《芯片设计自动化与智能优化》Placement

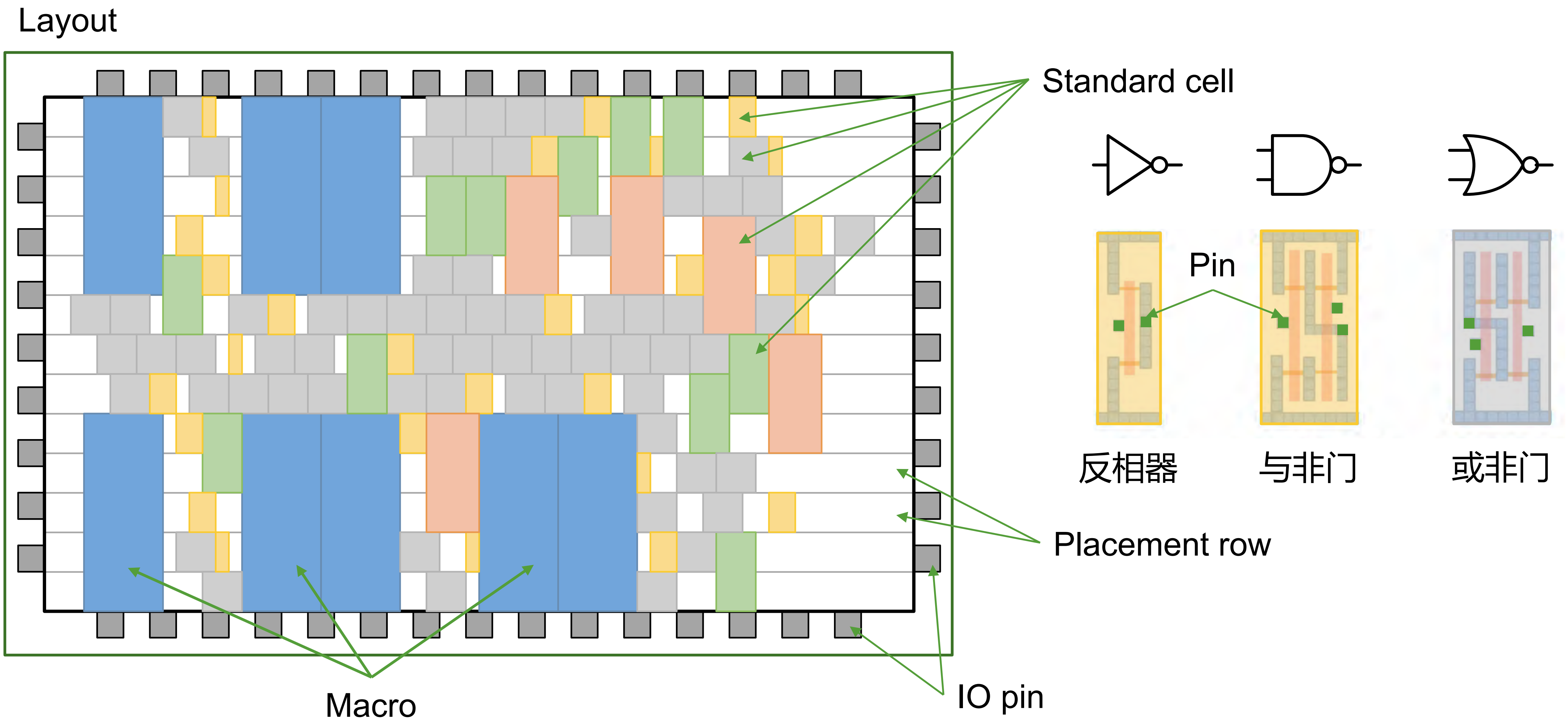
Yibo Lin

Peking University

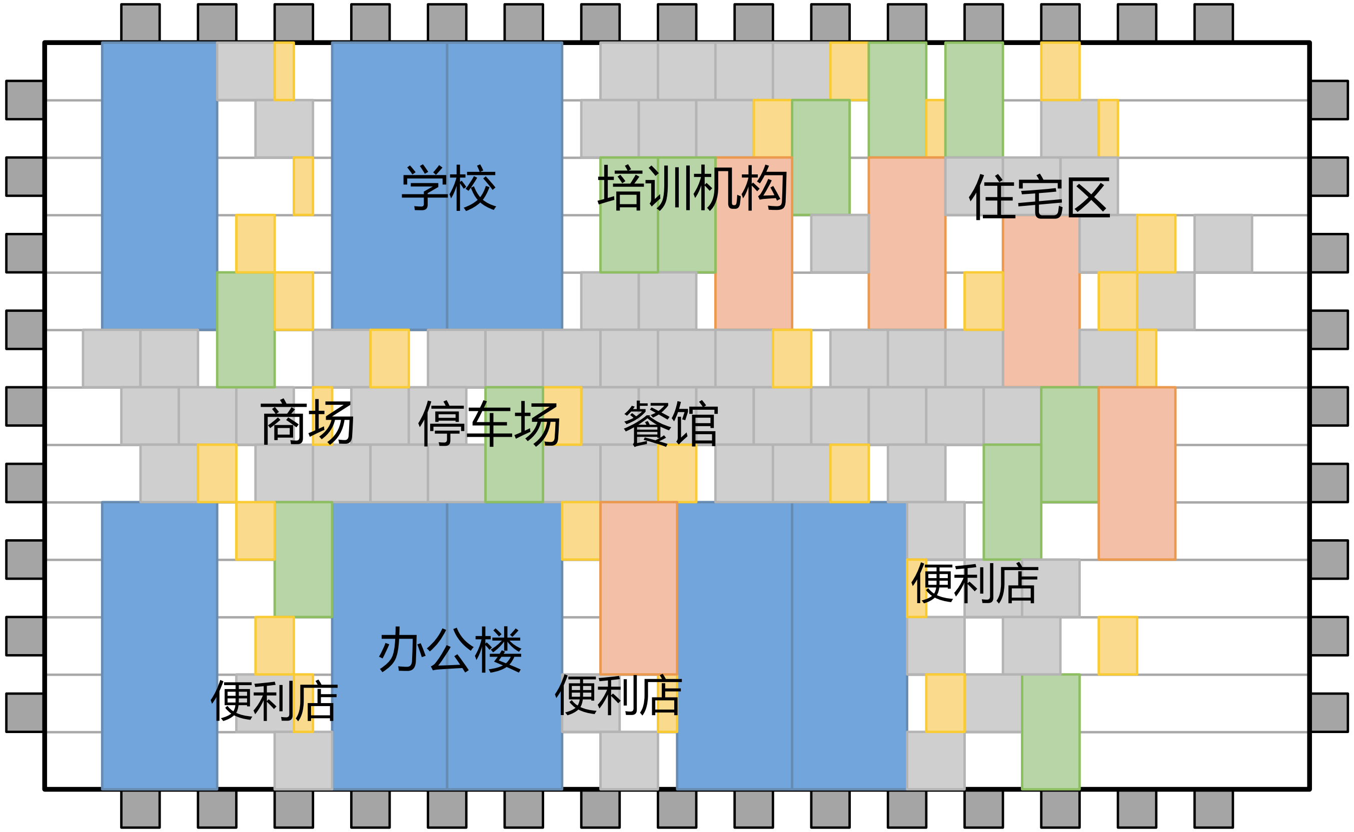
Outline

- What is placement
- History of placement algorithms
- Global placement
  - Simulated annealing: DRAGON
  - Partitioning: CAPO
  - 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

What is Placement



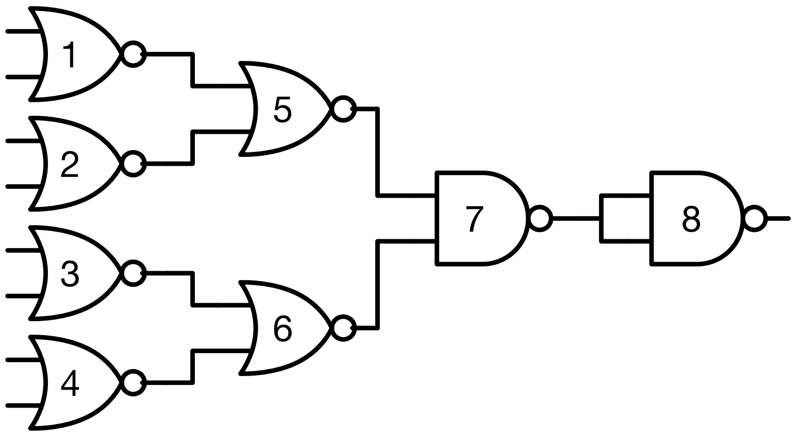
Analog to Urban Planning

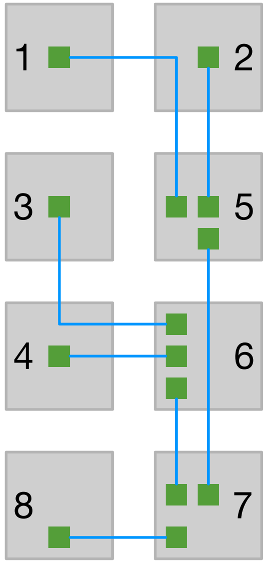


好的城市规划应使功能关联建筑之间的距离尽可能小

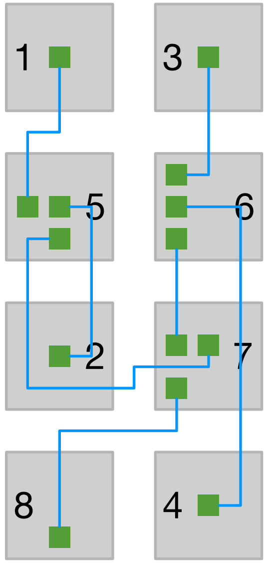
Metrics for Placement

- Wirelength: length of physical wires connecting cells





Wirelength=100



Wirelength=150

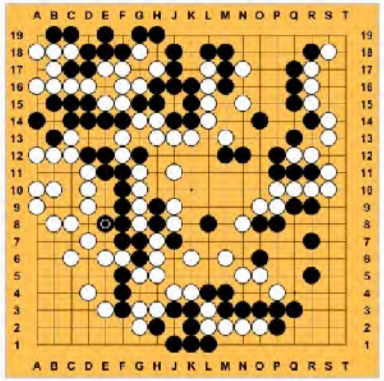
好的布局应使相连Cell之间的距离尽可能小

Problem Formulation for Placement

- Input
  - Blocks (standard cells and macros) $B_1, \dots, B_n$
  - Shapes and Pin Positions for each block B_i
  - Nets $N_1, \dots, N_m$
- 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

How Difficult Placement is

#states: ~10^{123}



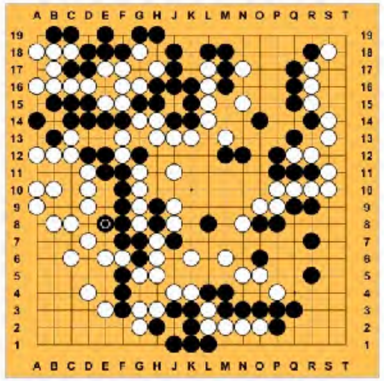
#states: ~10^{360}

Google AlphaGo
Train **40 days** using **176 GPUs**

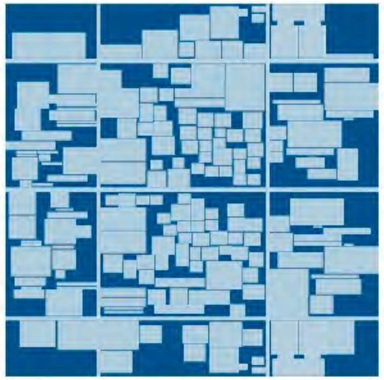
How Difficult Placement is

- Huge problem sizes : tens of millions of cells
- Huge solution space : larger than $1K \times 1K$ grids in a layout

#states: $\sim 10^{123}$



#states: $\sim 10^{360}$



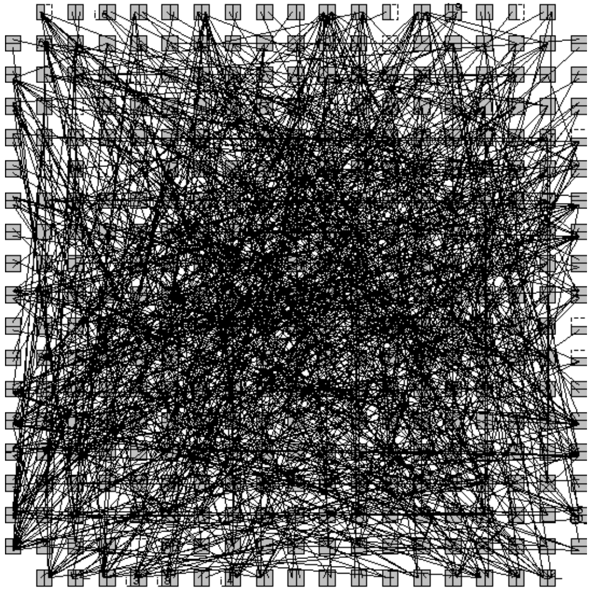
#states: $> 10^{100,000}$

Google AlphaGo
Train **40 days** using **176 GPUs**

Good Placement vs Bad Placement

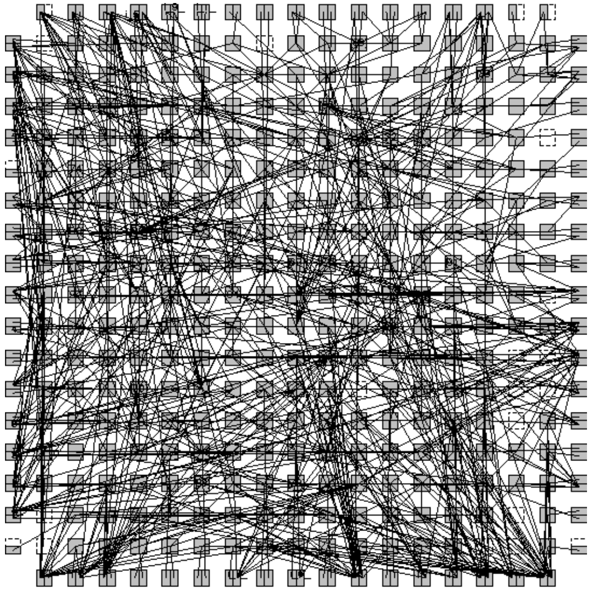
- **230** cells in FPGA (design *e64* in the MCNC benchmark suite)

Random Initial	Final Solution	Routing Solution



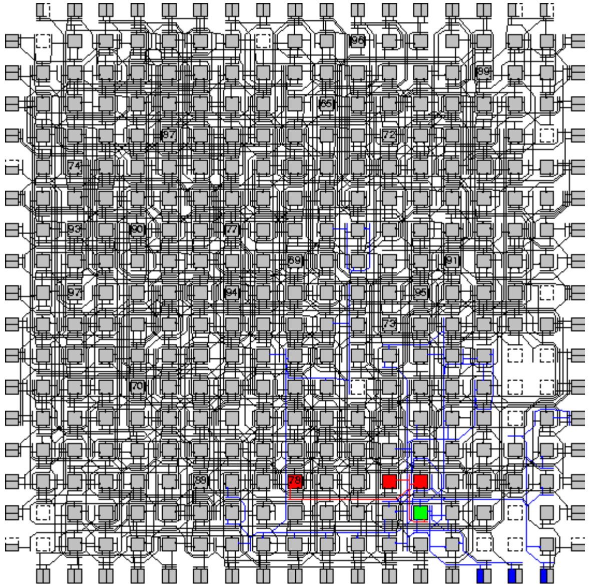
Initial Placement. Cost: 74.5562. Channel Factor: 100

WL = 5.47e+4



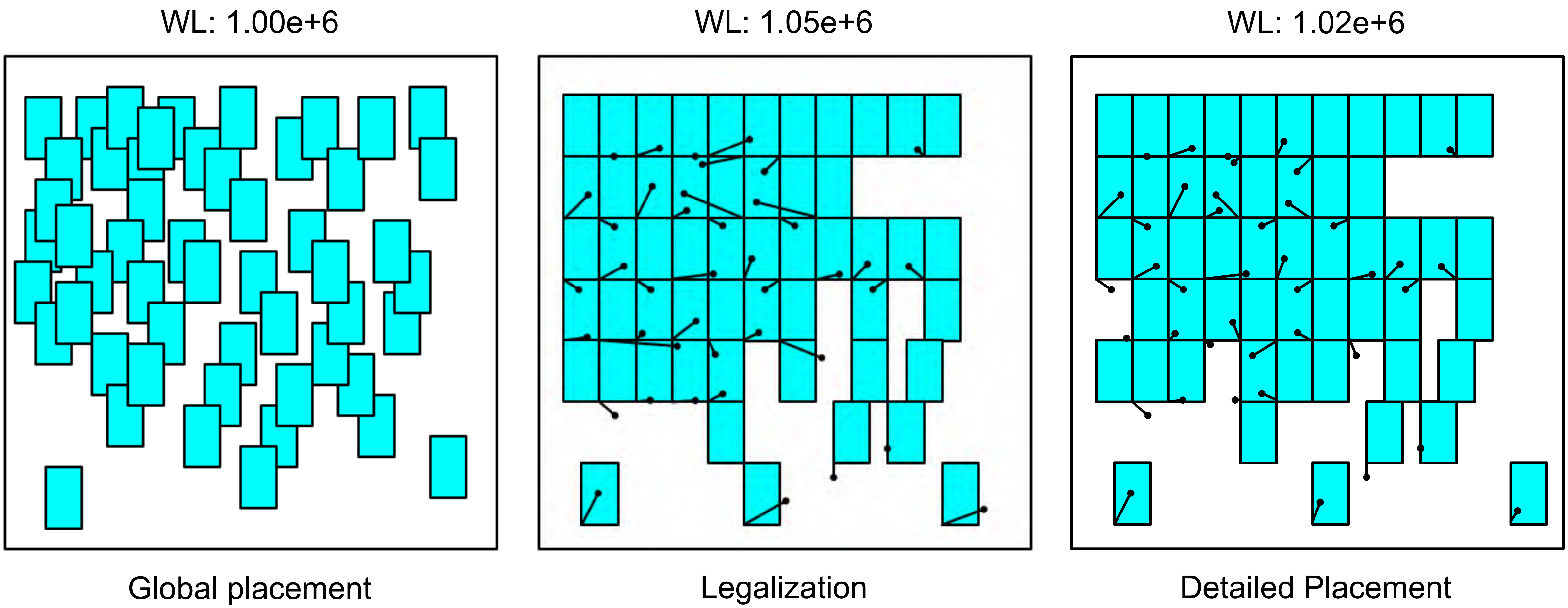
Final Placement. Cost: 28.5384. Channel Factor: 100

WL = 6.73e+3



Routing succeeded with a channel width factor of 7.

Typical Placement Flow



The History of Placement Algorithms

<1970-1980s	1980s-1990s
Partitioning	Simulated Annealing

<1970-1980s	1980s-1990s
Breuer	Timberwolf VPR
Dunlop & Kernighan	Dragon
Quadratic Assignment	
Cadence QPlace	

The History of Placement Algorithms

<1970-1980s	**1980s-1990s**	**1990s-2010s**			**>2010s**	
Partitioning	Simulated Annealing	Min-Cut (Multi-level)	Analytic		Analytic	
			Quadratic	Nonlinear	Quadratic	Nonlinear
Breuer	**Timberwolf** VPR	FengShui	GORDIAN	APlace	POLAR	ePlace RePlAce
Dunlop & Kernighan	**Dragon**	**Capo**	BonnPlace	Naylor Synopsis	SimPL ComPLx	DREAMPlace
Quadratic Assignment		Capo +Rooster	mFar	NTUplace	MAPLE	
Cadence QPlace			Kraftwerk	mPL6		
			FastPlace			
			Warp3			

Outline

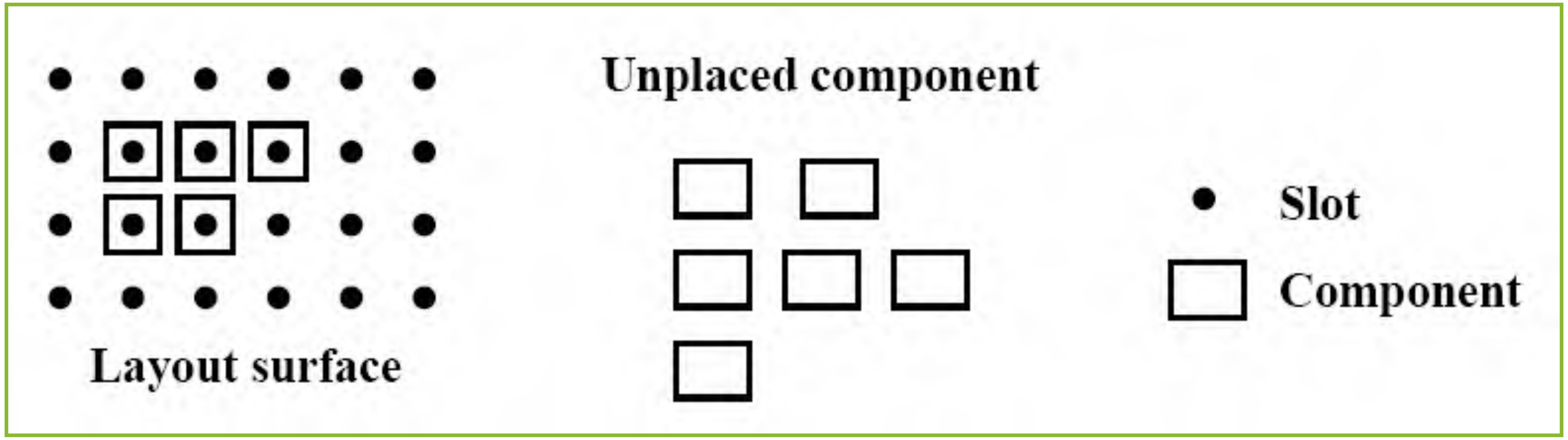
- What is placement
- History of placement algorithms
- **Global placement**
  - Simulated annealing: DRAGON
  - Partitioning: CAPO
  - Quadratic placement: FastPlace & SimPL
  - Nonlinear placement: NTUplace & ePlace
- Legalization
  - Tetris
  - Row-based algorithms: Abacus, DP, LP, MCF
  - Integer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

Simulated Annealing

- Timberwolf package [JSSC-85, DAC-86]
  - Sechen, Carl, and Alberto Sangiovanni-Vincentelli. "The TimberWolf placement and routing package." *IEEE Journal of Solid-State Circuits* 20.2 (1985): 510-522.
  - Sechen, Carl, and Alberto Sangiovanni-Vincentelli. "TimberWolf3. 2: A new standard cell placement and global routing package." *23rd ACM/IEEE Design Automation Conference*. IEEE, 1986.
- Dragon [ICCAD-00]
  - Yang, Xiaojian, and Majid Sarrafzadeh. "Dragon2000: Standard-cell placement tool for large industry circuits." *IEEE/ACM International Conference on Computer Aided Design. ICCAD-2000. IEEE/ACM Digest of Technical Papers (Cat. No. 00CH37140)*. IEEE, 2000.

A Down-to-the-Earth Method

- Select unplaced components and place them in slots
- SELECT: choose the unplaced component that is most strongly connected to all (or any single) of the placed component
- PLACE: place the selected component at a slot such that a certain "cost" of the partial placement is minimized
- Simple and fast: ideal for initial placement

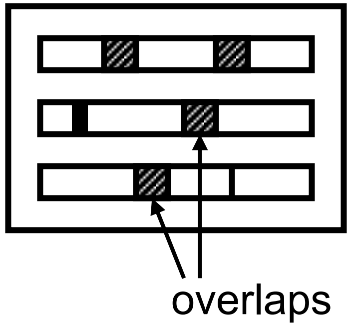


TimberWolf

- Stage 1
  - Modules are moved between different rows as well as within the same row
  - Module overlaps are allowed
  - When the temperature is reduced below a certain value, stage 2 begins
- Stage 2
  - Remove overlaps
  - Annealing process continues, but only interchanges adjacent modules within the same row

Solution Space

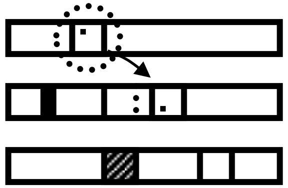
- All possible arrangements of modules into rows possibly with overlaps



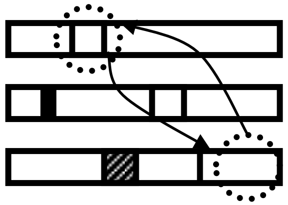
Neighboring Solutions

Three types of moves:

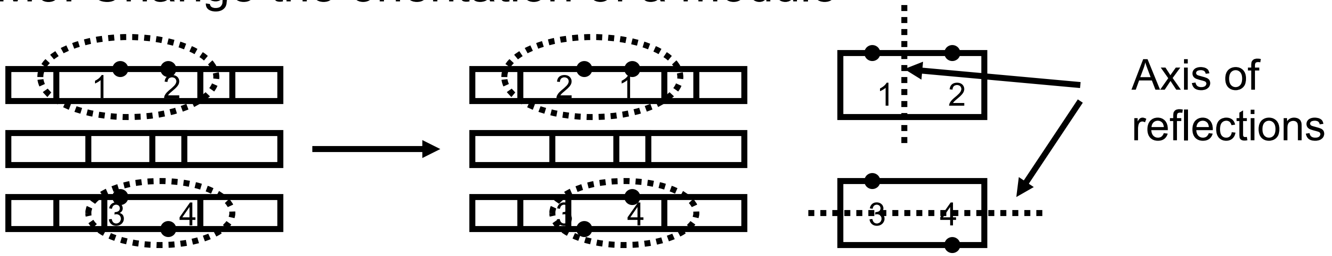
M1: Displace a module to a new location



M2: Interchange two modules



M3: Change the orientation of a module



Move Selection

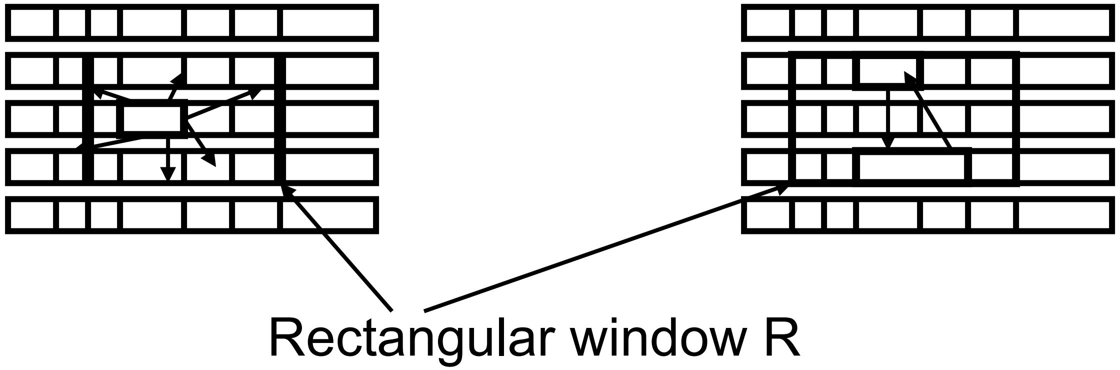
- Timber wolf first try to select a move betwee M1 and M2
  - Prob(M1)=4/5
  - Prob(M2)=1/5

M1: Displacement
M2: Interchange
M3: Reflection

- If a move of type M1 is chosen (for certain module) and it is rejected, then a move of type M3 (for the same module) will be chosen with probability 1/10
- Restriction on:
  - How far a module can be displaced
  - What pairs of modules can be interchanged

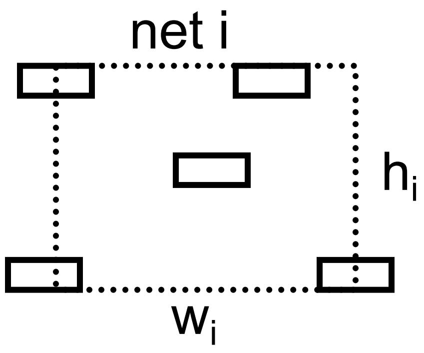
Move Restriction

- Range Limiter
  - At the beginning, R is very large, big enough to contain the whole chip
  - Window size shrinks slowly as the temperature decreases. In fact, height and width of $R \propto \log(T)$
  - Stage 2 begins when window size are so small that no inter-row modules interchanges are possible



Cost Function

$\Psi = C_1 + C_2 + C_3$



$C_1 : \sum_i (\alpha_i w_i + \beta_i h_i)$

α_i, β_i are horizontal and vertical weights, respectively

$\alpha_i = 1$, $\beta_i = 1 \Rightarrow$ 1/2 •perimeter of bounding box

- **Critical nets: Increase both α_i and β_i**
- **Preferred metal layer routing: if vertical wirings are "cheaper" than horizontal wirings, we can use smaller vertical weights, i.e. $\beta_i < \alpha_i$**

Cost Function (Cont'd)

C_2: Penalty function for module overlaps

O(i,j) = amount of overlaps in the X-dimension between modules i and j

$$C_2 = \sum_{i \neq j} \left(O(i,j) + \alpha \right)^2$$

α — offset parameter to ensure $C_2 \rightarrow 0$ when $T \rightarrow 0$

C_3: Penalty function that controls the row lengths

Desired row length = d(r)

l(r) = sum of the widths of the modules in row r

$$C_3 = \sum_r \beta \left| l(r) - d(r) \right|$$

Annealing Schedule

- **$T_k = r(k) \cdot T_{k-1}$ k= 1, 2, 3, ….**

r(k) increase from 0.8 to max value 0.94 and then decrease to 0.1

- **At each temperature, a total number of K•n attempts is made**

n= number of modules

K= user specified constant

Dragon2000

- Simulated annealing based
  - 1.9x faster than iTools 1.4.0 (commerical version of TimberWolf)
  - Comparable wirelength to iTools (i.e., very good)
  - Performs better for larger circuits
  - Still very slow compared with than other approaches
  - Also shown to have good routability
- Top-down hierarchical approach
  - hMetis to recursively quadrisect into 4^h bins at level h
  - Swapping of bins at each level by SA to minimize WL
  - Terminates when each bin contains < 7 cells
  - Then swap single cells locally to further minimize WL
- Detailed placement is done by greedy algorithm

Yang, Xiaojian, and Majid Sarrafzadeh. "Dragon2000: Standard-cell placement tool for large industry circuits." IEEE/ACM ICCAD 2000.

Partition based Methods

- Partitioning methods (already covered in previous lectures)
  - FM
  - Multilevel techniques, e.g., hMetis
- Two academic open source placement tools
  - Capo (UCLA/UCSD/Michigan): multilevel FM
  - Feng-shui (SUNY Binghamton): use hMetis
- Pros and cons
  - Fast
  - Not stable

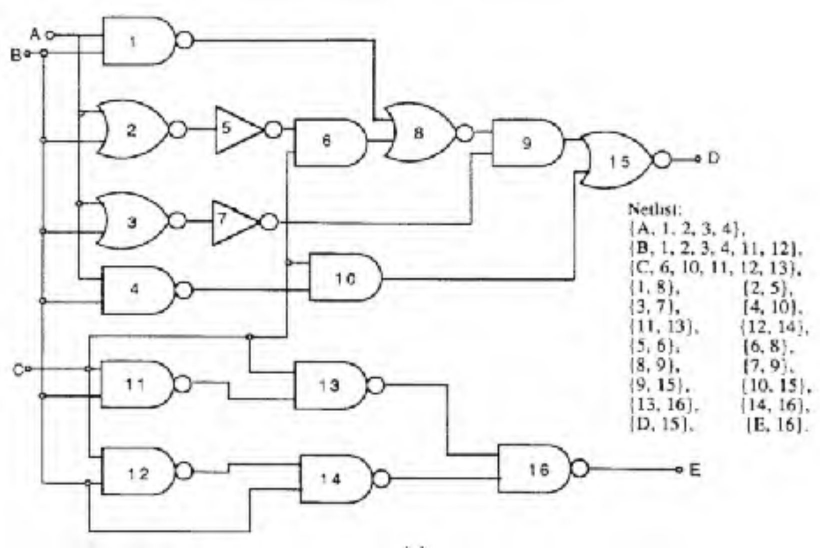
Caldwell, Andrew E., Andrew B. Kahng, and Igor L. Markov. "Can recursive bisection alone produce routable placements?." Proceedings of DAC 2000.
Agnihotri, Ameya R., Satoshi Ono, and Patrick H. Madden. "Recursive bisection placement: Feng Shui 5.0 implementation details." Proceedings of ISPD 2005.

Partitioning-based Approach

- Try to group closely connected modules together.
- Repetitively divide a circuit into sub-circuits such that the cut value is minimized.
- Also, the placement region is partitioned (by cutlines) accordingly.
- Each sub-circuit is assigned to one partition of the placement region.

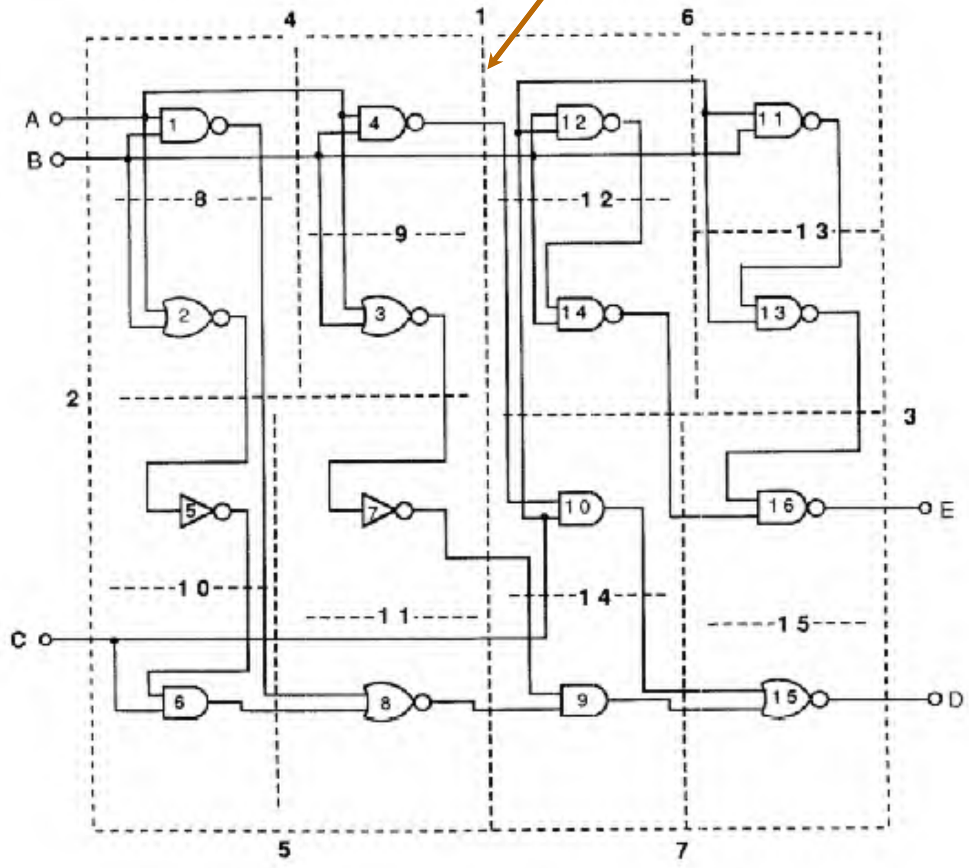
Note: Also called min-cut placement approach.

An Example



Circuit

Cutline



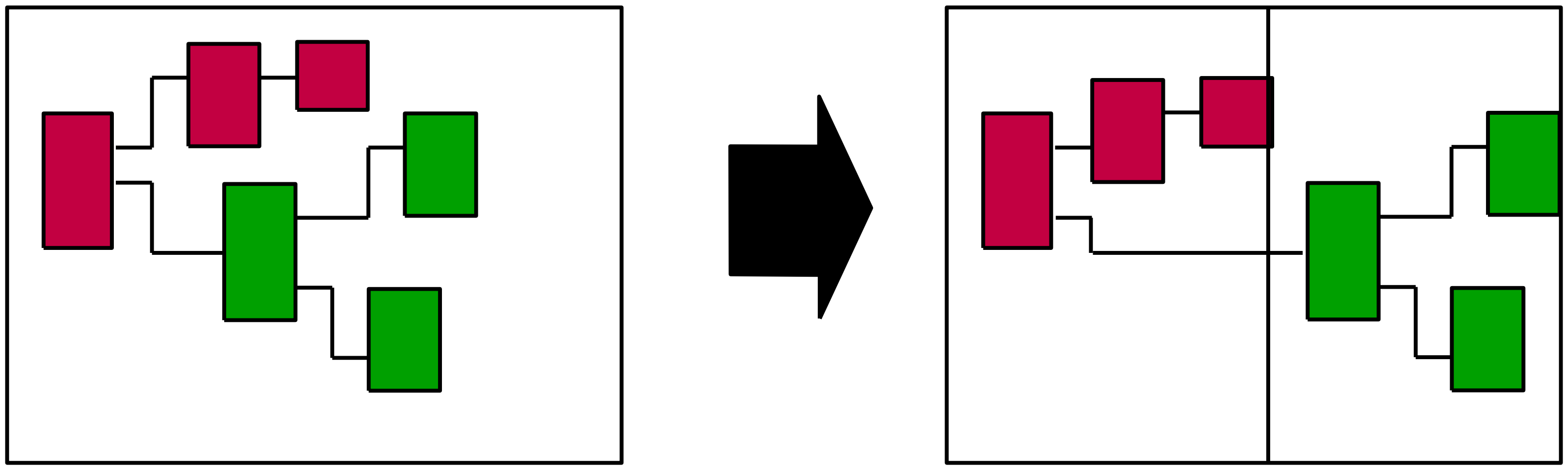
Placement

Variations

- There are many variations in the partitioning-based approach. They are different in:
  - The objective function used.
  - The partitioning algorithm used.
  - The selection of cutlines.

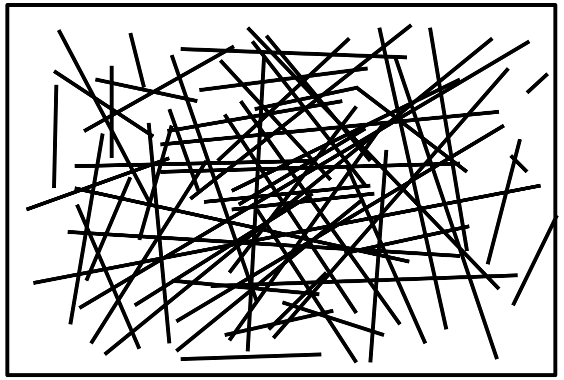
Partitioning

- Objective:
  - Given a set of interconnected blocks, produce two sets that are of equal size, and such that the number of nets connecting the two sets is minimized.

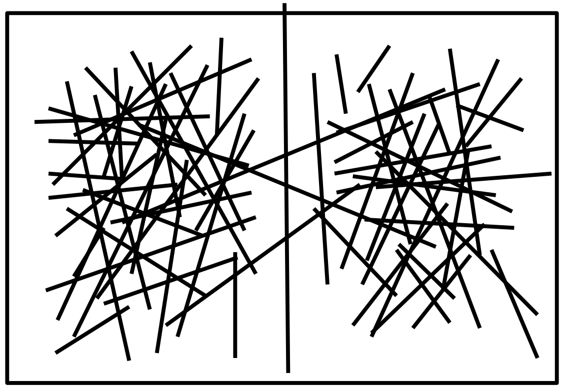


FM Partitioning

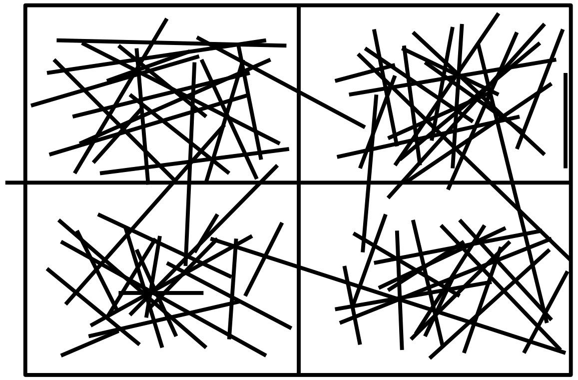
```
list_of_sets = entire_chip;
while(any_set_has_2_or_more_objects(list_of_sets))
{
      for_each_set_in(list_of_sets)
      {
            partition_it();
      }
      /* each time through this loop the number of   */
      /* sets in the list doubles.                   */
}
```



Initial Random Placement



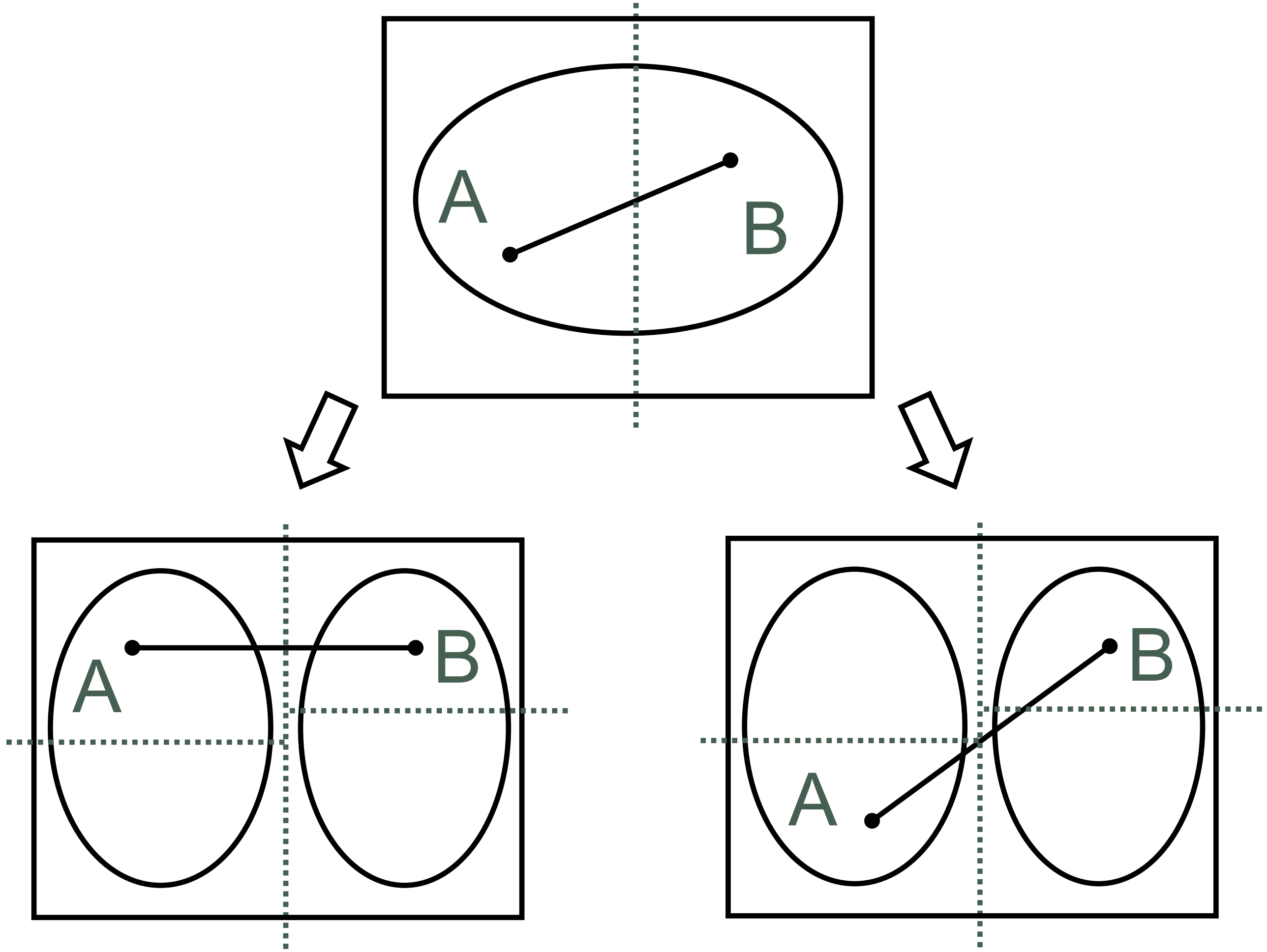
After Cut 1



After Cut 2

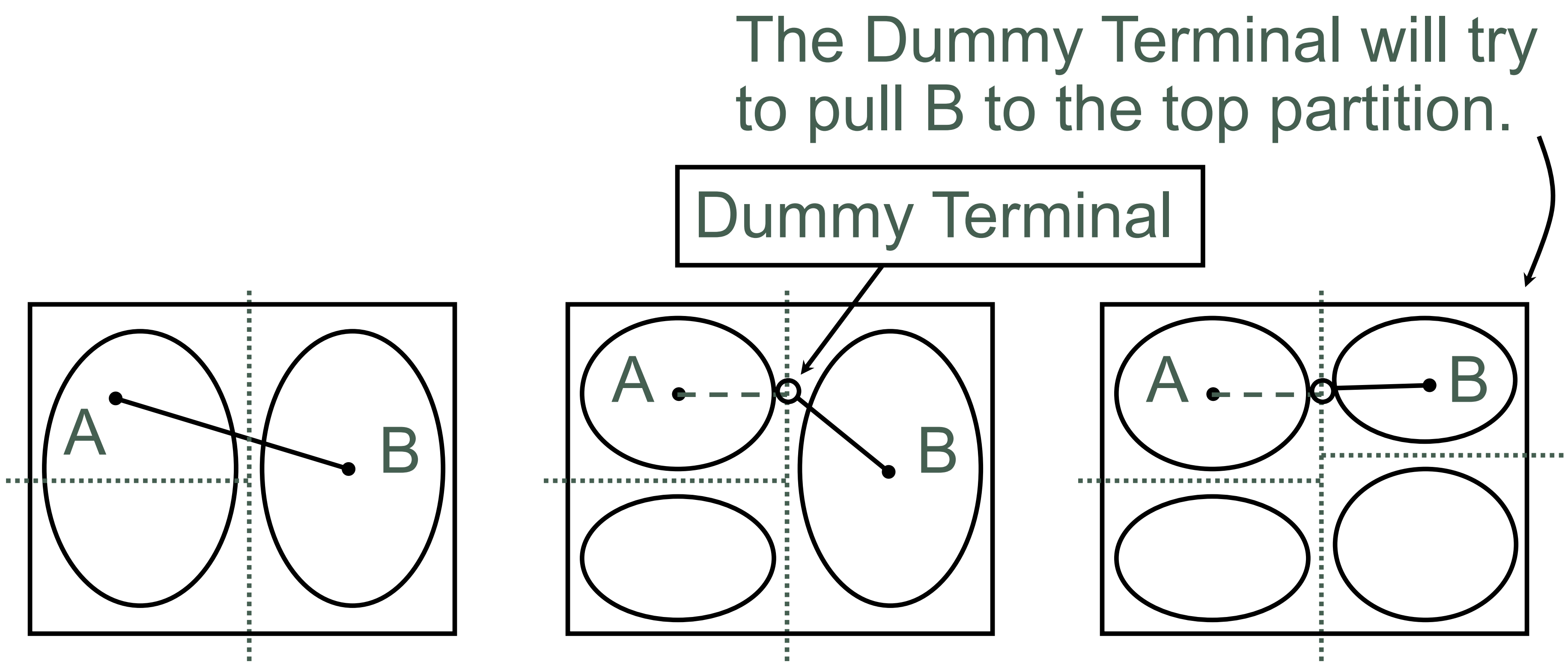
Problem of Partitioning Subcircuits

- Cost of these 2 partitionings are not the same.



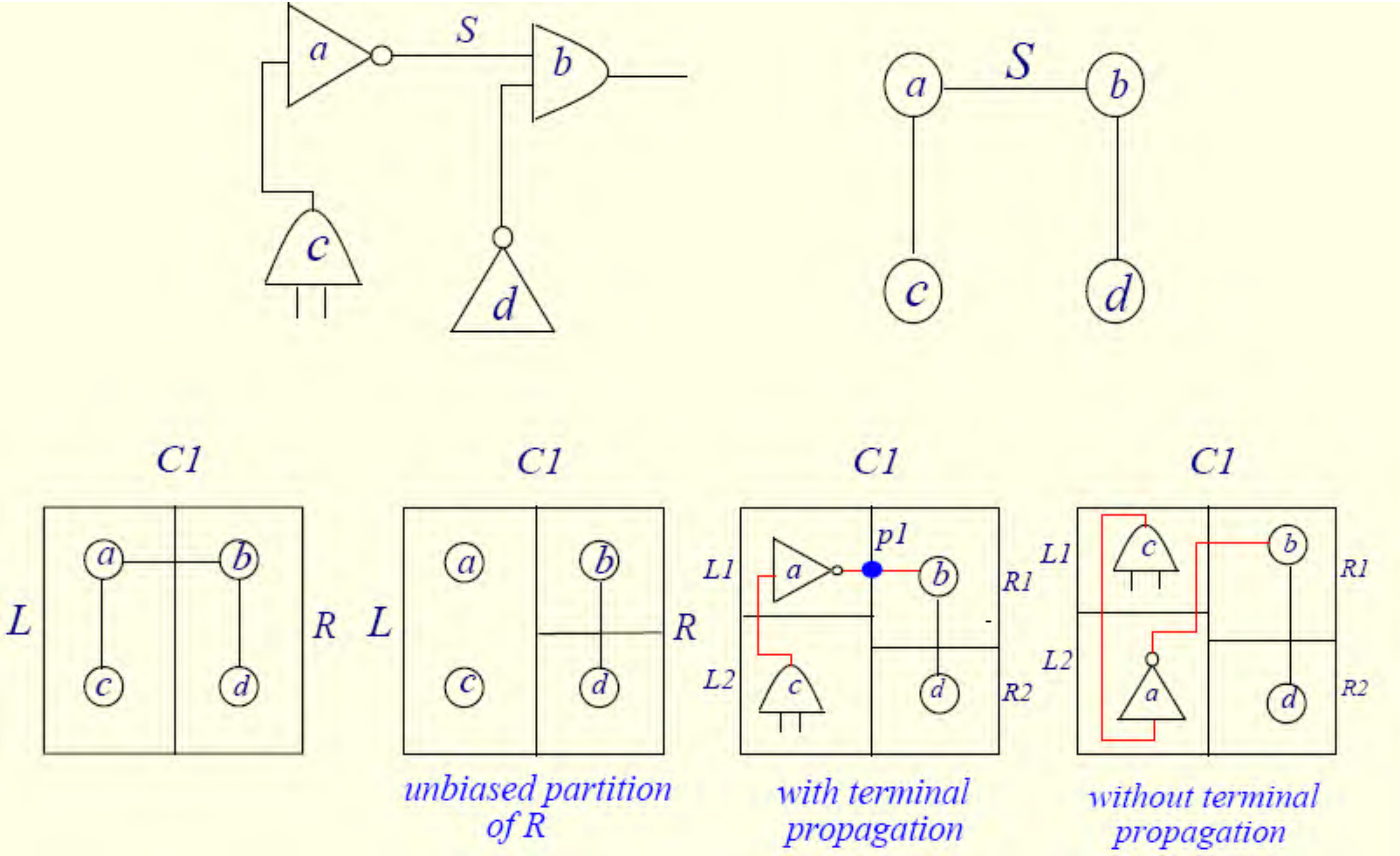
Terminal Propagation

- Need to consider nets connecting to external terminals or other modules as well.
- Do partitioning in a breath-first manner (i.e., finish all higher-level partitioning first).



Dunlop, Alfred E., and Brian W. Kernighan. "A procedure for placement of standard cell VLSI circuits." IEEE TCAD 4.1 (1985): 92-98.

Terminal Propagation



Capo: Can Recursive Bisection Alone Produce Routable Placement

- Standard cell placement, Fixed-die context
- Pure recursive bisectioning placer
  - Several minor techniques to produce good bisections
- Produce good results mainly because:
  - Improvement in mincut bisection using multi-level idea in the past few years
  - Pay attention to details in implementation
- Implementation with good interface (LEF/DEF and GSRC bookshelf) available on web

Caldwell, Andrew E., Andrew B. Kahng, and Igor L. Markov. "Can recursive bisection alone produce routable placements?." Proceedings of DAC 2000.

Capo Approach

- Recursive bisection framework:
  - Multi-level FM for instances with >200 cells
  - Flat FM for instances with 35-200 cells
  - Branch-and-bound for instances with <35 cells
- Careful handling partitioning tolerance:
  - Uncorking: Prevent large cells from blocking smaller cells to move
  - Repartitioning: Several FM calls with decreasing tolerance
  - Block splitting heuristics: Higher tolerance for vertical cut
  - Hierarchical tolerance computation: Instance with more whitespace can have a bigger partitioning tolerance

Summary for Partition Based Placement

- Pros
  - Very fast
  - Great quality
  - Scales nearly linearly with problem size

- Cons
  - Non-trivial to implement
  - Very directed algorithm, but this limits the ability to deal with miscellaneous constraints
  - Not stable (if there is minor change)

Summary for Partition Based Placement

- Improvement in mincut partitioning are conducive to better wirelength and congestion
- Routable placements can be produced in most cases without explicit congestion management
  - Explicit congestion control may still be useful in some cases
- Better weighted wirelength often implies better routed wirelength, but not always

The History of Placement Algorithms

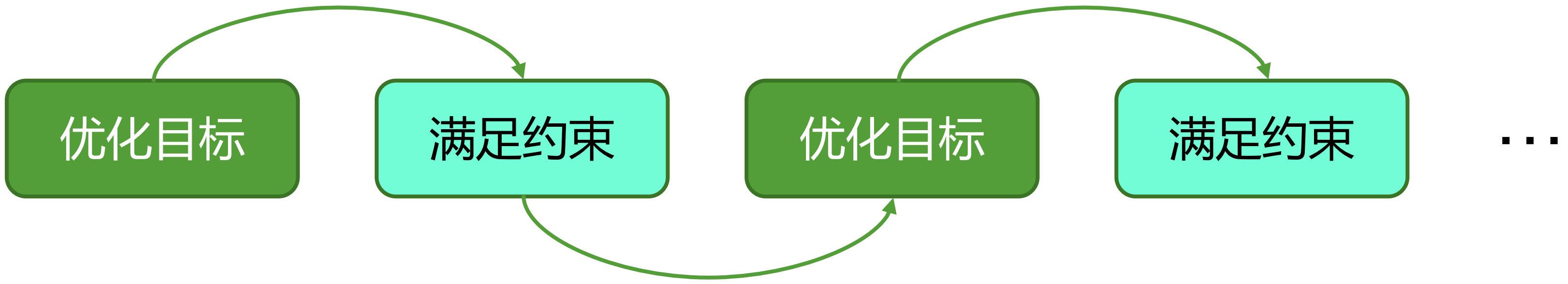
<1970-1980s	1980s-1990s	1990s-2010s			>2010s	
Partitioning	Simulated Annealing	Min-Cut (Multi-level)	Analytic		Analytic	
			Quadratic	Nonlinear	Quadratic	Nonlinear
Breuer	Timberwolf VPR	FengShui	**GORDIAN**	APlace	POLAR	ePlace RePlAce
Dunlop & Kernighan	Dragon	Capo	BonnPlace	Naylor Synopsis	**SimPL** ComPLx	DREAMPlace
Quadratic Assignment		Capo +Rooster	mFar	NTUplace	MAPLE	
Cadence QPlace			**Kraftwerk**	mPL6		
			FastPlace			
			Warp3			
Low quality	Low efficiency					

Quadratic Placement

- Viswanathan, Natarajan, and CC-N. Chu. "FastPlace: Efficient analytical placement using cell shifting, iterative local refinement, and a hybrid net model." IEEE TCAD 2005
- Kim, Myung-Chul, Dong-Jin Lee, and Igor L. Markov. "SimPL: An effective placement algorithm." IEEE TCAD 2011
- Lin, Tao, et al. "POLAR: A high performance mixed-size wirelengh-driven placer with density constraints." IEEE TCAD 2015.

Quadratic Placement

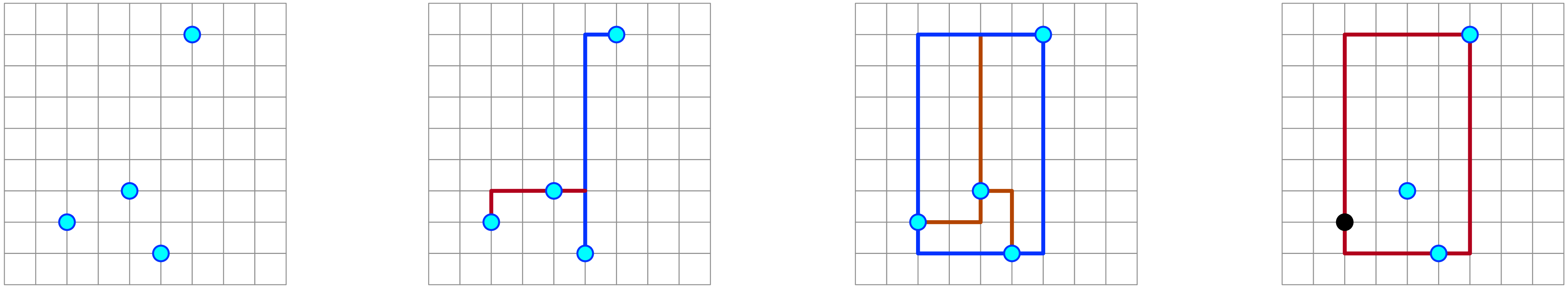
- Placement problem
  - Task: determine the locations of blocks
  - Objective: minimize wirelength
  - Constraints：no overlap between blocks
- Iterative optimization



Quadratic Placement – Optimizing Wirelength Objective

- How to compute wirelength
  - Consider how a net is routed

$$\sum |x_i - x_j| + \sum |y_i - y_j| \qquad max|x_i - x_j| + max\,|y_i - y_j|$$



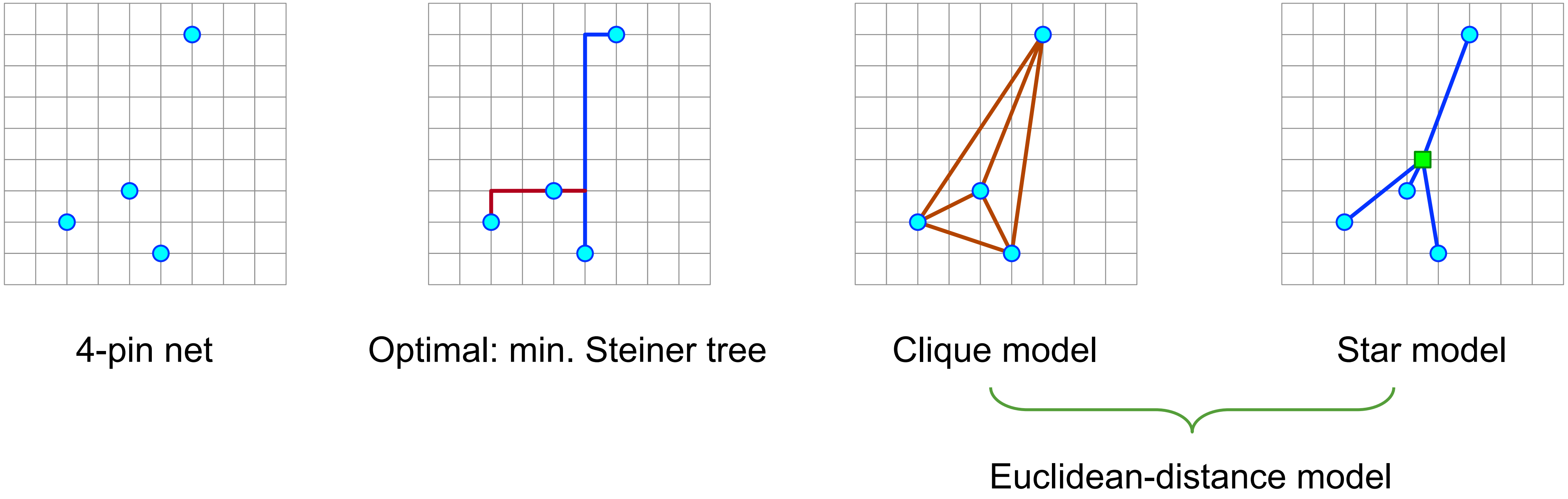
4-pin net | Optimal: min. Steiner tree | Clique model | HPWL

Manhattan-distance model

Quadratic Placement – Optimizing Wirelength Objective

- How to compute wirelength
  - Consider how a net is routed

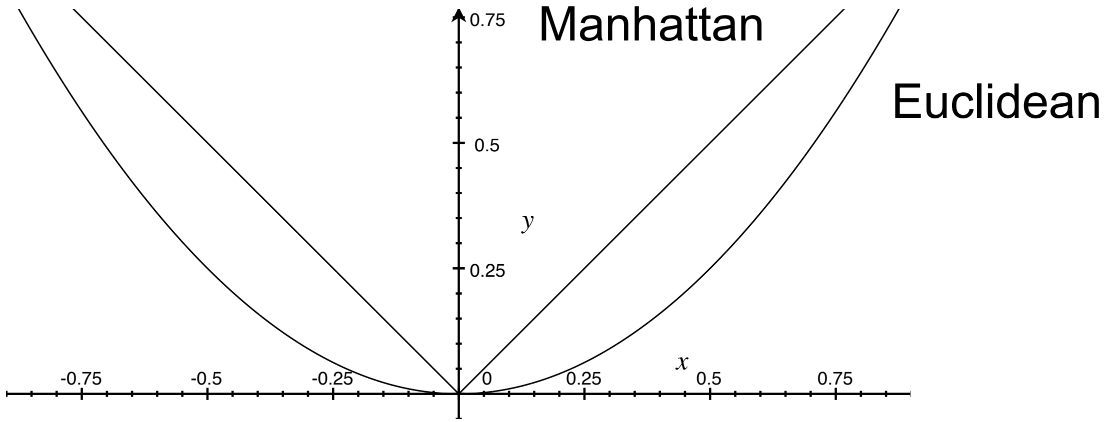
$$\sum (x_i - x_j)^2 + \sum (y_i - y_j)^2$$



Quadratic Placement – Optimizing Wirelength Objective

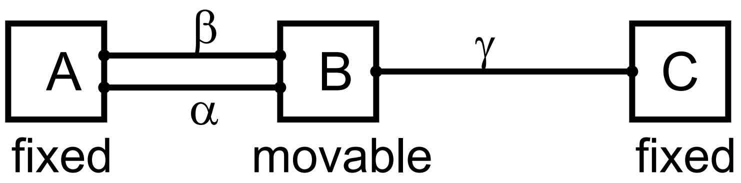
- How to compute wirelength
  - Consider how a net is routed

		Pessimistic		Optimistic	
Model	**Min. Steiner Tree**	**Clique Model**		**HPWL**	**Star Model**
Distance	Manhattan	Manhattan	Euclidean	Manhattan	Euclidean
Accuracy	★★★★★	★	★	★★★★	★★★
Smoothness	★	★★	★★★★★	★★	★★★★★

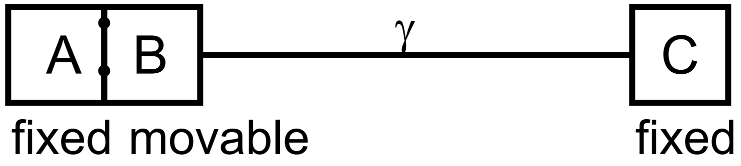


Another Perspective – Linear v.s. Quadratic Objective Function

- Differences between linear and quadratic objective function



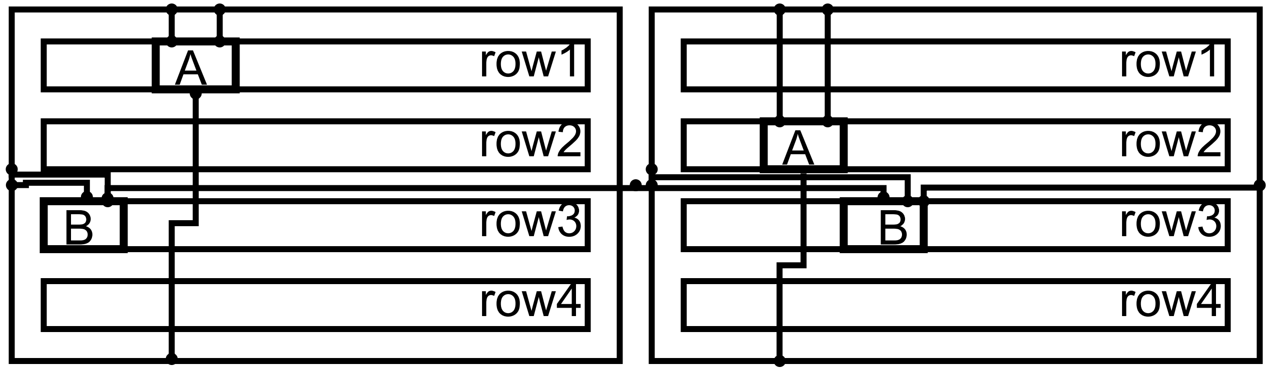
a) Quadratic objective function



b) Linear objective function

Another Perspective – Linear v.s. Quadratic Objective Function

- Quadratic objective function tends to make very long net shorter than linear objective function does, and let short nets become slightly longer



Linear objective function Quadratic objective function

Quadratic Placement – Optimizing Wirelength Objective

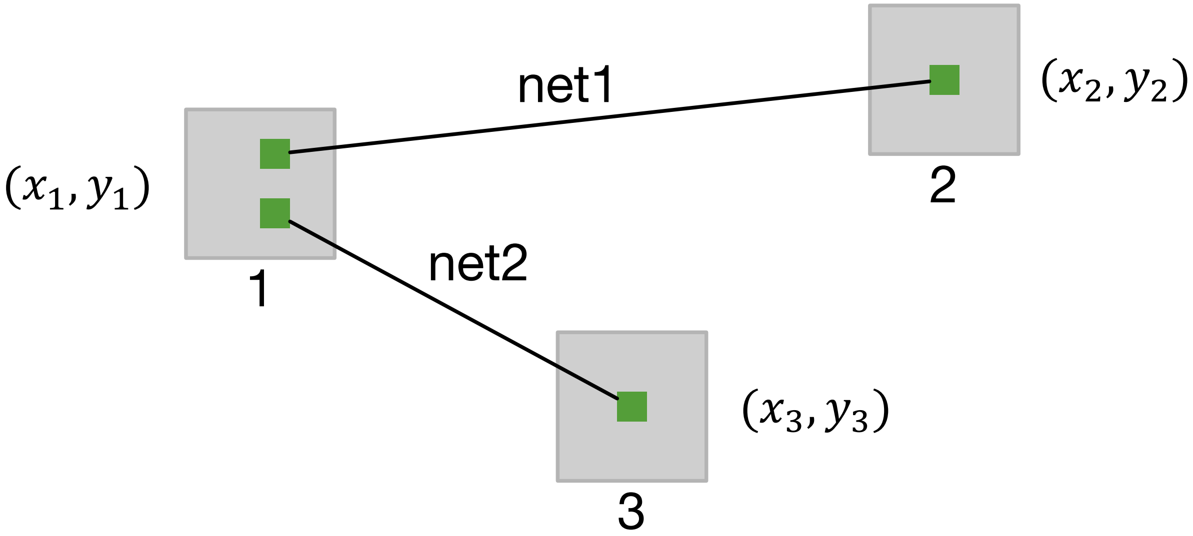
Compute wirelength for **net1**

$$WL_1 = (x_1 - x_2)^2 + (y_1 - y_2)^2$$

Compute wirelength for **net2**

$$WL_2 = (x_1 - x_3)^2 + (y_1 - y_3)^2$$

Minimize
total wirelength



Quadratic Placement – Optimizing Wirelength Objective

Compute wirelength for **net1**

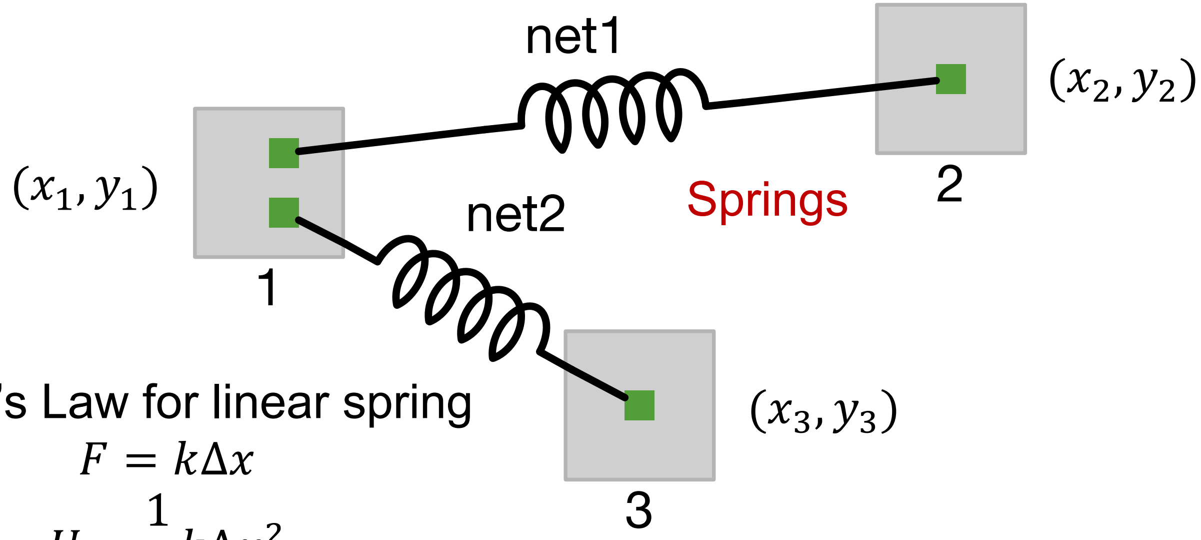
$$WL_1 = (x_1 - x_2)^2 + (y_1 - y_2)^2$$

Compute wirelength for **net2**

$$WL_2 = (x_1 - x_3)^2 + (y_1 - y_3)^2$$

Minimize total wirelength

Physical intuition?



Hooke's Law for linear spring

$$F = k\Delta x$$

$$U = \frac{1}{2}k\Delta x^2$$

How to solve?

Quadratic Programming (QP)

$$\min. \frac{1}{2}x^T Ax - b^T x$$

Gradient of x

$$Ax - b = 0$$

Quadratic Placement – Optimizing Wirelength Objective

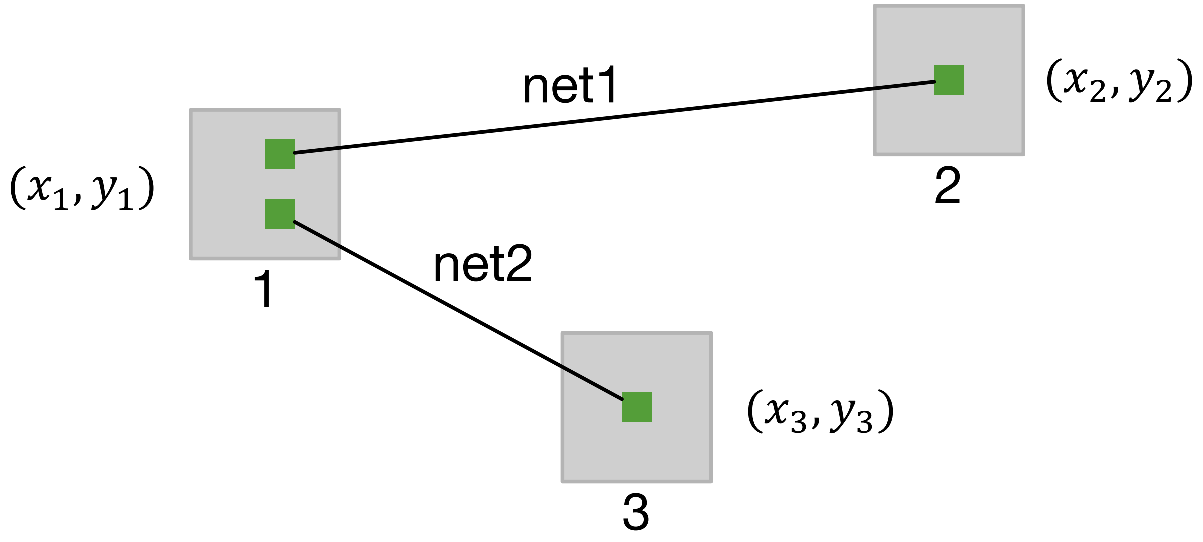
Compute wirelength for **net1**

$$WL_1 = (x_1 - x_2)^2 + (y_1 - y_2)^2$$

Compute wirelength for **net2**

$$WL_2 = (x_1 - x_3)^2 + (y_1 - y_3)^2$$

Minimize total wirelength



Any problem?

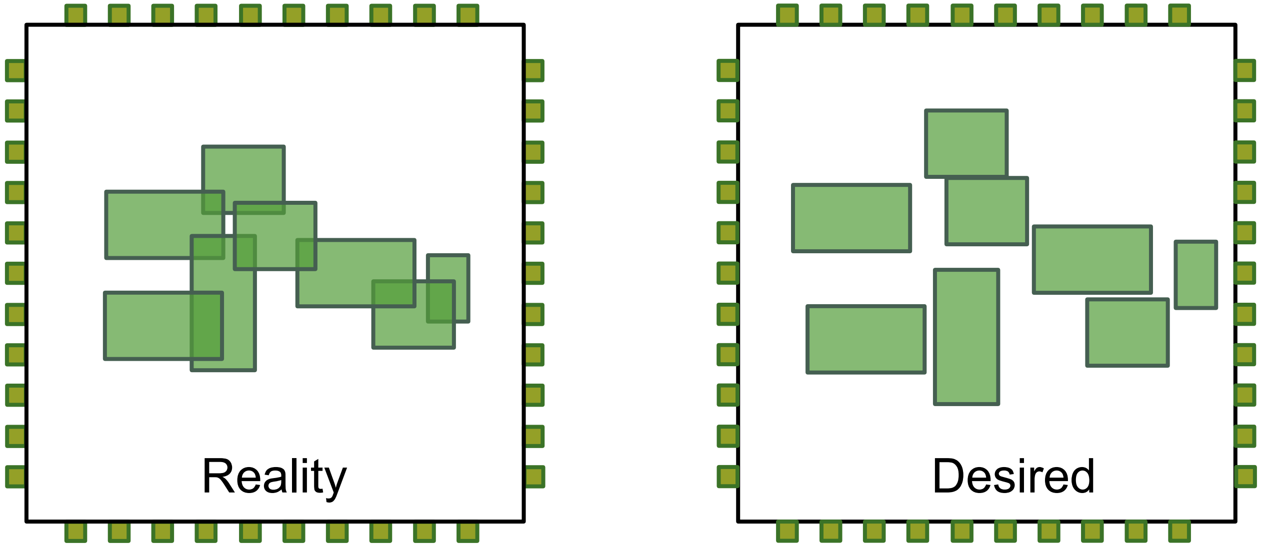
Optimal solution

$$x_1 = x_2 = x_3$$
$$y_1 = y_2 = y_3$$

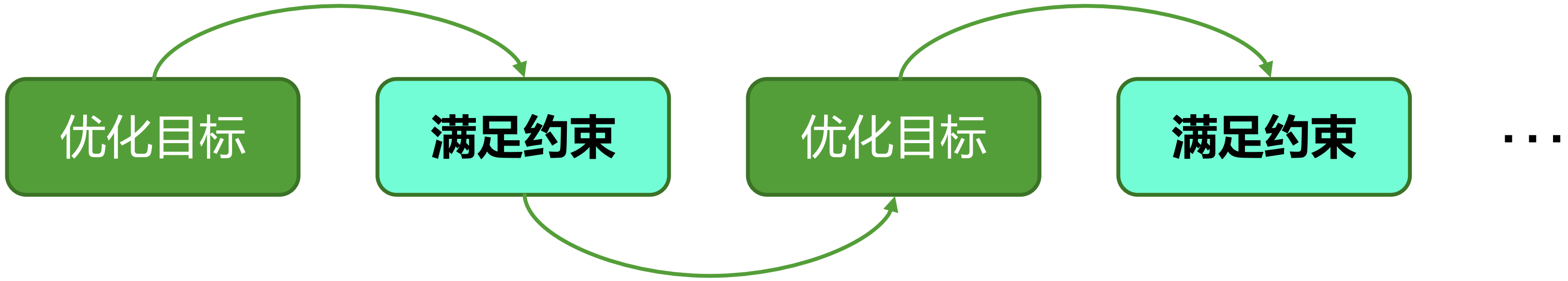
All blocks overlap!

Quadratic Placement – Optimizing Wirelength Objective

- Optimizing the objective results in overlaps

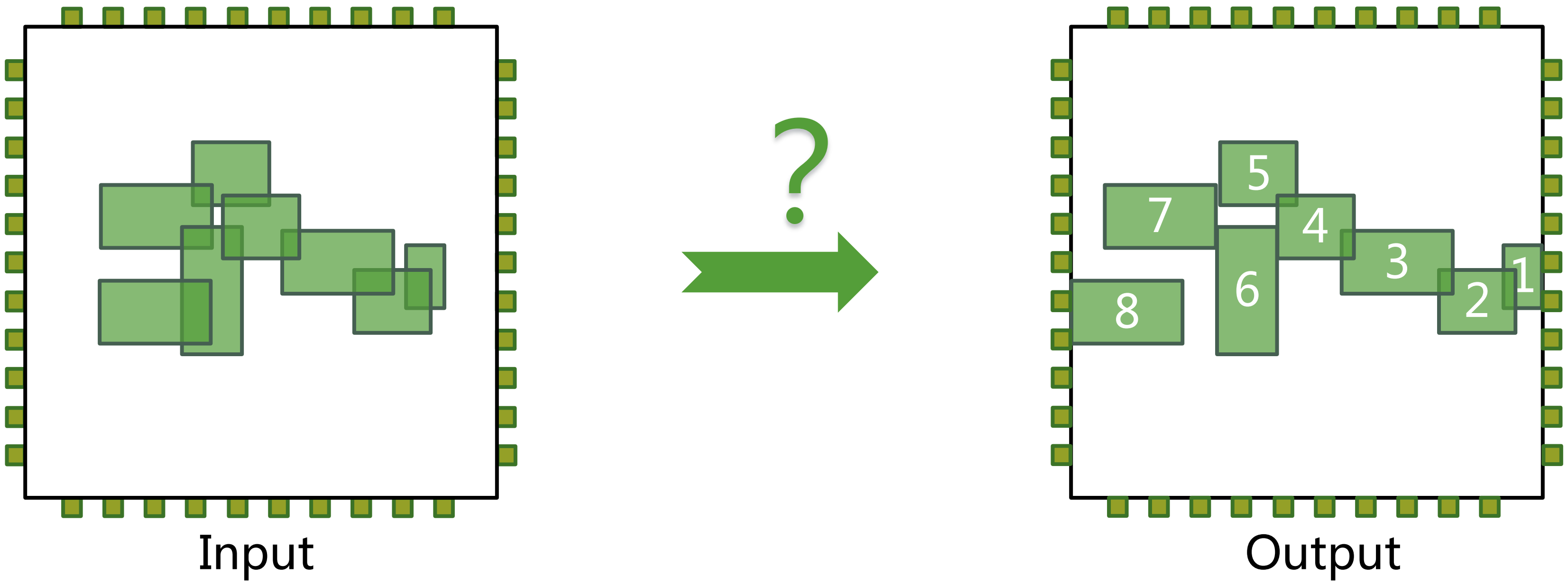


- Iterative optimization



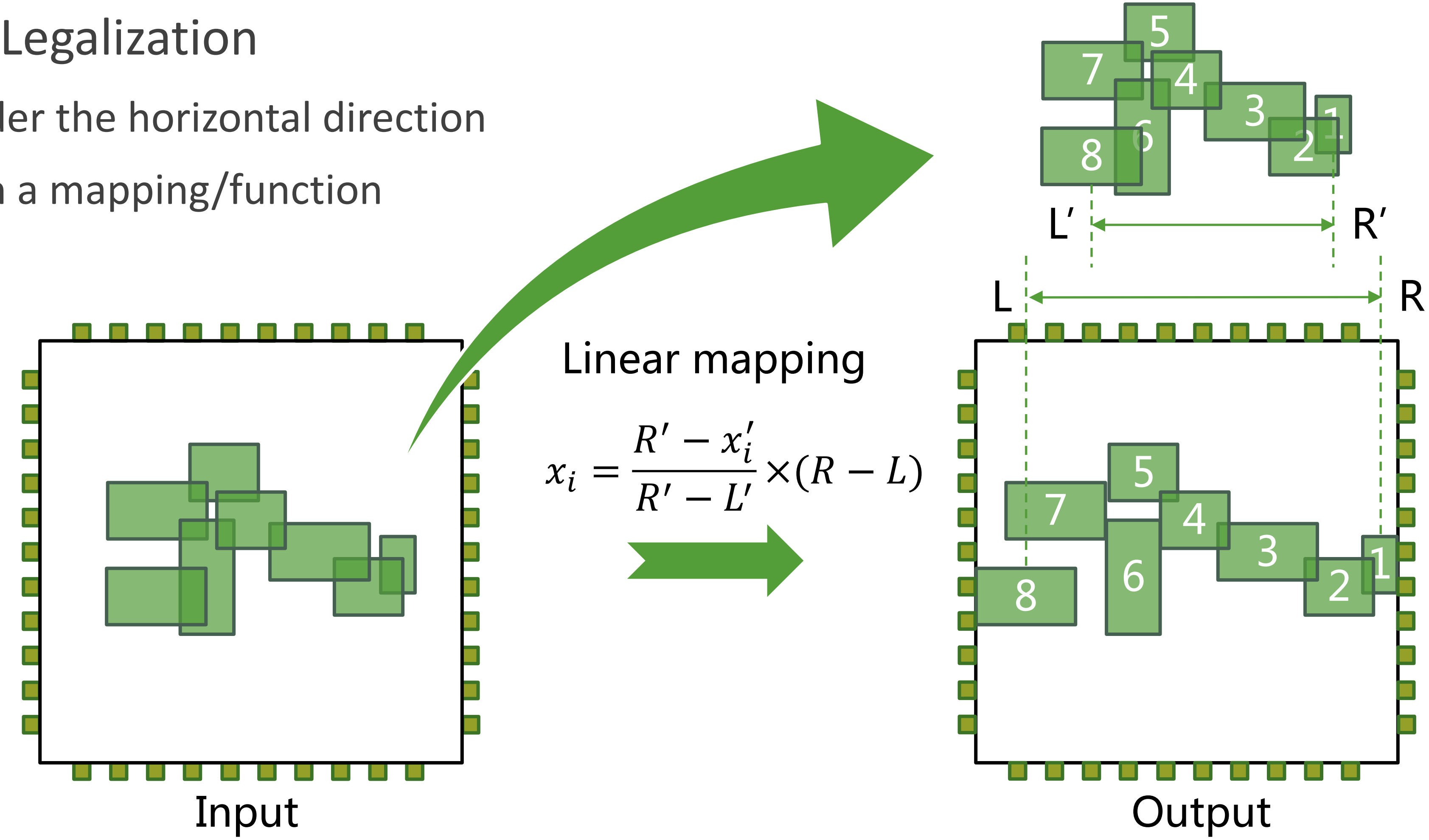
Quadratic Placement – Satisfying Constraints

- Rough Legalization
  - Consider the horizontal direction
  - Design a mapping/function



Quadratic Placement – Satisfying Constraints

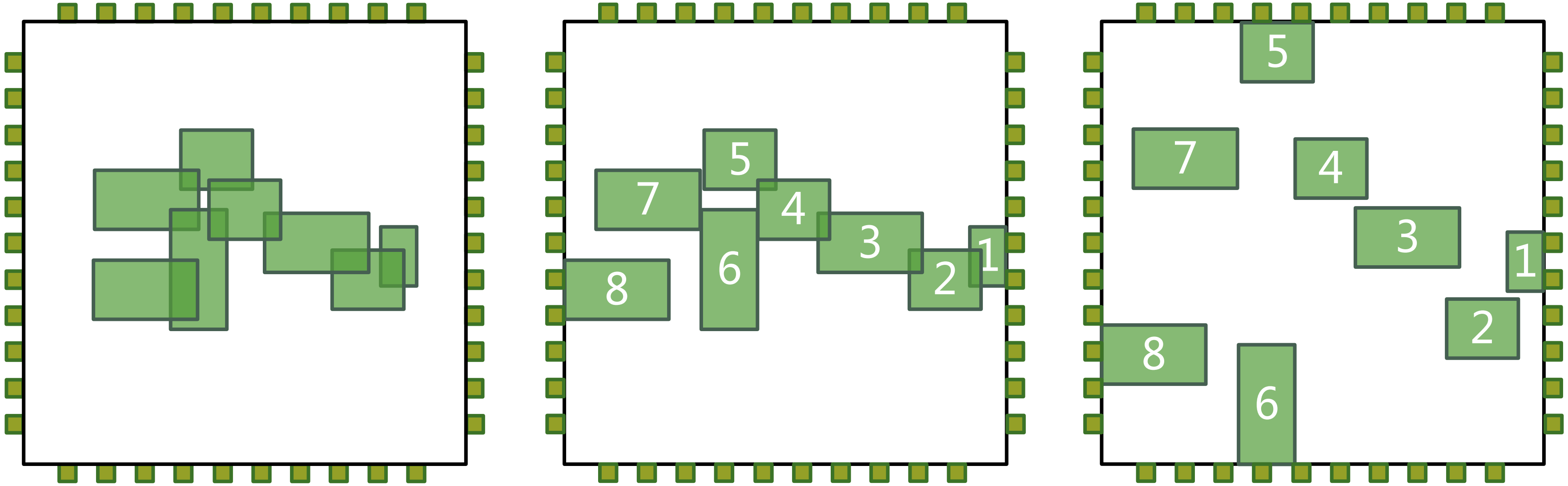
- Rough Legalization
  - Consider the horizontal direction
  - Design a mapping/function



$$x_i = \frac{R' - x_i'}{R' - L'} \times (R - L)$$

Quadratic Placement – Satisfying Constraints

- Rough Legalization
  - Do it for horizontal and vertical directions

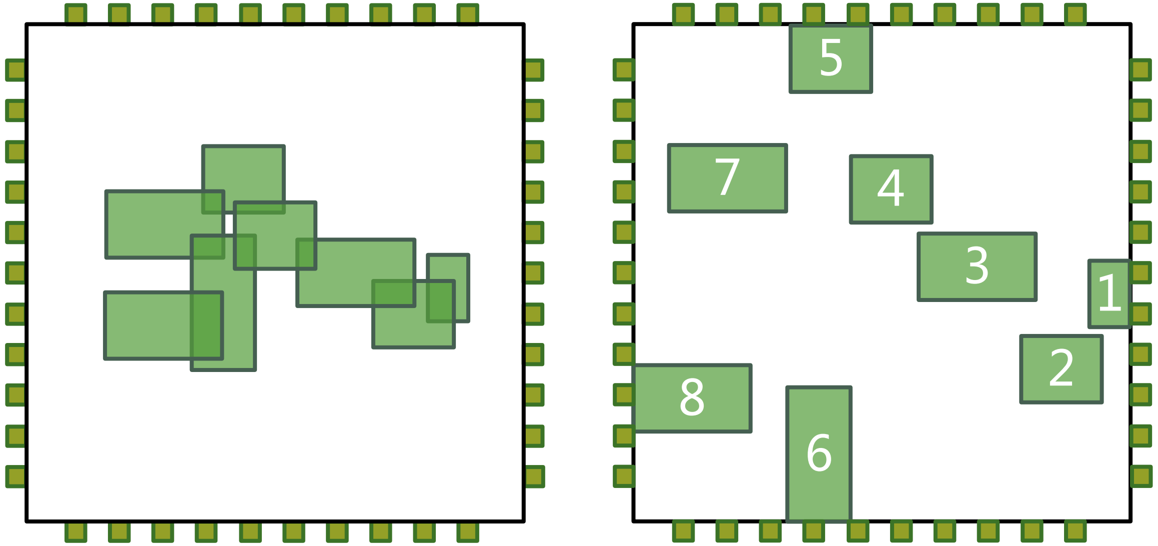


Input Horizontal Vertical

Quadratic Placement – Satisfying Constraints

- Rough Legalization

Original obj. $$min. \cdots + \overbrace{\sum (x_6 - x_j)^2}^{\text{j connected to 6}} + \cdots$$



Input Rough legalization

有没有问题？

优化目标

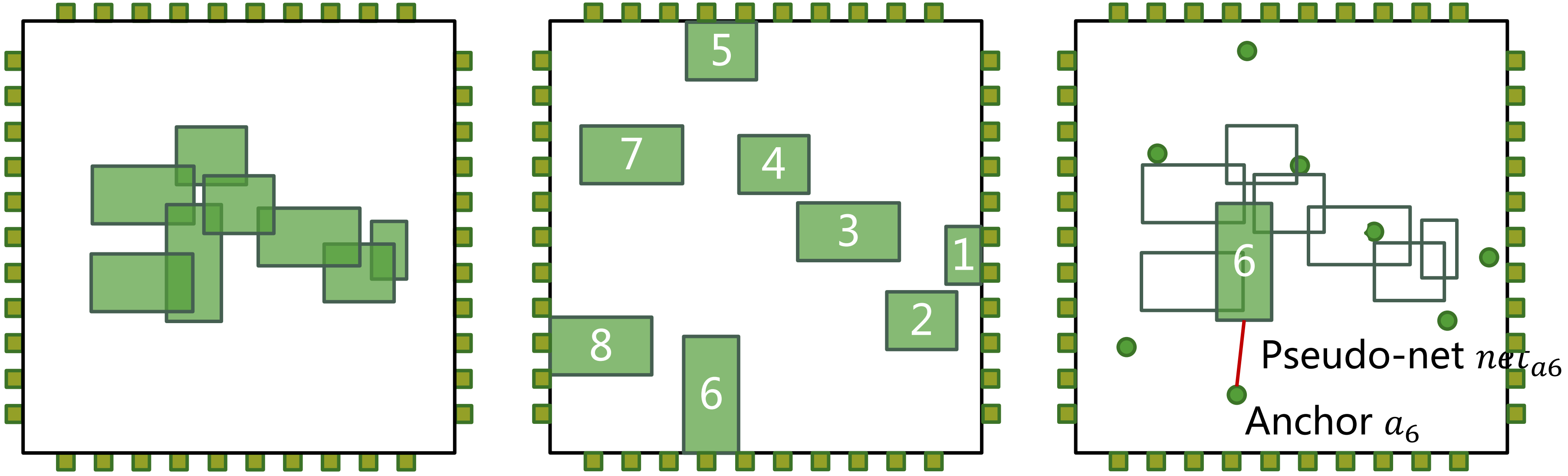
Quadratic Placement – Satisfying Constraints

- Rough Legalization
  - Leverage **anchors**

Original obj. $min. \cdots + \overbrace{\sum (x_6 - x_j)^2}^{\text{j connected to 6}} + \cdots$

Anchor loc.

Obj. with anchors $min. \cdots + \sum (x_6 - x_j)^2 + \underbrace{w_6 (x_6 - x_{a_6})^2}_{WL_{net_{a6}}} + \cdots$

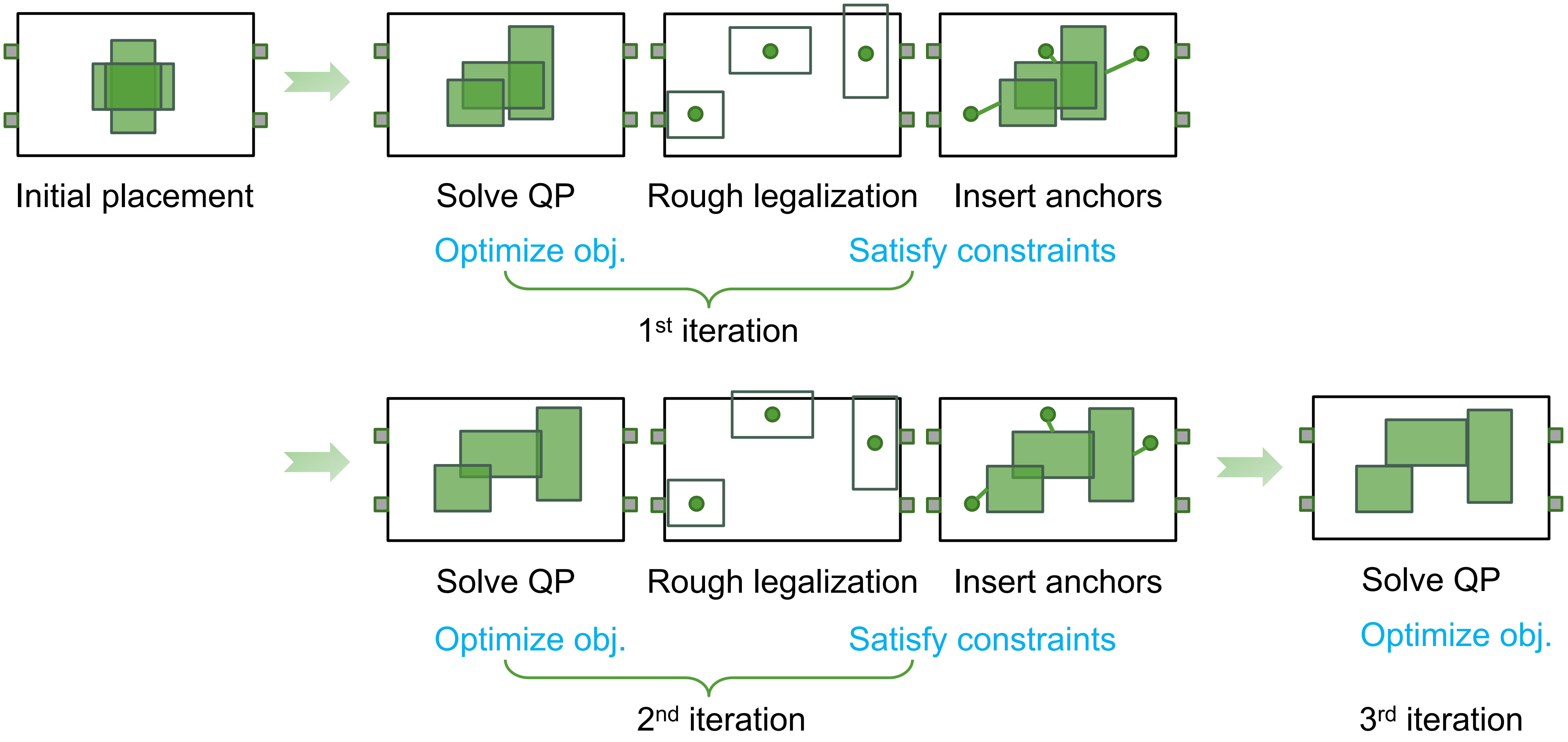


Input

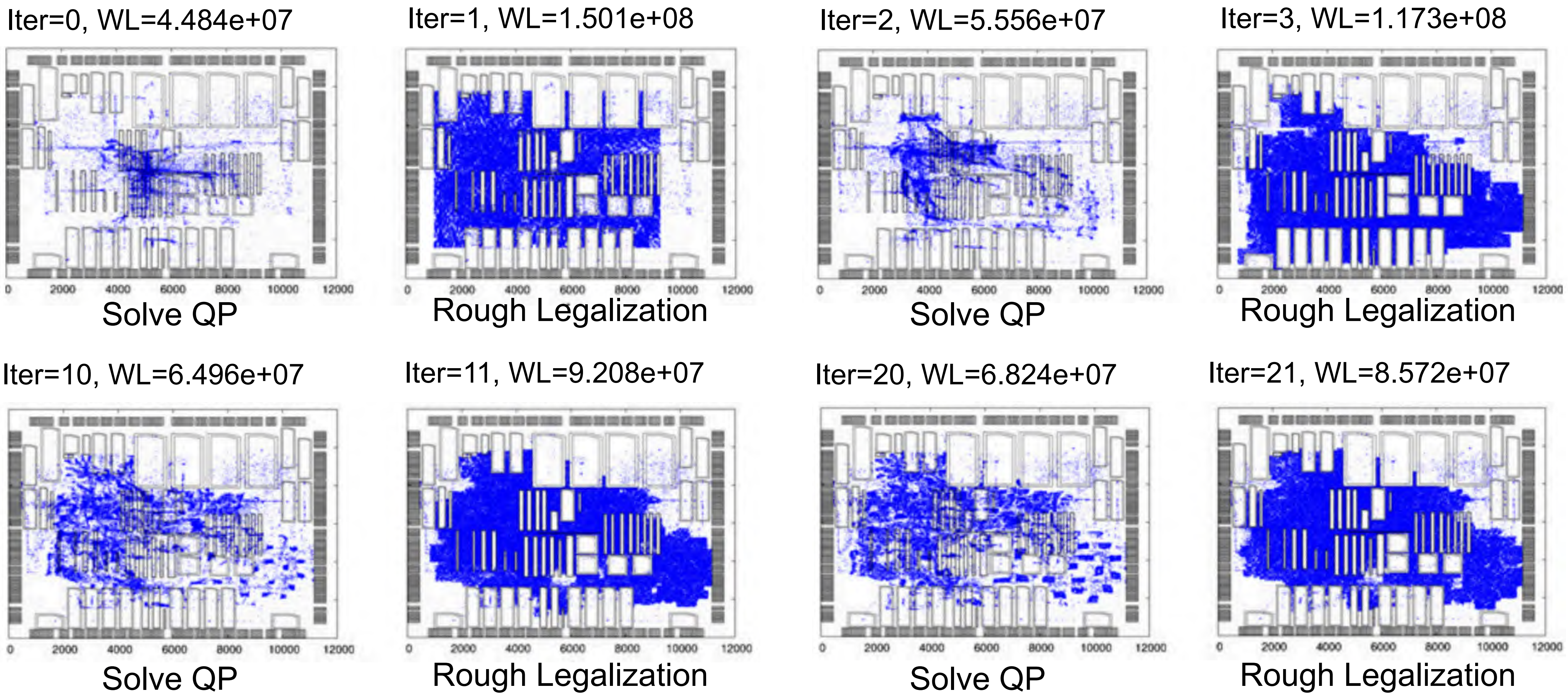
Rough legalization

Insert anchors

Quadratic Placement – Overall Optimization Flow



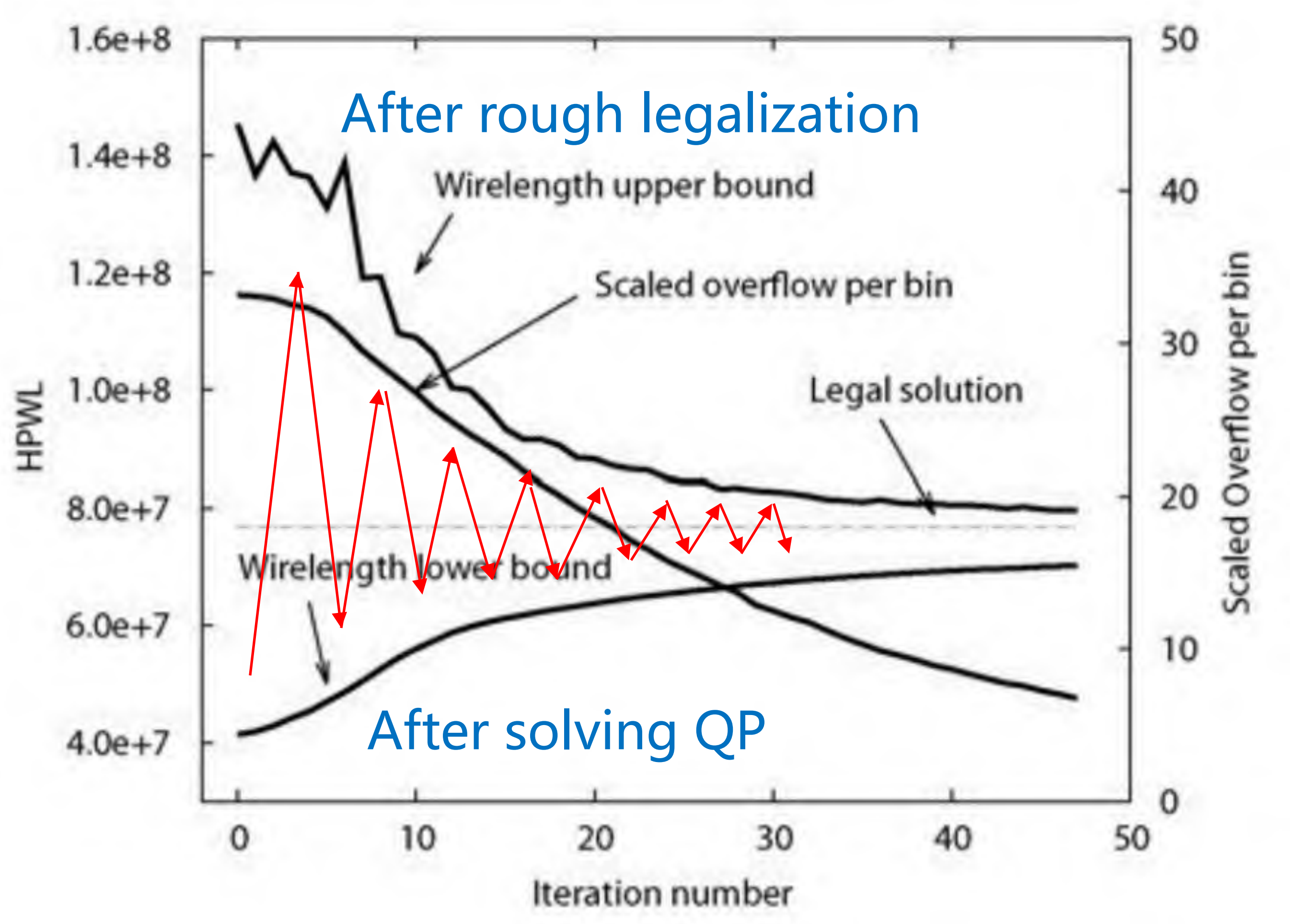
Quadratic Placement – 211K-Cell Example



Quadratic Placement – 211K-Cell Example

Measuring the density distribution

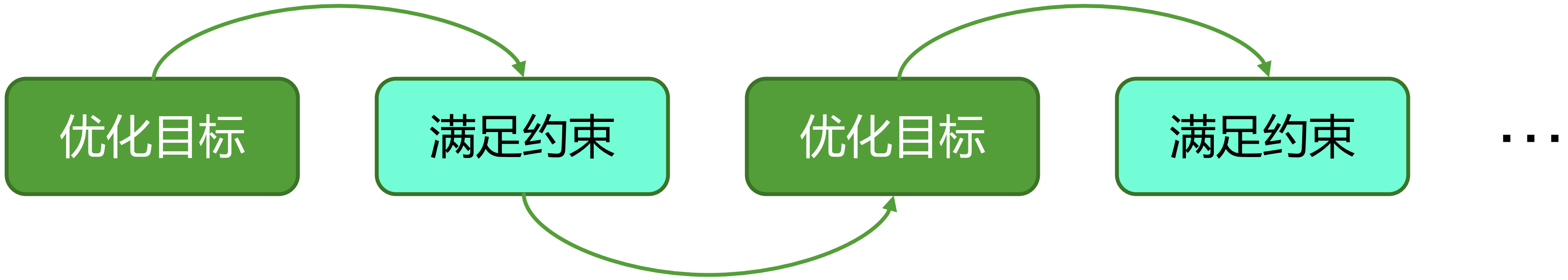
$$overflow = \sum_{i \in all\ bins} \max(D_i - 1, 0)$$



[SimPL, TCAD2011]

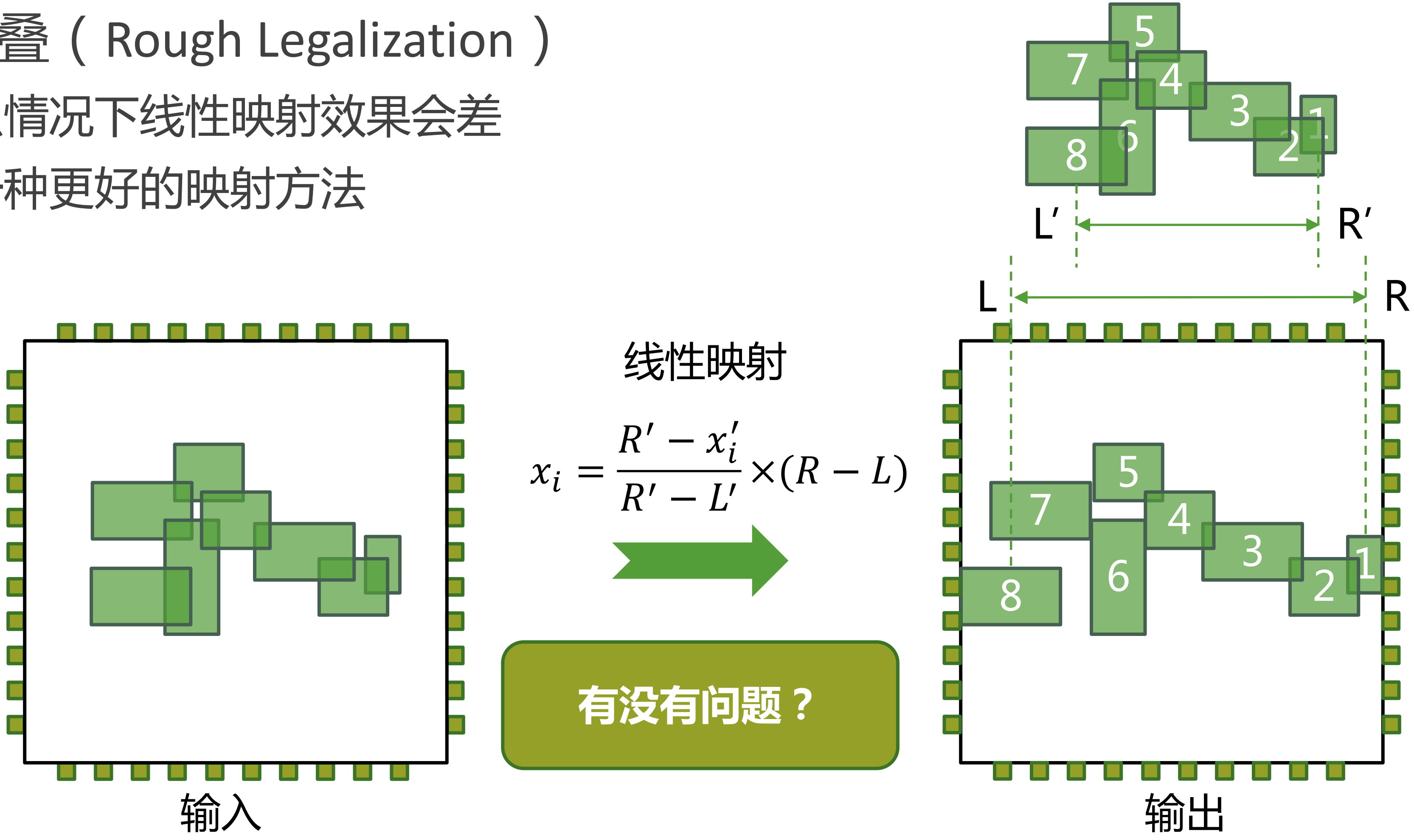
Quadratic Placement – Summary

- Iterative optimization
- Wirelength models：HPWL, clique model, star model
- Rough legalization



课后思考

- 去除重叠（Rough Legalization）
  - 在什么情况下线性映射效果会差
  - 设计一种更好的映射方法



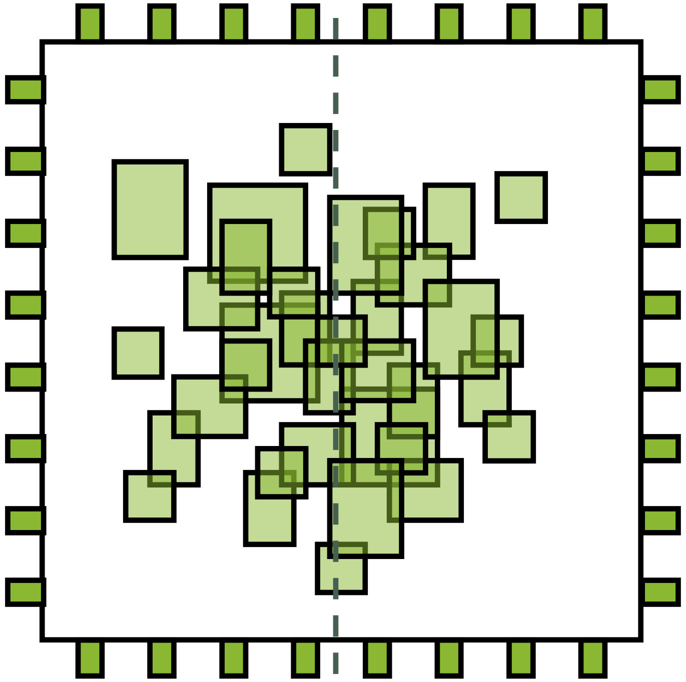
Quadratic Placement – Gordian

- Global optimization
  - Solves a sequence of quadratic programming problems
  - $\min_{x,y} \sum w_i \left((x_i - x_j)^2 + \left(y_i - y_j\right)^2 \right)$
- Partitioning
  - Enforces the non-overlap constraints

Kleinhans, Jürgen M., et al. "GORDIAN: VLSI placement by quadratic programming and slicing optimization." IEEE TCAD 10.3 (1991): 356-365.
Sigl, Georg, Konrad Doll, and Frank M. Johannes. "Analytical placement: A linear or a quadratic objective function?." Proceedings of DAC. 1991.

Partitioning

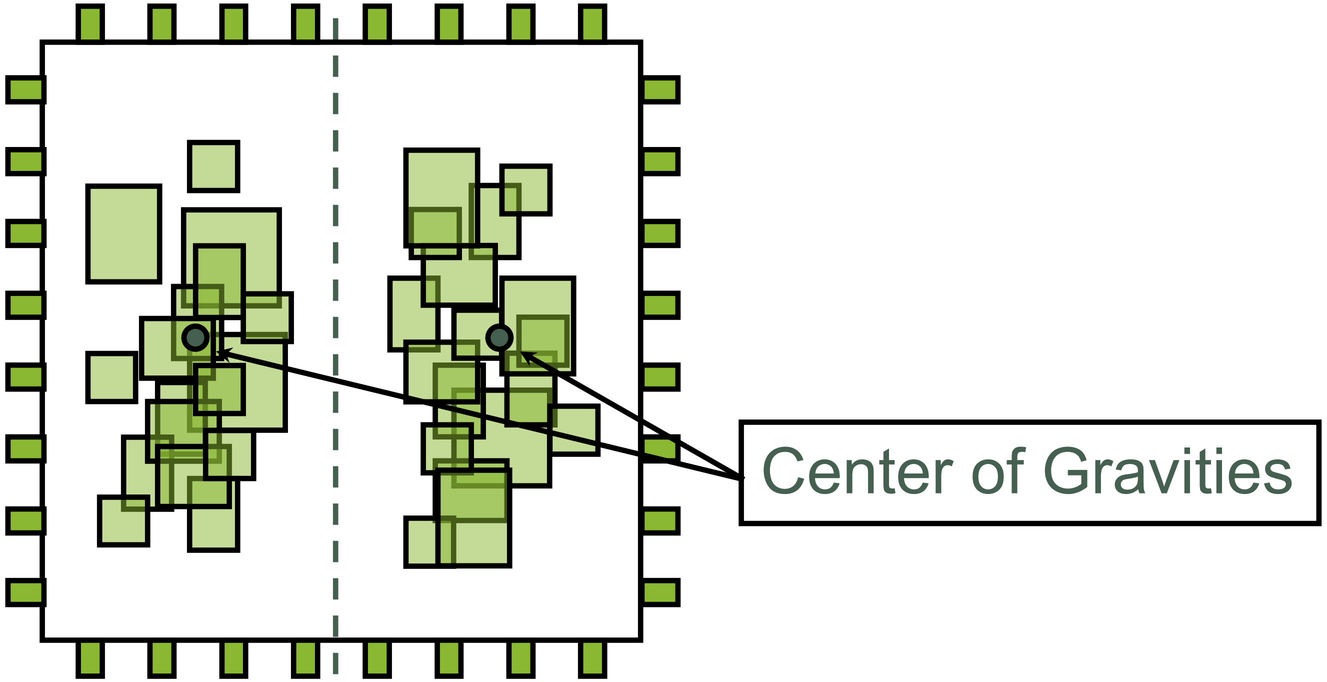
- Find a good cut direction and position.



- Improve the cut value using FM.

Applying the Idea Recursively

- Before every level of partitioning, do the Global Optimization again with additional constraints that the center of gravities should be in the center of regions.



- Always solve a single QP (i.e., global).

Center of Gravity Constraints

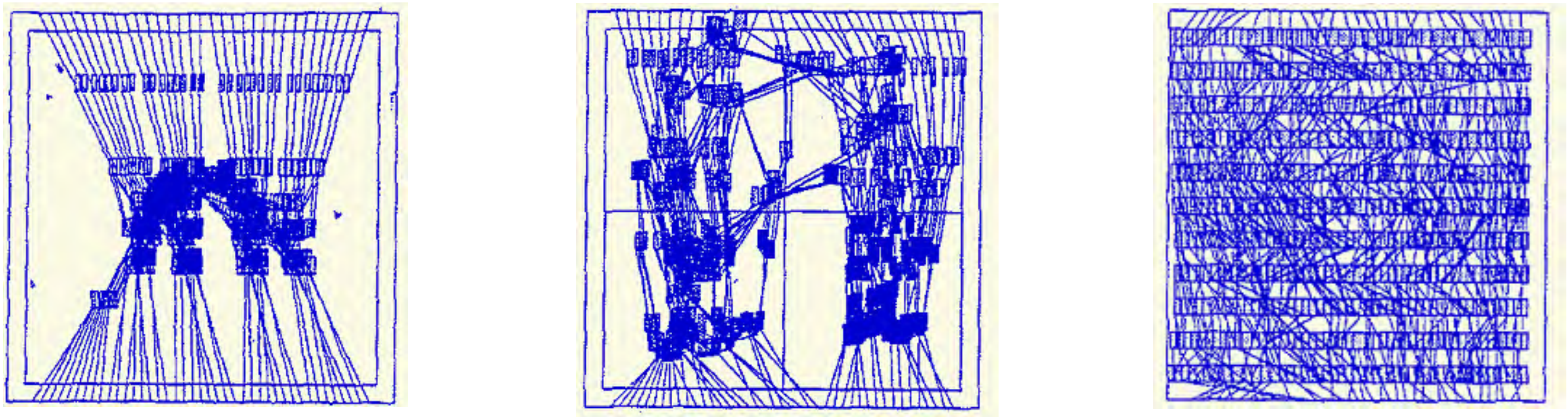
The center of gravity constraints

- At level l, chip is divided into q ($\leq 2^l$) regions
- For region p, the center coordinates: (u_p, v_p)
- M_p: set of modules in region p
- Matrix from for all regions

$$\text{constraint}: \sum_{u \in M_p} F_u x_u = u_p \sum_{u \in M_p} F_u$$

$$A^l X = u^l,\ a_{iu} = \begin{cases} F_i / \sum_{i \in M_p} F_i & \text{if } i \in M_p \\ 0 & \text{otherwise} \end{cases}$$

Process of Gordian



(a) Global placement with 1 region

(b) Global placement with 4 region

(c) Final placements

Force-Directed Placement – Kraftwerk

- Iteratively solve the quadratic formulation:

$$\text{Min} \quad f(p) = \tfrac{1}{2} p^T C p + d^T p + const$$

$$\Rightarrow Cp + d = 0$$

// equivalent to spring force
// equilibrium

- Spread cells by additional forces:

$$Cp + d + f = 0$$

Eisenmann, Hans, and Frank M. Johannes. "Generic global placement and floorplanning." Proceedings of DAC. 1998.

Requirements to the Additional Force

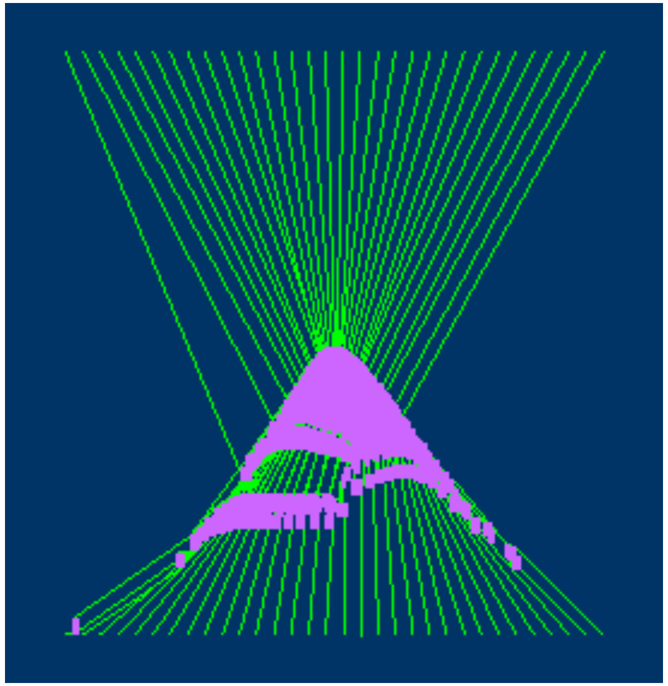
- For a given placement, the additional force working on a cell depends only on the coordinates of the cells
- Regions with higher density are the sources of the forces. Regions with lower density are the sinks
- The forces do not form circles
- In infinity, the force should be zero
- Density-based force proposed
  - Push cells away from dense region to sparse region

$$f(x,y) = \frac{k}{2\pi}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} D(x',y')\frac{\vec{r}-\vec{r}\,'}{\left|\vec{r}-\vec{r}\,'\right|^2}\,dx'dy'$$

where $\vec{r} = (x,y)$ and $\vec{r}\,' = (x',y')$

Some Potential Problems of Kraftwerk

- Convergence is difficult to control
  - Large K → oscillation
  - Small K → slow convergence



- Example: Layout of a multiplier

- Density-based force is expensive to compute

$$f(x,y)=\frac{k}{2\pi}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}D(x',y')\frac{\vec{r}-\vec{r}'}{\left|\vec{r}-\vec{r}'\right|^{2}}dx'dy'$$

The History of Placement Algorithms

<1970-1980s	1980s-1990s	1990s-2010s			>2010s	
Partitioning	Simulated Annealing	Min-Cut (Multi-level)	Analytic		Analytic	
			Quadratic	Nonlinear	Quadratic	Nonlinear
Breuer	Timberwolf VPR	FengShui	GORDIAN	APlace	POLAR	**ePlace RePlAce**
Dunlop & Kernighan	Dragon	Capo	BonnPlace	Naylor Synopsis	SimPL ComPLx	**DREAMPlace**
Quadratic Assignment		Capo +Rooster	mFar	NTUplace	MAPLE	
Cadence QPlace			Kraftwerk	mPL6		
			FastPlace			
			Warp3			
Low quality	Low efficiency					

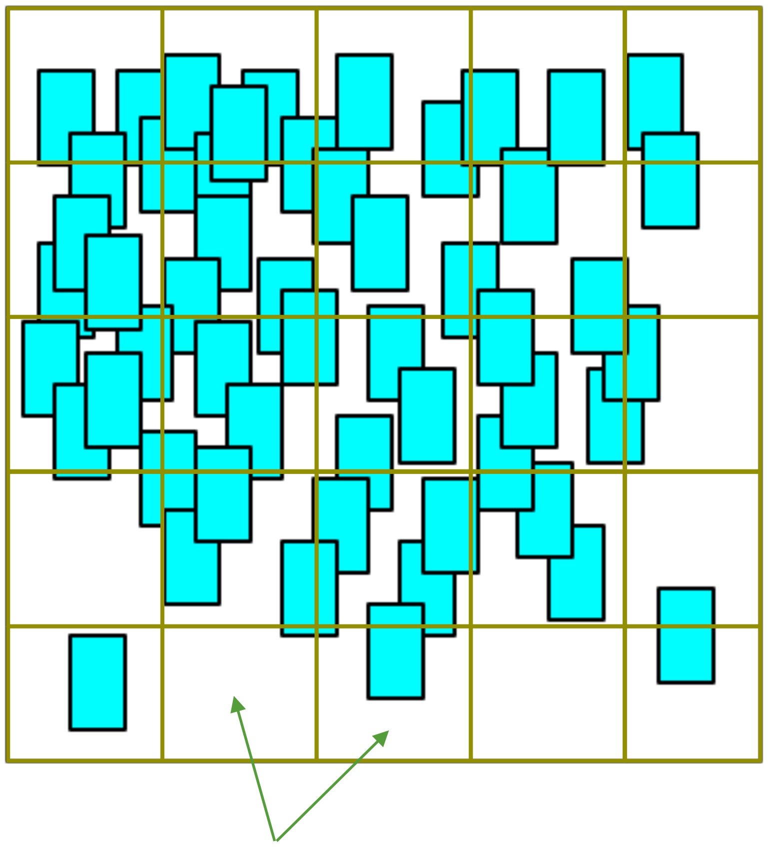
Nonlinear Placement

- Mathematical formulation
  - d_i denotes the density of bin i

$$\begin{aligned} &\min_{\boldsymbol{x},\boldsymbol{y}} \quad WL(\boldsymbol{x},\boldsymbol{y}), \\ &s.t. \quad d_b(\boldsymbol{x},\boldsymbol{y}) \leq t_d, \forall b \in Bins \end{aligned}$$

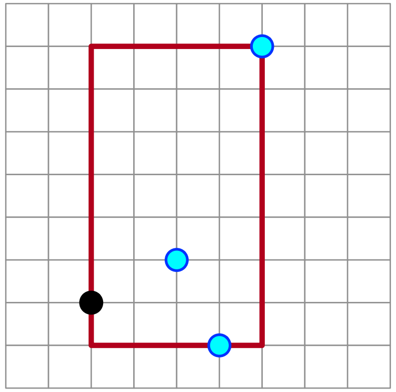
- Nonlinear placement objective
  - Lagrangian relaxation

$$\min_{\boldsymbol{x},\boldsymbol{y}} \underbrace{WL(\boldsymbol{x},\boldsymbol{y})}_{\text{Wirelength}} + \lambda \underbrace{D(\boldsymbol{x},\boldsymbol{y})}_{\text{Density}}$$



Bins

Wirelength Smoothing

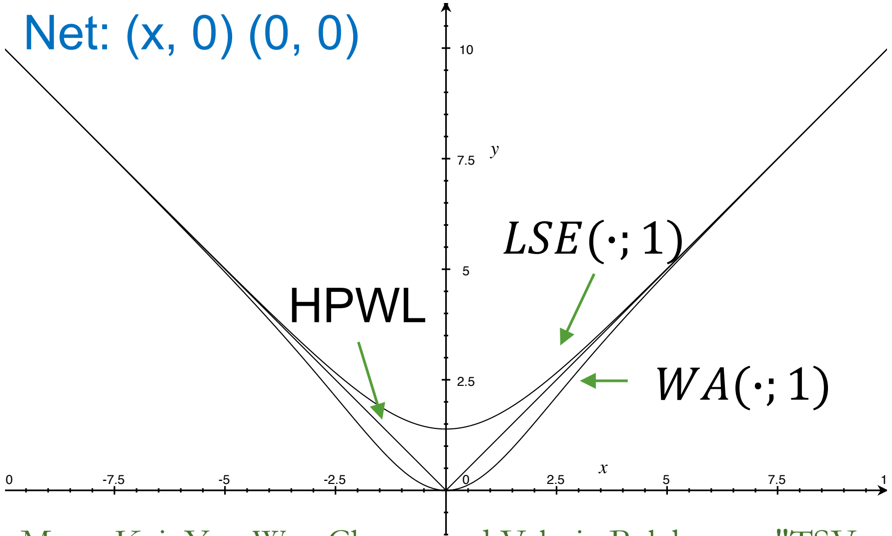


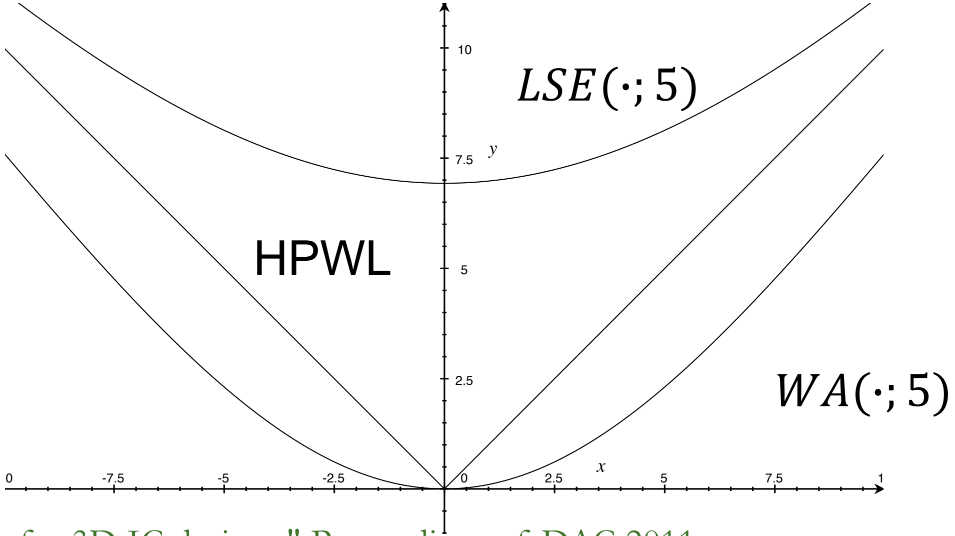
- $WL(\boldsymbol{x},\boldsymbol{y}) = \sum_{e\in E} WL_e(\boldsymbol{x},\boldsymbol{y})$
- $HPWL = max|x_i - x_j| + max\,|y_i - y_j|$
  - Equivalently $\left(\max_i x_i - \min_i x_i\right) + \left(\max_i y_i - \min_i y_i\right)$
- Log-sum-exp (LSE)
  - $LSE(\boldsymbol{x};\gamma) = \gamma \ln \sum_i e^{\frac{x_i}{\gamma}}$
  - $\max\{x_1,\cdots,x_n\} < LSE(\boldsymbol{x};\gamma) \le \max\{x_1,\cdots,x_n\} + \gamma\ln(n)$
  - $LSE(\boldsymbol{x};\gamma) \approx \max\{x_1,\cdots,x_n\}$
  - $-LSE(\boldsymbol{x};-\gamma) \approx \min\{x_1,\cdots,x_n\}$
  - $WL_e(\boldsymbol{x},\boldsymbol{y};\gamma) = \gamma(\underbrace{\ln\sum_{v_i\in e} e^{\frac{x_i}{\gamma}} + \ln\sum_{v_i\in e} e^{-\frac{x_i}{\gamma}}}_{\text{x}} + \underbrace{\ln\sum_{v_i\in e} e^{\frac{y_i}{\gamma}} + \ln\sum_{v_i\in e} e^{-\frac{y_i}{\gamma}}}_{\text{y}})$

Wirelength Smoothing

- Weighted average (WA)
  - $$WL_e(\boldsymbol{x},\boldsymbol{y};\gamma) = \underbrace{\left(\frac{\sum_{v_i\in e} x_i e^{x_i/\gamma}}{\sum_{v_i\in e} e^{x_i/\gamma}} - \frac{\sum_{v_i\in e} x_i e^{-x_i/\gamma}}{\sum_{v_i\in e} e^{-x_i/\gamma}}\right)}_{\text{x}} + \underbrace{\left(\frac{\sum_{v_i\in e} y_i e^{y_i/\gamma}}{\sum_{v_i\in e} e^{y_i/\gamma}} - \frac{\sum_{v_i\in e} y_i e^{-y_i/\gamma}}{\sum_{v_i\in e} e^{-y_i/\gamma}}\right)}_{\text{y}}$$
- Larger γ → smoother, but less accurate

More recent work
DAC2019
BiG: Bivariant smoothing





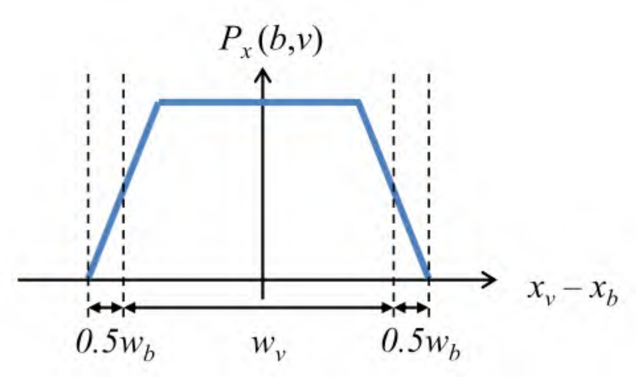
Hsu, Meng-Kai, Yao-Wen Chang, and Valeriy Balabanov. "TSV-aware analytical placement for 3D IC designs." Proceedings of DAC 2011.
Sun, Fan-Keng, and Yao-Wen Chang. "BiG: A bivariate gradient-based wirelength model for analytical circuit placement." Proceedings of DAC 2019.

Nonlinear Placement – NTUplace

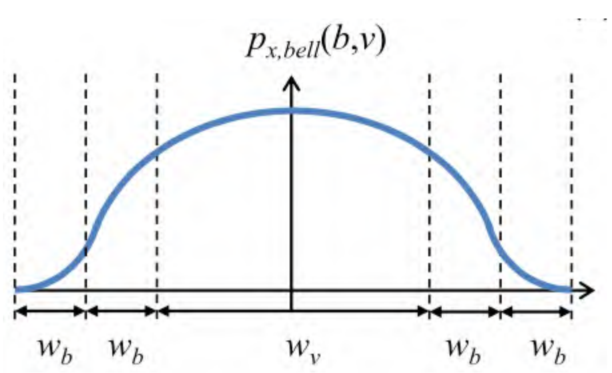
- Chen, Tung-Chieh, et al. "NTUplace: A ratio partitioning based placement algorithm for large-scale mixed-size designs." ISPD 2005
- Chen, Tung-Chieh, et al. "NTUplace3: An analytical placer for large-scale mixed-size designs with preplaced blocks and density constraints." IEEE TCAD 2008.
- Hsu, Meng-Kai, et al. "NTUplace4h: A novel routability-driven placement algorithm for hierarchical mixed-size circuit designs." IEEE TCAD 2014
- Huang, Chau-Chin, et al. "NTUplace4dr: a detailed-routing-driven placer for mixed-size circuit designs with technology and region constraints." IEEE TCAD 2017

Density Penalty

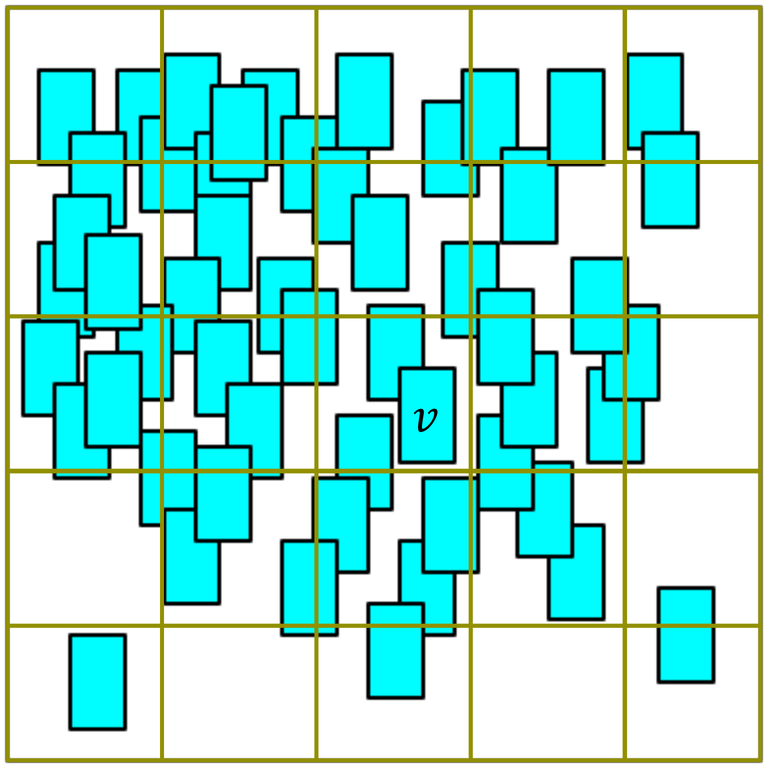
- Potential function for standard cells
  - $P_x(b,v)$ and $P_y(b,v)$ are the overlap functions between bin b and cell v
  - $D_b(\boldsymbol{x},\boldsymbol{y}) = \sum_{v \in V} P_x(b,v) P_y(b,v)$



Non-smooth
Non-convex

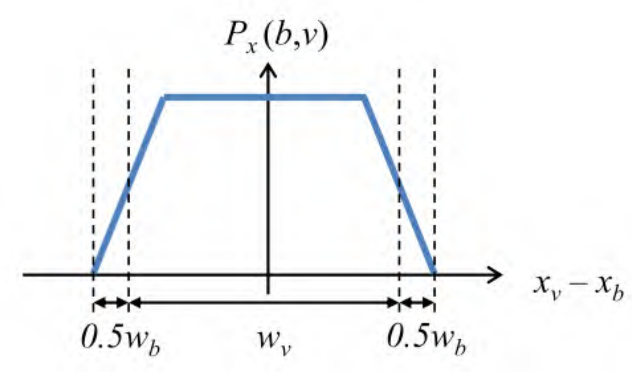


Bell-shape smoothing

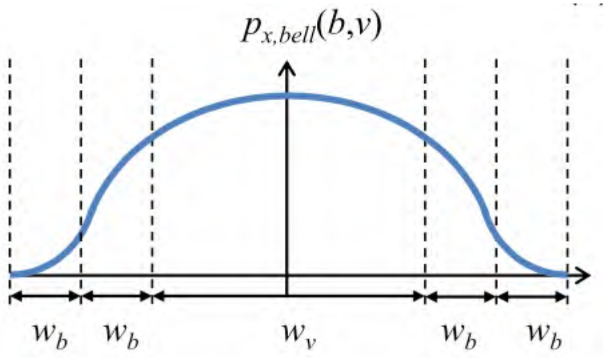


Density Penalty

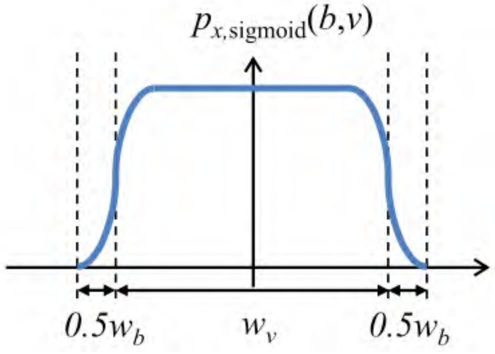
- Potential function for standard cells
  - $P_x(b, v)$ and $P_y(b, v)$ are the overlap functions between bin b and cell v
  - $D_b(\boldsymbol{x}, \boldsymbol{y}) = \sum_{v \in V} P_x(b, v) P_y(b, v)$



Non-smooth
Non-convex



Bell-shape smoothing



Sigmoid smoothing
[NTUplace4h, TCAD2014]

If $d_x \le \frac{w_v}{2} + w_b$,

$$\hat{P}_x(b, v) = 1 - a d_x^2$$

If $\frac{w_v}{2} + w_b \le d_x \le \frac{w_v}{2} + 2w_b$,

$$\hat{P}_x(b, v) = b\left(d_x - \frac{w_v}{2} - 2w_b\right)^2$$

Otherwise,

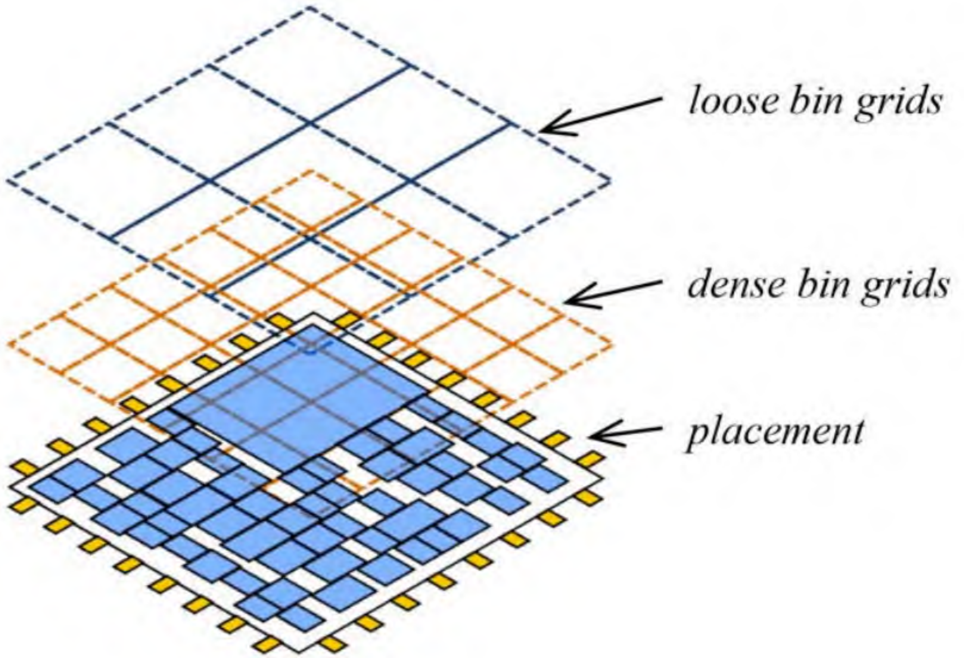
$$\hat{P}_x(b, v) = 0$$

Density Penalty

- Potential function for standard cells
  - Smoothed potential function
  - $\widehat{D_b}(\boldsymbol{x},\boldsymbol{y}) = \sum_{v\in V} \widehat{P_x}(b,v)\widehat{P_y}(b,v)$
- $\min_{\boldsymbol{x},\boldsymbol{y}} \; WL(\boldsymbol{x},\boldsymbol{y}) + \lambda D(\boldsymbol{x},\boldsymbol{y})$

$$\lambda \sum_{b} \left(\widehat{D_b}(\boldsymbol{x},\boldsymbol{y}) - t_d\right)^2$$

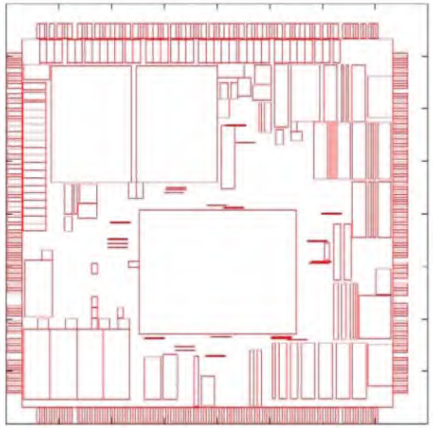
- Challenges
  - Gradient only has local view
  - Need multi-level bins



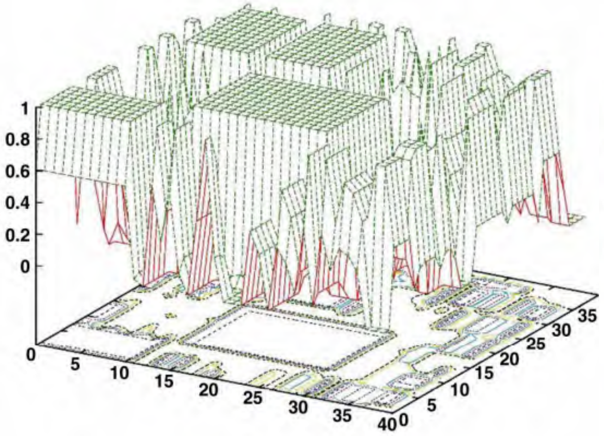
Multi-level bins

Density Penalty – Fixed Macros are Different

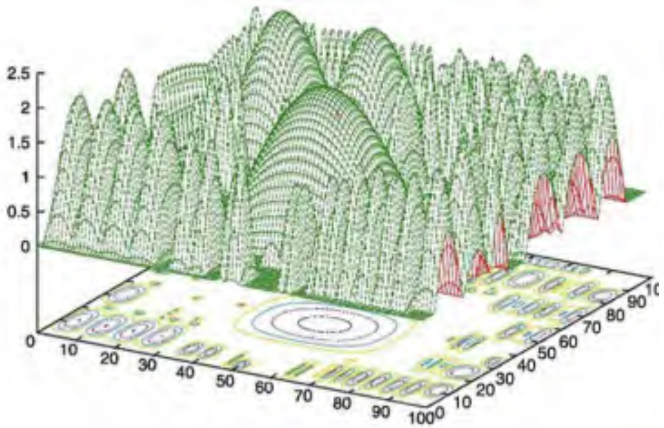
- Potential function for fixed macros
  - Bell-shape smoothing works well for standard cells
  - For fixed macros, $P'(x,y) = G(x,y) * P(x,y)$



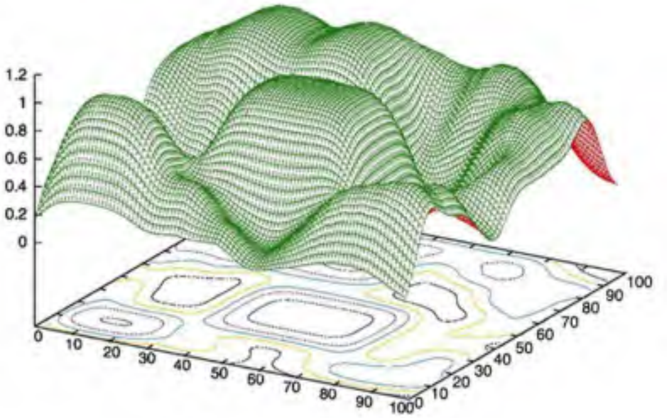
ISPD2005 adaptec2



Exact potential $P(x,y)$



Bell-shape smoothing



Gaussian smoothing $P'(x,y)$

Nonlinear Placement – ePlace

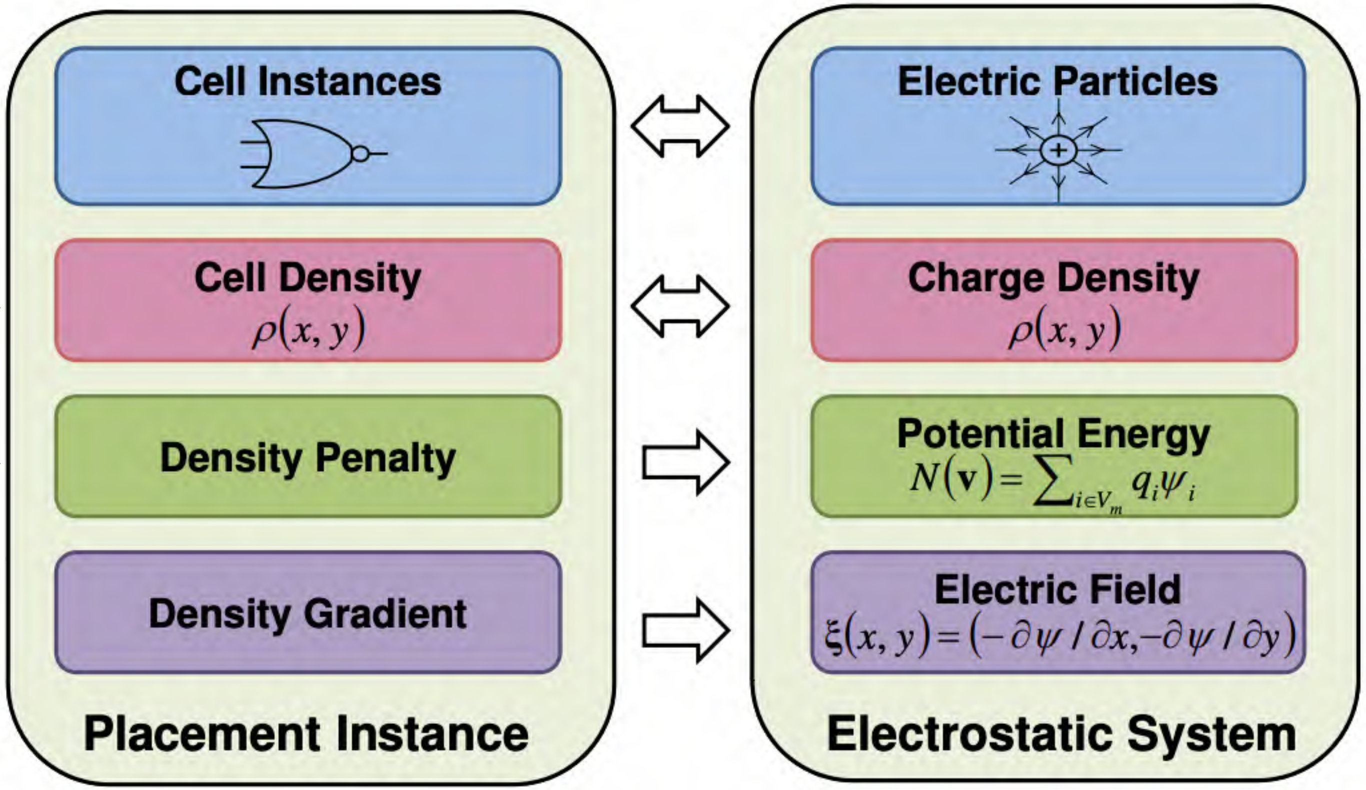
- http://vlsi-cuda.ucsd.edu/~ljw/ePlace/
- Lu, Jingwei, et al. "ePlace: Electrostatics-based placement using fast fourier transform and Nesterov's method." ACM TODAES 2015.
- Cheng, Chung-Kuan, et al. "RePlAce: Advancing solution quality and routability validation in global placement." IEEE TCAD 2018.
- Lin, Yibo, et al. "DREAMPlace: Deep learning toolkit-enabled gpu acceleration for modern vlsi placement." IEEE TCAD 2020. (DAC 2019 Best Paper Award)

Electric Potential

- $\min_{\boldsymbol{x},\boldsymbol{y}} \; WL(\boldsymbol{x},\boldsymbol{y}) + \lambda\, D(\boldsymbol{x},\boldsymbol{y})$

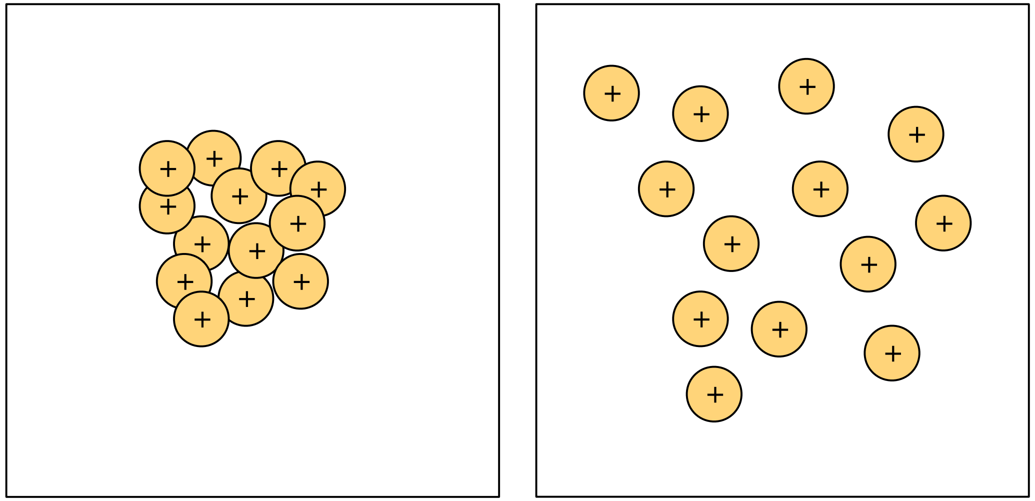
Exact density

$D(\boldsymbol{x},\boldsymbol{y})$



Electrostatic System

- Isolated electrostatic system
  - Balanced distribution ⟺ minimum potential energy
- If we can minimize the potential energy, then cells are spread out
- To consider $t_d < 1$
  - $s.t.\ \ d_b(\boldsymbol{x}, \boldsymbol{y}) \le t_d, \forall b \in Bins$
  - Insert fillers: dummy cells filling the area
  - $area_{fillers} + area_{cells} = area_{placeable} \times t_d$
  - Fillers have no connections



Poisson's Equation for Electrostatic System

$$\begin{cases} \nabla \cdot \nabla \psi(x,y) = -\rho(x,y), \\ \hat{\mathbf{n}} \cdot \nabla \psi(x,y) = \mathbf{0},\ (x,y) \in \partial R, \\ \underbrace{\iint_R \rho(x,y)}_{\text{Total charge}} = \underbrace{\iint_R \psi(x,y)}_{\text{Total energy}} = 0. \end{cases}$$

Solution $\Rightarrow$

$$a_{u,v} = \frac{1}{m^2} \sum_{x=0}^{m-1} \sum_{y=0}^{m-1} \rho(x,y) \cos(w_u x) \cos(w_v y).$$

$$\rho_{DCT}(x,y) = \sum_{u=0}^{m-1} \sum_{v=0}^{m-1} a_{u,v} \cos(w_u x) \cos(w_v y),$$

$$\psi_{DCT}(x,y) = \sum_{u=0}^{m-1} \sum_{v=0}^{m-1} \frac{a_{u,v}}{w_u^2 + w_v^2} \cos(w_u x) \cos(w_v y),$$

$$\begin{cases} \xi_{X_{DSCT}} = \sum_u \sum_v \frac{a_{u,v} w_u}{w_u^2 + w_v^2} \sin(w_u x) \cos(w_v y), \\ \xi_{Y_{DCST}} = \sum_u \sum_v \frac{a_{u,v} w_v}{w_u^2 + w_v^2} \cos(w_u x) \sin(w_v y). \end{cases}$$

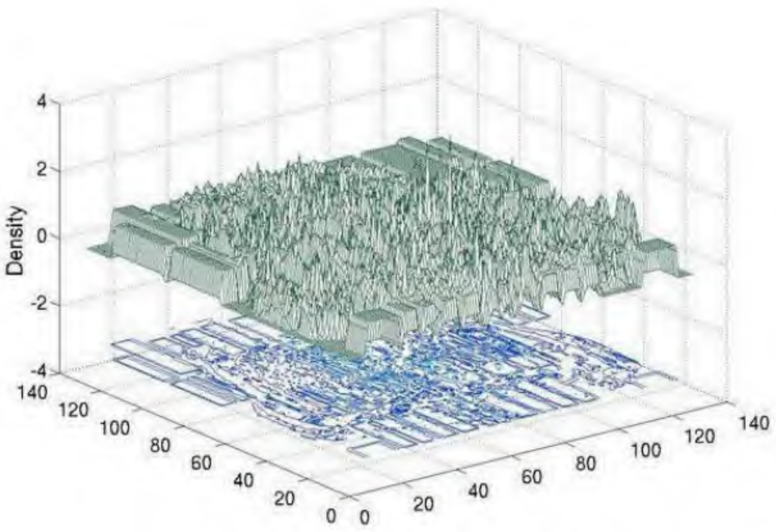
To remove DC component

$a_{0,0} = 0$

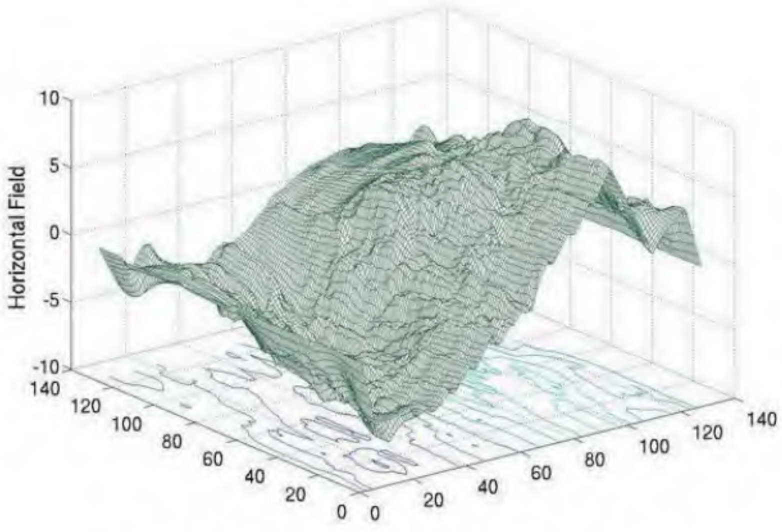
Zero-frequency component

In forms of DCT and DST

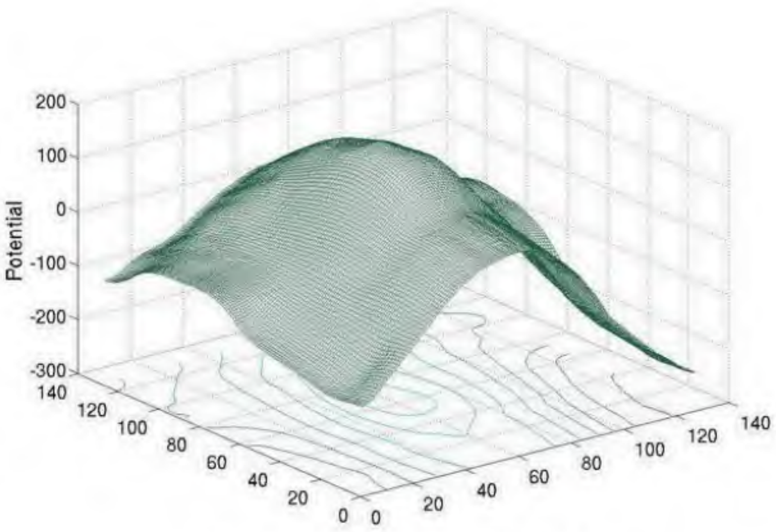
Electric Potential



(a) Electric density.



(b) Horizontal electric field



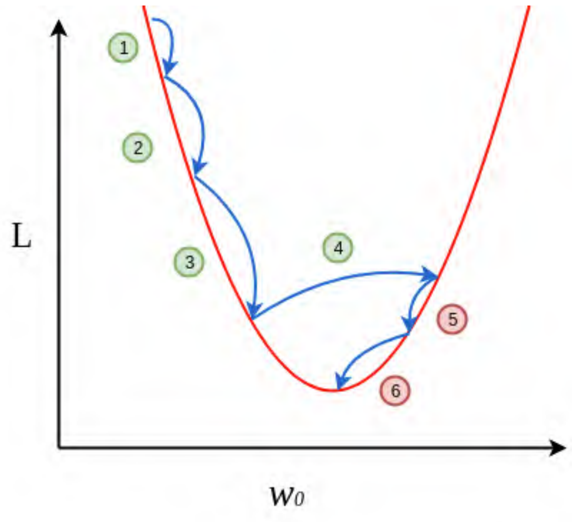
(c) Electric potential.

无约束可微优化问题

- 优化目标

$$\min_{x\in R^n} f(x) = WL(\boldsymbol{x},\boldsymbol{y}) + \lambda D(\boldsymbol{x},\boldsymbol{y})$$

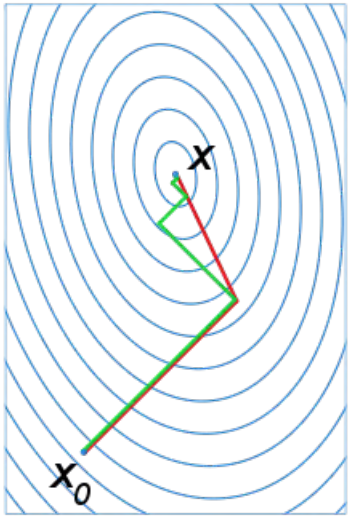
- 梯度下降
  - 线搜索：$x^{k+1} = x^k + \alpha_k d^k$
  - 先确定下降方向：负梯度、牛顿方向、拟牛顿方向等
  - 按某种准则搜索步长



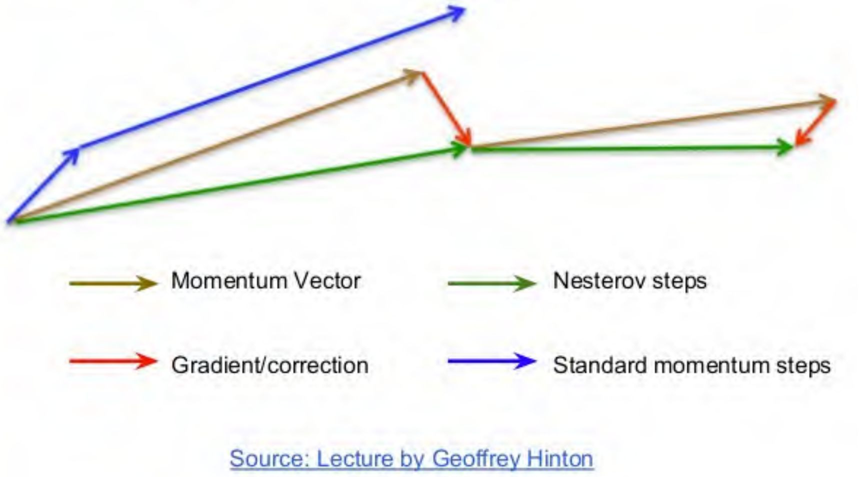
- 盲人下山
  - 求解$f(x)$的最小值点如同盲人下山，无法一眼望知谷底，而是
  - 首先确定下一步改往哪个方向行走
  - 再确定沿着该方向行走多远后停下以便选取下一个下山方向

下降方向选取

- 线搜索类算法的数学表述： $x^{k+1} = x^k + \alpha_k d^k$
- d^k为迭代点x^k处的搜索方向
- α_k为相应的步长
- 下降方向的要求：$(d^k)^T \nabla f(x^k) < 0$



共轭梯度法



Nesterov加速梯度法

步长α_k选取

- 精确线搜索算法
  - 首先构造一元辅助函数：$\phi(\alpha) = f(x^k + \alpha d^k)$
  - 其中，$\alpha > 0$是该辅助函数的自变量
- 线搜索的目标是选取合适的α_k使得$\phi(\alpha_k)$尽可能小，这要求
  - α_k应该使得f充分下降
  - 不应在寻找α_k上花费过多的计算量
- 一个自然的想法是寻找α_k使得：$\alpha_k = \mathrm{argmin}_{\alpha>0}\, \phi(\alpha)$
  - 即α_k为最佳步长；这种线搜索算法称为精确线搜索算法
  - 最佳步长求解计算量大，实际中应用较少

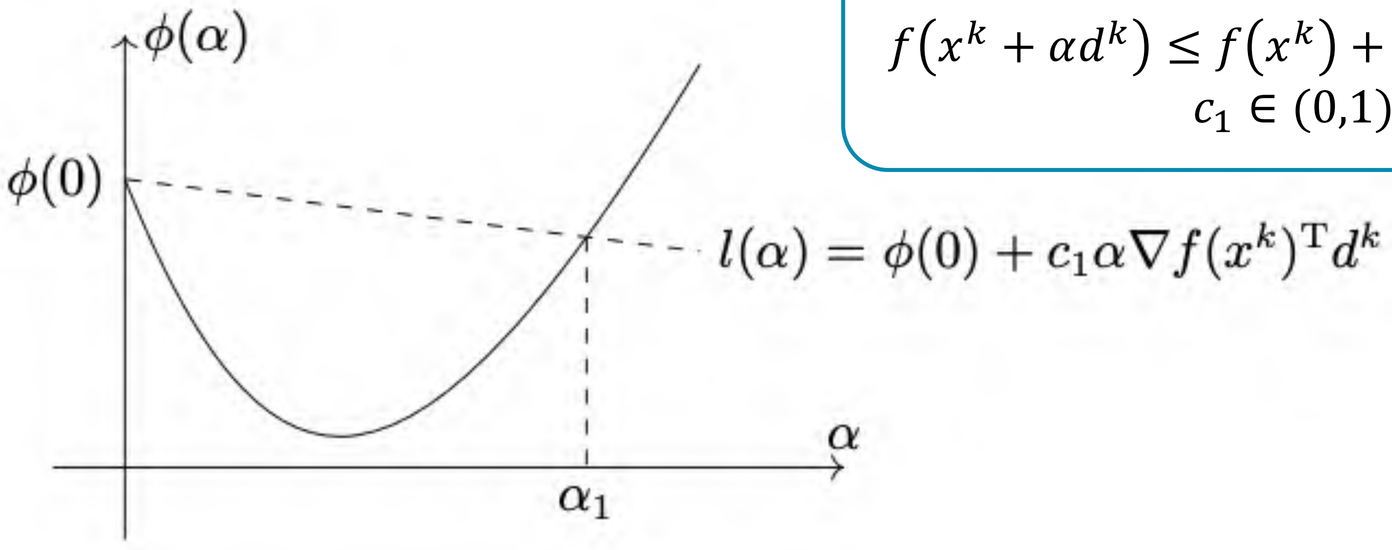
步长α_k选取

- 精确线搜索算法
  - 首先构造一元辅助函数：$\phi(\alpha) = f(x^k + \alpha d^k)$
  - 其中，$\alpha > 0$是该辅助函数的自变量
- 线搜索的目标是选取合适的α_k使得$\phi(\alpha_k)$尽可能小，这要求
  - α_k应该使得f充分下降
  - 不应在寻找α_k上花费过多的计算量

Armijo准则

$$f\left(x^k + \alpha d^k\right) \le f\left(x^k\right) + c_1 \alpha \nabla f\left(x^k\right)^T d^k$$

$$c_1 \in (0,1)$$



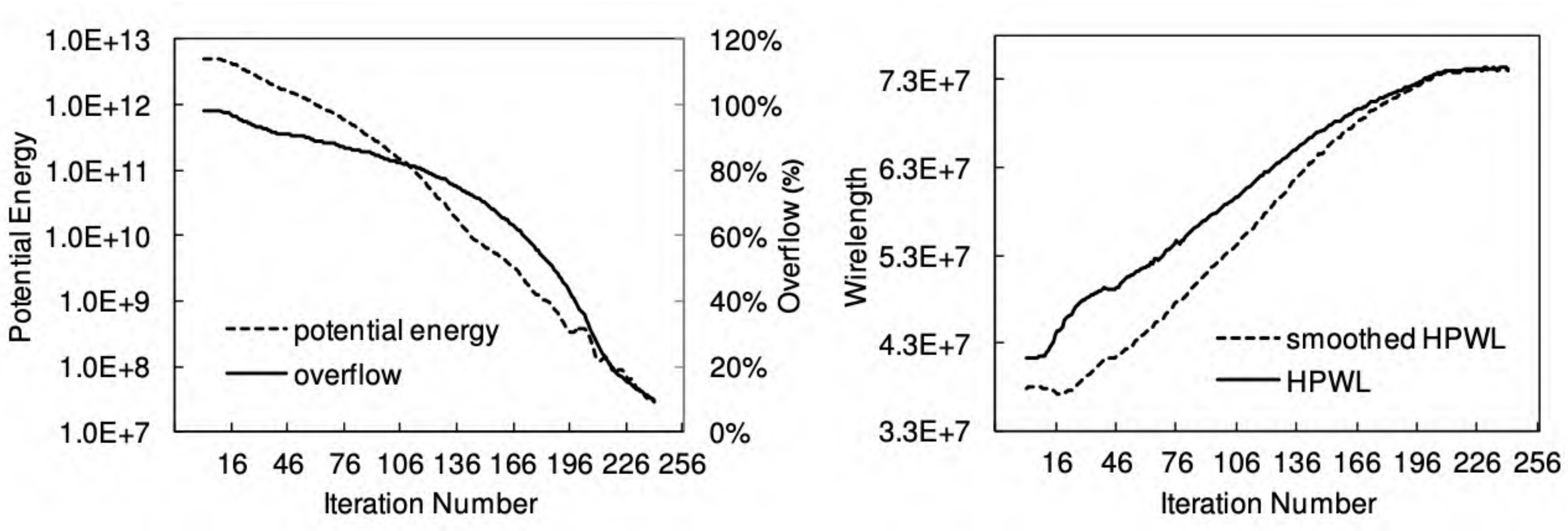
步长α_k选取

- 精确线搜索算法
  - 首先构造一元辅助函数：$\phi(\alpha) = f(x^k + \alpha d^k)$
  - 其中，$\alpha > 0$是该辅助函数的自变量
- 线搜索的目标是选取合适的α_k使得$\phi(\alpha_k)$尽可能小，这要求
  - α_k应该使得f充分下降
  - 不应在寻找α_k上花费过多的计算量

Algorithm 1 线搜索回退法

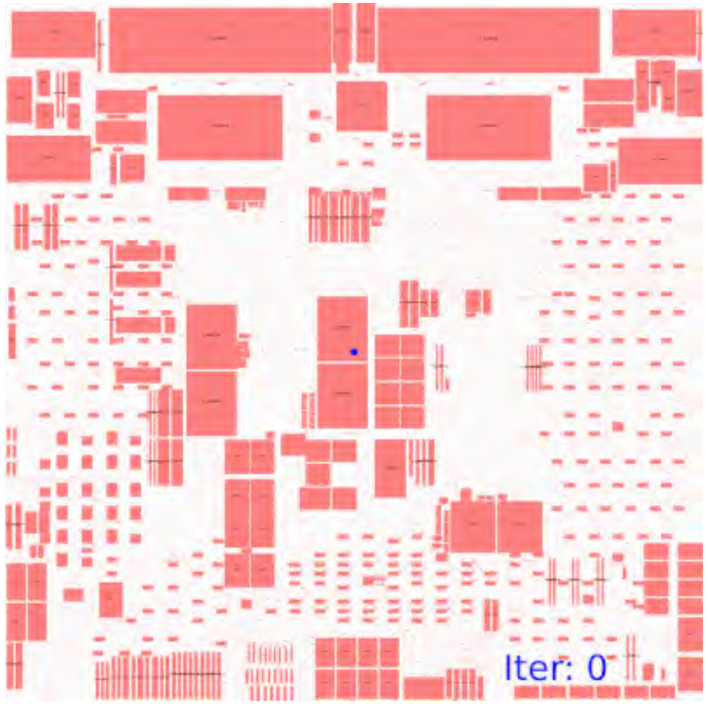
1: 选择初始步长$\hat{\alpha}$, 参数$\gamma, c \in (0,1)$. 初始化$\alpha \leftarrow \hat{\alpha}$.
2: **while** $f(x^k + \alpha d^k) > f(x^k) + c\alpha \nabla f(x^k)^{\mathrm{T}} d^k$ **do**
3: 　令$\alpha \leftarrow \gamma\alpha$.
4: **end while**
5: 输出$\alpha_k = \alpha$.

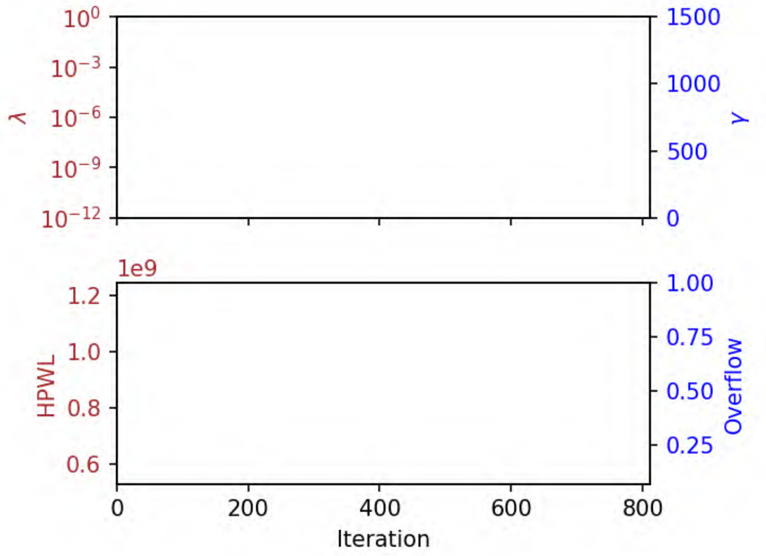
Gradient Descent Iterations



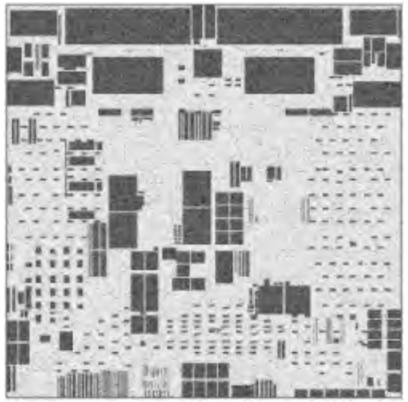
Bigblue4 (2M-Cell Design)

DREAMPlace impl. of the ePlace algorithm

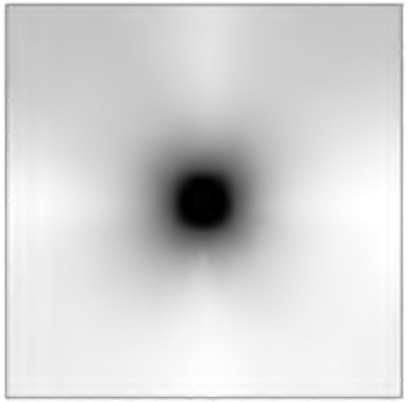




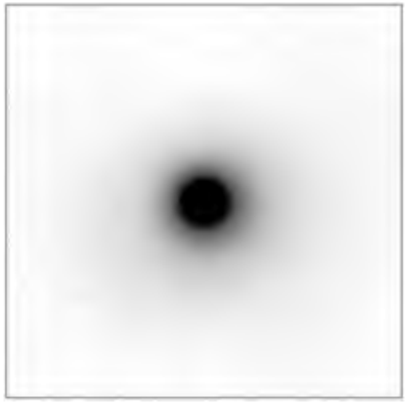
Placement Metrics



Density Map



Potential Map



Field Map

Nonlinear Placement – Summary

- Nonlinear placement
  - Wirelength smoothing: LSE and WA
  - Density potential (NTUplace)
  - Electric potential (ePlace)
- Open-source tools
  - DREAMPlace
  - https://github.com/limbo018/DREAMPlace
  - RePlAce
  - https://github.com/The-OpenROAD-Project/RePlAce

课后思考

- 北京大学计划筹建新校区，请你帮忙规划新校区的建筑分布
  - 已知宿舍楼50幢、食堂10个、教学楼10幢、绿地20块
  - 假设各建筑形状为矩形，且已知大小
  - 假设新校区形状如下图
  - 尝试利用Quadratic Placement或Nonlinear Placement框架设计算法流程求解建筑位置
  - **要求1**：宿舍楼、教学楼和食堂尽可能靠近，且不同宿舍楼、教学楼尽量靠近不同的食堂
  - **要求2**：不同建筑不能重叠
  - **要求3**：写出优化目标、约束条件以及算法流程，并解释设计理由

Summary of Global Placement

<1970-1980s	1980s-1990s	1990s-2010s			>2010s	
Partitioning	Simulated Annealing	Min-Cut (Multi-level)	Analytic		Analytic	
			Quadratic	Nonlinear	Quadratic	Nonlinear
Breuer	**Timberwolf** VPR	FengShui	**GORDIAN**	APlace	POLAR	**ePlace** RePlAce
Dunlop & Kernighan	**Dragon**	**Capo**	BonnPlace	Naylor Synopsis	**SimPL** ComPLx	DREAMPlace
Quadratic Assignment		Capo +Rooster	mFar	**NTUplace**	MAPLE	
Cadence QPlace			**Kraftwerk**	mPL6		
			FastPlace			
			Warp3			

Typical Placement Flow