IEEE/ACM ICCAD 2026 · San Jose, CA · Paper 39

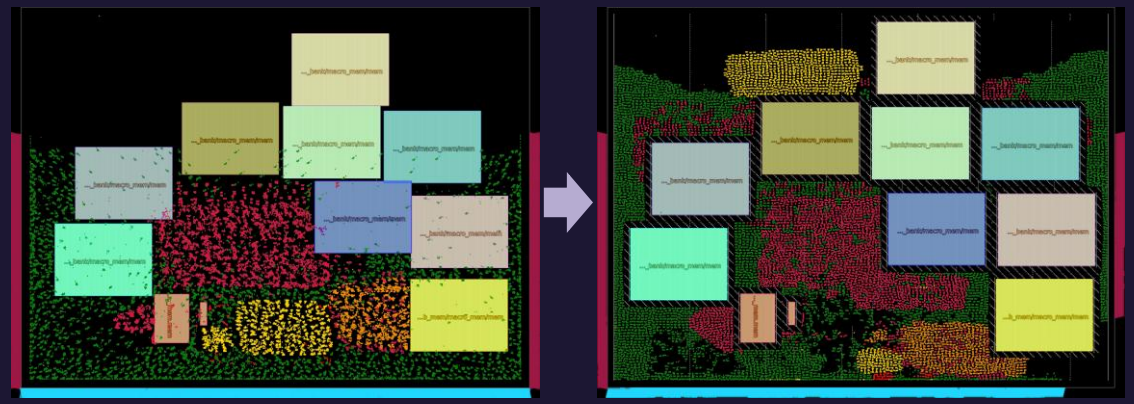
Quality-Preserving Legalization for Analytical Floorplanning

Yunqi He, You Li, Ruofan Huang, Hai Zhou

Northwestern University, Evanston, IL

Session: Scalable Optimization for Floorplanning, Partitioning, and Chiplet-Based Design

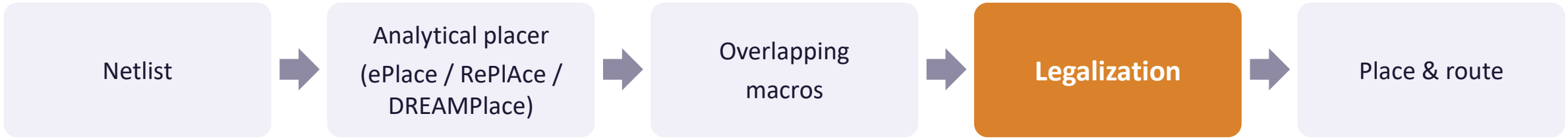
November 9, 2026 · San Jose Ballroom



Analytical floorplan (RePlAce) Legal floorplan (ours)

bp_fe_top, nangate45, OpenROAD

Analytical floorplanning leaves a bottleneck: legalization



1 Positions

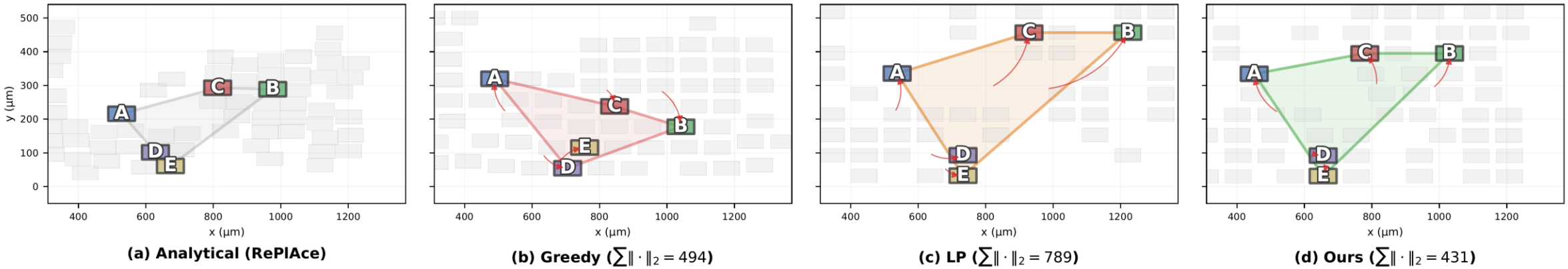
Where each macro sits. Optimized by the placer to minimize wirelength under the density penalty, together with the standard cells.

2 Topology

How macros relate: which is left of, right of, above, or below which. Determines where standard-cell clusters land, hence timing, congestion, and routability.

Existing legalizers minimize displacement only. Topology is left to chance.

A small change in topology costs timing



Five data-cache macros (A–E) of ariane133. Red arrows: displacement before and after legalization. $\Sigma\|\cdot\|_2$ = total ℓ^2 displacement in µm.

Greedy

3 pairs flipped

- $\Sigma\|\cdot\|_2$ = 494 µm
- WNS −0.32 ns (worst)
- Fixed die

LP

Orders kept

- $\Sigma\|\cdot\|_2$ = 789 µm (largest)
- WNS −0.21 ns
- Die enlarged by 44%

Ours

Orders kept

- $\Sigma\|\cdot\|_2$ = 431 µm (smallest)
- WNS −0.20 ns (best)
- Fixed die

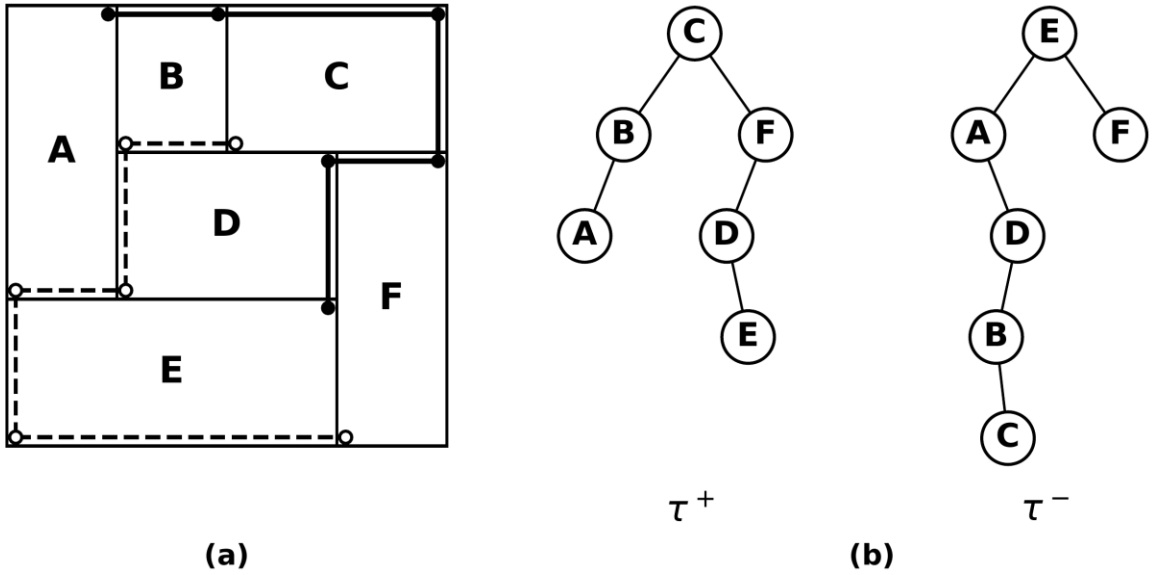
Our idea: legalize without undoing the placer's decisions

Prior approach	How it legalizes	What goes wrong
Greedy	Place macros sequentially at the closest legal spot	Flips relative orders; scanning order pushes some macros far away
Simulated annealing	Random moves on sequence pair / B*-tree	Topology drifts from the analytical solution; many iterations
Linear programming	Pairwise non-overlap constraints from a constraint graph	$O(n^2)$ redundant constraints over-constrain; often needs die expansion
RL on a grid	Agent places macros on a discretized grid	Restricted solution space; needs a refinement step

OUR APPROACH

1. **Capture** the adjacency of the analytical floorplan with twin binary trees
2. **Fix** the trees, so the topology is invariant by construction
3. **Optimize** only the filler-block sizes, by gradient descent, to minimize displacement
4. **Guarantee** a legal layout at every iteration, with no die expansion

Background: mosaic floorplans and twin binary trees



(a) a mosaic floorplan with six blocks; (b) its twin binary trees τ^+ and τ^- (Yao et al., 2003)

Why twin binary trees

- Non-redundant: one pair of trees per mosaic floorplan
- Complete adjacency information, unlike a single B*-tree
- Block coordinates are continuous functions of block sizes

1 **Mosaic floorplan**

Partition of the chip into non-overlapping rectangles; every internal boundary ends in a T-junction, never a cross.

2 **Twin binary trees (τ^+, τ^-)**

Same in-order traversal; complementary 2-labelings. One-to-one with mosaic floorplans.

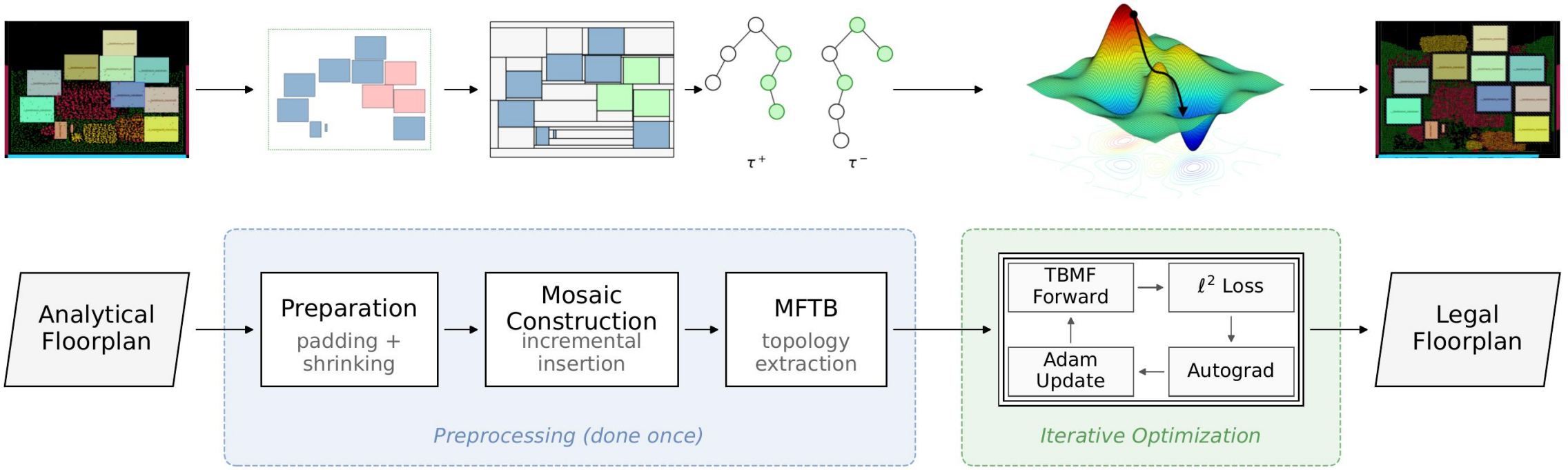
3 **MFTB: mosaic → trees**

Parent of a block = the block at its upper-right T-junction (τ^+) or lower-left T-junction (τ^-). O(n).

4 **TBMF: trees → mosaic**

Build horizontal and vertical constraint graphs from the trees; coordinates = longest paths. O(n), never overlaps.

Pipeline: preprocess once, then iterate



Preprocessing (done once)

Padding + uniform shrinking → incremental mosaic construction → MFTB extracts (τ^+, τ^-). Structural stages finish in under a second.

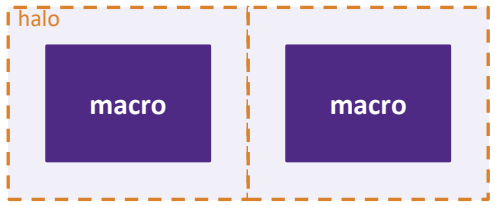
Iterative optimization

TBMF forward → ℓ^2 displacement loss → autograd → Adam step on filler sizes. Macros keep their real size; the trees never change.

Preparation and mosaic construction

1 Macro padding

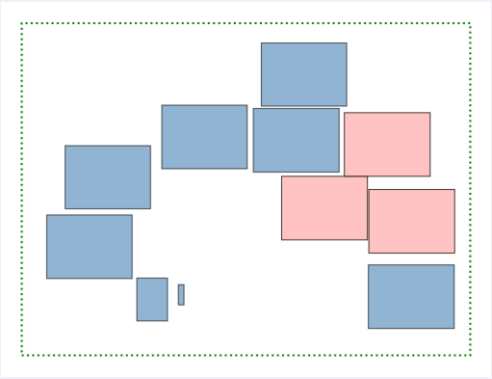
- Uniform halo added to each macro in the LEF before the analytical placer runs
- The halo enforces minimum macro spacing, so the analytical floorplan already reflects spacing-aware relations



Halos touch ⇒ macros keep the minimum spacing

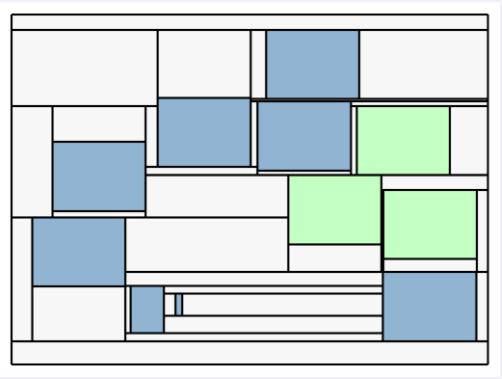
2 Uniform shrinking

- $w' = s \cdot w$, $h' = s \cdot h$, center unchanged
- s^* = largest s with no macro overlap, by binary search in $O(n^2 \log 1/\varepsilon)$
- Orientations are kept from the placer

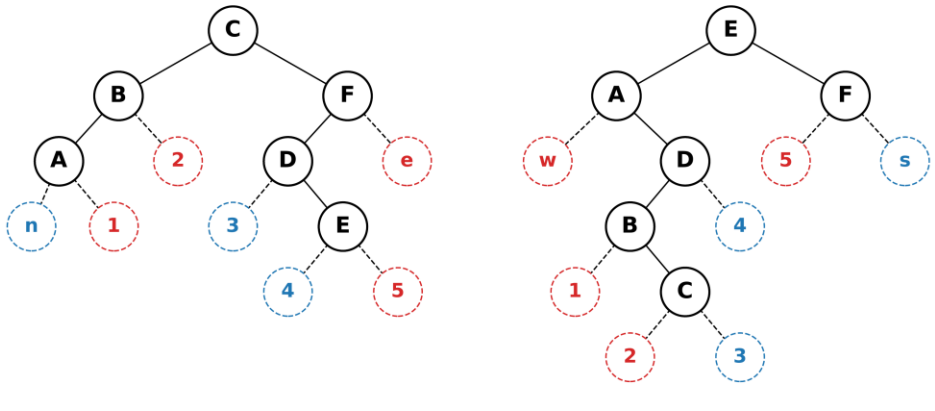


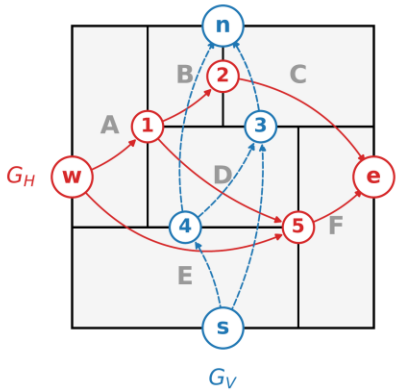
3 Incremental mosaic builder

- Start with the chip as one filler; insert macros one by one
- Each cut extends one pair of opposite edges: T-junctions only, never a cross
- Merge fillers sharing a full edge; slicing tree as a simpler alternative



TBMF as a differentiable forward model





1 Boundaries become vertices

Augment each tree with virtual leaves; a leaf is one boundary of its block (top/right in τ^+, left/bottom in τ^-). Each block is an edge weighted by its width (horizontal graph) or height (vertical graph).

2 Coordinates are longest paths

x_v = max over incoming edges of (x_v' + w_e). Only additions and max: piecewise linear in the filler sizes. Each max routes the gradient to its current longest predecessor, like a ReLU.

3 Loss on filler sizes only

$L(f) = \Sigma_i \|c_i(f) - c_i^{init}\|^2$ over macro centers. Macros keep their real dimensions; f = widths and heights of the fillers. Optimized with Adam via autograd.

4 Guarantees (Theorems 1 and 2)

For any sizes $s \geq 0$ the reconstructed floorplan has no overlaps, and the relative positions of all blocks do not depend on s. Legal and topology-preserving at every iteration.

Experimental setup

Four research questions

RQ1 **Topology preservation**

Does preserving topology during displacement minimization yield more consistent post-route PPA?

RQ2 **End-to-end competitiveness**

Can analytical floorplanning + our legalization compete with state-of-the-art macro placers?

RQ3 **Design choices**

How much does each pipeline stage contribute, and does the optimization converge stably?

RQ4 **Efficiency**

How do legalization and end-to-end runtime scale with design complexity?

Flows and benchmarks

- OpenROAD v2.0 flow (RePlAce, OpenDP, TritonCTS, TritonRoute): all 8 macro-containing nangate45 designs, 2–136 macros
- Cadence flow (Genus 21.18 + Innovus 21.19): ariane133 (133 macros) and bp_quad (220 macros), on nangate45 and ASAP7
- Both TILOS MacroPlacement designs with RTL and published baselines

Baselines (all metrics post-route)

- Legalizers (RQ1): Greedy (sequential, nearest overlap-free spot) and LP (constraint graph, minimum total displacement)
- OpenROAD macro placers: Hier-RTLMP (sequence pair + SA), TritonMacroPlace (B*-tree + SA)
- Cadence: CMP (Cadence Concurrent Macro Placer), SA, Circuit Training (CT_AC, CT_Scratch), human expert; from the TILOS repository

RQ1: preserving topology gives consistent post-route PPA

Design	#M	Mean displacement (µm)			WNS (ns)			LP die
		Ours	Greedy	LP	Ours	Greedy	LP	
ariane133	132	**85**	115	217	**–0.20**	–0.32	–0.21	+44%
ariane136	136	**101**	115	225	**0**	0	0	+42%
black_parrot	24	**22**	27	32	**–0.45**	fails	–0.19	+12%
bp_be_top	10	**22**	41	54	**0**	0	0	+4%
bp_fe_top	11	**11**	31	12	**0**	0	0	0%
bp_multi_top	26	**42**	58	73	**–2.06**	–1.85	–2.01	+29%
swerv_wrapper	28	**14**	9	8	**0**	0	–0.11	+25%
tinyRocket	2	**3**	3	9	**0**	0	0	0%

Same RePlAce output legalized by all three methods; OpenROAD flow, nangate45. Greedy “fails”: downstream flow failed (N/A). LP die: die enlargement LP needs to be feasible.

Greedy

Max displacement up to 145 µm; worst WNS on ariane133; breaks the flow on black_parrot

LP

Keeps orders, but needs die expansion on 6 of 8 designs, up to +44%

Ours

Lowest mean displacement on 7 of 8 designs, fixed die, topology preserved

RQ2: competitive with OpenROAD's macro placers end to end

Design	#M	WNS (ns)			Routed wirelength (Mμm)		
		Ours	Hier-RTLMP	TritonMP	Ours	Hier-RTLMP	TritonMP
ariane133	132	**−0.20**	−0.49	−0.25	**5.83**	6.82	6.40
ariane136	136	**0**	0	0	**6.64**	7.04	6.75
black_parrot	24	**−0.45**	−0.81	−0.67	**7.78**	8.22	8.07
bp_be_top	10	**0**	0	0	**2.08**	2.29	2.34
bp_fe_top	11	**0**	0	0	**1.41**	**1.34**	1.49
bp_multi_top	26	**−2.06**	**−1.81**	−1.94	**3.50**	3.86	4.01
swerv_wrapper	28	**0**	−0.08	0	**3.88**	3.99	**3.75**
tinyRocket	2	**0**	0	0	**0.60**	0.61	0.61

OpenROAD flow, nangate45. Ours = RePlAce + our legalization. Bold = best in row. TNS follows the same pattern (paper, Table 2).

7 of 8

designs with best or tied-best WNS and TNS

6 of 8

designs with the shortest routed wirelength

Baselines

Hier-RTLMP: hierarchical, sequence pair + SA. TritonMacroPlace: B*-tree + SA. Both are OpenROAD's built-in macro placers.

RQ2: on a commercial flow, PPA comparable to CMP

Method	ariane133 · NG45 (1.3 ns)		bp_quad · NG45 (1.3 ns)		ariane133 · ASAP7 (0.9 ns)		bp_quad · ASAP7 (0.85 ns)	
	WNS (ps)	rWL (Mµm)	WNS (ps)	rWL (Mµm)	WNS (ps)	rWL (Mµm)	WNS (ps)	rWL (Mµm)
Ours (RePlAce + legalization)	**−100**	**4.41**	**−221**	**26.4**	**−12**	**0.93**	**−113**	**7.00**
CMP (Innovus 21.19)	−167	3.97	−204	25.3	−12	0.93	−116	6.01
CMP (Innovus 21.10, TILOS)	−154	4.06	−144	23.1	−124	0.84	−111	6.10
SA	−126	3.98	−230	33.2	−124	0.89	−120	7.27
CT_AC (Circuit Training)	−88	5.00	−230	33.2	−108	1.01	−201	8.88
CT_Scratch	−140	4.63	−205	47.8	−142	1.15	−203	11.4
Human expert	−88	4.47	−97	25.9	−104	1.18	−89	6.52

Post-route results on the Cadence Innovus flow. Ours and CMP 21.19 run by us; the other rows are TILOS MacroPlacement results (Innovus 21.10). TNS and power in the paper, Table 3.

Shorter wirelength than CT

than Circuit Training in all four configurations, with fewer timing violations in three of four

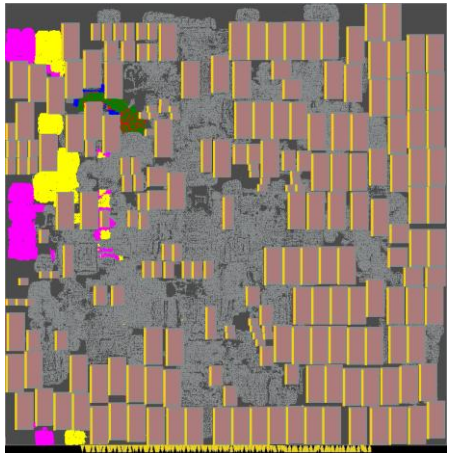
PPA comparable to CMP

identical on ariane133 ASAP7; within a few percent elsewhere

Minutes on one CPU

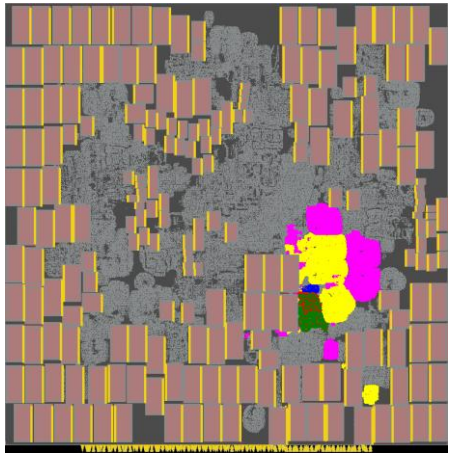
versus hours on 80 CPUs (SA) or 8 GPUs + 576 CPUs (CT); details on the runtime slide

Post-route layouts of bp_quad (nangate45, 1.3 ns clock)



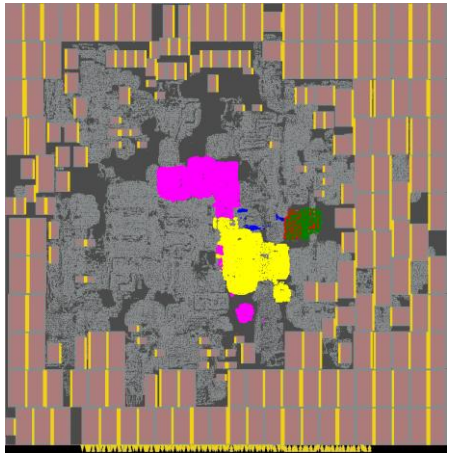
Ours

rWL 26.4 Mμm · WNS −221 ps



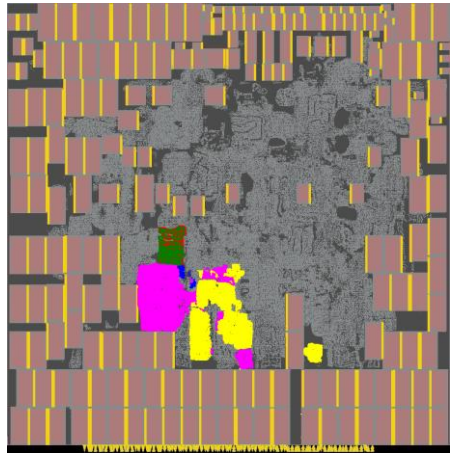
CMP

rWL 25.3 Mμm · WNS −204 ps



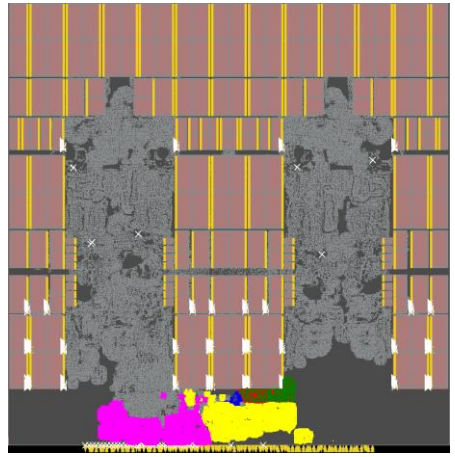
SA

rWL 33.2 Mμm · WNS −230 ps



CT_AC

rWL 33.2 Mμm · WNS −230 ps



Human

rWL 25.9 Mμm · WNS −97 ps

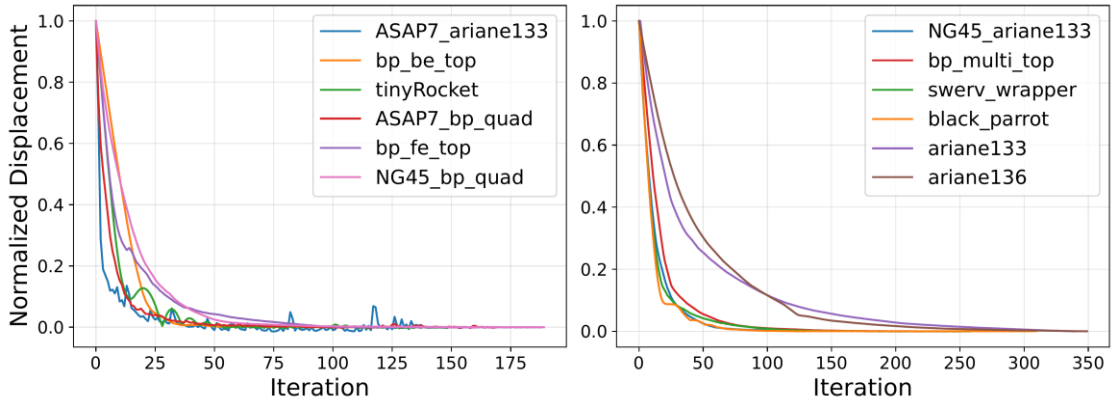
Reading the layouts

- Same netlist, same Innovus flow; only the macro placement differs. Macros are the rectangular blocks; colored regions are selected logical modules.
- Ours preserves the analytical topology: macros grouped toward the periphery, contiguous standard-cell area in the middle.
- Routed wirelength: ours within 5% of CMP and the human expert; SA and CT_AC need about 25% more.

RQ3: input preparation matters; the optimizer converges

Configuration (ariane133, Cadence)	Mean	Max	Median	Viol.
Slicing tree builder	78.6	311.9	58.2	11
Slicing tree builder + macro padding	66.7	269.2	47.4	11
Mosaic builder	76.8	179.9	77.1	16
Mosaic builder + macro padding	**12.7**	**80.9**	**10.2**	**5**

ℓ^2 displacement in µm; Viol. = boundary violations before post-fit.



Normalized displacement vs. iteration, all 10 benchmarks (Cadence left, OpenROAD right)

6×

lower mean displacement with padding + mosaic builder (78.6 → 12.7 µm)

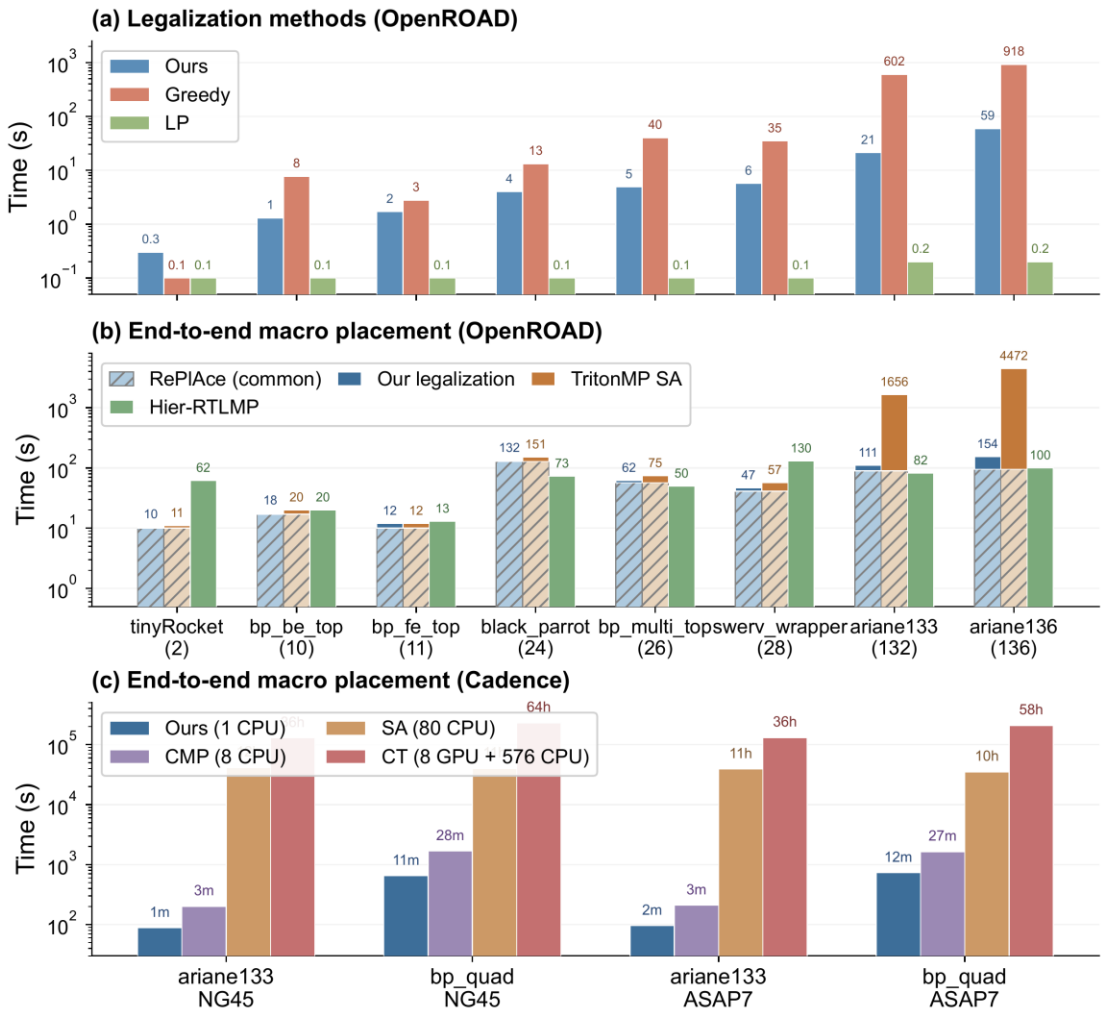
4×

lower maximum displacement (311.9 → 80.9 µm)

One optimizer setting for every design

- Adam, learning rate 1.0, gradient clipping at 100
- Early stop at 0.1% relative improvement
- Converges smoothly in a few hundred iterations on all 10 benchmarks
- No per-design hyperparameter tuning

RQ4: seconds to legalize, minutes end to end



< 1 minute

to legalize any OpenROAD design: 0.3–59 s on one CPU; structural stages take under a second, the rest is gradient descent

63–74× faster

than TritonMacroPlace end to end on ariane133 and ariane136 (RePlAce + our legalization vs. SA-based macro placement)

1–12 min on 1 CPU

end to end on the Cadence flow, vs. CMP 3–28 min on 8 CPUs, SA 10–12 h on 80 CPUs, Circuit Training 36–64 h on 8 GPUs + 576 CPUs

Conclusion

1 Finding

Topology preservation is a key factor in legalization quality

Confirmed on OpenROAD and Cadence flows: a legalizer that keeps both positions and topology gives consistent post-route PPA.

2 Model

A piecewise-linear, legal-by-construction optimization

Twin binary trees fix the topology; filler sizes are the only variables; TBMF is the differentiable forward model. No overlaps at any iteration.

3 Workflow

Analytical floorplanning becomes competitive end to end

RePlAce + our legalization matches or surpasses commercial and open-source macro placers in timing and wirelength, in minutes on one CPU.

Future work

- Layout optimization with a generative model or an RL agent on top of the same differentiable model
- Hierarchical legalization with super-blocks for very large macro counts; first-class fence and region constraints

Thank you

Questions?

Yunqi He · yunqi.he@u.northwestern.edu

Northwestern University

Paper: https://doi.org/10.1145/3831252.3833939

Scan for the paper, slides, and details

actasclown.github.io/papers/qp-legalization-iccad26