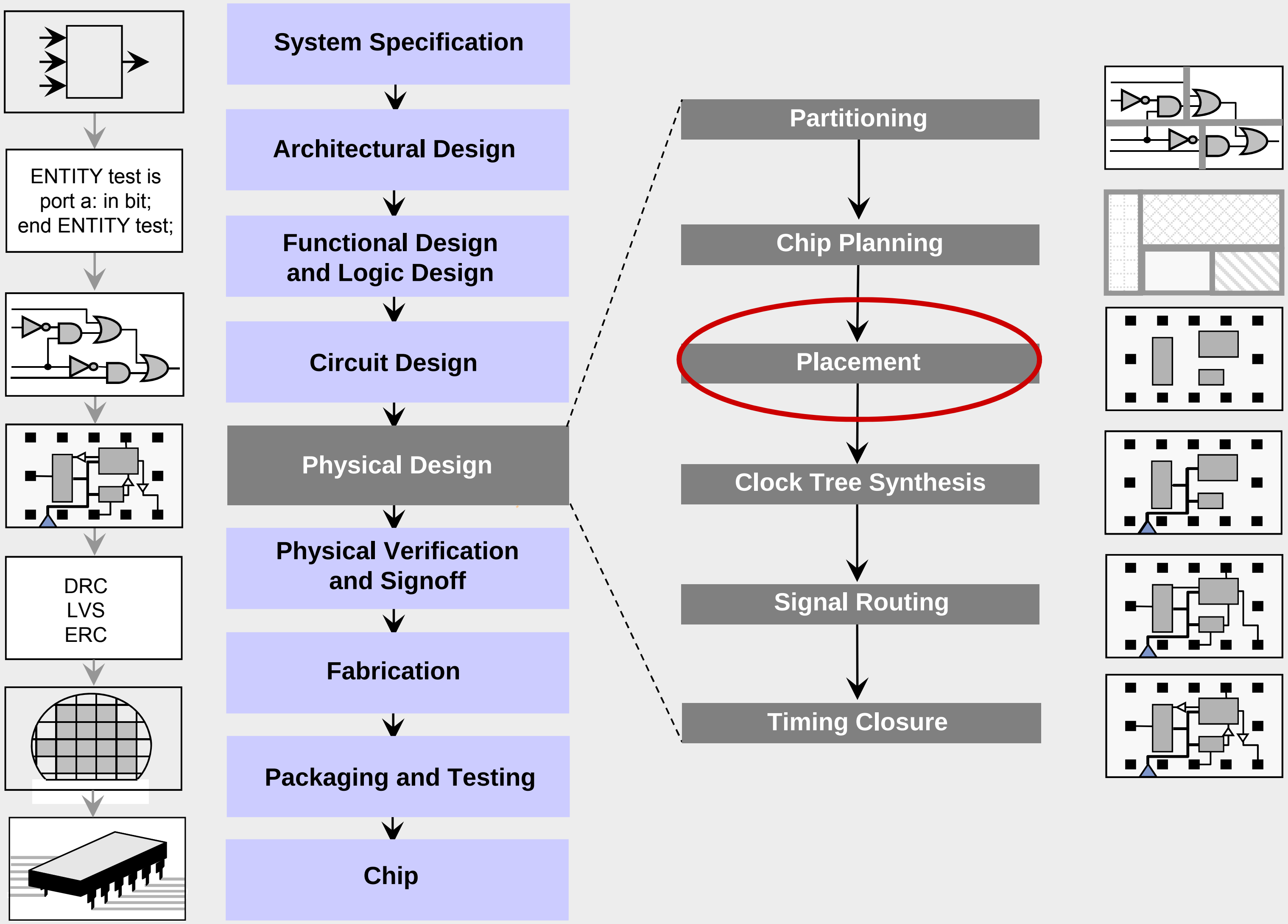
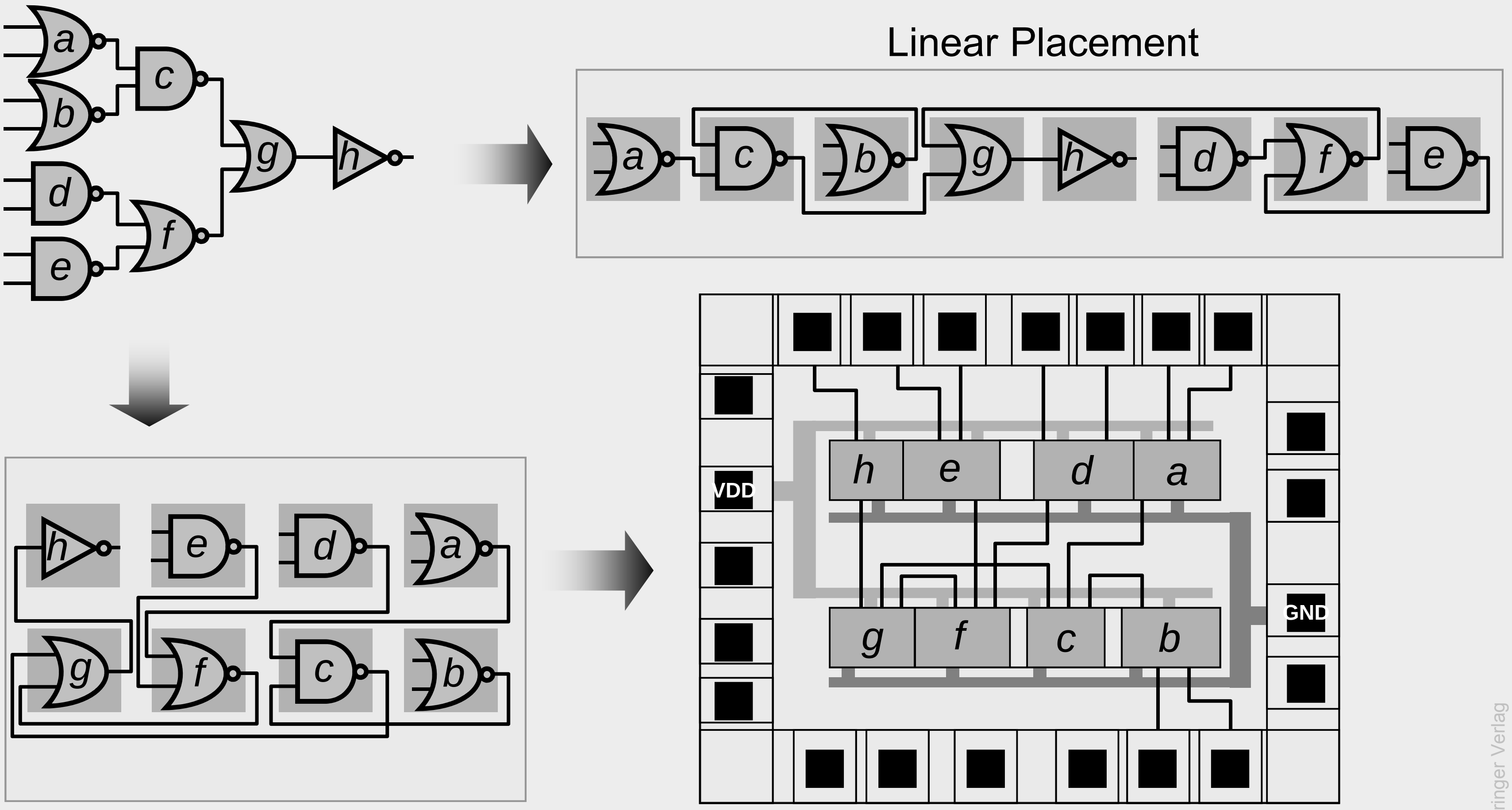
ECEN 687
VLSI Design Automation

Lecture 15 *Placement*

Introduction



Introduction

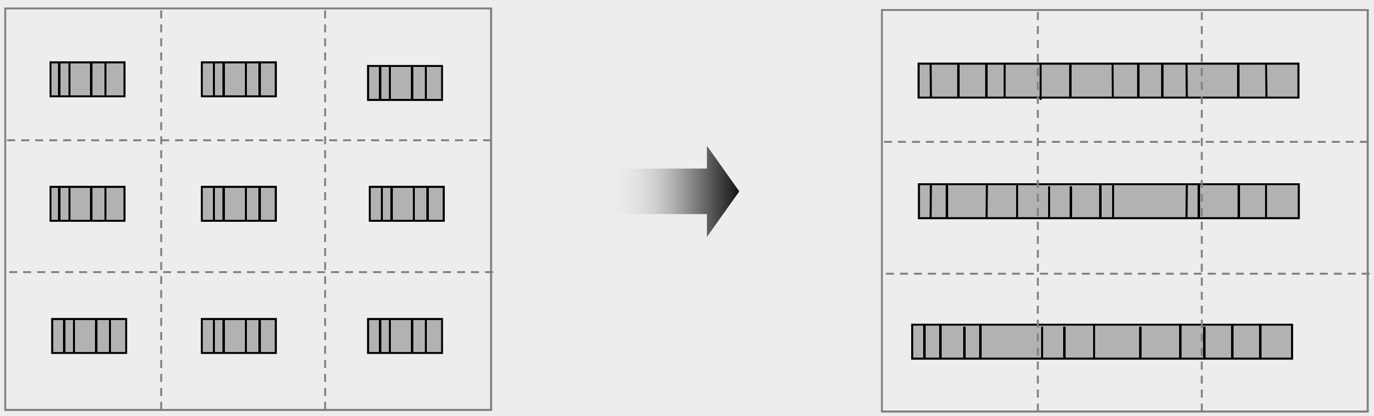


2D Placement

Placement and Routing with Standard Cells

Global Placement

Detailed Placement



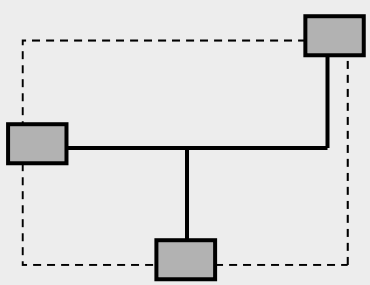
Optimization Objectives

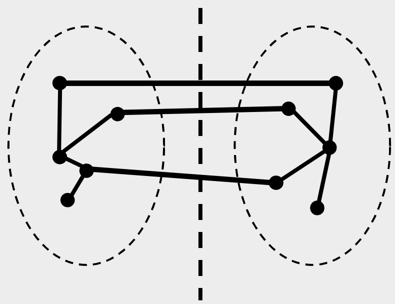
Total Wirelength

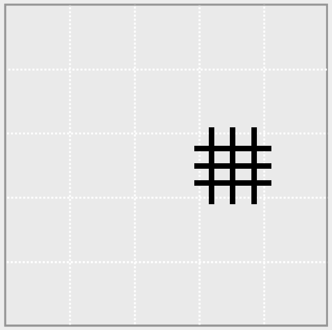
Number of Cut Nets

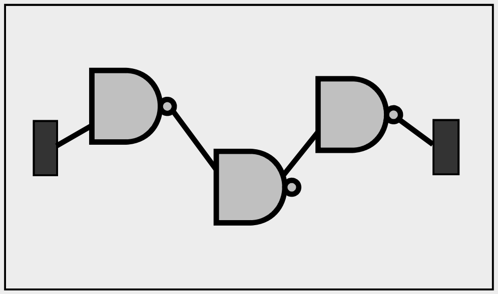
Wire Congestion

Signal Delay

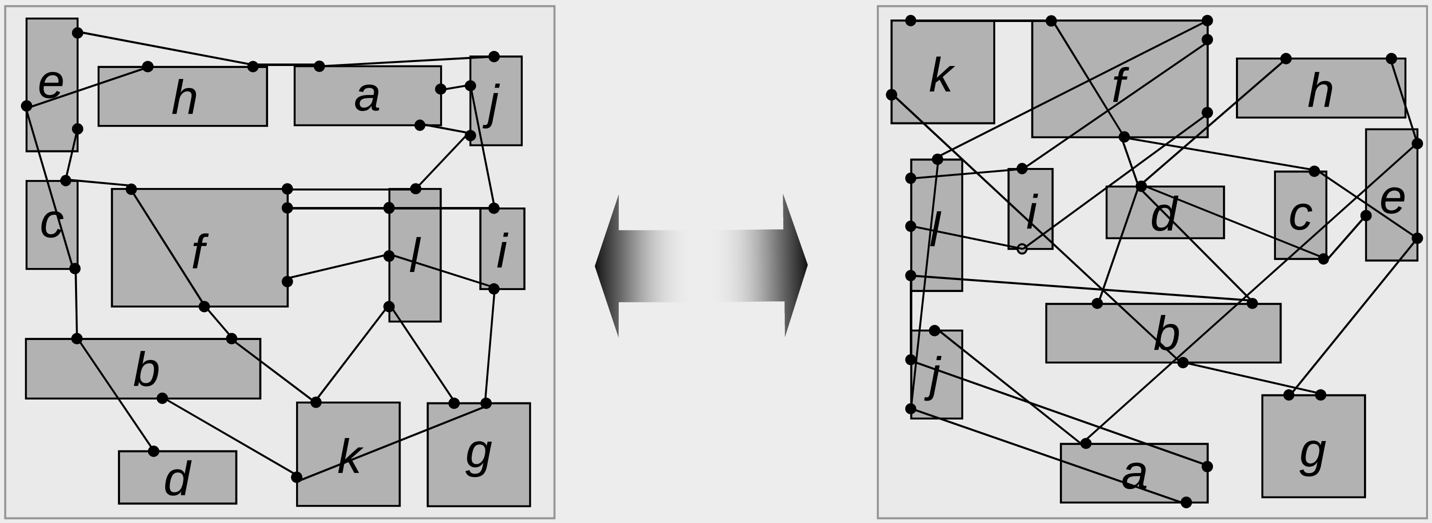






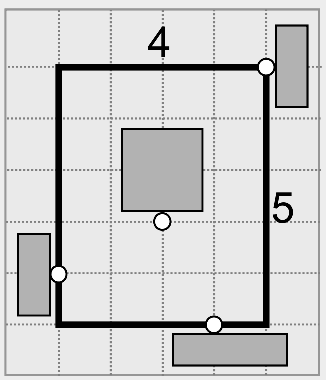


Optimization Objectives – Total Wirelength

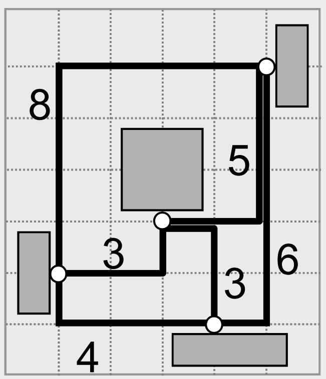


Wirelength estimation for a given placement

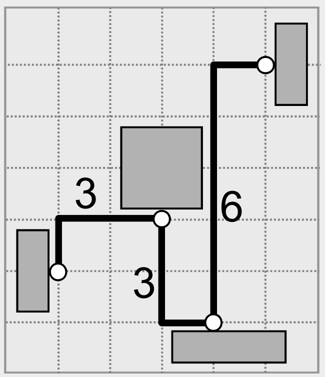
Half-perimeter wirelength (HPWL)	Complete graph (clique)	Monotone chain	Star model



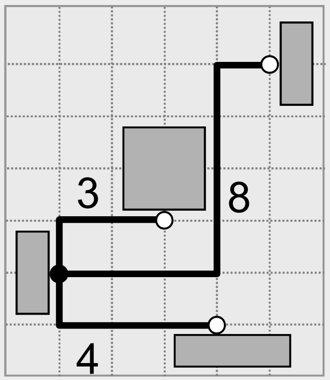
HPWL = 9



Clique Length = $(2/p)\Sigma_{e \in \text{clique}} d_M(e)$ = 14.5



Chain Length = 12

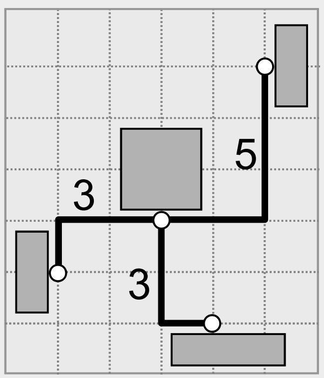


Star Length = 15

 Courtesy of I. Markov

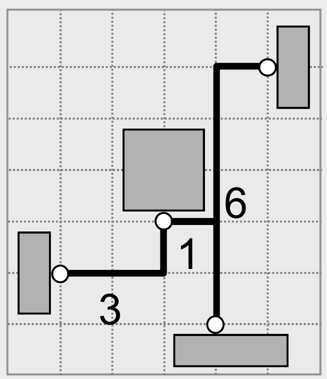
Wirelength estimation for a given placement (cont‘d.)

Rectilinear minimum spanning tree (RMST)



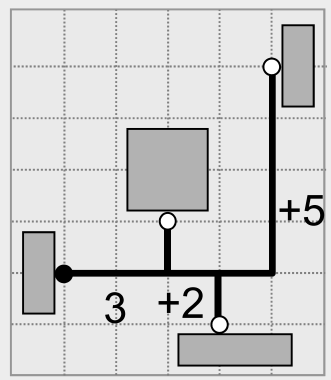
RMST Length = 11

Rectilinear Steiner minimum tree (RSMT)



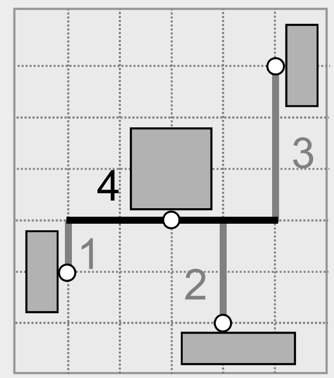
RSMT Length = 10

Rectilinear Steiner arborescence model (RSA)



RSA Length = 10

Single-trunk Steiner tree (RSMT)



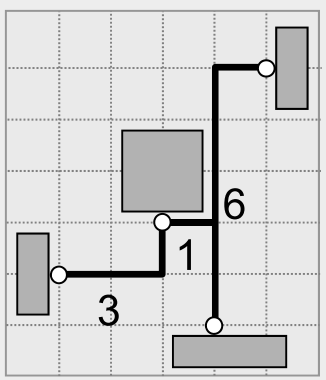
STST Length = 10

Courtesy of I. Markov

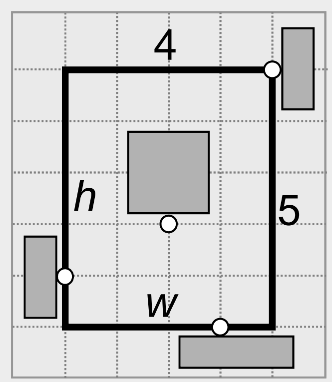
Wirelength estimation for a given placement (cont'd.)

Preferred method: Half-perimeter wirelength (HPWL)

- Fast (order of magnitude faster than RSMT)
- Equal to length of RSMT for 2- and 3-pin nets
- Margin of error for real circuits approx. 8% [Chu, ICCAD 04]



RSMT Length = 10



HPWL = 9

$$L_{\mathrm{HPWL}} = w + h$$

Total wirelength with net weights (weighted wirelength)

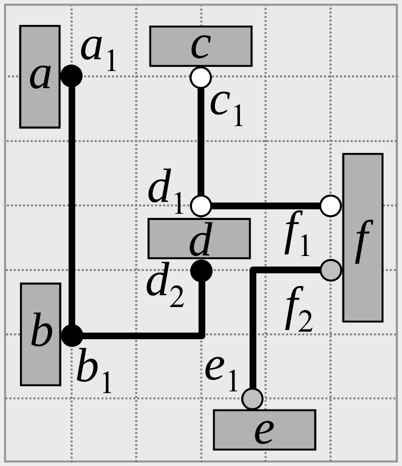
- For a placement P, an estimate of total weighted wirelength is

$$L(P) = \sum_{net \in P} w(net) \cdot L(net)$$

where $w(net)$ is the weight of net, and $L(net)$ is the estimated wirelength of net.

- Example:

Nets	Weights
$N_1 = (a_1, b_1, d_2)$	$w(N_1) = 2$
$N_2 = (c_1, d_1, f_1)$	$w(N_2) = 4$
$N_3 = (e_1, f_2)$	$w(N_3) = 1$



$$L(P) = \sum_{net \in P} w(net) \cdot L(net) = 2 \cdot 7 + 4 \cdot 4 + 1 \cdot 3 = 33$$

Optimization Objectives – Number of Cut Nets

Cut sizes of a placement

- To improve total wirelength of a placement *P*, separately calculate the number of crossings of global vertical and horizontal cutlines, and minimize

$$L(P) = \sum_{v \in V_P} \psi_P(v) + \sum_{h \in H_P} \psi_P(h)$$

where $\Psi_P(cut)$ be the set of nets cut by a cutline *cut*

Optimization Objectives – Number of Cut Nets

Cut sizes of a placement

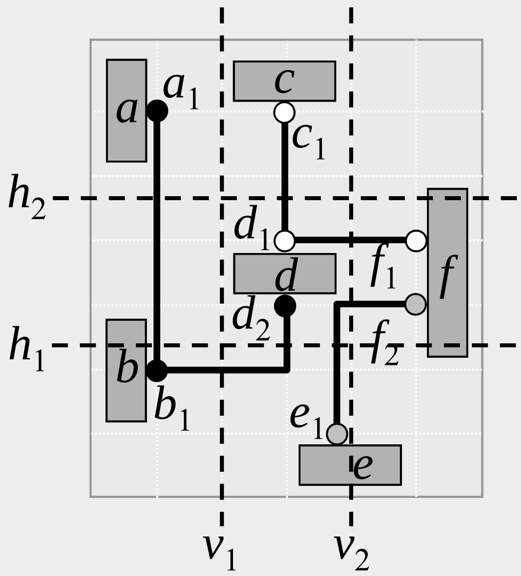
- Example:

  Nets

  $N_1 = (a_1, b_1, d_2)$

  $N_2 = (c_1, d_1, f_1)$

  $N_3 = (e_1, f_2)$



- Cut values for each global cutline

  $\psi_P(v_1) = 1$ $\psi_P(v_2) = 2$

  $\psi_P(h_1) = 3$ $\psi_P(h_2) = 2$

- Total number of crossings in P

  $\psi_P(v_1) + \psi_P(v_2) + \psi_P(h_1) + \psi_P(h_2) = 1 + 2 + 3 + 2 = 8$

- Cut sizes

  $X(P) = \max(\psi_P(v_1), \psi_P(v_2)) = \max(1,2) = 2$

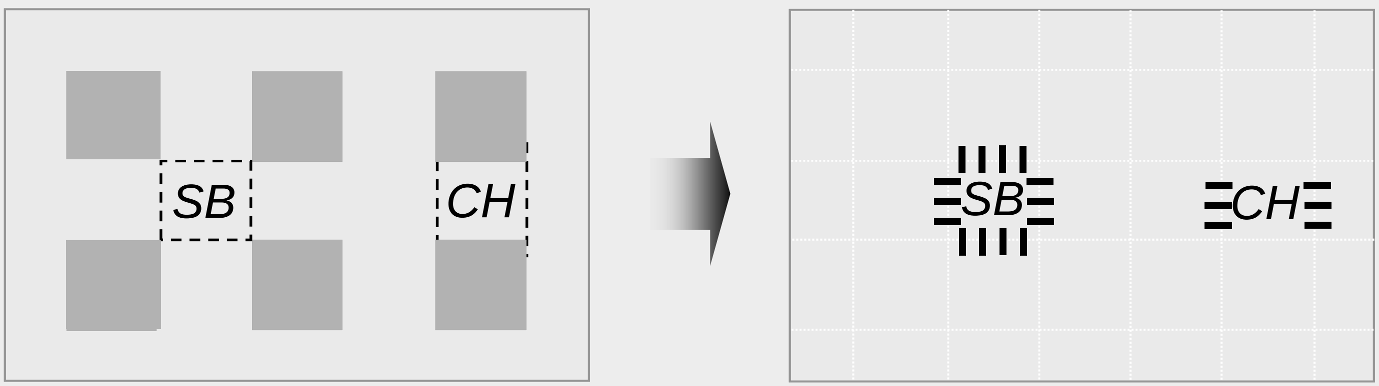
  $Y(P) = \max(\psi_P(h_1), \psi_P(h_2)) = \max(3,2) = 3$

Courtesy of I. Markov

Optimization Objectives – Wire Congestion

Routing congestion of a placement

- Ratio of demand for routing tracks to the supply of available routing tracks
- Estimated by the number of nets that pass through the boundaries of individual routing regions



Wire capacities

Optimization Objectives – Wire Congestion

Routing congestion of a placement

- Formally, the local wire density $\varphi_P(e)$ of an edge e between two neighboring grid cells is

$$\varphi_P(e) = \frac{\eta_P(e)}{\sigma_P(e)}$$

where $\eta_P(e)$ is the estimated number of nets that cross e and
$\sigma_P(e)$ is the maximum number of nets that can cross e

- If $\varphi_P(e) > 1$, then too many nets are estimated to cross e, making P more likely to be unroutable.
- The wire density of P is $$\Phi(P) = \max_{e \in E}\left(\varphi_P(e)\right)$$

where E is the set of all edges

- If $\Phi(P) \leq 1$, then the design is estimated to be fully routable, otherwise routing will need to detour some nets through less-congested edges

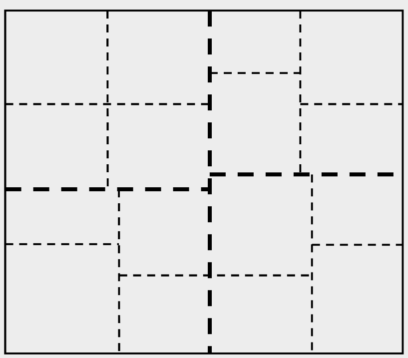
Global Placement

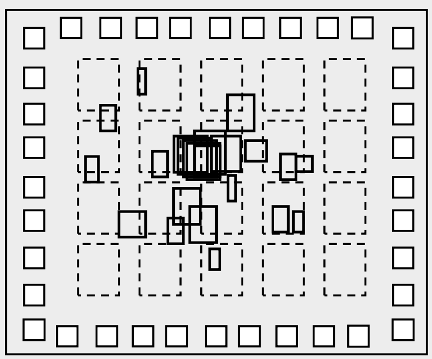
- Partitioning-based algorithms:
  - The netlist and the layout are divided into smaller sub-netlists and sub-regions, respectively
  - Process is repeated until each sub-netlist and sub-region is small enough to be handled optimally
  - Detailed placement often performed by optimal solvers, facilitating a natural transition from global placement to detailed placement
  - Example: min-cut placement
- Analytic techniques:
  - Model the placement problem using an objective (cost) function, which can be optimized via numerical analysis
  - Examples: quadratic placement and force-directed placement
- Stochastic algorithms:
  - Randomized moves that allow hill-climbing are used to optimize the cost function
  - Example: simulated annealing

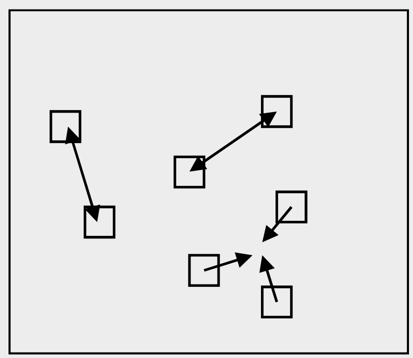
Global Placement

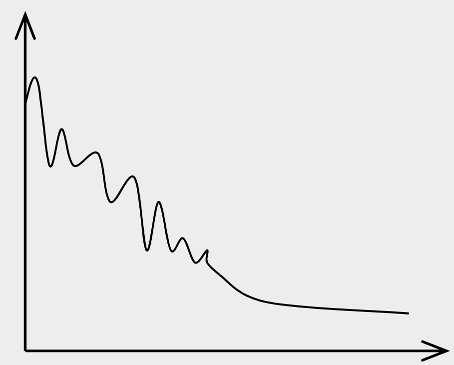
Partitioning-based | Analytic | Stochastic

- **Min-cut placement** (Partitioning-based)
- **Quadratic placement** (Analytic)
- **Force-directed placement** (Analytic)
- **Simulated annealing** (Stochastic)









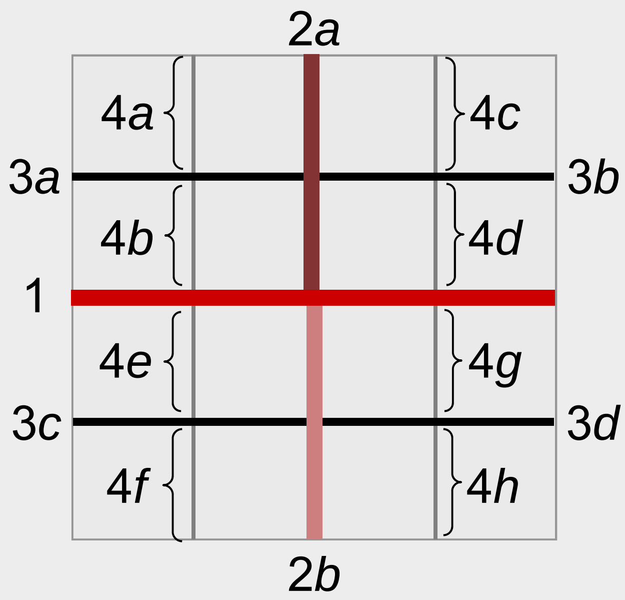
Courtesy of I. Markov

Min-Cut Placement

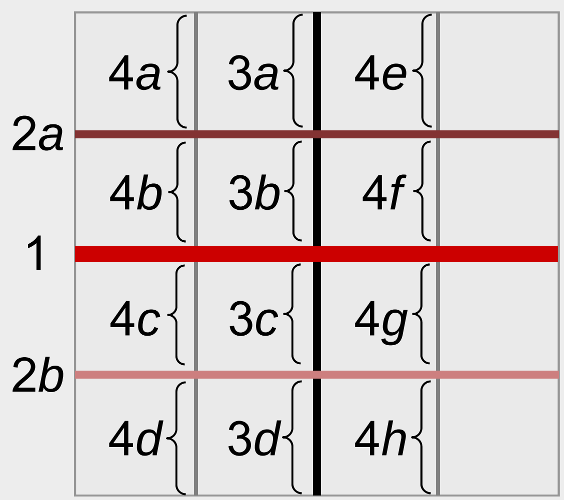
- Uses partitioning algorithms to divide (1) the netlist and (2) the layout region into smaller sub-netlists and sub-regions
- Conceptually, each sub-region is assigned a portion of the original netlist
- Each cut heuristically minimizes the number of cut nets using, for example,
  - Kernighan-Lin (KL) algorithm
  - Fiduccia-Mattheyses (FM) algorithm

Courtesy of I. Markov

Alternating cutline directions



Repeating cutline directions



Courtesy of I. Markov

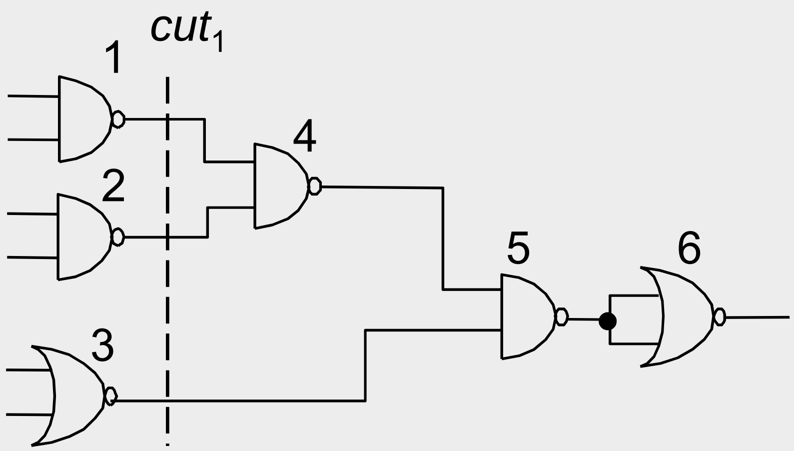
Min-Cut Placement

Input: netlist *Netlist*, layout area *LA*, minimum number of cells per region *cells_min*
Output: placement *P*

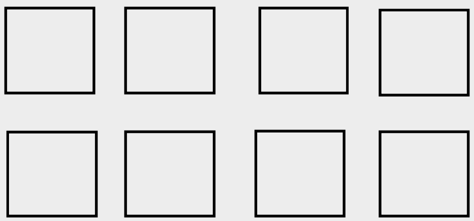
```
P = Ø
regions = ASSIGN(Netlist,LA)                        // assign netlist to layout area
while (regions != Ø)                                // while regions still not placed
    region = FIRST_ELEMENT(regions)                 // first element in regions
    REMOVE(regions, region)                         // remove first element of regions
    if (region contains more than cell_min cells)
        (sr1,sr2) = BISECT(region)                  // divide region into two subregions
                                                    //   sr1 and sr2, obtaining the sub-
                                                    //   netlists and sub-areas
        ADD_TO_END(regions,sr1)                     // add sr1 to the end of regions
        ADD_TO_END(regions,sr2)                     // add sr2 to the end of regions
    else
        PLACE(region)                               // place region
        ADD(P,region)                               // add region to P
```

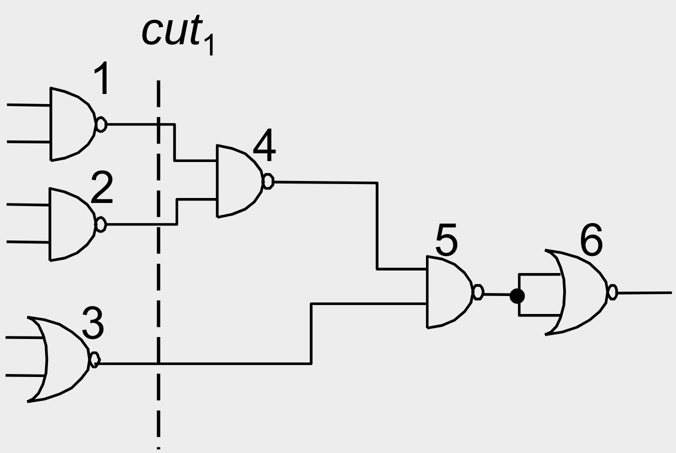
Min-Cut Placement – Example

Given:

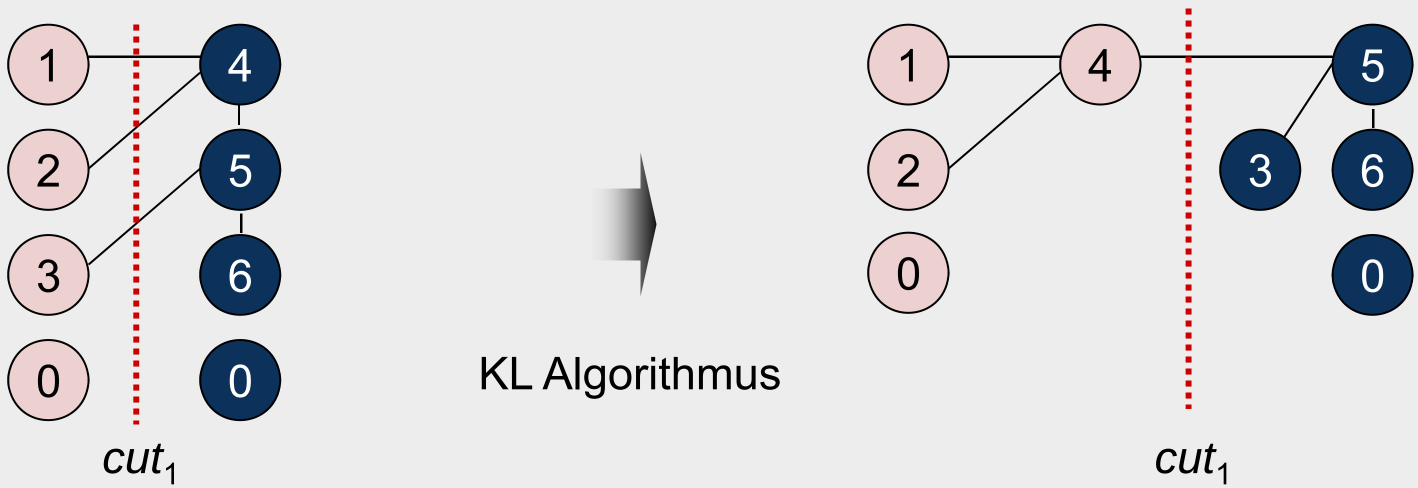


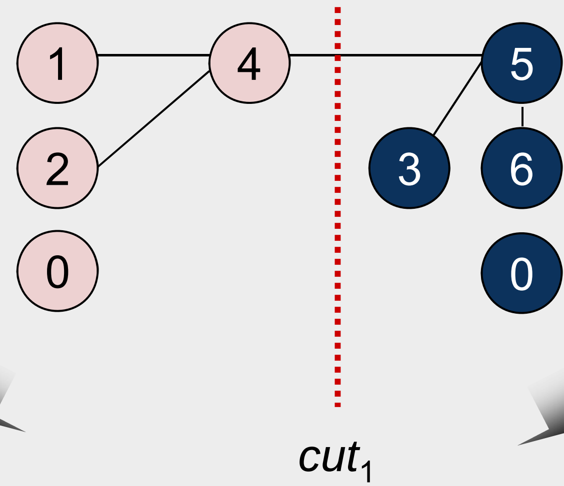
Task: 4 x 2 placement with minimum wirelength using alternative cutline directions and the KL algorithm





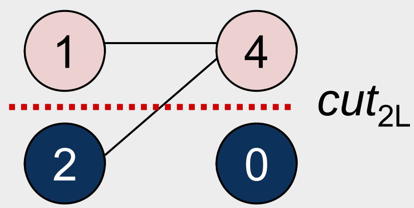
Vertical cut cut_1: L={1,2,3}, R={4,5,6}

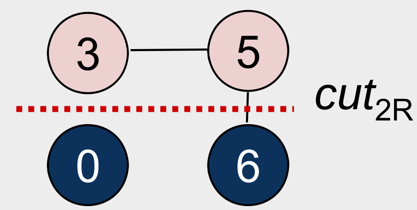


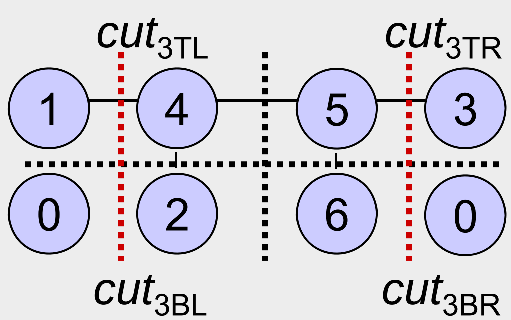


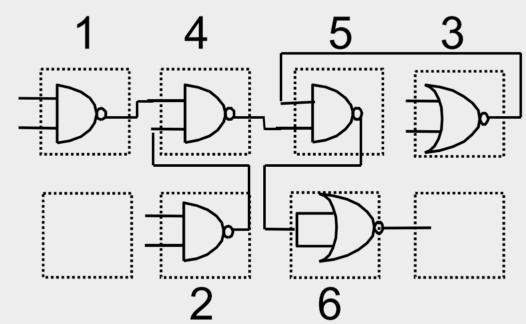
Horizontal cut cut_{2L}: T={1,4}, B={2,0}

Horizontal cut cut_{2R}: T={3,5}, B={6,0}



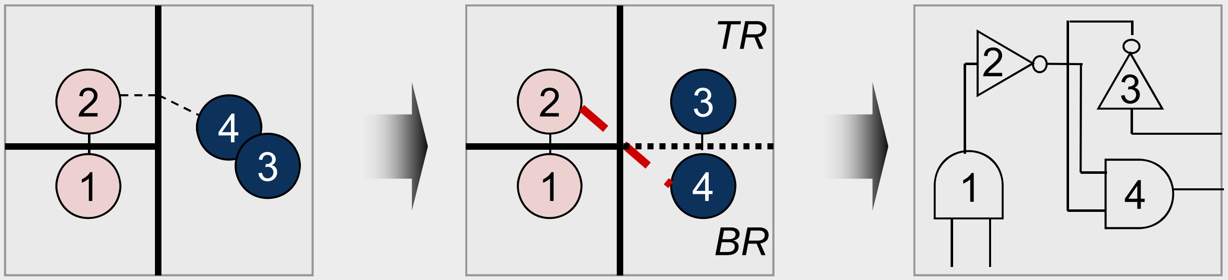




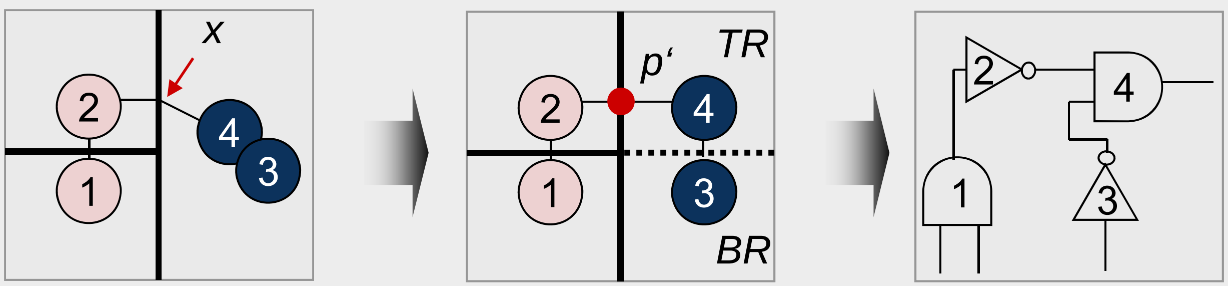


Courtesy of I. Markov

Min-Cut Placement – Terminal Propagation



- Terminal Propagation
  - External connections are represented by artificial connection points on the cutline
  - Dummy nodes in hypergraphs



Min-Cut Placement

- Advantages:
  - Reasonable fast
  - Objective function and be adjusted, e.g., to perform timing-driven placement
  - Hierarchical strategy applicable to large circuits
- Disadvantages:
  - Randomized, chaotic algorithms – small changes in input lead to large changes in output
  - Optimizing one cutline at a time may result in routing congestion elsewhere

Analytic Placement – Quadratic Placement

- Objective function is quadratic; sum of (weighted) squared Euclidean distance represents placement objective function

$$L(P) = \frac{1}{2} \sum_{i,j=1}^{n} c_{ij} \left(\left(x_i - x_j \right)^2 + \left(y_i - y_j \right)^2 \right)$$

where n is the total number of cells, and $c(i,j)$ is the connection cost between cells i and j.

- Only two-point-connections
- Minimize objective function by equating its derivative to zero which reduces to solving a system of linear equations

Analytic Placement – Quadratic Placement

$$L(P) = \frac{1}{2} \sum_{i,j=1}^{n} c_{ij} \left(\left(x_i - x_j \right)^2 + \left(y_i - y_j \right)^2 \right)$$

where n is the total number of cells, and $c(i,j)$ is the connection cost between cells i and j.

- Each dimension can be considered independently:

$$L_x(P) = \sum_{i=1,\, j=1}^{n} c(i,j)(x_i - x_j)^2 \qquad L_y(P) = \sum_{i=1,\, j=1}^{n} c(i,j)(y_i - y_j)^2$$

- Convex quadratic optimization problem: any local minimum solution is also a global minimum
- Optimal x- and y-coordinates can be found by setting the partial derivatives of $L_x(P)$ and $L_y(P)$ to zero

Analytic Placement – Quadratic Placement

$$L(P) = \frac{1}{2}\sum_{i,j=1}^{n} c_{ij}\left(\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2\right)$$

where n is the total number of cells, and $c(i,j)$ is the connection cost between cells i and j.

- Each dimension can be considered independently:

$$L_x(P) = \sum_{i=1,\, j=1}^{n} c(i,j)(x_i - x_j)^2 \qquad L_y(P) = \sum_{i=1,\, j=1}^{n} c(i,j)(y_i - y_j)^2$$

$$\frac{\partial L_x(P)}{\partial X} = AX - b_x = 0 \qquad \frac{\partial L_y(P)}{\partial Y} = AY - b_y = 0$$

where A is a matrix with $A[i][j] = -c(i,j)$ when $i \neq j$,
and $A[i][i]$ = the sum of incident connection weights of cell i.

X is a vector of all the x-coordinates of the non-fixed cells, and b_x is a vector with $b_x[i]$ = the sum of x-coordinates of all fixed cells attached to i.

Y is a vector of all the y-coordinates of the non-fixed cells, and b_y is a vector with $b_y[i]$ = the sum of y-coordinates of all fixed cells attached to i.

Analytic Placement – Quadratic Placement

$$L(P) = \frac{1}{2}\sum_{i,j=1}^{n} c_{ij}\left(\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2\right)$$

where n is the total number of cells, and $c(i,j)$ is the connection cost between cells i and j.

- Each dimension can be considered independently:

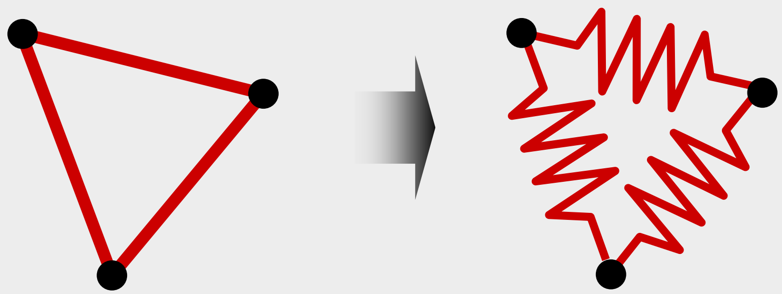
$$L_x(P) = \sum_{i=1,\, j=1}^{n} c(i,j)(x_i - x_j)^2 \qquad L_y(P) = \sum_{i=1,\, j=1}^{n} c(i,j)(y_i - y_j)^2$$

$$\frac{\partial L_x(P)}{\partial X} = AX - b_x = 0 \qquad \frac{\partial L_y(P)}{\partial Y} = AY - b_y = 0$$

- System of linear equations for which iterative numerical methods can be used to find a solution

Analytic Placement – Quadratic Placement

- Mechanical analogy: mass-spring system

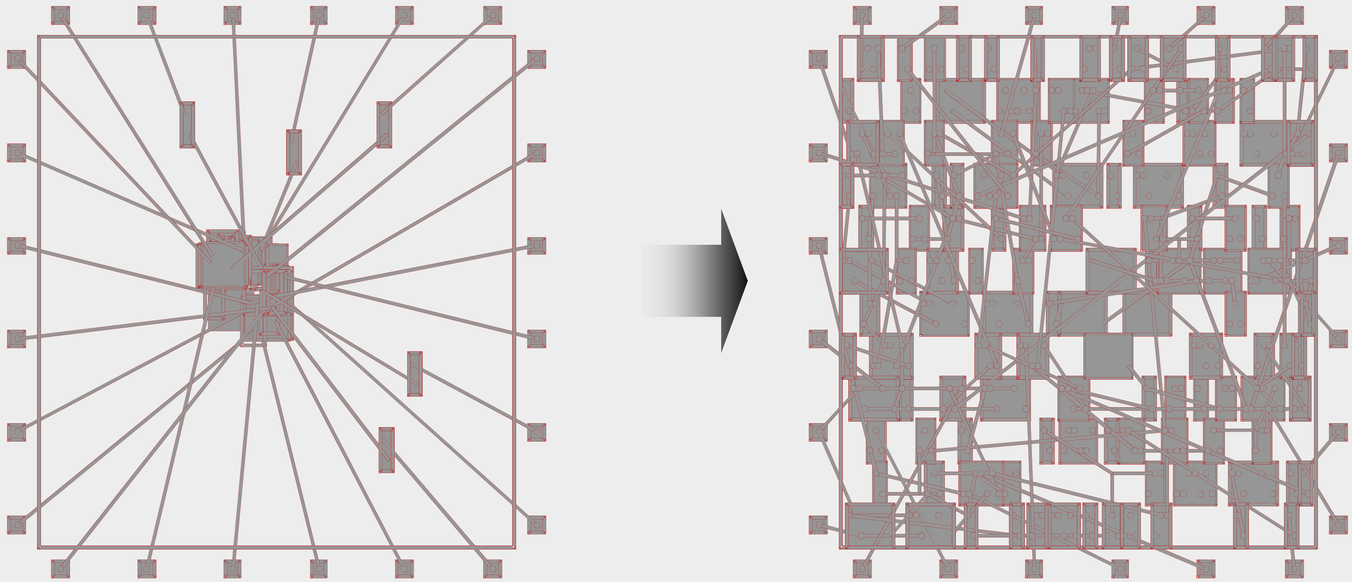


  – Squared Euclidean distance is proportional to the energy of a spring between these points
  – Quadratic objective function represents total energy of the spring system; for each movable object, the *x* (*y*) partial derivative represents the total force acting on that object
  – Setting the forces of the nets to zero, an equilibrium state is mathematically modeled that is characterized by zero forces acting on each movable object
  – At the end, all springs are in a force equilibrium with a minimal total spring energy; this equilibrium represents the minimal sum of squared wirelength

→ Result: many cell overlaps

Analytic Placement – Quadratic Placement

- Second stage of quadratic placers: cells are spread out to remove overlaps
- Methods:
  - Adding fake nets that pull cells away from dense regions toward anchors
  - Geometric sorting and scaling
  - Repulsion forces, etc.



Courtesy of I. Markov

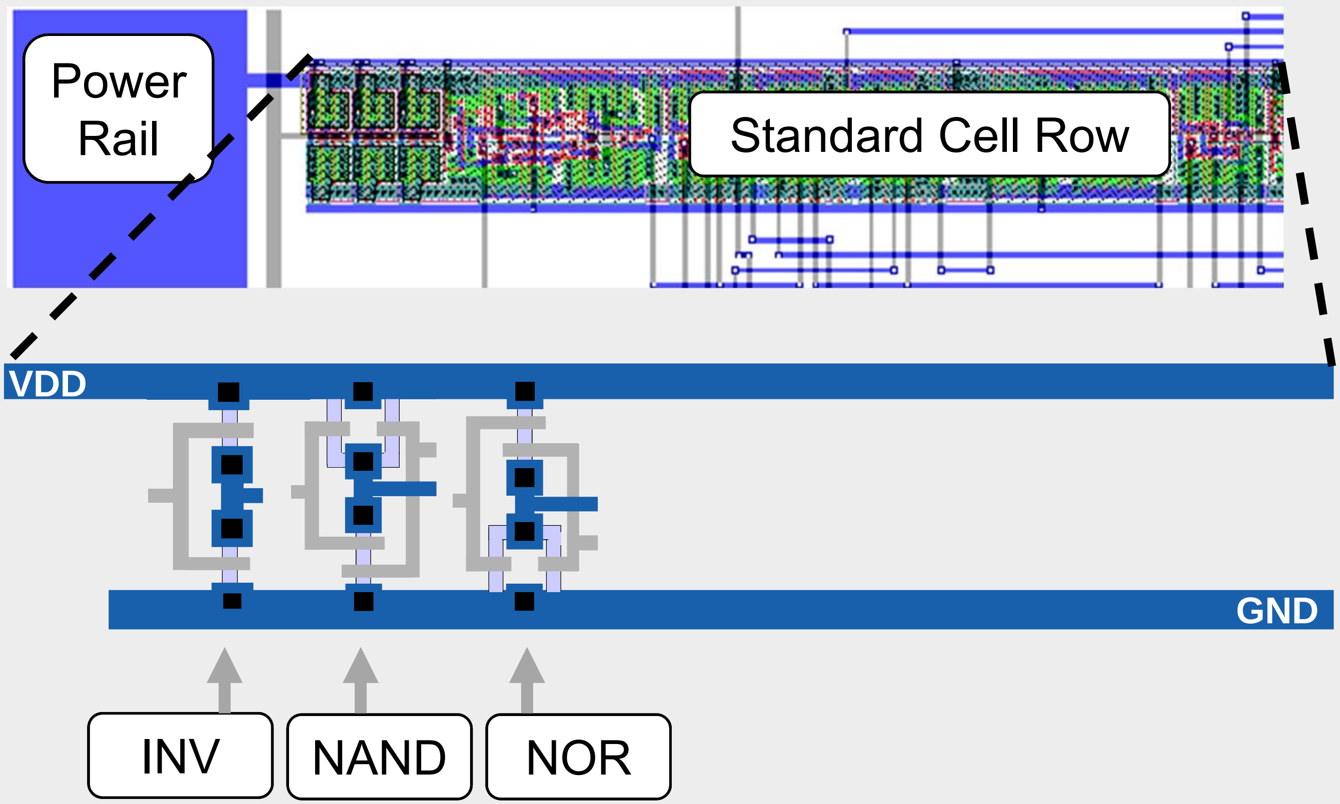
Analytic Placement – Quadratic Placement

- Advantages:
  - Captures the placement problem concisely in mathematical terms
  - Leverages efficient algorithms from numerical analysis and available software
  - Can be applied to large circuits without netlist clustering (flat)
  - Stability: small changes in the input do not lead to large changes in the output
- Disadvantages:
  - Connections to fixed objects are necessary: I/O pads, pins of fixed macros, etc.

Legalization and Detailed Placement

- Global placement must be legalized
  - Cell locations typically do not align with power rails
  - Small cell overlaps due to incremental changes, such as cell resizing or buffer insertion
- Legalization seeks to find legal, non-overlapping placements for all placeable modules
- Legalization can be improved by detailed placement techniques, such as
  - Swapping neighboring cells to reduce wirelength
  - Sliding cells to unused space
- Software implementations of legalization and detailed placement are often bundled

Legal positions of standard cells between VDD and GND rails



 Courtesy of I. Markov