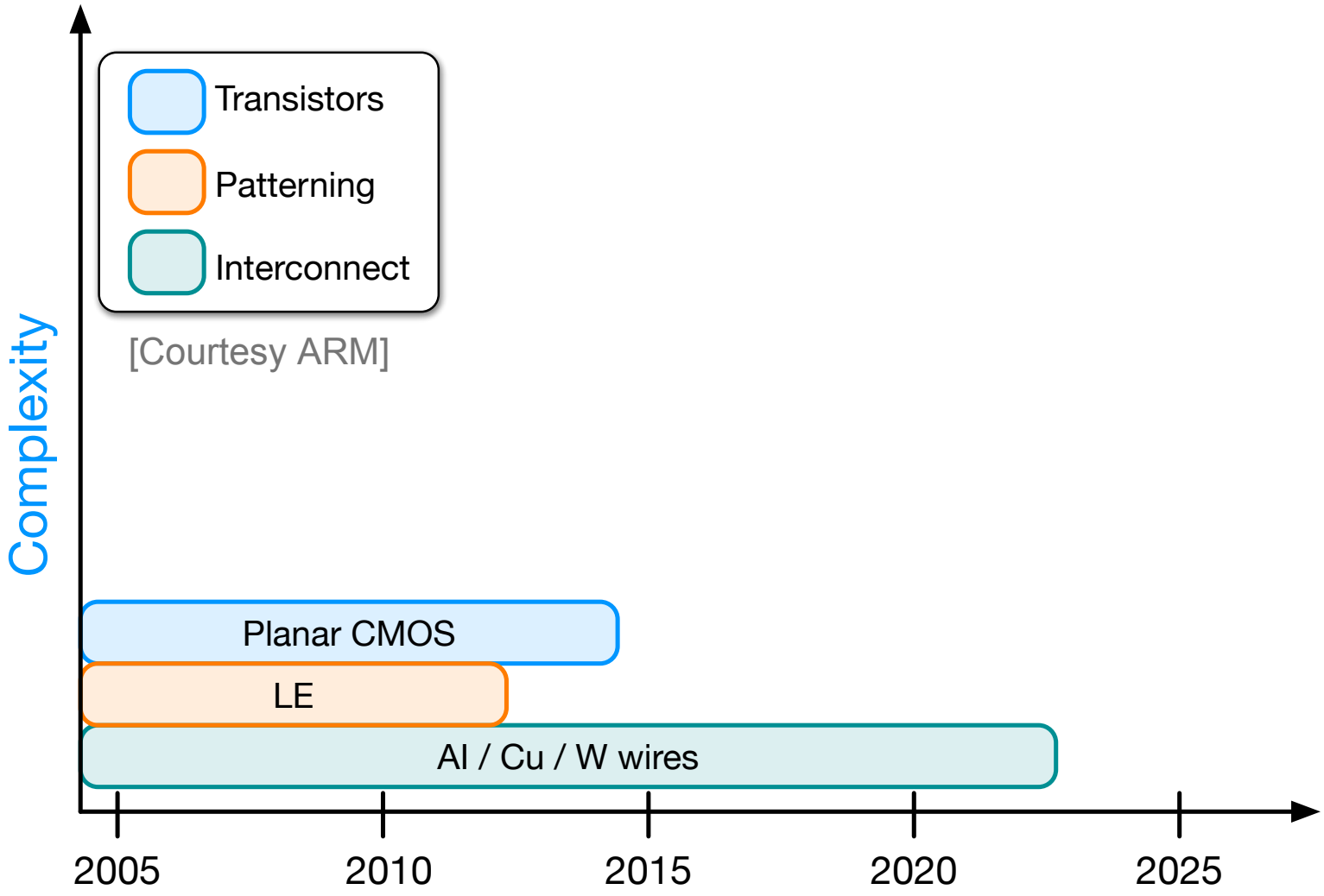
[1]University of Texas at Austin
[2]The Chinese University of Hong Kong
[3]Cadence Design Systems



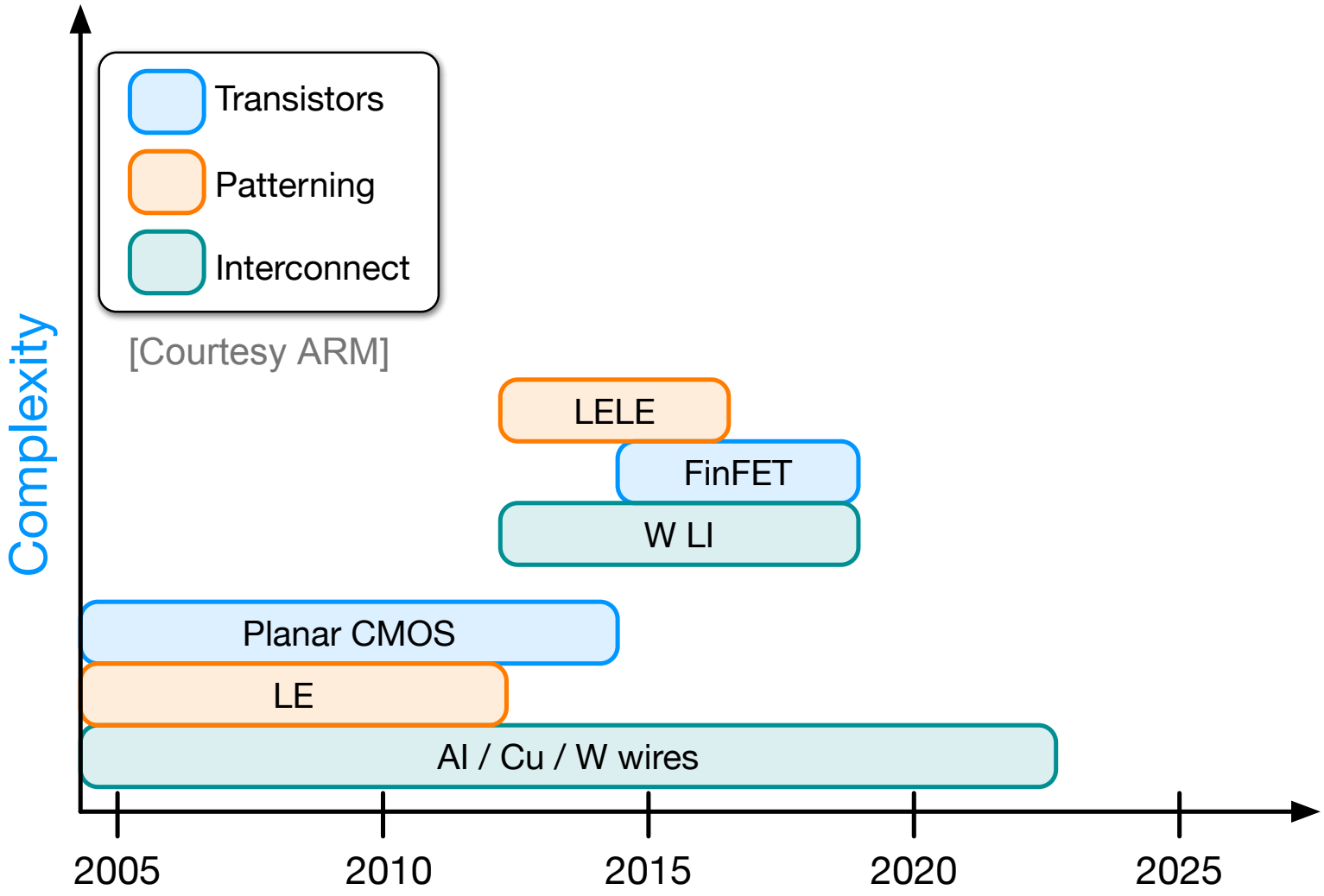
cādence

Outline

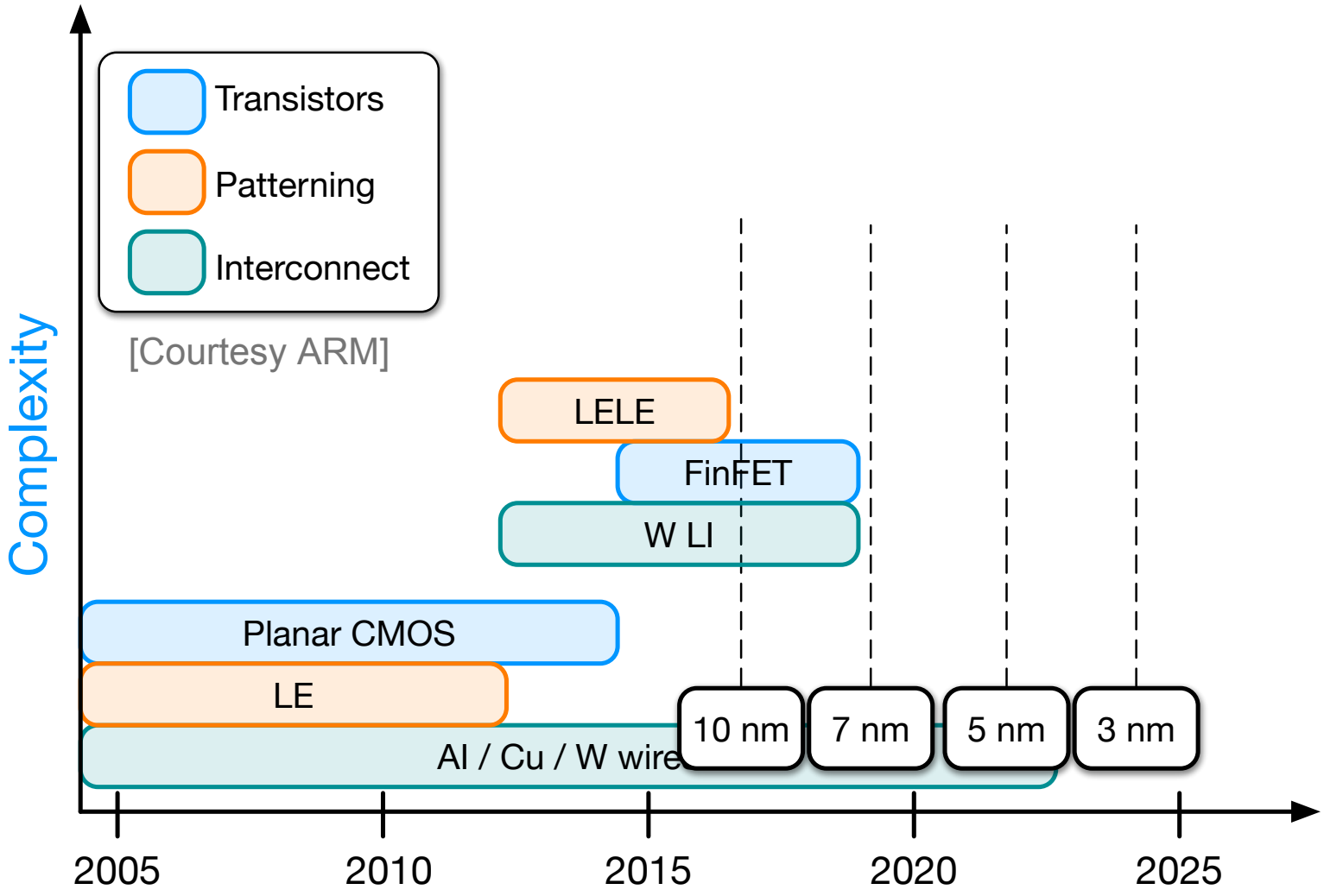
Introduction: Technology Scaling



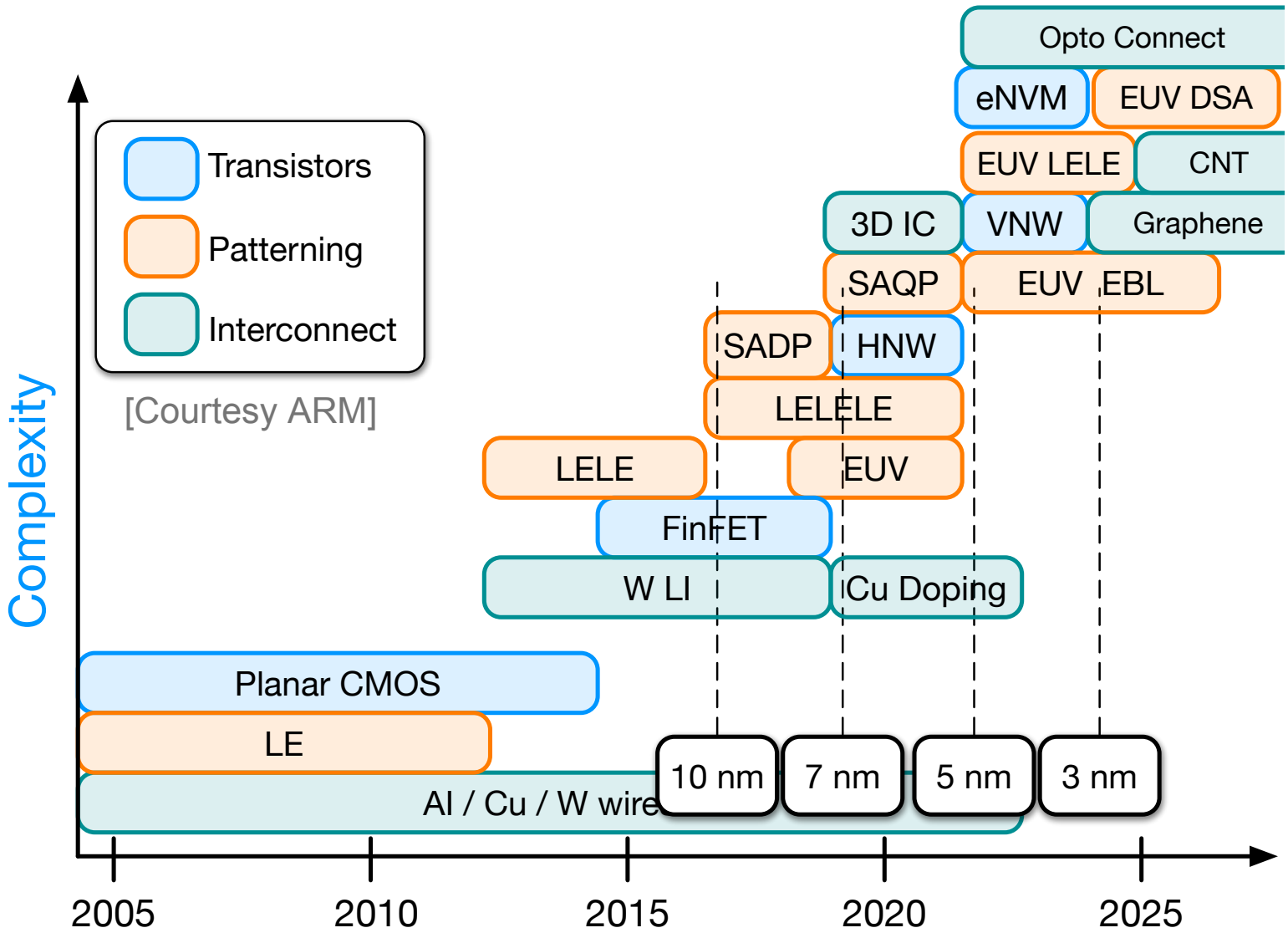
Introduction: Technology Scaling



Introduction: Technology Scaling



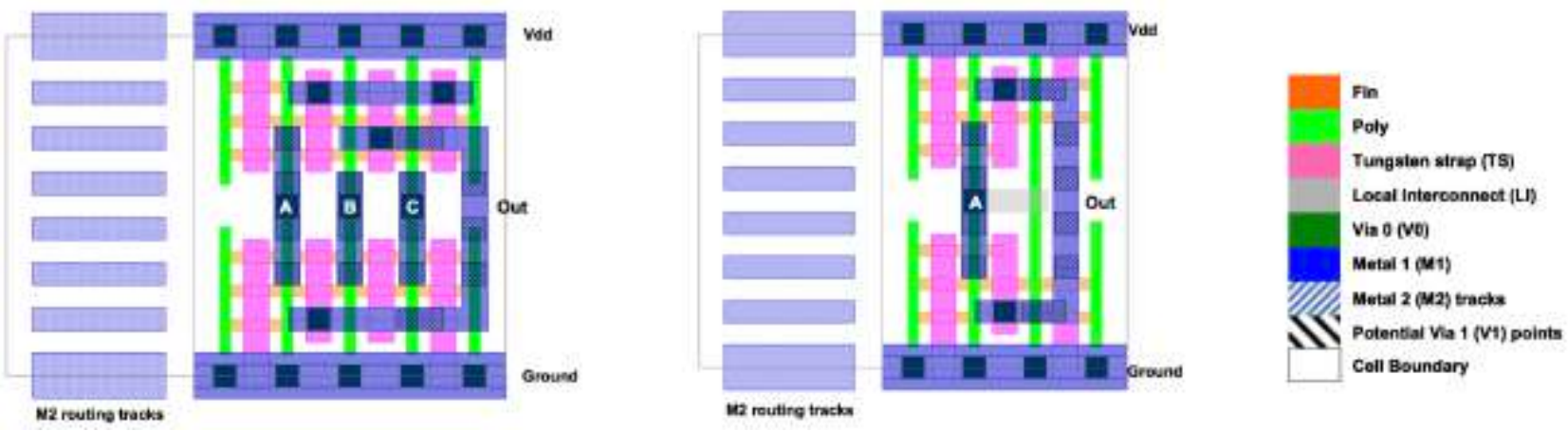
Introduction: Technology Scaling



Technology Scaling: Fewer Tracks

Track # per row decreases:

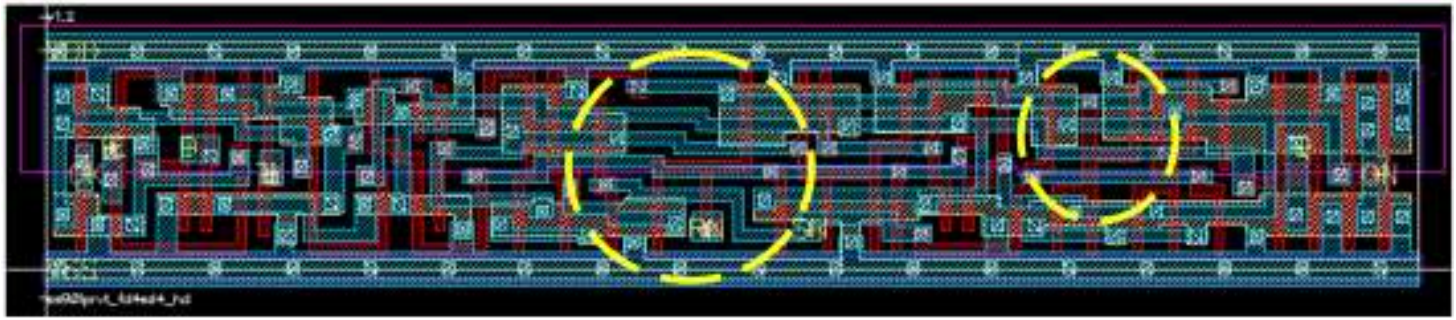
- From 10 to 7.5
- Exploring 7.5T for 7nm technology node
- Even with EUV, additional metal layer may be required



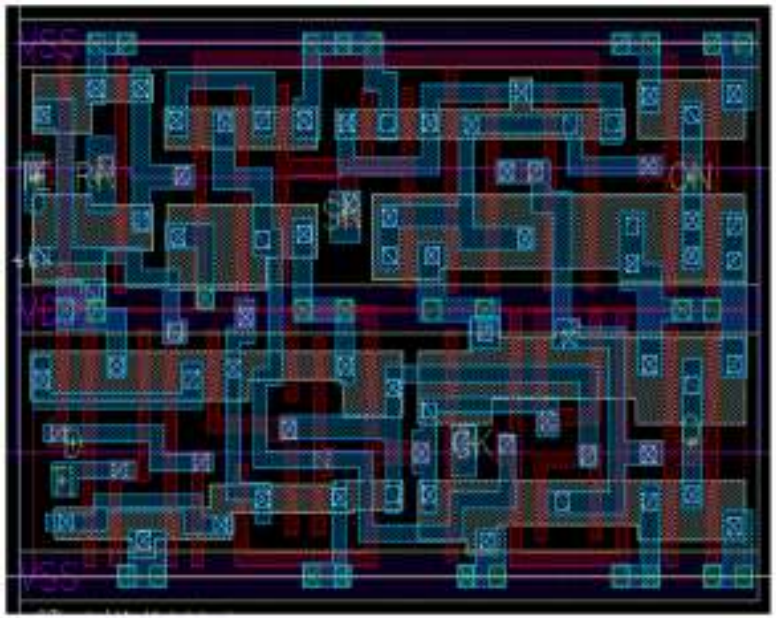
(a) And-or-invert (AOI); (b) 2-finger inverter [Liebman+,SPIE'15].

Motivation of Multiple-Row Cells 1

- Complex standard cells, such as flip-flops, MUXes, etc.
- Intra-Cell Routability



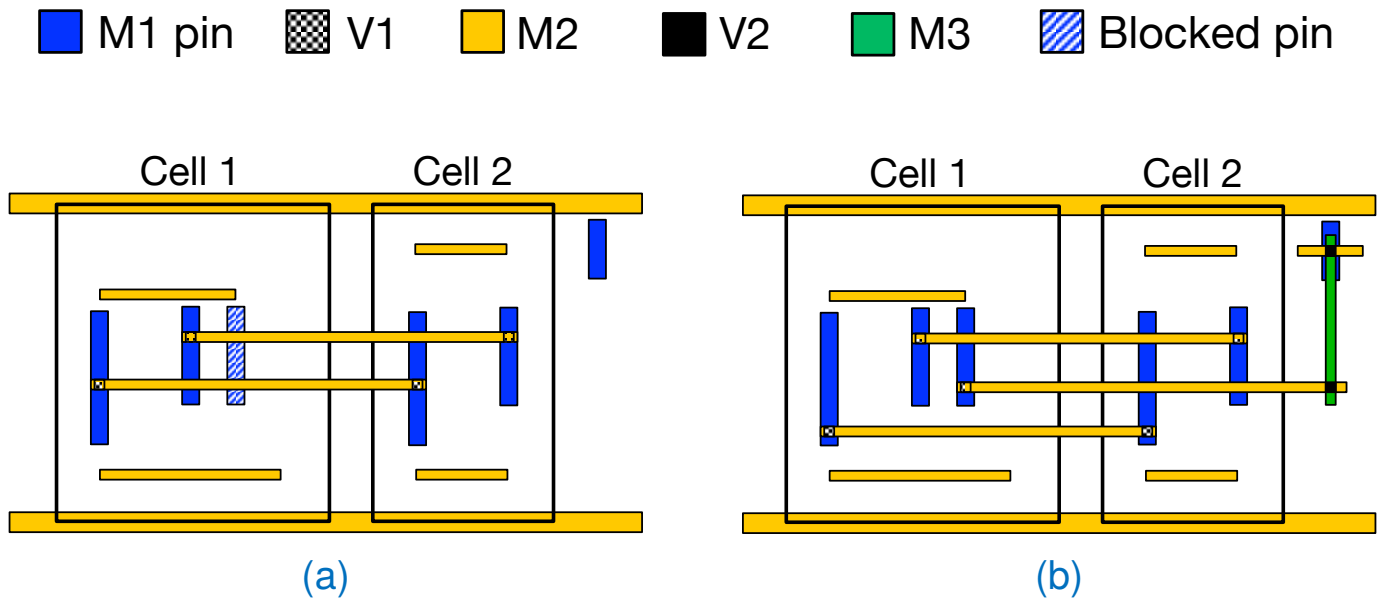
(a) Cell size 54 grids



(b) Cell size 48 grids

Motivation of Multiple-Row Cells 2

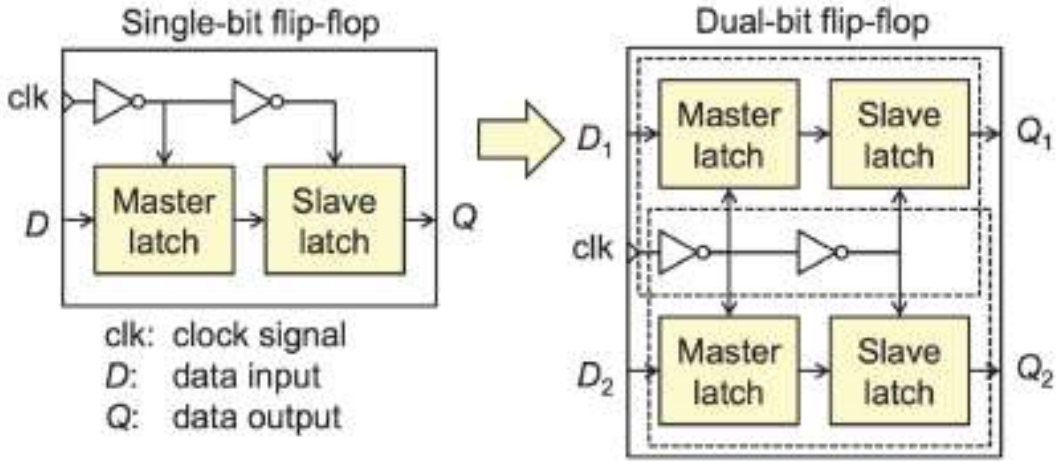
Pin access problem [Taghavi+,ICCAD'10]



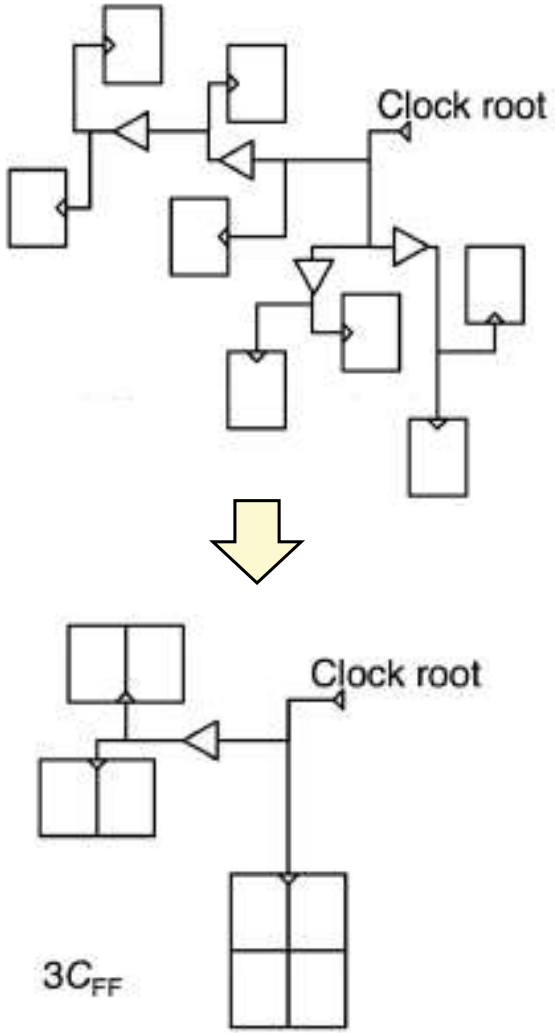
(a) pin access failure; (b) pin access success. [Xu+,DAC'14]

Motivation of Multiple-Row Cells 3

Multi-bit flip-flops (MBFF)



[Jiang+,ISPD'11]



[Pokala+,ASIC'92]

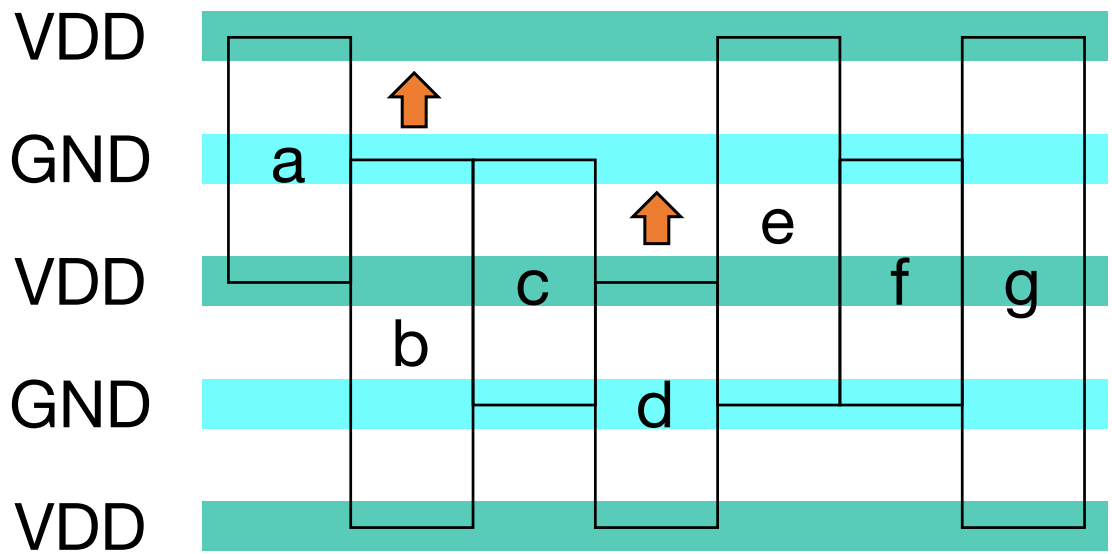
Power Line Alignment

Odd-row height cells

- Misalignment fixable with vertical flipping

Even-row height cells

- Misalignment NOT fixable with vertical flipping
- New placement techniques are highly necessary



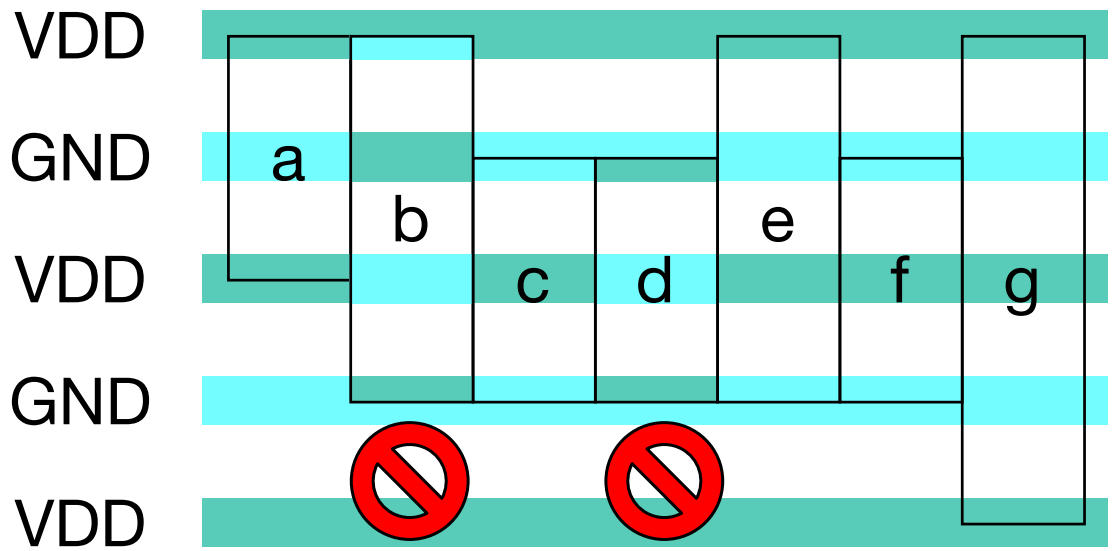
Power Line Alignment

Odd-row height cells

- Misalignment fixable with vertical flipping

Even-row height cells

- Misalignment NOT fixable with vertical flipping
- New placement techniques are highly necessary



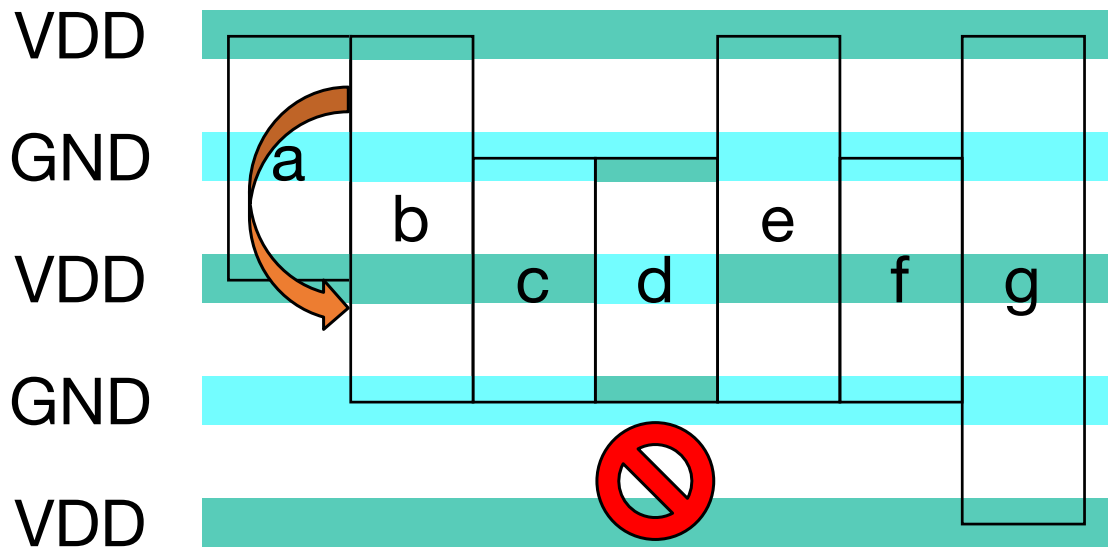
Power Line Alignment

Odd-row height cells

- Misalignment fixable with vertical flipping

Even-row height cells

- Misalignment NOT fixable with vertical flipping
- New placement techniques are highly necessary



Previous Works

Double-row height cells [Wu+,TCAD'15]

- Group and extend single-row height cells into double-row height blocks
- Re-use existing detailed placement frameworks
- Incapable to handle three- and four-row height cells
- Power alignment not addressed

Legalization for Multiple-row height cells [Chow+,DAC'16]

- General to heterogeneous-sized cells
- Minimize total displacement while removing overlaps
- Power alignment addressed
- No performance optimization

Wirelength and Density Metrics

Cell Density: **ABU** [ICCAD'13 Contest]

$$\text{overflow}_\gamma = \max\left(0, \frac{\text{ABU}_\gamma}{d_t} - 1\right)$$

$$\text{ABU} = \frac{\sum_{\gamma \in \Gamma} w_\gamma \cdot \text{overflow}_\gamma}{\sum_{\gamma \in \Gamma} w_\gamma}, \Gamma \in \{2, 5, 10, 20\}$$

Scaled wirelength (sHPWL)

$$\text{sHPWL} = \text{HPWL} \cdot (1 + \text{ABU})$$

Wirelength and Density Metrics

Cell Density: **ABU** [ICCAD'13 Contest]

$$\text{overflow}_\gamma = \max\left(0, \frac{\text{ABU}_\gamma}{d_t} - 1\right)$$

$$\text{ABU} = \frac{\sum_{\gamma \in \Gamma} w_\gamma \cdot \text{overflow}_\gamma}{\sum_{\gamma \in \Gamma} w_\gamma}, \Gamma \in \{2, 5, 10, 20\}$$

Scaled wirelength (sHPWL)

$$\text{sHPWL} = \text{HPWL} \cdot (1 + \text{ABU})$$

APU

Average Pin Utilization: capture pin distribution of the layout.

Problem Formulation: MrDP

Multi-row Detailed Placement (MrDP)

Input:

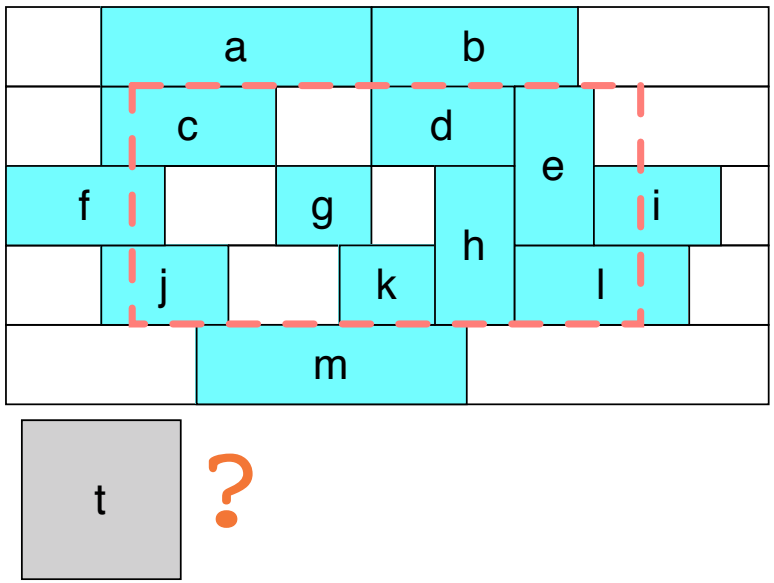
- A netlist with heterogeneous-sized cells
- Initial placement with fixed macro blocks

Output:

- Legal placement
- Minimize wirelength and density cost, i.e., **sHPWL** and **APU**

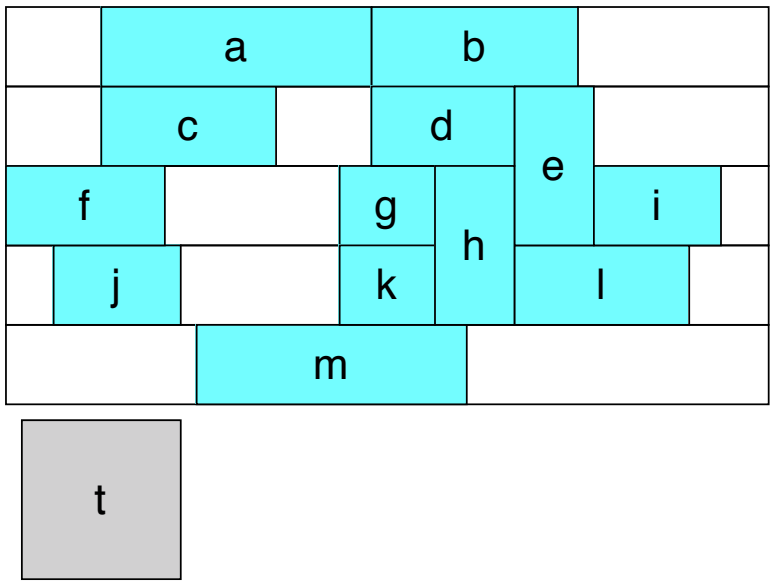
Conventional Global Move

- Pick a cell and move to better position
- More difficult with heterogeneous-sized cells



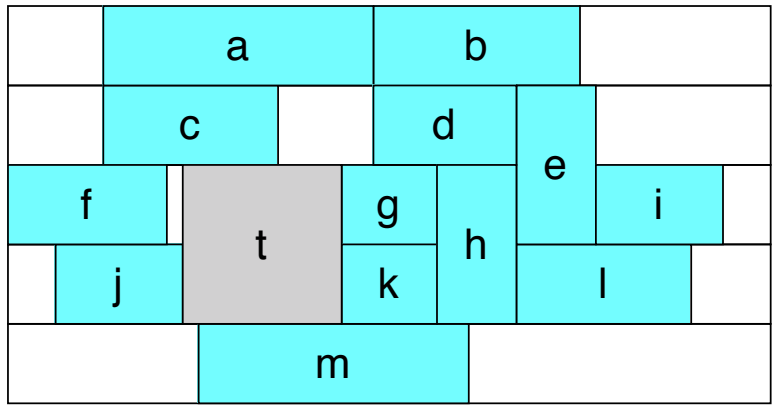
Conventional Global Move

- Pick a cell and move to better position
- More difficult with heterogeneous-sized cells



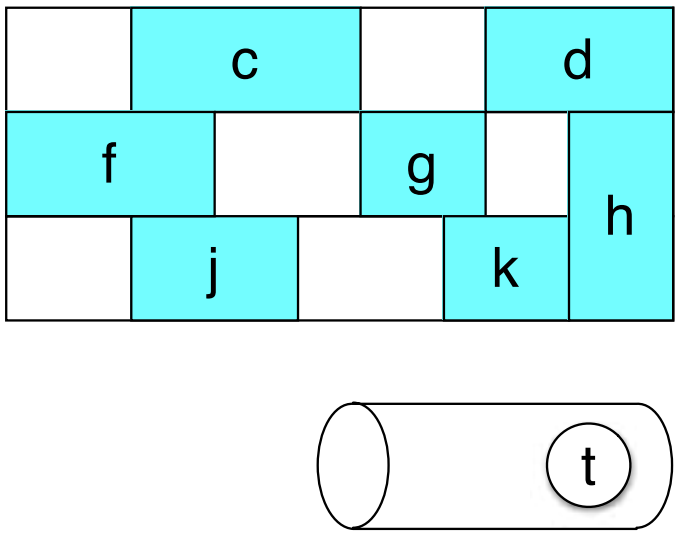
Conventional Global Move

- Pick a cell and move to better position
- More difficult with heterogeneous-sized cells



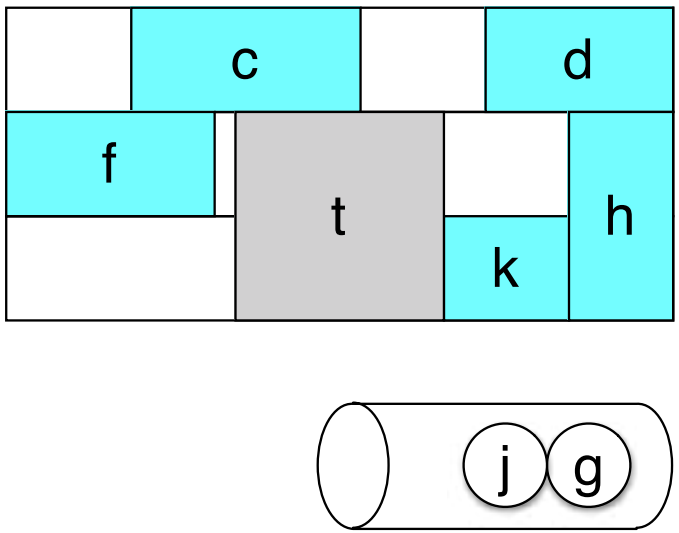
Chain Move

- **Cell Pool**:
  A queue structure used for temporary storage of cells within a chain move
- **Scoreboard**:
  Consists of an array of chain move entries with corresponding changes in wirelength cost for each chain move
- Inspired by KL and FM algorithms in partitioning [KL'70][FM,DAC'82]
- Look for cumulatively good cost



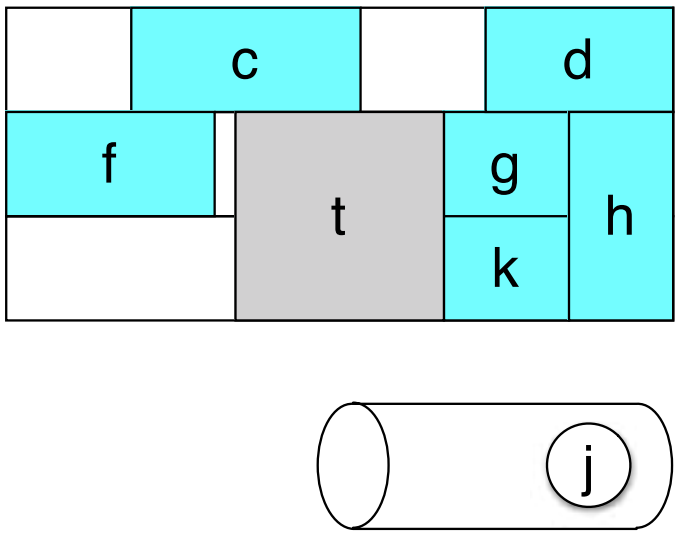
Chain Move

- **Cell Pool**:
  A queue structure used for temporary storage of cells within a chain move
- **Scoreboard**:
  Consists of an array of chain move entries with corresponding changes in wirelength cost for each chain move
- Inspired by KL and FM algorithms in partitioning [KL'70][FM,DAC'82]
- Look for cumulatively good cost



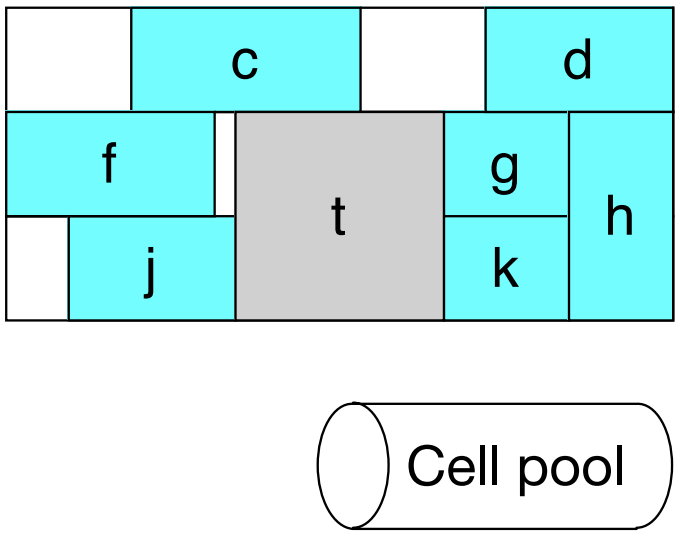
Chain Move

- **Cell Pool**:
  A queue structure used for temporary storage of cells within a chain move
- **Scoreboard**:
  Consists of an array of chain move entries with corresponding changes in wirelength cost for each chain move
- Inspired by KL and FM algorithms in partitioning [KL'70][FM,DAC'82]
- Look for cumulatively good cost



Chain Move

- **Cell Pool**:
  A queue structure used for temporary storage of cells within a chain move
- **Scoreboard**:
  Consists of an array of chain move entries with corresponding changes in wirelength cost for each chain move
- Inspired by KL and FM algorithms in partitioning [KL'70][FM,DAC'82]
- Look for cumulatively good cost



Scoreboard
⋮
Chain move entry
$\begin{pmatrix}\text{Cell } t\text{: } p_1^0 \to p_1\\ \text{Cell } g\text{: } p_2^0 \to p_2\\ \text{Cell } j\text{: } p_3^0 \to p_3\end{pmatrix}$, ΔWL
⋮

Chain Move Discussion

- Order is important
- Max prefix sum of wirelength improvement
- Discard long chains

Cost for a Cell:

$$cost = \Delta \mathrm{WL} \cdot (1 + \alpha \cdot c_d) + \beta \cdot c_{ov}$$

- Δ WL: wirelength cost
- c_d: density cost (average of cell and pin densities)
- c_{ov}: overlap cost

Chain Move Discussion

- Order is important
- Max prefix sum of wirelength improvement
- Discard long chains

Cost for a Cell:

$$cost = \Delta \mathrm{WL} \cdot (1 + \alpha \cdot c_d) + \beta \cdot c_{ov}$$

- Δ WL: wirelength cost
- c_d: density cost (average of cell and pin densities)
- c_{ov}: overlap cost

Theorem

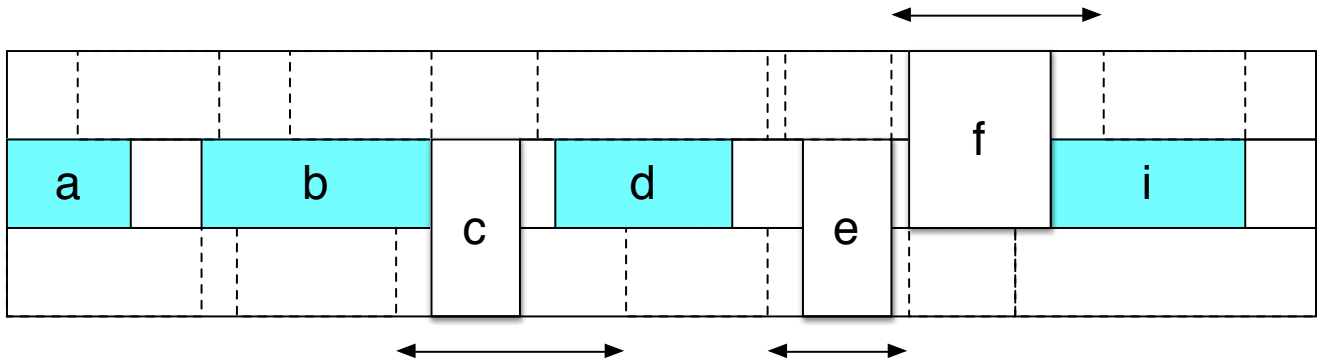
If the input is legal, then the output is guaranteed legal

Ordered Single-Row (OSR) Placement

Well explored for single-row height cells

- Free-to-move [Vygen,DATE'98] [Kahng+,ASPDAC'99]
- Max displacement [Taghavi+,ICCAD'10] [Lin+,ASPDAC'16]

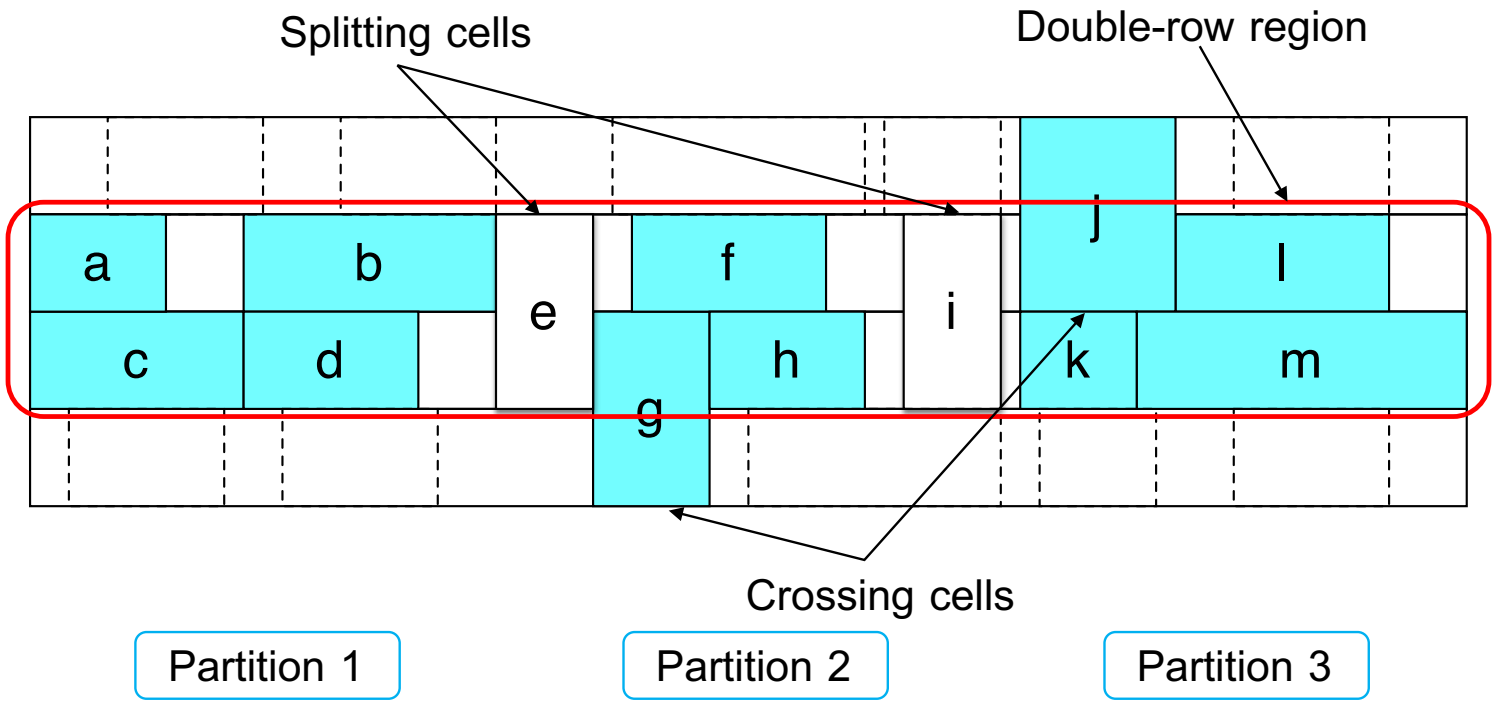
How to deal with multiple-row height cells?



Limited movements by multiple rows.

Ordered Double-Row (ODR) Placement

- Extend single-row to double-row placement
- Some definitions



Problem Formulation: ODR Placement

Ordered Double-Row (ODR) Placement

Input:

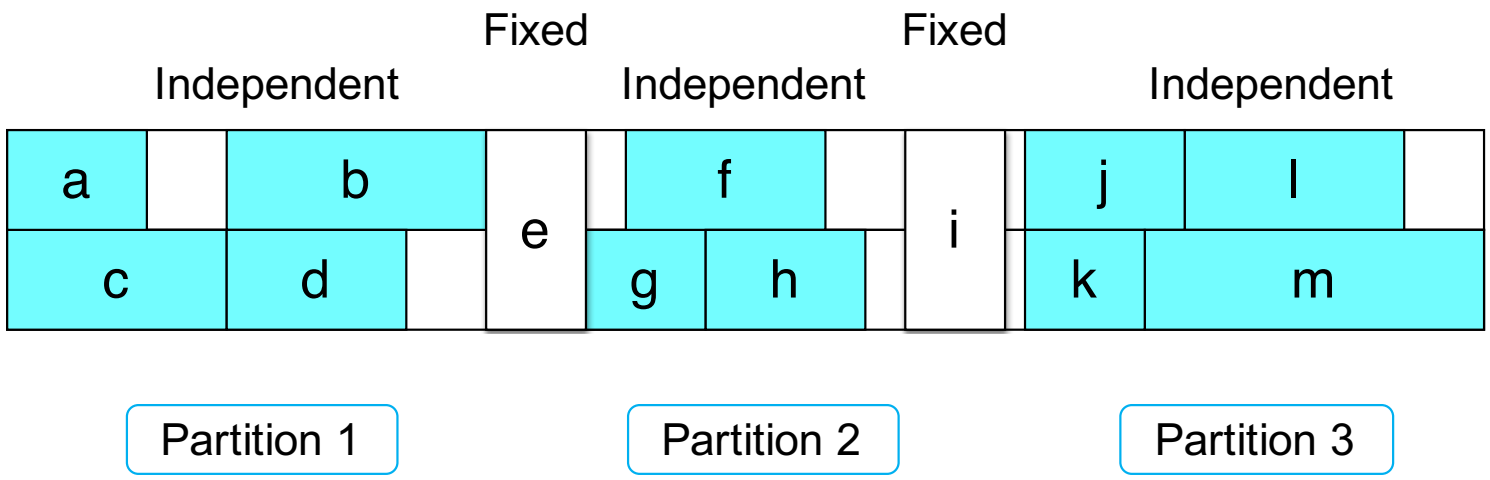
- Two rows of cells in a double-row region
- Ordered from left to right within each row
- Maximum displacement M for each cell
- All other cells outside double-row region are fixed

Output:

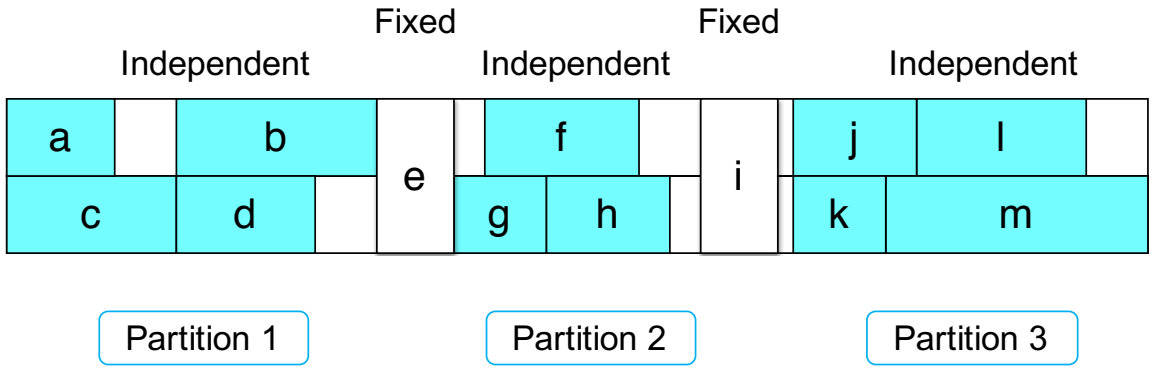
- Horizontally shift cells
- Optimize HPWL while keep the order of cells within each row

ODR Placement: Ideal Cases

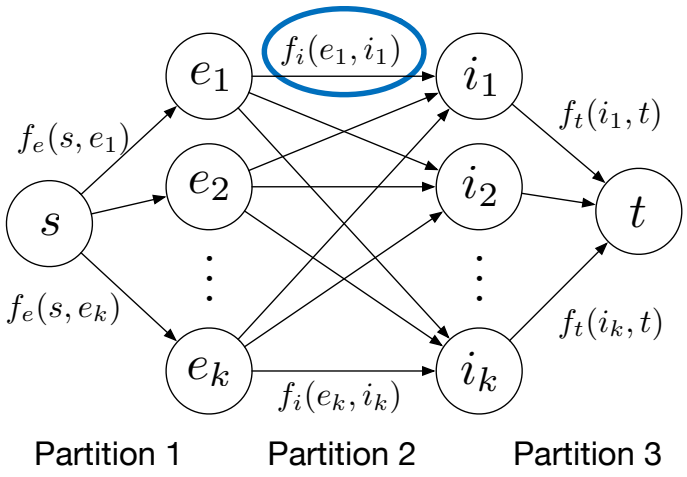
- Only double-row splitting cells
- No crossing cells
- No inter-row connection within double-row region
- Solve ideal case **optimally**



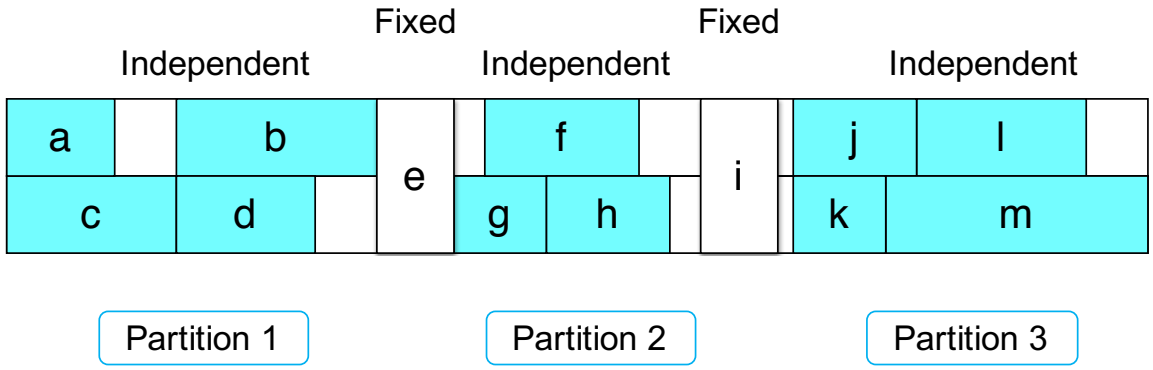
Nested Dynamic Programming

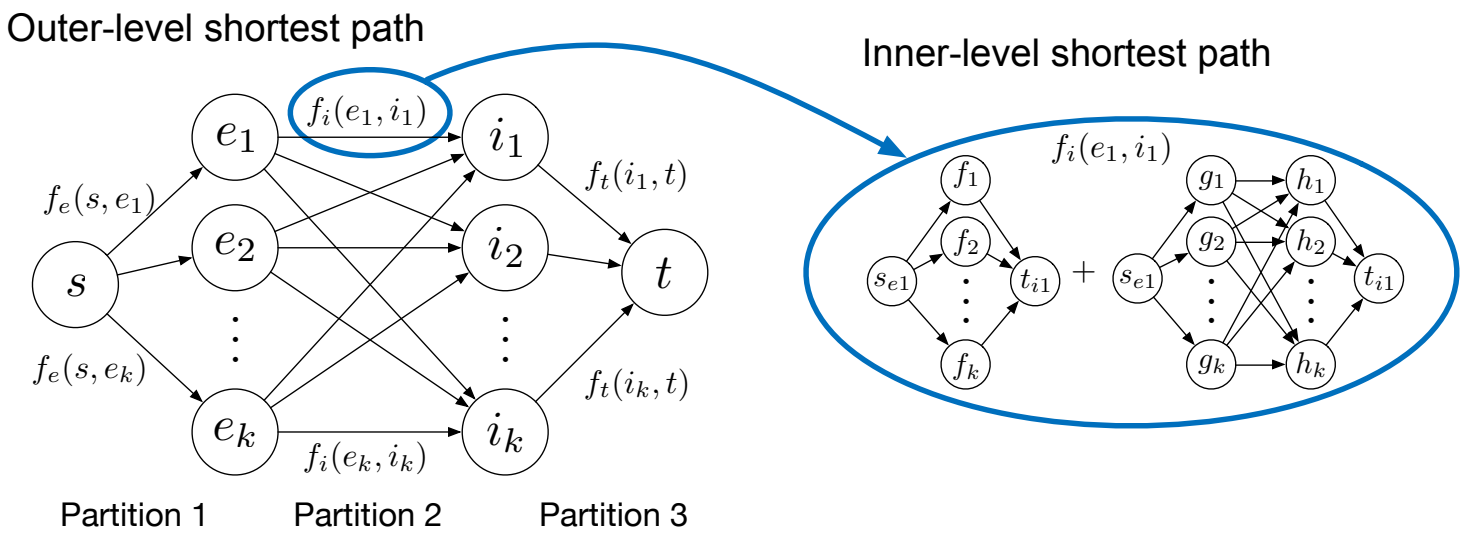


Outer-level shortest path



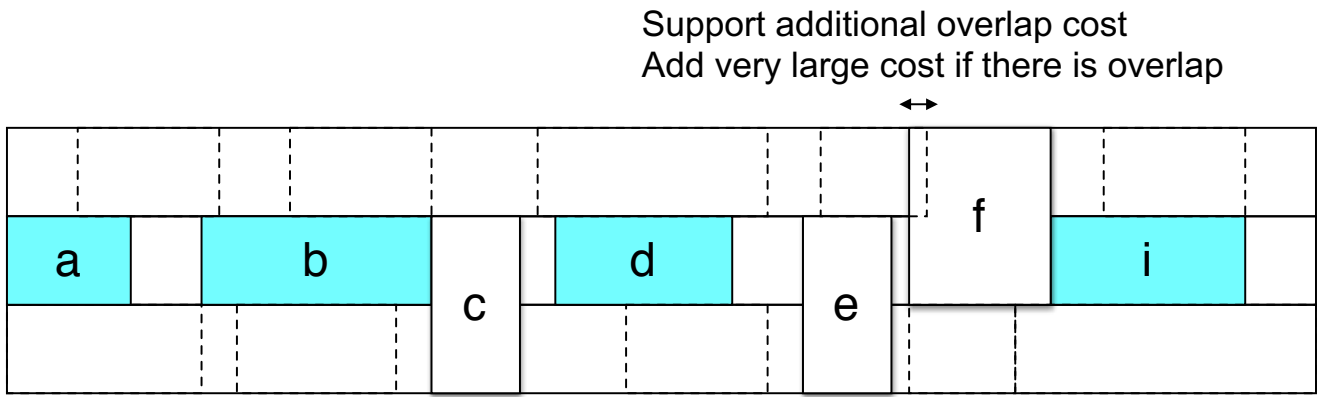
Nested Dynamic Programming





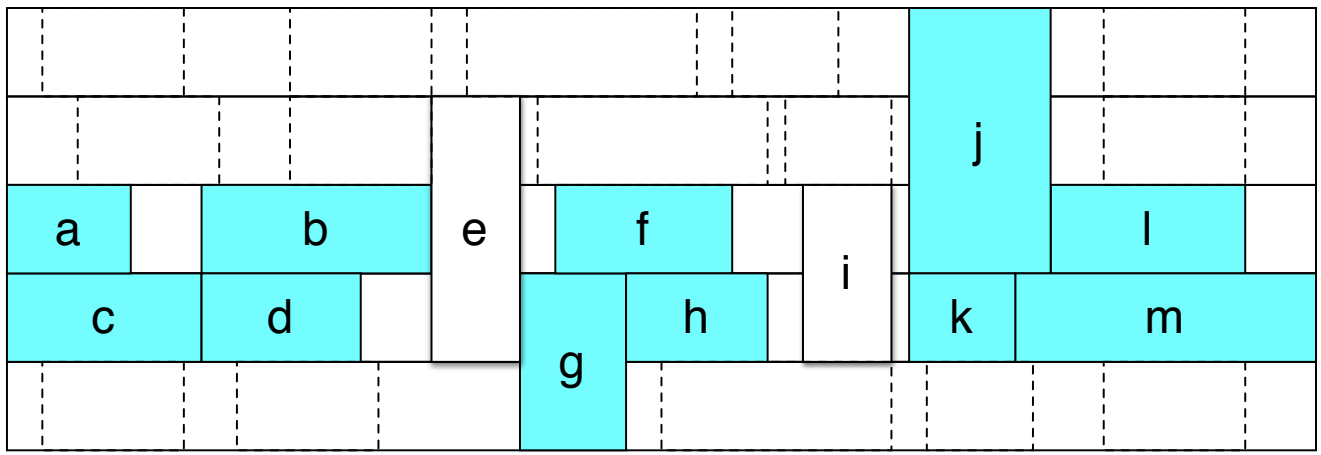
Nested Dynamic Programming

- Any shortest path algorithm can be applied
- Adopt dynamic programming [Lin+,ASPDAC'16]
- $\mathcal{O}(nM)$ for single-row placement
- $\mathcal{O}(nM^2)$ for double-row placement
- Flexible to any cost that only depends on cell itself

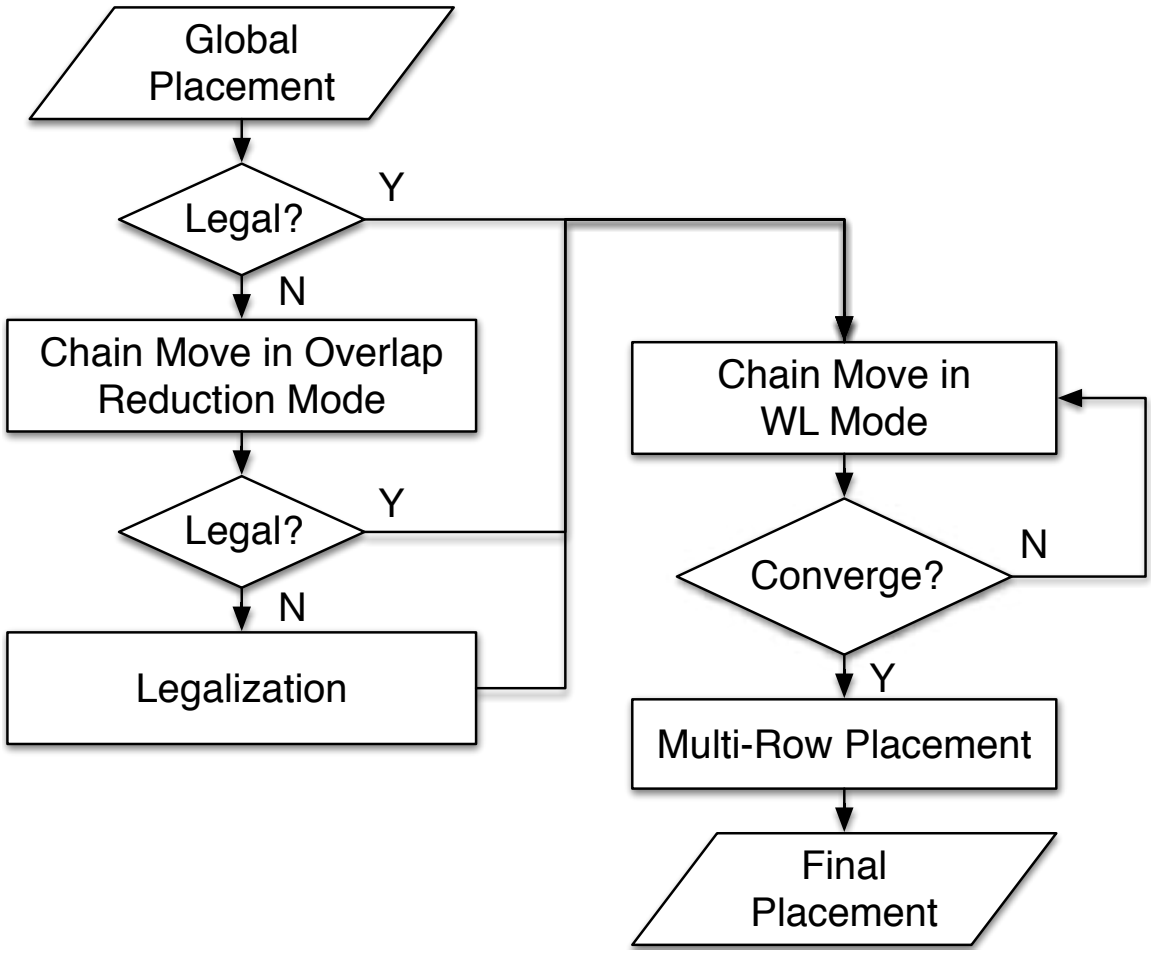


ODR Placement: General Cases

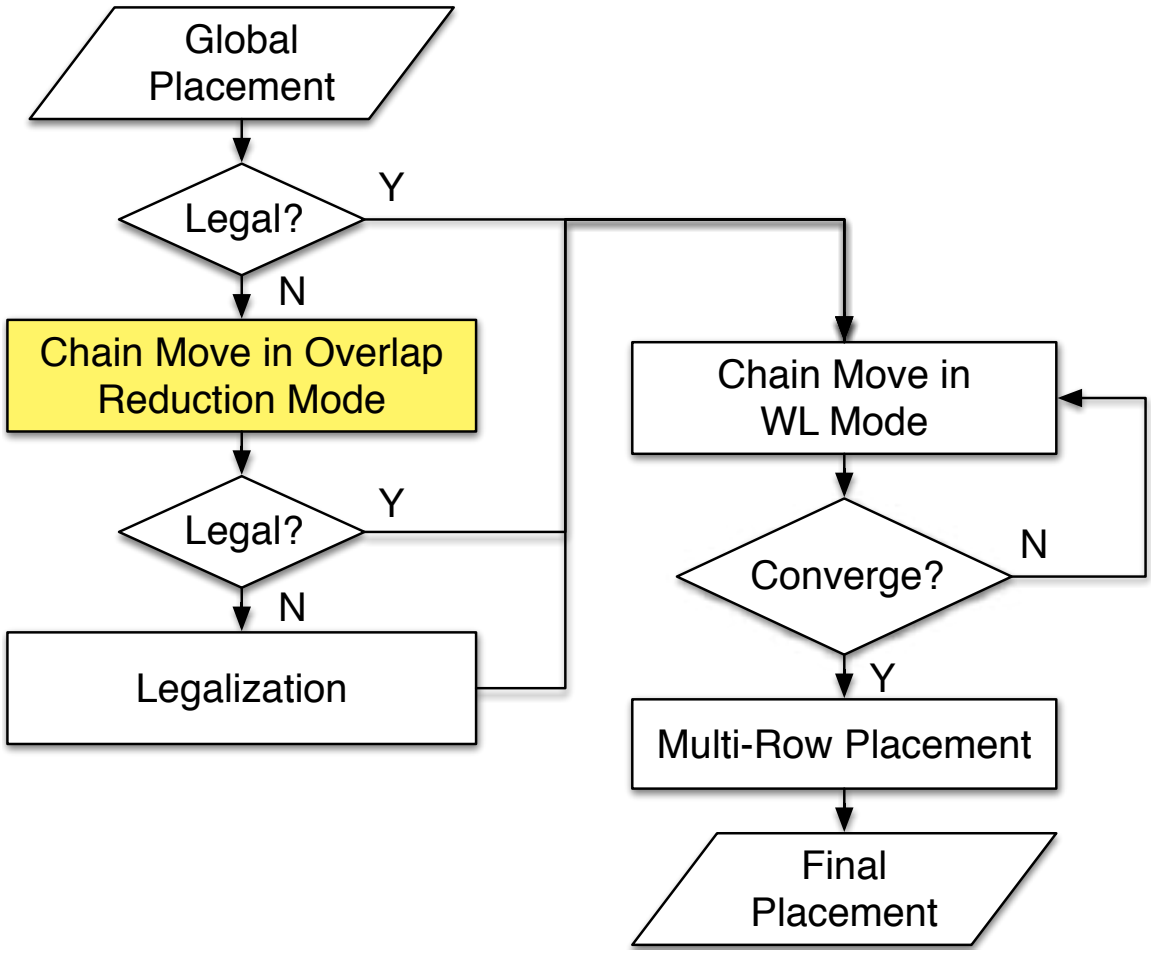
- Multiple-row height splitting cells
- Multiple-row height crossing cells: Add overlap cost
- Inter-row connections within double-row region: Lose optimality



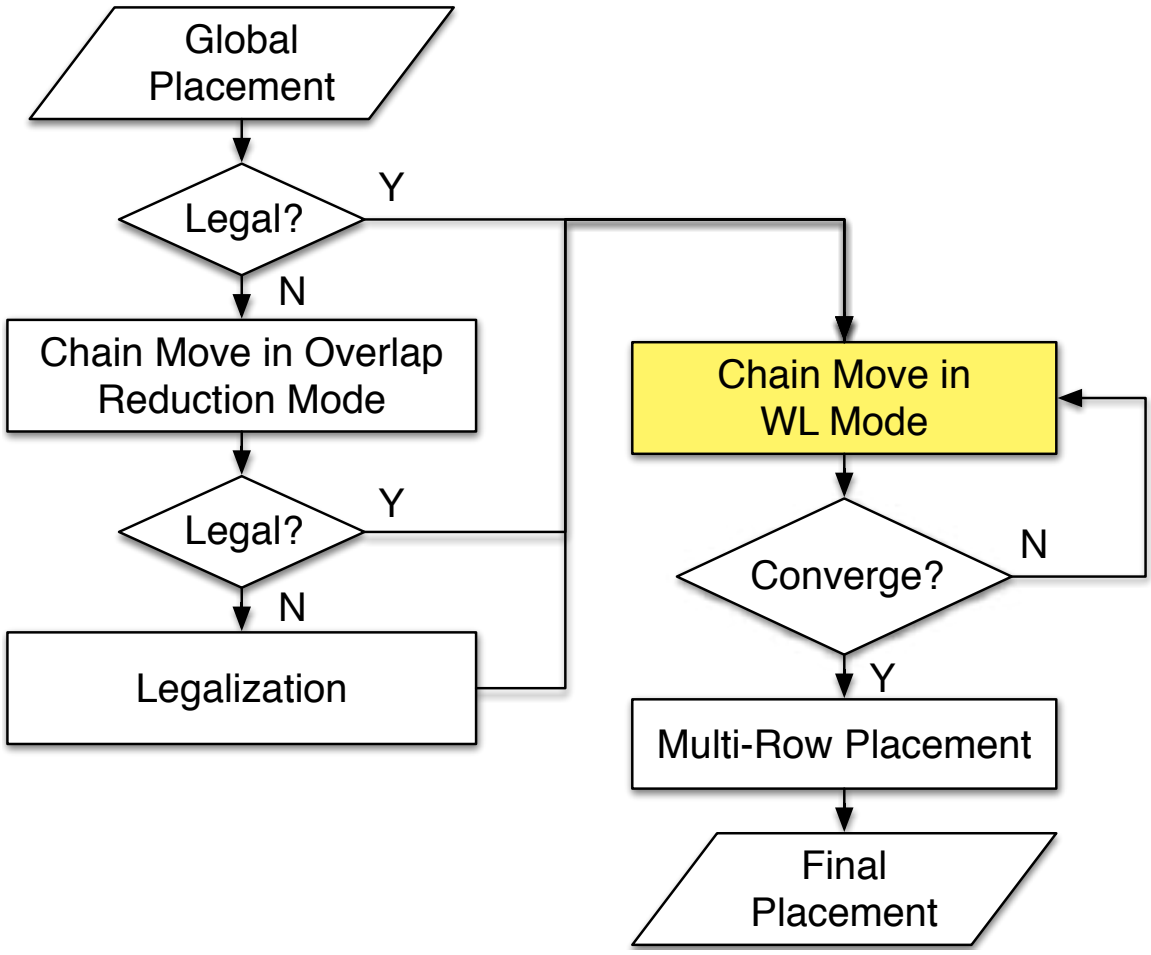
Overall Flow



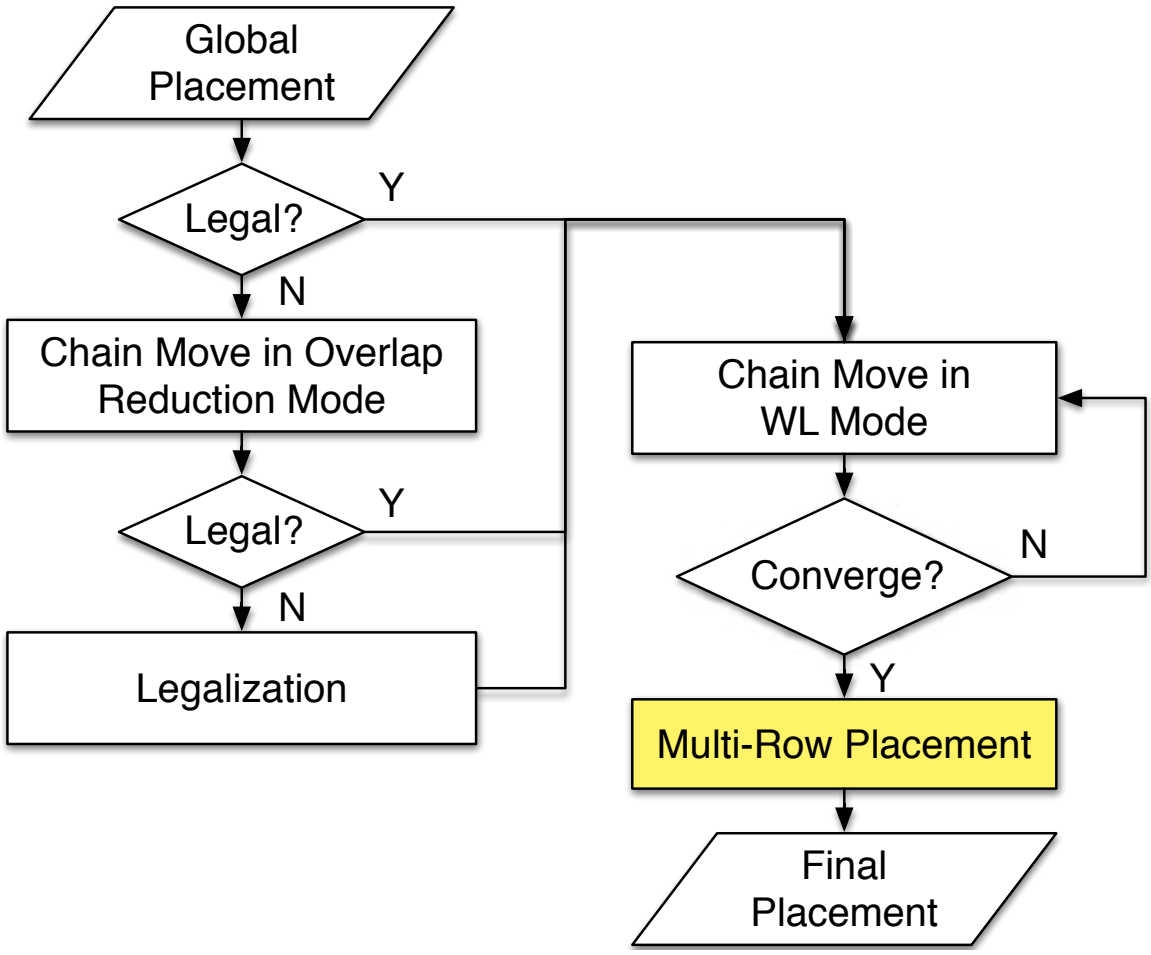
Overall Flow

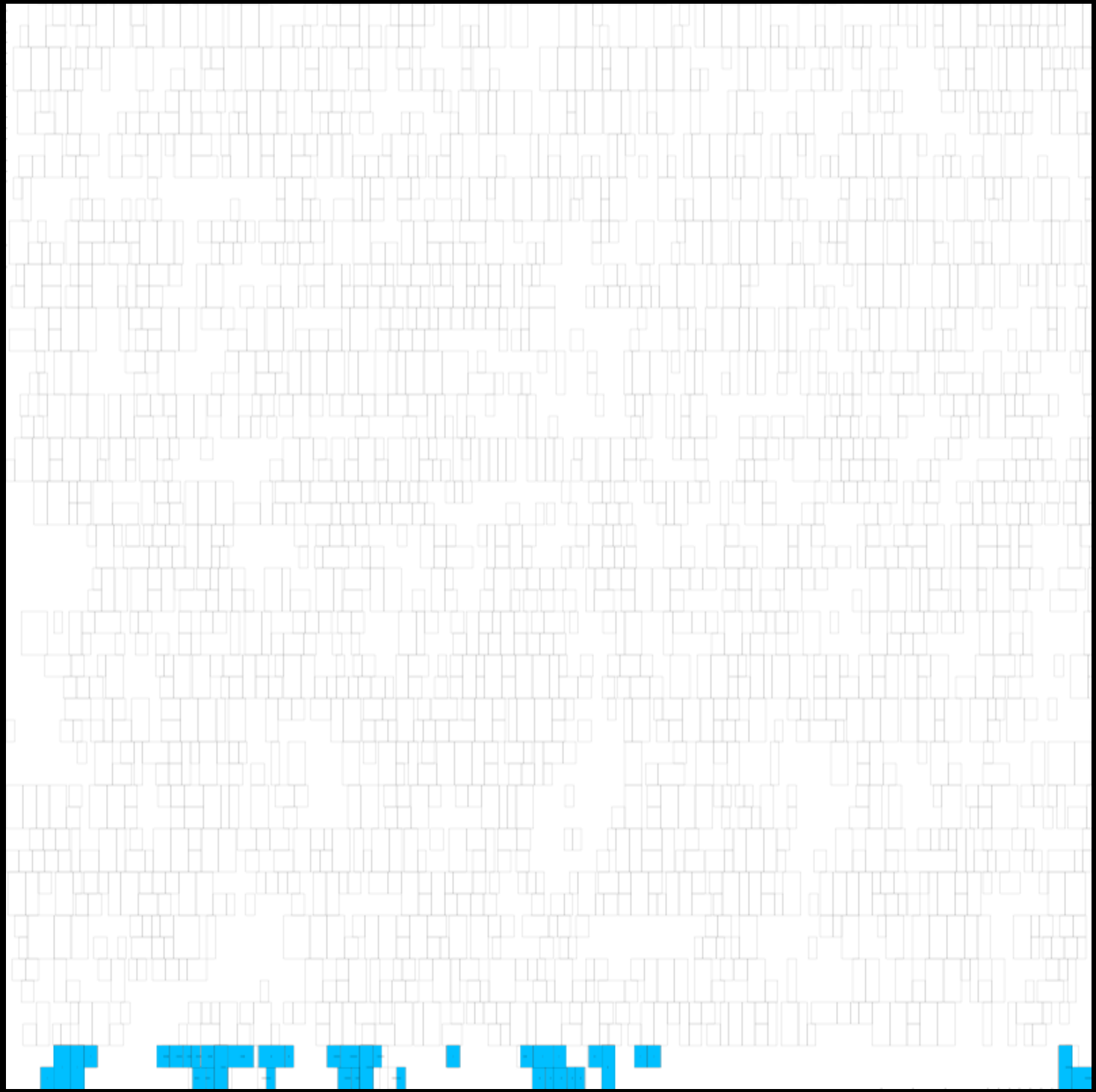


Overall Flow



Overall Flow

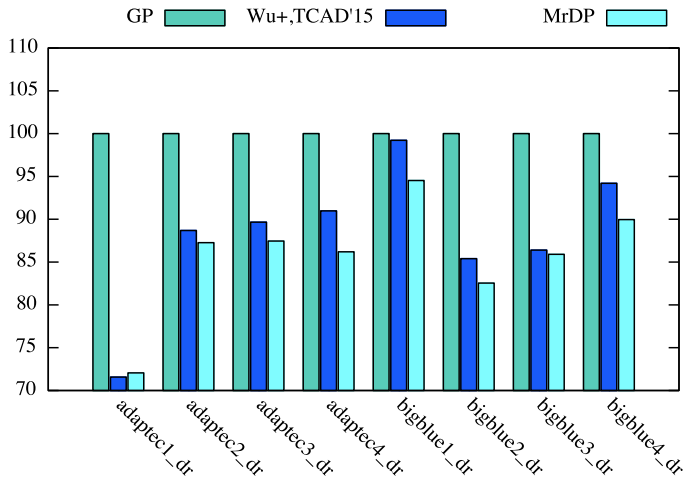




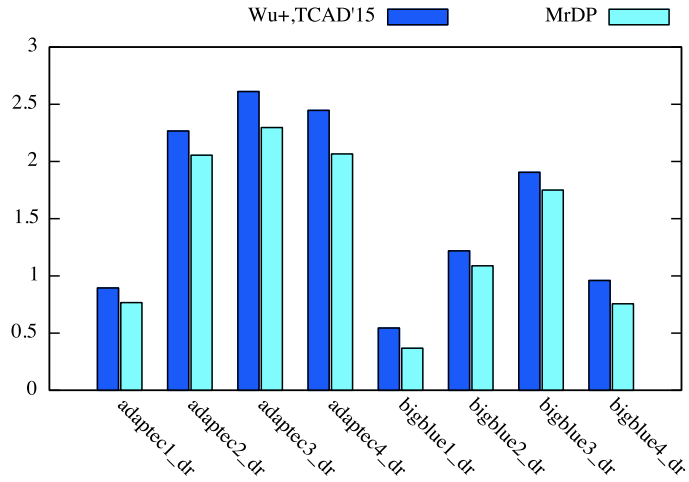
Experimental Setup

- Implemented in C++
- 8-Core 3.4GHz Linux server
- 32GB RAM
- **ISPD 2005 Contest Benchmark:**
  - Double-row height cells [Wu+,TCAD'15]
  - Benchmark sizes: 200K to 2M
  - Utilization: 67% to 91%
  - Double-Row Ratio: around 30%
- **ICCAD 2014 Contest Benchmark:**
  - Multiple-row height cells (2–4 rows)
  - Benchmark sizes: 133K to 961K
  - Utilization: 47% to 65%
  - Multiple-Row Ratio: 15% to 41%

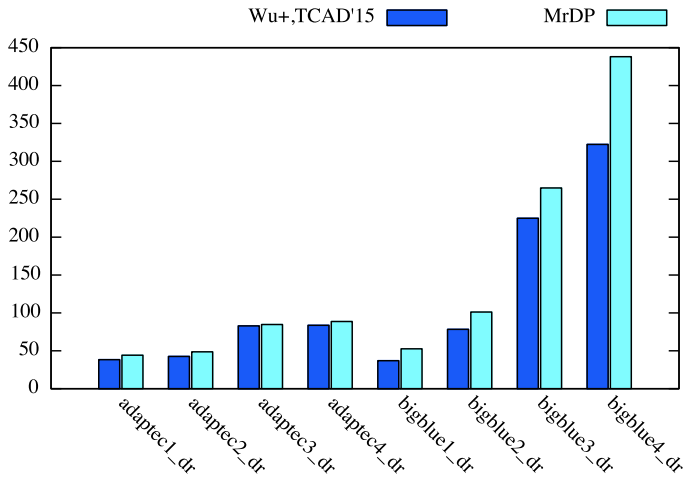
Results on Double-row Height Cells



(a) Normalized sHPWL



(b) APU penalty

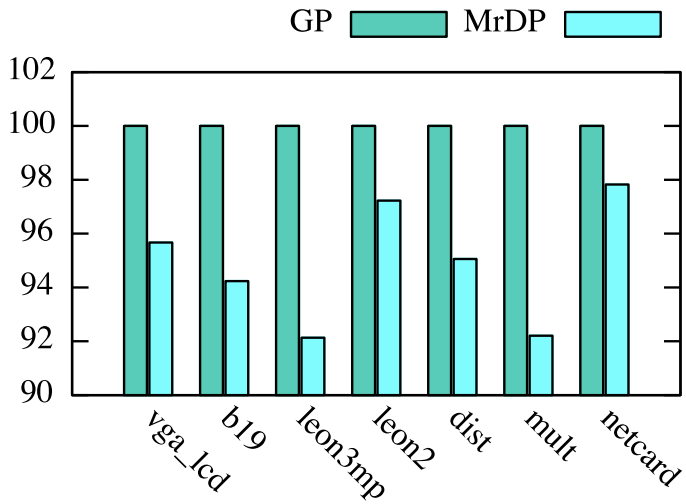


(c) Runtime (s)

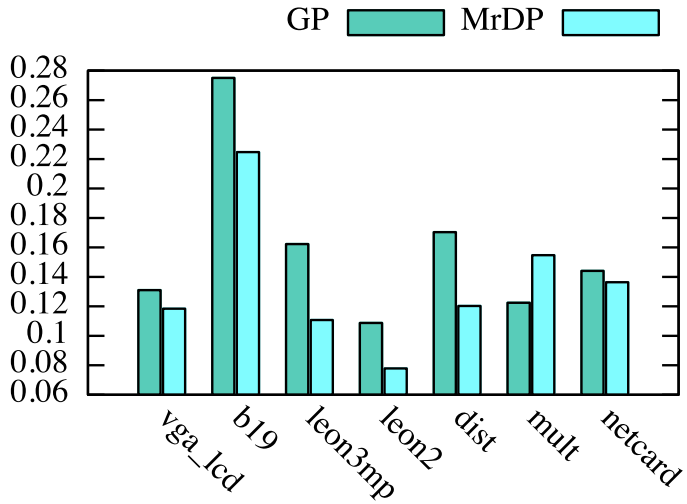
MrDP v.s. [Wu+,TCAD'15]

- **3% better** sHPWL
- **13.2% better** APU
- 23.5% runtime overhead

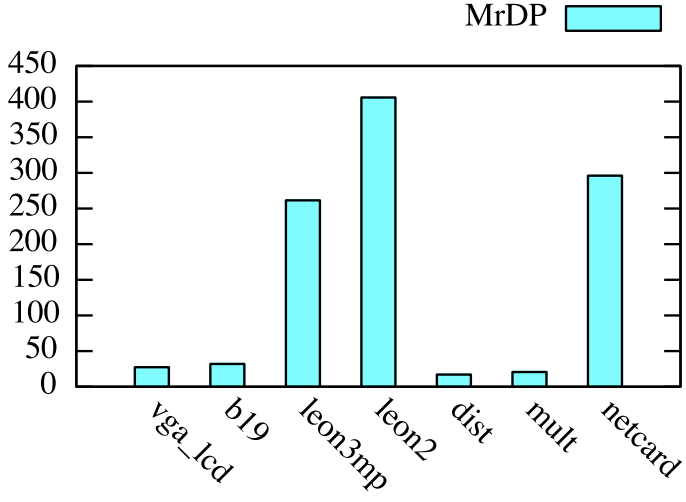
Results on Heterogeneous-sized Cells



(a) Normalized sHPWL



(b) APU penalty



(c) Runtime (s)

MrDP v.s. GP

- 3.7% better sHPWL
- 15.3% better APU

Conclusion

Placement challenges with heterogeneous-sized standard cells in advanced technology nodes

- A placement framework to optimize wirelength and congestion
- Chain move scheme
- Ordered double-row placement

Conclusion

Placement challenges with heterogeneous-sized standard cells in advanced technology nodes

- A placement framework to optimize wirelength and congestion
- Chain move scheme
- Ordered double-row placement

Future work

- Explore the impacts of legalization step
- Different configurations of placement flows

Thank You

Yibo Lin (yibolin@cerc.utexas.edu)

Bei Yu (byu@cse.cuhk.edu.hk)

Xiaoqing Xu (xiaoqingxu@cerc.utexas.edu)

Jhih-Rong Gao (jrgao@cadence.com)

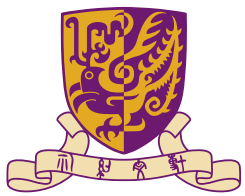
Natarajan Viswanathan (nviswan@cadence.com)

Wen-Hao Liu (wliu@cadence.com)

Zhuo Li (zhuoli@cadence.com)

Charles J. Alpert (alpert@cadence.com)

David Z. Pan (dpan@ece.utexas.edu)



cādence