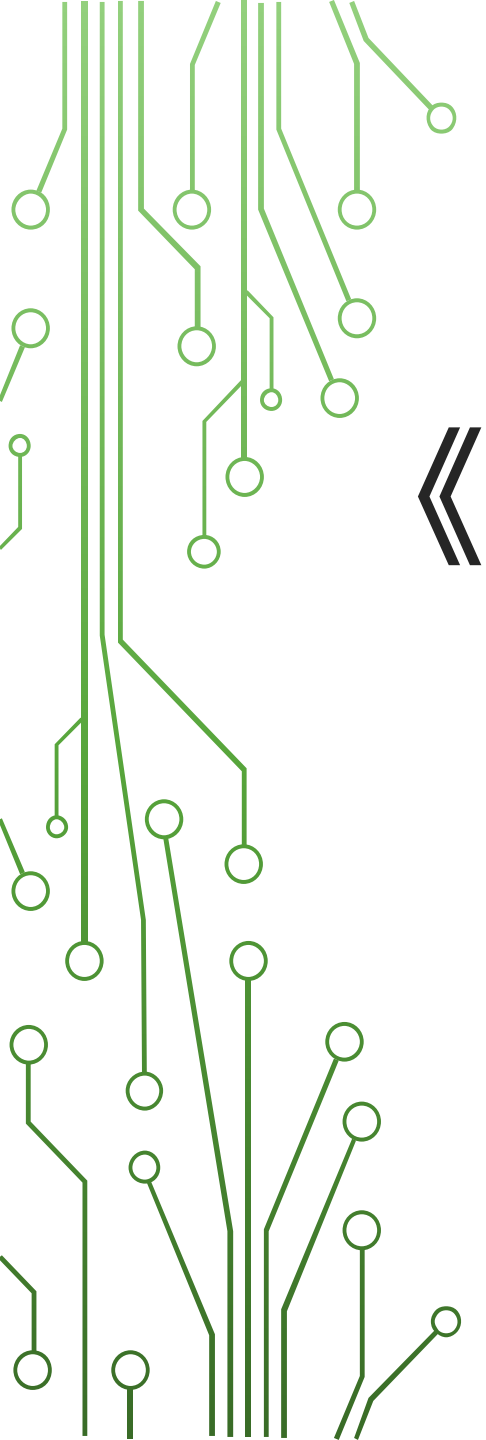


《芯片设计自动化与智能优化》
Placement – Other Topics

Yibo Lin

Peking University

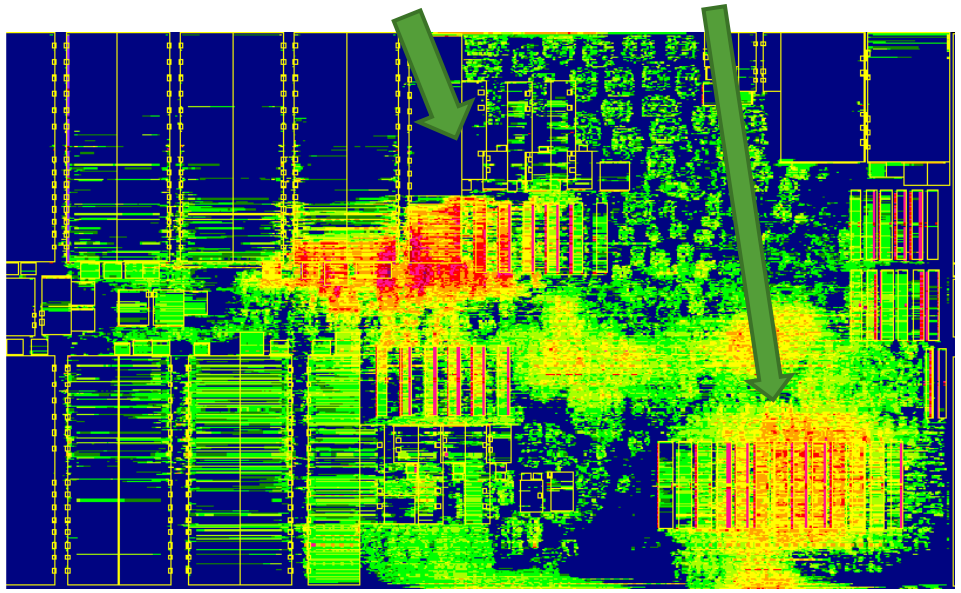
Outline

- What is placement
- History of placement algorithms
- Global placement
  - Quadratic placement: FastPlace & SimPL
  - Nonlinear placement: NTUplace & ePlace
- Legalization
  - Tetris
  - Row-based algorithms: Abacus, DP, LP, MCF
  - Integer linear programming
- Detailed placement
  - Global move & swap
  - Independent set matching
  - Local reordering
  - Row-based algorithms: DP, LP, MCF
- **Other topics**
  - Routability-driven placement
  - Timing-driven placement
  - Macro placement

Routability-driven Placement

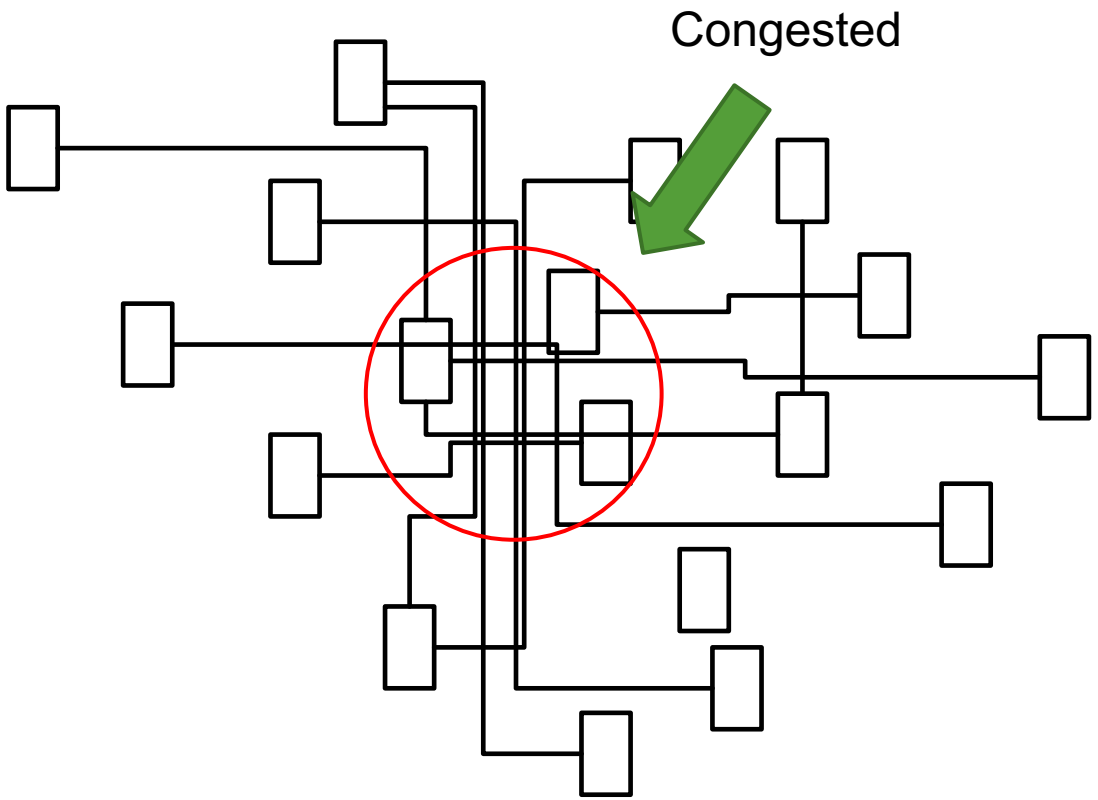
- A placement solution may not be routable

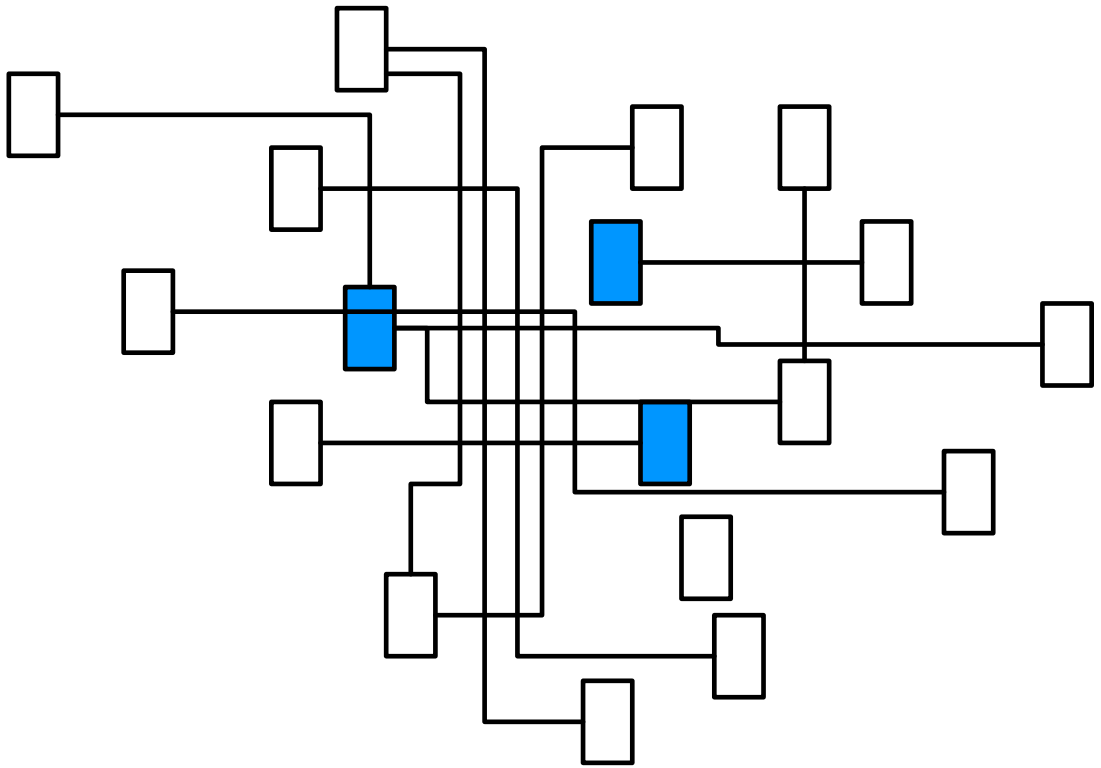
Routing congestion



Routability-driven Placement

- A placement solution may not be routable



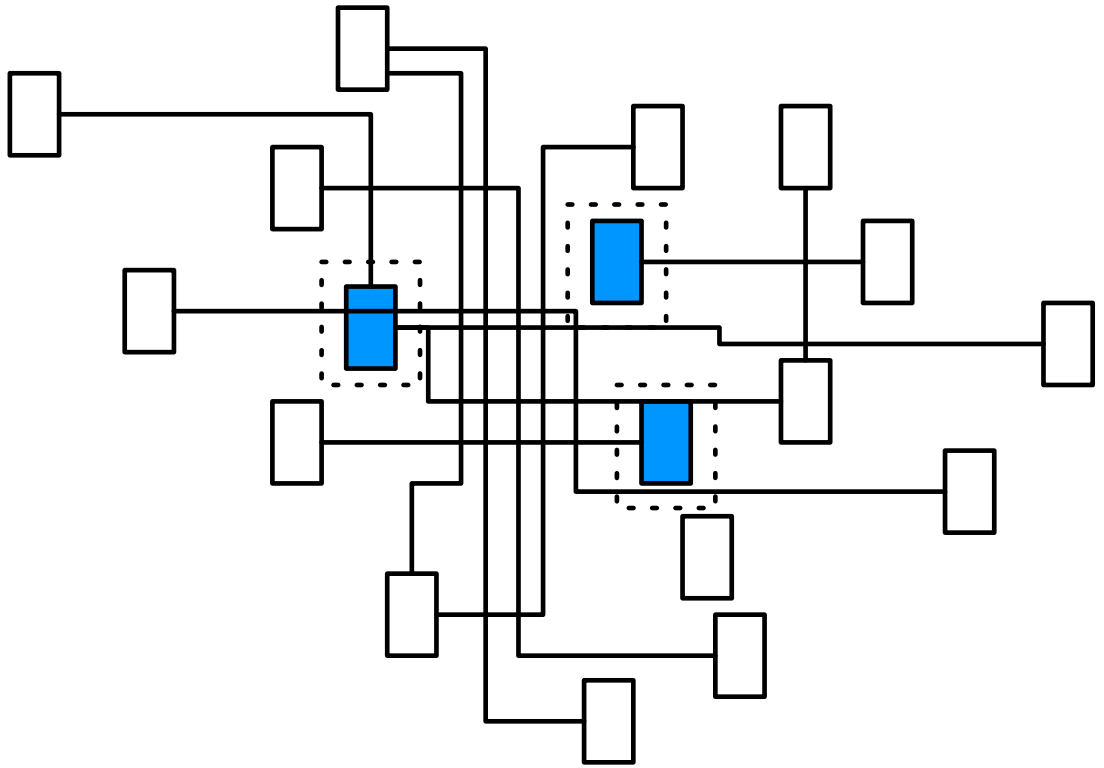


Typical Strategy

- Recall mathematical formulation

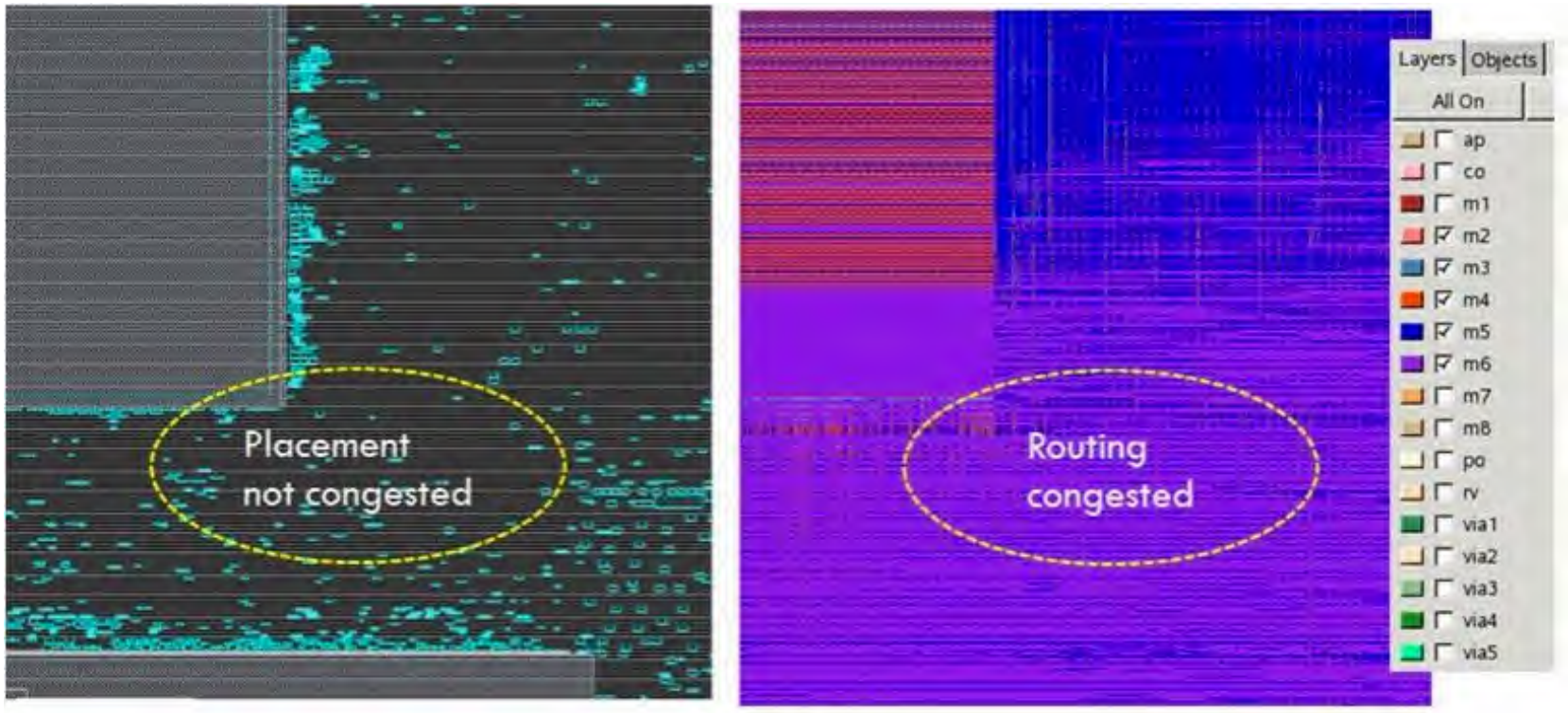
$$\min_{\boldsymbol{x},\boldsymbol{y}} \; WL(\boldsymbol{x},\boldsymbol{y}),$$
$$s.t. \;\; d_b(\boldsymbol{x},\boldsymbol{y}) \le t_d, \forall b \in Bins$$

- Cell inflation
  - Only know congestion when cells are spread
  - Inflate cells at congested regions
  - Restart placement iterations



Routability-driven Placement

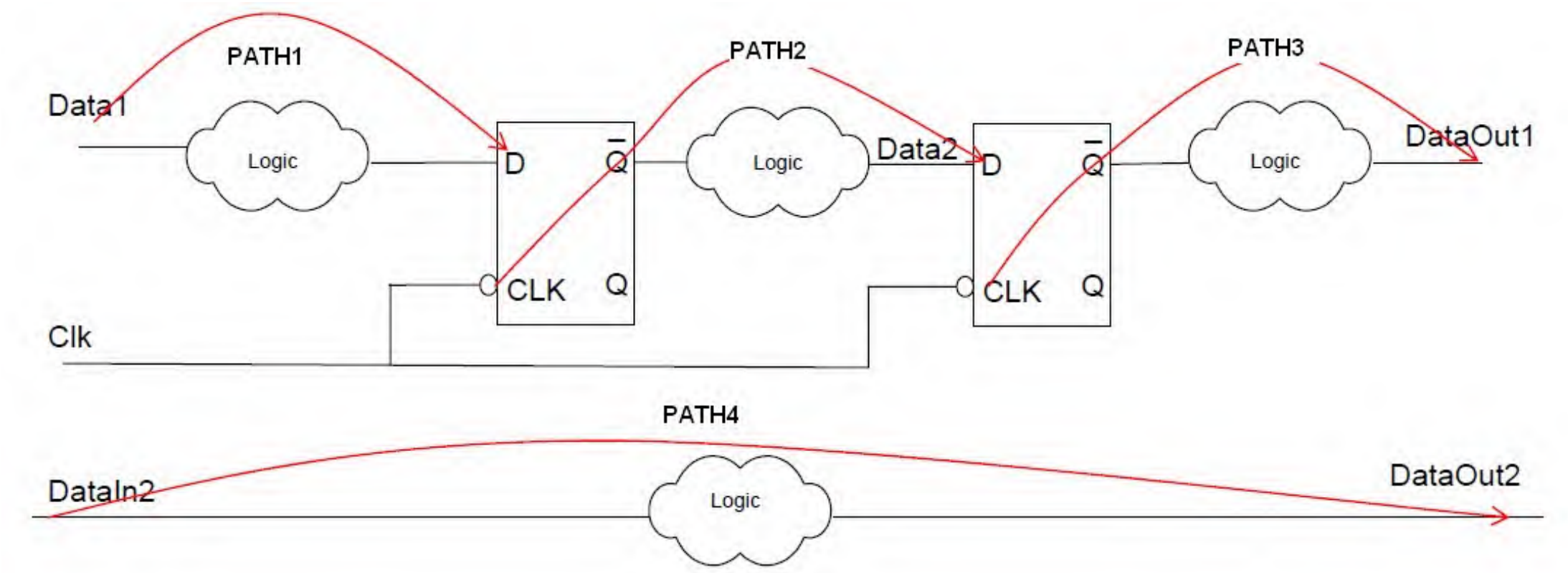
- A non-trivial case with routability issue
- Optimize routability in an elegant way
  - Still an open problem



Curtesy semiwiki

Timing-driven Placement

- Timing determines how fast a circuit can run

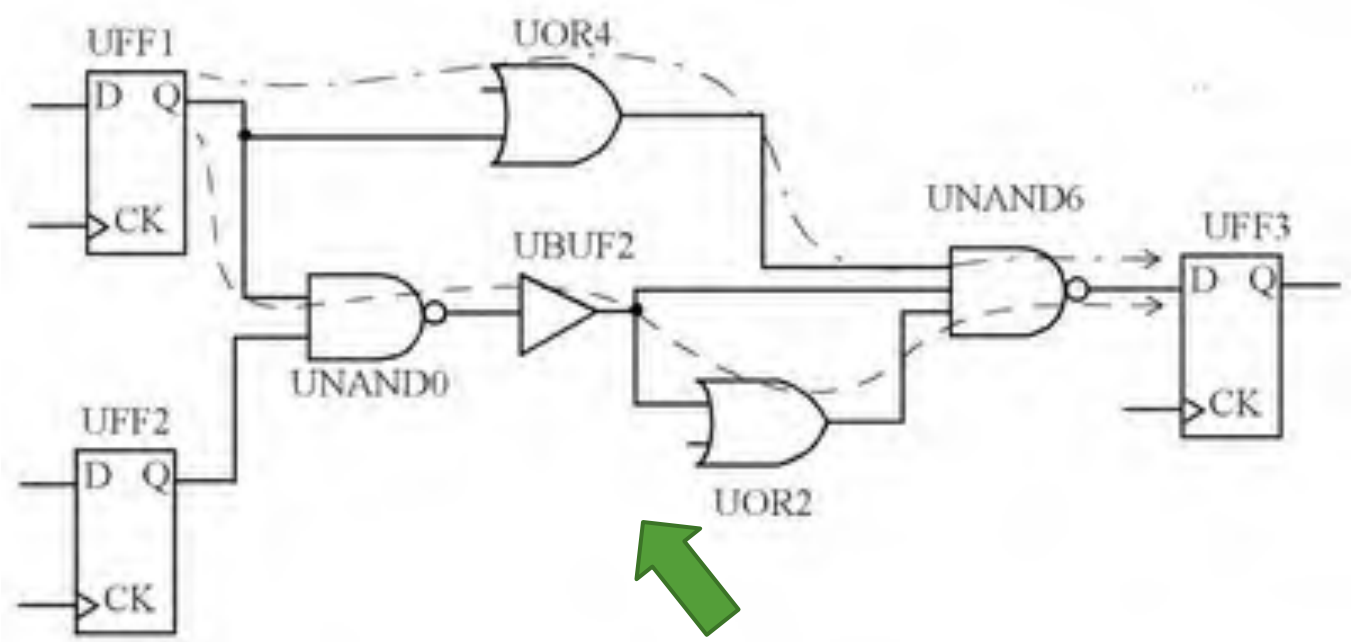


Peiyu Liao, Siting Liu, Zhitang Chen, Wenlong Lv, Yibo Lin and Bei Yu, "DREAMPlace 4.0: Timing-driven Global Placement with Momentum-based Net Weighting", DATE 2022.
Zizheng Guo and Yibo Lin, "Differentiable-Timing-Driven Global Placement", DAC 2022.

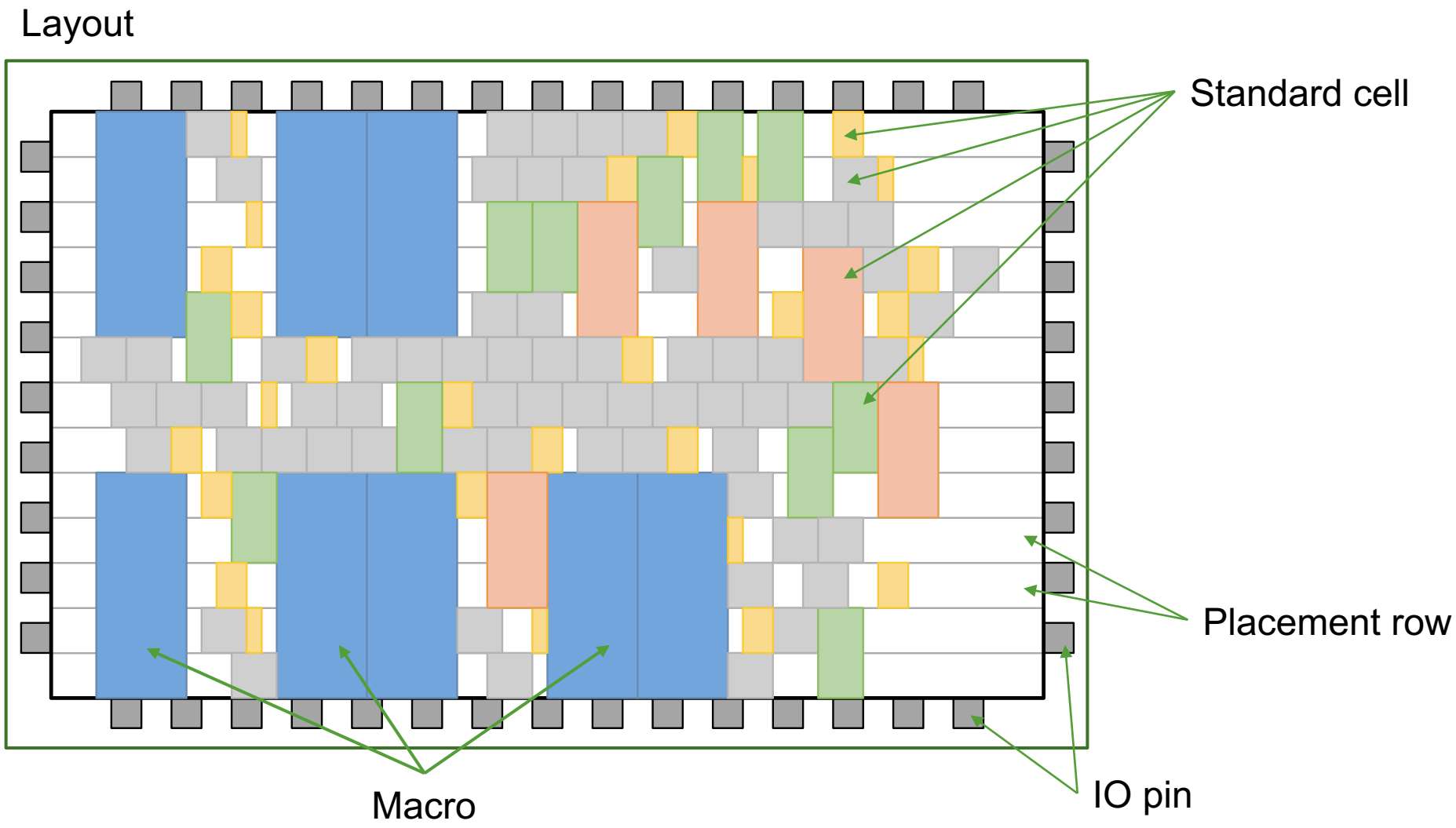
Timing-driven Placement

- Wirelength is just a “first-order approximation” to timing
- How to minimize delay along critical paths
  - Net weighting is a useful strategy
  - Also an open problem. Better approach?

$$\min_{\boldsymbol{x},\boldsymbol{y}} \; WL(\boldsymbol{x},\boldsymbol{y}),$$
$$s.t. \;\; d_b(\boldsymbol{x},\boldsymbol{y}) \le t_d, \forall b \in Bins$$



Macro Placement



Macro Placement – Problem Formulation

- Also known as Mixed-Size Placement
- Input
  - Blocks (standard cells and macros) B_1, ... , B_n
  - Shapes and Pin Positions for each block B_i
  - Nets N_1, ... , N_m
  - Movable blocks include standard cells and **macros**
- Output
  - Coordinates (x_i , y_i) for block B_i.
  - No overlaps between blocks within a fixed layout area
- Objective
  - The total wirelength is minimized
- Other objectives: timing, routability, clock, buffering

Macro Placement v.s. Standard Cell Placement

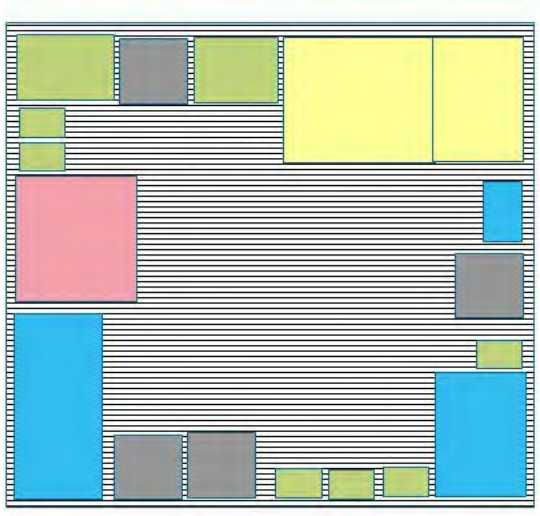
- Minimize wirelength and routability
- Need legalization
- Large
- Heterogeneous sizes
- May not align to rows
- 10 to 10K
- Not easy to legalize

- Minimize wirelength and routability
- Need legalization
- Small
- Regular sizes
- Align to rows
- 100K to 1M
- Many strategies for legalization

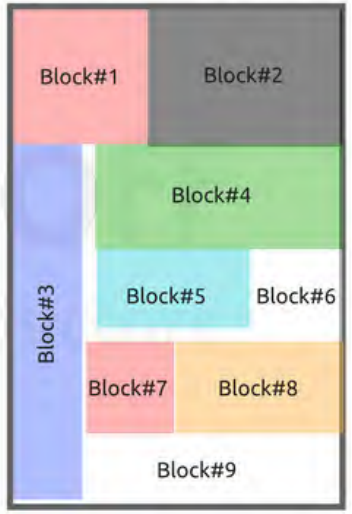
Macro locations can be pre-determined before standard cell placement or determined together through mixed-size placement

Macro Legalization vs Floorplanning

- Heterogeneous sizes
- Minimize displacement
- Blocks are not packed
- Fixed outline

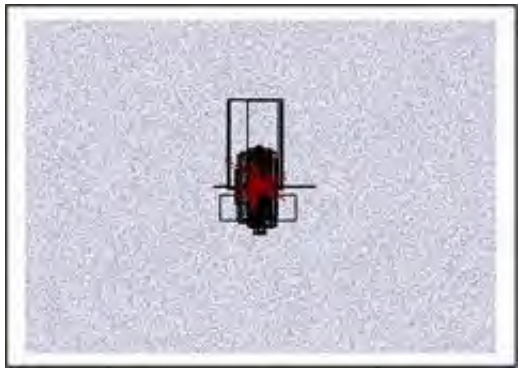


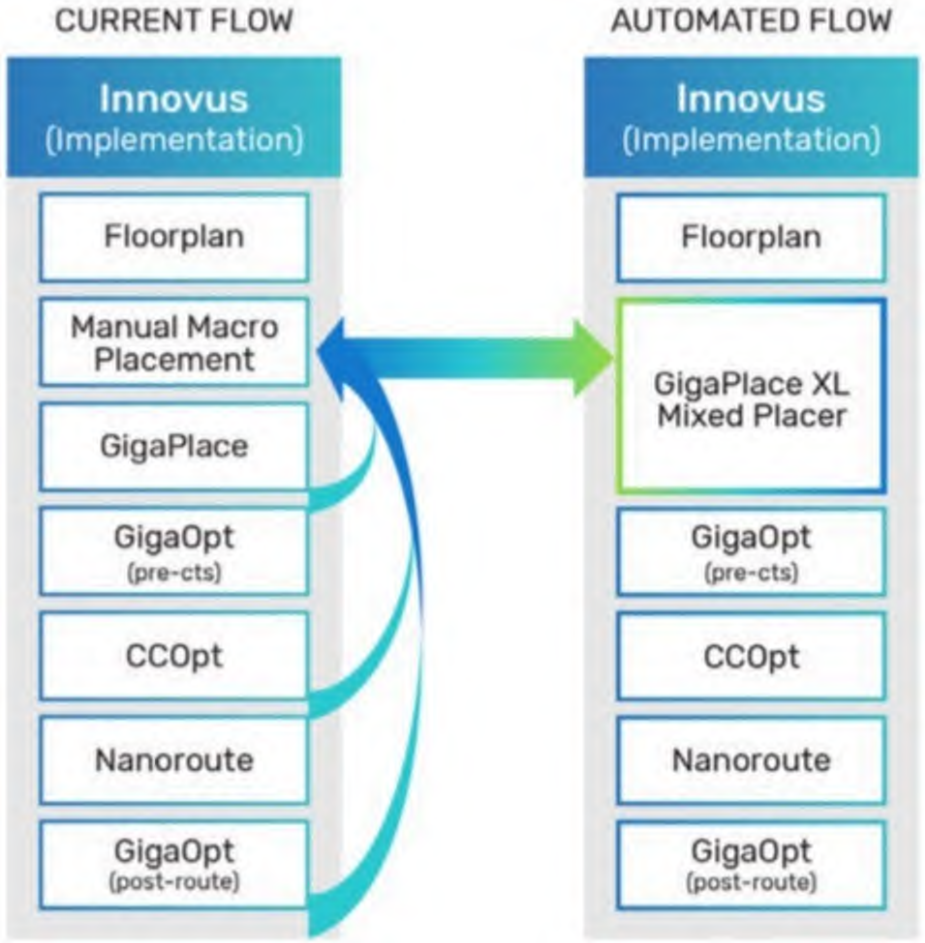
- Heterogeneous sizes
- Minimize area and wirelength
- Blocks are packed
- Outline may not be fixed



Cadence Announced a Mixed-Size Placer in 2020

- Locations of macros have huge impacts to wirelength and routability
- Previous work try simulated annealing to finetune macro placement/legalization
  - [ePlace-MS, TCAD'2015]





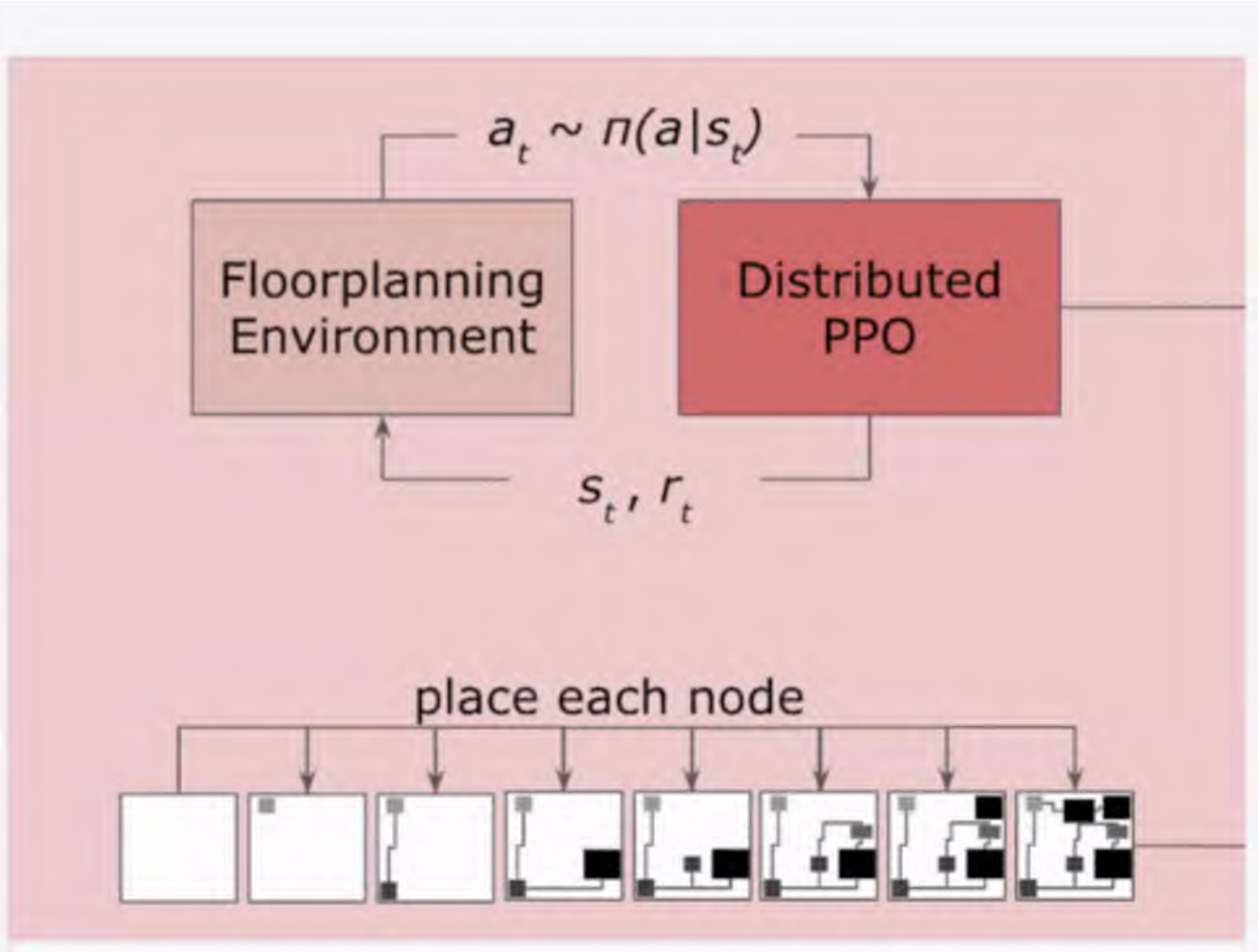
Curtesy Cadence
http://vlsi-cuda.ucsd.edu/~ljw/ePlace-MS/

Google's Reinforcement Learning Approach

State: Graph embedding of chip netlist, embedding of the current node, and the canvas.

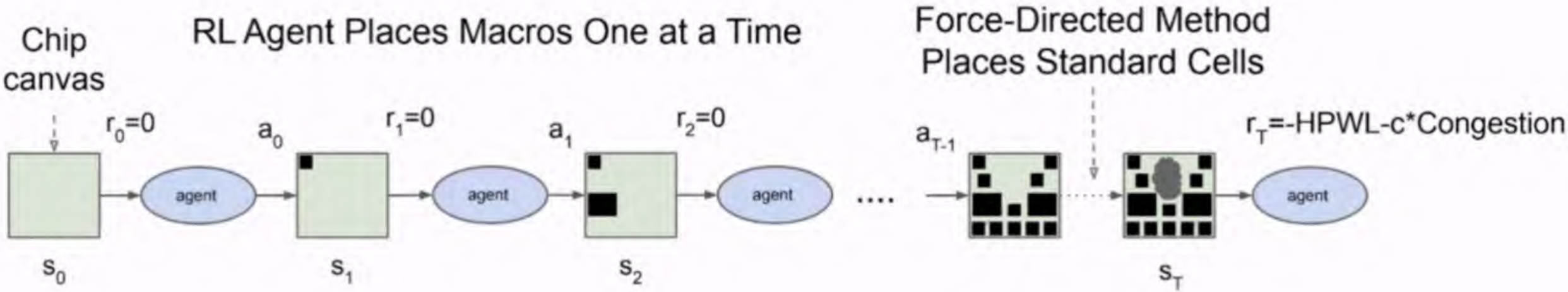
Action: Placing the current node onto a grid cell.

Reward: A weighted average of total wirelength, density, and congestion



Curtesy Google Brain

Place One Macro at a Time



Reward corresponding to placement p of netlist (graph) g

$$J(\theta, G) = \frac{1}{K} \sum_{g \sim G} E_{g, p \sim \pi_\theta}[R_{p,g}]$$

Set of training graphs G

K is size of training set

RL policy parameterized by theta

$$R_{p,g} = -Wirelength(p,g) - \lambda\, Congestion(p,g)$$
$$s.t.\ density(p,g) \leq density_{max}$$

#blocks ~20
Need finetune for each design
~6h per design

No longer need to tune for different constraints: found a general method

Curtesy Google Brain

Summary

- Routability-driven placement
- Timing-driven placement
- Macro placement

- Many open problems for good strategies