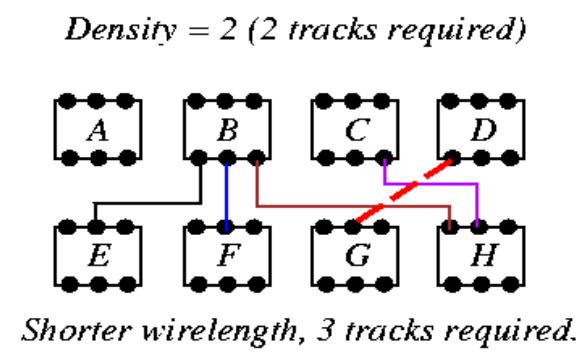
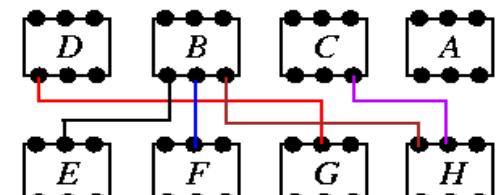
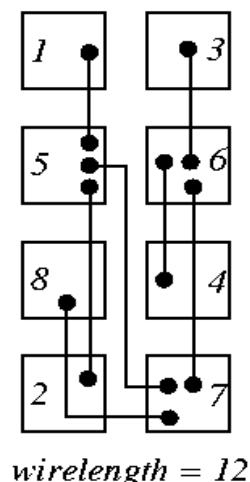
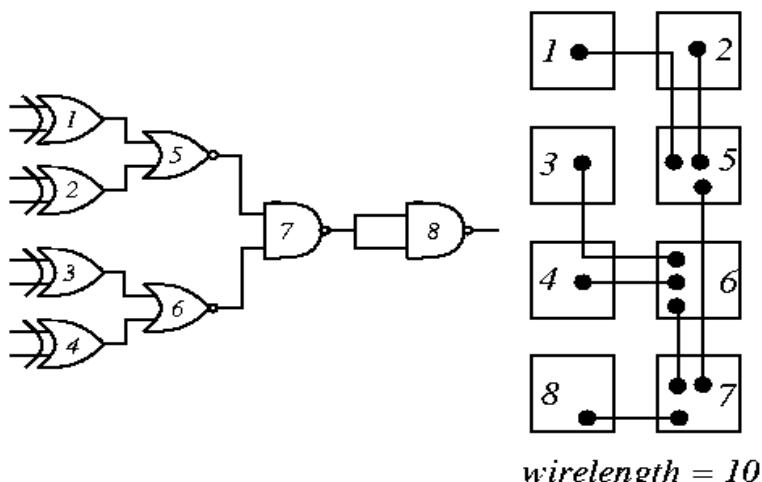


Placement

- The process of arranging the circuit components on a layout surface.
- Inputs: A set of pre-placed/movable cells/modules and a netlist.
- Goal: Find the best position for each movable cell/module on the chip (or placement region) according to appropriate cost functions.
 - wirelength, routability/channel density, cut size, timing, power, thermal, I/O pads, manufacturability, etc.

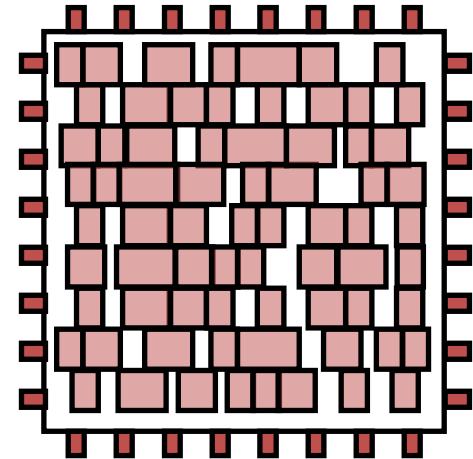


Placement for Different Design Styles

- Standard cell placement
- Gate array / FPGA placement
- Macro block placement
 - Typically called floorplanning
- Mixed-size placement
 - A few large macro blocks
 - A large number of small cells

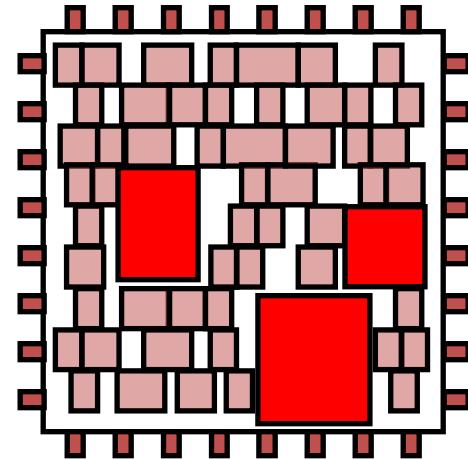
Wirelength-Driven Standard Cell Placement

- Given:
 - A netlist
 - Standard cells C_1, \dots, C_n with given shapes and pin locations
 - Nets N_1, \dots, N_m
 - A placement region with given cell rows
- Place cells in the placement region:
 - Determine coordinates (x_i, y_i) for cell C_i
 - No overlap among cells
 - The total wirelength is minimized



Wirelength-Driven Mixed-Size Placement

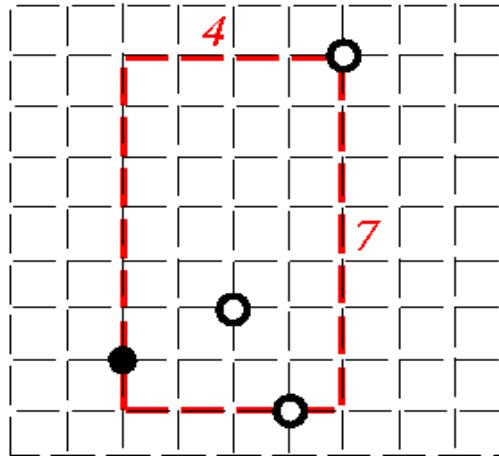
- Given:
 - A netlist
 - Standard cells C_1, \dots, C_n and macros M_1, \dots, M_r with given shapes and pin locations
 - Nets N_1, \dots, N_m
 - A placement region with given cell rows
- Place cells/modules in the placement region:
 - Determine coordinates (x_i, y_i) for C_i / M_i
 - No overlap among cells and macros.
 - The total wire length is minimized.



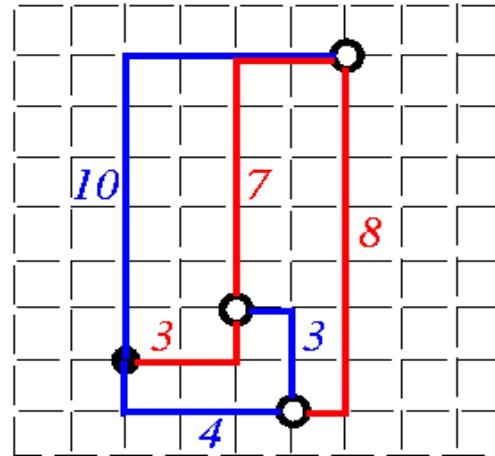
Estimation of Wirelength

- **Half-perimeter wirelength (HPWL)**: Half the perimeter of the smallest bounding rectangle that encloses all the pins of the net to be connected. **Most widely used approximation!**
- **Complete graph**: Since #edges $\binom{n(n-1)}{2}$ in a complete graph is $\frac{n}{2} \times$ # of tree edges ($n - 1$), $wirelength \approx \frac{2}{n} \sum_{(i,j) \in net} dist(i,j)$.
- **Minimum chain**: Start from one vertex and connect to the closest one, and then to the next closest, etc.
- **Source-to-sink connection**: Connect one pin to all other pins of the net.
- **Steiner-tree (approximation)**
- **Minimum spanning tree**

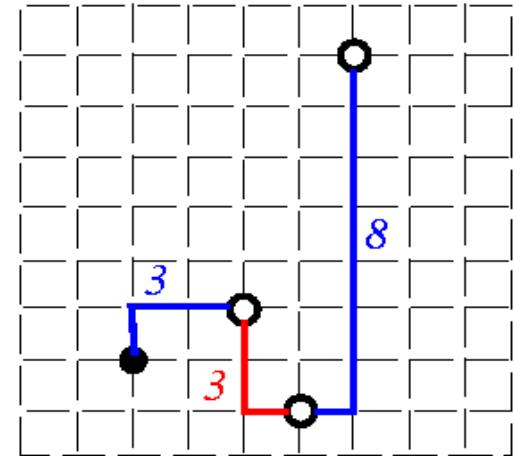
Estimation of Wirelength (cont'd)



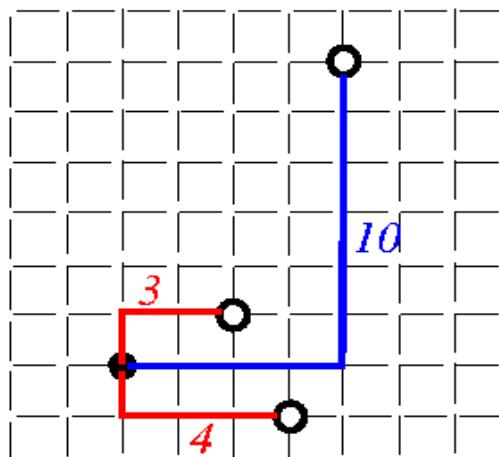
*semi-perimeter len = 11
half*



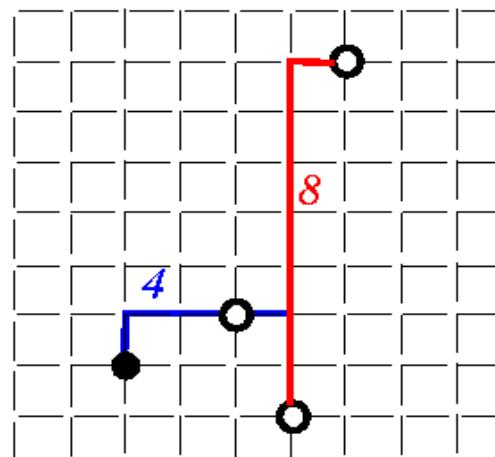
*complete graph len * 2/n = 17.5*



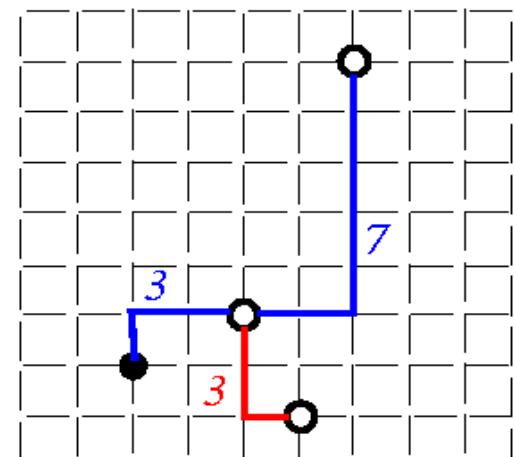
chain len = 14



*source-to-sink len = 17
Unit 5*



Steiner tree len = 12



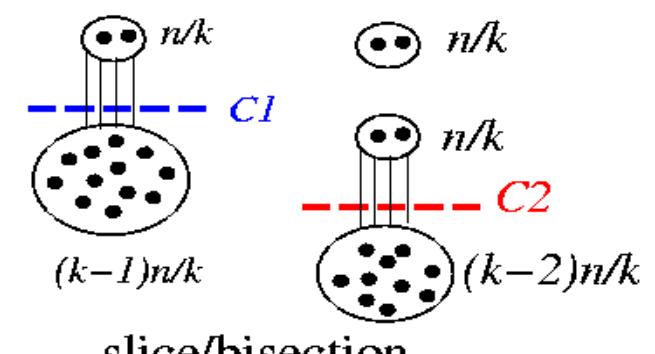
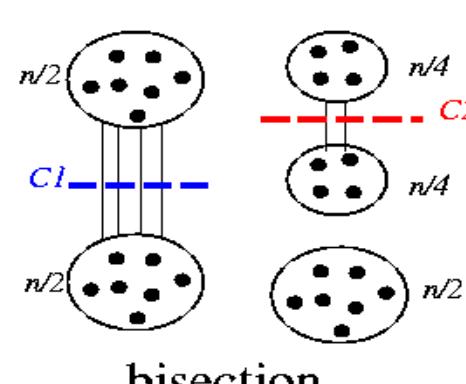
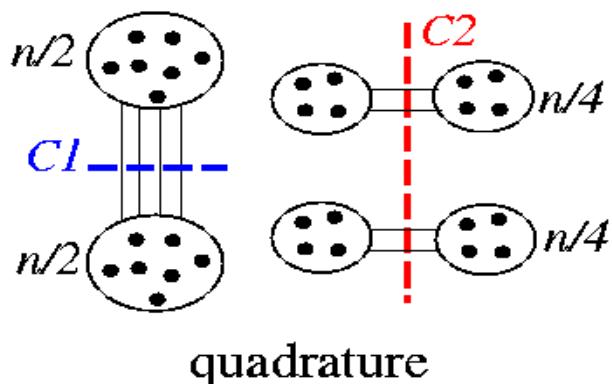
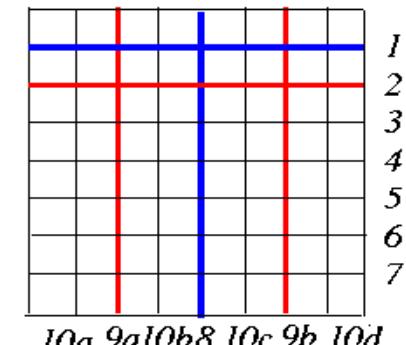
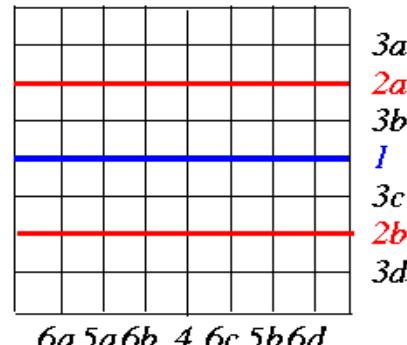
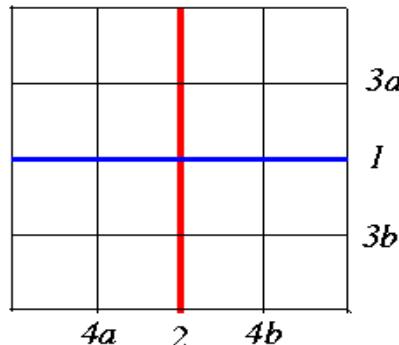
Spanning tree len = 13

Placement Is Very Hard

- Even this simplified placement problem is NP-hard:
 - 1-D placement
 - 2-pin nets only
 - All modules are of the same size
 - Wirelength as the only cost function
- Realistic placement problems:
 - 2-D placement
 - Multi-pin nets
 - Modules of different sizes
 - Other cost functions in addition to total WL
 - Different constraints for different design styles
 - Huge problem size

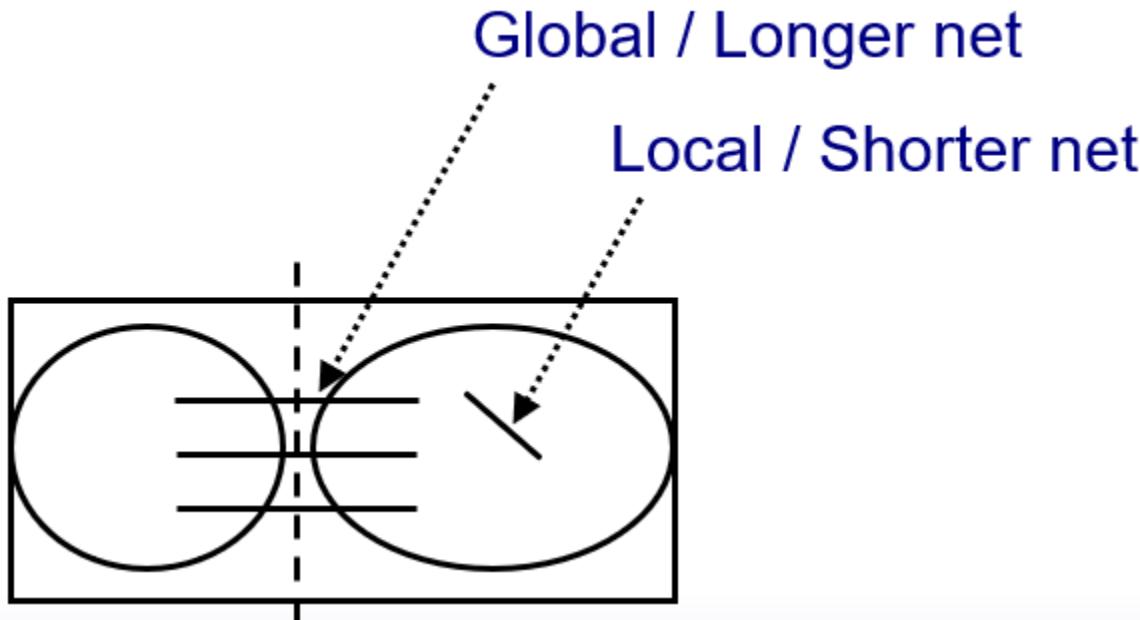
Min-Cut Placement

- Breuer, “A class of min-cut placement algorithms,” DAC-77.
- **Quadrature:** suitable for circuits with high density in the center.
- **Bisection:** good for standard-cell placement.
- **Slice/Bisection:** good for cells with high interconnection on the periphery.



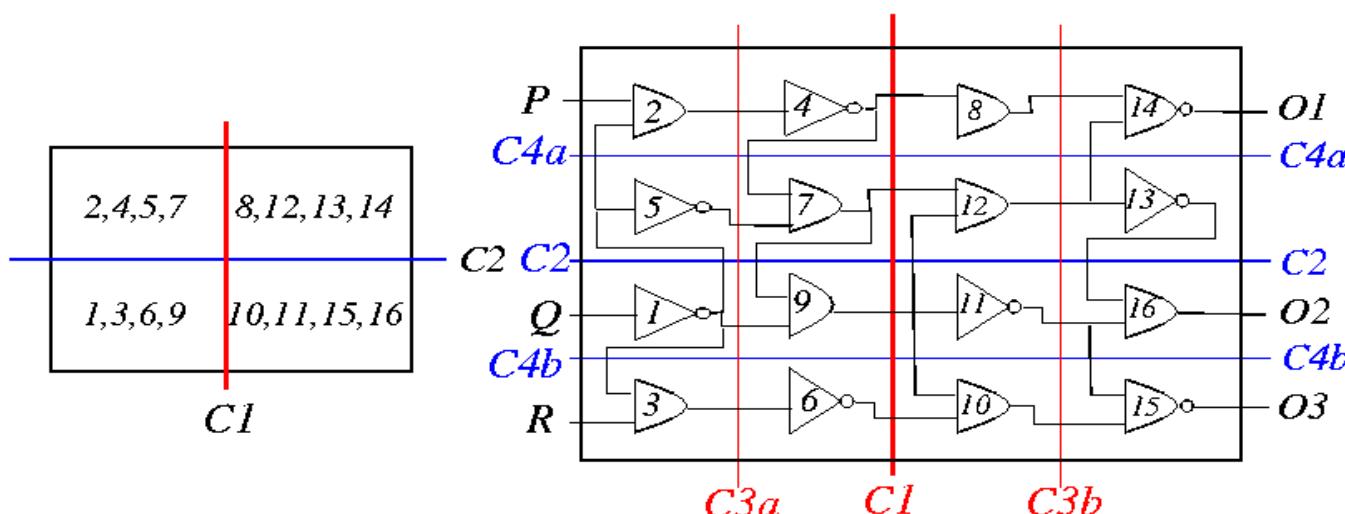
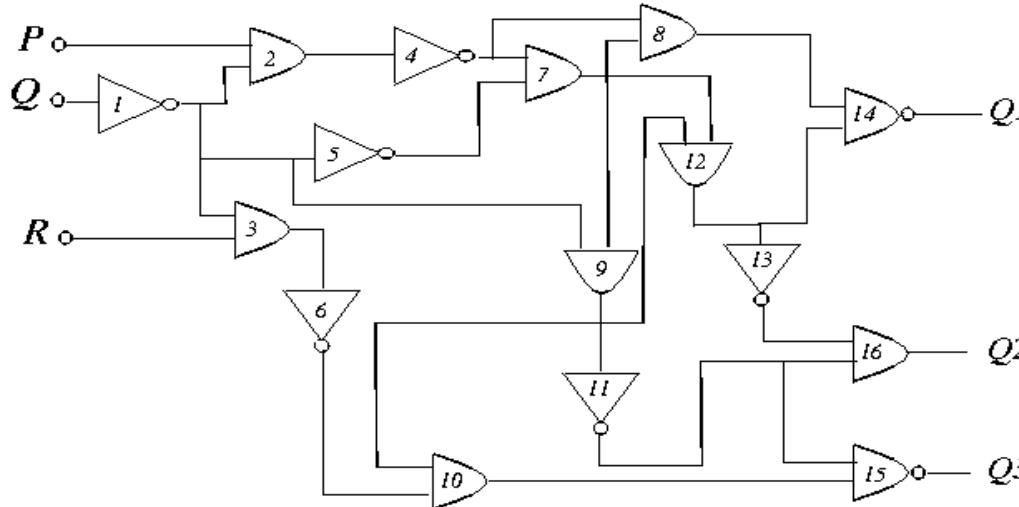
How Cut Size and Wirelength Related?

- Nets across the cut are supposed to be more global and longer nets
- Minimizing cut size is an indirect way to minimize wirelength



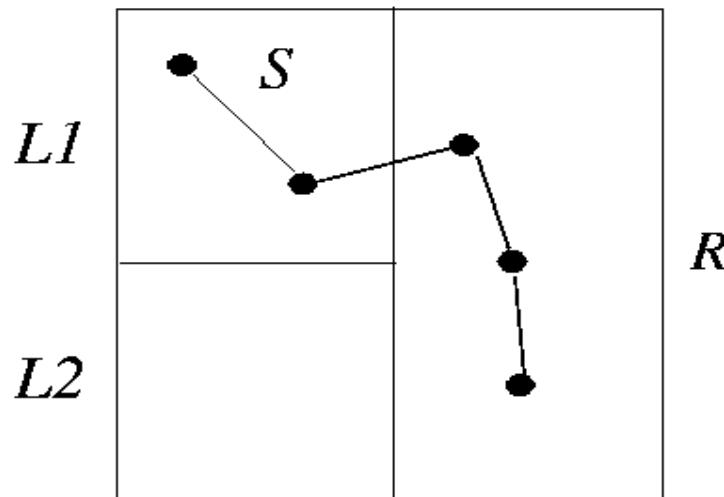
Quadrature Placement Example

- Apply K-L heuristic to partition + Quadrature Placement: Cost $C_1 = 4$, $C_{2L} = C_{2R} = 2$, etc.

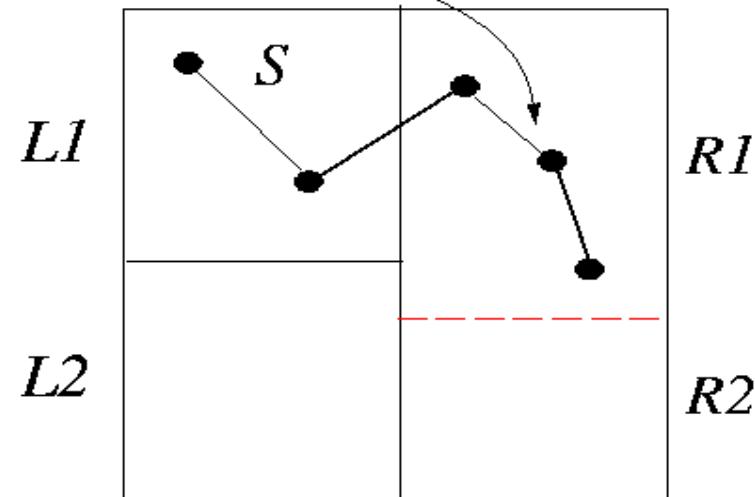


Min-Cut Placement with Terminal Propagation

- Dunlop & Kernighan, “A procedure for placement of standard-cell VLSI circuits,” *IEEE TCAD*, Jan. 1985.
- Drawback of the original min-cut placement: Does not consider the positions of pins that enter a region.
 - What happens if we swap $\{1, 3, 6, 9\}$ and $\{2, 4, 5, 7\}$ in the previous example?

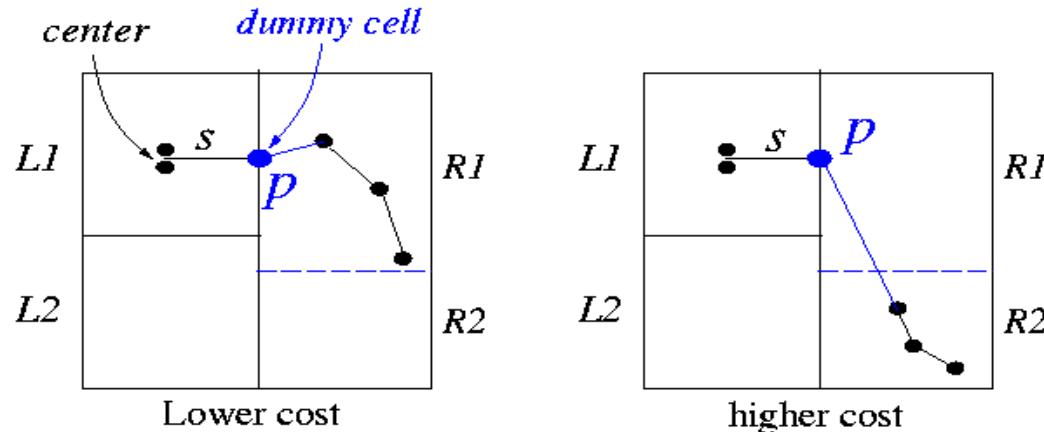


prefer to have them in R1



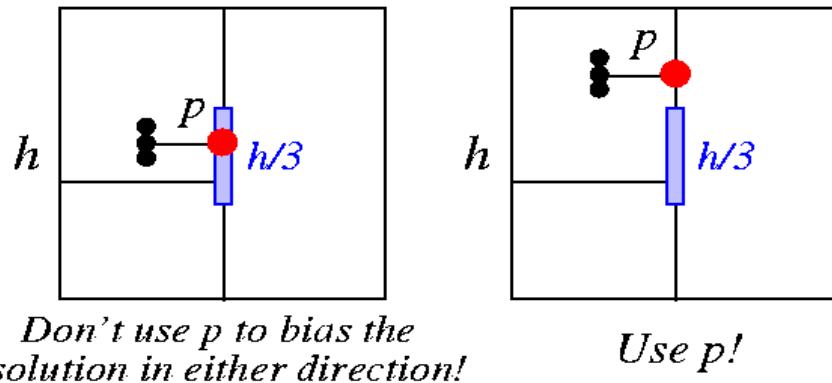
Terminal Propagation

- We should use the fact that s is in L_1 !

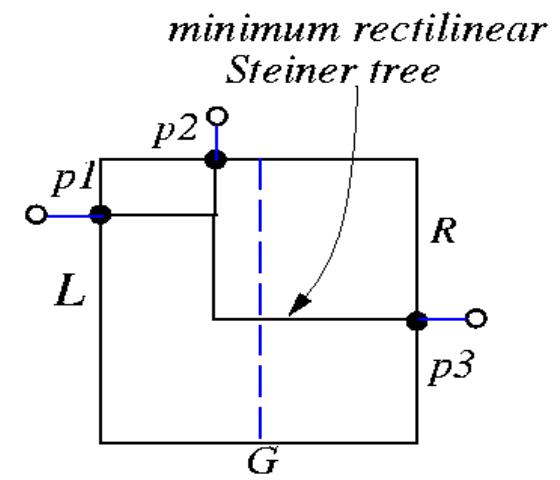


P will stay in RI for the rest of partitioning!

- When not to use p to bias partitioning? Net s has cells in many groups?

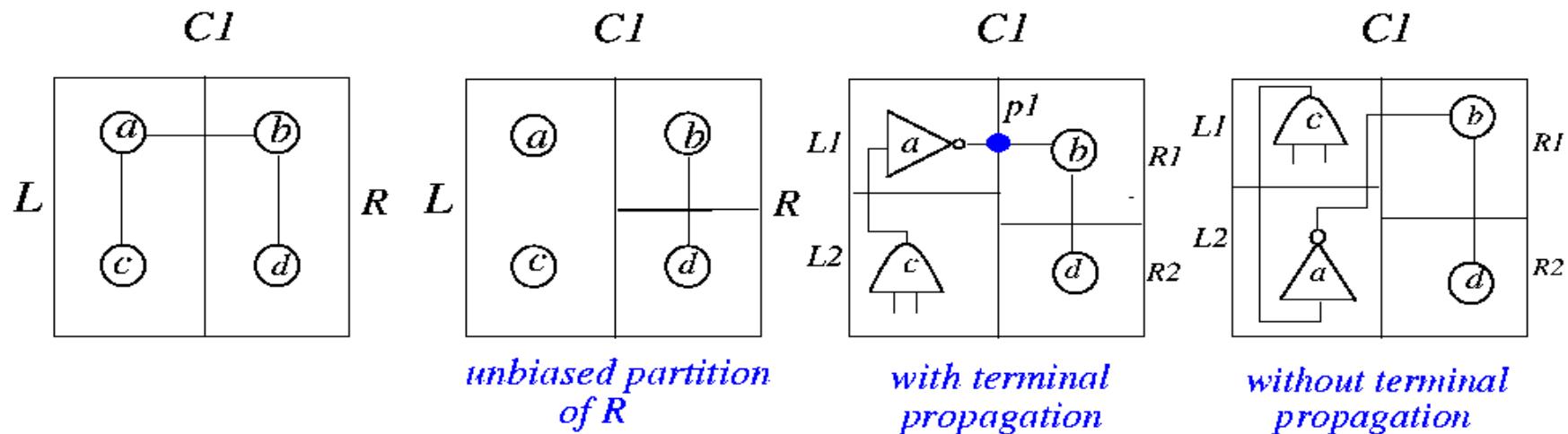
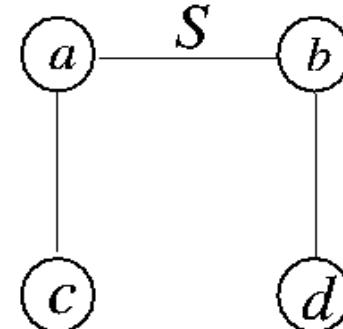
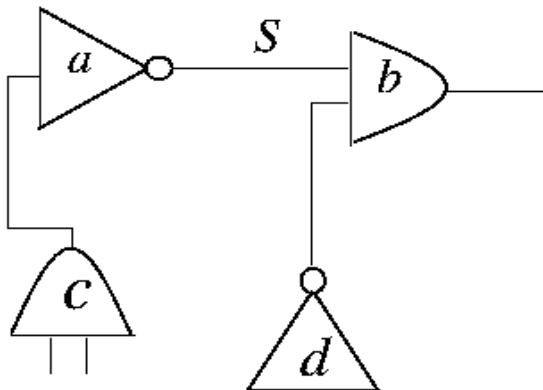


Unit 5



Terminal Propagation Example

- Partitioning must be done breadth-first, not depth-first.

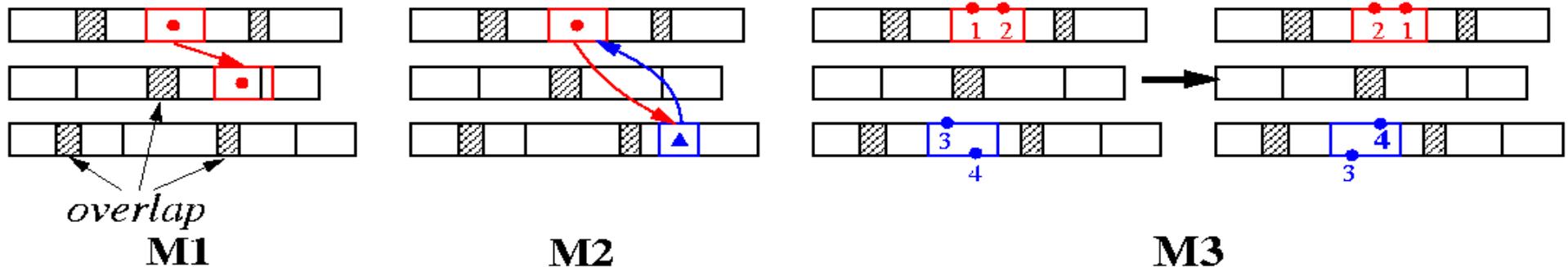


Placement by Simulated Annealing

- Sechen and Sangiovanni-Vincentelli, “The TimberWolf placement and routing package,” *IEEE J. Solid-State Circuits*, Feb. 1985; “TimberWolf 3.2: A new standard cell placement and global routing package,” DAC’86.
- TimberWolf: Stage 1
 - Modules are moved between different rows as well as within the same row.
 - Modules overlaps are allowed.
 - When the temperature is reached below a certain value, stage 2 begins.
- TimberWolf: Stage 2
 - Remove overlaps.
 - Annealing process continues, but only interchanges adjacent modules within the same row.

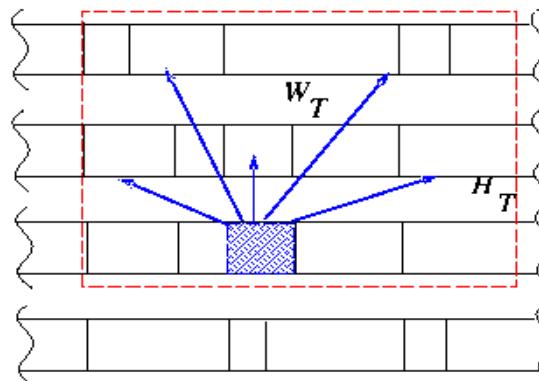
Solution Space & Neighborhood Structure

- **Solution Space:** All possible arrangements of the modules into rows, possibly with overlaps.
- **Neighborhood Structure:** 3 types of moves
 - M_1 : Displace a module to a new location.
 - M_2 : Interchange two modules.
 - M_3 : Change the orientation of a module.



Neighborhood Structure

- TimberWolf first tries to select a move between M_1 and M_2 : $Prob(M_1) = 0.8$, $Prob(M_2) = 0.2$
- If a move of type M_1 is chosen and it is rejected, then a move of type M_3 for the same module will be chosen with probability 0.1.
- Restrictions: (1) what row for a module can be displaced? (2) what pairs of modules can be interchanged?
- **Key: Range Limiter**
 - At the beginning, (W_T, H_T) is very large, big enough to contain the whole chip.
 - Window size shrinks slowly as the temperature decreases. Height and width $\propto \log(T)$.
 - Stage 2 begins when window size is so small that no inter-row module interchanges are possible.



Cost Function

- Cost function: $C = C_1 + C_2 + C_3$.
- C_1 : **total estimated wirelength.**
 - $C_1 = \sum_{i \in Nets} (a_i w_i + b_i h_i)$
 - a_i, b_i are horizontal and vertical weights, respectively.
 - $a_i=1, b_i=1 \Rightarrow \frac{1}{2} \times$ perimeter of the bounding box of Net i .
 - Critical nets: Increase both a_i and b_i .
 - If vertical wirings are “cheaper” than horizontal wirings, use smaller vertical weights: $b_i < a_i$.
- C_2 : penalty function for module overlaps.
 - $C_2 = \gamma \sum_{i \neq j} O_{ij}^2$ γ : penalty weight.
 - O_{ij} : amount of overlaps in the x -dimension between modules i and j .
- C_3 : penalty function that controls the row length.
 - $C_3 = \delta \sum_{r \in Rows} |L_r - D_r|$ δ : penalty weight.
 - D_r : desired row length.
 - L_r : sum of the widths of the modules in row r .

Annealing Schedule

- $T_k = r_k T_{k-1}$, $k = 1, 2, 3, \dots$
- r_k increases from 0.8 to max value 0.94 and then decreases to 0.8.
- At each temperature, a total # of nP attempts is made. n : # of modules; P : user specified constant.
- Termination: $T < 0.1$.