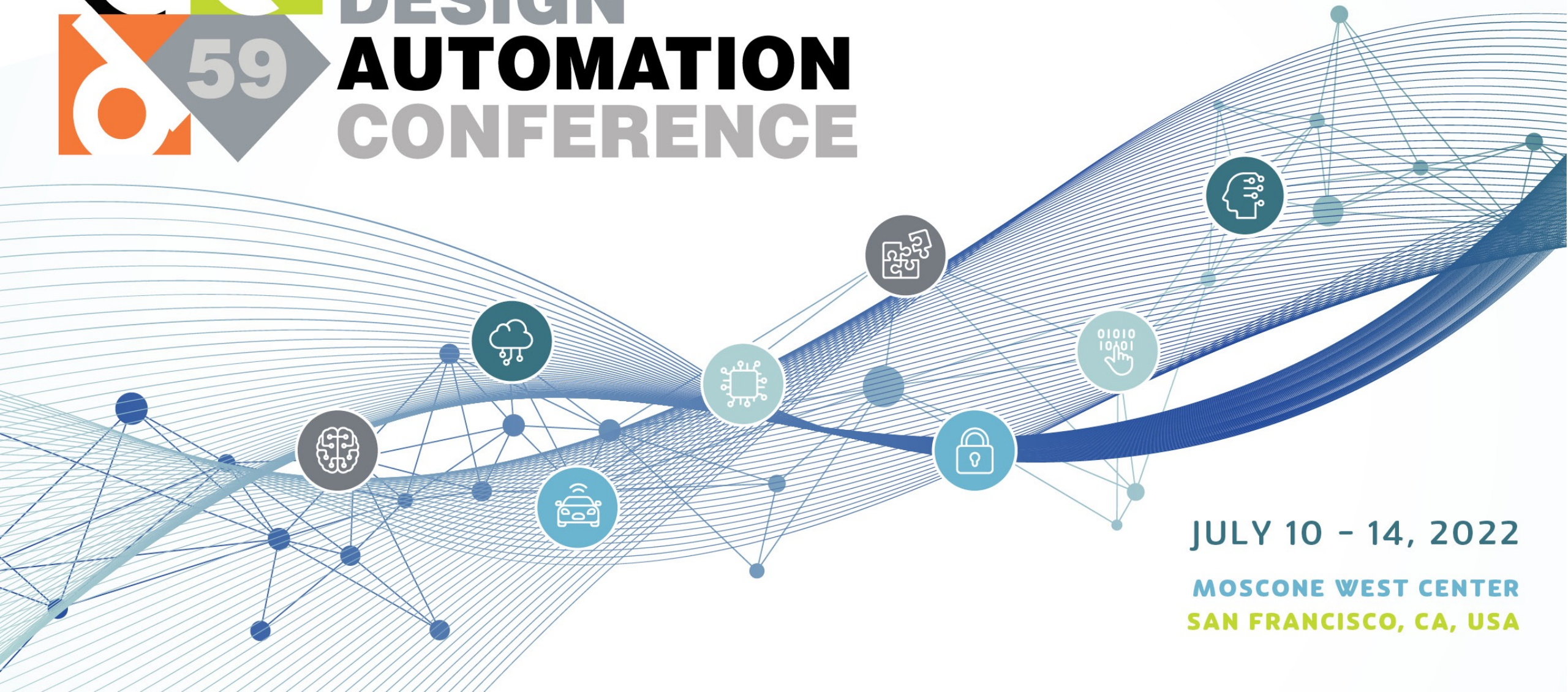




DESIGN **AUTOMATION** CONFERENCE



JULY 10 - 14, 2022

MOSCONE WEST CENTER
SAN FRANCISCO, CA, USA

Multi-Electrostatic FPGA Placement Considering SLICEL-SLICEM Heterogeneity and Clock Feasibility

Jing Mai¹, Yibai Meng¹, Zhixiong Di², and Yibo Lin¹

¹Peking University

²Southwest Jiaotong University

jingmai@pku.edu.cn



北京大學
PEKING UNIVERSITY



北京大学高能效计算与应用中心
Center for Energy-efficient Computing and Applications



西南交通大學
Southwest Jiaotong University



Outline

Motivation & Challenges

Proposed Algorithm

Experimental Results

Conclusion & Future work

Modern FPGA Placement Challenges

The modern FPGAs come with advanced technologies along with more challenges.

Massive Design

- ▶ Millions of cells in a single design

Highly Heterogeneity

- ▶ SLICEL-SLICEM Heterogeneity
- ▶ CARRY Chain

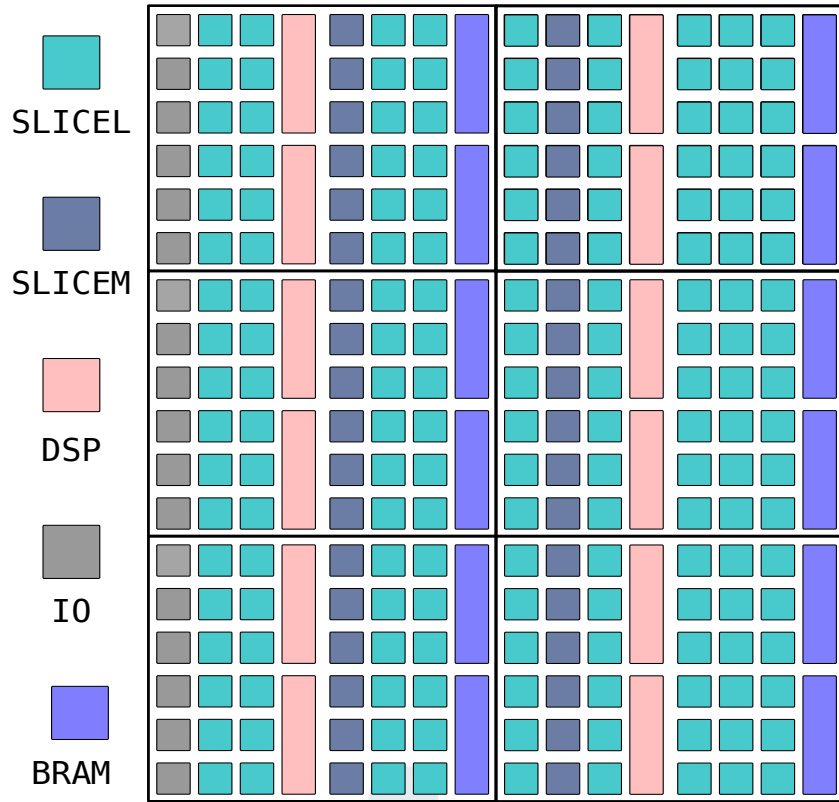
Highly Discrete

- ▶ Clock Region Constraints and Half-Column Constraints

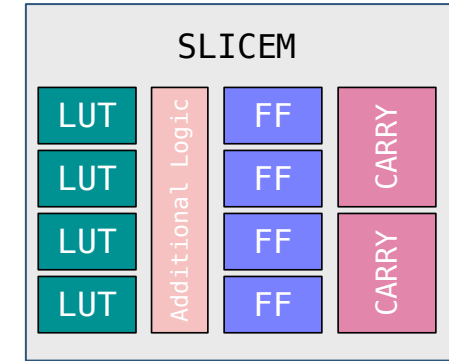
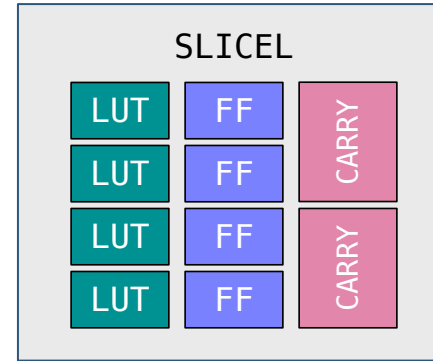
Two Categories of Slices: SLICEL and SLICEM

Modern FPGAs contain heterogeneous resources, including **CLB**, **DSP**, **BRAM**, **IO**, etc.

A Configurable Logic Block (CLB) is further categorized to **SLICEL** and **SLICEM** with **asymmetric** compatibility.



FPGA Layout



CLB Slice	LUTs	FFs	CARRY	Distributed RAM	SHIFT
SLICEL	8	16	1	NA	NA
SLICEM	8	16	1	512 bits	512 bits

Logic Resource on One CLB Slice

Distributed RAM or SHIFT as SLICEM

Three LUT-like instances: regular LUT, distributed RAM and shift registers (SHIFT).



SLICEL

- ▶ LUT can only be used for logic (not memory)
- ▶ Has Carry chains

SLICEM

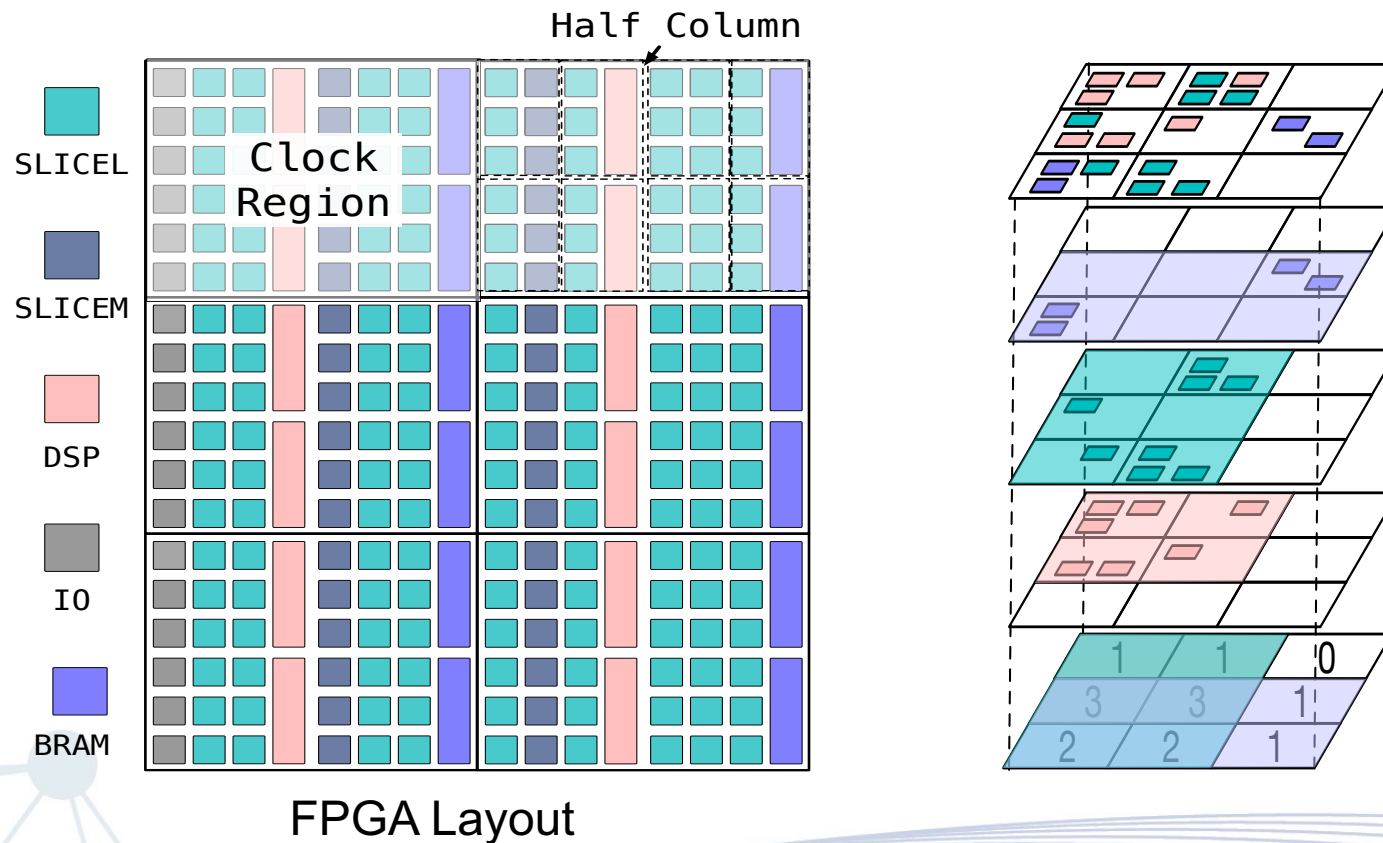
- ▶ LUT can only be used for logic and **memory/SRL**¹
- ▶ Has carry chains
- ▶ **No mixing** between LUTs, distributed RAMs, and SHIFTs in a SLICEM is allowed.

¹SRL (Shift Register Lookup table).

Clock Constraints

Clock Region Constraint¹: The clock demand of each clock region is at most 24.

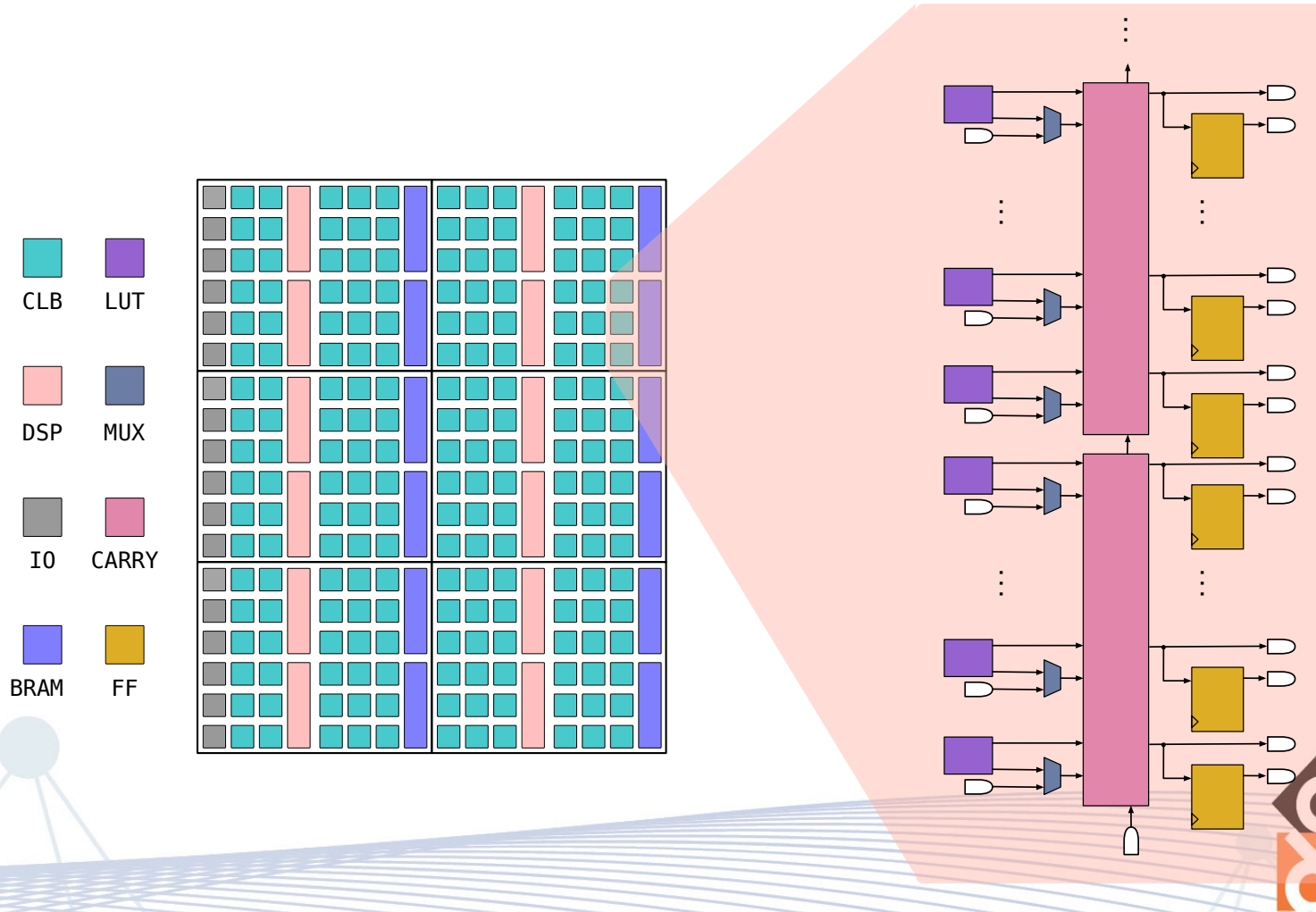
Half Column Constraint¹: The number of clock nets within the half column to be at most 12.



Illustrations of the clock demand of a clock region.

Cascaded CARRY Chain

Cascaded CARRYs and the immediate LUTs and FFs should be placed in **consecutive** CLB in the same column.



Prior Work Taxonomy

Placer	Clock Constraints	LUT, FF, BRAM, DSP Supported	CARRY, SHIFT, distributed RAM Supported	GPU-Acceleration	Algorithm Category
RippleFPGA	✗	✓	✗	✗	Quadratic
GPlace	✗	✓	✗	✗	Quadratic
UTPlaceF	✗	✓	✗	✗	Quadratic
elfPlace	✗	✓	✗	✓	Nonlinear
GPlace 3.0	✗	✓	✗	✗	Quadratic
RippleFPGA Clock-Aware	✓	✓	✗	✗	Quadratic
UTPlaceF 2.0 & 2.X	✓	✓	✗	✗	Quadratic
NTUfPlace	✓	✓	✗	✗	Nonlinear
Ours	✓	✓	✓	✓	Nonlinear

Our Contributions

A heterogeneous FPGA placement algorithm considering SLICEL-SLICEM heterogeneity and clock feasibility

- ▶ A new multi-electrostatic formulation for asymmetric slice compatibility from **SLICEL-SLICEM** heterogeneity
- ▶ A **CARRY chain** alignment technique to achieve the hard constraint in an iterative manner
- ▶ A **quadratic penalization** technique to eliminate violations of the discrete clock constraints.
- ▶ A **nested** Lagrangian relaxation-based technique for wirelength, routability, and clock optimization with a **dynamically-adjusted** preconditioning technique to stabilize the convergence

Outline

Motivation & Challenges

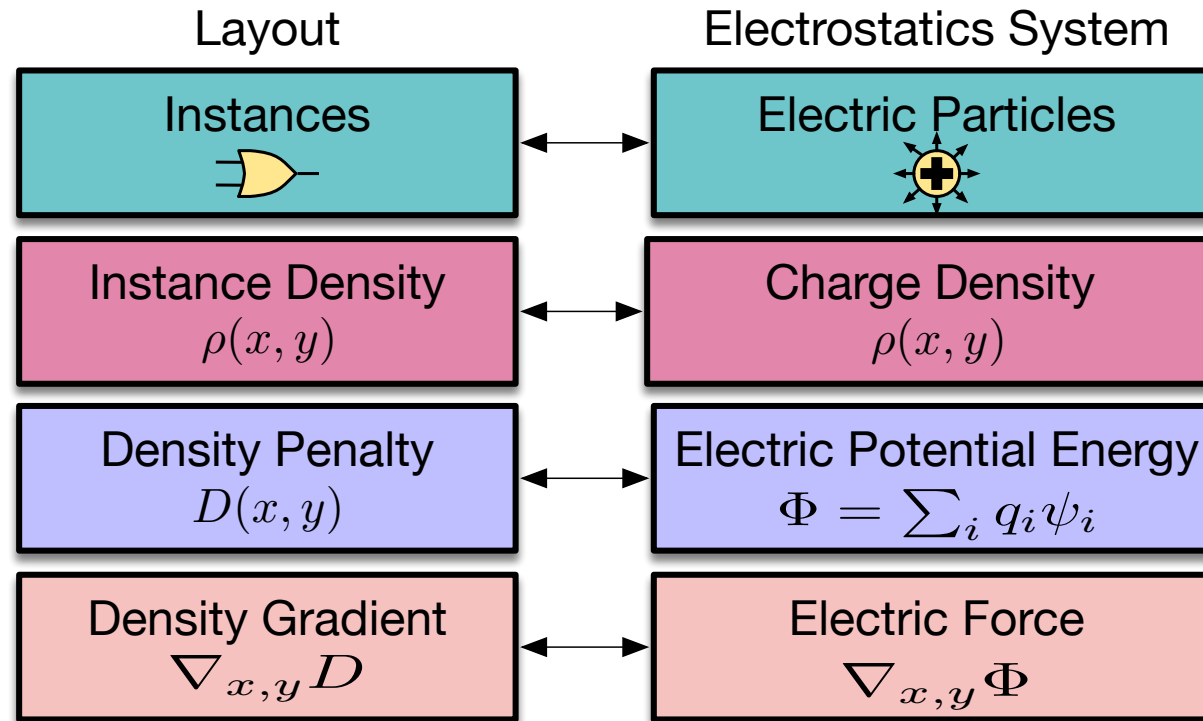
Proposed Algorithm

Experimental Results

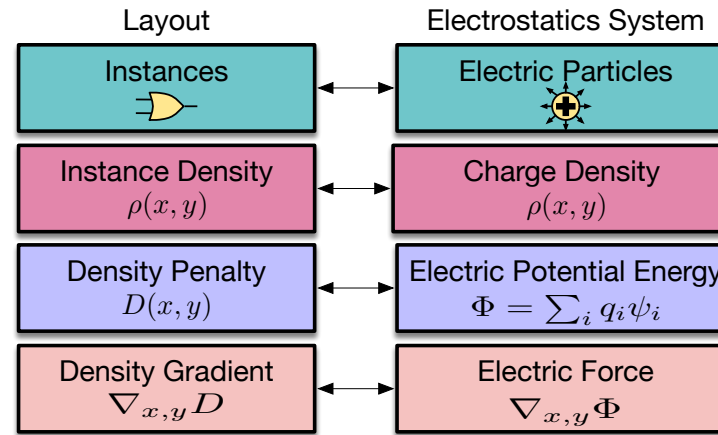
Conclusion & Future work

Multi-Electrostatics based FPGA Placement

Analogy between placement of a single resource type and an **electrostatic** system, ePlace [Lu+, TCAD'15]



Multi-Electrostatics based FPGA Placement



Each resource type has a **separate** electrostatic system, elfPlace [Meng+, TCAD'2021]

$$\begin{aligned} \min_{\mathbf{x}, \mathbf{y}} \tilde{W}(\mathbf{x}, \mathbf{y}), \\ \text{s. t. } \Phi_s(\mathbf{x}, \mathbf{y}) = 0, \forall s \in S = \{LUT, FF, DSP, BRAM\}. \end{aligned}$$

$\tilde{W}(\cdot)$: Weighted-Average (WA) wirelength model [Ray+, DATE'2013].

Unconstrained Problem Formulation

Formulate with **clock** constraints and **carry chain** alignment constraints,

$$\begin{aligned} & \min_{\mathbf{x}, \mathbf{y}} \widetilde{W}(\mathbf{x}, \mathbf{y}), \\ & \text{s. t. } \Phi_s(\mathbf{x}, \mathbf{y}) = 0, \forall s \in S, \\ & \text{clock penalization term: } \Gamma(\mathbf{x}, \mathbf{y}) = 0 \\ & \text{Carry chain alignment constraint.} \end{aligned}$$

Transfer the constrained problem into an **unconstrained** one,

$$\begin{aligned} \min_{\mathbf{x}, \mathbf{y}} \mathcal{L}(\mathbf{x}, \mathbf{y}; \boldsymbol{\lambda}, \mathcal{A}, \eta) &= \widetilde{W}(\mathbf{x}, \mathbf{y}) + \sum_{s \in S} \lambda_s \mathcal{D}_s + \eta \Gamma(\mathbf{x}, \mathbf{y}) \\ \mathcal{D}_s &= \Phi_s + \frac{1}{2} \mathcal{C}_s \Phi_s^2, \forall s \in S. \end{aligned}$$

Electric Field Setup

Define the field type set S with special field setups to handle **SLICEL-SLICEM**

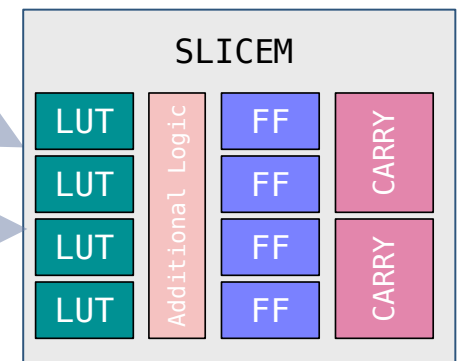
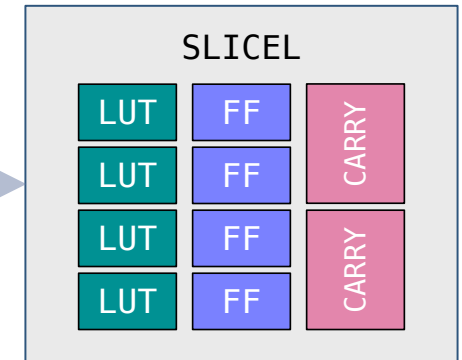
heterogeneity, $S = \{\text{LUTL}, \text{LUTM-AL}, \text{FF}, \text{CARRY}, \text{DSP}, \text{BRAM}\}$

LUTL

Electric field for the LUT resource (both SLICEL and SLICEM)

LUTM-AL

Electric field for the Additional Logic resource (**SLICEM only**)



Special Field Mechanism

Fields **LUT** and **LUTM-AL** form an **asymmetric** combination mechanism together.



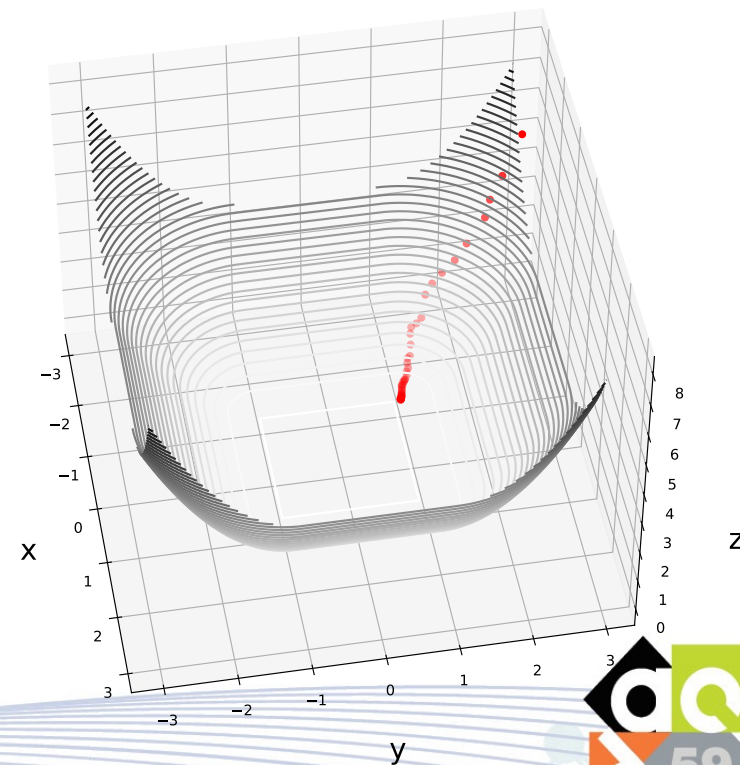
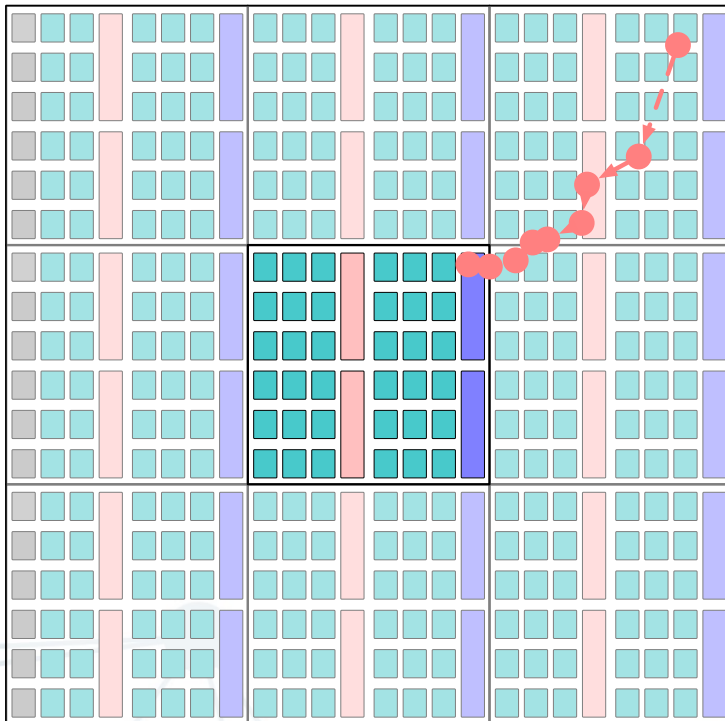
		LUTL Field		LUTM-AL Field		Two Fields
		ρ_{LUT}	Φ_{LUT}	$\rho_{LUTM-AL}$	$\Phi_{LUTM-AL}$	$\Phi_{LUT} + \Phi_{LUTM-AL}$
Initial			-		-	-
Solution I			Low		Low	Low
Solution II			Low		High	High



Two-Stage Clock Network Planning

1. Generate instance-to-clock region mapping using the *branch-and-bound method*
2. Introduce bowl-like and **differentiable** “gravitational” attraction terms for optimization

$$\min_{x,y} \mathcal{L}(x, y; \lambda, \mathcal{A}, \eta) = \widetilde{W}(x, y) + \sum_{s \in S} \lambda_s \mathcal{D}_s + \eta \Gamma(x, y)$$

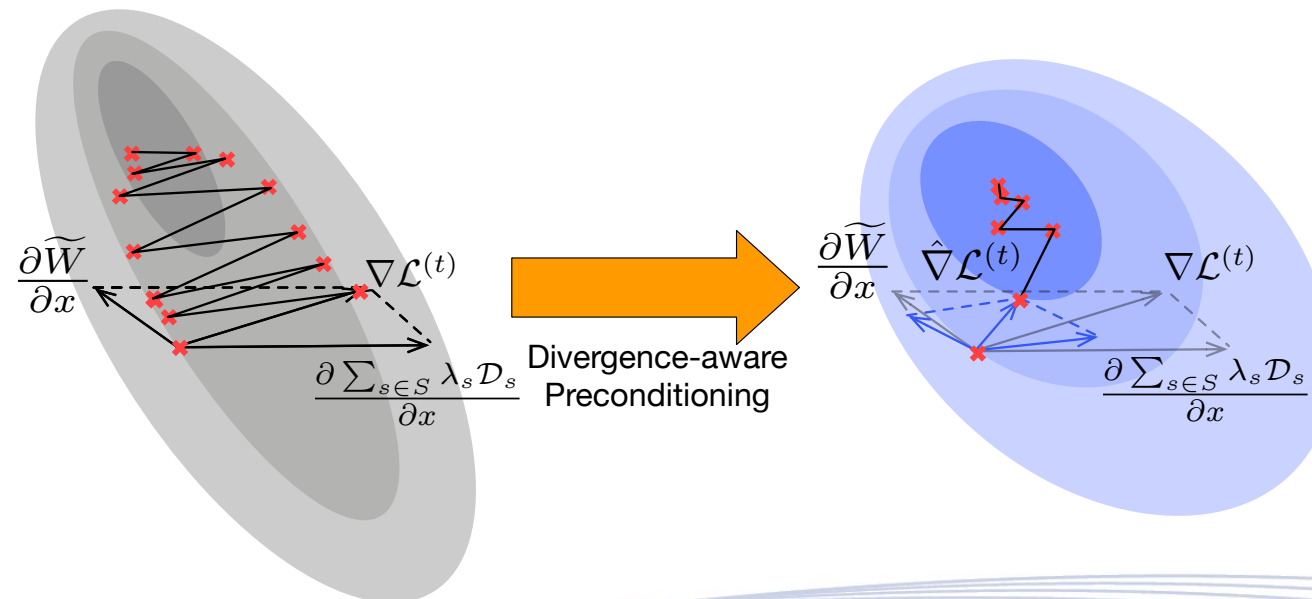


Divergence-aware Preconditioning

Stabilizes convergence by preconditioning the gradient $\nabla\mathcal{L}^{(t)}$ by $\nabla\mathcal{L}^{(t)} \odot \mathcal{P}^{(t)}$

Improve elfPlace's approaches using the divergence-aware weighting method

- ▶ elfPlace: $\mathcal{P}_i^{(t)} \sim \max\left(1, [\mathcal{P}_i^W + \lambda_s^{(t)} \mathcal{A}_i^s]^{-1}\right), \forall i \in \mathcal{V}_s, \mathcal{P}_i^W = \frac{\partial^2 \tilde{W}(x, y)}{\partial x_i^2} \sim \sum_{e \in E_i} \frac{w_e}{|e| - 1}, \forall i \in \mathcal{V}$
- ▶ Ours: $\mathcal{P}_i^{(t)} \sim \max\left(1, [\mathcal{P}_i^W + \sum_{s \in S} \alpha_s^{(t)} \lambda_s^{(t)} \mathcal{A}_i^s]^{-1}\right)$

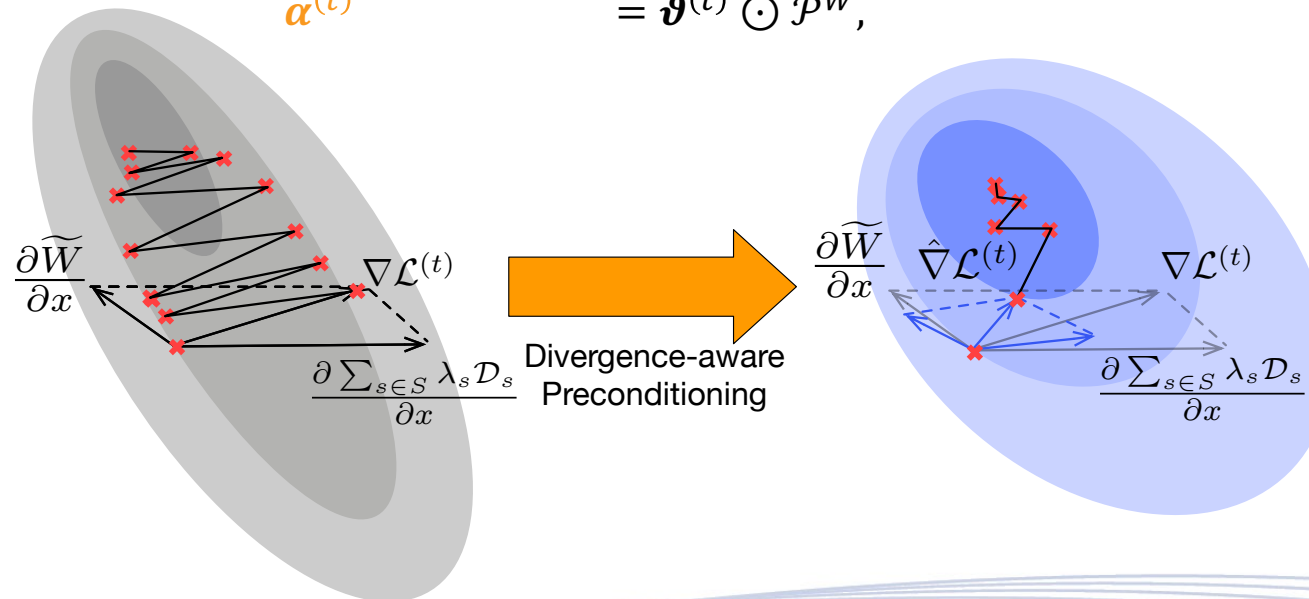


Divergence-aware Preconditioning

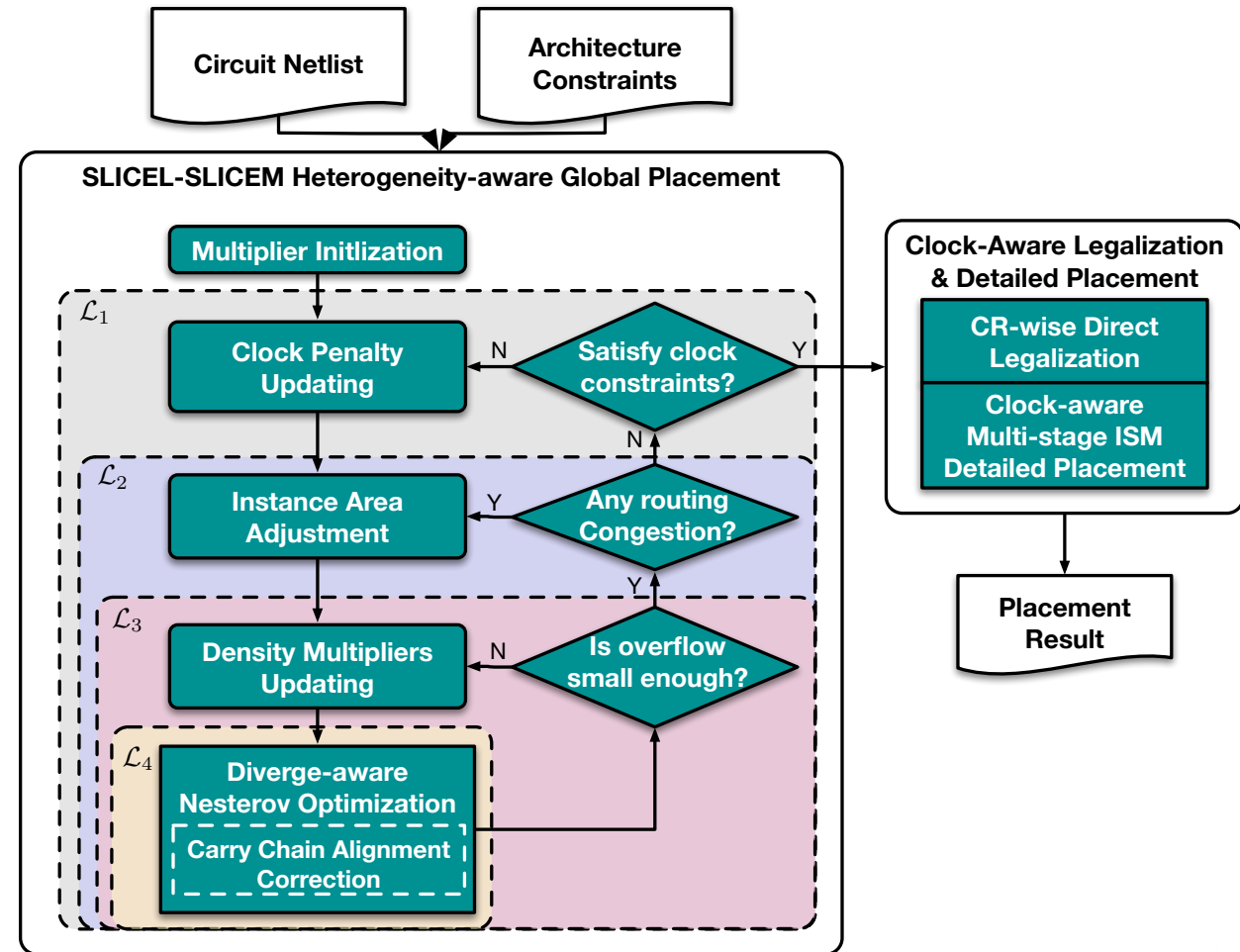
$\alpha_s^{(t)}$ balances the ratio of the gradient norms from density and wirelength

The fast increasing of this ratio indicates the divergence

$$\vartheta_s^{(t)} = \max\left(1, \frac{\nabla \mathcal{D}_s}{\sum_{i \in \mathcal{V}_s^r} |\partial \tilde{W} / \partial x_i|}\right), \forall s \in S,$$
$$\bar{\mathcal{P}}_s^W = \frac{\sum_{i \in \mathcal{V}_s^r} \mathcal{P}_i^W}{|\mathcal{V}_s^r|}, \forall s \in S,$$
$$\alpha^{(t)} = \vartheta^{(t)} \odot \bar{\mathcal{P}}^W,$$



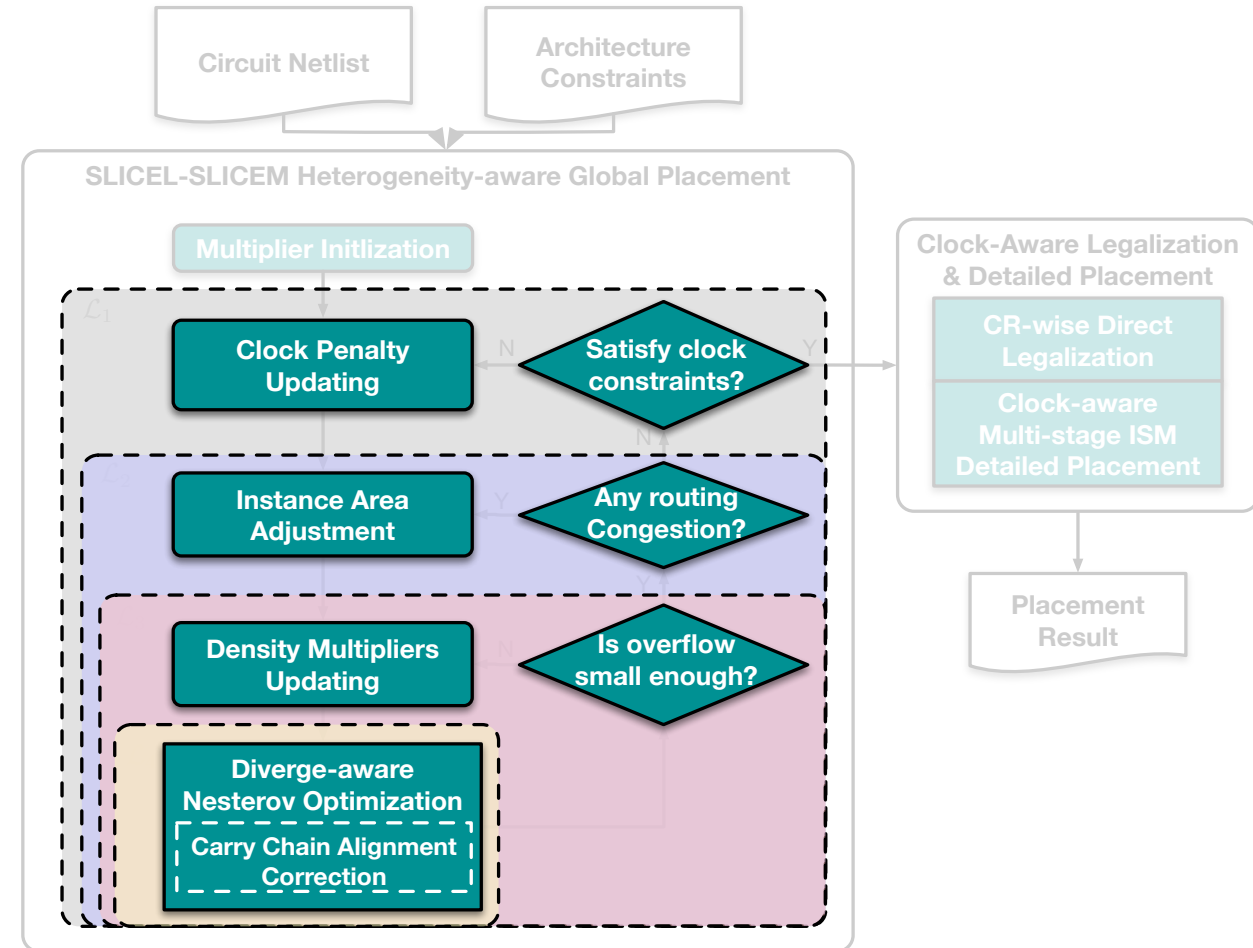
Overall Flow: Nested Optimization Framework



Overall Flow: Nested Optimization Framework

Clock Opt.

- ▶ $\mathcal{L}_1 = \max_{\eta} \mathcal{L}_2(\eta)$
- ▶ Two-stage clock network planning



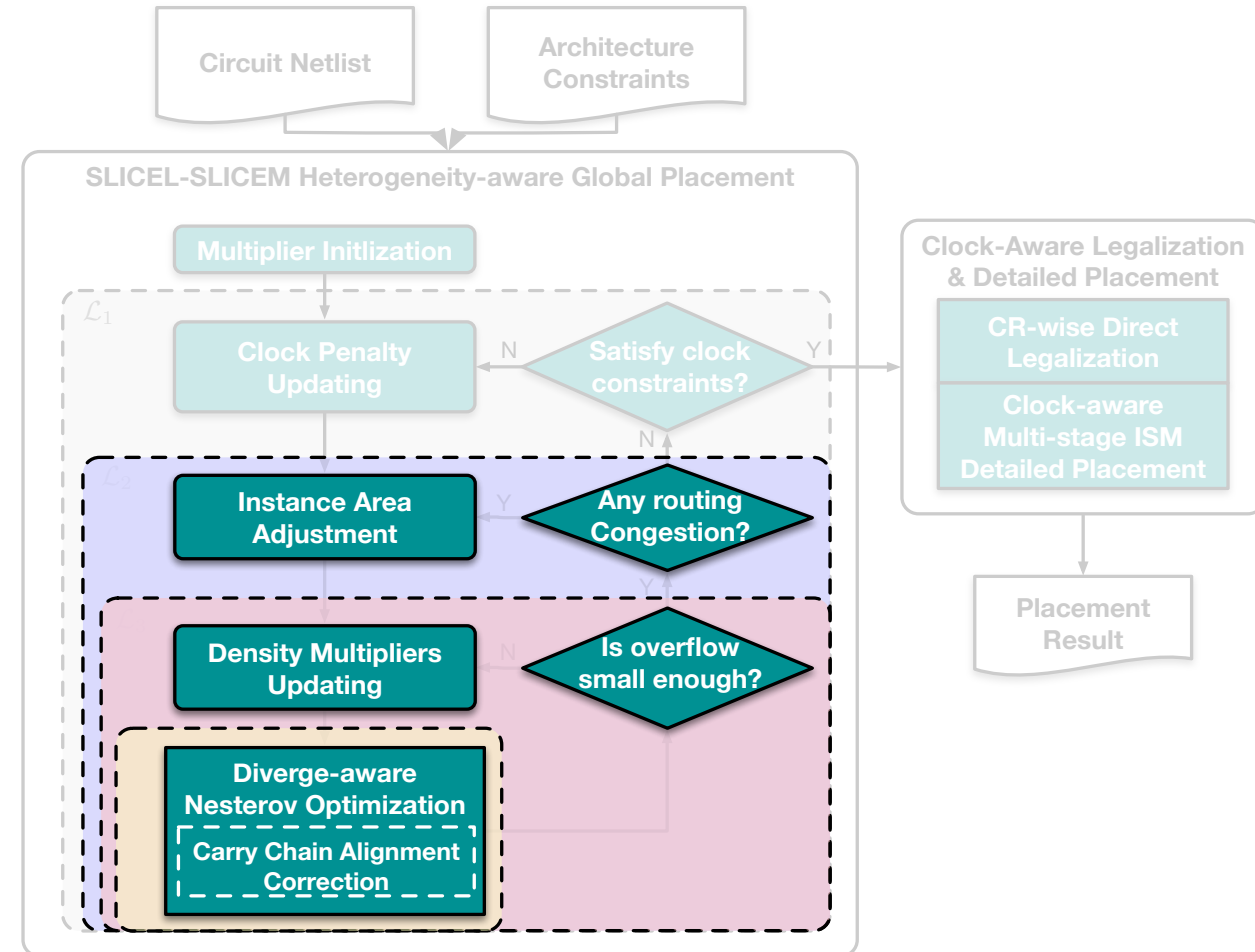
Overall Flow: Nested Optimization Framework

Clock Opt.

- ▶ $\mathcal{L}_1 = \max_{\eta} \mathcal{L}_2(\eta)$
- ▶ Two-stage clock network planning

Routability Opt.

- ▶ $\mathcal{L}_2(\eta) = \max_{\mathcal{A}} \mathcal{L}_3(\mathcal{A}, \eta)$
- ▶ Area adjustment scheme



Overall Flow: Nested Optimization Framework

Clock Opt.

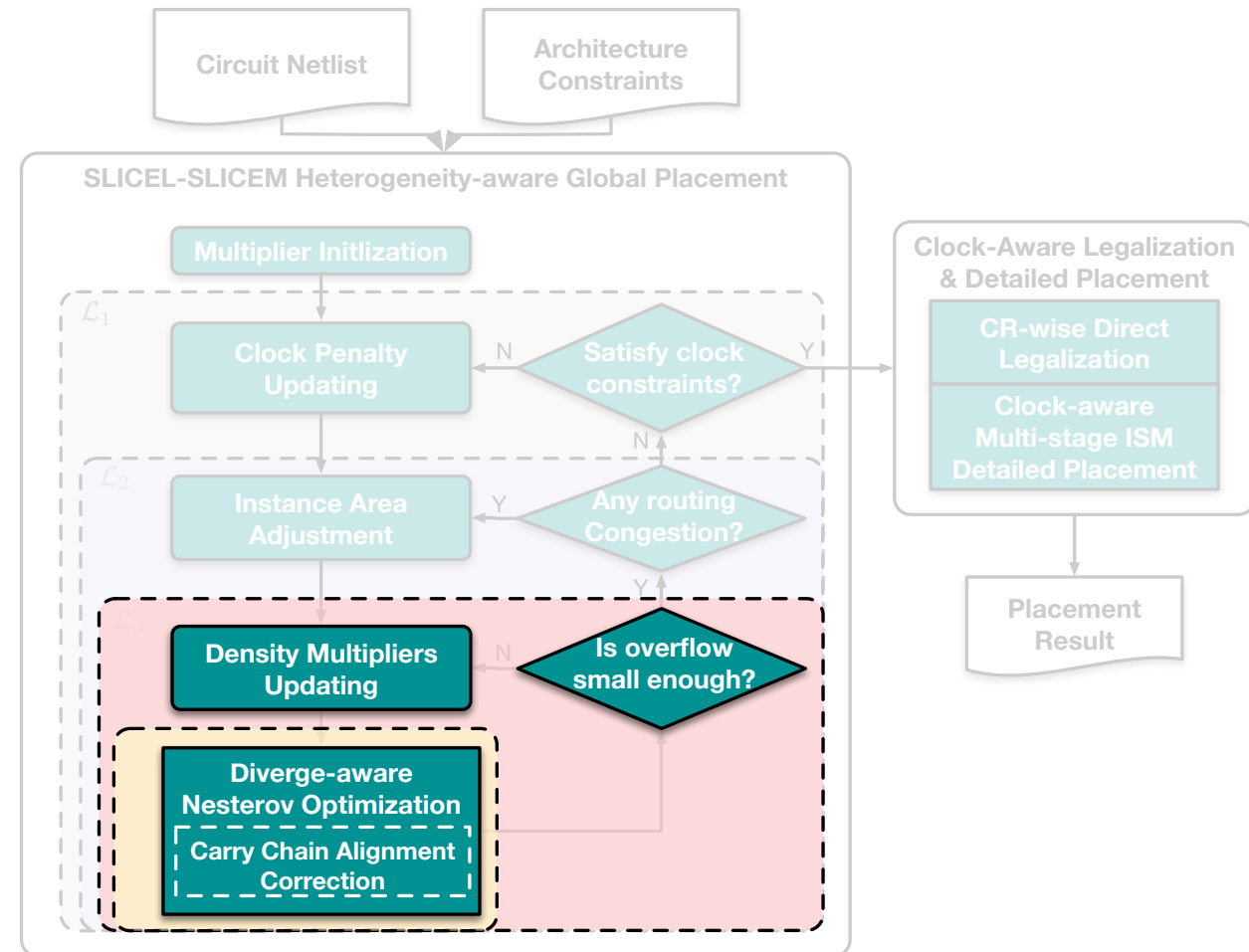
- ▶ $\mathcal{L}_1 = \max_{\eta} \mathcal{L}_2(\eta)$
- ▶ Two-stage clock network planning

Routability Opt.

- ▶ $\mathcal{L}_2(\eta) = \max_{\mathcal{A}} \mathcal{L}_3(\mathcal{A}, \eta)$
- ▶ Area adjustment scheme

Wirelength Opt.

- ▶ $\mathcal{L}_3(\mathcal{A}, \eta) = \max_{\lambda} \mathcal{L}_4(\lambda, \mathcal{A}, \eta)$
- ▶ Weighted Average (WA) wirelength model



Overall Flow: Nested Optimization Framework

Clock Opt.

- ▶ $\mathcal{L}_1 = \max_{\eta} \mathcal{L}_2(\eta)$
- ▶ Two-stage clock network planning

Routability Opt.

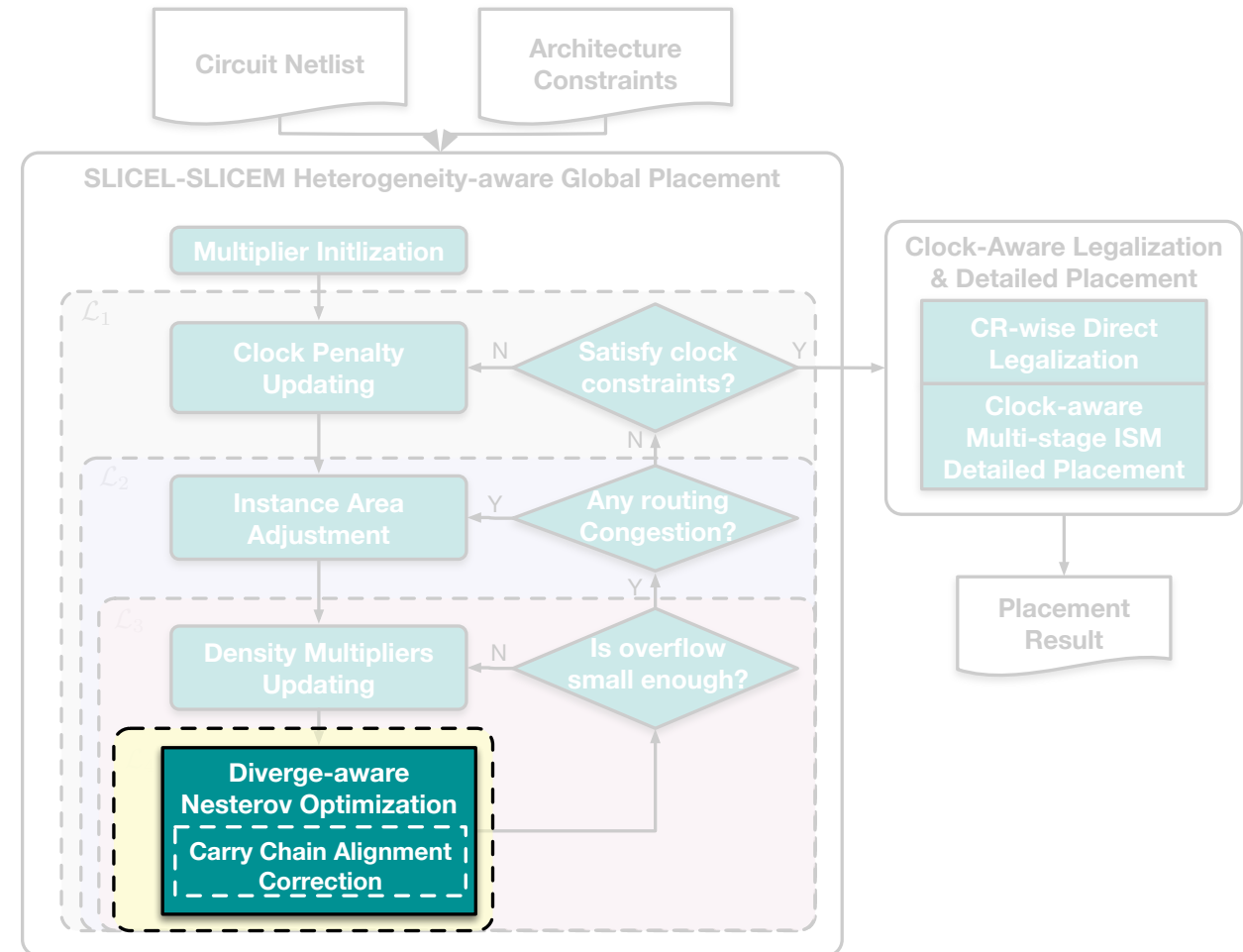
- ▶ $\mathcal{L}_2(\eta) = \max_{\mathcal{A}} \mathcal{L}_3(\mathcal{A}, \eta)$
- ▶ Area adjustment scheme

Wirelength Opt.

- ▶ $\mathcal{L}_3(\mathcal{A}, \eta) = \max_{\lambda} \mathcal{L}_4(\lambda, \mathcal{A}, \eta)$
- ▶ Weighted Average (WA) wirelength model

Subproblem Opt.

- ▶ $\mathcal{L}_4(\lambda, \mathcal{A}, \eta) = \min_{x,y} \mathcal{L}(x, y, \lambda, \mathcal{A}, \eta)$
- ▶ Divergence-aware Preconditioning
- ▶ CARRY chain alignment after each descending step



Outline

Motivation & Challenges

Proposed Algorithm

Experimental Results

Conclusion & Future work

Experimental Setup

Machine

- ▶ Intel Xeon Silver 4214 CPUs (2.20 GHz and 24 cores)
- ▶ One NVIDIA TITAN RTX GPU
- ▶ C++ and Python with Pytorch for GPU acceleration

Benchmark suites

- ▶ ISPD 2017 clock-aware FPGA placement benchmarks with 0.4M – 0.9M cells
- ▶ Industry benchmarks with heterogeneous cells, i.e., SHIFTs, distributed RAMs and CARRYs

Placers for comparison

- ▶ UTPlaceF 2.0 [Li+, TODAES'2018]
- ▶ RippleFPGA [Pui+, ICCAD'2017]
- ▶ UTPlaceF 2.X [Li+, FPGA'2019]
- ▶ NTUfplace [Chen+, TCAD'2020]

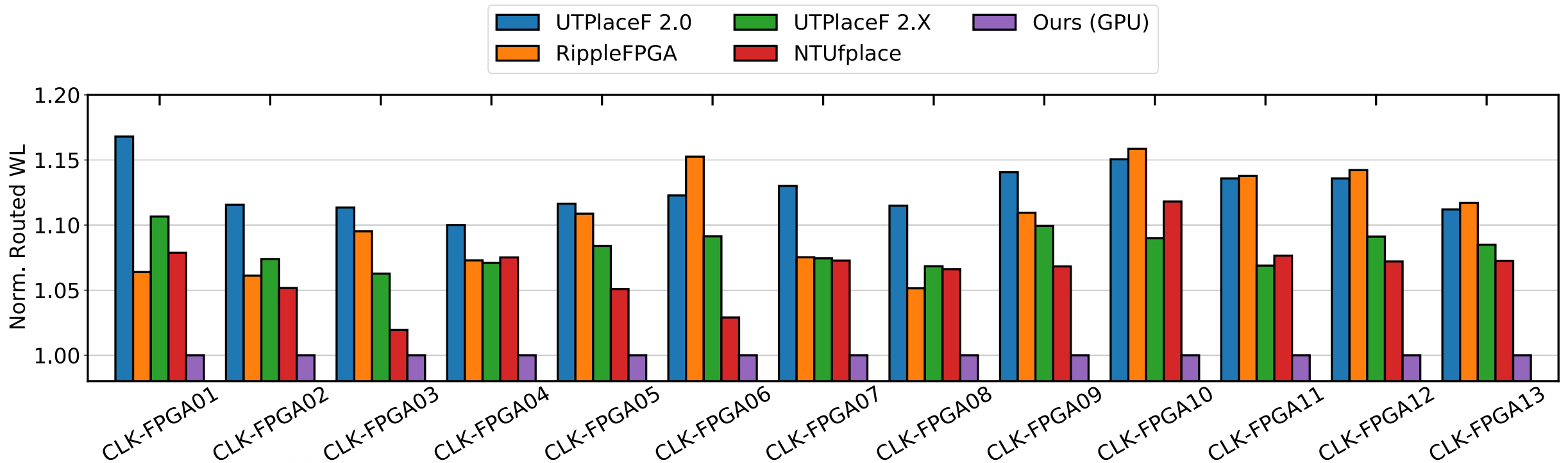
Routed Wirelength Comparison

▶ **14.2%** better than UTPlaceF 2.0

▶ **14.2%** better than UTPlaceF 2.X

▶ **11.7%** better than RippleFPGA

▶ **11.7%** better than NTUfplace



Designs with {LUT, FF, DSP, BRAM, IO}

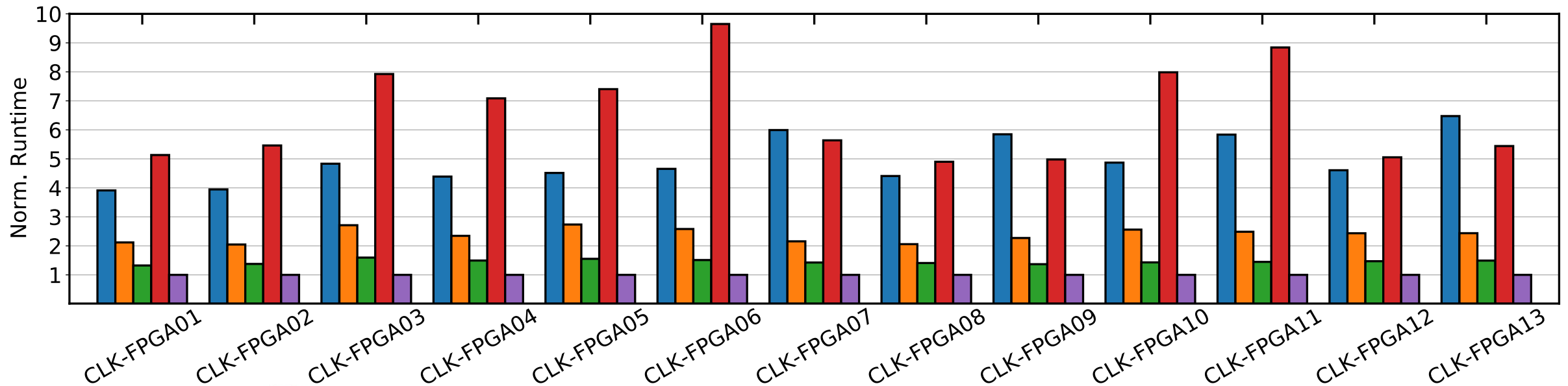
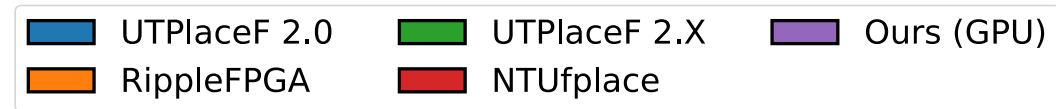
Runtime Comparison

▶ **4.94X** faster than UTPlaceF 2.0

▶ **1.45X** faster than UTPlaceF 2.X

▶ **2.38X** faster than RippleFPGA

▶ **6.58X** faster than NTUfplace



1.45-6.58X speedup with GPU acceleration !

Ablation Study on Heterogeneous Industrial Benchmarks

w/o precond or chain align

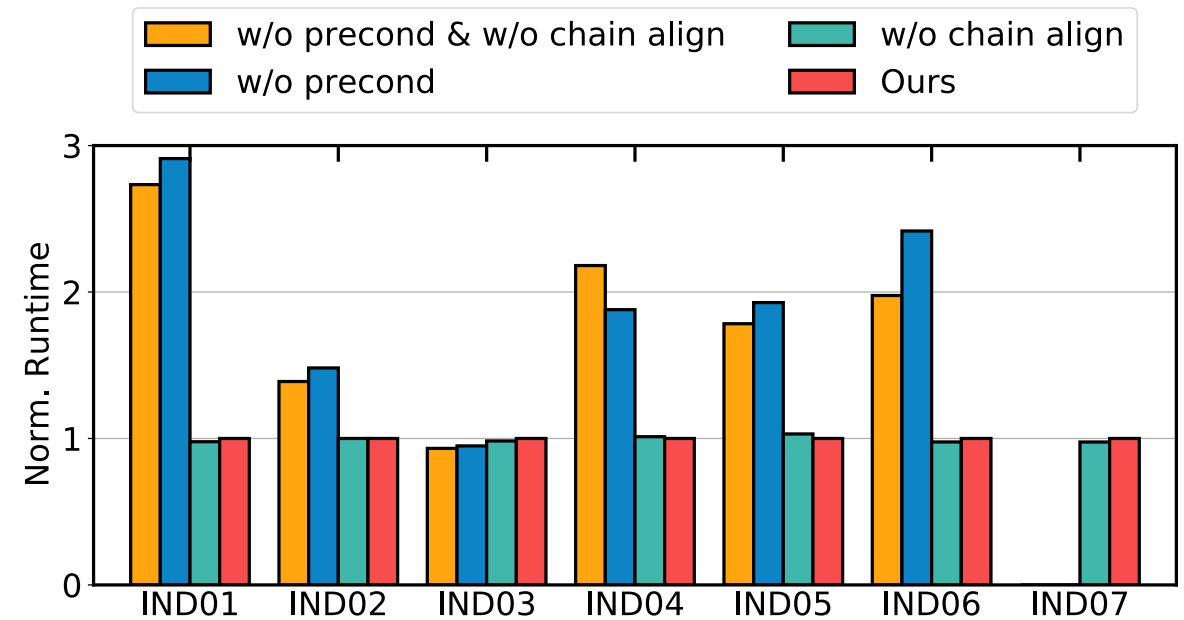
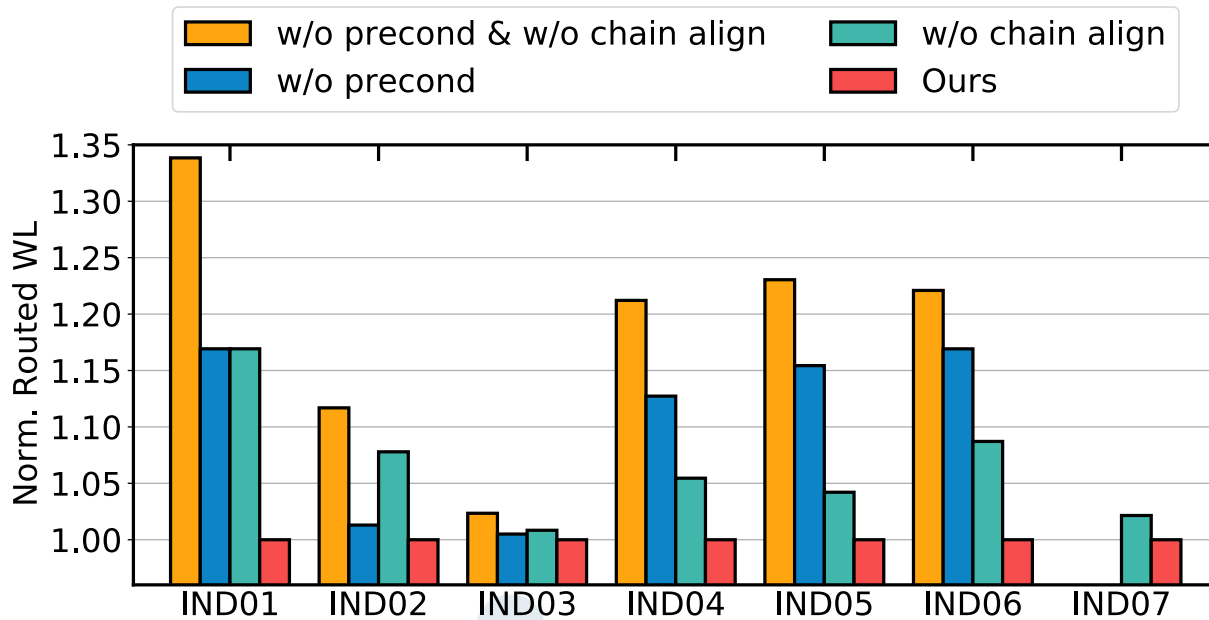
- ▶ **1/7** designs diverge
- ▶ **+19.3%** routed wirelength
- ▶ **+84.1%** runtime

w/o precond

- ▶ **1/7** designs diverge
- ▶ **+10.7%** routed wirelength
- ▶ **+93.5%** runtime

w/o chain alignment

- ▶ No divergence
- ▶ **+6.6%** routed wirelength
- ▶ Almost the same runtime



Designs with {LUT, FF, DSP, BRAM, IO,
Distributed RAM, SHIFT, CARRY}

Outline

Motivation & Challenges

Proposed Algorithm

Experimental Results

Conclusion & Future work

Conclusion & Future Work

Conclusion

- ▶ A multi-electrostatic FPGA placement algorithm, considering **SLICEL-SLICEM** heterogeneity
- ▶ A nested optimization paradigm for wirelength, routability, and clock feasibility
- ▶ Divergence-aware preconditioning technique
- ▶ Smooth clock penalization technique
- ▶ **7.9-14.2%** better wirelength with **1.45-6.58X** speedup

Future Work

- ▶ Timing-driven placement
- ▶ Parallelize legalization and detailed placement

Thank You!

