International Symposium on Physical Design

# LEGALM: Efficient Legalization for Mixed-Cell-Height Circuits with Linearized Augmented Lagrangian Method

**Jing Mai**[1], Chuanyuan Zhao[1], Zuodong Zhang[1], Zhixiong Di[2], Yibo Lin[1], Runsheng Wang[1], Ru Huang[1]

[1]Peking University

[2]Southeast Jiaotong University

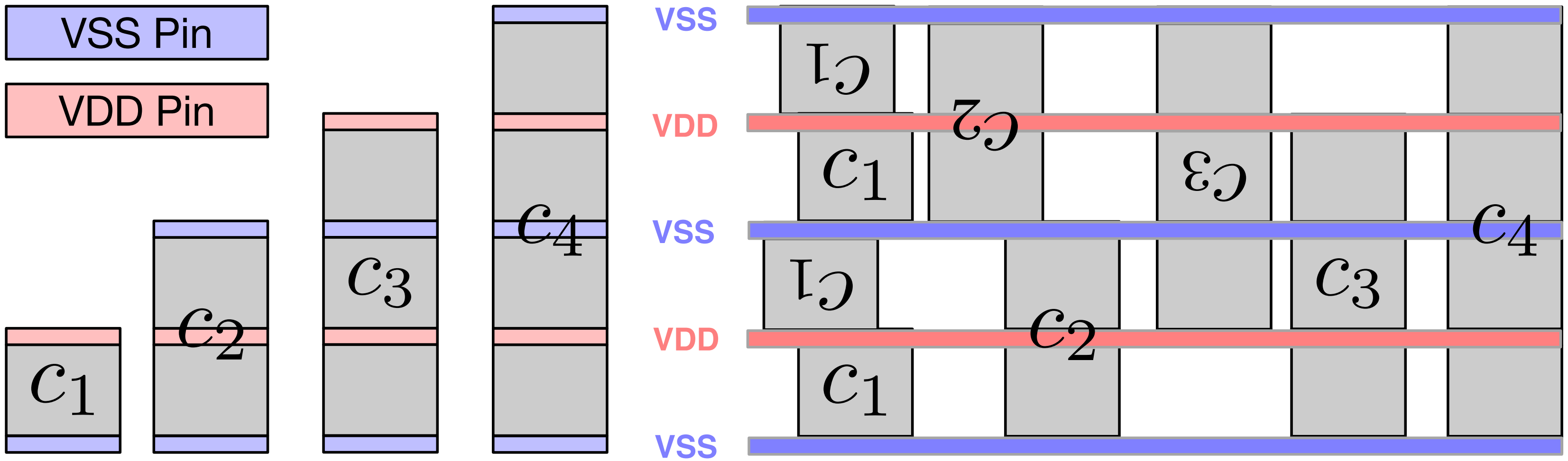
Mar 17, 2025

# Outline

# Introduction

# Mixed-Cell-Height Circuits

- Recent mixed-cell-height designs combine higher and smaller cells to optimize PPA in modern ASICs.
  - Higher cells boost performance and routability for critical paths.
  - Smaller cells improve area efficiency and reduce power for non-critical logic.



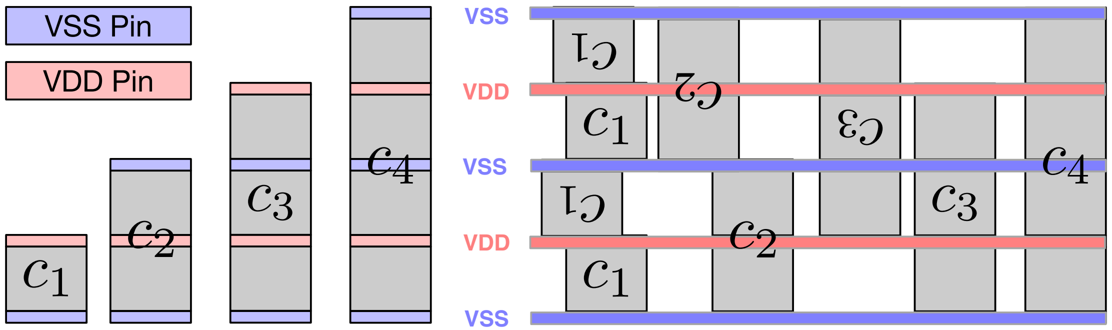
# Challenges & Motivation

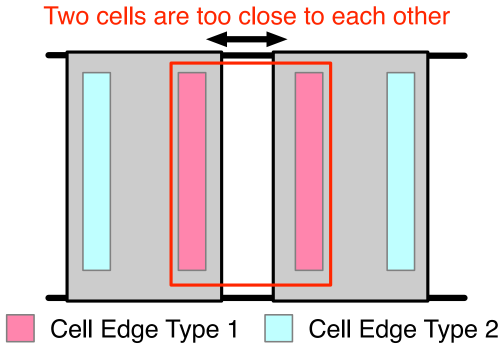
## ❑ Why Legalization matters:

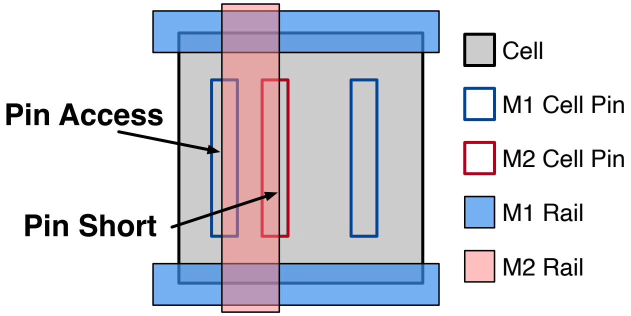
- Eliminates design rule violations post-global placement.
- Impacts downstream routing and performance.

## ❑ Challenges in Mixed-Cell-Height:

- Site alignment, overlap-free.
- Cross-row shapes, P/G alignment, fence regions, edge spacing, pin access.



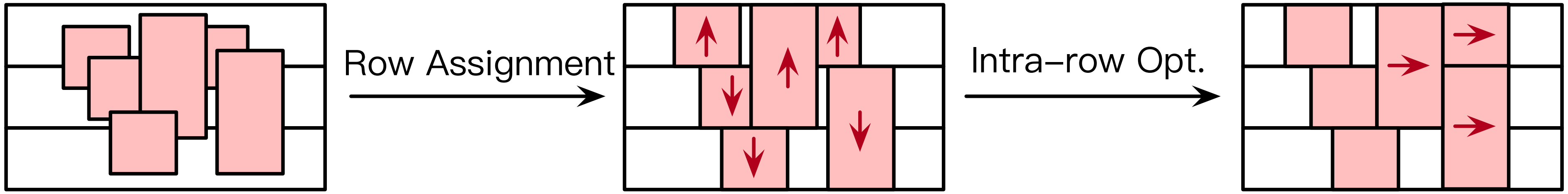



# Related Work

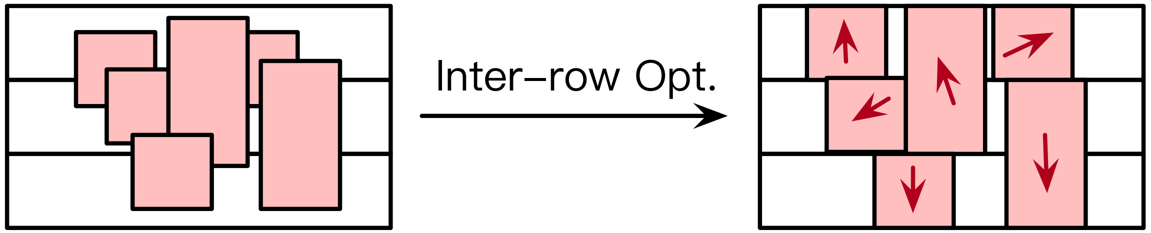
## Intra-row Methods:

- Row assignment + intra-row optimization algorithm (e.g., ILP [Li+,DAC'18], LCP [Chen+, DAC'17]).
- Limited vertical movement space during optimization.




## Inter-row Methods:

- Network flow / ILP-based (e.g., [Darav+, ISPD'17])
- High flexibility but computationally expensive.




Gap:

Existing methods struggle with complex constraints and scalability.

# Contribution

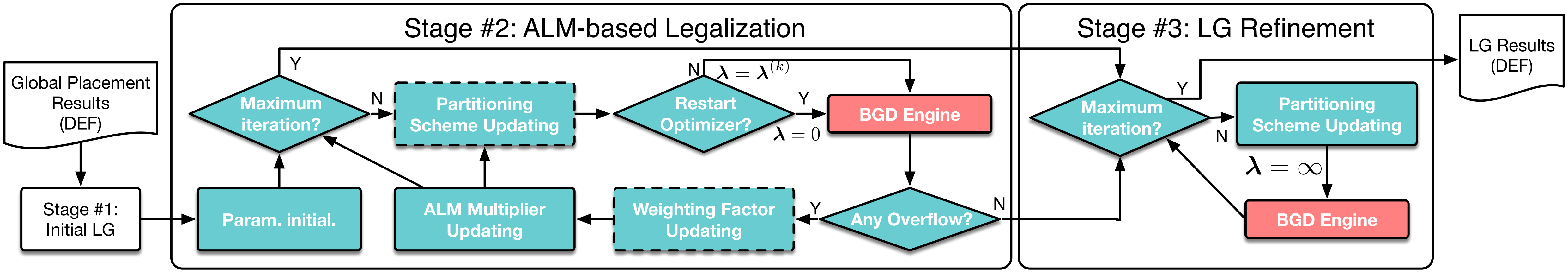
**LEGALM**, an efficient legalization method for mixed-cell-height circuits using the linearized augmented Lagrangian method.

1. **Augmented Lagrangian Method (ALM)** for efficient vertical and horizontal cell movements.
2. **Block Gradient Descent (BGD)** for parallel cell updates.
3. **Triple-fold partitioning** for GPU efficiency. 94.2× speedup with less than 0.5% quality degradation.

**Results:** 6-36% better quality, 2.25-5.99× speedup on million-cell designs.

# The LEGALM Algorithm

# Framework




- **3-Stage Workflow:**
  1. **Initial Legalization**: Minimal displacement, ignore overlaps.
  2. **ALM-based Legalization**: Optimize displacement + eliminate overlaps.
  3. **Refinement**: Strict no-overlap optimization.

- **Key Components:**
  - Linearized ALM formulation for constraint relaxation.
  - BGD for parallel cell updates.
  - GPU-friendly partitioning.

# Augmented Lagrangian Formulation

- **Objective**: Minimize displacement while satisfying constraints.

$$
\begin{aligned}
\min \sum_{j\in S}\sum_{i\in N}\sum_{t\in\mathcal{T}_i} & w_{i,t,j} x_{i,t,j}, \\
s.t.\ x_{i,t,j} & \in \{0,1\}, \\
g_j(\boldsymbol{x}) & = \sum_{i\in N}\sum_{t\in\mathcal{T}_i} x_{i,t,j} - 1 \le 0, \quad \forall j \in S, \\
& \textit{connected sub-cell constraints}, \\
& \textit{fence region constraints},
\end{aligned}
$$

- $\mathcal{T}_i$ Partition cell $i$ into sub-cells of width 1 and height $H$ (row height).
- $x_{i,t,j}$ Binary indicator if sub-cell $t$ ($t \in \mathcal{T}_i$) of cell $i$ is placed at site $j$ ($j \in S$) .
- $w_{i,t,j}$ Displacement cost for placing sub-cell $t$ ($t \in \mathcal{T}_i$) at site $j$ ($j \in S$).
- $g_j(x)$ Overflow at site $j$ ($j \in S$).

# Augmented Lagrangian Formulation (cont'd)

- **Lagrangian relaxation** to handle overlap-free constraints.
- **Slack variables** $r_j$ for site overflow.

$$
\begin{aligned}
\mathcal{L}(\boldsymbol{x}, \boldsymbol{r}, \boldsymbol{\lambda}) = & \sum_{j \in S} \sum_{i \in N} \sum_{t \in \mathcal{T}_i} w_{i,t,j} x_{i,t,j} \\
& + \sum_{j \in S} \lambda_j \left[ (g_j(\boldsymbol{x}) + r_j) + \frac{\sigma}{2} (g_j(\boldsymbol{x}) + r_j)^2 \right] \\
& + I_{\mathcal{X}}(\boldsymbol{x}),
\end{aligned}
$$

- Solve for optimal $r_j$ using **elimination method**: $r_j = \max(0, -\frac{1}{\sigma} - g_j(\boldsymbol{x}))$.

- Iterative multiplier updates based on **KKT conditions**.

$$
\boldsymbol{x}^{(k+1)} = \operatorname*{argmin}_{\boldsymbol{x}} \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}^{(k)}) = \operatorname*{argmin}_{\boldsymbol{x}} \psi(\boldsymbol{x}, \boldsymbol{\lambda}^{(k)}) + I_{\mathcal{X}}(\boldsymbol{x}).
$$

$$
\lambda_j^{(k+1)} = \max\left( \lambda_j^{(k)} + h_f \cdot \left[ (g_j(\boldsymbol{x}) + r_j) + \frac{\sigma}{2} (g_j(\boldsymbol{x}) + r_j)^2 \right], 0 \right),
$$

# Linearized Proximal Gradient Method

$$\boldsymbol{x}^{(k+1)} = \underset{\boldsymbol{x}}{\operatorname{argmin}}\, \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}^{(k)}) = \underset{\boldsymbol{x}}{\operatorname{argmin}}\, \psi(\boldsymbol{x}, \boldsymbol{\lambda}^{(k)}) + I_{\mathcal{X}}(\boldsymbol{x}).$$

## Iterative Optimization:

- In each iteration, given fixed $\boldsymbol{\lambda}$, let $f(\boldsymbol{x}) = \psi(\boldsymbol{x}, \boldsymbol{\lambda})$, solve the following function

$$\min_{\boldsymbol{x}} f(\boldsymbol{x}) + I_{\mathcal{X}}(\boldsymbol{x}).$$

- Proximal Mapping Operator (can be proved a linear operator).

$$P_{\mathcal{X}}(\boldsymbol{v}) = \arg\min_{\boldsymbol{u}\in\mathcal{X}} \|\boldsymbol{u} - \boldsymbol{v}\|_2^2,$$

- Proximal Gradient Method.

$$\boldsymbol{x}^{(k+1)} \in P_{\mathcal{X}}\left(\boldsymbol{x}^{(k)} - \tau\nabla f(\boldsymbol{x}^{(k)})\right),$$

## Final Cost Function:

$$cost_{i,t,j} = w_{i,t,j} + \lambda_j \max\{1 + \sigma g_j(\boldsymbol{x}), 0\} + p \cdot H \cdot R_{i,t,j}$$

- $w_{i,t,j}$: displacement cost for placing sub-cell $t$ ($t \in \mathcal{T}_i$) at site $j$ ($j \in S$).
- $g_j(x)$: Overflow at site $j$ ($j \in S$).
- $R_{i,t,j}$ : total number of DRVs for placing sub-cell $t$ ($t \in \mathcal{T}_i$) at site $j$ ($j \in S$).

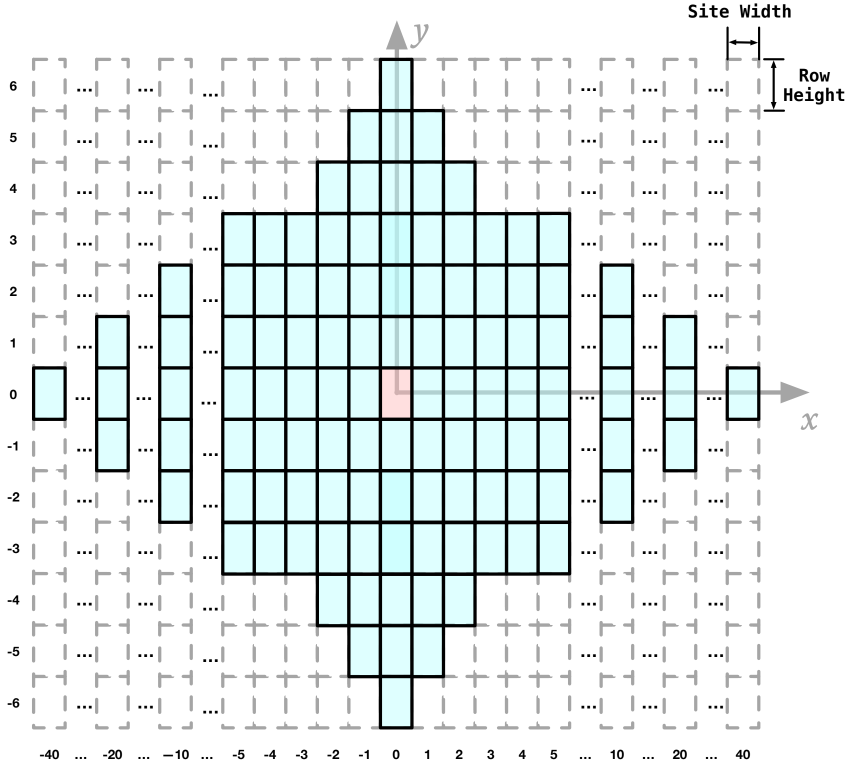



Figure: Diamond search range for the cell located at the red site.

# Linearized Proximal Gradient Method (cont'd)

$$\boldsymbol{x}^{(k+1)} = \underset{\boldsymbol{x}}{\operatorname{argmin}}\, \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}^{(k)}) = \underset{\boldsymbol{x}}{\operatorname{argmin}}\, \psi(\boldsymbol{x}, \boldsymbol{\lambda}^{(k)}) + I_{\mathcal{X}}(\boldsymbol{x}).$$

## Final Cost Function:

$$cost_{i,t,j} = w_{i,t,j} + \lambda_j \max\{1 + \sigma g_j(\boldsymbol{x}), 0\} + p \cdot H \cdot R_{i,t,j}$$

- $w_{i,t,j}$: displacement cost for placing sub-cell $t$ ($t \in \mathcal{T}_i$) at site $j$ ($j \in S$).
- $g_j(x)$: Overflow at site $j$ ($j \in S$).
- $R_{i,t,j}$ : total number of DRVs for placing sub-cell $t$ ($t \in \mathcal{T}_i$) at site $j$ ($j \in S$).

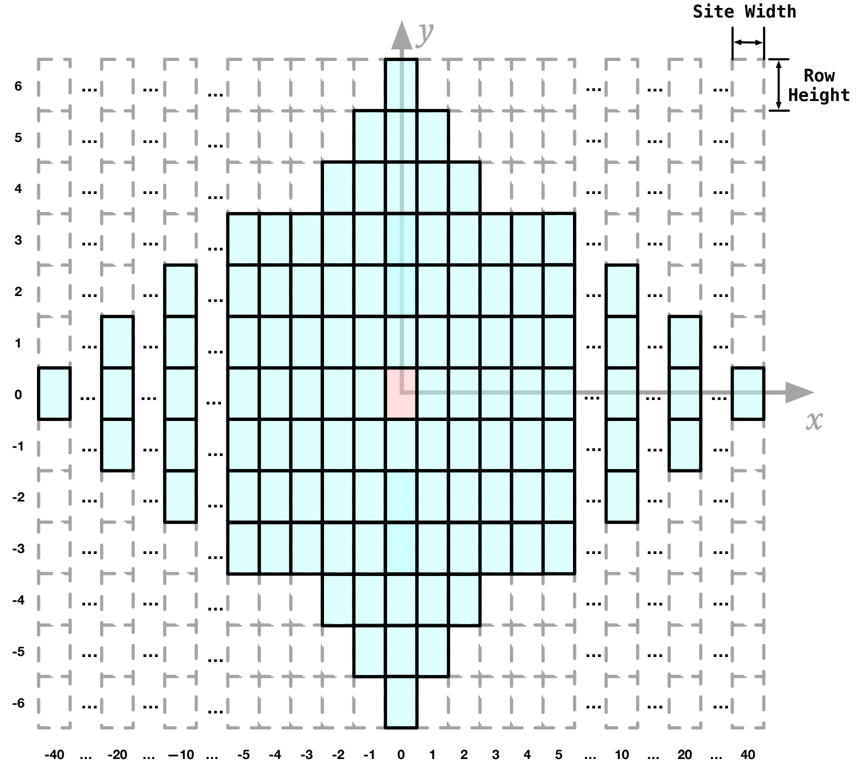



Figure: Diamond search range for the cell located at the red site.

## Steps:

- Enumerate candidate positions (diamond search).
- Compute costs for displacement, overflow and technology.
- Select minimum-cost positions.

## Advantages:

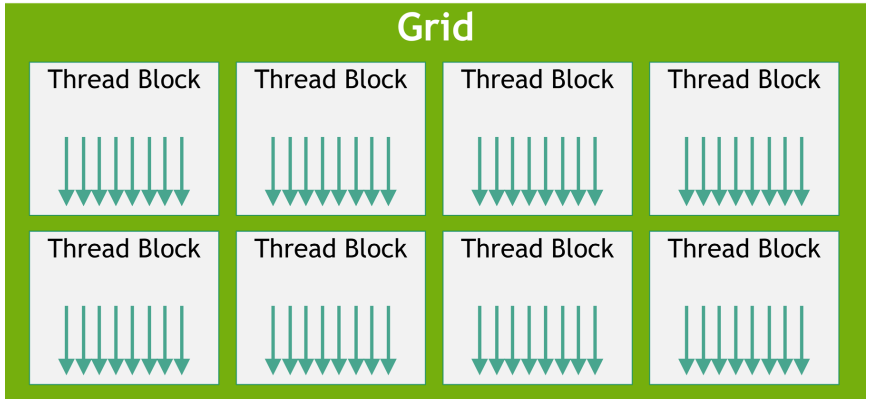
- Balance displacement and technology constraints.

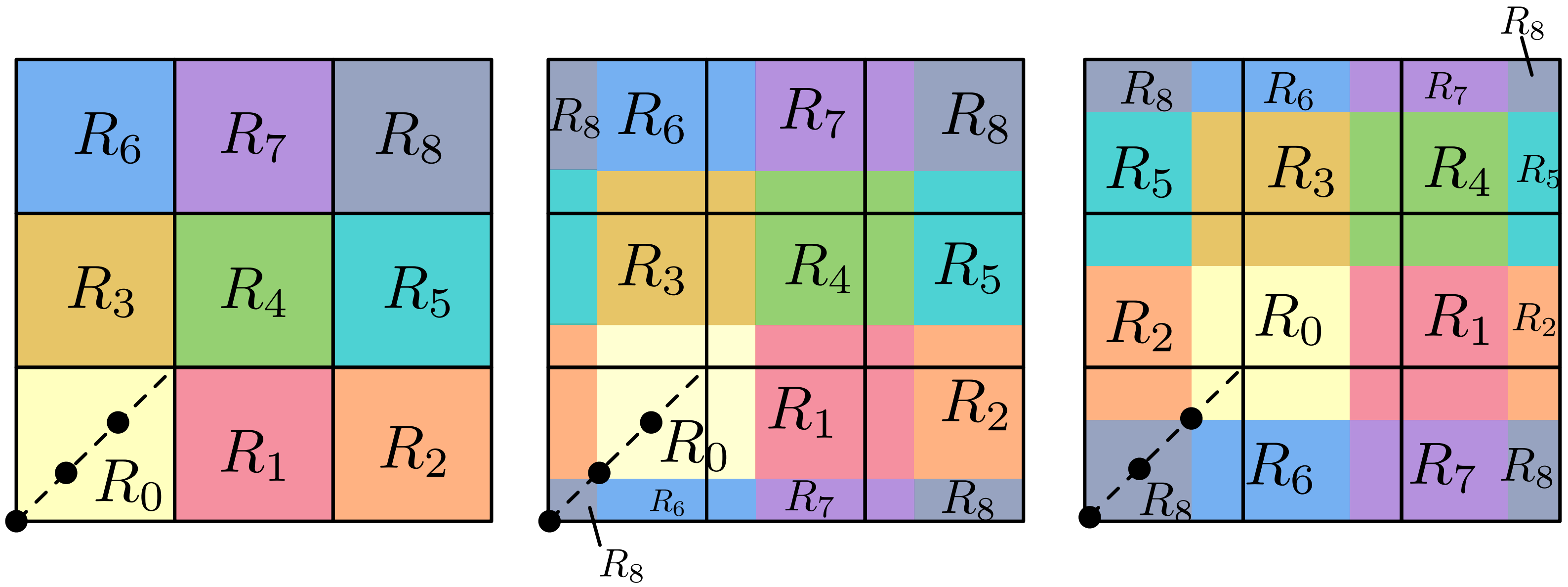
# Block Gradient Descent

- **Goal:** Parallelize cell updates without convergence loss.

- **Triple-fold Partitioning:**
  - Divide layout into non-overlapping sub-regions (totally 3 partition schemes).
  - Rotate partitions schemes to avoid update conflicts.

- **GPU Acceleration:**
  - Kernel design: 1 grid per sub region, threads for candidate selections.
  - Shared memory for cost reduction.



# Experimental Results

# Experiment Setup

## Benchmarks:

- ICCAD-2017 (routability / fence regions). [Darav+, ICCAD'17]
- Modified ISPD-2015 (million-cell designs). [Chow+, DAC'16]

## Metrics:

- Quality score $S$: Combine HPWL variation $S_{hpwl}$, weighted averaged displacement $S_{am}$, maximum displacement $M_{max}$, #pin short/access DRVs $N_p$, and #edge spacing DRVs $N_e$.
- Runtime and speedup.

## Platform:

- One NVIDIA A800 GPU.
- Two Intel Xeon Platinum 8358 CPUs (2.60GHz, 32 cores) with 1024GB RAM.

Table: Statistics of ICCAD-2017 Benchmarks.

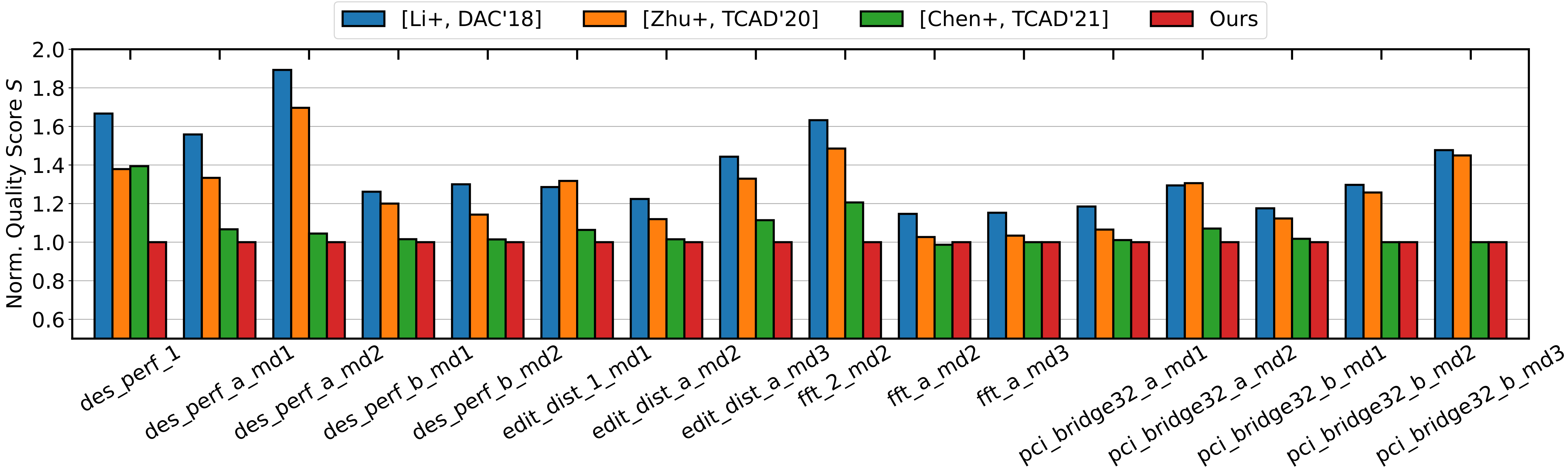
| Case | #Cells of Different Heights ($H$) | | | | Den. (%) | #Regions |
|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | | |
| des_perf_1 | 112644 | 0 | 0 | 0 | 90.6 | 0 |
| des_perf_a_md1 | 103589 | 4699 | 0 | 0 | 55.1 | 4 |
| des_perf_a_md2 | 105030 | 1086 | 1086 | 1086 | 55.9 | 4 |
| des_perf_b_md1 | 106782 | 5862 | 0 | 0 | 55.0 | 12 |
| des_perf_b_md2 | 101908 | 6781 | 2260 | 1695 | 64.7 | 12 |
| edit_dist_1_md1 | 118005 | 7994 | 2664 | 1998 | 67.4 | 0 |
| edit_dist_a_md2 | 115066 | 7799 | 2599 | 1949 | 59.4 | 1 |
| edit_dist_a_md3 | 119616 | 2599 | 2599 | 2599 | 57.2 | 1 |
| fft_2_md2 | 28930 | 2117 | 705 | 529 | 82.7 | 0 |
| fft_a_md2 | 27431 | 2018 | 672 | 504 | 32.3 | 0 |
| fft_a_md3 | 28609 | 672 | 672 | 672 | 31.2 | 0 |
| pci_bridge32_a_md1 | 26680 | 1792 | 597 | 448 | 49.5 | 4 |
| pci_bridge32_a_md2 | 25239 | 2090 | 1194 | 994 | 57.7 | 4 |
| pci_bridge32_b_md1 | 26134 | 1756 | 585 | 439 | 26.6 | 3 |
| pci_bridge32_b_md2 | 28038 | 292 | 292 | 292 | 18.3 | 3 |
| pci_bridge32_b_md3 | 27452 | 292 | 585 | 585 | 22.2 | 3 |

Table: Statistics of ISPD-2015 Benchmarks.

| Case | #Cells of Different Heights ($H$) | | Den. (%) |
|---|---|---|---|
| | 1 | 2 | |
| mgc_superblue11_a | 861314 | 64302 | 43 |
| mgc_superblue12 | 1172586 | 114362 | 45 |
| mgc_superblue14 | 564769 | 47474 | 56 |
| mgc_superblue16_a | 625419 | 47474 | 48 |
| mgc_superblue19 | 478109 | 27988 | 52 |

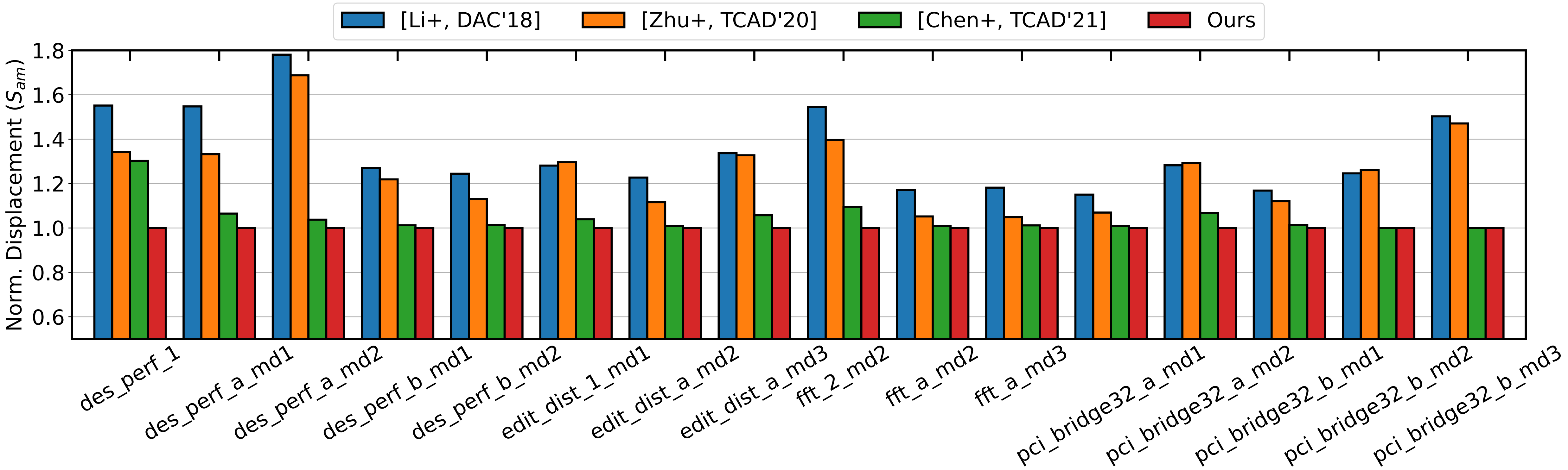
# Result on ICCAD-2017

- **Quality:** LEGALM improve overall score $S$ by 6-36% vs. SOTA with 1.03-3.83× speedup.



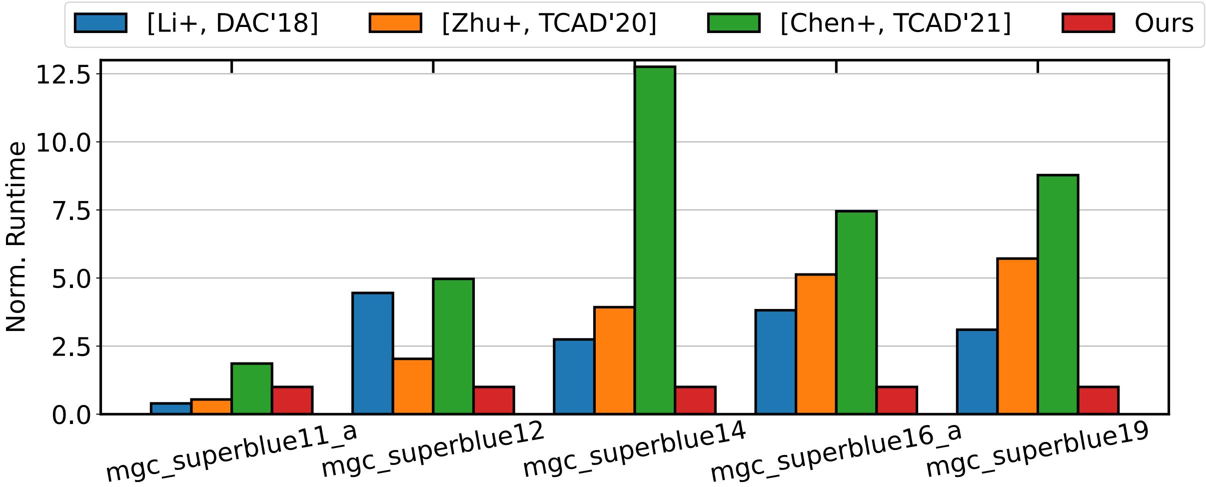
# Result on ICCAD-2017 (cont'd)

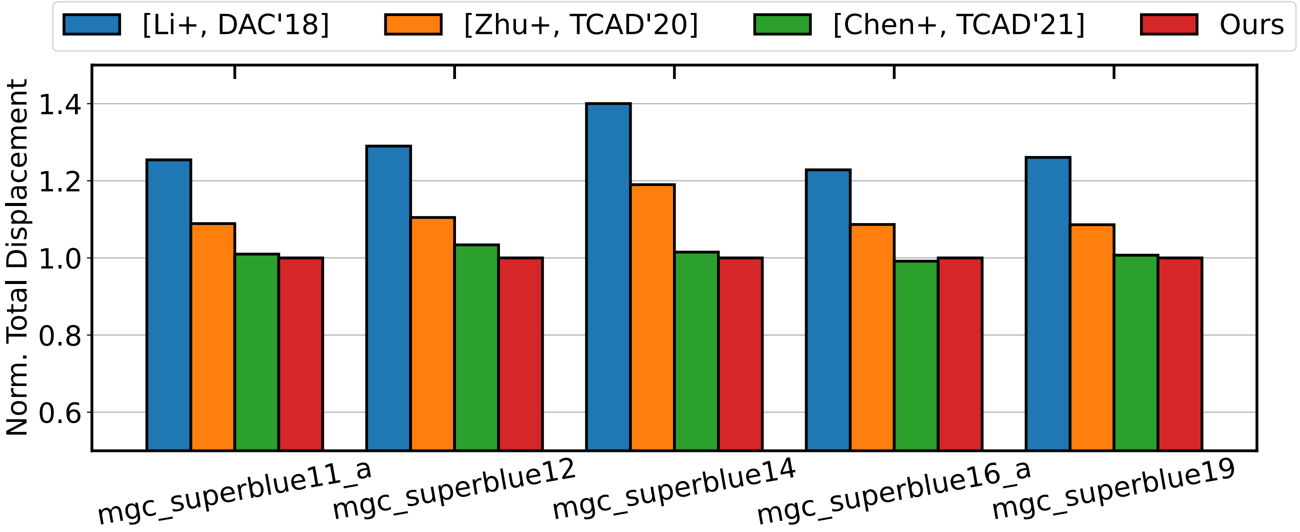
- **Quality:** LEGALM improve overall score $S$ by 6-36% vs. SOTA with 1.03-3.83× speedup.
- **Breakdown**:
  - 13-34% better HPWL variation $S_{hpwl}$.
  - 4.4-33.2% better weighted average displacement $S_{am}$.
  - 10% better maximum displacement $M_{max}$.



# Scalability on Large Designs

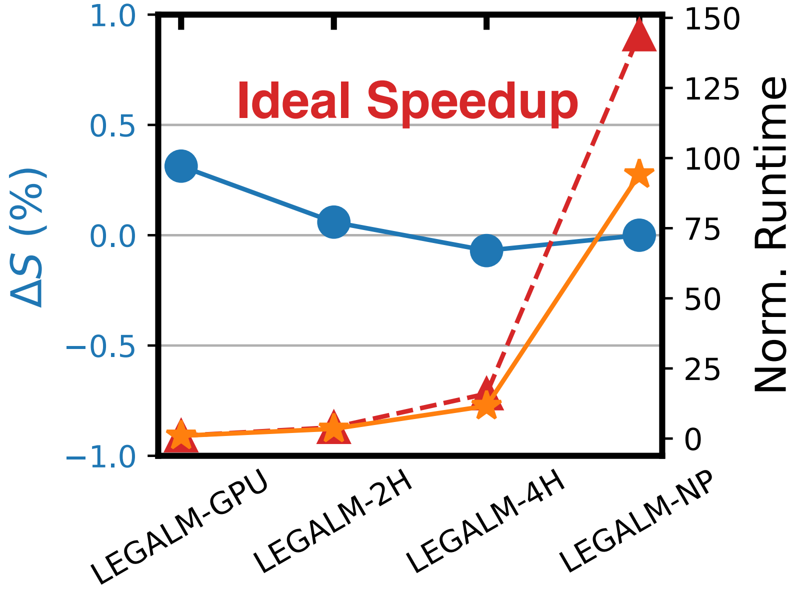
- **Cases:** `mgc_superblue` (1M+ cells)
- **Results:**
  - 2.25–5.99× faster than SOTA.
  - 1.1-28.5% better displacement than SOTA.
  - Legalization in < 10 seconds for 3/5 cases.
- **Why?:** GPU parallelism + efficient partitioning.



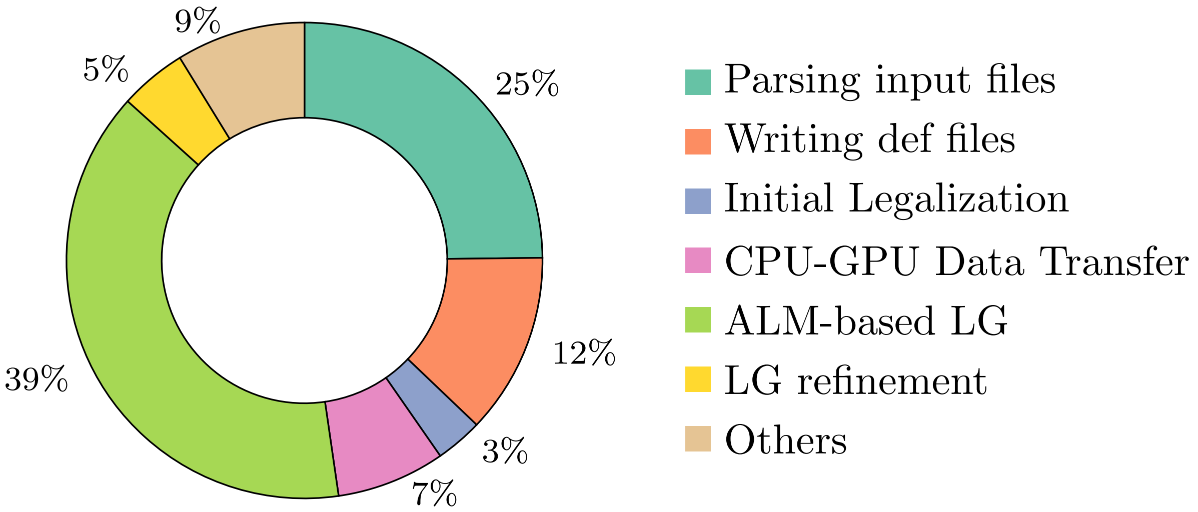
# Ablation Study & Runtime Breakdown

- **Triple-fold Partitioning (TP) Impact:**
  - 94.2× speedup vs. no partitioning (LEGALM-NP).
  - <0.5% quality loss.
- **Grid Size:** Larger sub-regions reduce parallelism but improve movement.




LEGALM-2H: double sub-region size
LEGALM-4H: quadruple sub-region size

- **Runtime Breakdown:**
  - ALM-based Legalization: 39%.
  - Initial Legalization: 3%
  - Legalization Refinement: 5%.



# Conclusion & Future Work

# Conclusion & Future Work

**LEGALM**, an efficient legalization method for mixed-cell-height circuits using the linearized augmented Lagrangian method.

## ❑ Conclusion:

- Linearized ALM formulation for mixed-cell-height legalization.
- Block Gradient Descent (BGD) + Triple-fold partitioning for GPU acceleration.
- 6–36% better quality, 94.2$\times$ speedup.

## ❑ Future Work:

- Advanced parallelization strategies.

# THANK YOU!

jingmai@pku.edu.cn

Personal Website

magic3007.github.io