An Effective Legalization Algorithm for Mixed-Cell-Height Standard Cells

C.-H. Wang, Y.-Y. Wu, J.-L. Chen, Y.-W. Chang
S.-Y. Kuo, W.-X. Zhu, G.-H. Fan

Jan. 18, 2017

Dependable Distributed Systems and Networks Laboratory
Graduate Institute of Electrical Engineering

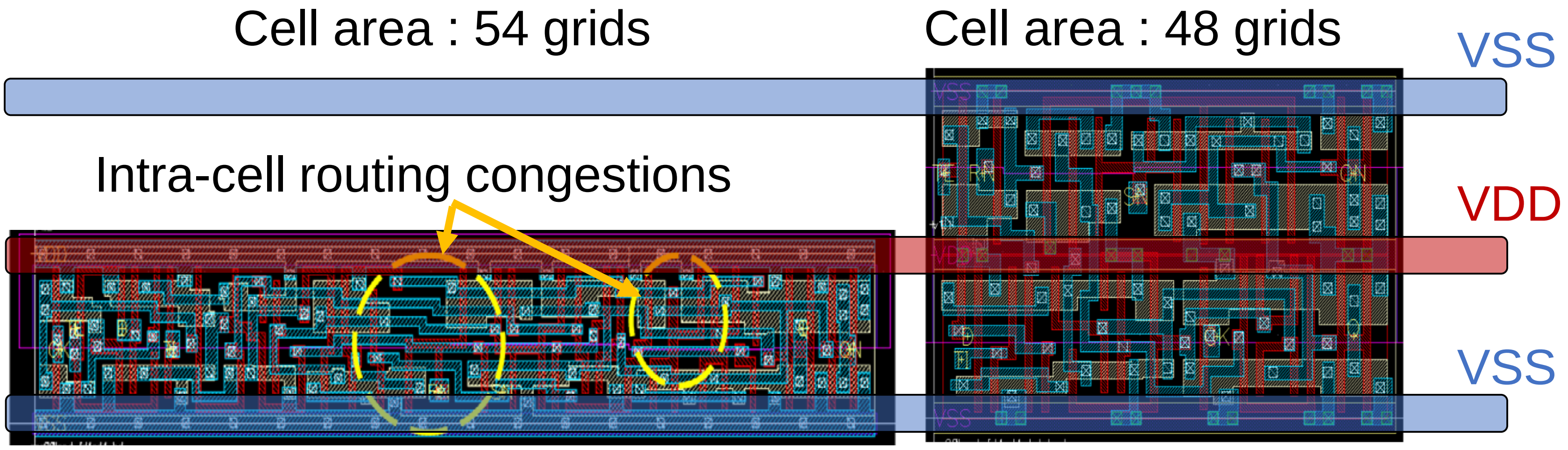
臺灣大學
National Taiwan University

Outline

- Introduction
- Previous Works
- Proposed Method
- Experimental Results
- Conclusions

Multiple-Row-Height Cells

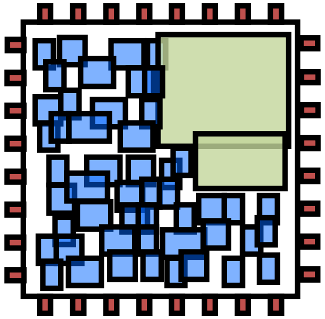
- Multiple-row-height cells [Beak et al., *SPIE'08*]
  - Minimize cell area
  - Minimize intra-cell routing space



Single-row-height & double-row-height cells with an identical function

Modern Placement Flow

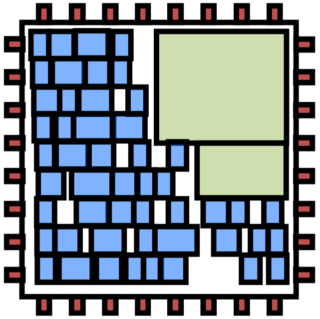
Global Placement



Compute the best cells positions, ignoring cells overlaps

↓

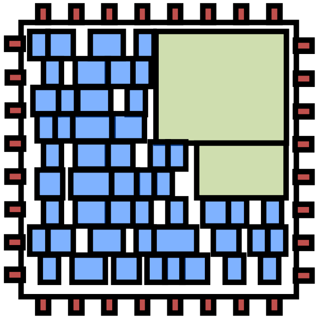
Legalization



Place cells into rows & removes all overlaps among cells

↓

Detailed Placement

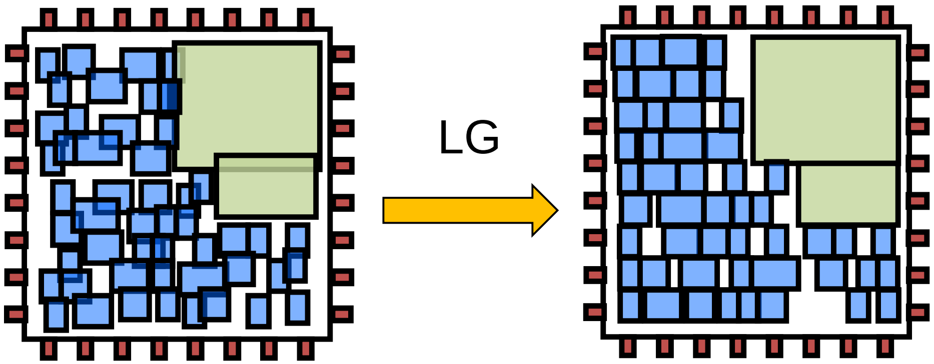


Refines the solution

Legalization

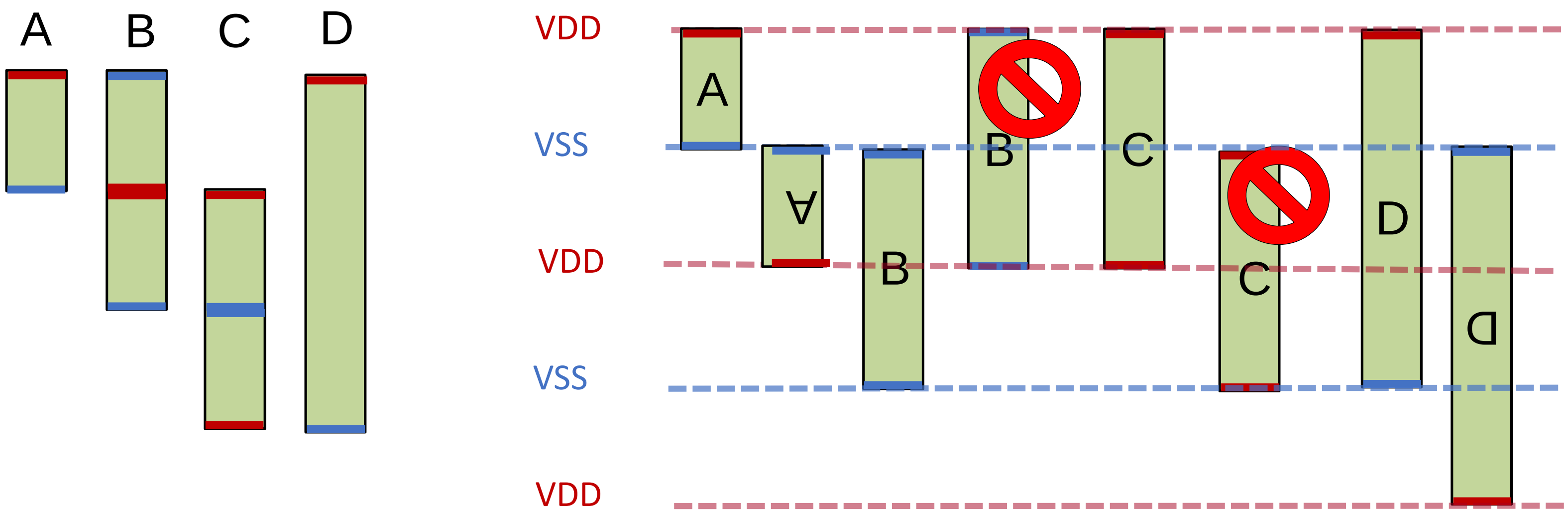
- Given
  - A global placement result with cell location (x_i', y_i')
- Objective
  - Minimize $\sum_{i=1}^{n}(|x_i - x_i'| + |y_i - y_i'|)$ (total displacement)

    (x_i, y_i): legal location of cell i
- Constraints
  - No cell overlaps & cells are in the chip
  - Cells are aligned in rows



Power-rail Alignment

- Power-rail alignment issues
  - An odd-row-height cell alignment can be achieved by vertical cell flipping
  - Two types of even-row-height cells: VDD or VSS runs along its top and bottom boundaries

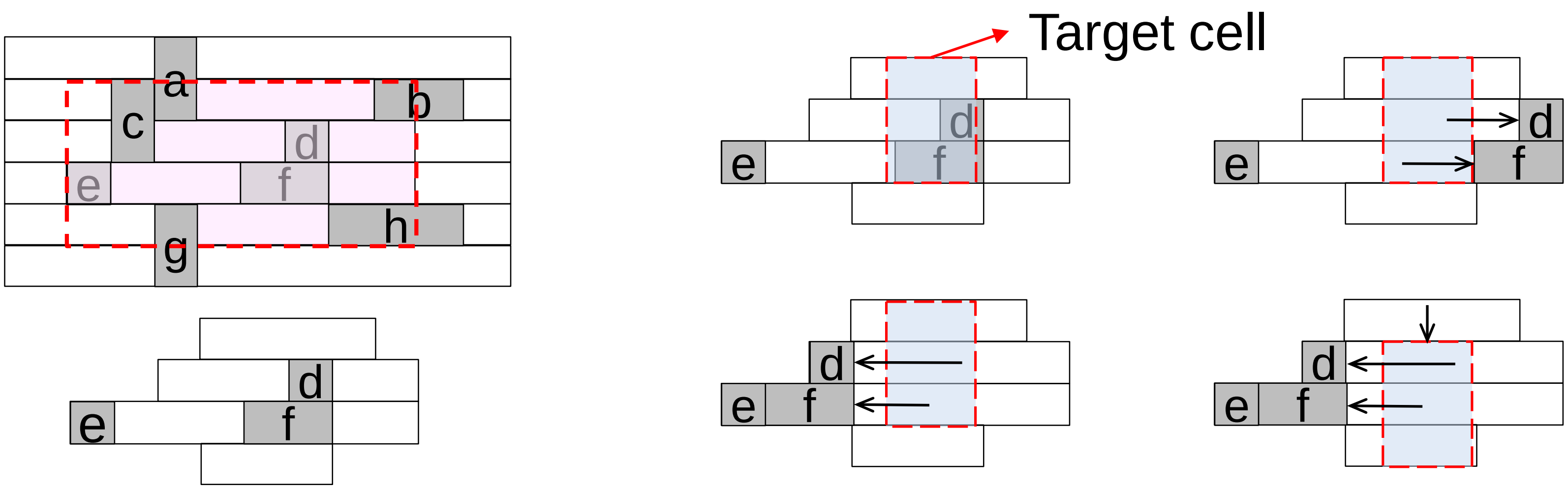


Outline

- Introduction
- **Previous Works**
- Proposed Method
- Experimental Results
- Conclusions

Window-based Legalization

- Cells partially inside the window are *non-local cells*
- Rows are divided by *non-local* cells into continuous segments
- Find insertion intervals to insert a triple-row-height cell [Chow et al., *DAC'16*]

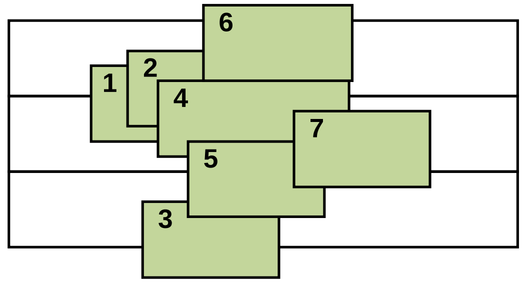


Single-Row-Height Cells Legalization

- Not much work on legalization with multiple-row-height standard cells is reported in the literature; however, single-row-height standard cells legalization has been studied for a long time
  - Tetris Legalization [Hill, *Patent 2002*]
  - Abacus Legalization [Spindler et al., *ISPD'08*]

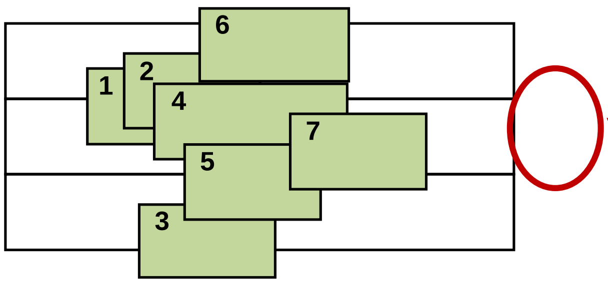
[ISPD'08] Abacus: Fast Legalization of Standard Cell Circuits with Minimal Movement
[Patent 2002] Method and System for High Speed Detailed Placement of Cells within an Integrated Circuit Design

Modern Legalizers: Tetris vs. Abacus

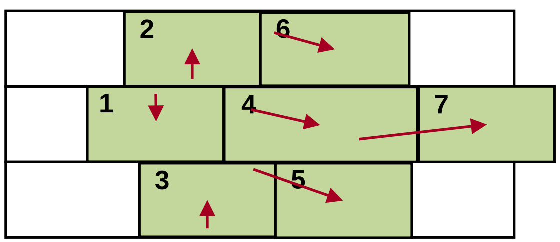


Global placement

Tetris



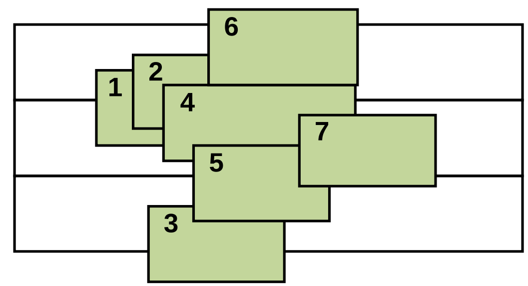
Violation

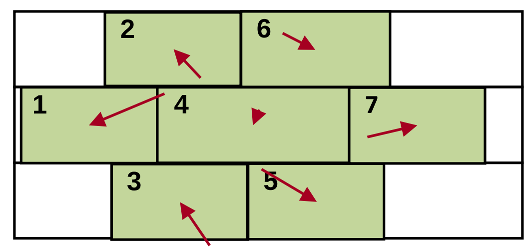


Greedy heuristic
⇒ Larger displacement

Abacus

Small displacement w/o violation





Dynamic programming
⇒ Smaller displacement

Abacus Legalization

- Legalize a cell at a time from left to right in a row
- Minimize quadratic displacement (QD)

Displacement weight for module *i*

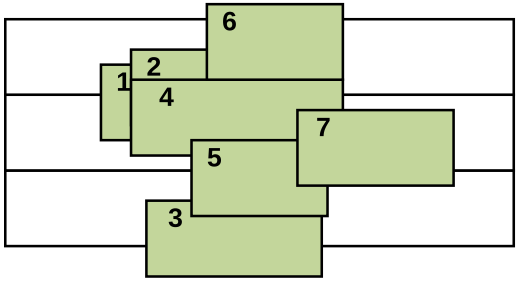
QD: $\min \sum_{i=1}^{N_r} e_i (x_i - x_i')^2$

Legal x-pos Global x-pos

Abutment constraint in one row

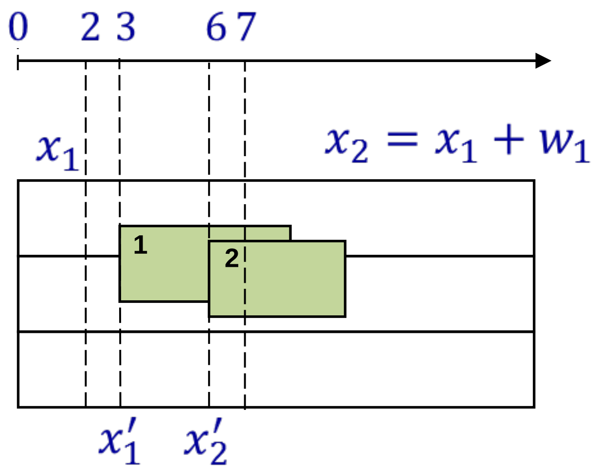
s.t. $x_i \geq x_{i-1} + w_{i-1}$

s.t. $x_i = x_{i-1} + w_{i-1}$



Abacus Legalization

- Legalize a cell at a time from left to right in a row



$$\text{QD: min } (x_1 - \underset{3}{x_1'})^2 + (x_2 - \underset{6}{x_2'})^2 \quad\Rightarrow\quad \text{QD: min } 2\left(x_1 - \frac{x_1' + (x_2' - w_1)}{2}\right)^2$$

$$\text{s.t. } x_2 \geq x_1 + \underset{5}{w_1} \quad\Rightarrow\quad x_2 = x_1 + w_1$$

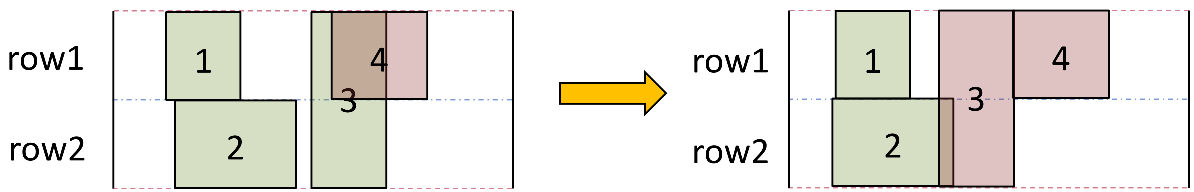
Inequality to equality

Unknown legal x-pos

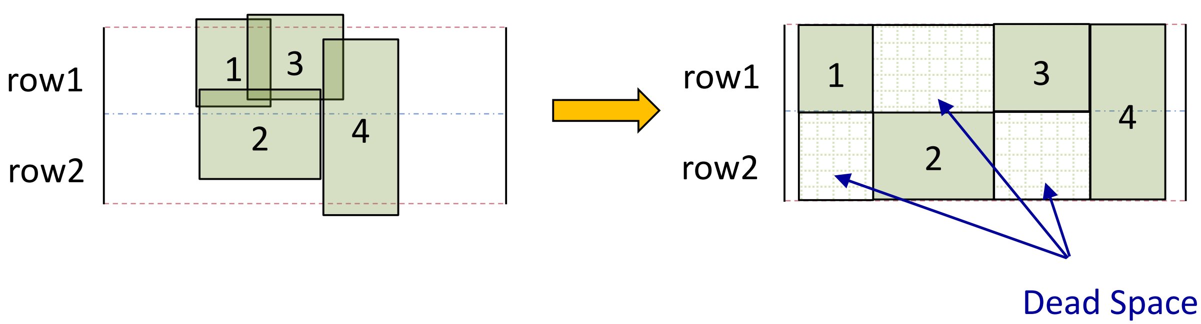
$$\Rightarrow x_1 = \frac{3 + (6 - 5)}{2} = 2, x_2 = 7$$

Abacus Extension

- Single-row Abacus ignores other rows



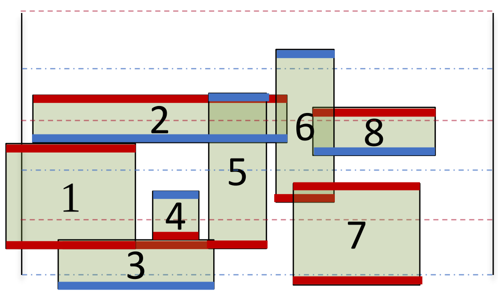
- If we consider the maximal cell height as a new row, we extend the dynamic programming method directly. It causes large number of dead space



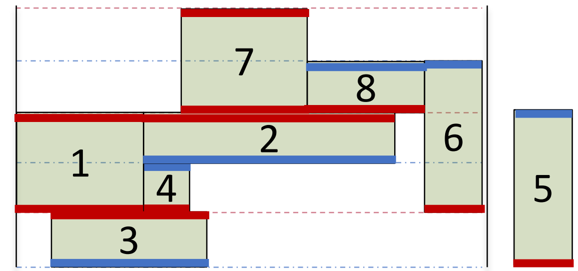
Insufficiency of Previous Works

- Window-based legalization insufficiency
  - Decide the order of cells arbitrarily
  - Find consecutive vacant rows difficultly
  - Incur large displacement when moving to other window
- Abacus insufficiency
  - Inhibit legalization in one row when cells legalized in another row
  - Incur vertical overlap chain

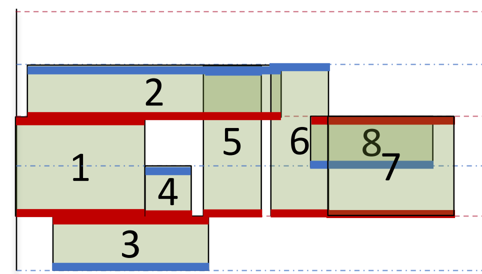
Previous Works Comparisons



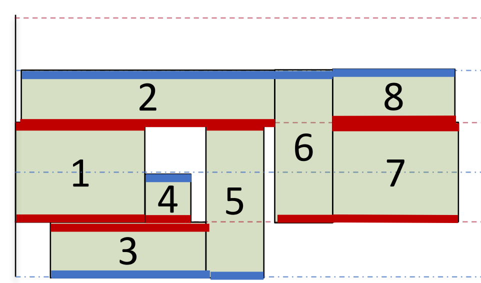
Global placement result



Window-based legalization



Abacus legalization



Our legalization

Contributions

- Propose a multiple-row-height cells legalization based on Abacus
  - Remedy Abacus' insufficiencies & extend its advantages
- Introduce a dead-space-aware cost function for multiple-row-height cells legalization
- Achieve a smaller displacement compared to a leading academic legalizer
  - About 50% smaller ΔHPWL than the state-of-the-art work

Outline

Introduction

Previous Works

Proposed Method

Experimental Results

Conclusions

Problem Formulation

- Given
  - A global placement result with multiple-row-height cells in location (x_i', y_i')
- Objective
  - Minimize $$\sum_{i=1}^{n}(|x_i - x_i'| + |y_i - y_i'|) + D \times \alpha$$

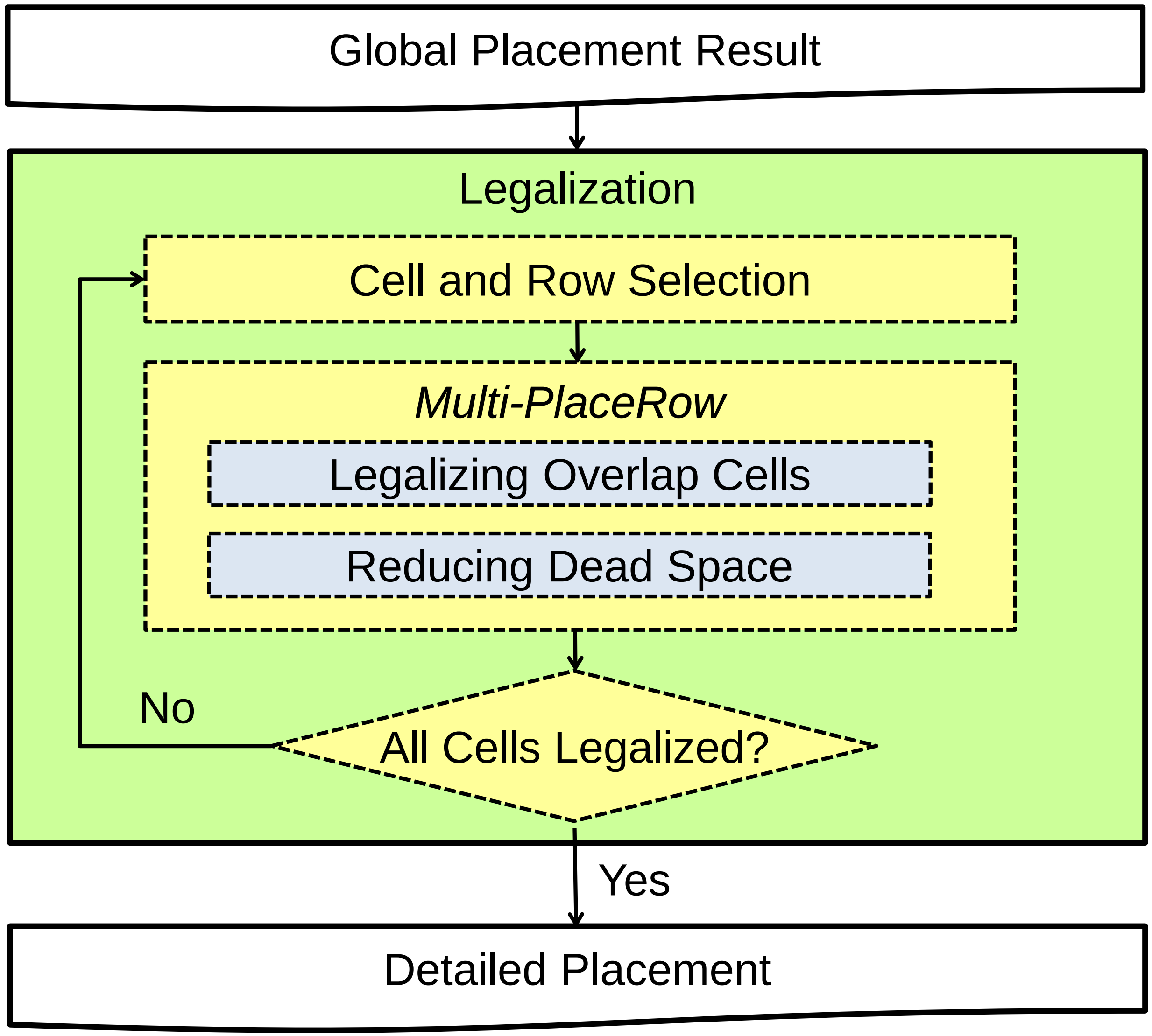
(x_i, y_i): legal location of cell i

D: total dead space

α: user-defined parameter

- Constraints
  - No cell overlaps & cells are in the chip
  - Cells are aligned in rows
  - Cells orient to power rails with VDD/VSS constraints

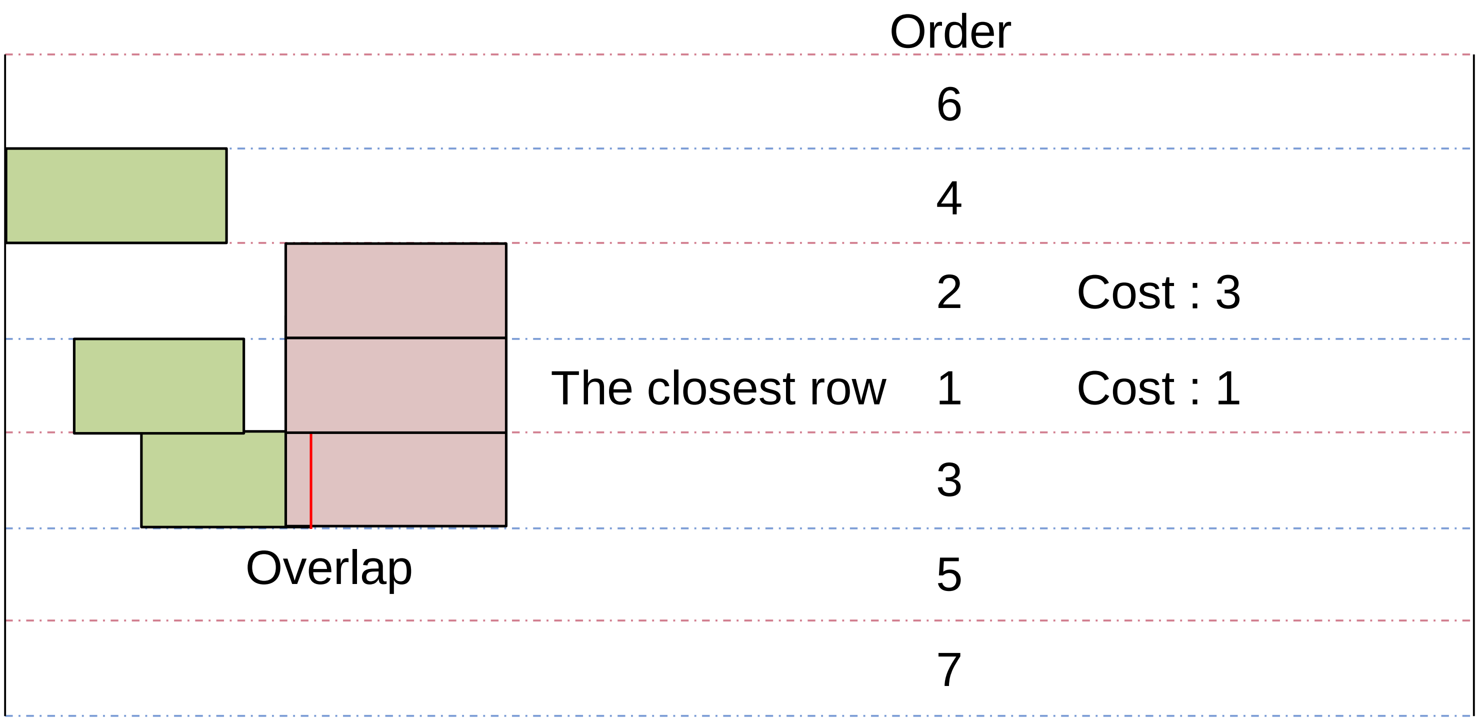
Multi-Height Cells Legalization



Cell and Row Selection

- Order the cells starting from the edge of the cells going to be legalized
- Find the closest row and find the corresponding cost

- The green cells has been legalized, and the closest row
- Try every row



PlaceRow Method

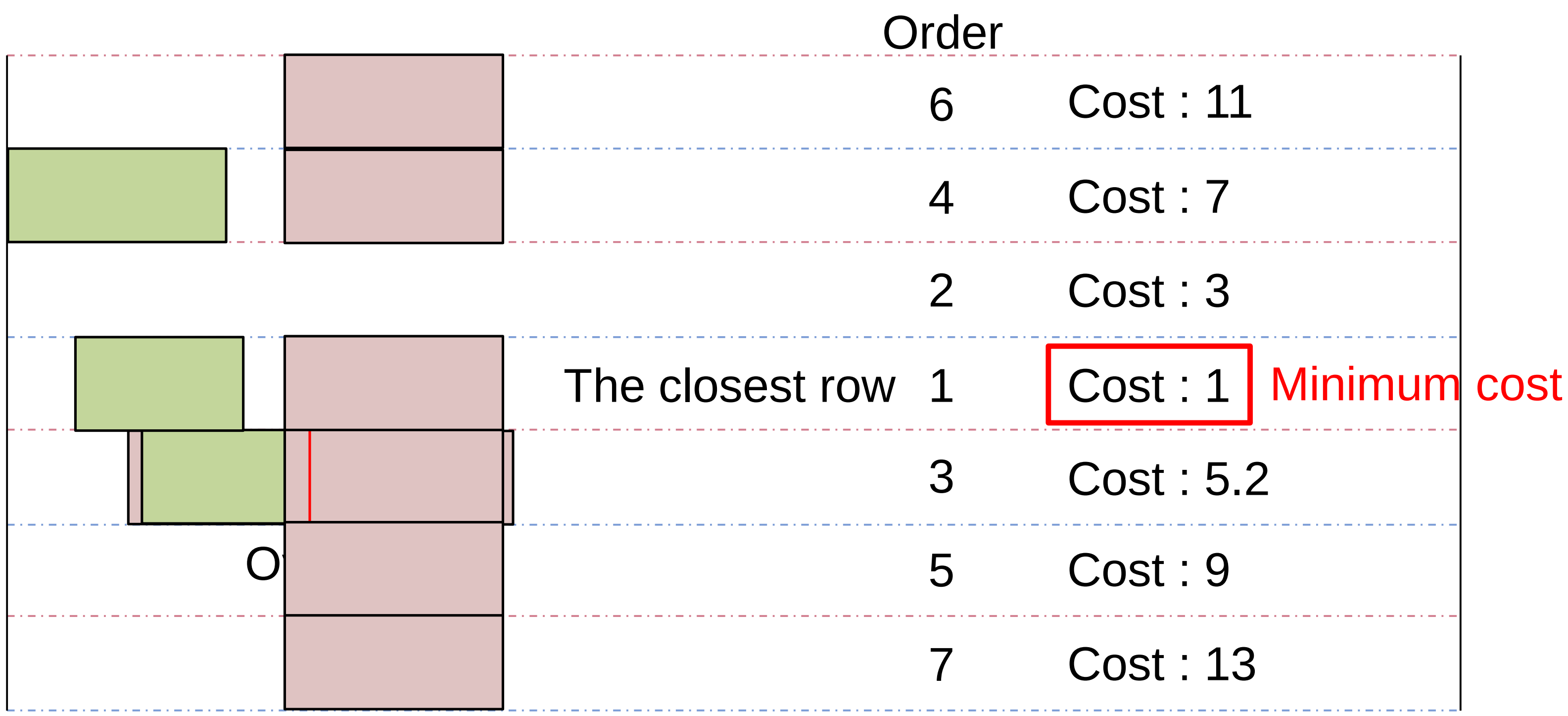
- Solve QD by dynamic programming approach:
  - solve sub problems optimally to obtain the final solution

Cell 2 overlaps with previous cell

Cluster cell 1 and cell 2 and move the cluster to new global x-pos

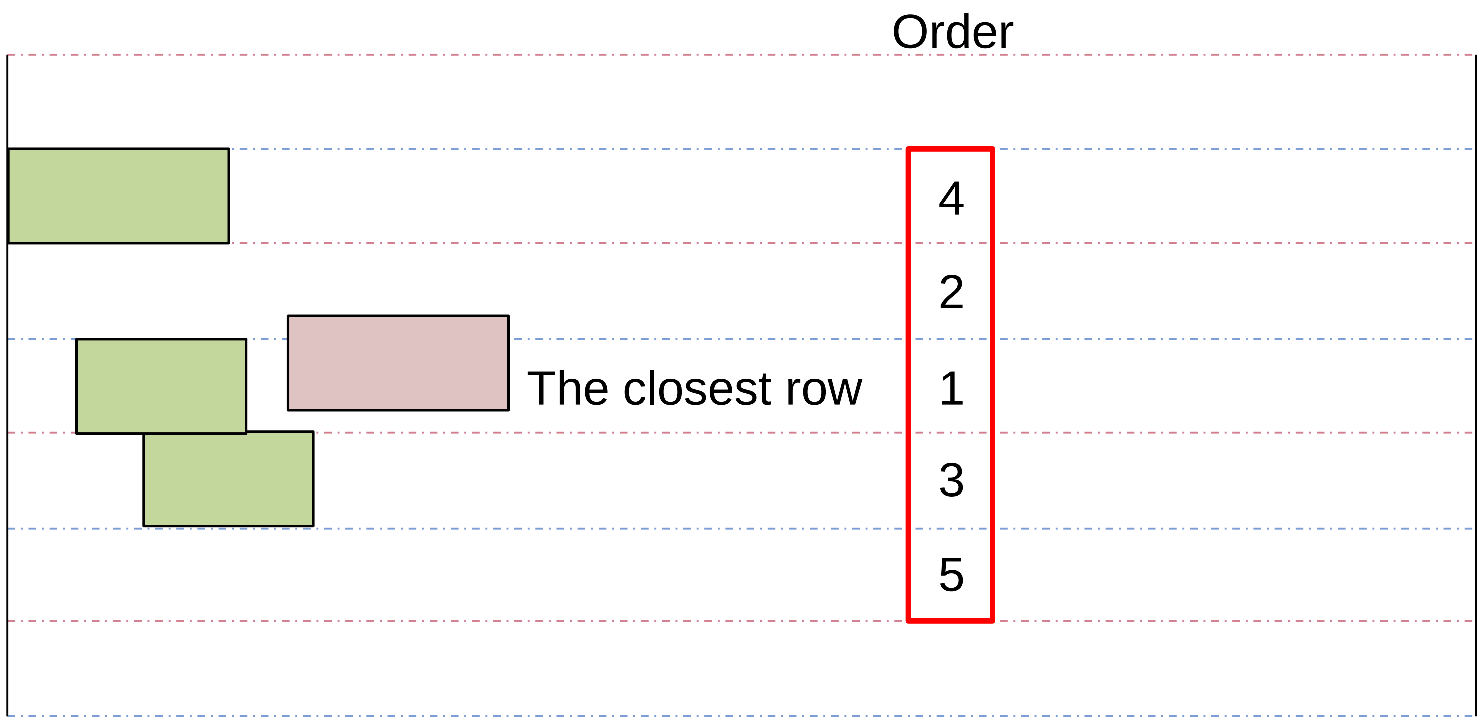
Minimum Cost Selection

- Legalize the overlap cells, and get the cost
- Select the minimum cost row



Execution Time Reduction

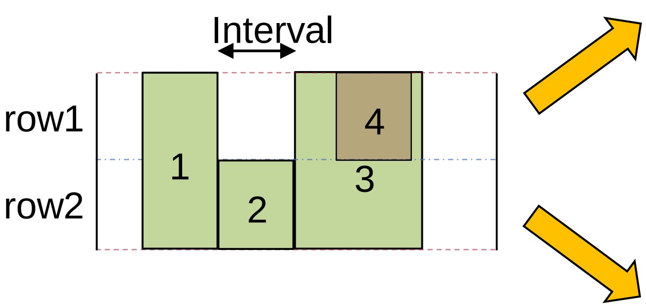
- Speed up this process by only placing cells into their neighboring rows

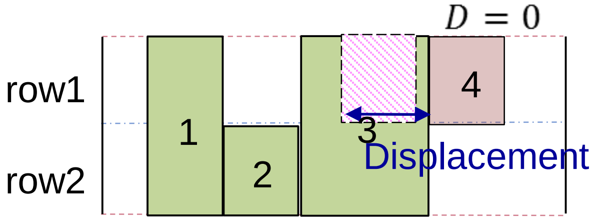


Dead Space Consideration

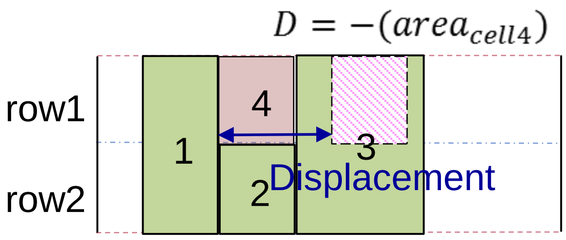
- Add the area of dead space into the cost function
  - $\min \sum_{i=1}^{n} (|x_i - x_i'| + |y_i - y_i'|) + \alpha \times D$

    α is user-defined parameter
    D is total dead space

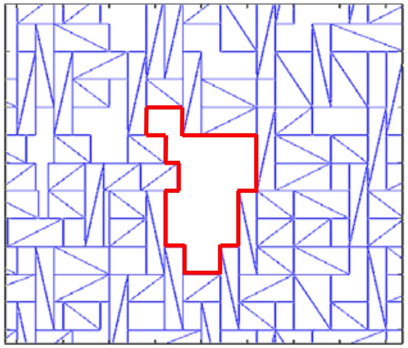




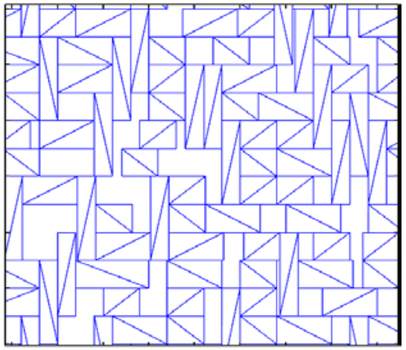
Without placing into interval



Placing into interval



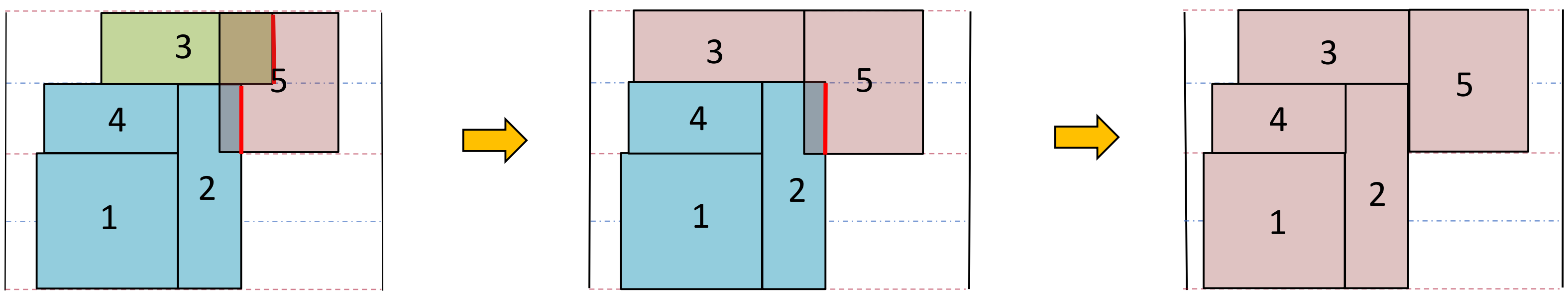
Original cost function



Dead space-aware cost function

Multiple Overlaps Solution

- Choose the row with maximum overlapping area when overlapping with previous cells in different rows
- Do *Multi-PlaceRow* on that row



Cell 5 overlaps with cell 3 and cell 2

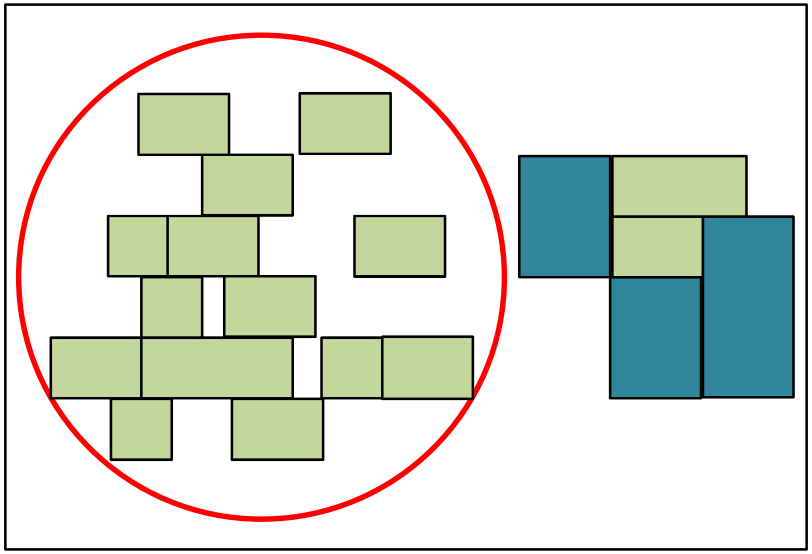
Cell 5 clusters with cell 3

Clustered cells then cluster with cell 2

Multi-PlaceRow Analysis

- If every multiple-row-height cell is never clustered and only the horizontal movement is considered, the solution generated by *Multi-PlaceRow* is optimal

Keep optimal



Outline

- Introduction
- Previous Works
- Proposed Method
- **Experimental Results**
- Conclusions

Experimental Settings

- Platform
  - C++ programming language
  - 64-bit Linux machine
  - Intel Xeon 2.93 GHz CPU with 48GB memory
- Comparison
  - Comparison with Chow et al., *DAC'16*
- Benchmarks
  - Same designs adapted by Chow et al.

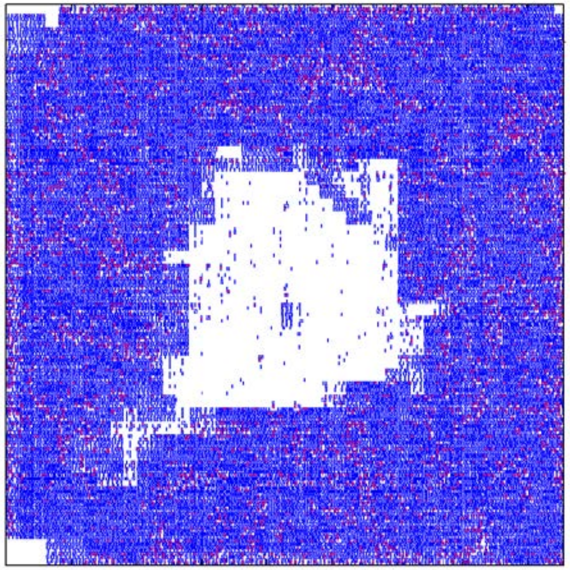
Experimental Results

- 52% smaller ΔHPWL than the ILP method
- 59% smaller ΔHPWL than the state-of-the-art work

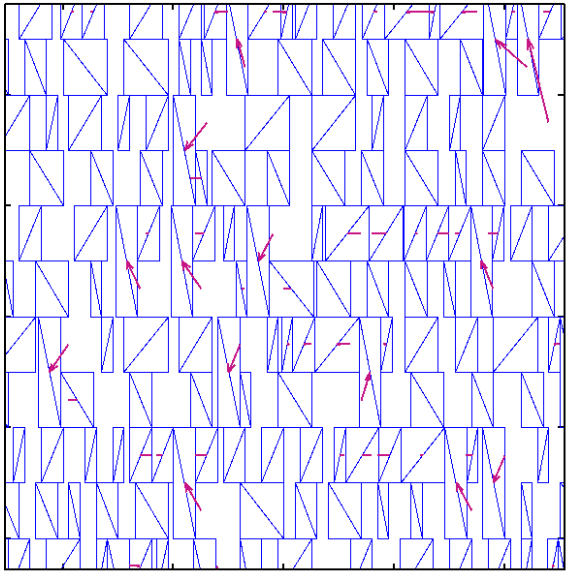
Benchmark	#S. Cell	#D. Cell	Density	GP HPWL (m)	Disp. (sites)			ΔHPWL			Runtime (s)		
					ILP	DAC'16	Ours	ILP	DAC'16	Ours	ILP	DAC'16	Ours
des_perf_1	103842	8802	0.91	1.43	2.13	3.32	3.46	2.61%	2.85%	0.96%	4098.7	7.0	18.0
des_perf_a	99775	8513	0.43	2.57	0.66	0.96	0.68	0.11%	0.28%	0.14%	193.8	2.6	6.1
des_perf_b	103842	8802	0.50	2.13	0.62	0.85	0.64	0.12%	0.31%	0.16%	250.8	2.4	6.2
fft_1	121913	5500	0.46	5.25	0.45	0.47	0.47	0.09%	0.10%	0.10%	206.0	1.9	7.8
fft_2	30297	1984	0.84	0.46	1.58	1.81	1.55	2.25%	1.66%	0.93%	776.8	1.1	1.3
fft_a	30297	1984	0.50	0.46	0.66	0.86	0.64	0.55%	0.87%	0.68%	72.7	0.4	0.8
fft_b	28718	1907	0.25	0.75	0.60	0.64	0.64	0.32%	0.33%	0.33%	38.2	0.3	0.8
matrix_mult_1	28718	1907	0.28	0.95	0.73	0.80	0.62	0.32%	0.33%	0.27%	61.9	0.4	1.0
matrix_mult_2	152427	2898	0.80	2.39	0.49	0.53	0.48	0.36%	0.28%	0.22%	967.4	3.9	9.1
matrix_mult_a	152427	2898	0.79	2.59	0.45	0.49	0.44	0.30%	0.22%	0.17%	825.0	4.0	8.9
matrix_mult_b	146837	2813	0.42	3.77	0.27	0.33	0.27	0.09%	0.14%	0.09%	150.7	1.3	9.3
matrix_mult_c	143695	2740	0.31	3.43	0.25	0.30	0.25	0.09%	0.13%	0.09%	127.8	1.3	8.9
pci_bridge32_a	143695	2740	0.31	3.29	0.27	0.29	0.27	0.11%	0.11%	0.11%	139.0	1.4	9.0
pci_bridge32_b	26268	3249	0.38	0.46	0.88	0.95	0.88	0.52%	0.58%	0.63%	49.4	0.3	0.8
superblue12	25734	3180	0.14	0.98	0.95	0.96	0.52	0.12%	0.13%	0.12%	15.3	0.2	0.7
superblue11_a	861314	64302	0.43	42.94	1.85	1.94	1.86	0.15%	0.15%	0.14%	3073.6	23.4	80.2
superblue12	1172586	114362	0.45	39.23	1.45	1.63	1.63	0.18%	0.22%	0.25%	5079.0	106.5	91.1
superblue14	564769	47474	0.56	27.98	2.56	2.62	2.38	0.22%	0.22%	0.16%	3360.6	17.1	70.3
superblue16_a	625419	55031	0.48	31.35	1.61	1.73	1.68	0.10%	0.12%	0.11%	2470.7	21.7	61.0
superblue19	478109	27988	0.52	20.76	1.52	1.60	1.68	0.14%	0.14%	0.11%	1848.8	10.9	44.6
				Average	**1.00**	**1.16**	**1.05**	**0.44%**	**0.46%**	**0.29%**	**1190.3**	**10.4**	**21.8**
				N. Average	**0.95**	**1.11**	**1.00**	**1.52**	**1.59**	**1.00**	**54.6**	**0.48**	**1.0**

Layout

- The benchmark circuit fft_2



Legalization result



A partial layout of legalization result

Outline

Introduction

Previous Works

Proposed Method

Experimental Results

Conclusions

Conclusions

- Develop an effective algorithm for the mixed-cell-height standard cells legalization problem
- Derive a dead-space-aware objective function and an optimization scheme to handle this issue
- Achieve best wirelength among all published methods in reasonable running time as shown in experiment results, e.g., about 50% smaller wirelength increase than the state-of-the-art work which is a best paper nominee at DAC'16

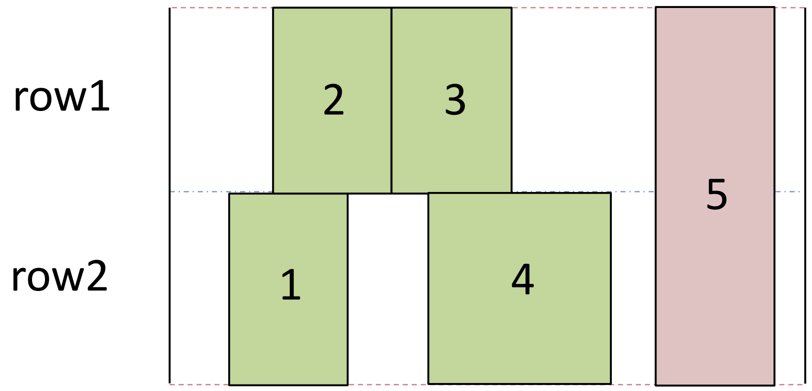
Thank You!

National Taiwan University

Backup

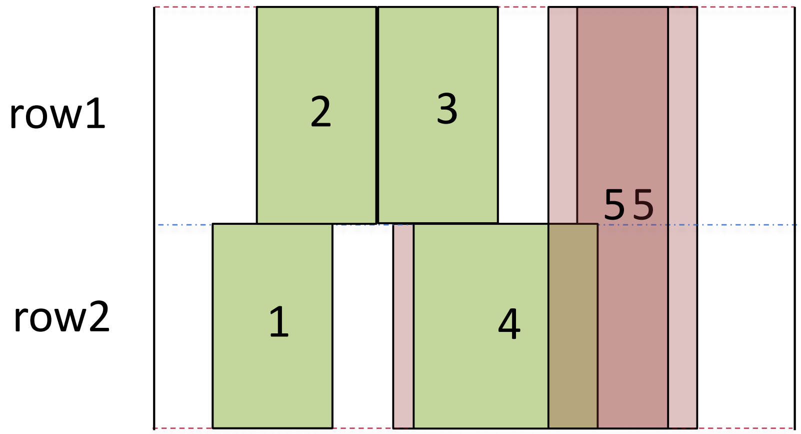
Multi-PlaceRow Analysis — Case I

- The to-be-inserted cell 5 keeps its position while there is no overlap



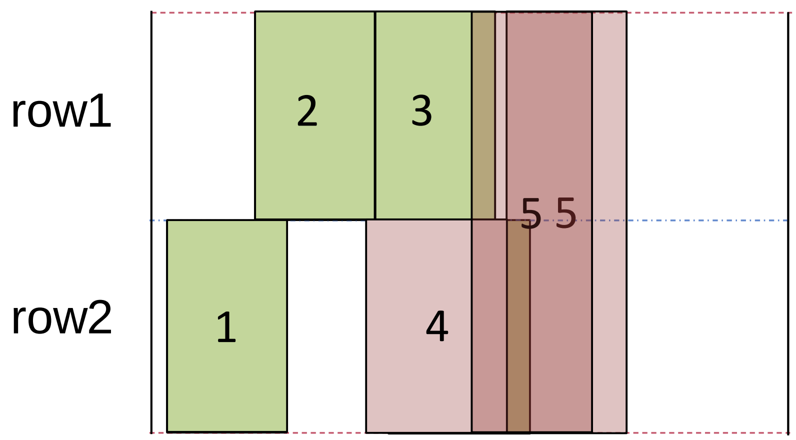
Multi-PlaceRow Analysis — Case II

- The to-be-inserted cell 5 overlaps with previous cell 4 in row 2



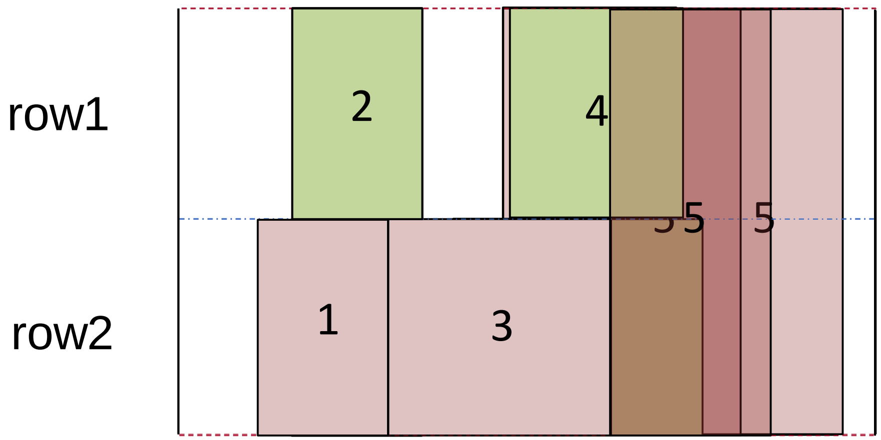
Multi-PlaceRow Analysis — Case III-A

- The to-be-inserted cell 5 overlaps with previous cell 4 in row 2 and cell 3 in row 1



Multi-PlaceRow Analysis — Case III-B

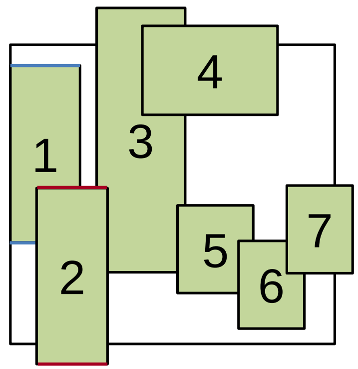
- The overlapping area between cell 3 and cell 5 is larger than that between cell 4 and cell 5
  - Cluster cell 5, cell 3, and cell 1
- After the moving, cell 5 overlaps cell 4
  - Add cell 4 to the previous cluster
- After all cells are non-overlapping, cell 1 exits its cluster to restore to original position

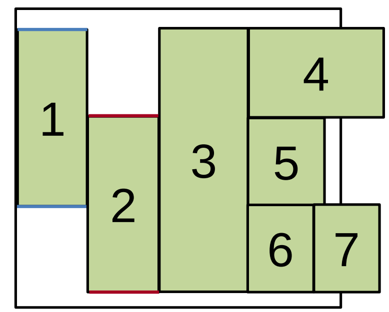


Future Work: Order of Cells Determination

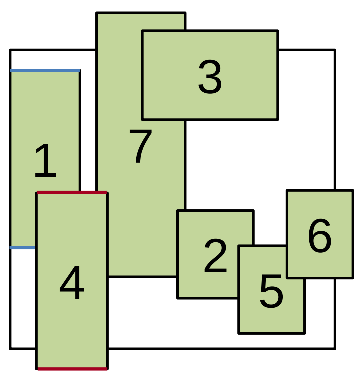
- Determine the legalizing order of cells before legalization, instead of deciding order only by x-coordinate

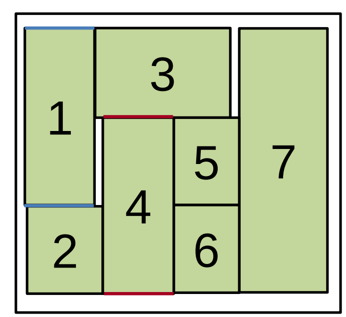
Determine by x-coordinate





Determine by preprocess





Future Work: Vertical Power Rail Awareness

- Vertical power rails are usually implemented by Metal 2 in process, and some designs also use Metal 2 to implement the power rails of multiple-row-height cells
- We should avoid vertical power rails overlapping with the power rails of multiple-row-height cells