3D Placement with D2D Vertical Connections

B10732014 何雨澂

B10732027 吳秉寰

B10715041 馬孝傑

B10732040 李宇哲

Flow

- Global placement
- Partition
- Cell Legalization
- Detail Placement
- Terminal Placement

Global Placement (1/3)

- Use the gradient method to solve the objective function

$$\min\ \underbrace{\mathbf{W}(\mathbf{x},\mathbf{y})}_{\text{wirelength}} + \lambda \underbrace{\sum (\mathrm{D}(\mathrm{x},\mathrm{y}) - \mathrm{Mb})^2}_{\text{density}}$$

- By the "λ" increasing round by round

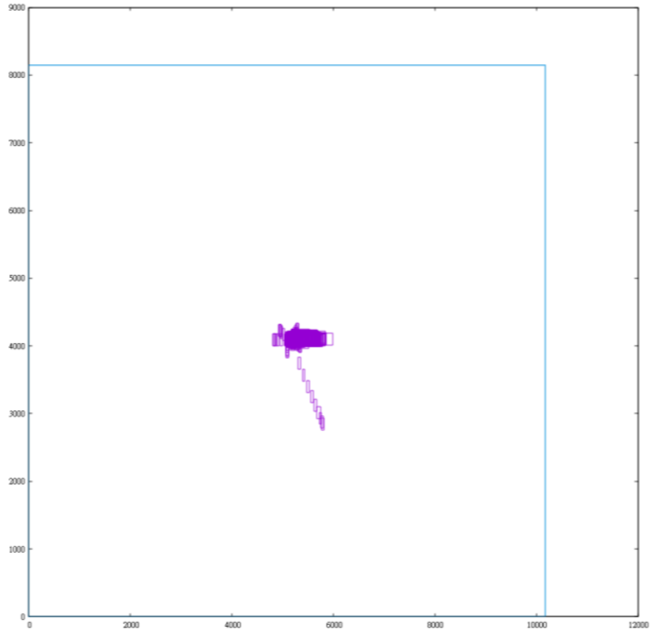
round 1: $\lambda = 0$ stepSize = (dieWidth + dieHeight) x 4

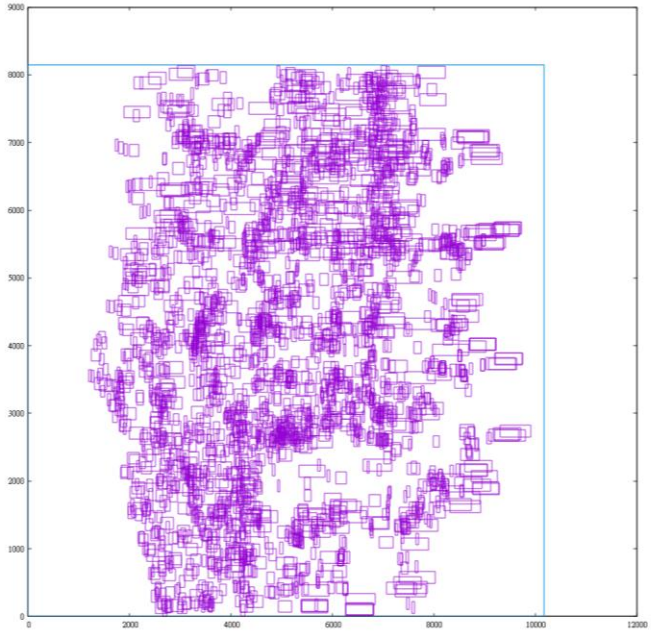
round 2: $\lambda = \beta$ stepSize /= 8

round 3: $\lambda = 2 \times \lambda$

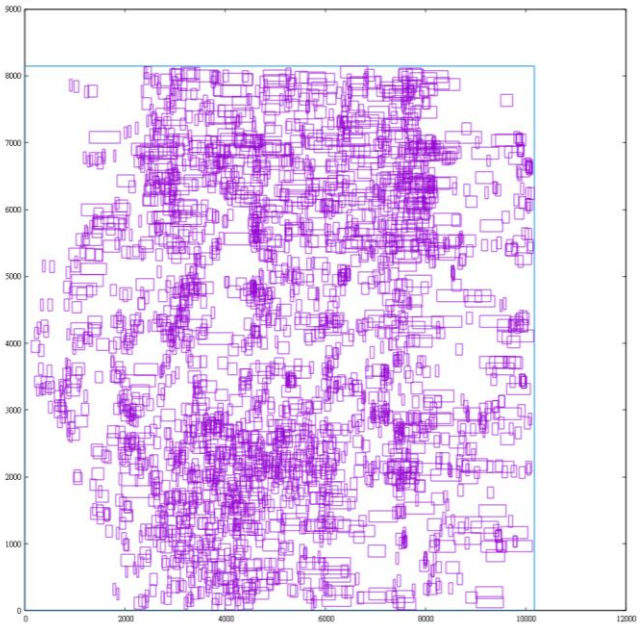
round 4: $\lambda = 2 \times \lambda$

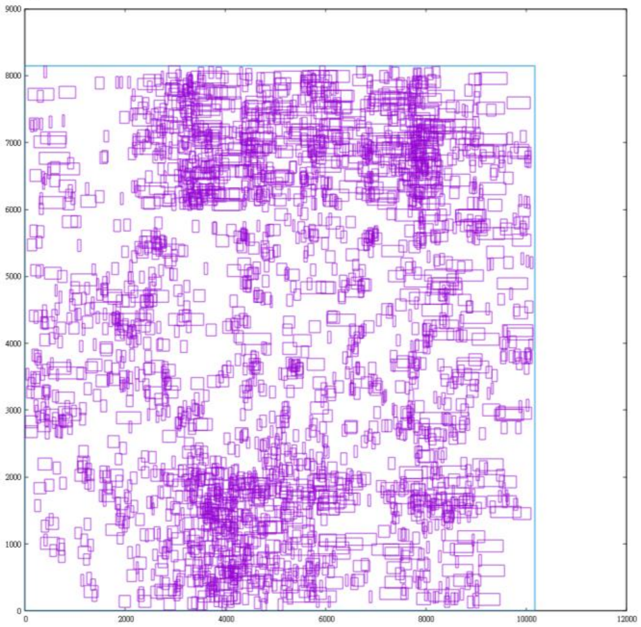
Global Placement (2/3)





Global Placement (3/3)





Cell Legalization (1/2)

- Solve quadratic program based on one linear equation
  - Assuming ‘=’ constraint

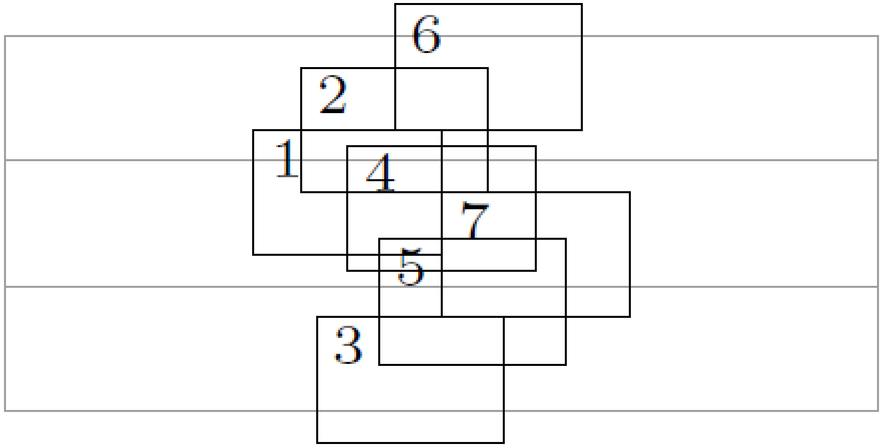
$$\min \sum_{i=1}^{N_r} e(i)[x(i) - x'(i)]^2 \qquad (1)$$

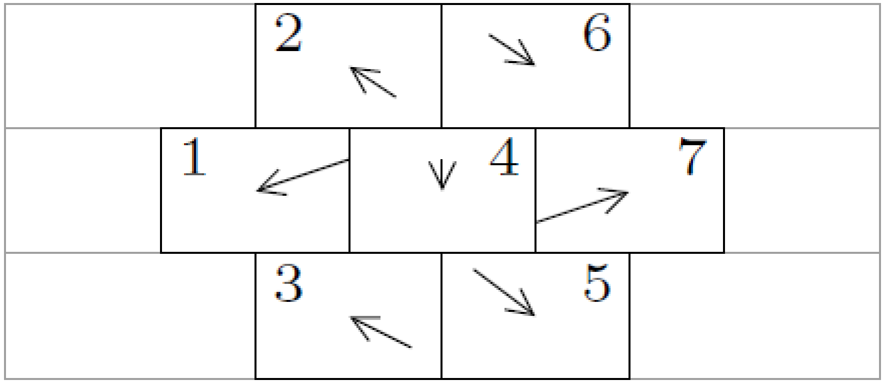
$$s.t. \quad x(i) - x(i-1) \geq w(i-1),\ i = 2, \dots, N_r$$

$$x(i) = x(1) + \sum_{k=1}^{i-1} w(k), i = 2, \dots, N_r \qquad (2)$$

Cell Legalization (2/2)

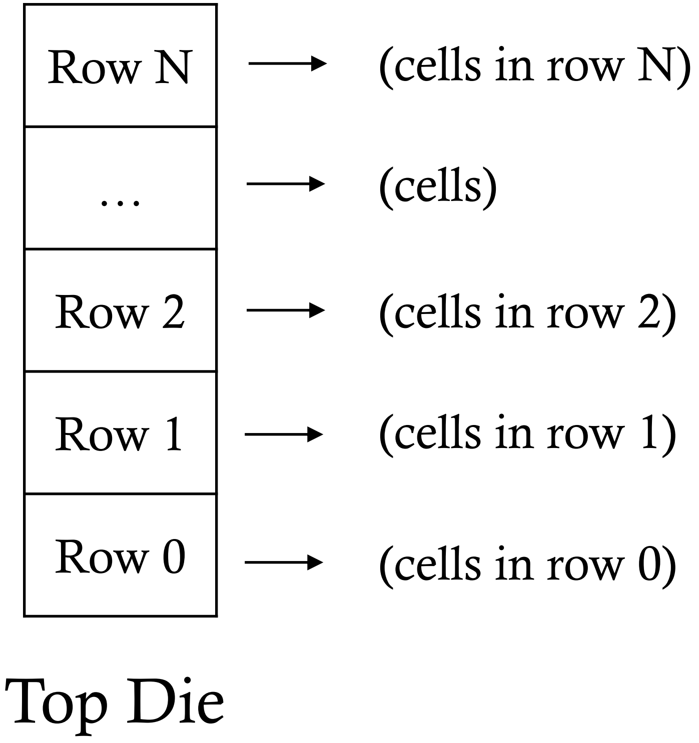
- Partition netlist into two dies with minimal terminals
- Sort cells according to x-position
  - Dynamic programming
- Insert each cell to rows
  - Determine the best row
  - Collapse clusters if there is overlap





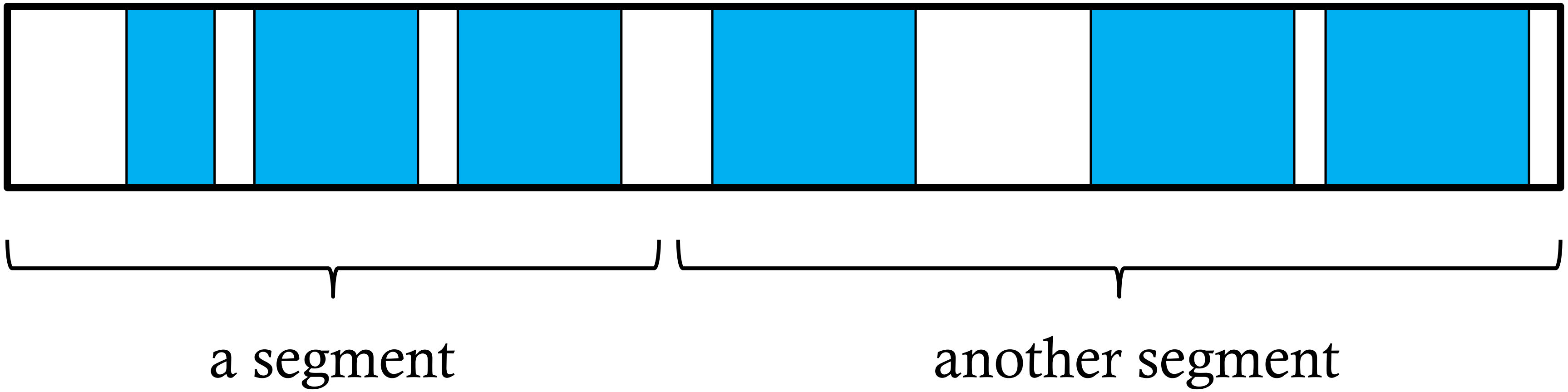
Detailed Placement (1/9)

- Classify instances with their location (die and row)
- Cells in each row are sorted by their coordinate



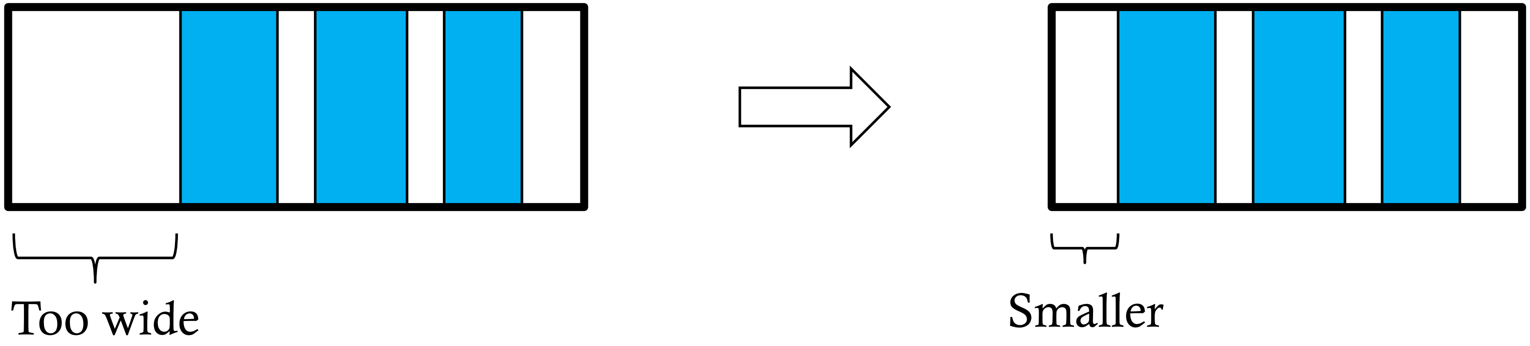
Detailed Placement (2/9)

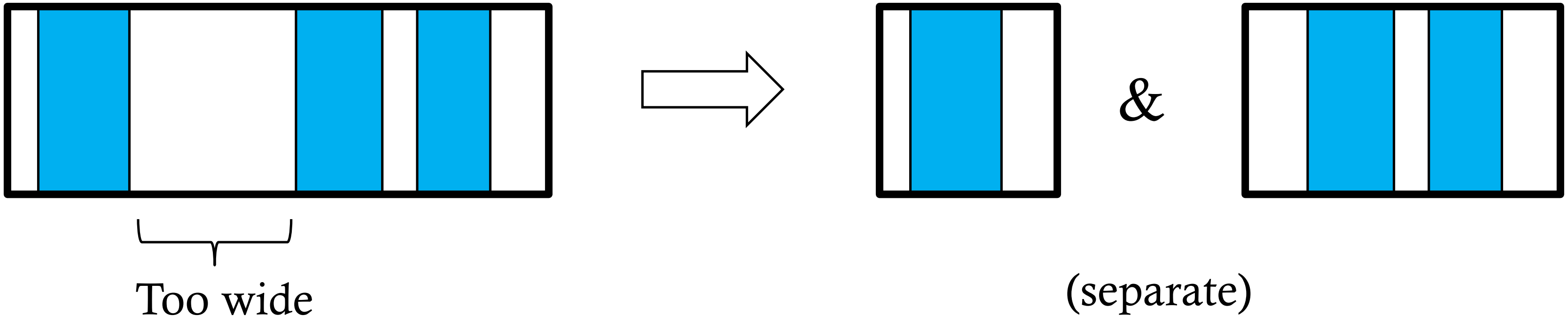
- In each row, construct segments by selecting three consecutive instances
- Segments are independent



Detailed Placement (3/9)

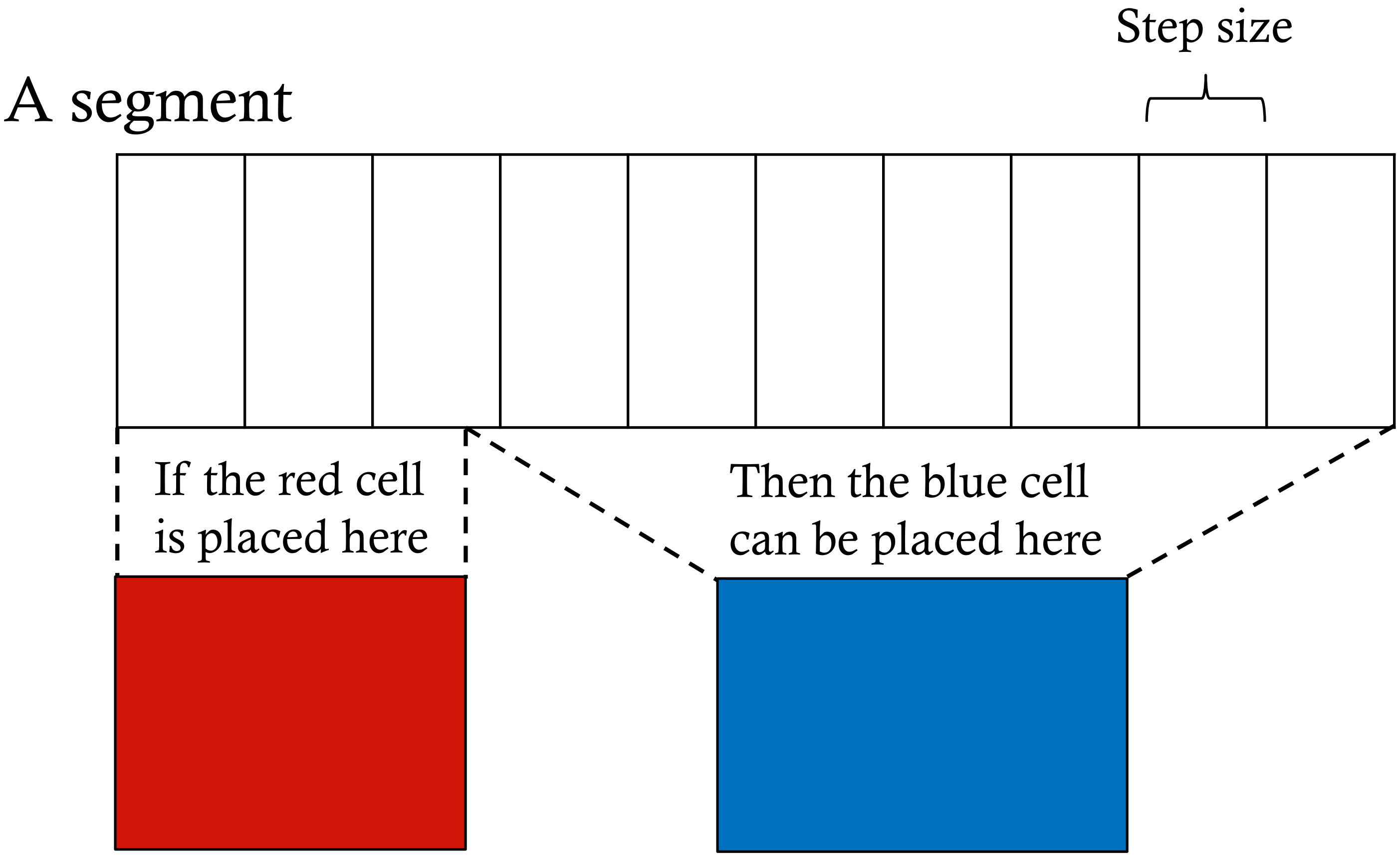
- **Adjust segments' size**





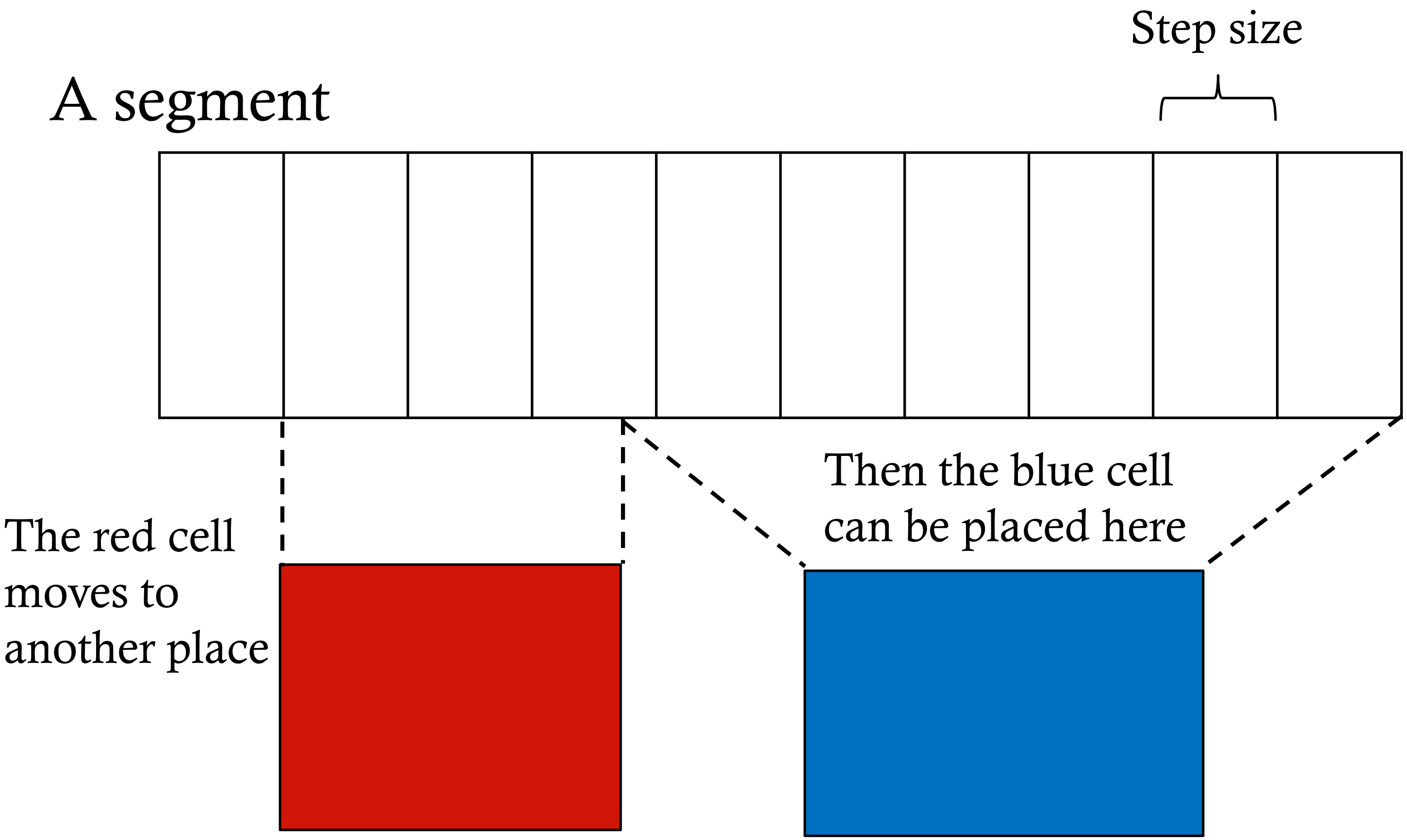
Detailed Placement (4/9)

- **For each segment, do DFS branch-and-bound (red→blue)**
- **Initialize “best_WL” by the HPWL before DP**



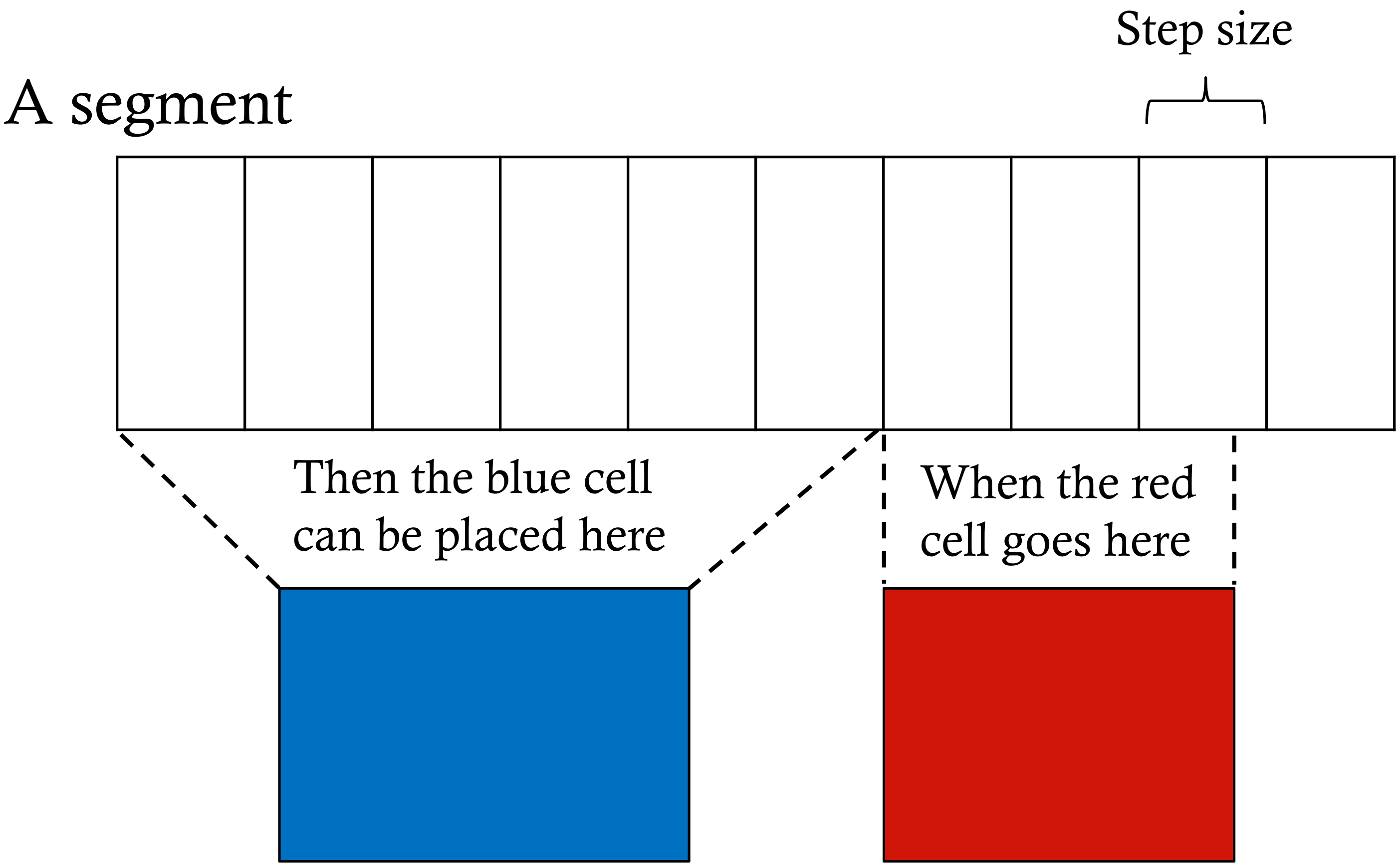
Detailed Placement (5/9)

- **For each segment, do DFS branch-and-bound (red→blue)**
- **Initialize “best_WL” by the HPWL before DP**


Step size
A segment
The red cell moves to another place
Then the blue cell can be placed here

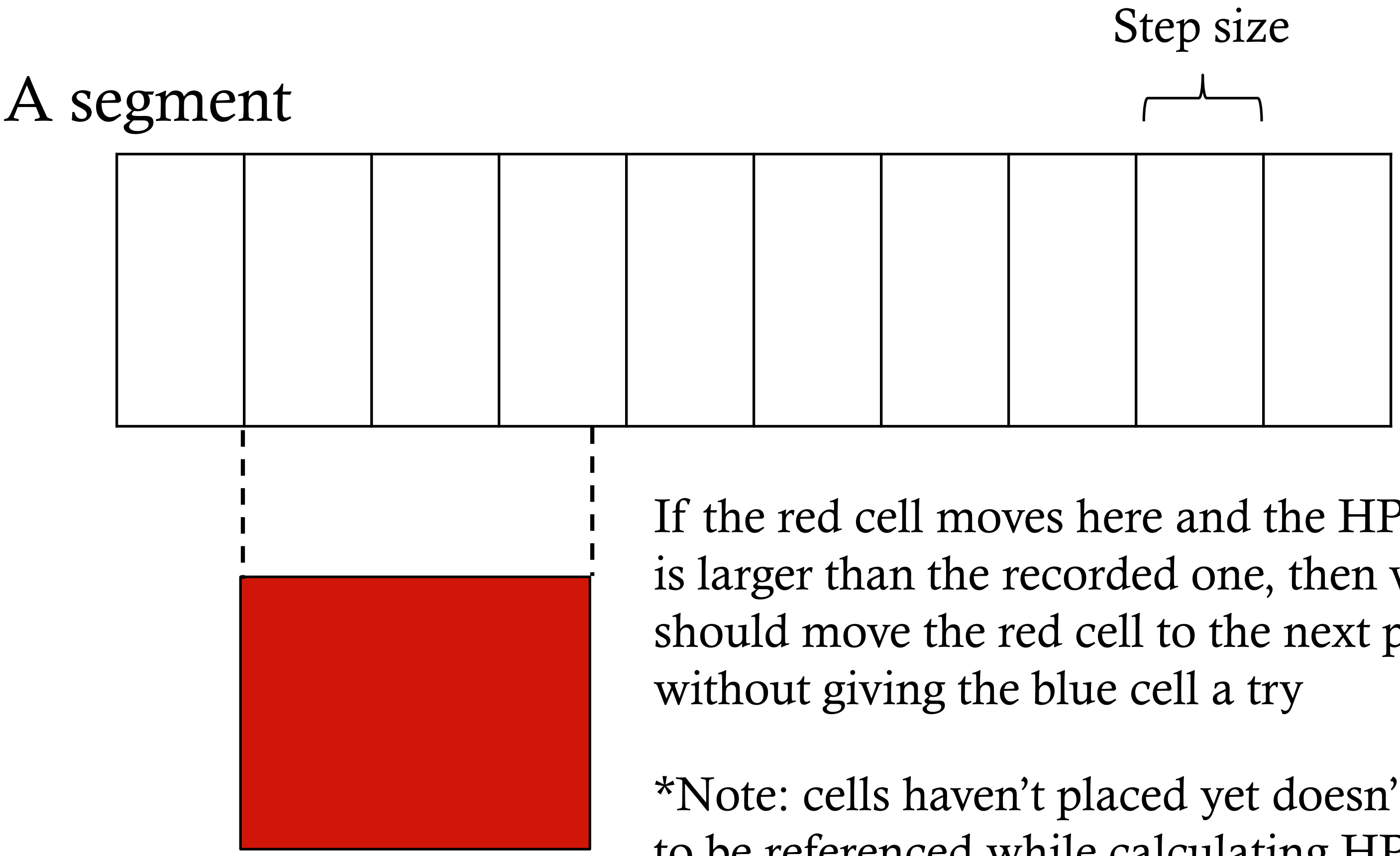
Detailed Placement (6/9)

- For each segment, do DFS branch-and-bound (red→blue)
- Initialize "best_WL" by the HPWL before DP


Step size
A segment
Then the blue cell can be placed here
When the red cell goes here

Detailed Placement (7/9)

- For each segment, do DFS branch-and-bound (red→blue)
- Initialize “best_WL” by the HPWL before DP

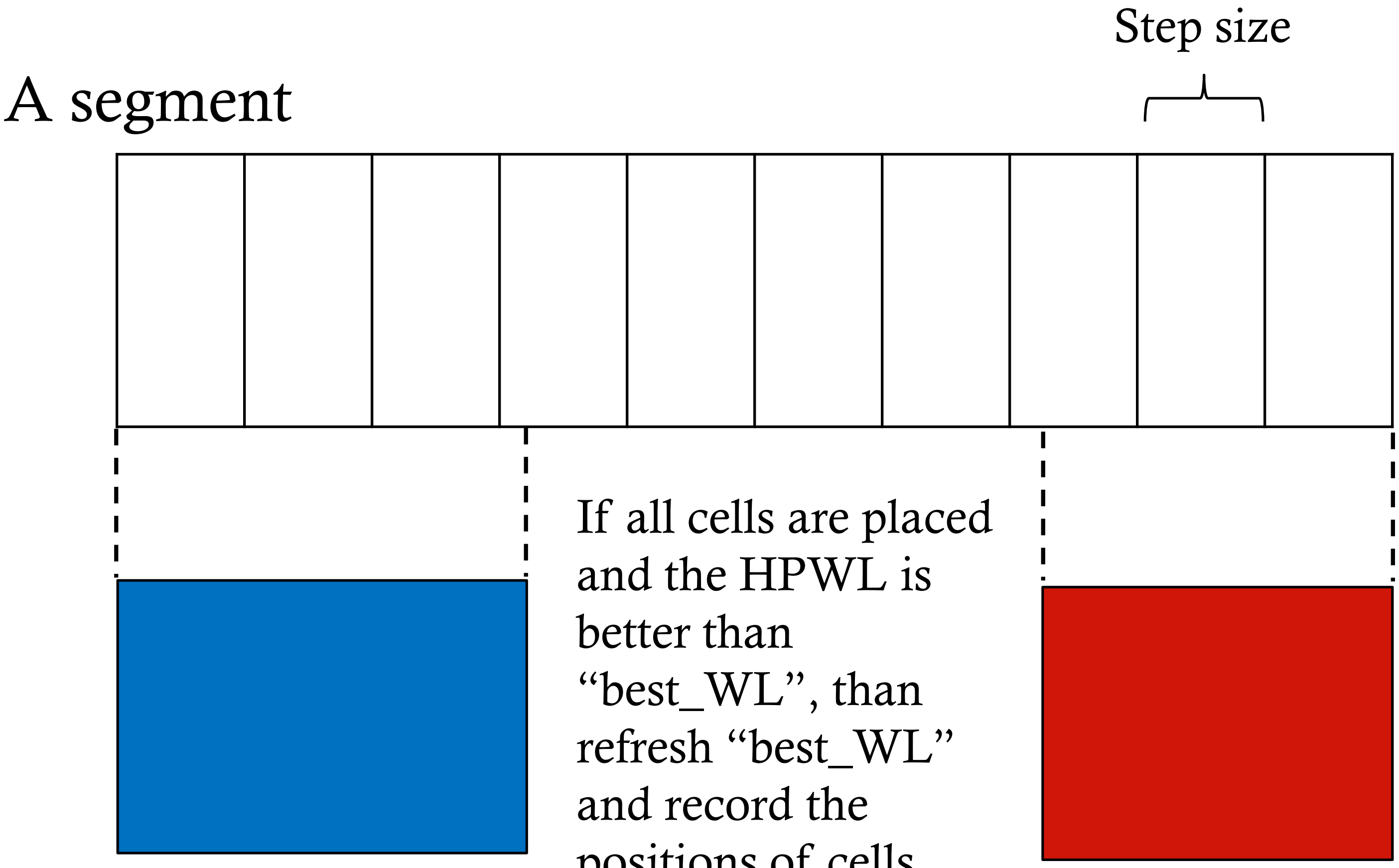


If the red cell moves here and the HPWL is larger than the recorded one, then we should move the red cell to the next place without giving the blue cell a try

*Note: cells haven’t placed yet doesn’t need to be referenced while calculating HPWL

Detailed Placement (8/9)

- For each segment, do DFS branch-and-bound (red→blue)
- Initialize “best_WL” by the HPWL before DP



If all cells are placed and the HPWL is better than “best_WL”, than refresh “best_WL” and record the positions of cells

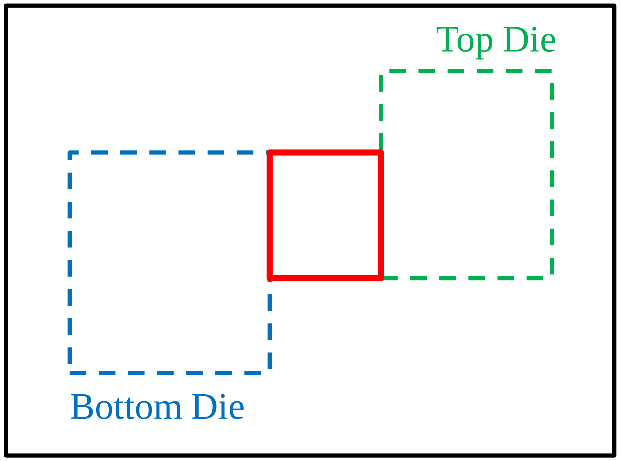
Detailed Placement (9/9)

- **Result (final HPWL)**
- **Running detailed placement for 1 iteration**

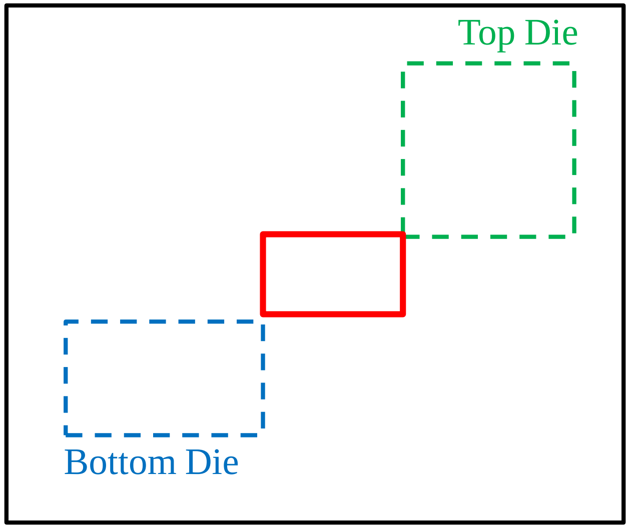
	Without DP	With DP	reduction
case1	123	118	4%
case2	10272799	10076549	2.5%

Terminal Placement (1/2)

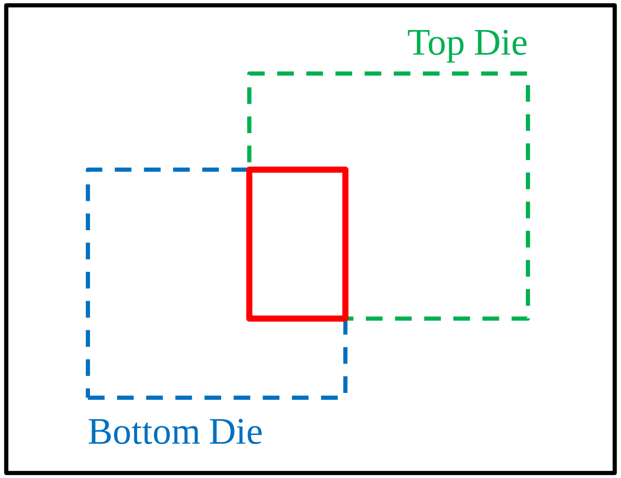
- Find the best boundary to insert terminal
  - Scenario1: no axis overlap
  - Scenario2: one axis overlap
  - Scenario3: two axes overlap



one axis overlap



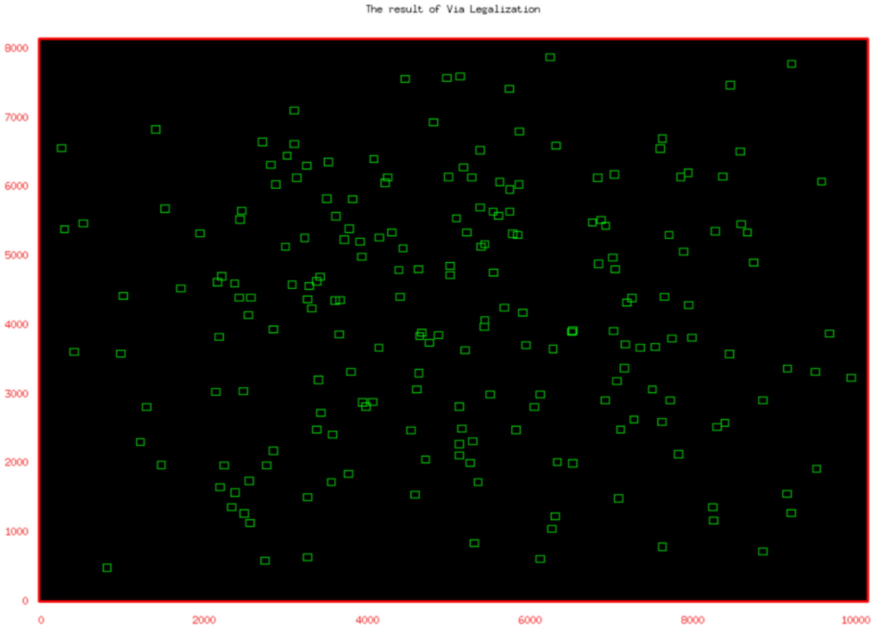
no axis overlap

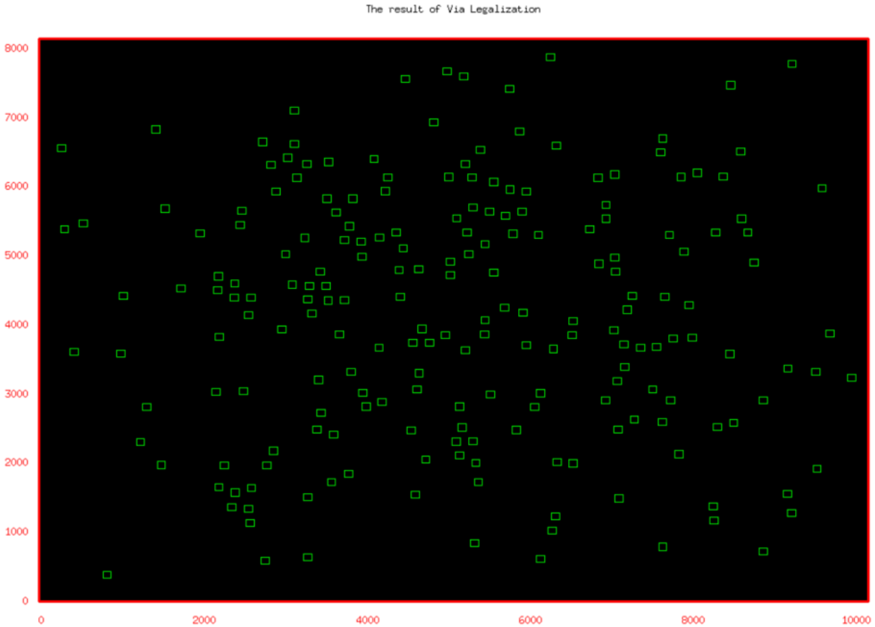


two axes overlap

Terminal Placement (2/2)

- **Remove overlap between terminals**
  - **Build horizontal and vertical constraint graph**
  - **Transform to non-overlapping constraints**
  - **Solved by linear programming**
    - **Minimizing total displacement**





lp_solve reference guide (sourceforge.net)

Thank you for listening!