《芯片设计自动化与智能优化》 Detailed Placement

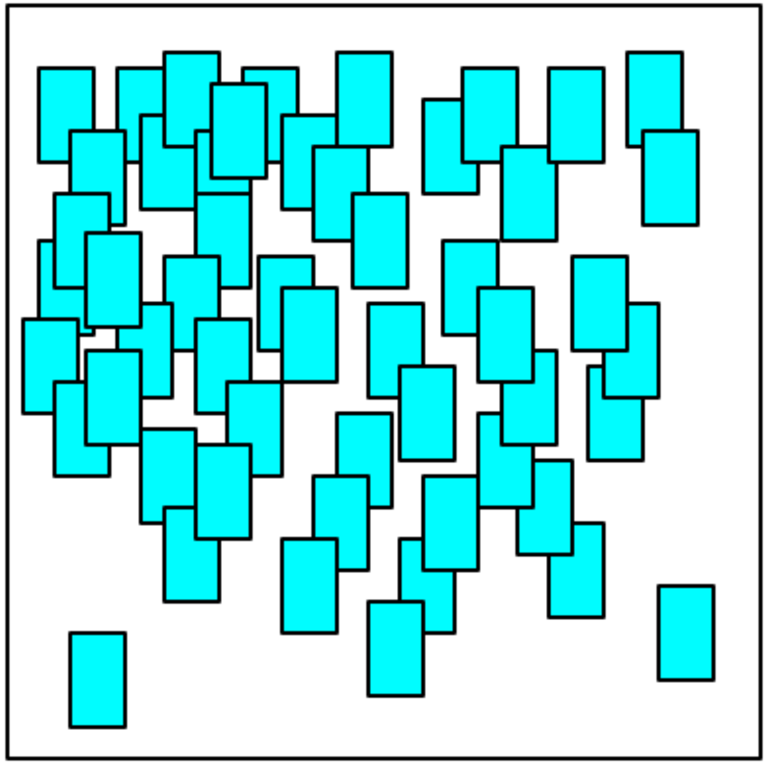
The slides are partly based on Prof. David Z. Pan's lecture notes at UT Austin.

Yibo Lin

Peking University

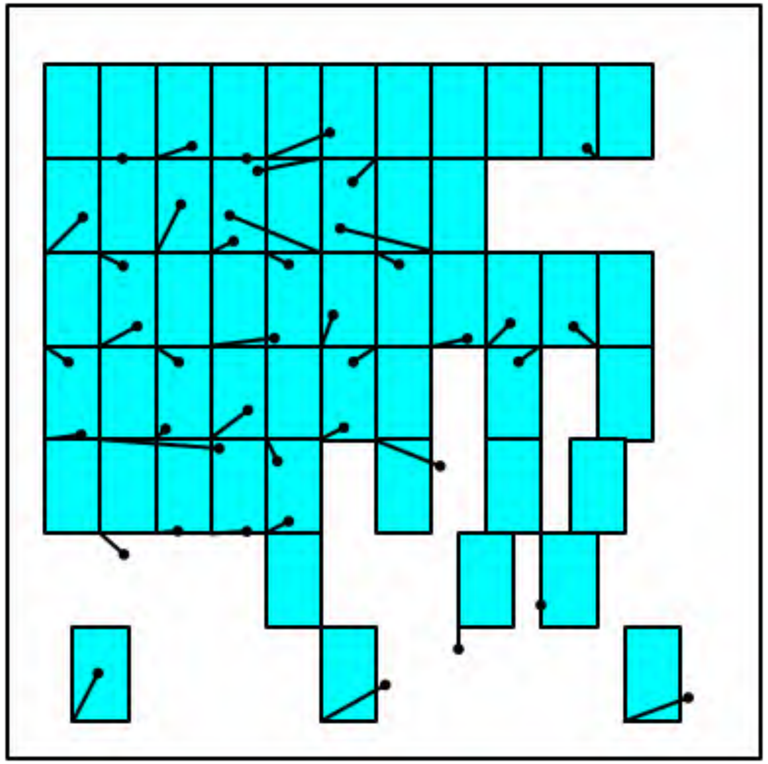
Typical Placement Flow

WL: 1.00e+6



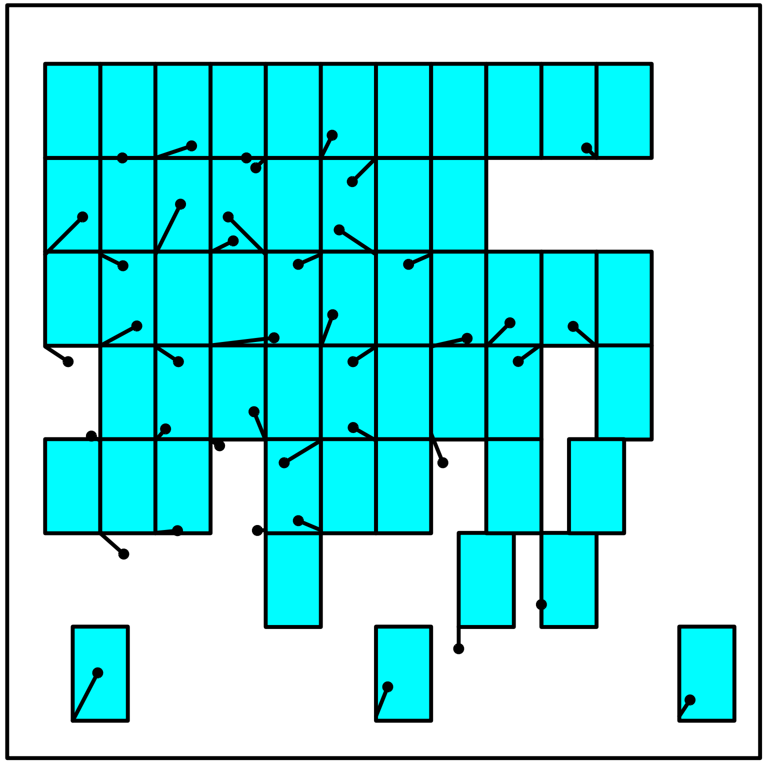
Global placement

WL: 1.05e+6



Legalization

WL: 1.02e+6



Detailed Placement

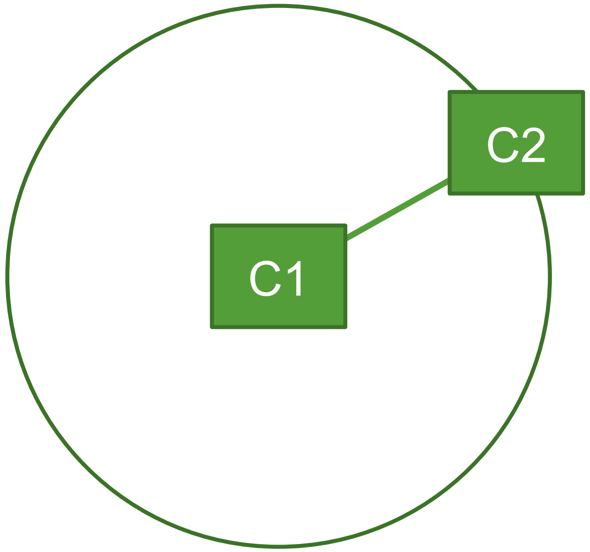
Outline

- What is placement
- History of placement algorithms
- Global placement
  - Quadratic placement: FastPlace & SimPL
  - Nonlinear placement: NTUplace & ePlace
- Legalization
  - Tetris
  - Row-based algorithms: Abacus, DP, LP, MCF
  - Integer linear programming
- **Detailed placement**
  - Global move & swap
  - Independent set matching
  - Local reordering
  - Row-based algorithms: DP, LP, MCF
- Other topics
  - Routability-driven placement
  - Timing-driven placement
  - Macro placement

Detailed Placement – Problem Formulation

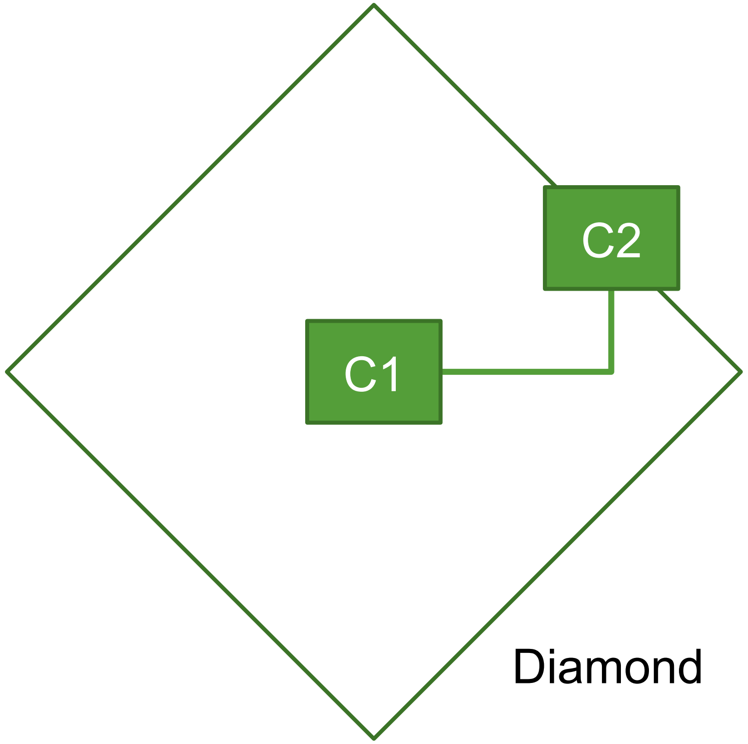
- Input
  - Legal placement
- Output
  - Refine the locations of blocks
  - Guarantee legality
- Objective
  - Minimize wirelength
- Other objectives
  - Timing, routability, etc.

Basic Intuition for Wirelength Refinement



Circle

$$(x_{c2} - x_{c1})^2 + (y_{c2} - y_{c1})^2 = c$$



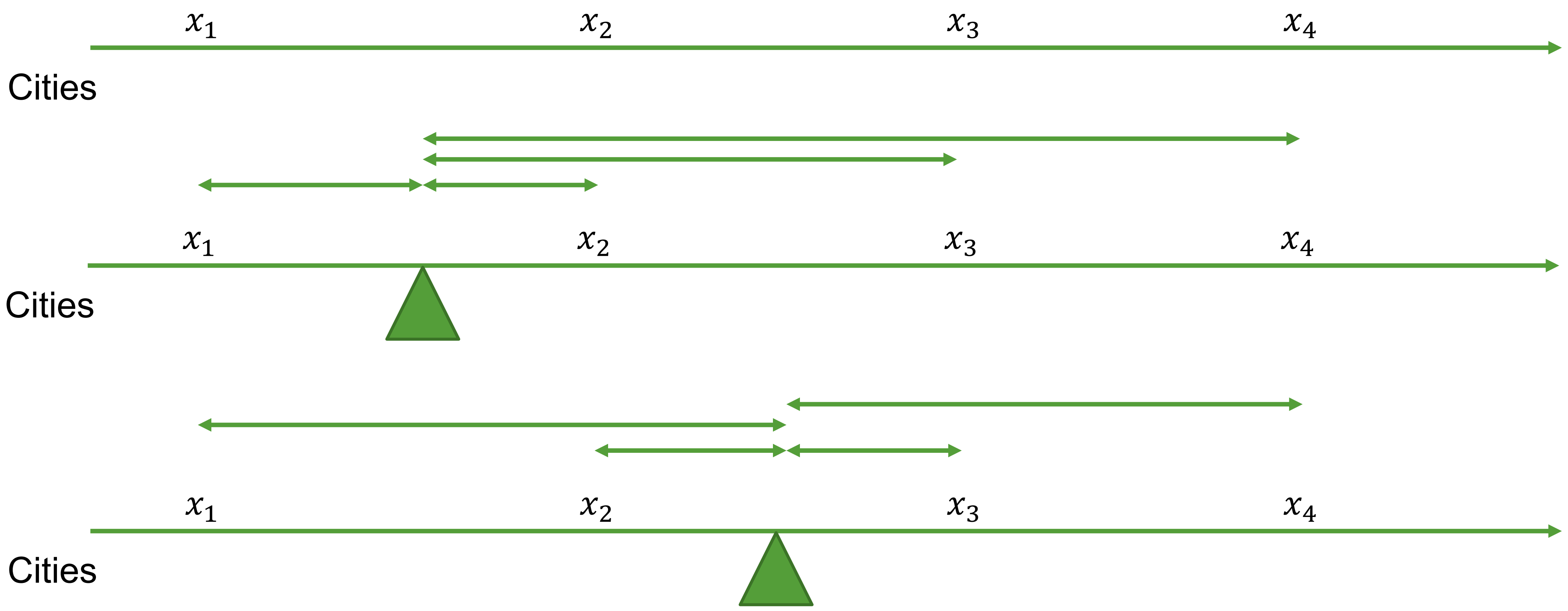
Diamond

$$|x_{c2} - x_{c1}| + |y_{c2} - y_{c1}| = c$$

Basic Intuition for Wirelength Refinement

Find the optimal warehouse location that minimizes the distances to all cities?
Cities are located in $x_1, x_2, \cdots, x_n$

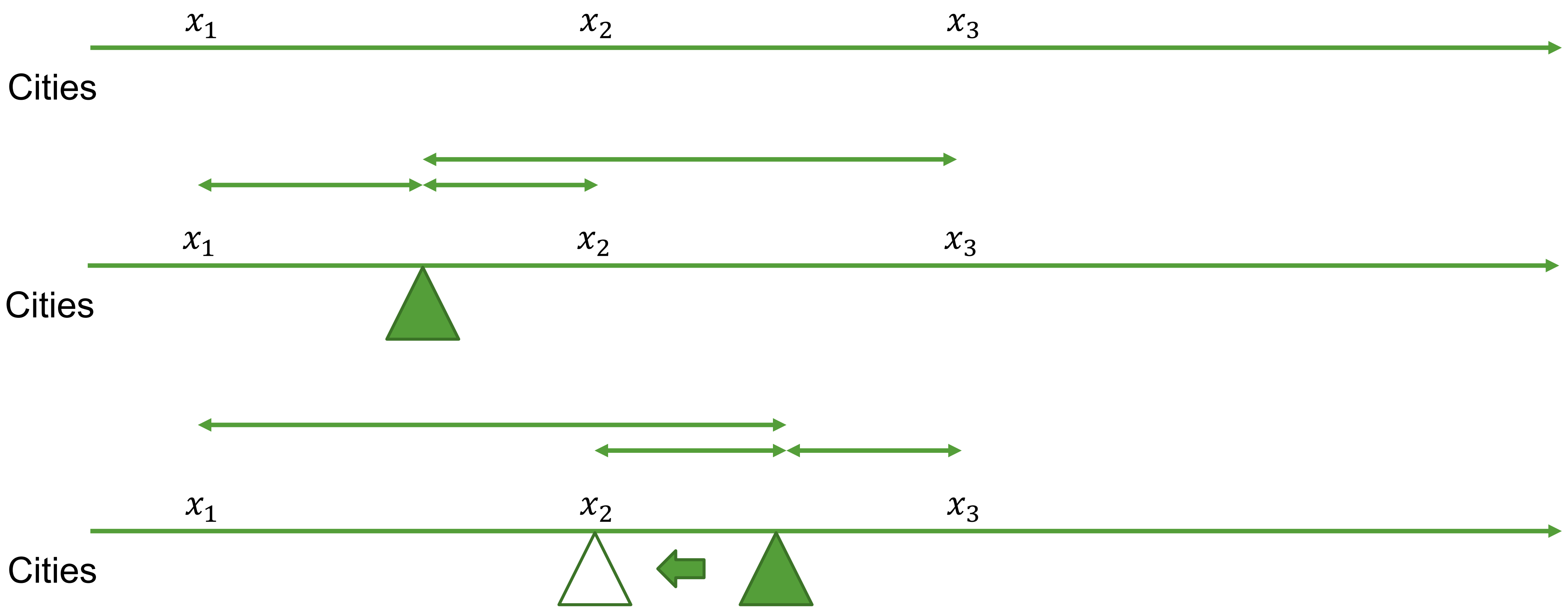
$$\min_{x_w} \quad |x_w - x_1| + |x_w - x_2| + |x_w - x_3| + |x_w - x_4|$$



Basic Intuition for Wirelength Refinement

Find the optimal warehouse location that minimizes the distances to all cities?
Cities are located in $x_1, x_2, \cdots, x_n$

$$\min_{x_w} \quad |x_w - x_1| + |x_w - x_2| + |x_w - x_3|$$



Basic Intuition for Wirelength Refinement

Find the optimal warehouse location that minimizes the distances to all cities?
Cities are located in $x_1, x_2, \cdots, x_n$

Optimal solution

Median of $\{x_1, x_2, \cdots, x_n\}$

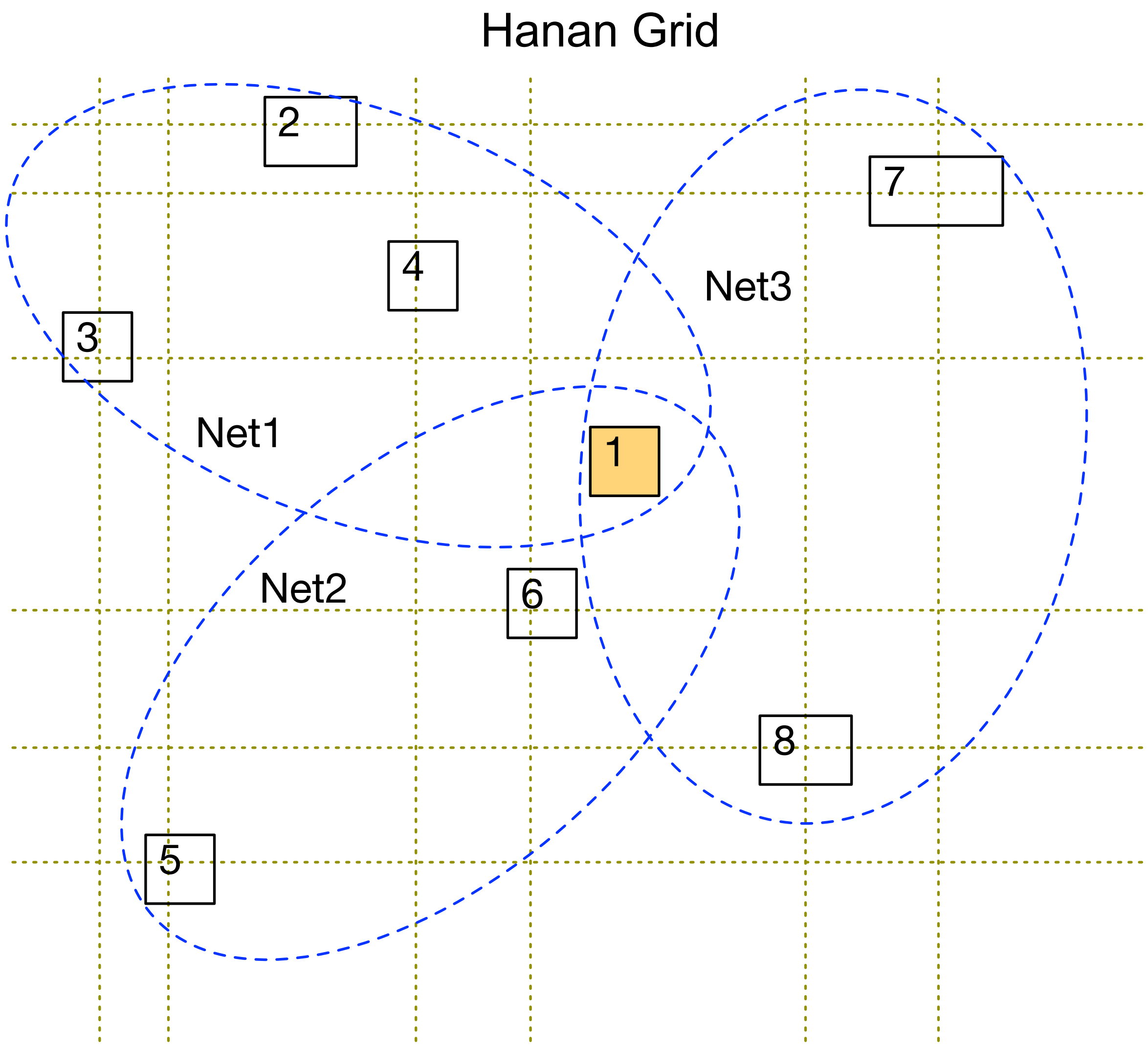
Case I: n is an odd number
Case II: n is an even number

Time complexity to compute median

Sorting: $O(n \log n)$
Kth smallest/largest I: expected $O(n)$
Kth smallest/largest II: worst case $O(n)$

Optimal Region

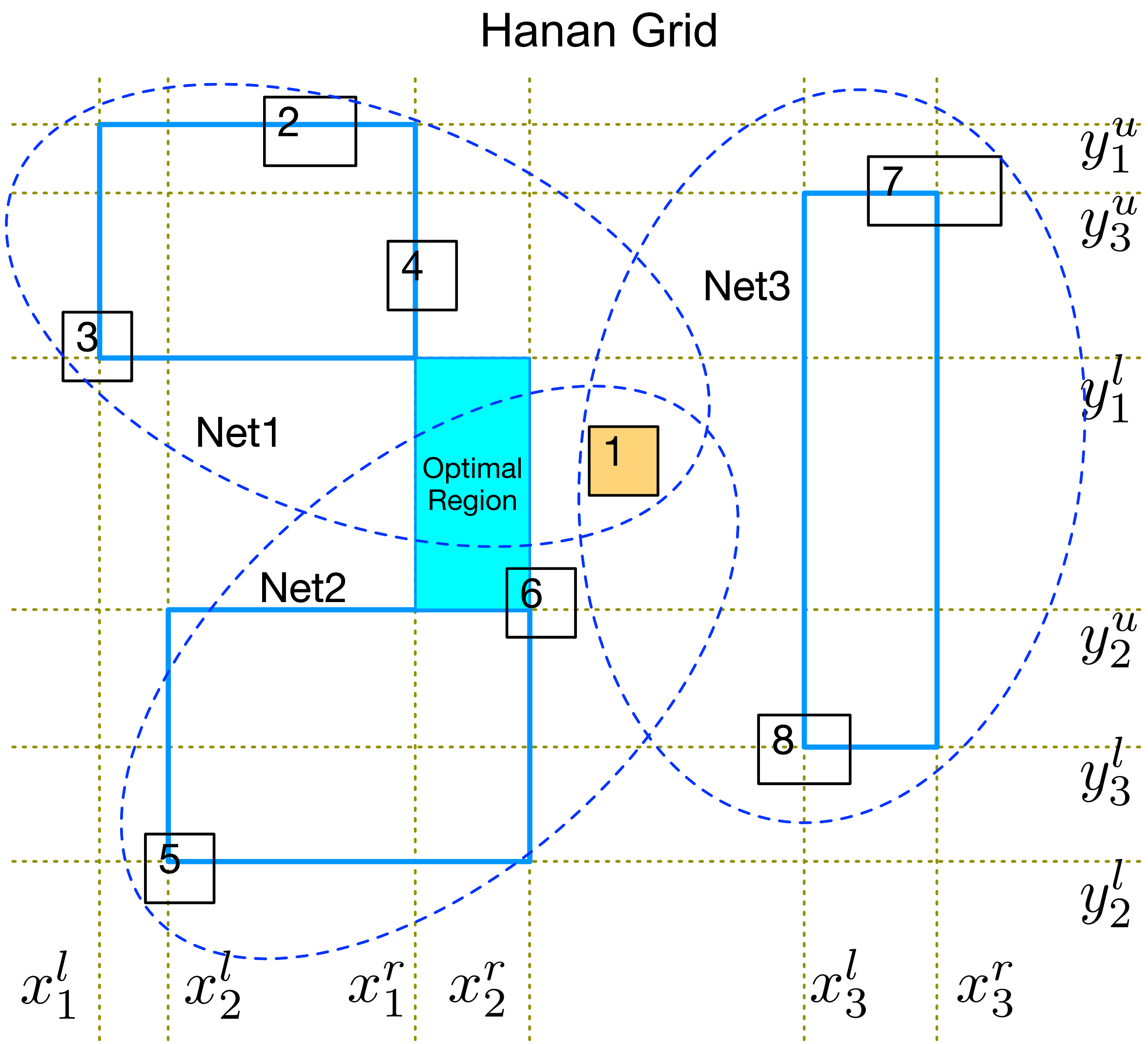
- Consider cell 1
  - Net1: {1, 2, 3, 4}
  - Net2: {1, 5, 6}
  - Net3: {1, 7, 8}
- Assume all other cells are fixed
- The region for cell 1 to achieve minimum total wirelength



Pan, Min, Natarajan Viswanathan, and Chris Chu. "An efficient and effective detailed placement algorithm." ICCAD 2005.

Optimal Region

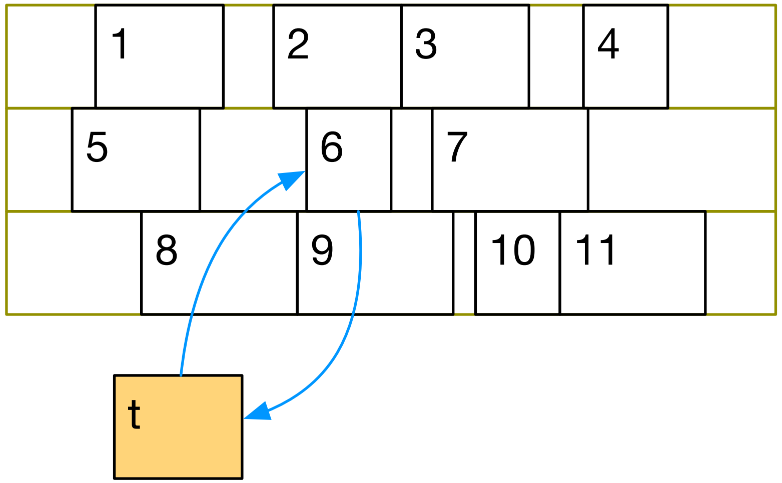
- Consider cell 1
  - Net1: {1, 2, 3, 4}
  - Net2: {1, 5, 6}
  - Net3: {1, 7, 8}
- Assume all other cells are fixed
- The region for cell 1 to achieve minimum total wirelength
  - Medians of $\{x_1^l, x_1^r, x_2^l, x_2^r x_3^l, x_3^r\}$
  - Medians of $\{y_1^l, y_1^u, y_2^l, y_2^u, y_3^l, y_3^u\}$
  - A REGION, not a point!



Pan, Min, Natarajan Viswanathan, and Chris Chu. "An efficient and effective detailed placement algorithm." ICCAD 2005.

Global Move & Swap

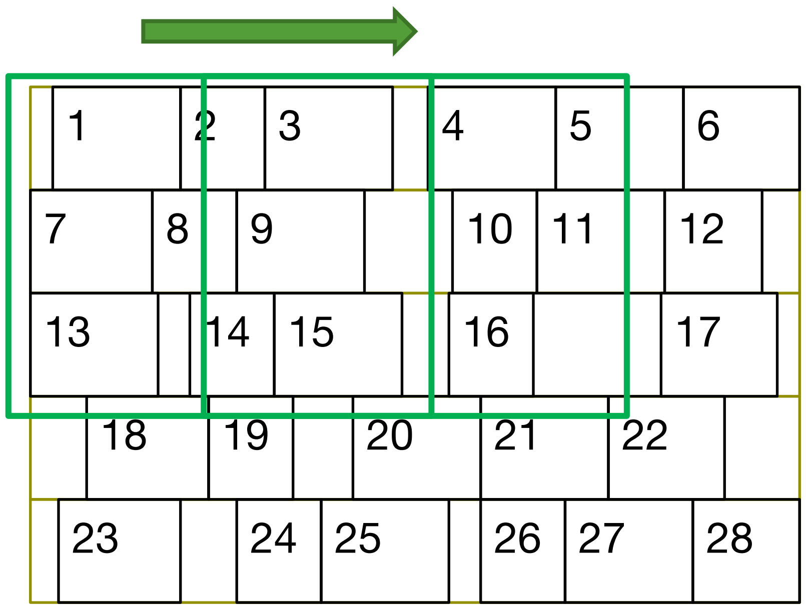
- Generalize swap and move
  - Regard move as swapping with a whitespace
- For each cell
  - Compute its optimal region
  - Enumerate all swap candidates within the optimal region (or expanded regions) and compute cost
  - Swap with the best one if applicable
- Heuristic cost function
  - Wirelength changes
  - Penalizing if swapping with cells of different sizes
  - Penalizing if causing overlaps

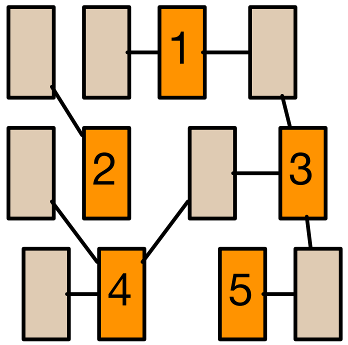


Pan, Min, Natarajan Viswanathan, and Chris Chu. "An efficient and effective detailed placement algorithm." ICCAD 2005.

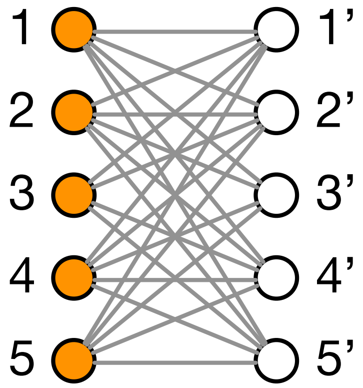
Independent Set Matching

- "Independent"
  - Cells do not directly connect with each other
- Sliding windows





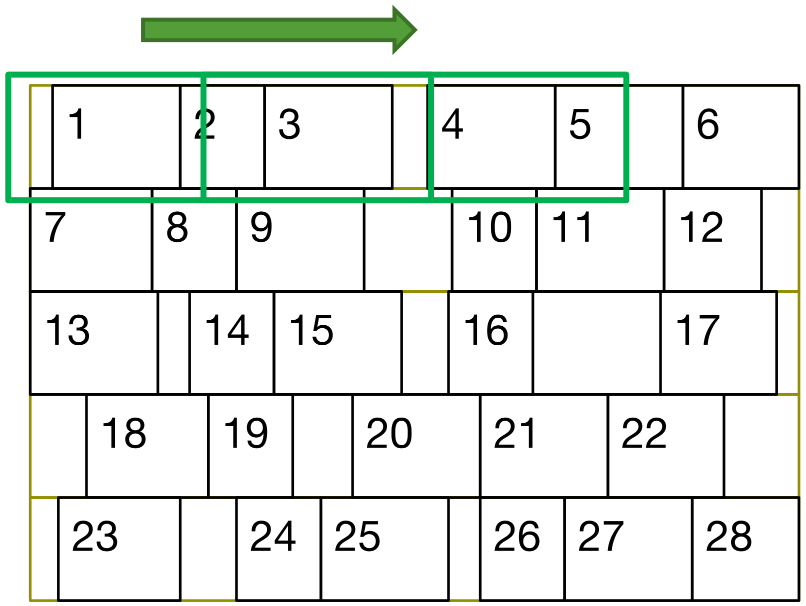
Extract an independent set of cells

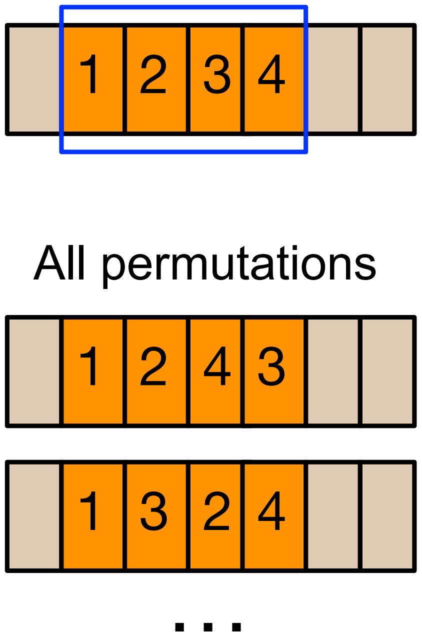


Bipartite matching

Local Reordering

- For a small window in a row
  - Enumerating all the combinations of orders
  - Choose the order with the best cost
- Sliding window



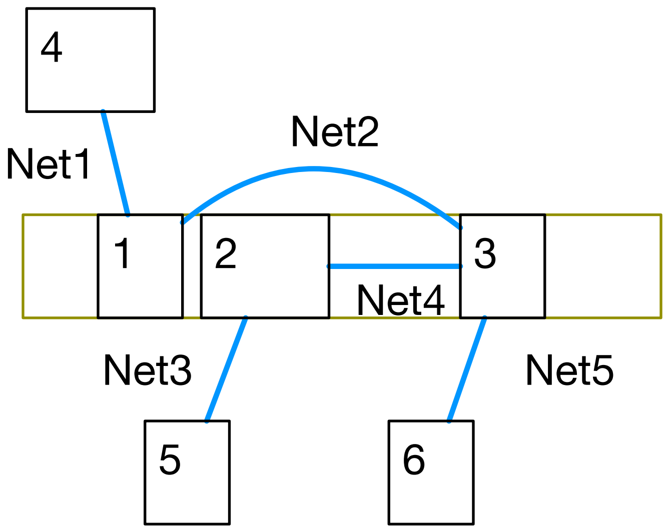


Row-based Algorithms – Linear Placement

- Consider a single row of placement
  - Keep the **relative order** of cells in a row
  - Cells on other rows are fixed
- Multiple algorithms to solve
  - Linear programming
  - Dynamic programming [Taghavi+, ICCAD2010]
  - Clumping algorithm [Kahng+, ASPDAC1999]
- Legalization problem
  - $\min_{\boldsymbol{x}} \quad \sum_i |x_i - x_i^{gp}|,$
  - $s.t. \quad x_i - x_{i-1} \geq w_i,$
  - $\quad x_i \geq L, x_i \leq R$
- Detailed placement problem
  - $\min_{\boldsymbol{x}, \boldsymbol{R_e}, \boldsymbol{L_e}} \quad \sum_{e \in E}(R_e - L_e),$
  - $s.t. \quad x_i - x_{i-1} \geq w_i,$
  - $\quad x_i \geq L_e, x_i \leq R_e, \forall i \in e, e \in E$
  - $\quad x_i \geq L, x_i \leq R$

Row-based Algorithms – Linear Placement

- Consider a row of cells {1, 2, 3}
  - Cells {4, 5, 6} are fixed
  - 5 nets: {1, 4}, {1, 3}, {2, 5}, {2, 3}, {3, 6}
  - Introduce $\{L_i, R_i\}$ as the bounding box for each net i
  - $\min\limits_{x,R_i,L_i} \quad \sum_{i=1}^{5} R_i - L_i$,
  - $s.t. \quad x_i - x_{i-1} \geq w_i,$
  - Net1: $x_1 \geq L_1, x_1 \leq R_1, x_4^c \geq L_1, x_4^c \leq R_1,$
  - Net2: $x_1 \geq L_2, x_1 \leq R_2, x_3 \geq L_2, x_3 \leq R_2,$
  - Net3: $x_2 \geq L_3, x_2 \leq R_3, x_5^c \geq L_3, x_5^c \leq R_3,$
  - Net4: $x_2 \geq L_4, x_2 \leq R_4, x_5^c \geq L_4, x_5^c \leq R_4,$
  - Net5: $x_3 \geq L_5, x_3 \leq R_5, x_6^c \geq L_5, x_6^c \leq R_5,$
  - $x_i \geq L, x_i \leq R$



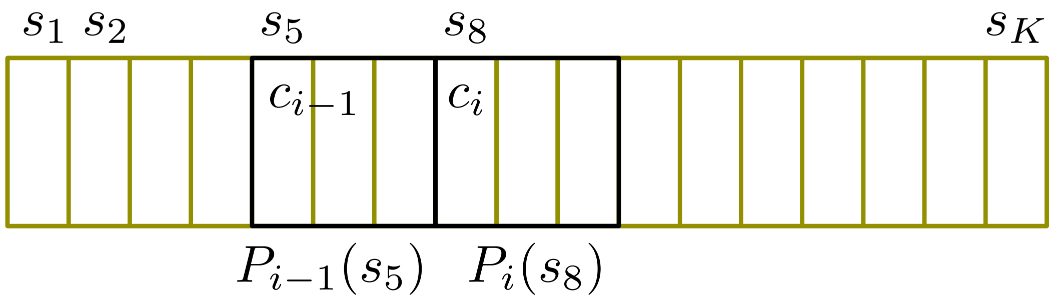
Differential constraints only

Can be solved with dual min-cost flow

Row-based Algorithms – Linear Placement

- Solve with dynamic programming
  - Find an independent subproblem
  - Place cells $c_1, \ldots, c_i$, c_i being at or to the left of the i^{th} feasible location s_i,
  - Minimize $\sum_{k=1}^{i} cost_k(x)$
- Recursion
  - $P_i(s_j) = \min\{P_i(s_{j-1}), P_{i-1}(s_j - w_{i-1}) + cost_i(s_j)\}$

How to handle cell c_i connected to cell c_j in the same row ($i < j$)?

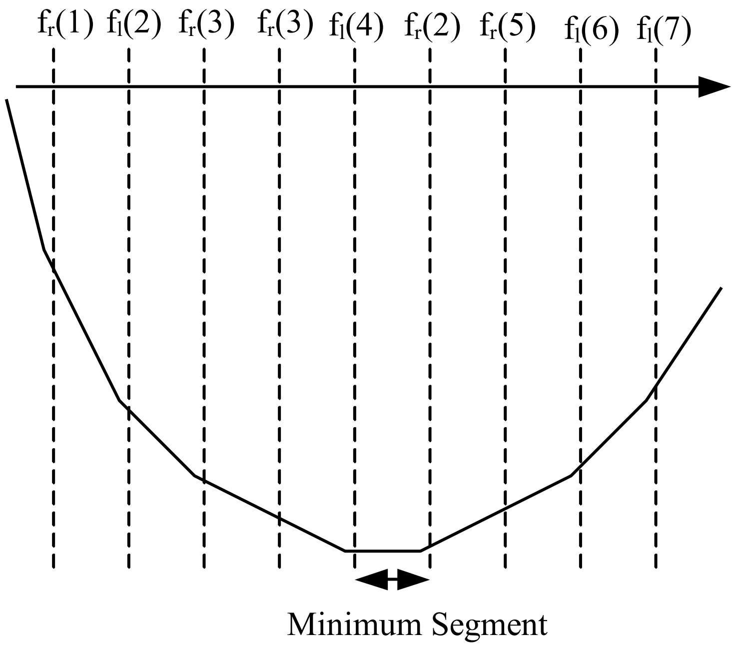


- Runtime complexity
  - $(i\ range) \times (j\ range) = n \times (K - \sum w_i) \approx O(n^2)$

Taghavi, Taraneh, et al. "New placement prediction and mitigation techniques for local routing congestion." ICCAD 2010.

Row-based Algorithms – Linear Placement

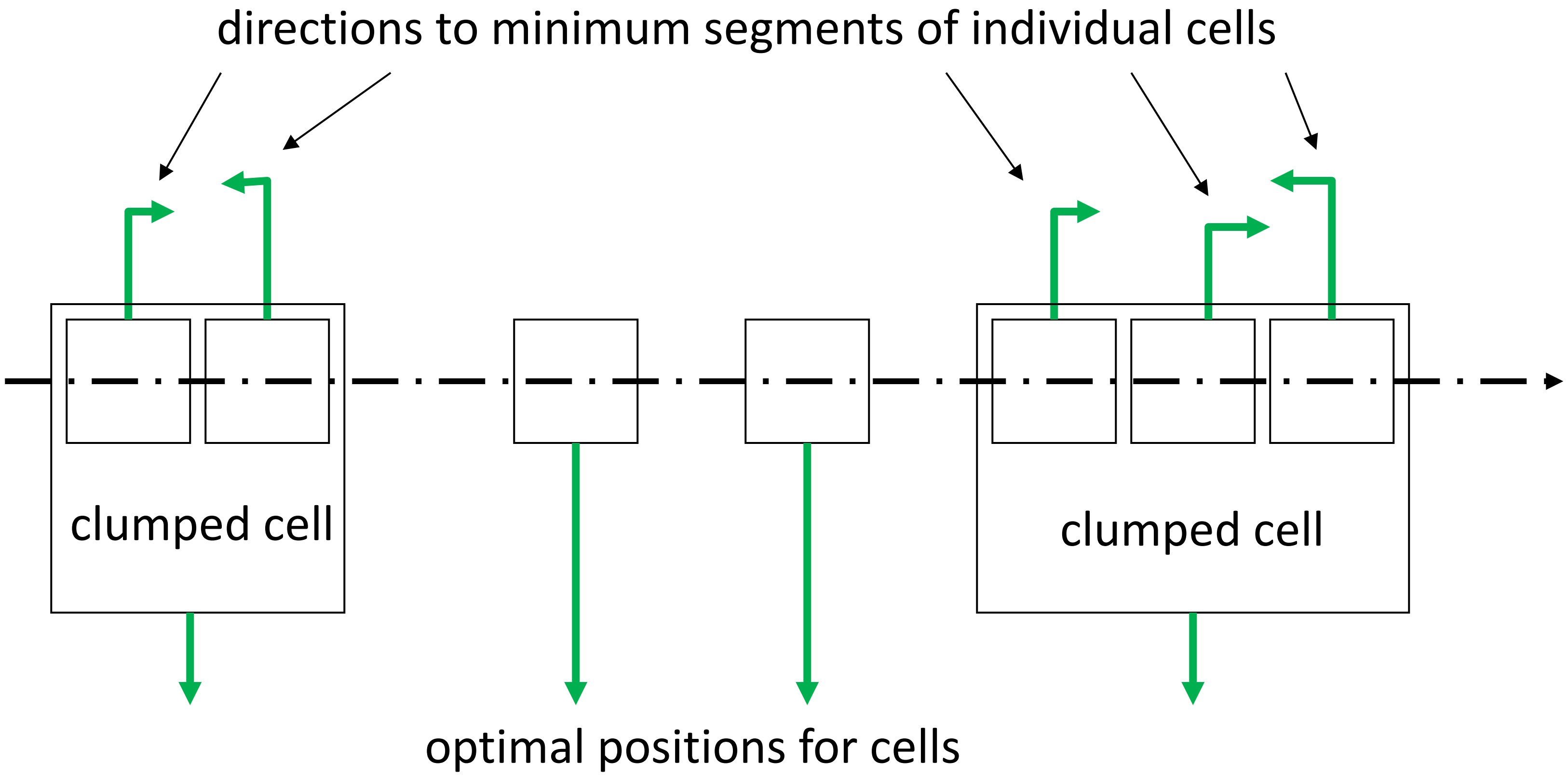
- Clumping algorithm
  - Piece-wise linear and convex $cost_i$
  - Find the minimum interval for c_i, i.e., where slope of $cost_i$ is 0.
  - Scan adjacent cell starting from left
  - If c_{i-1} and c_i can not be put in their minimum intervals without overlap, replace them with c'_{i-1}, with all the pins replicated, then consider placing c'_{i-1}
  - Otherwise place c_i on the leftmost legal site
  - Complexity $O(m^2)$, m is #nets
  - Use RB tree to reduce to $O(m \log m)$



Kahng, Andrew B., Paul Tucker, and Alexander Zelikovsky. "Optimization of linear placements for wirelength minimization with free sites." ASPDAC 1999.

Row-based Algorithms – Linear Placement

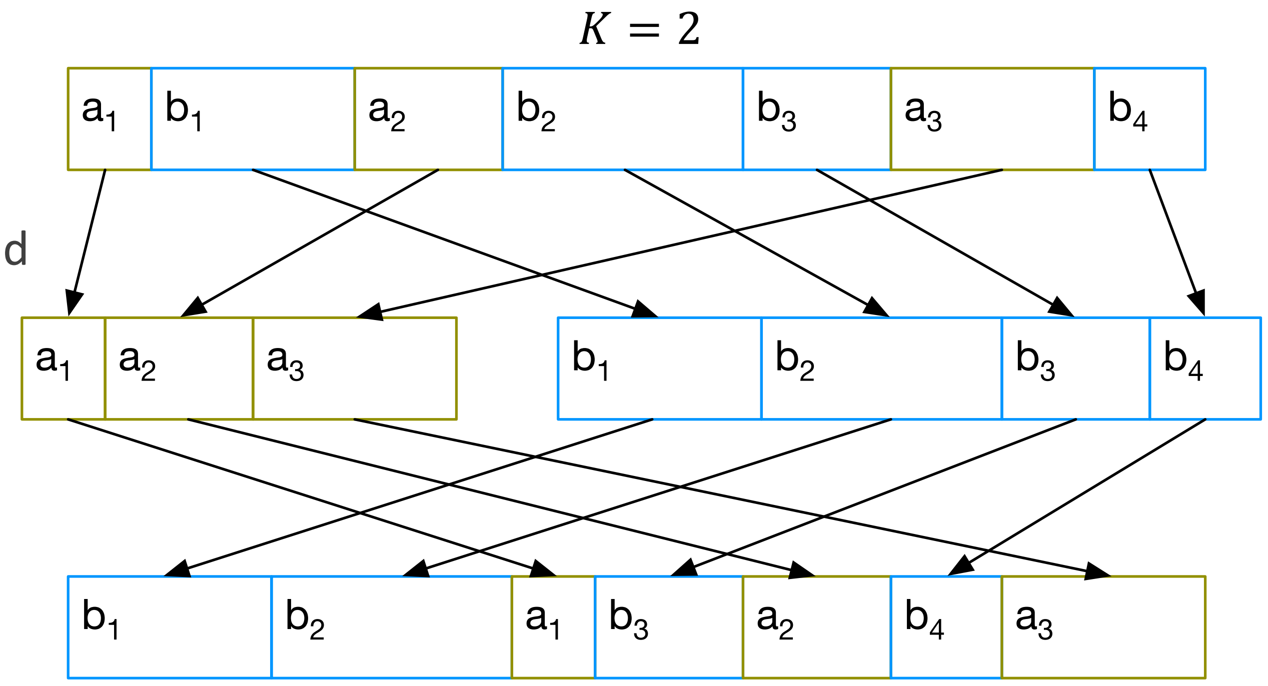
- Clumping algorithm
  - Piece-wise linear and convex $cost_i$



Kahng, Andrew B., Paul Tucker, and Alexander Zelikovsky. "Optimization of linear placements for wirelength minimization with free sites." ASPDAC 1999.

Row-based Algorithms – Window Interleaving

- Any way to permute cells rather than keep the order?
  - Divide a row of cells into K partitions
  - Keep the relative order within each partition
- Dynamic programming
  - <u>Subproblem</u>
  - Denote S_{ij} as the interleaving if $\{a_1, a_2, \cdots, a_i\}$ and $\{b_1, b_2, \cdots, b_j\}$, with $C(S_{ij})$ minimized
  - $C(S_{ij})$ denotes the cost of S_{ij} ,
  - <u>Recursion</u>
  - $S_{0,0} = 0, C(S_{0,0}) = 0$
  - $S_{ij} = \begin{cases} S_{i-1,j}a_i, if\ C(S_{i-1,j}a_i) < C(S_{i,j-1}b_j), \\ S_{i,j-1}b_j, otherwise \end{cases}$



Hur, Sung-Woo, and John Lillis. "Mongrel: Hybrid techniques for standard cell placement." ICCAD 2000.
Lin, Yibo, et al. "Triple patterning aware detailed placement toward zero cross-row middle-of-line conflict." TCAD 2017.

Famous Detailed Placement Algorithms

- Kahng, Andrew B., Paul Tucker, and Alexander Zelikovsky. "Optimization of linear placements for wirelength minimization with free sites." ASPDAC 1999.
- Pan, Min, Natarajan Viswanathan, and Chris Chu. "An efficient and effective detailed placement algorithm." ICCAD 2005.
- Tang, Xiaoping, Ruiqi Tian, and Martin DF Wong. "Optimal redistribution of white space for wire length minimization." ASPDAC. 2005.

- Parallel detailed placement algorithms on CPU and GPU
  - Lin, Yibo, Wuxi Li, Jiaqi Gu, Haoxing Ren, Brucek Khailany, and David Z. Pan. "ABCDPlace: Accelerated batch-based concurrent detailed placement on multithreaded CPUs and GPUs." TCAD 2020.

Summary of Detailed Placement

- Optimal region
- Global move & swap
- Independent set matching
- Local reordering
- Row-based placement
  - Linear placement: linear programming, min-cost flow, dynamic programming
  - Window interleaving