MAPLE: Multilevel Adaptive PLacEment for Mixed-Size Designs

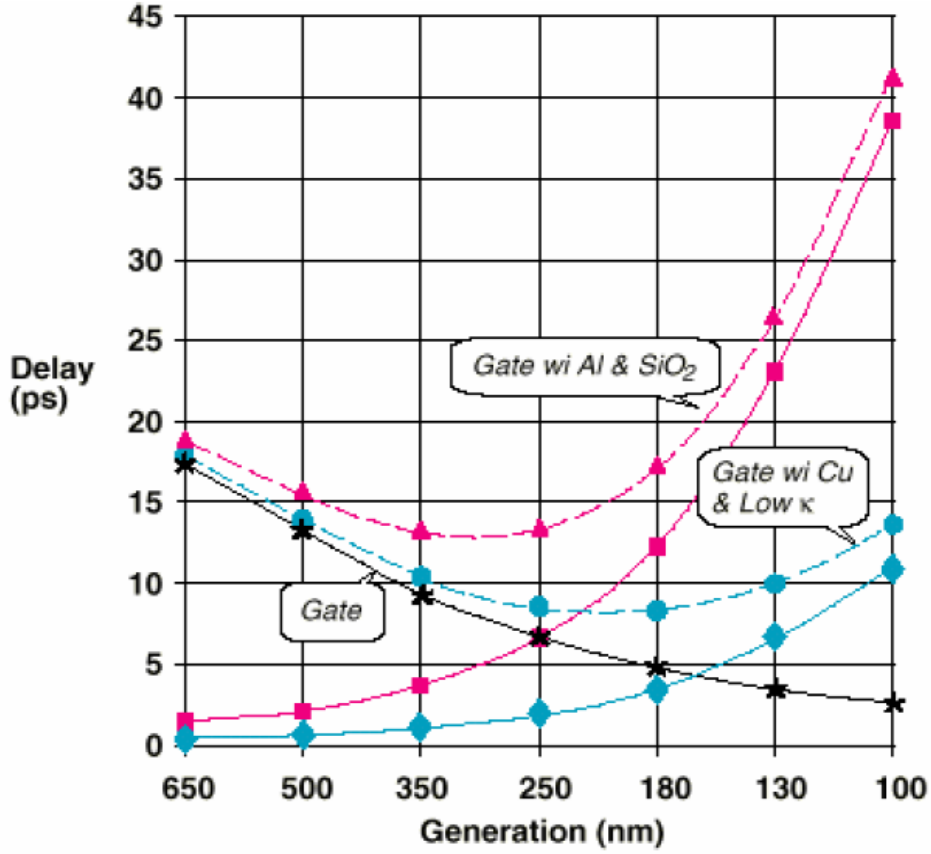
Myung-Chul Kim[†], Natarajan Viswanathan[‡], Charles J. Alpert[‡], Igor L. Markov[†], Shyam Ramji[‡]

[†] Dept. of EECS, University of Michigan

[‡]IBM Corporation

Motivation: Interconnect-driven Placement

- **Interconnect lagging in performance while transistors continue scaling**
  - Circuit delay, power dissipation and area dominated by interconnect
  - Routing quality highly controlled by placement

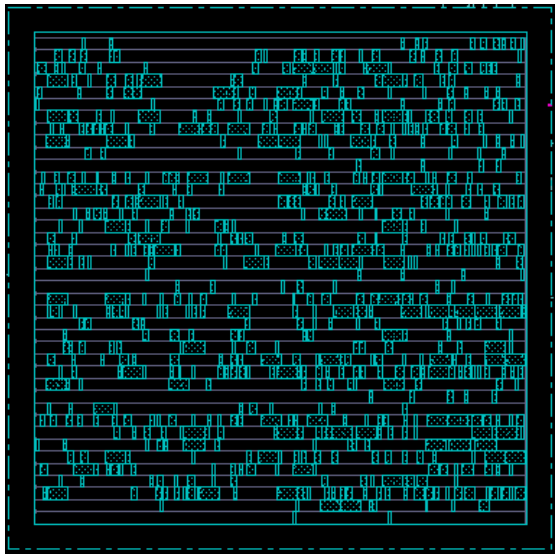


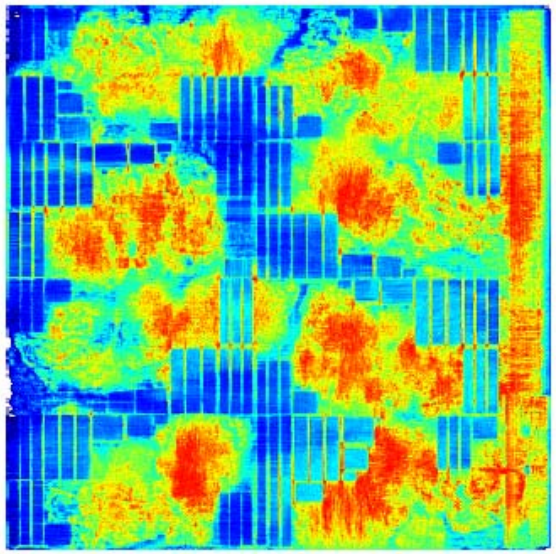
- **Interconnect-driven placement remains one of the most influential optimization in physical design**
  - The choice of the wirelength-driven placement engine is paramount even in multi-objective placement

Placement Formulation

- **Objective:** Minimize estimated wirelength (Half-Perimeter WireLength)

- **Subject to *constraints*:**
  - **Legality**: Row-based placement with no overlaps
  - **Routability**: Limiting local interconnect congestion for successful routing
  - **Timing**: Meeting performance target of a design





Perspectives

- **Comparisons and trade-off between linear and quadratic wirelength functions**
  - Is there a tangible gap between B2B net model and HPWL objective in practice?
  - Can quadratic optimization with linear net model be effectively improved on multi-million gate netlists?
  - Is multilevel placement optimization compatible with B2B net model and competitive in performance ?
- **Methodology for module spreading and handling of whitespace**
- **The composition of multiple optimizations into a high-precision, reliable multi-objective optimization process**

Key features of MAPLE

- **A multilevel force-directed placement algorithm**
  - The coarsest level placement – a variant of SimPL
  - Multilevel extensions reinforced by Progressive Local Refinement (ProLR)
  - Techniques to avoid or suppress disruptions inherent in analytic placement algorithms
  - Adaptive to current placements relying on a new placement density metric – ABUγ
  - Handling of movable macros

- **MAPLE produces strong results both in wirelength and the quality of spreading on standard benchmarks**

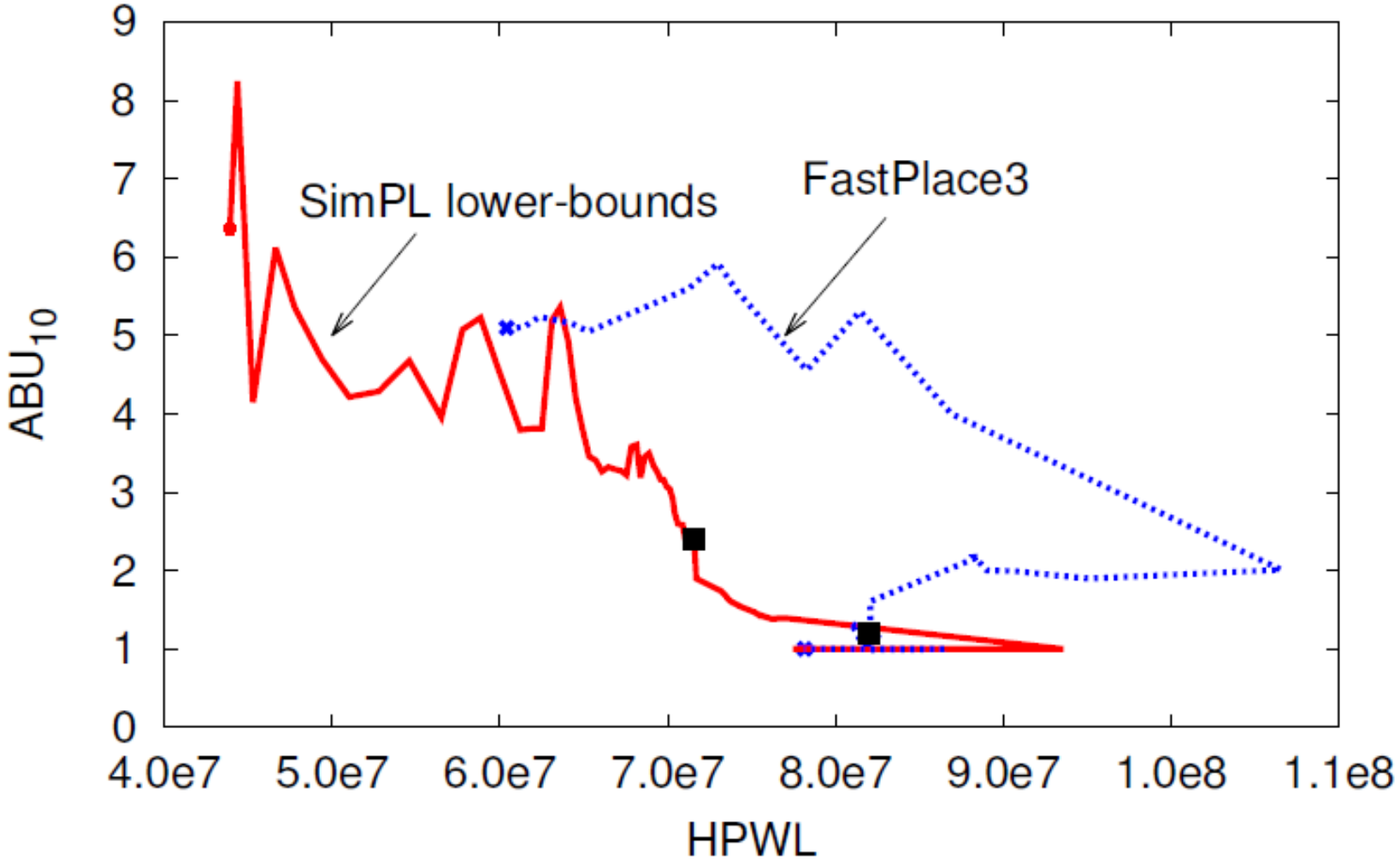
A Placement Density Metric – ABUγ (1)

- **Density metrics during global placement**
  - Provide insights into the quality of module spreading in intermediate placements
  - Estimate wirelength impact of legality enforcement
  - Global placer can adaptively adjust its parameters

- **ABUγ: Average Bin Utilization of the top γ% densest bins**
  - Reflects the nonuniformity of module distribution
  - More intuitive than overflow-based metrics
  - Enables comparisons of different parameter settings and even different analytical placers' iterations

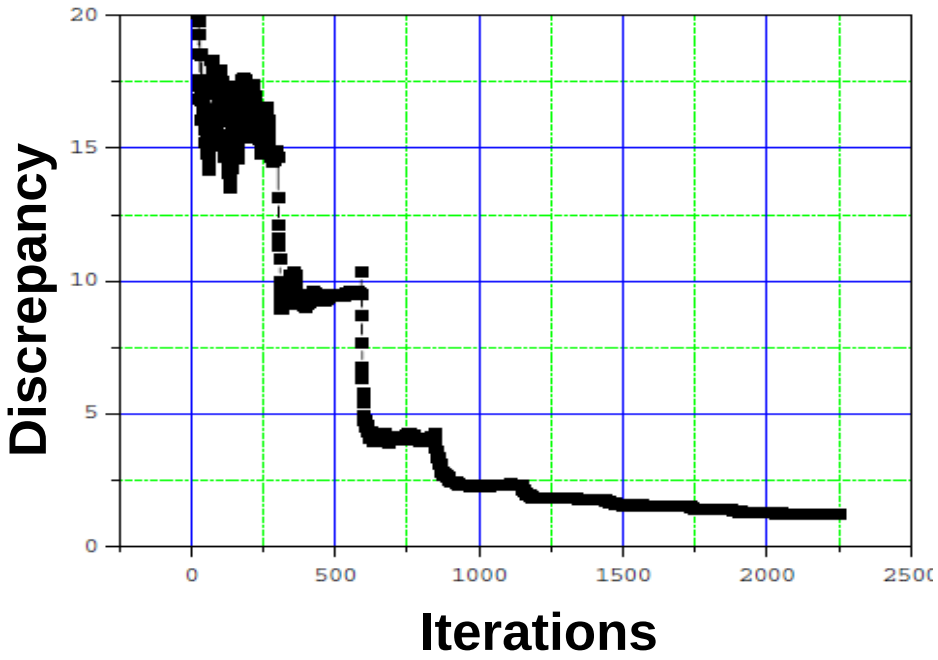
A Placement Density Metric – ABUγ (2)

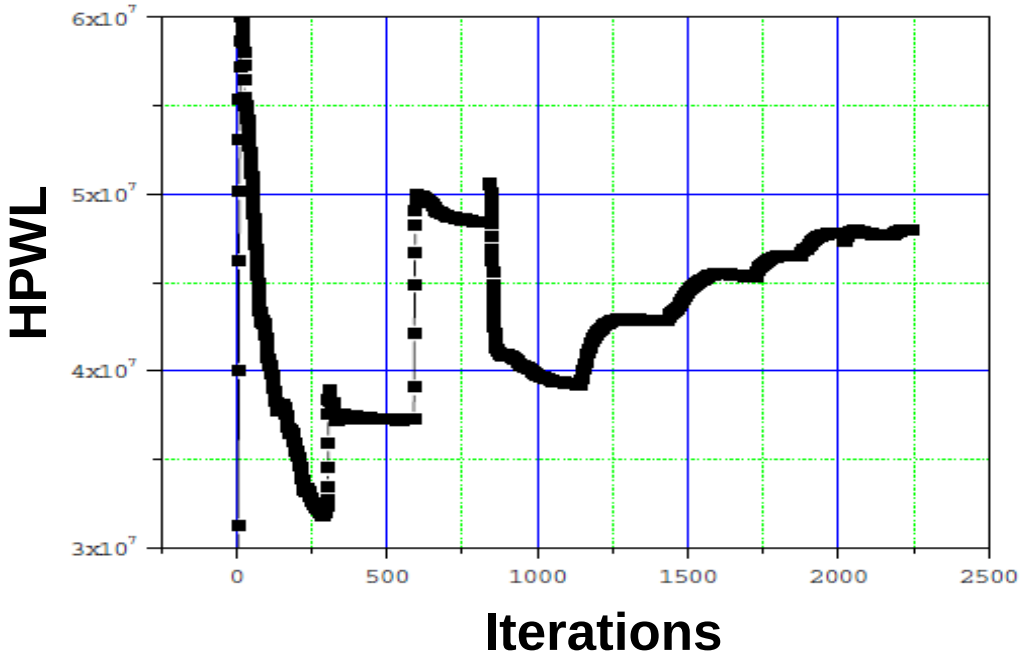
- **Comparisons with different placers speed up new algorithm development**



Analysis of Noise during Analytical Opt. (1)

- **Unclustering**
  - Often include changes to the optimization objectives as well as the netlist
  - When wirelength weight is decreased, wirelength and module density sharply change and then refined

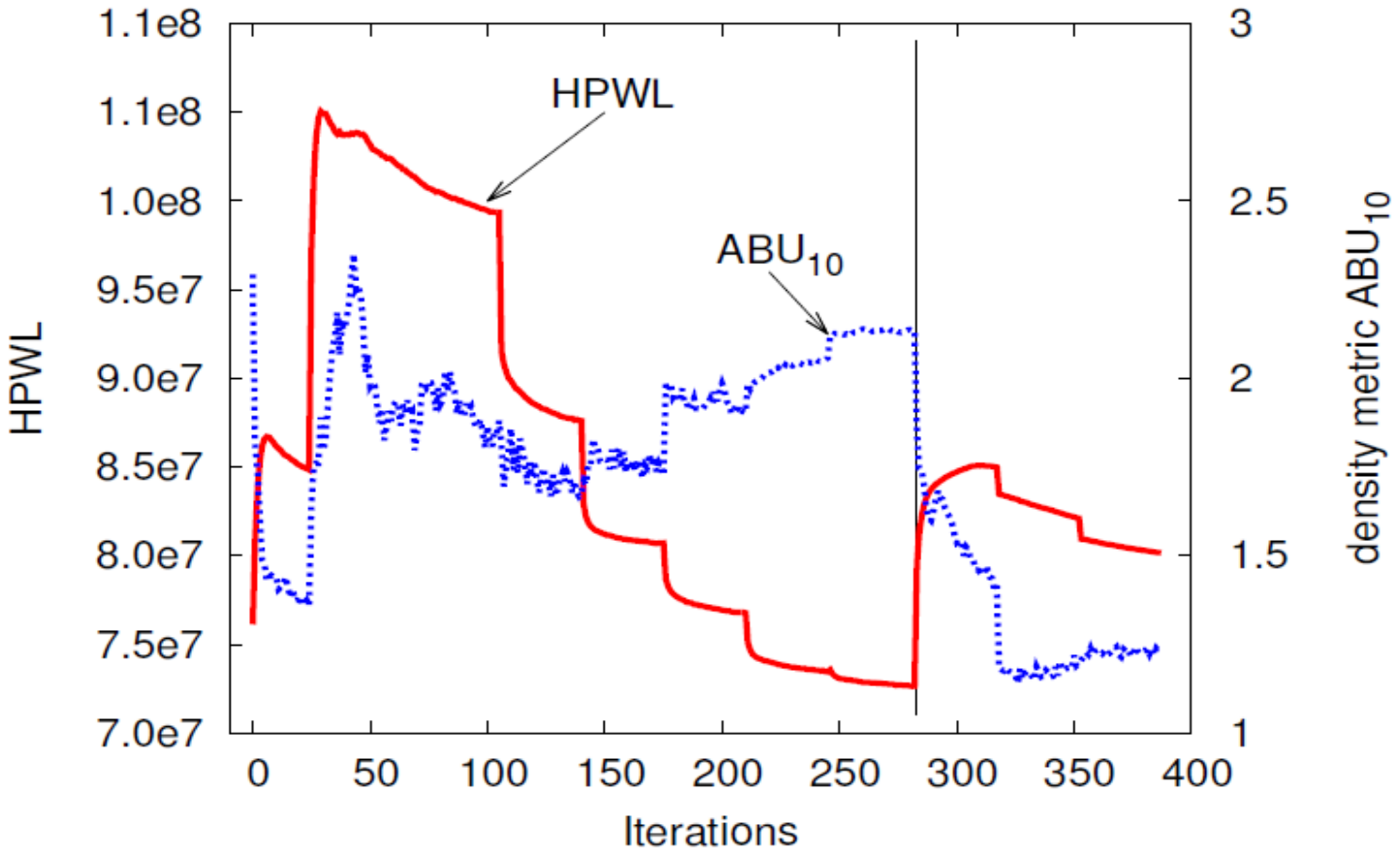




Figures are from A. B. Kahng, Q. Wang, "Implementation and Extensibility of an Analytic Placer", IEEE TCAD 24(5), 2005

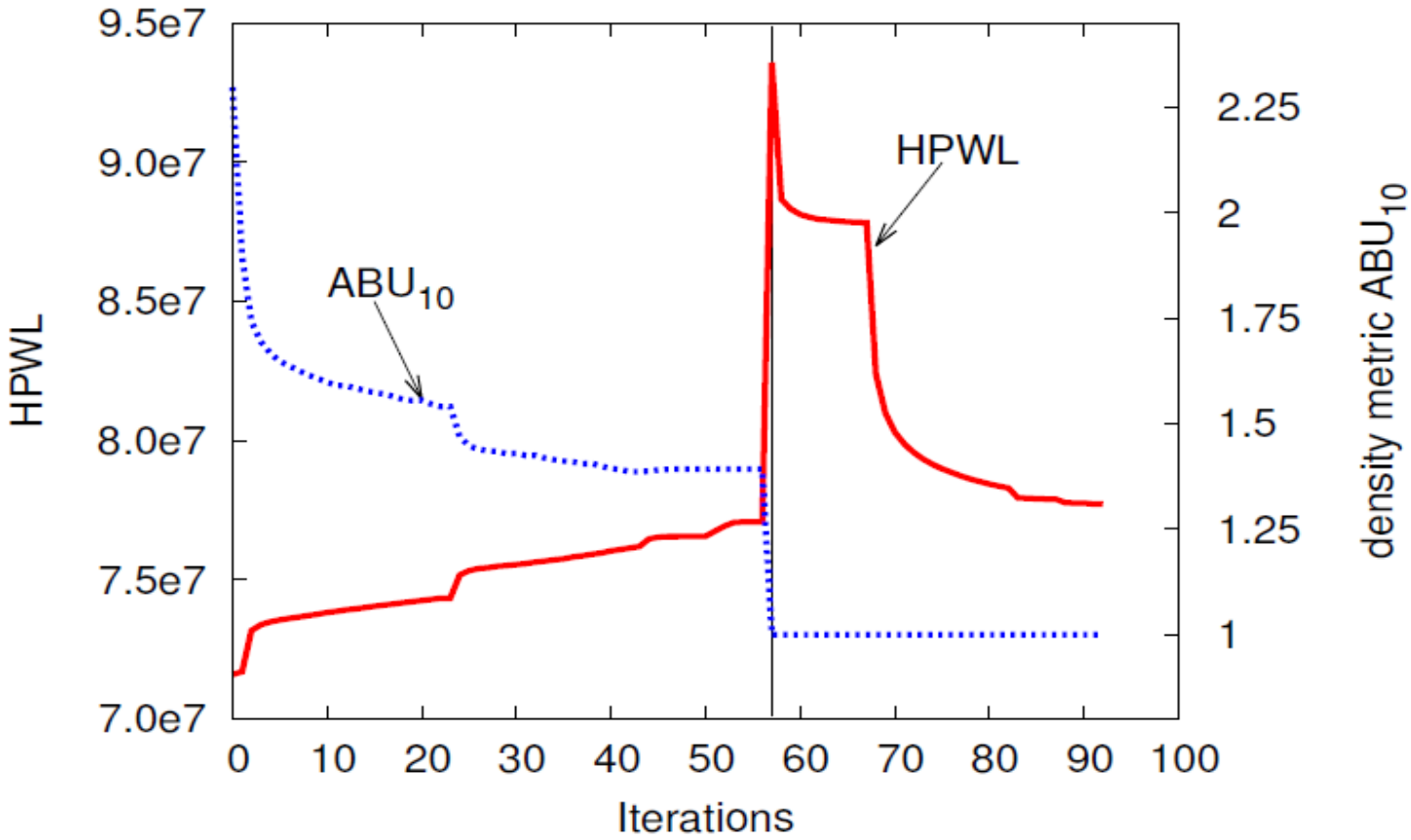
Analysis of Noise during Analytical Opt. (2)

- **Transition to the HPWL objective**
  - Quadratic optimization-based placers often use techniques to recover HPWL
  - ILR [FastPlace, DPlace2, RQL] increasingly penalize dense bins and allow abrupt moves to decrease local density



Analysis of Noise during Analytical Opt. (3)

- **Hand-off to detailed placement**
  - Global placement solutions may exceed target utilization and undergo significant changes during full legalization
  - Even with detailed placement, such abrupt changes are detrimental to solution quality



Strategies for Mitigating Disruptions

- **Purpose: ensuring gradual transitions between successive optimizations**

- **The overall placement flow is modified at the points where the objective function abruptly changes**

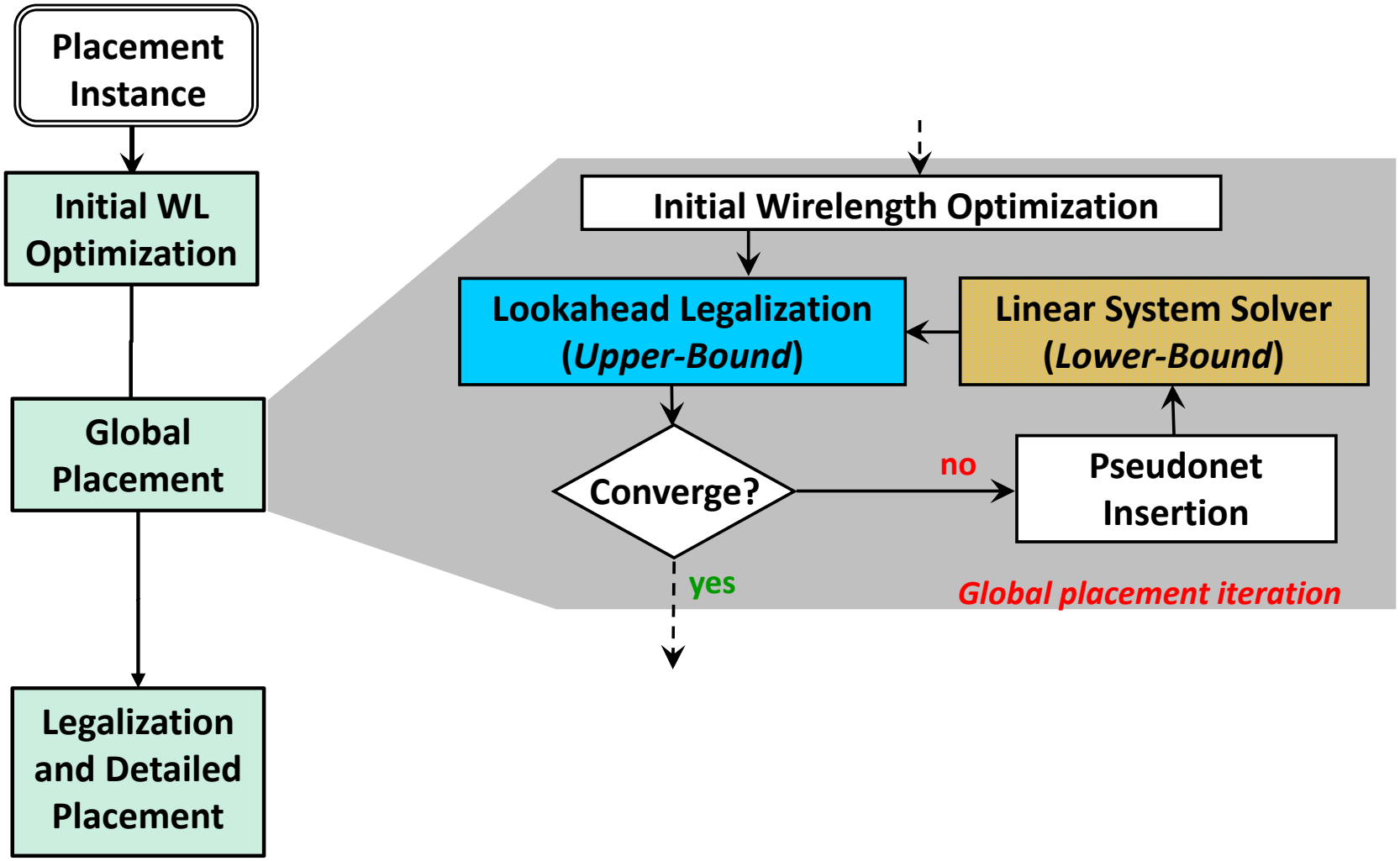
  A. Before/after unclustering and before detailed placement

  B. Optimizes a linear combination of the preceding and succeeding objective functions and adaptively modify parameters according to ABU_{10}

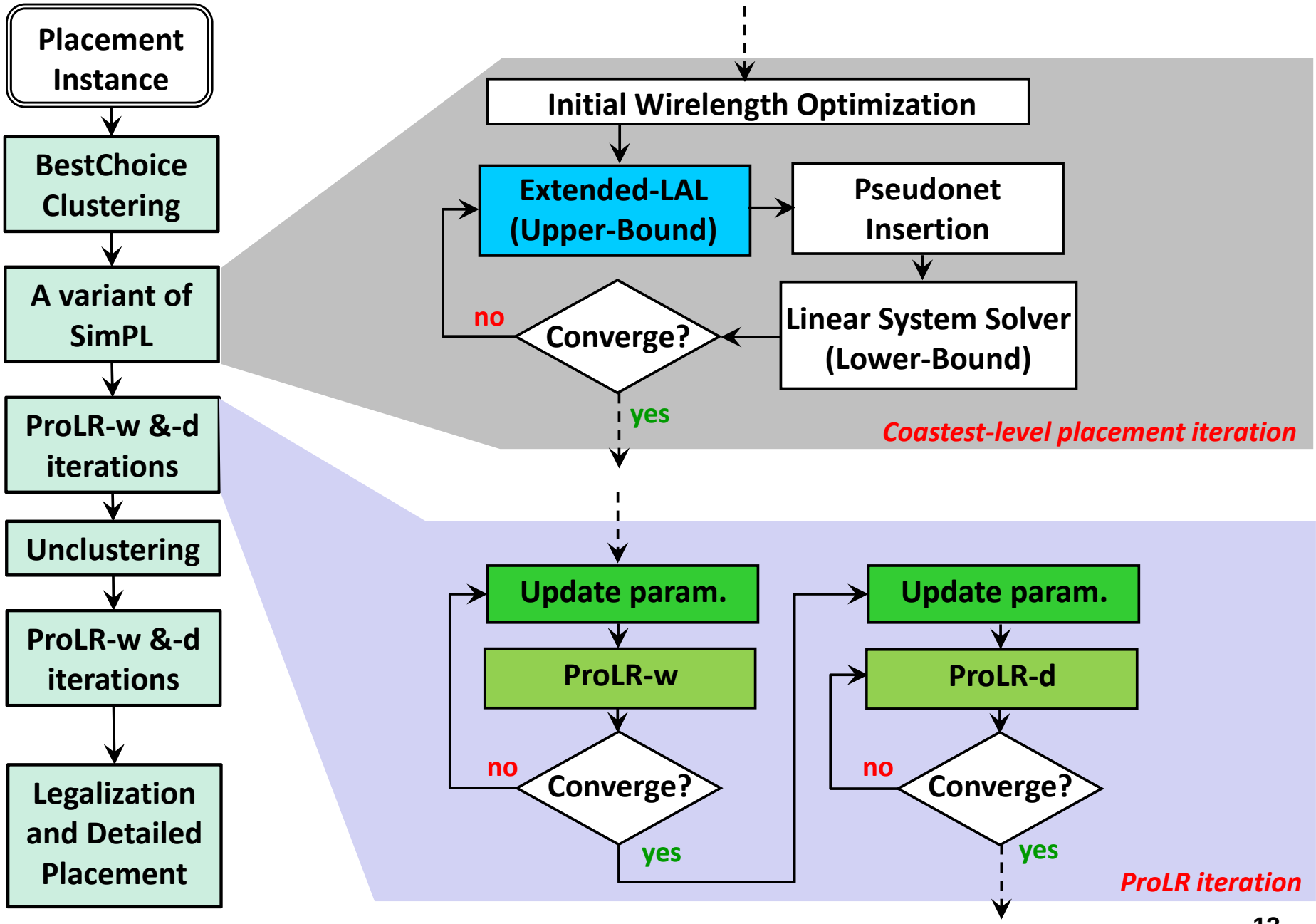
  C. Seek near monotone improvement of either wirelength or module density in a predictable manner w/o disrupting the other objective

- **Our implementation: Progressive Local Refinement (ProLR)**

SimPL Flow


Placement Instance
Initial WL Optimization
Global Placement
Legalization and Detailed Placement
Initial Wirelength Optimization
Lookahead Legalization (*Upper-Bound*)
Linear System Solver (*Lower-Bound*)
Converge?
no
Pseudonet Insertion
yes
Global placement iteration

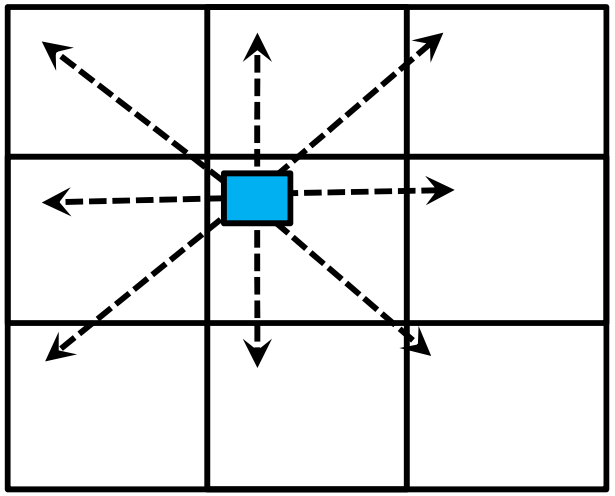
MAPLE Flow


Placement Instance
BestChoice Clustering
A variant of SimPL
ProLR-w &-d iterations
Unclustering
ProLR-w &-d iterations
Legalization and Detailed Placement
Initial Wirelength Optimization
Extended-LAL (Upper-Bound)
Pseudonet Insertion
Linear System Solver (Lower-Bound)
Converge?
no
yes
Coastest-level placement iteration
Update param.
ProLR-w
Converge?
no
yes
Update param.
ProLR-d
Converge?
no
yes
ProLR iteration

A Methodology for Graceful Optimization

- **ProLR adopts single iteration of ILR [FastPlace, RQL] – Local Refinement (LR) – as a baseline and a vehicle for placement modification**

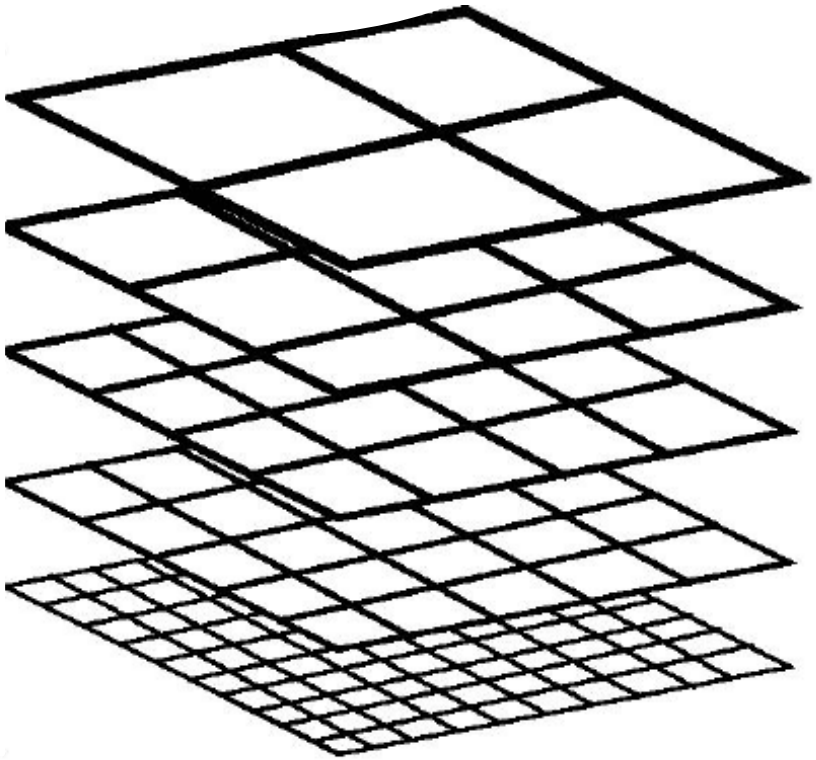
$$\text{Score}(m) = \alpha \cdot \Delta_{HPWL} + \beta \cdot \theta \cdot \Delta_{density}$$



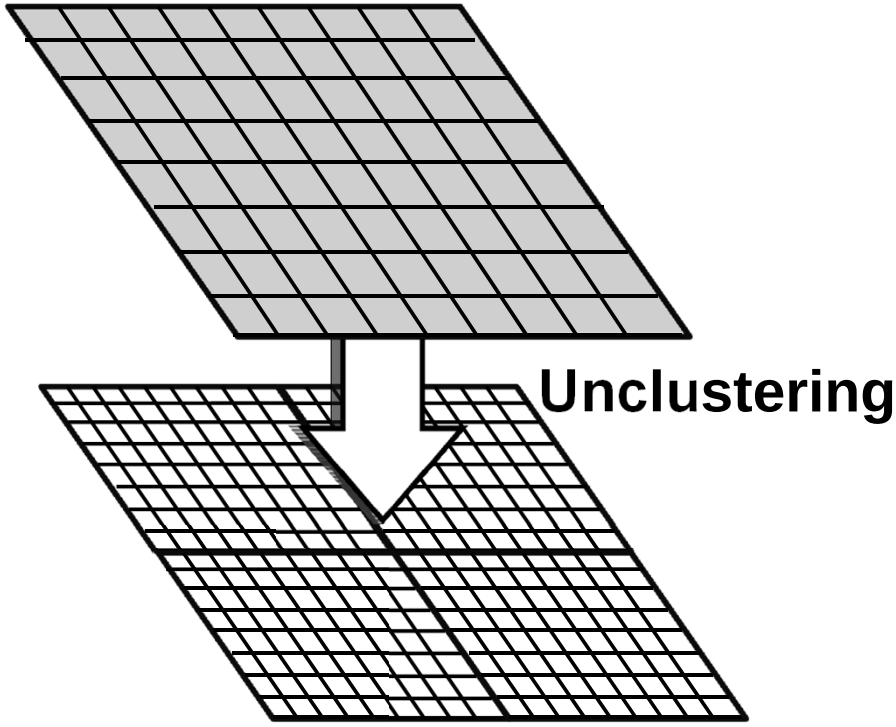
- **But, ProLR promotes gradual traditions via**
  - Limited bin resizing
  - Explicit Bin-Blocking (EBB)
  - A two-tire technique to reduce wirelength and max module density – ProLR-d and ProLR-w

ProLR versus ILR

- **Limited bin resizing**
  - Unlike ILR, the bins in ProLR are small and remain unchanged during each invocation of LR to restrict moves
  - Each bin is 5x the average movable module area



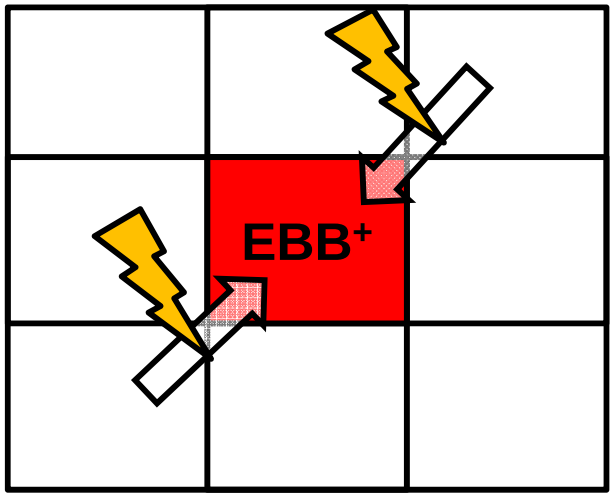
Regular ILR Bin Structure

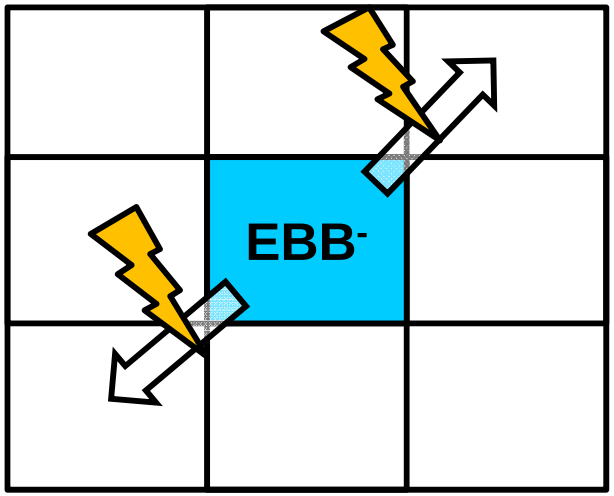


ProLR Bin Structure

ProLR versus ILR

- **Explicit Bin-Blocking (EBB)**
  - Makes local-refinement moves less disruptive
  - ***EBB+*** : For bins whose utilization exceeds ABU_{10}
    - Block the inflow of modules to the bins and redirect modules to other bins
  - ***EBB-*** *:* For bins with below-target utilization
    - Block the outflow of module from the bins and attract modules from remaining bins





ProLR-w and ProLR-d

- **Joint optimization of density and wirelength**

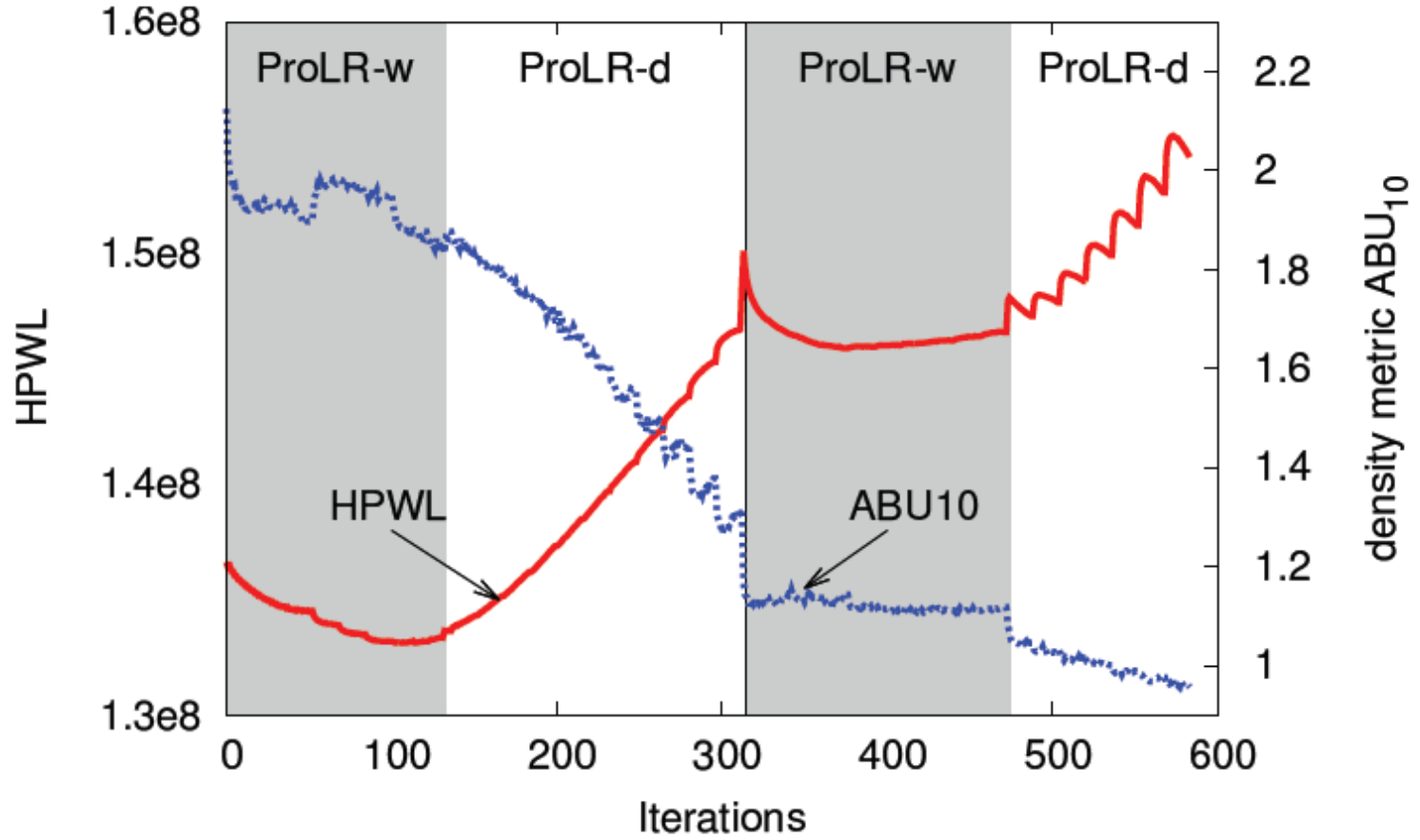
$$\text{Score}(m) = \alpha \cdot \Delta_{HPWL} + \beta \cdot \theta \cdot \Delta_{density}$$

  - But, ProLR performs two simpler optimizations

- **ProLR inspects best moves for each objective and select those that do not harm the other objective**
  - ***ProLR-w*:** Optimizes wirelength
    - Start with small utilization θ_w^0. EBB$^+$ is applied.
    - For flat netlist $\theta_w^1 = \theta_d^{k-1}$
  - ***ProLR-d*:** Optimizes module density
    - $\theta_d^k = \theta_d^{k-1} + \theta_{step}^k$ where $\theta_{step}^k = \theta_{step}^{k-1} \cdot (1 + \frac{ABU_{10}}{100\Upsilon_{target}})$
    - Progressively puts a greater emphasis on spreading over multiple iterations. EBB$^-$ is applied.

Unclustering and Refinement

- **When a cluster is broken down, constituent modules are placed by side by side**
  - The placement is refined by ProLR
  - We schedule ProLR-d before the disruption and ProLR-w after the disruption



Handling of Movable Macro Blocks

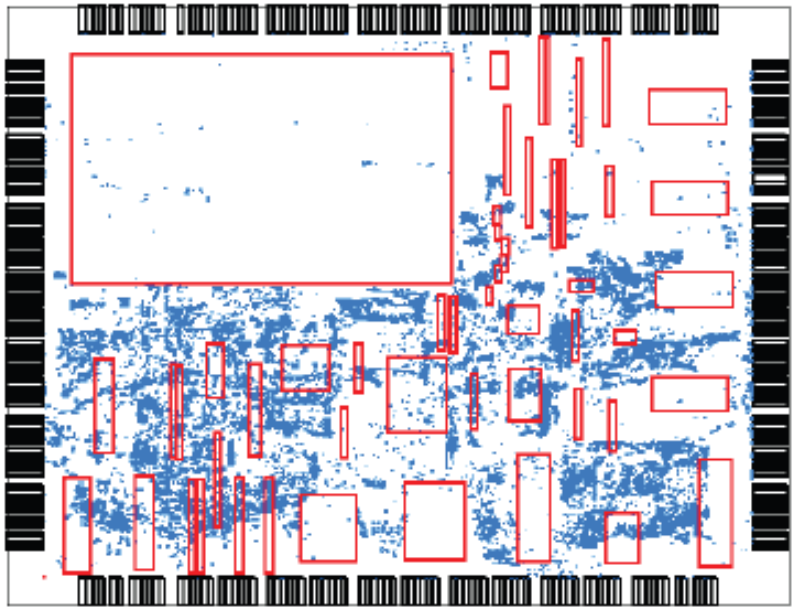
- **We developed E-LAL to handle movable macros and upper-bound placements are generated in two steps:**

(1) Movable macro legalization – a variant of cell shifting [FP2]

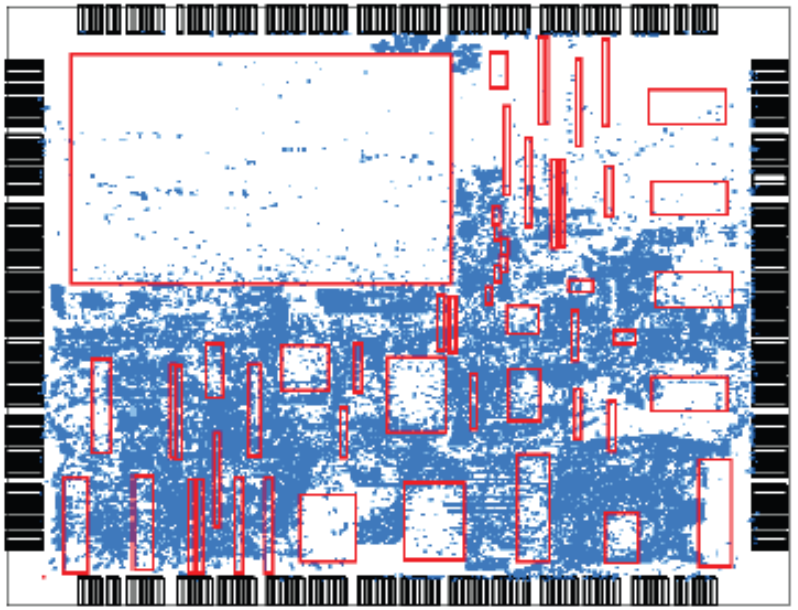
a. Larger regular bins and 3 x 3 Laplacian to smoothing

b. Fix movable macros upon stabilization from upper-bound placement

(2) Regular lookahead legalization for standard cells



Iter=30, HPWL=6.27e7



Iter=50, HPWL=6.22e7

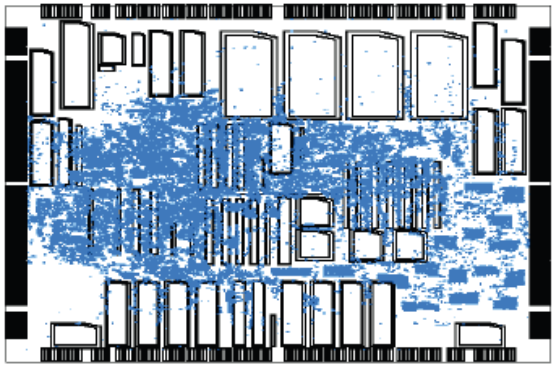
Empirical Validation – ProLR versus ILR

- **Experimental setup**
  - Single threaded runs on a 2.8GHz Intel core i7 Linux station
  - MAPLE is implemented from scratch within an industry infrastructure, including FastPlace-DP for final legalization and detailed placement
- **MAPLE w/ ProLR is compared to MAPLE w/ ILR on ISPD 2005 benchmarks**
  - On bigblue3 and bigblue4, ProLR was 1.5x slower than ILR

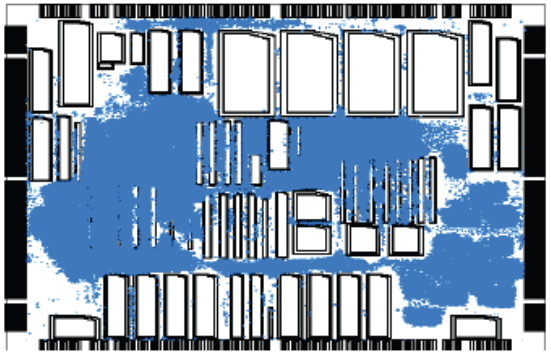
Ckts	MAPLE w/ ILR	MAPLE w/ ProLR	Improv.
ad1	77.41	76.36	1.37%
ad2	89.07	86.95	2.38%
ad3	210.13	209.78	0.17%
ad4	190.07	179.91	5.35%
bb1	95.25	93.74	1.59%
bb2	149.84	144.55	3.53%
bb3	345.20	323.05	6.42%
bb4	792.20	775.71	2.08%
Avg	**1.03×**	**1.00×**	**2.86%**

Empirical Validation – ProLR vs ILR

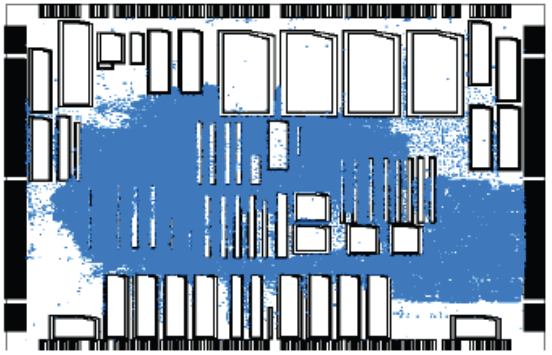
Phase1 (Coarsest) HPWL=6.81e7



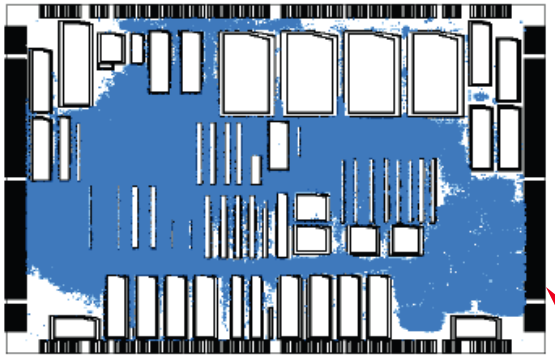
Phase2a (ILR), HPWL=7.99e7



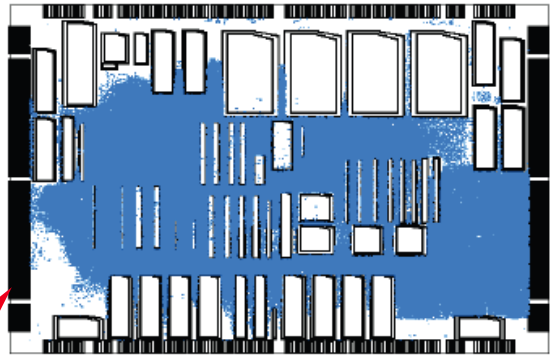
Phase2b (ProLR), HPWL=7.33e7



Phase2b (ILR), HPWL=8.25e7



Phase2b (ProLR), HPWL=7.94e7



Empirical Validation – ISPD 2005

- **MAPLE found placements with the lowest HPWL for seven out of eight circuits**
  - MAPLE improves wirelength by > 2% on average
  - 1.13x, 2.28x faster than mPL6, APlace2, and 2.32x, 6.25x, 7.14x slower than NTUPlace3, FastPlace3, SimPL

Benchmarks	APLACE2 [16]	NTUPLACE3 [10]	FASTPLACE3 [28]	MPL6 [7]	SIMPL [18]	RQL* [29]	MAPLE
ADAPTEC1	78.35	81.82	78.66	77.93	78.58	77.82	**76.36**
ADAPTEC2	95.70	88.79	94.06	92.04	91.24	88.51	**86.95**
ADAPTEC3	218.52	214.83	214.13	214.16	**208.90**	210.96	209.78
ADAPTEC4	209.28	195.93	197.50	193.89	185.39	188.86	**179.91**
BIGBLUE1	100.02	98.41	96.67	96.80	97.54	94.98	**93.74**
BIGBLUE2	153.75	151.55	155.74	152.34	145.28	150.03	**144.55**
BIGBLUE3	411.59	360.66	365.16	344.10	340.24	323.09	**323.05**
BIGBLUE4	871.29	866.43	836.20	829.44	801.35	797.66	**775.71**
Geomean	**1.10×**	**1.07×**	**1.07×**	**1.05×**	**1.03×**	**1.02×**	**1.00×**

Empirical Validation – ISPD 2006

- **MAPLE improves scaled HPWL > 3%**
  - Compared to RQL and NTUPlace3, MAPLE achieves lower overflow penalty on average.

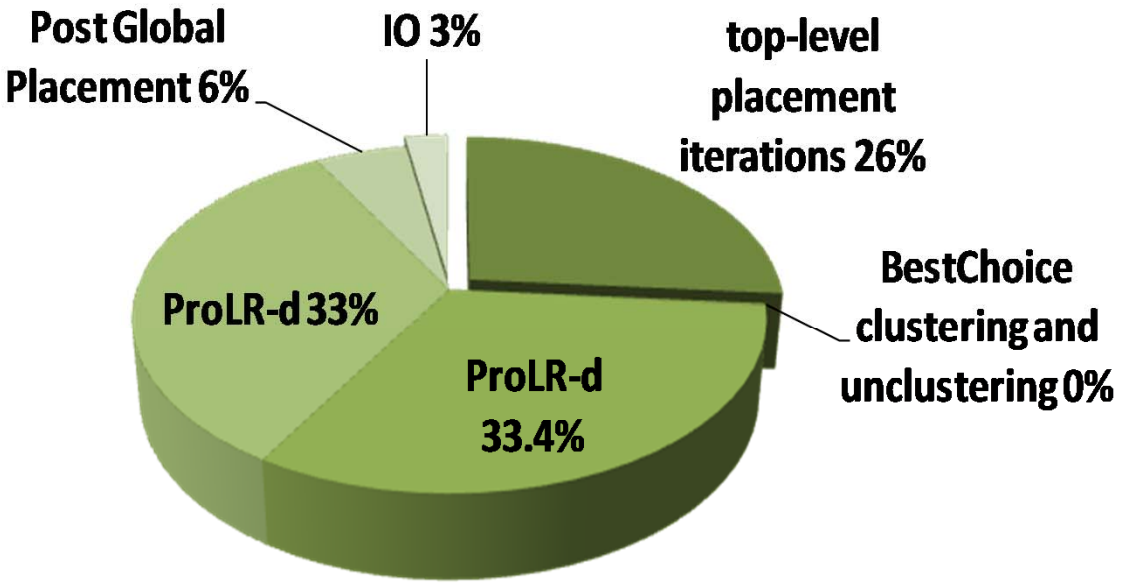
Benchmarks (Υ_{target})	APLACE3* [22]	NTUPLACE3 [10]	FASTPLACE3 [28]	MPL6 [7]	RQL* [29]	MAPLE
ADAPTEC5 (0.5)	520.97 (15.9)	430.73 (12.2)	541.22 (36.5)	431.27 (1.09)	443.28 (9.25)	**407.33 (4.76)**
NEWBLUE1 (0.8)	73.31 (0.14)	**62.39 (0.76)**	76.56 (1.02)	68.08 (0.14)	64.43 (0.34)	69.25 (1.05)
NEWBLUE2 (0.9)	198.24 (0.42)	211.77 (3.21)	240.56 (1.97)	201.85 (1.52)	199.60 (1.45)	**191.66 (1.01)**
NEWBLUE3 (0.8)	273.64 (0.00)	280.19 (0.01)	301.72 (0.78)	284.11 (0.59)	269.33 (0.07)	**268.07 (0.77)**
NEWBLUE4 (0.5)	384.12 (1.74)	302.25 (9.22)	306.07 (7.74)	300.58 (1.63)	308.75 (15.2)	**282.49 (5.86)**
NEWBLUE5 (0.5)	613.86 (12.5)	547.20 (20.82)	633.72 (28.31)	537.14 (1.42)	537.49 (13.6)	**515.04 (4.05)**
NEWBLUE6 (0.8)	522.73 (0.03)	518.25 (6.08)	531.56 (1.26)	522.54 (1.40)	515.69 (4.33)	**494.82 (1.08)**
NEWBLUE7 (0.8)	1098.9 (0.06)	1114.2 (5.19)	1116.7 (1.33)	1084.4 (1.14)	1057.8 (2.57)	**1032.6 (1.70)**
Geomean	**1.13 × (0.32)**	**1.04 × (2.55)**	**1.16 × (3.47)**	**1.06 × (1.22)**	**1.03 × (2.30)**	**1.00 × (1.90)**

Summary

- **New wirelength-driven global placement algorithm – *MAPLE***
  - Employs a strong force-directed placer for the coarsest level
  - Multilevel extensions reinforced by two-tier Progressive Local Refinement (ProLR)
  - Techniques to facilitate graceful transitions between multiple optimizations during global placement

- **MAPLE is implemented and evaluated under an industry framework**
  - Empirical evaluation shows strong results on standard benchmarks
  - Many more applications exist in physical synthesis

Thank you!


Post Global Placement 6%
IO 3%
top-level placement iterations 26%
BestChoice clustering and unclustering 0%
ProLR-d 33%
ProLR-d 33.4%

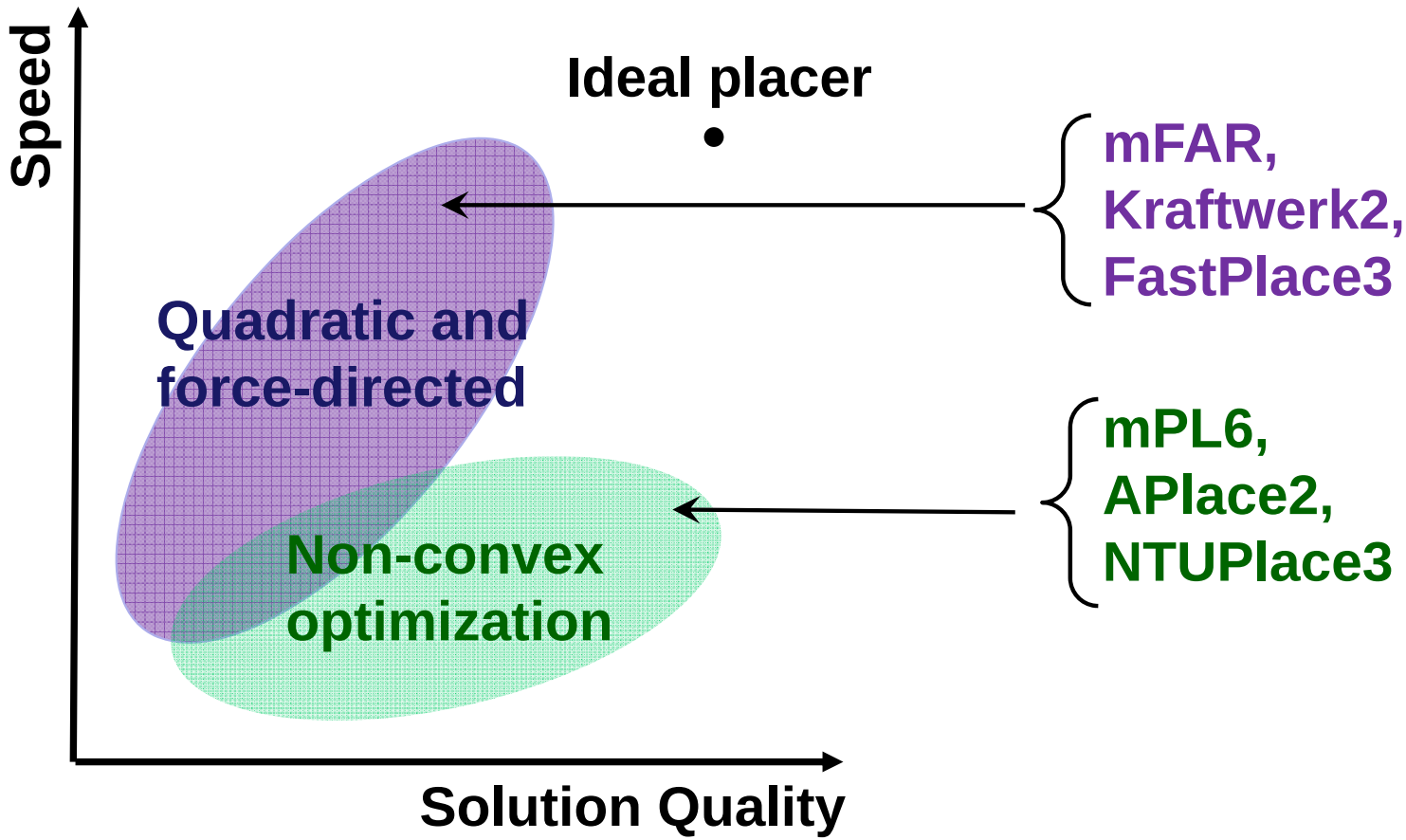
Ckts	AP2	NTU3	MPL6	FP3	SIMPL	MP
AD1	46.29	7.92	21.45	2.36	2.48	17.48
AD2	65.49	7.28	21.87	3.58	3.46	24.30
AD3	144.27	14.98	67.14	7.56	6.43	47.34
AD4	158.30	15.47	57.70	6.69	5.44	44.32
BB1	56.68	12.67	24.56	3.67	3.53	24.31
BB2	110.96	25.18	65.44	6.51	6.36	43.96
BB3	233.70	49.70	88.87	19.85	13.25	94.36
BB4	516.37	109.82	199.74	32.27	29.50	214.86
Avg	2.68×	0.43×	1.13×	0.16×	0.14×	1.00×

Computation of Initial Step θ^0_{step}

- **MAPLE uses a step function that distinguishes different cases**
  - (1) emphasis on wirelength optimization
  - (2) no bias
  - (3) emphasis on spreading

$$\theta^0_{step} = \begin{cases} 0.0250, & \text{if } \Upsilon_{target} - \Upsilon_{design} \geq 0.5 \\ 0.0275, & \text{if } \Upsilon_{target} - \Upsilon_{design} \geq 0.05 \\ 0.0375, & \text{if } \Upsilon_{target} - \Upsilon_{design} < 0.05 \end{cases}$$

Prior Work



- Ideal Placer
  - Fast runtime without sacrificing solution quality
  - Reasonable runtime with superior solution quality