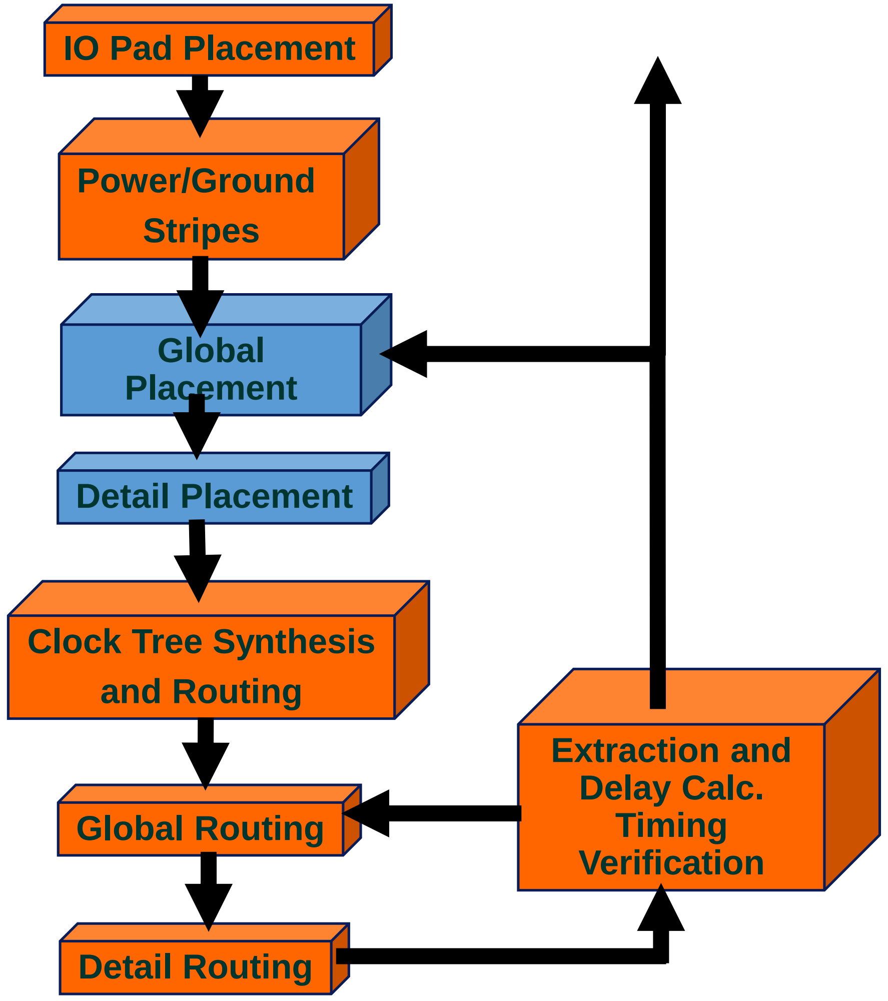
Placement

VLSI Design Flow and Physical Design Stage



Definitions:

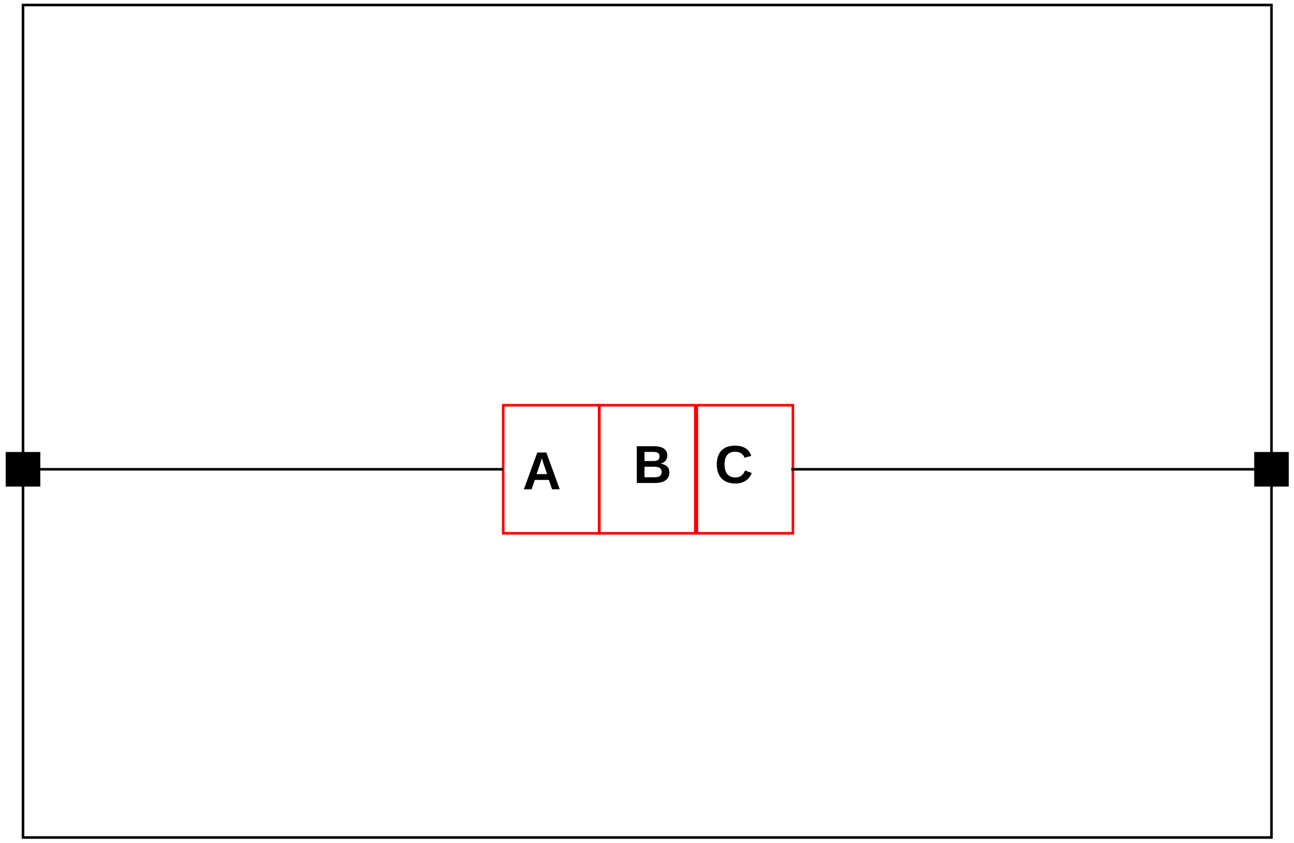
•Cell: a circuit component to be placed on the chip area.

- In placement, the functionality of the component is ignored.

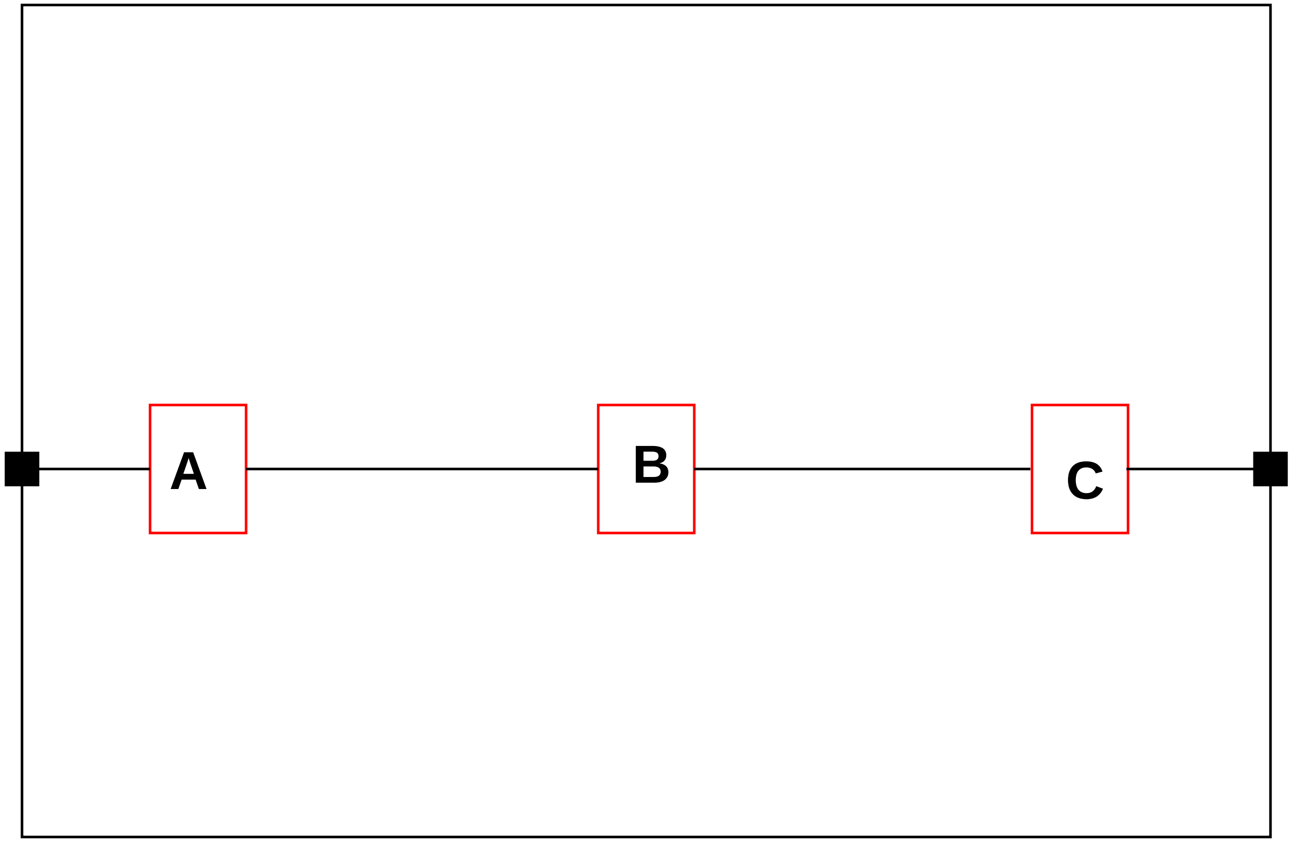
•Net: specifying a subset of terminals, to connect several cells.

•Netlist: a set of nets which contains the connectivity information of the circuit.

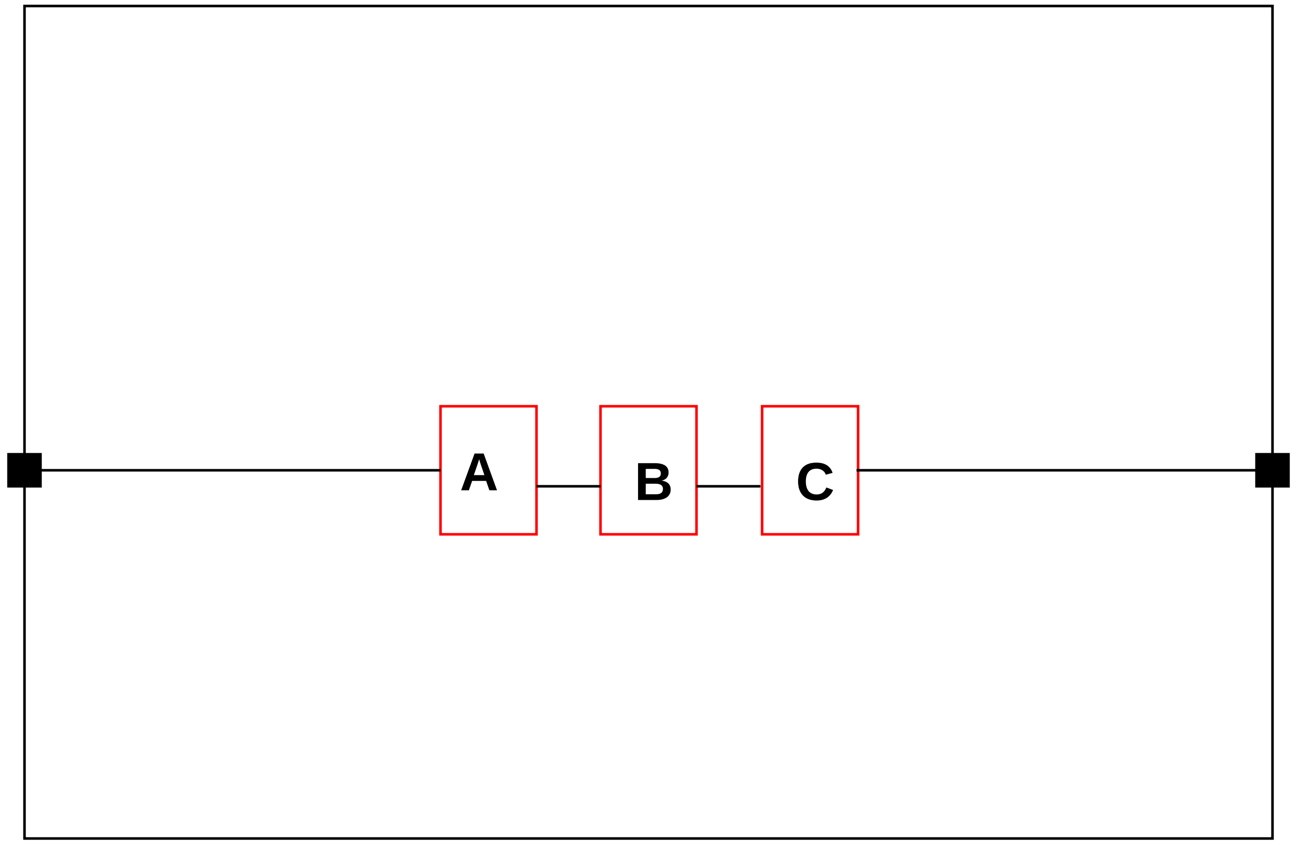
Optimal Relative Order:



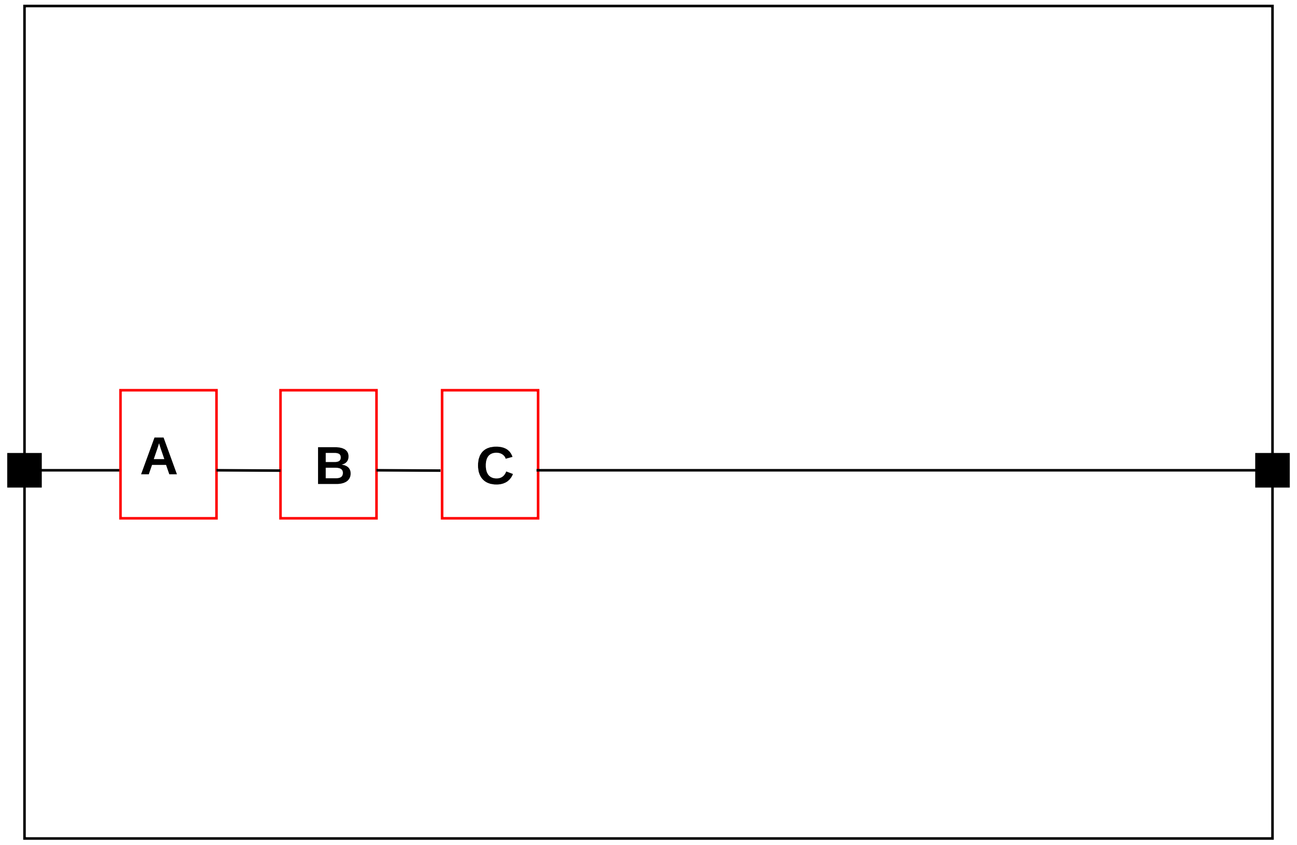
To spread …



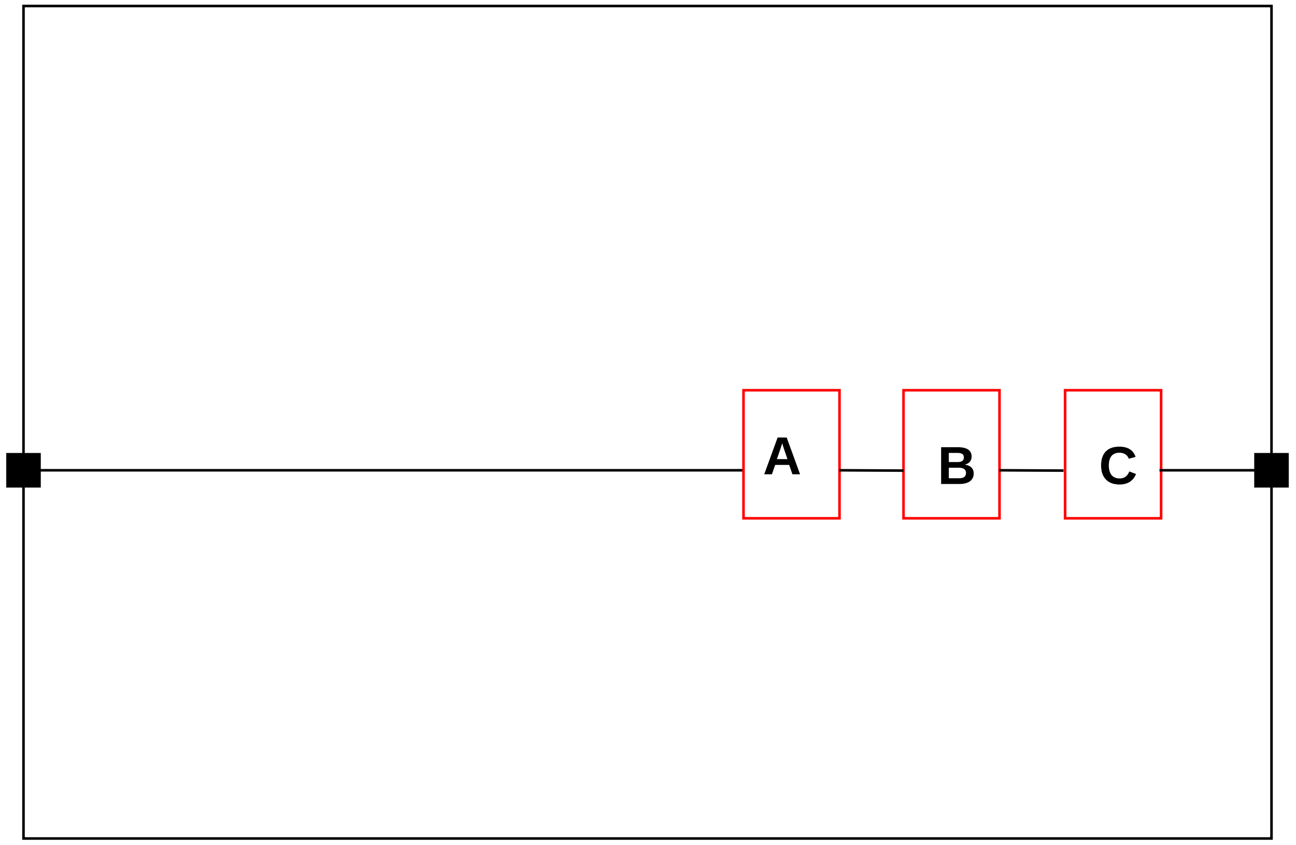
.. or not to spread



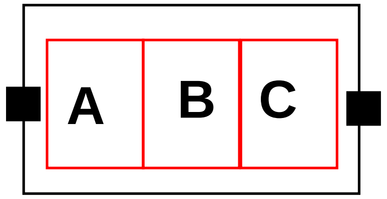
Place to the left



… or to the right

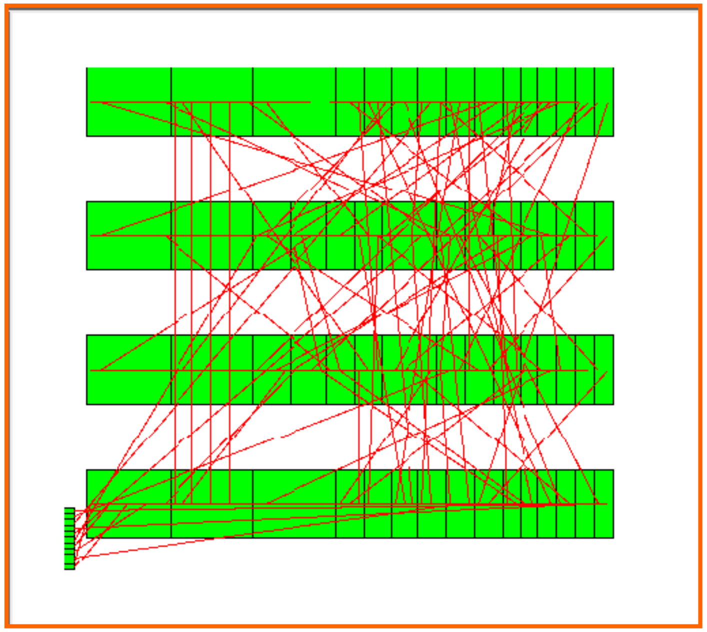


Optimal Relative Order:

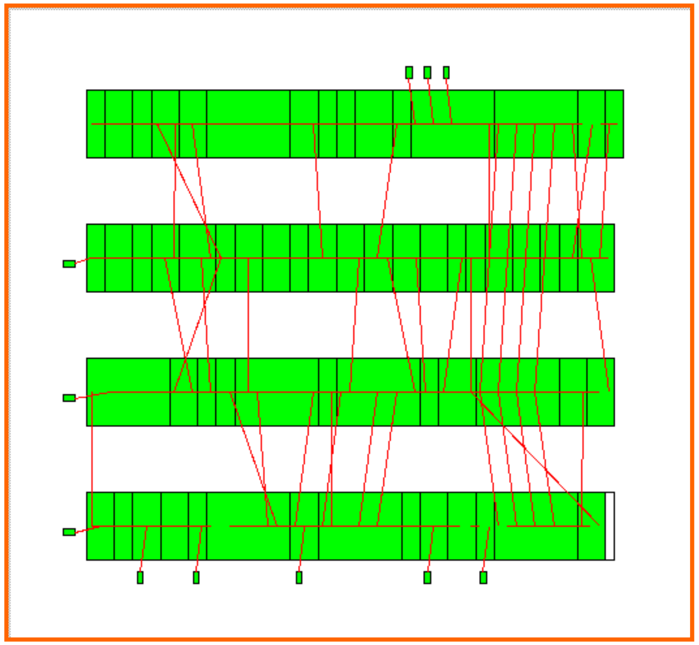


Without "free" space, the placement problem is dominated by order

Placement Problem



A bad placement



A good placement

Types of Cells used in Placement

Standard Cells

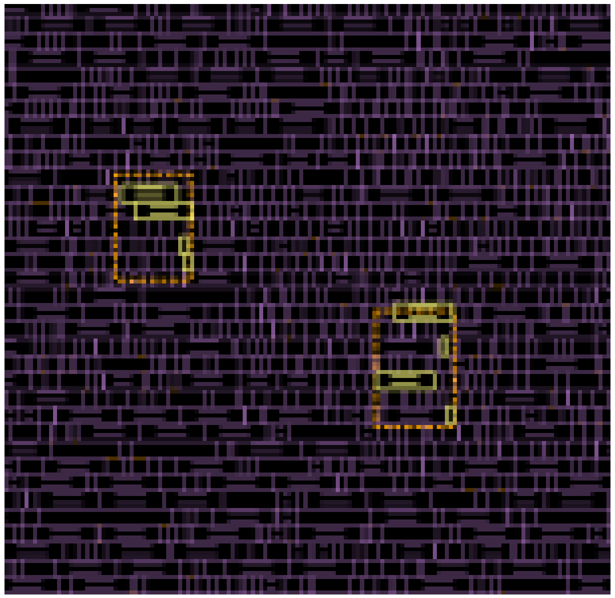
These are the logic cells present in the netlist.

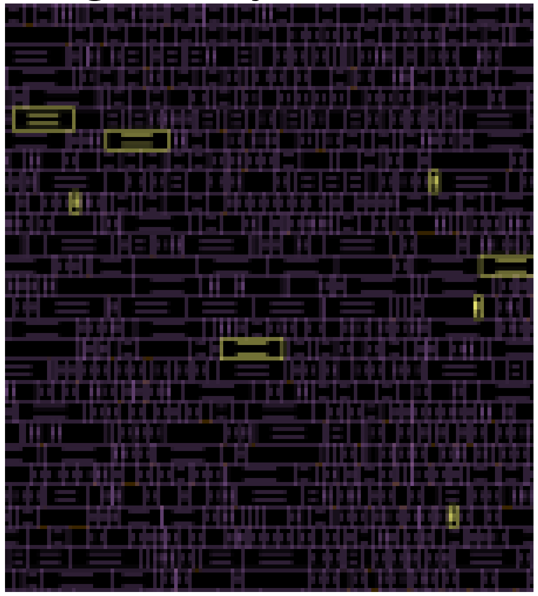
Macro cell

They carry specific functionality. They are optimized for power, area and performance.

Spare cells

Spare cells are extra cells placed in your layout for ECO. it might become necessary to have some functionality changes to the design after placing base layers.


Grouped Spare cells


Sprinkled Spare cells

Goal, Objectives & Inputs - Outputs

Goals

To arrange the all the logic cells with in the specified core area

Objectives

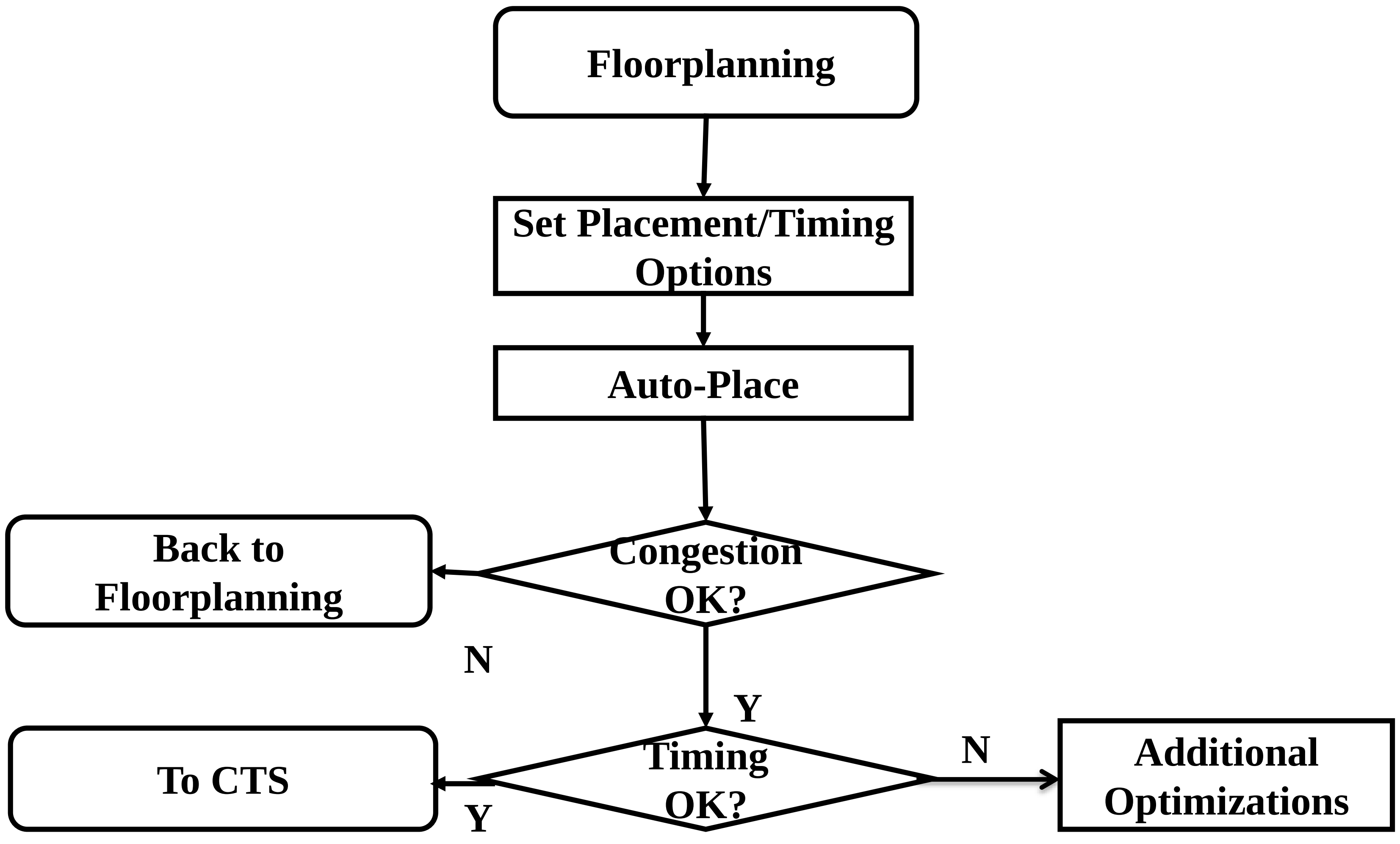
a. To make modular placement.

b. Minimize the all the critical net delays

c. To place cells without congestion

- **Inputs:**
  - Connectivity information between cells(netlist information)
  - Floorplan database
  - Design Constraints file
- **Output:**
  - Congestion reports
  - Cell density reports
  - Timing reports
  - Utilization reports

Placement Flow



Placement Terminology

Global placer	• Place cells to the approximate locations to reduce the wire length
Coarse placer	• Modifies the placement of the standard cells to reduce the congestion, by distributing the cells.
Detail placer	• Assign the exact locations to the cells & remove overlaps of Cells
Routing capacity	• Maximum amount of routing tracks for a die of particular size. • Larger the die area, more number of routing tracks.
Routing density	• Number of inter connects needed in a specific area.
Congestion	• Occurs when the number of wires going through region exceeds the capacity of that region.
Congestion map	• Another form of routability display

Placement Steps

- **Global Placement(Initial Placement):**
  - Spread out cells as much as is necessary to get a maximum local utilization of roughly 60% (by default) – approximate locations
  - It distributes the cell over the core area
  - It minimizes the wire length and congestion of placement
- **Detailed Placement(legalize Placement):**
  - Detailed placement will do the legal positioning of the cells and sizing the cells depend upon the critical paths - local adjustment to obtain the final non-overlapping placement

Placement Steps

Placement options

- **Timing Driven Placement:**

  The objective of timing driven placement is to modify the standard cell placement to meet the timing constraints

- **Congestion Driven Placement:**
  - ➔ Placement of standard cells are changed to reduce the congestion, so that the router can complete routing with out much DRC violations
  - ➔ Coarse placement can be used after global placement to reduce the congestion by reducing the wire length & cell density.

Placement Constraints

- Bounds, Macros, Placement Blockages, Physical cells, Spare cells.
- **Net weight** : Helps in assigning routing priority for critical nets
- **Attributes**

  Floating: Cells are free to placed by any of the automatic placer tool.

  Fixed: Cells are in fixed location and placer can't move them.

  Cover: The placer can't move ,in addition manually also not possible .

What is legal position of cell ?

- Is horizontally aligned with placement grid
- Is vertically aligned with cell rows
- Is not on top of placement blockages
- Introduces no shorts with existing pre-routes
- Lies with in the region/floor plan to which it has been assigned

Placement conclusions

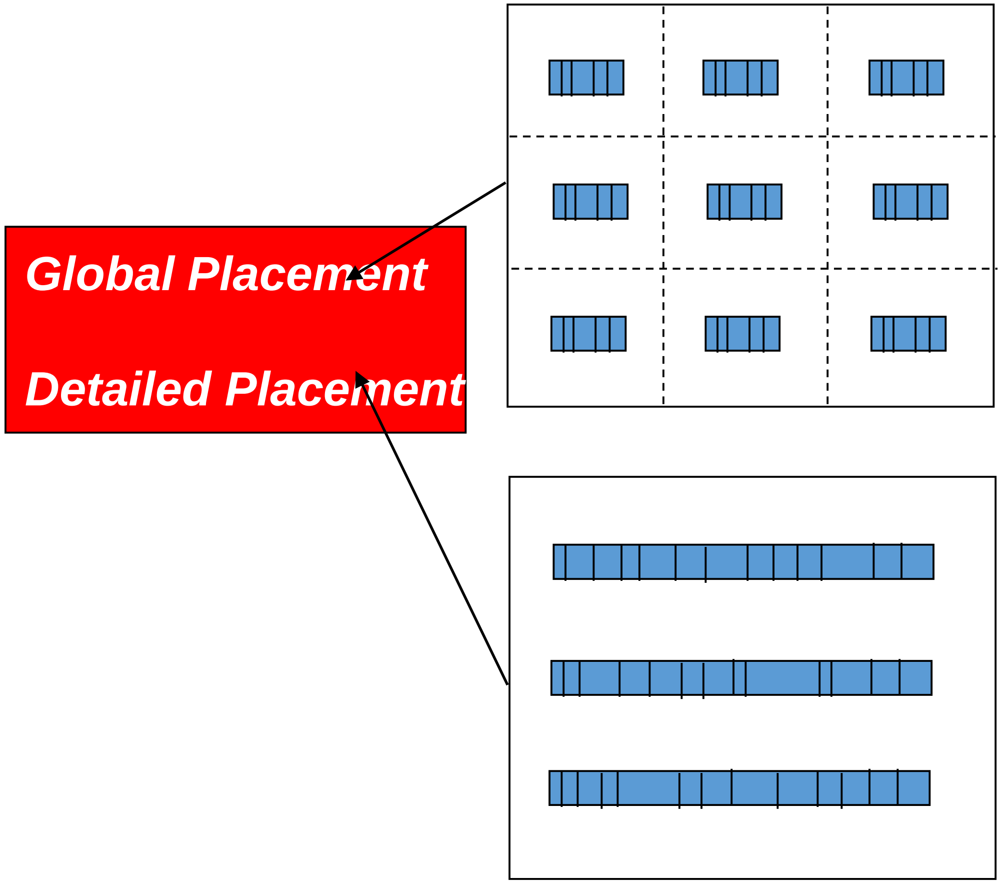
The best Placed Design should meet following points:

- Check for utilization after placement
- Check the timing violations & analyze the same
- Dump the placement information such as Core Area and Gate count etc
- No placement congestion
- No cell overlapping
- No illegal position of cells
- No power Opens or Shorts
- No pre-routed Shorts due to cells.
- Timing requirement should meet (no -ve slack)

Global and Detailed Placement

- In global placement, we decide the <u>approximate locations</u> for cells by placing cells in global bins.

- In detailed placement, we make some <u>local adjustment to obtain</u> the final non-overlapping placement.



Placement

- The goal of standard cell placement is to map ASIC components, or cells, onto positions of the ASIC core area, or standard cell placement region, which is defined by rows.
- The standard cells must be placed in the assigned region such that the ASIC can be routed efficiently and the overall timing requirements can be satisfied.

Placement

- Standard cell placement is a key factor for achieving
  - Physical designs with optimized area usage
  - Routing congestion
  - Timing behavior

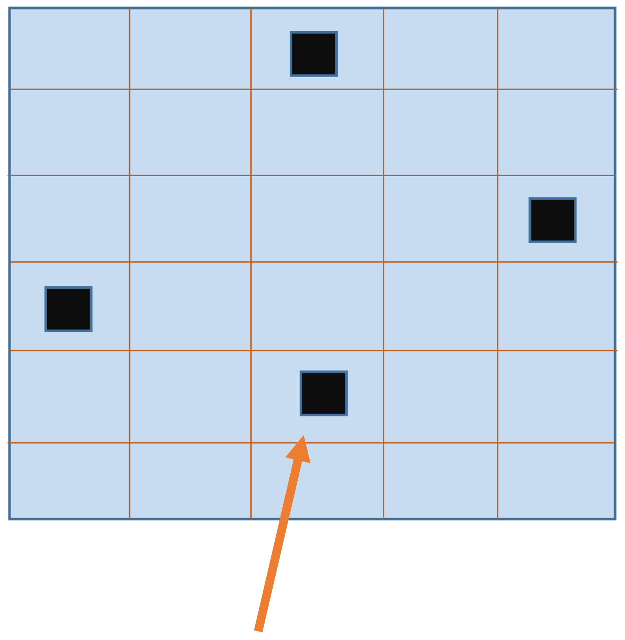
- Algorithms used places the standard cells automatically and are very complex and are being improved frequently.

Reference

Placement by Chris Chu, 11th Chapter

Simple Placer

- Cells fit in grid slots
- Pins are fixed at edges
- Assume
- All gates are of equal sizes
- Each grid slot holds one cell



Gates may be moved to any grid slot

Placer

- Approximates – wire length

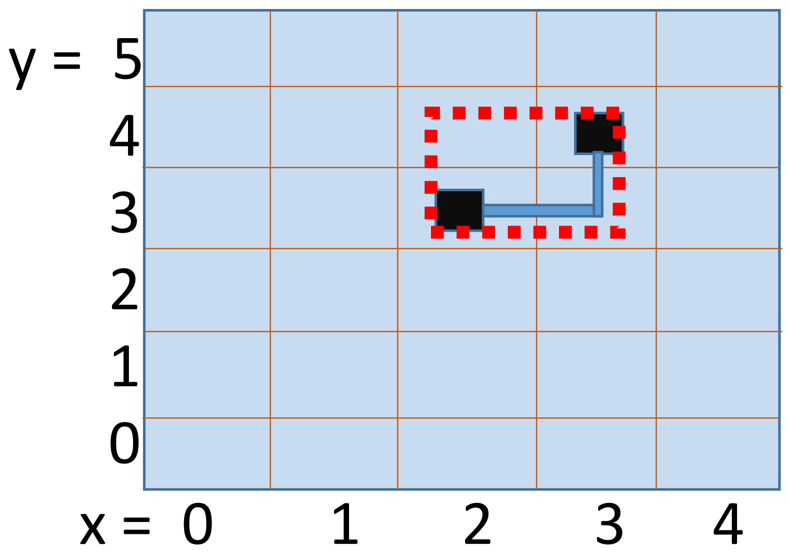
- Optimizes the ability of router to connect all the nets

- Solves for location of the gates to achieve minimum of

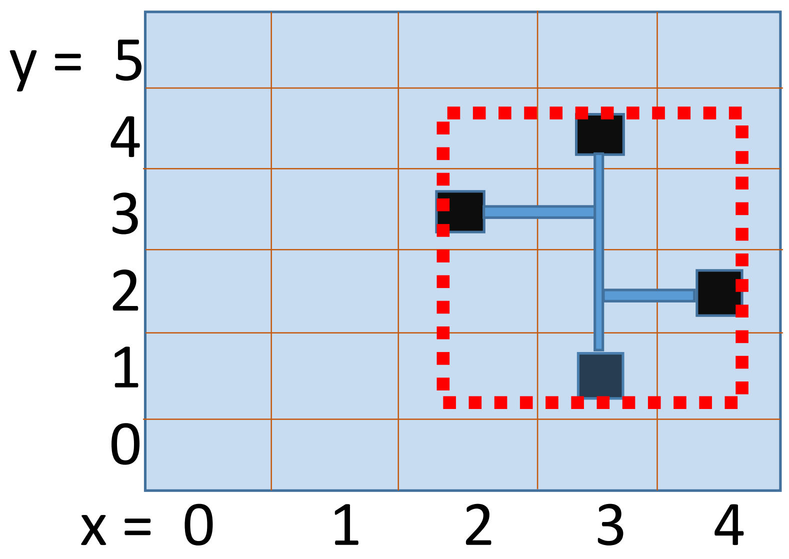
$$\sum_{W_i=1}^{n} Estimated\_Length(Wi)$$

Half-Perimeter Wirelength Estimator

- Net – Wire
- Netlist – Gates and wires
- Points – how many gates it connects (Fan-out)



2 – Point Net

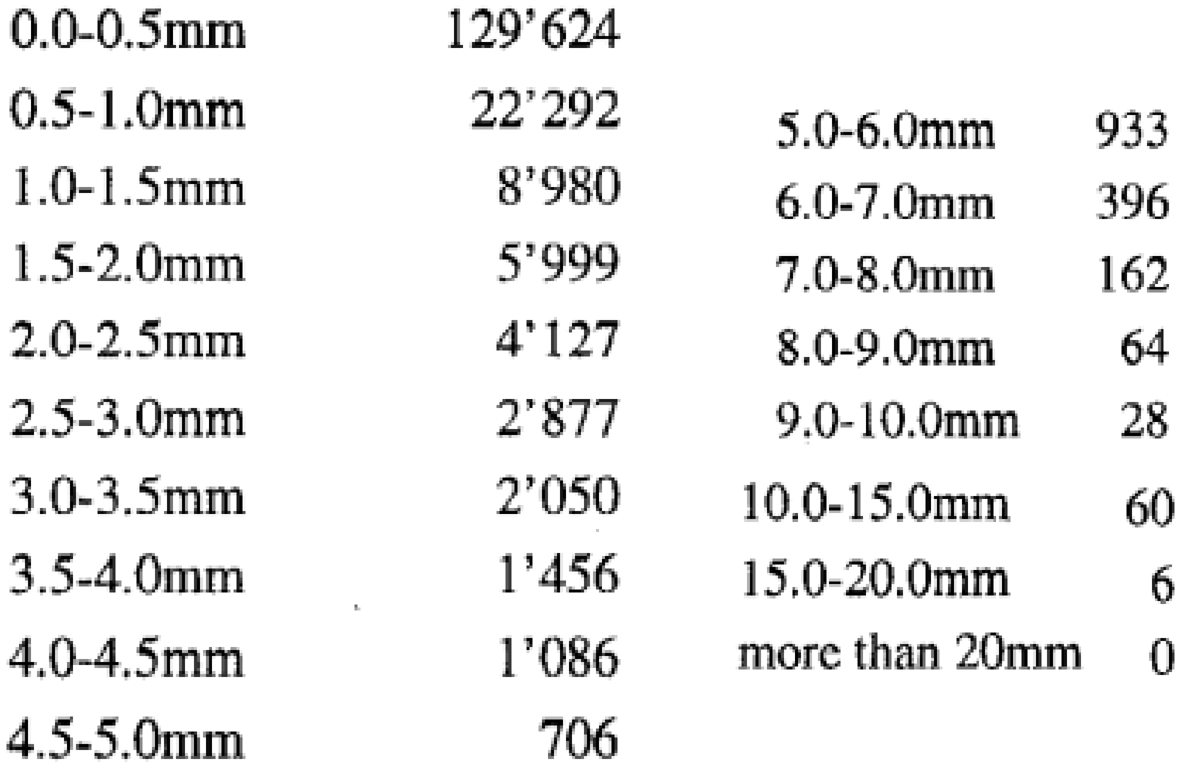


4 – Point Net

- Provides lower bound

Half-Perimeter Wirelength Estimator

- Bounding box wirelength estimation
- Example: For around 2,00,000 nets

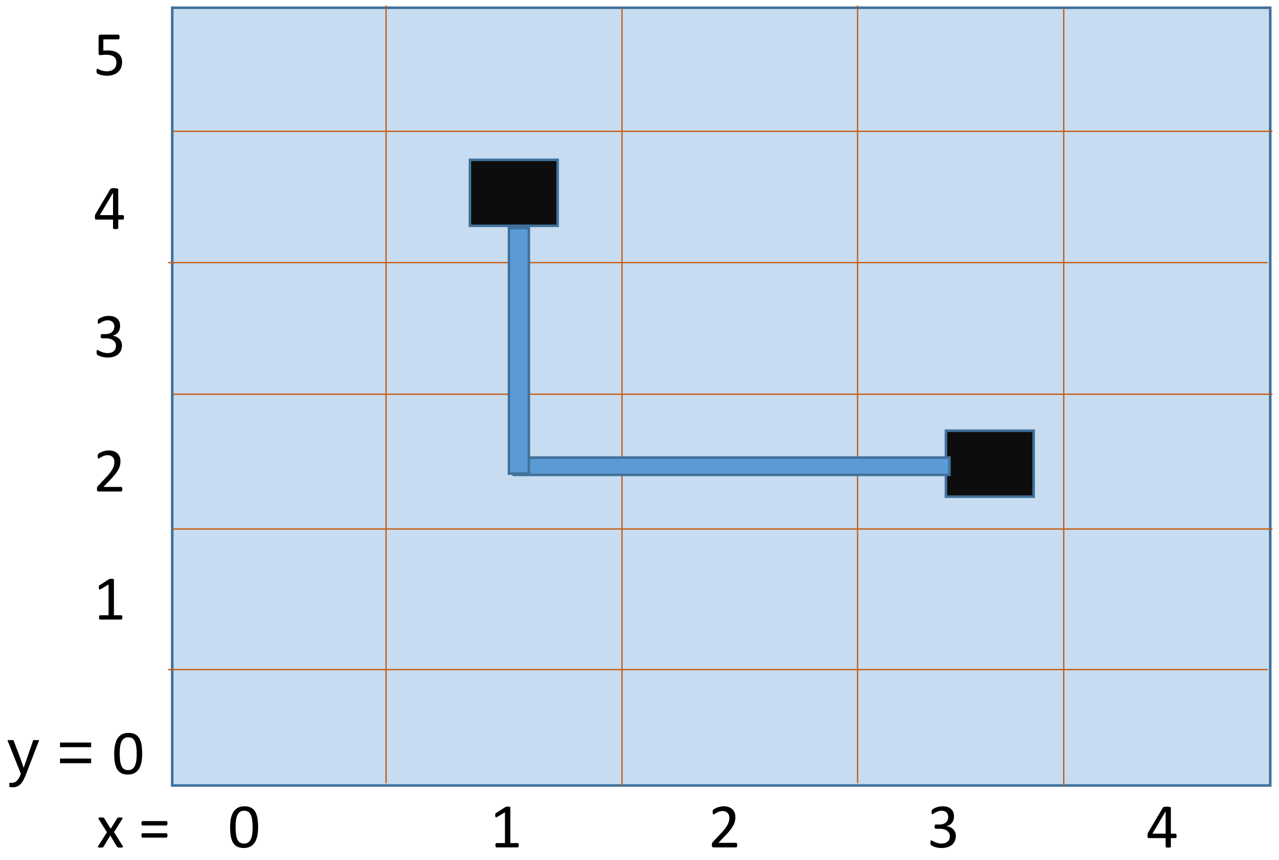


0.0-0.5mm	129'624
0.5-1.0mm	22'292
1.0-1.5mm	8'980
1.5-2.0mm	5'999
2.0-2.5mm	4'127
2.5-3.0mm	2'877
3.0-3.5mm	2'050
3.5-4.0mm	1'456
4.0-4.5mm	1'086
4.5-5.0mm	706
5.0-6.0mm	933
6.0-7.0mm	396
7.0-8.0mm	162
8.0-9.0mm	64
9.0-10.0mm	28
10.0-15.0mm	60
15.0-20.0mm	6
more than 20mm	0

Reference: Vygen, Jens. "**Algorithms for large-scale flat placement**." In *Proceedings of the 34th annual Design Automation Conference*, pp. 746-751. 1997.

Half-Perimeter Wire Length (HPWL)

- Puts- a smallest bounding box around all the gates
- Assume – gate lies at the center of the grid slotΔ



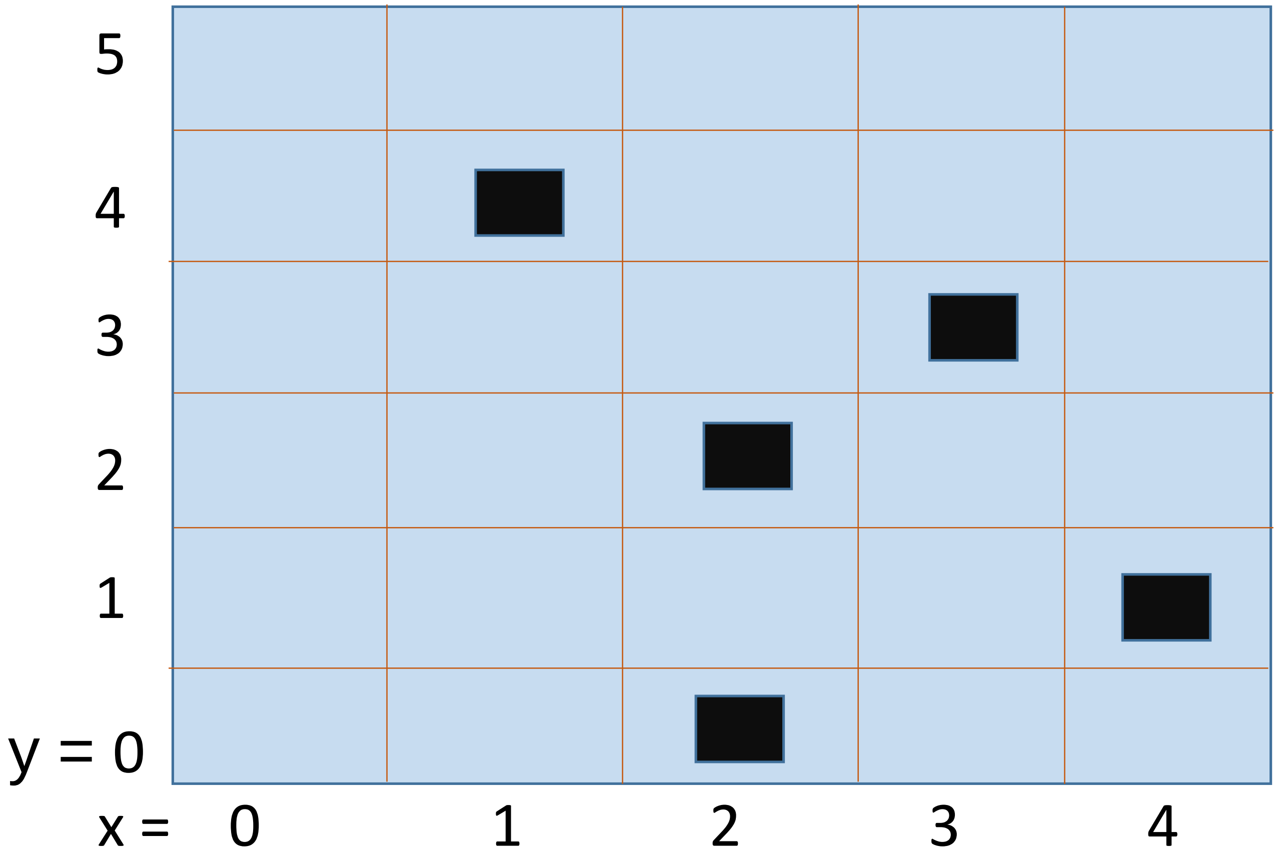
Δx =

Δy =

HPWL=

HPWL = max{x_i} – min{x_i} + max{y_i} – min{y_i}

Half-Perimeter Wire Length (HPWL) - Example



Simple Placer

- Initially use the random placer
  - Randomly assign each gate to a grid
  - Total wire length = sum of HPWL of all gates
- Random iterative improvement
  - Pick any 2 gates and swap their positions
  - Evaluate the change in total wire length
  - Find the difference
  - If less – retain the swap
  - Else – return to the previous positions

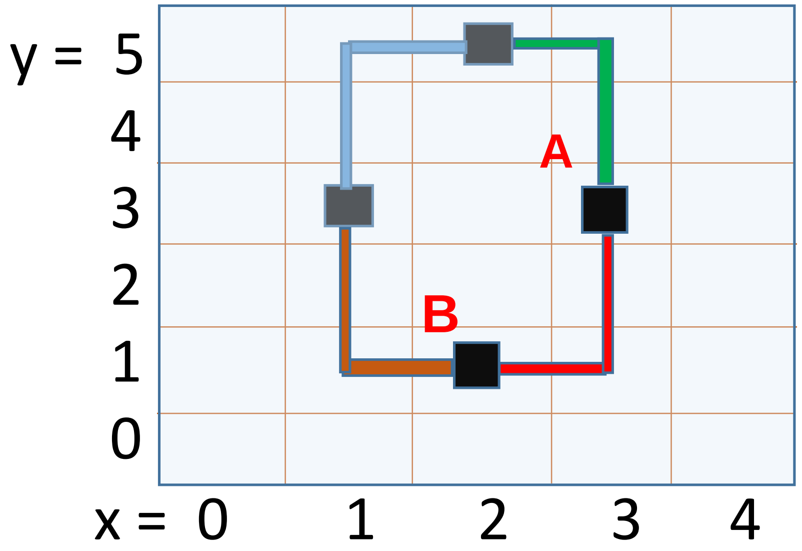
Repeat the procedure until there is no improvement

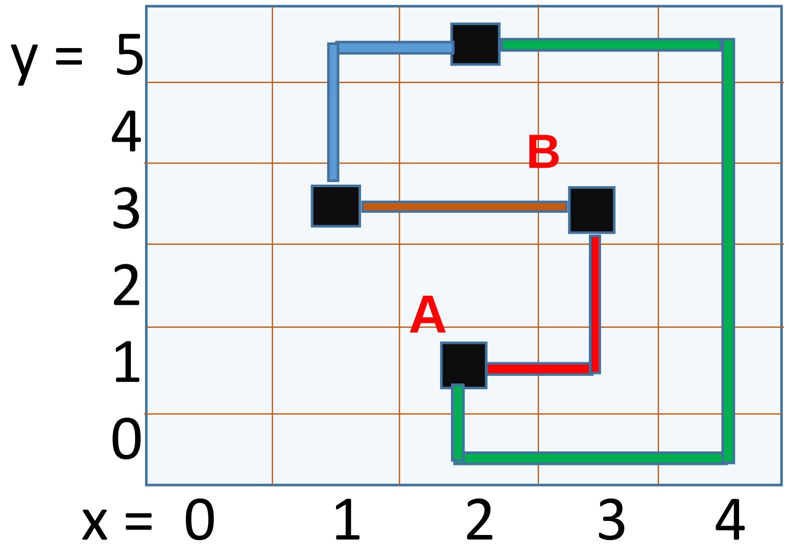
Simple Placer – Pseudo Code

```
//random primary placement
foreach( gate $A_i$ in netlist )
        place $A_i$ in random location (x,y) in grid not already occupied
//calculate initial HPWL wirelength for whole netlist
L=0
foreach( net $N_i$ in the netlist )
        L = L + HPWL($N_i$);
//main improvement loop
        while ( overall HPWL wirelength L is improving ) {
        pick random gate $A_i$; pick random gate $A_j$;
        swap gates $A_i$ and $A_j$
        evaluate $\Delta L$ = new HPWL - old HPWL;
        if ( $\Delta L < 0$ ) { // improved placement - Update HPWL
                $L = L + \Delta L$
        else // $\Delta L > 0$, this is a worse placement
                undo swap of $A_i$ and $A_j$
```

Simple Placer – Incremental

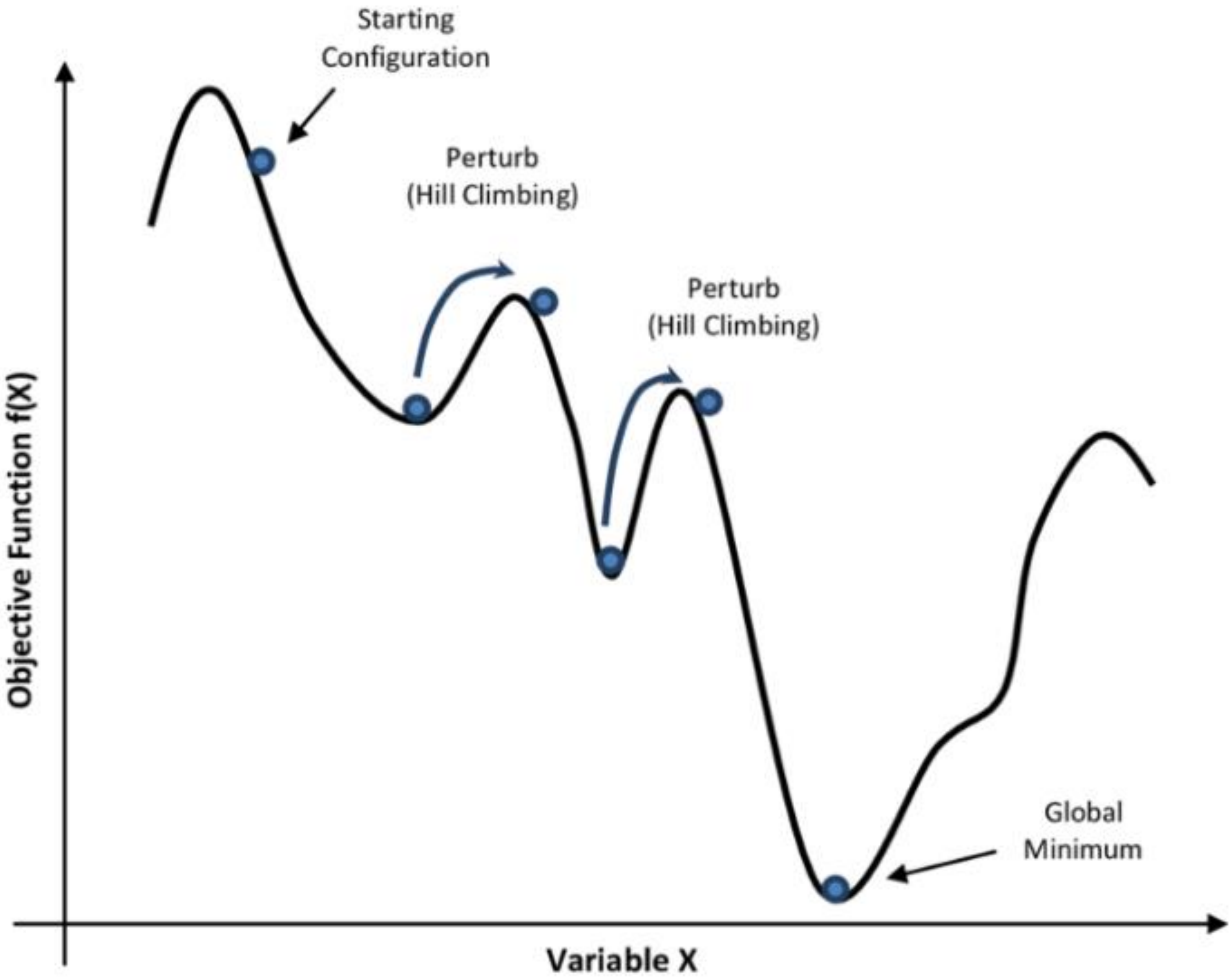
- No need to calculate HPWL for each net
- Compute only the nets for which the gate positions are interchanged





- Applicable for - less than 2500 gates with around 500 pins in an area of 50 X 50
- Time consuming process
- Millions of swaps may be necessary

Iterative Improvement



Iterative Improvement – Hill Climbing

- Previous idea
  - Find out the difference in change in wirelength
  - If less than 0, keep the swap else undo the swap
- New idea
  - Find out the difference in change in wirelength
  - If less than 0, keep the swap else use a function **(P)** with a control parameter **(Temperature)** to determine whether to keep the swap
- Function
  - **P(ΔL, T)**
- Evaluate

$$P = e^{\frac{-\Delta L}{T}}$$

- Compare with a uniformly random number **(R)** between 0 and 1
  - If R < P – keep the swap
  - If R > P – undo the swap

Iterative Improvement – Simulated Annealing

- Begin with T = Hot, very high temperature
- Compute P – is swap increases wirelength – accept it randomly
- Decrease - T until a particular temperature is reached and repeat the steps

Simulated Annealing – Pseudo Code

```
Begin with an initial placement randomly and compute HPWL
Temp =  hot; stop = false;
while ( ! stop ) {
        for (x=1; x< M* No. of gates; x++) {
        Swap 2 random gates $A_i$ and $A_j$
        Evaluate $\Delta L$ = [ΣHPWL after swap] - [ΣHPWL before swap]
        if ($\Delta L < 0$ ) then
                accept this swap // new, better placement
        else {
                if( $R < \exp(-\Delta L/Temp)$ ) // R, uniform random number
                        then accept this uphill swap //new, worse placement
                else undo this uphill swap
        }
        if (ΣHPWL decreasing over the last few temperatures)
                then Temp = 0.75 * Temp // decrease the temperature
        else stop = true
        }
return (final placement positions)
```

Metropolis Criterion

Working Metropolis Criterion

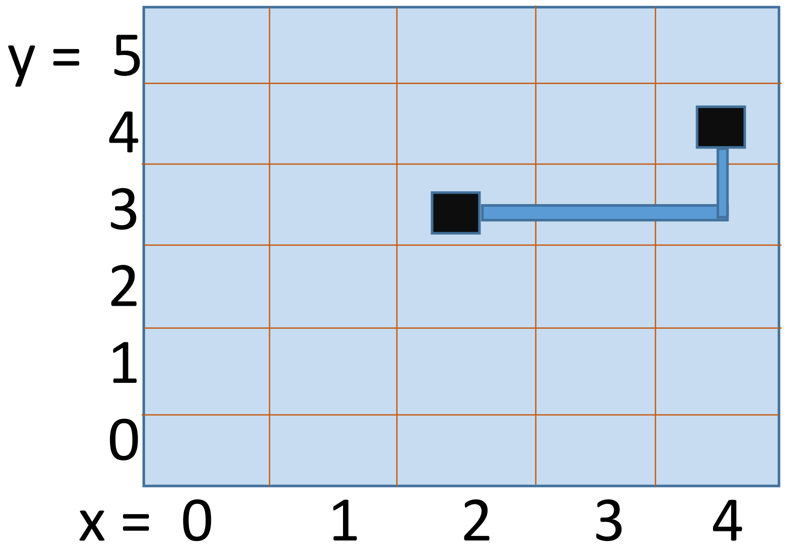
- Assume T=60 and ΔL = 30
- $\boldsymbol{e^{\frac{-\Delta L}{T}}}$ = 0.60 => implies probability of accept the swap is 60%
- Suppose, if 2000 different swaps are tried and get ΔL = 30, then around 0.60 * 2000 = 1231 will be accepted
- This will vary if a different random number (R) is used
- Simulated annealing is inefficient for the designs > 1M gates but works fine for 100K – 500K gates

Analytical Placement – Quadratic

- Quadratic Wirelength Model
- Assume gates as coordinate points (x_i, y_i)
- Write quadratic equations for each wire

Analytical Placement – Quadratic

- For a 2 point net

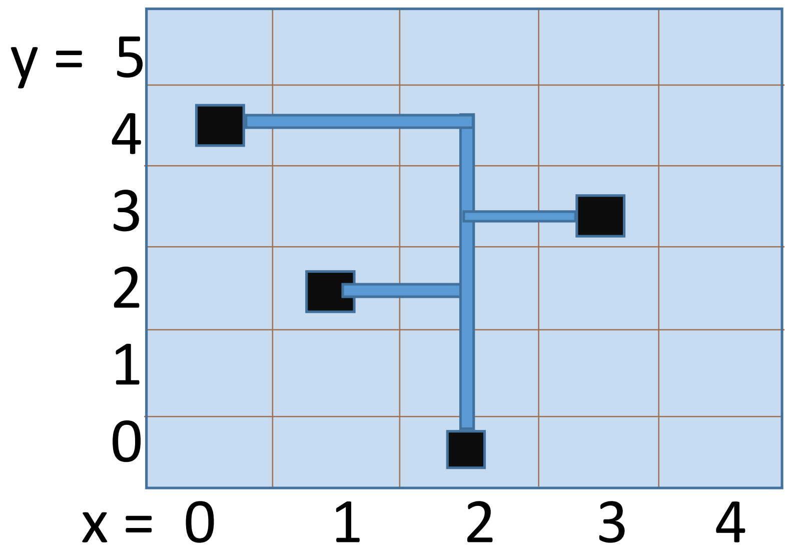


- Quadratic Wirelength

$(4 - 2)^2 + (4 - 3)^2 = 5$

Analytical Placement – Quadratic

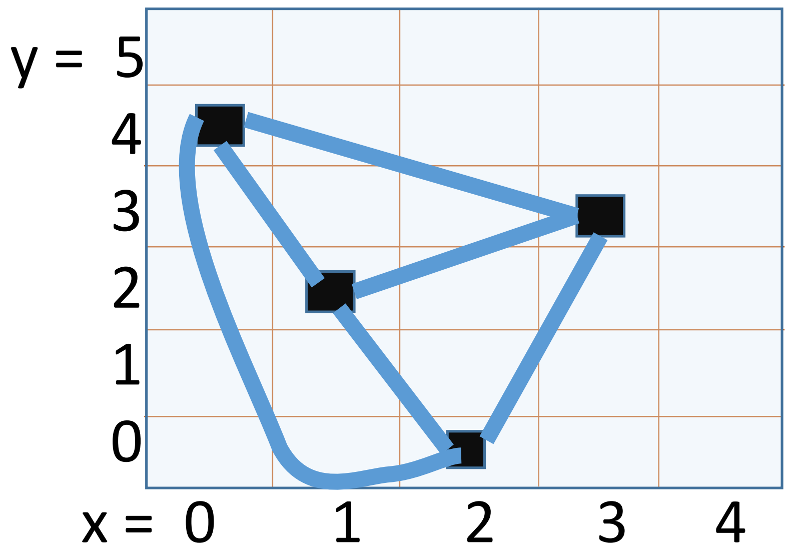
- For a k point net



- Add a new net in between every 2 gates to form fully connected clique model

Analytical Placement – Quadratic

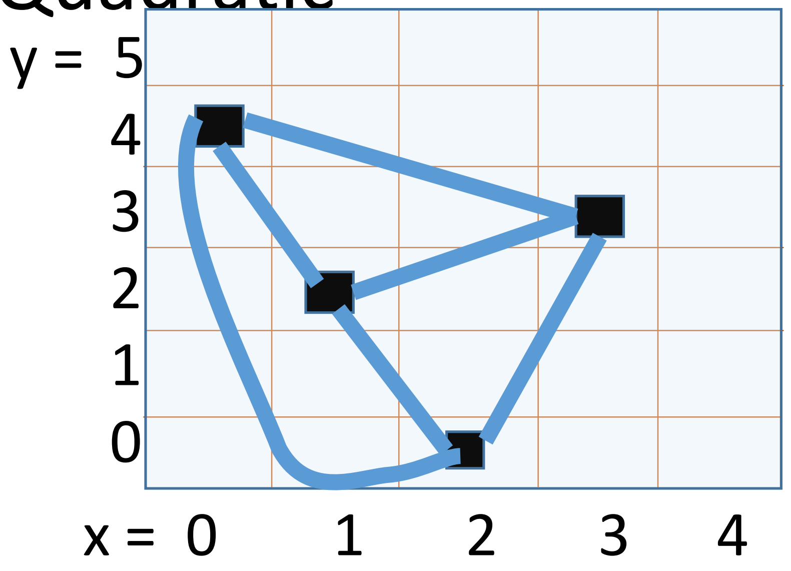
- For a k point net



- k(k-1)/2 = 6, 2-point nets
- Avoid over estimation by dividing factor, 1/(k-1)

Analytical Placement – Quadratic

- For a k point net



Quadratic Wirelength $= \frac{1}{3}[(3-0)^2 + (4-3)^2] +$

$\frac{1}{3}[(3-2)^2 + (3-0)^2] +$

$\frac{1}{3}[(3-1)^2 + (3-2)^2] +$

$\frac{1}{3}[(2-0)^2 + (4-0)^2] +$

$\frac{1}{3}[(2-1)^2 + (2-0)^2] +$

$\frac{1}{3}[(1-0)^2 + (4-2)^2]$

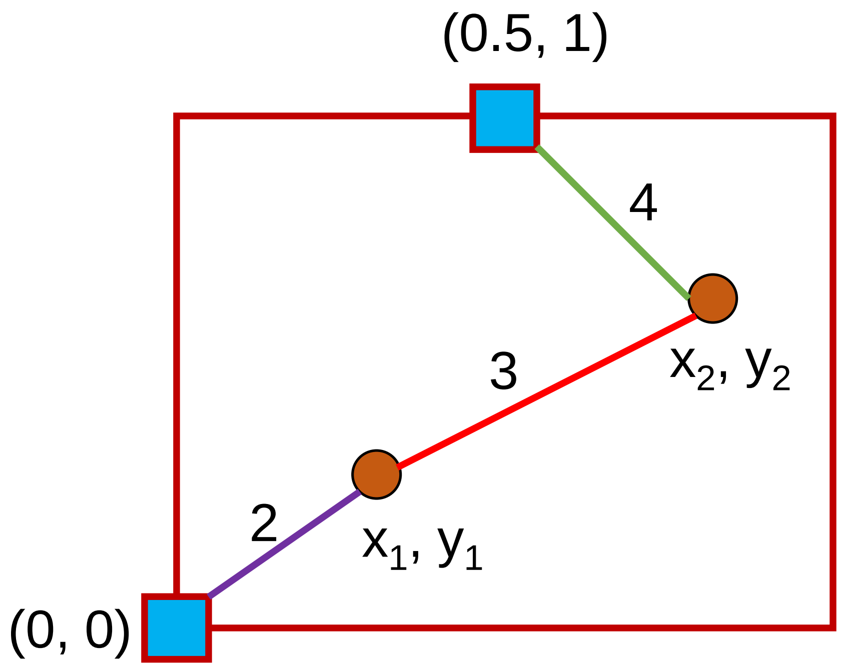
- Addition of 6 weighted, 2 point nets

- Answer = 55/3

  = 18.33

Analytical Placement – Quadratic

- Quadratic Wirelength Model
- Assume gates as coordinate points (x_i, y_i)
- Write quadratic equations for each wire



$$2(x_1-0)^2+2(y_1-0)^2$$

$$3(x_2-x_1)^2+3(y_2-y_1)^2$$

$$4(x_2-0.5)^2+4(y_2-1)^2$$

- No terms with an x_i multiplied by a y_j
- x and y terms can be separated

Analytical Placement – Quadratic

- Steps to minimize
- Use basic calculus, partial derivative

$2(x_1-0)^2+2(y_1-0)^2$

$3(x_2-x_1)^2+3(y_2-y_1)^2$

$4(x_2-0.5)^2+4(y_2-1)^2$

- $F(x) = 2(x_1-0)^2 + 3(x_2-x_1)^2 + 4(x_2-0.5)^2$
- $\frac{\partial F}{\partial x1} = 4x_1 - 6(x_2 - x_1)$
- $= 10x_1 - 6x_2$

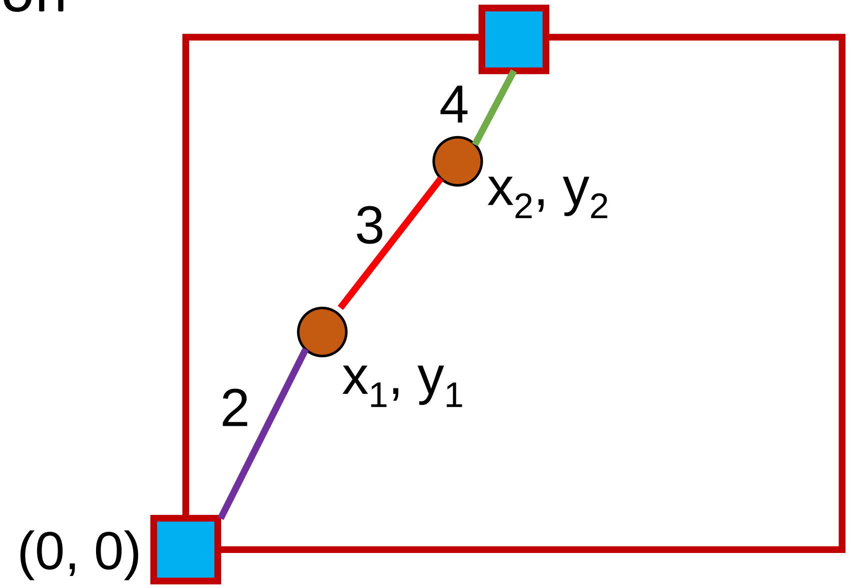
- $\frac{\partial F}{\partial x2} = 6(x_2 - x_1) + 8(x_2 - 0.5)$
- $= -6x_1 + 14x_2 - 4$

- $F(y) = 2(y_1-0)^2 + 3(y_2-y_1)^2 + 4(y_2-1)^2$
- $\frac{\partial F}{\partial y1} = 4y_1 - 6(y_2 - y_1)$
- $= 10y_1 - 6y_2$

- $\frac{\partial F}{\partial y2} = 6(y_2 - y_1) + 8(y_2 - 1)$
- $= -6y_1 + 14y_2 - 8$

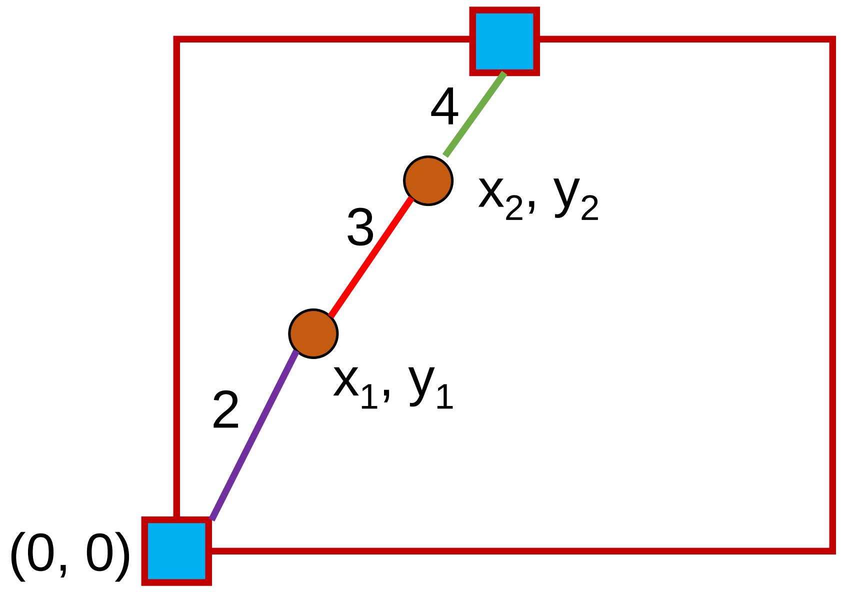
Analytical Placement – Quadratic

- Steps to minimize
- Solve the obtained linear equation
- $\begin{bmatrix} 10 & -6 \\ -6 & 14 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 4 \end{bmatrix}$
- $\begin{bmatrix} 10 & -6 \\ -6 & 14 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 8 \end{bmatrix}$
- $x_1 = 0.2308$, $y_1 = 0.4615$
- $x_2 = 0.3846$, $y_2 = 0.769$



- Two matrix equations with same matrix for x & y and with different RHS
- $Ax = b_x$ and $Ay = b_y$
- Size of the matrix is M X M for M number of gates

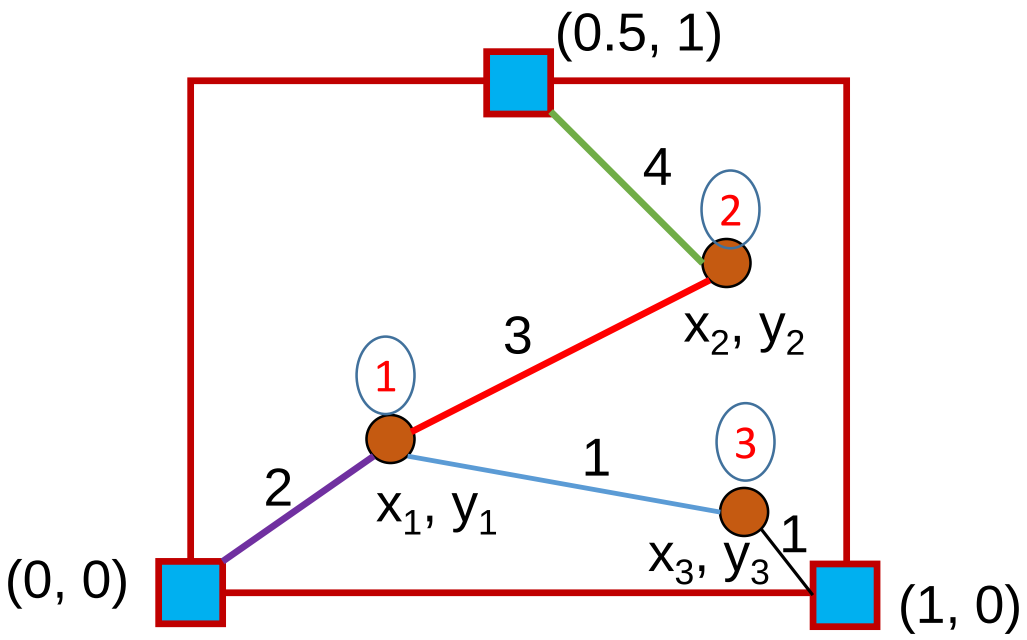
Analytical Placement – Quadratic



- All points tries to make a straight line between the I/O pads
- Higher the weight – shorter is the wirelength

Matrix to Build Quadratic Placement

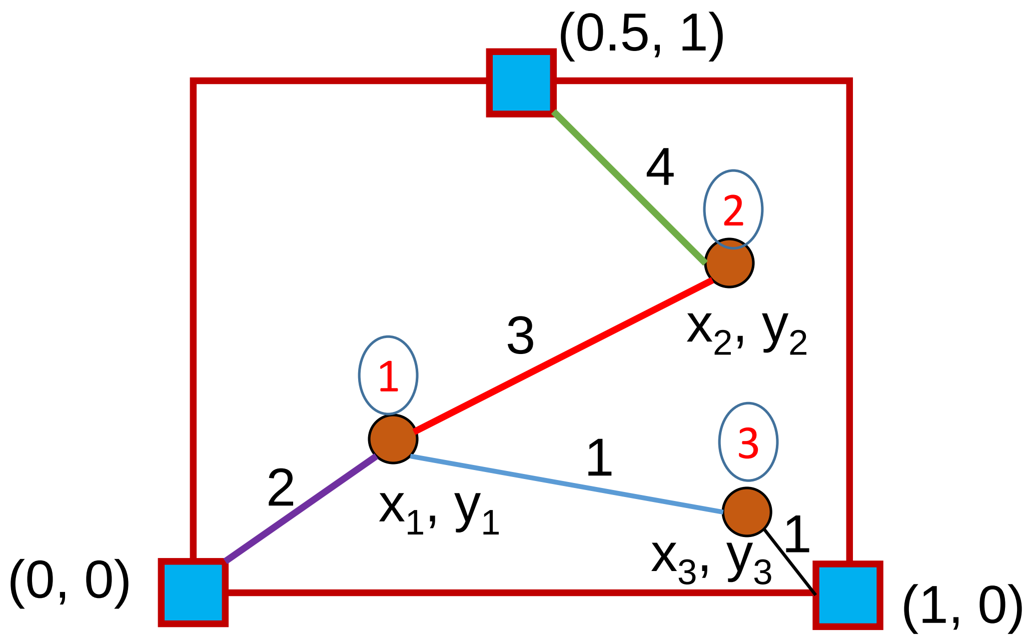
- Form matrices, C, A, b_x, and b_y



- Matrix C = $\begin{bmatrix} 0 & 3 & 1 \\ 3 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}$
- Matrix A = $\begin{bmatrix} 6 & -3 & -1 \\ -3 & 7 & 0 \\ -1 & 0 & 2 \end{bmatrix}$

Matrix to Build Quadratic Placement

- Form matrices, C, A, b_x, and b_y



- Matrix $b_x = \begin{bmatrix} 0 \\ 2 \\ 1 \end{bmatrix}$
- Matrix $b_y = \begin{bmatrix} 0 \\ 4 \\ 0 \end{bmatrix}$

Matrix to Build Quadratic Placement

Solve $\begin{bmatrix} 6 & -3 & -1 \\ -3 & 7 & 0 \\ -1 & 0 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \\ 1 \end{bmatrix}$

and $\begin{bmatrix} 6 & -3 & -1 \\ -3 & 7 & 0 \\ -1 & 0 & 2 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 4 \\ 0 \end{bmatrix}$

You may refer - Online Equation Solver: https://spark-public.s3.amazonaws.com/vlsicad/javascript_tools/solver.html