# Routing

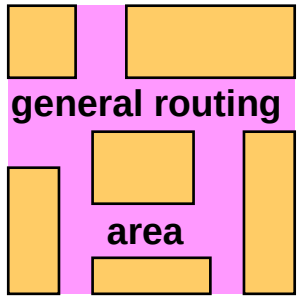
## (4541.554 Introduction to Computer-Aided Design)

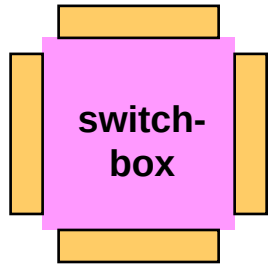
**School of EECS**

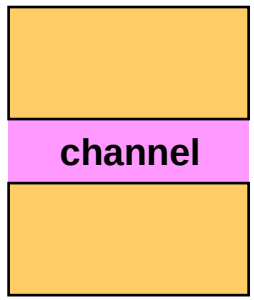
**Seoul National University**

# Introduction

- **Detailed routing**
  - **Unrestricted**
    - **Maze routing**
    - **Line routing**
  - **Restricted**
    - **Switch-box routing: fixed pins on four sides**
    - **Channel routing: fixed pins on two sides**
    - **River routing: Single layer (no crossing)**
- **Global routing (or loose routing)**
- **Channel definition and ordering**
- **Power and ground routing**
- **Clock routing**



# Detailed Routing

- **Problem definition**
  - Given a region with pins on its sides and possibly in the middle, and a net list, determine the interconnection geometry
- **Objective functions**
  - Minimize overall wire length
  - Minimize maximum wire length

    --> minimize maximum signal delay
  - Minimize number of tracks

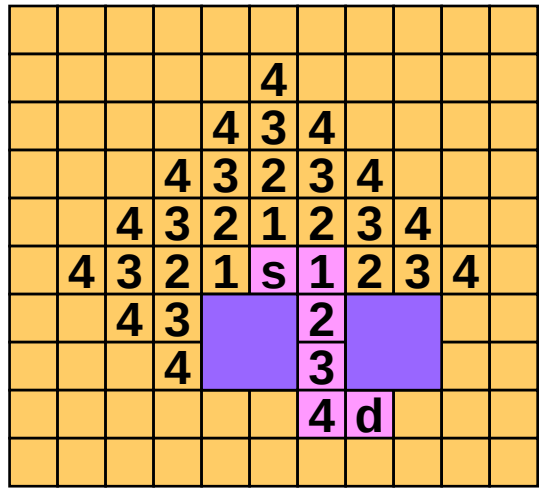
    --> minimize area occupied by routing
  - Minimize number of vias
    - yield, resistance/capacitance, area
  - Minimize (maximize) use of particular layers

# Unrestricted Routing

- **Lee-Moore Algorithm (Maze Routing)**
  - **Fixed grid, Manhattan**
  - **One net at a time**
  - **Problems**
    - **Dependency on net ordering**
    - **Large memory requirement**

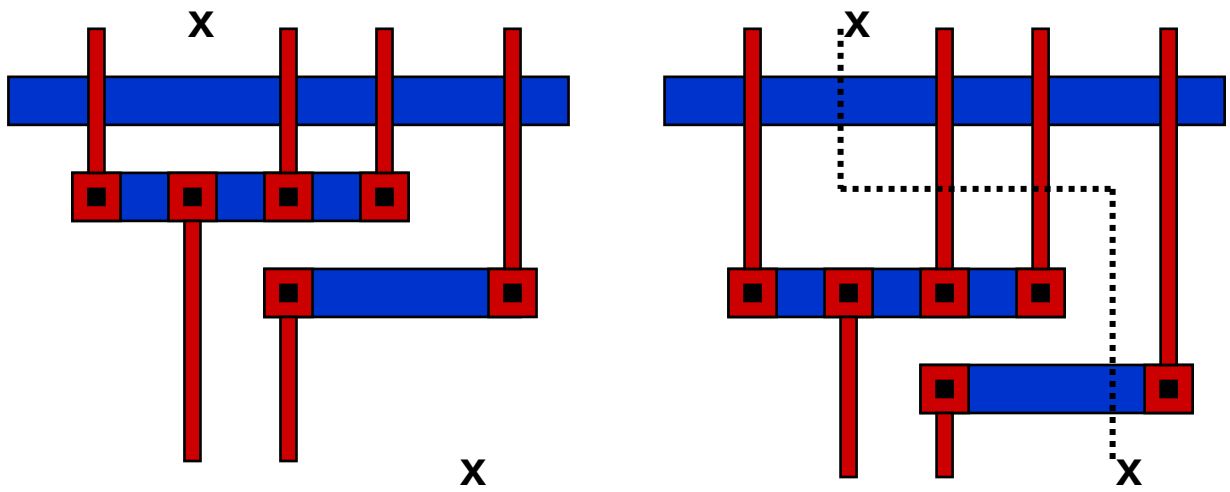
      **<-- one storage element (2 bits) for every cell**
    - **Long search time**

      **<-- $\propto$(connection length)$^2$**
  - **Improvement by rip-up and re-route**

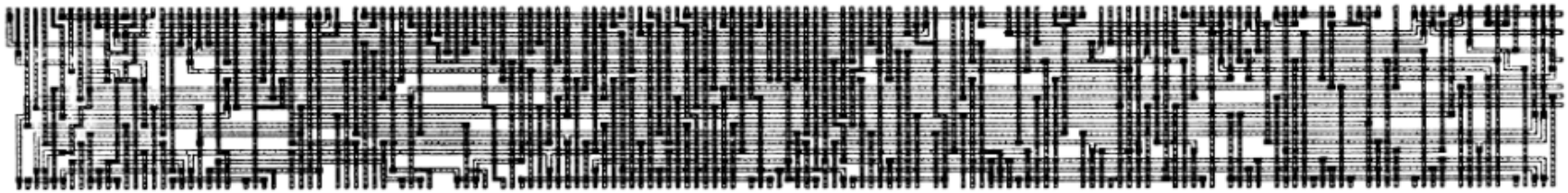


- **Rip-up and Re-route**
  - **H.Shin and A.Sangiovanni-Vincentelli, "Mighty: A 'rip-up and re-route' detailed router," *Proc. ICCAD*, 1986**
  - **Incremental router**
  - **Path finder**
    - **Maze routing is used to find the minimum cost path between two pins**
    - **Build a list of nets with the order of ascending cost**
    - **Vertical (horizontal) wire on horizontal (vertical) layer is penalized**
    - **Changing a layer is penalized to minimize the number of vias**
  - **Path conformer**
    - **After all paths have been found, implement the path according to the order**
    - **If the path is not feasible, call path finder**
    - **If no path is found or cost is too high, call weak modifier**

– **Weak modifier**
  - **Pushes existing wires to make space**
  - **No solution --> call strong modifier**

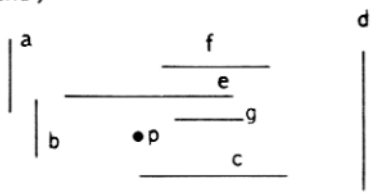


- **Strong modifier**
  - **Removes blocking nets to make space**
- **Post processing**
  - **All the nets are re-routed from the longest net**
  - **Vertical (horizontal) wires on horizontal (vertical) layer are less penalized**
- **Example**
  - **176 sec on VAX 11/785**
  - **31 weak modification**
  - **9 strong modification**



- **Line Routing**
  - **D.W.Hightower, "A solution to line routing problems on the continuous plane," *Proc. 6th Design Automation Workshop*, 1969**
  - **Gridless (often implemented on a grid)**
  - **Manhattan**
  - **Definitions**
    - **Escape line: pair of orthogonal lines passing through a point**
    - **Cover: Blockage of a point. Intersects with an escape line.**
    - **Escape point: A point on an escape line that is not covered by a horizontal (vertical) cover**

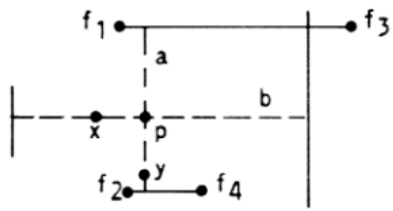
1. COVER (VERB)




LINE SEGMENTS b,c,d,e COVER p.
LINE SEGMENTS a,f AND g DO NOT.

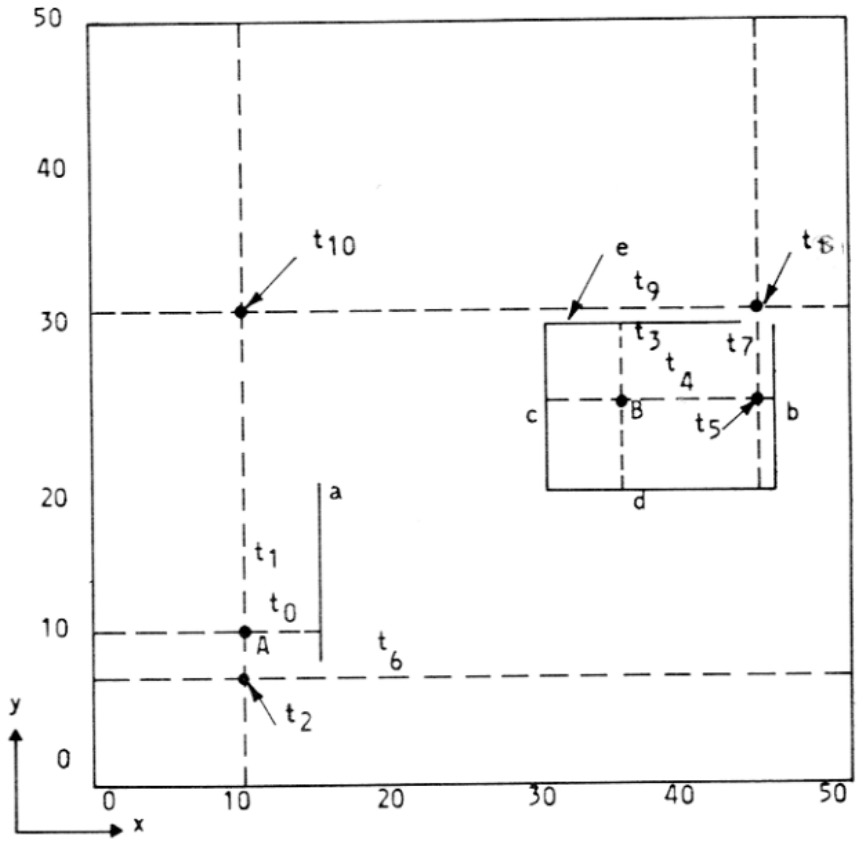
2. COVER (NOUN)
IN THE ABOVE DRAWING,
LINE SEGMENTS e AND c ARE HORIZONTAL COVERS
LINE SEGMENTS b AND d ARE VERTICAL COVERS

3. ESCAPE LINES




LINE SEGMENT a IS THE VERTICAL ESCAPE LINE.
LINE SEGMENT b IS THE HORIZONTAL ESCAPE LINE
POINT x IS A VERTICAL ESCAPE POINT.
POINT y IS A HORIZONTAL ESCAPE POINT.

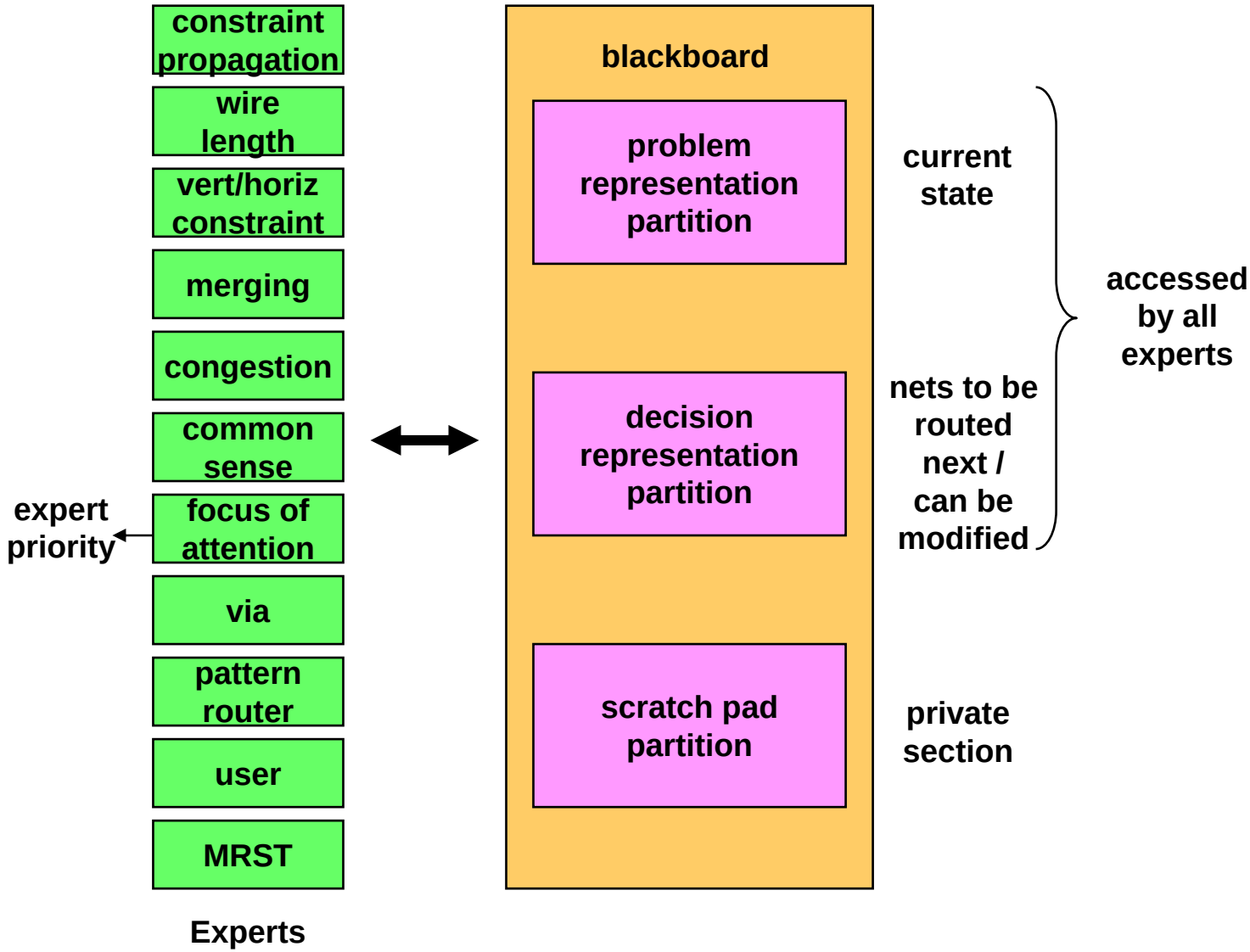
- **Algorithm**
  - **Starting from the source and sink, generate escape lines and determine escape points until escape lines intersect**



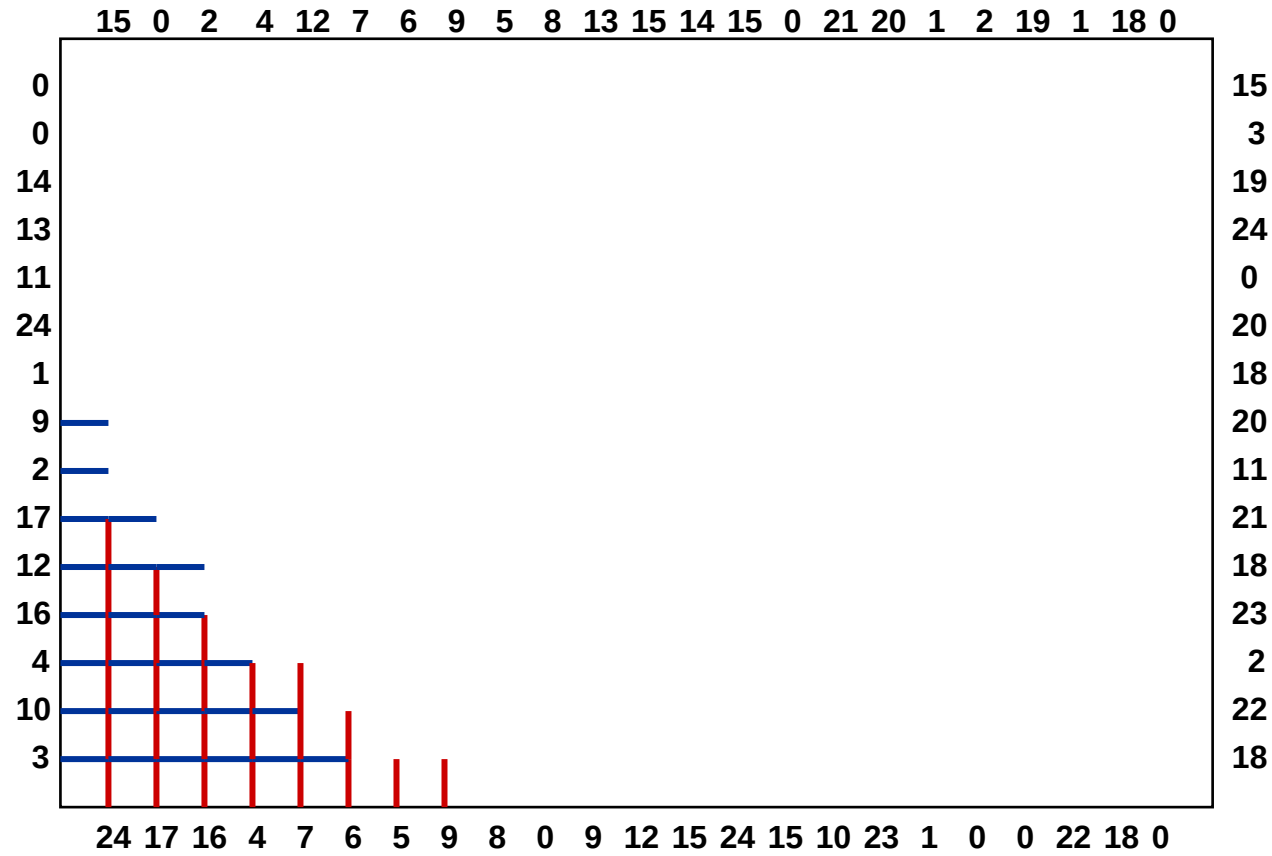
# Switch-Box Routing

- **Weaver**
  - **R.Joobbani and D.Siewiorek, "Weaver: a knowledge-based routing expert," *IEEE Design and Test of Computers*, Feb. 1986**
  - **Switch-box + channel routing**
  - **Knowledge-based expert system**
  - **Algorithmic part(C) + 700 rules(OPS5)**
  - **11 experts (including user)**
  - **Interactive**
    - **User can override system decision**
    - **Pre-route or delete wiring segments**
  - **Long running time**

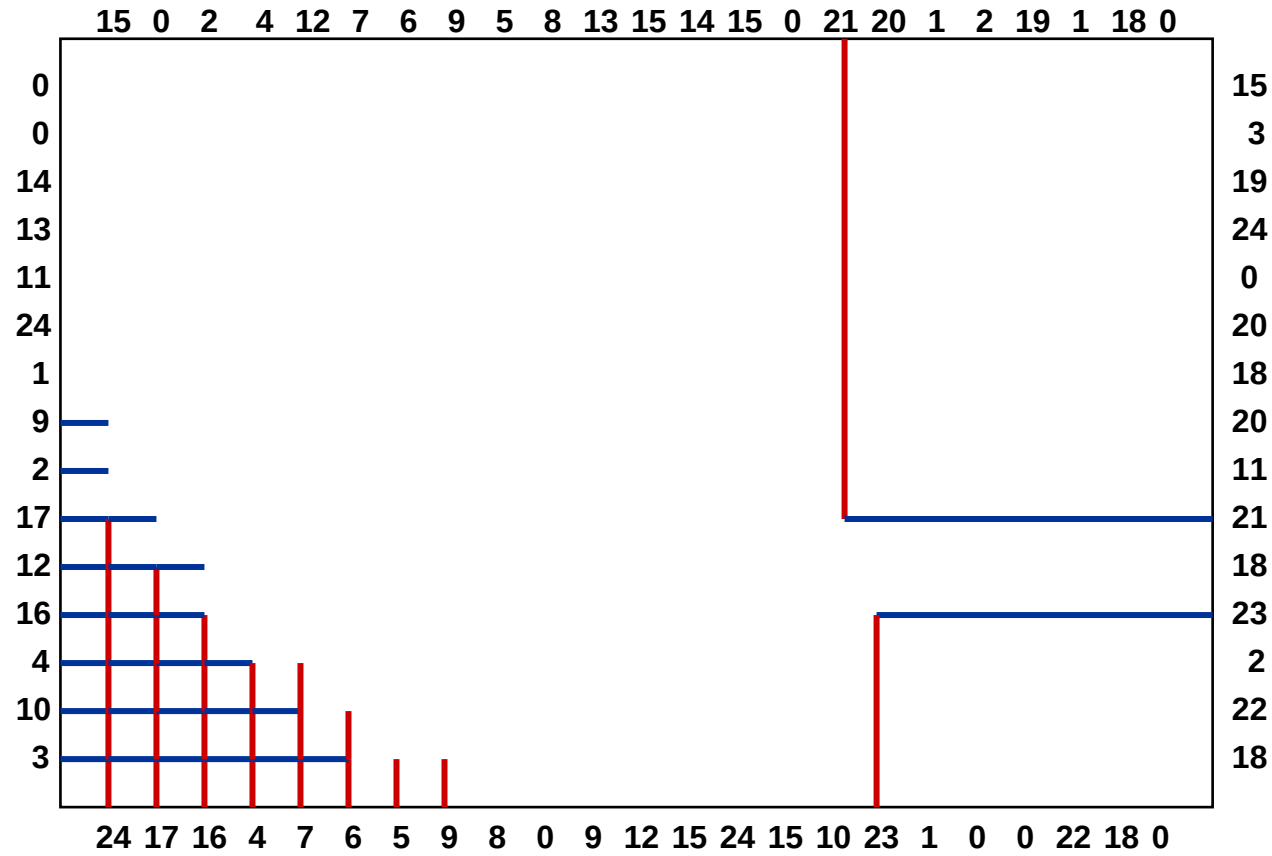
# – Weaver architecture



## – Constraint propagation

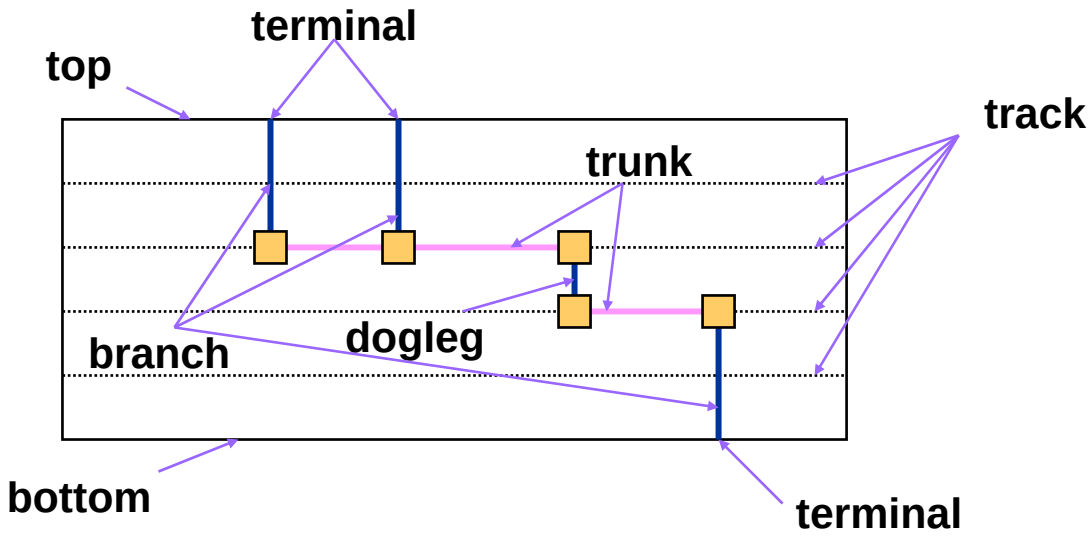


– **Pattern router**

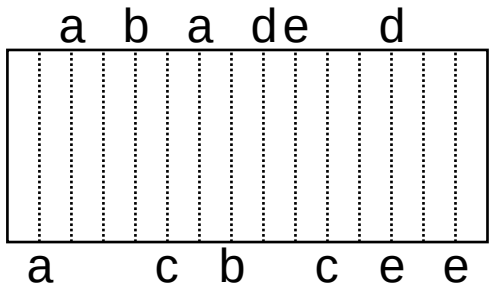


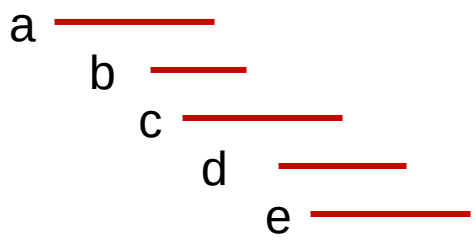
# Channel Routing

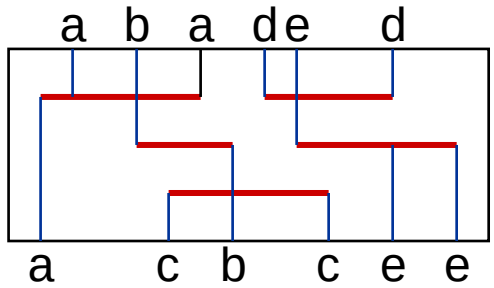
- **Problem Formulation**
  - **Assumption (original formulation)**
    - **Rectangular routing region with no obstructions**
    - **Fixed pins on two opposite sides**
    - **Floating pins on the other two sides**
    - **Two layers for interconnections**
    - **Pins are placed on a regular grid**
    - **Channels are subdivided into rows (tracks)**
  - **Minimize number of tracks and number of vias**

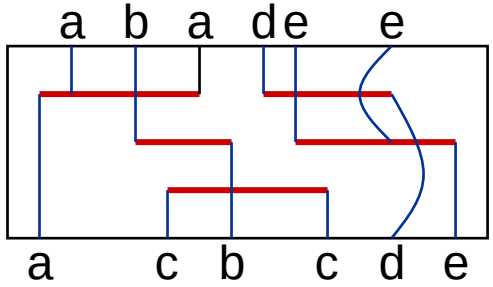


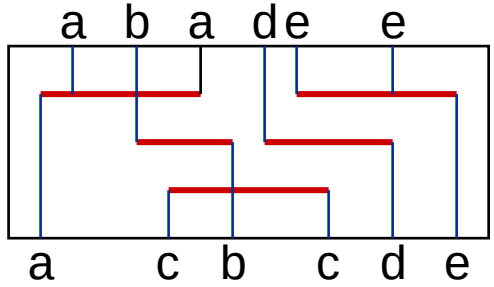
- **Algorithms**
  - **Left edge algorithm**
    - **At most one trunk per net**
    - **Compute intervals for each net**
    - **Sort the intervals in ascending order of left edge's location**
    - **Assign intervals to available tracks**
    - **Overlap problem**



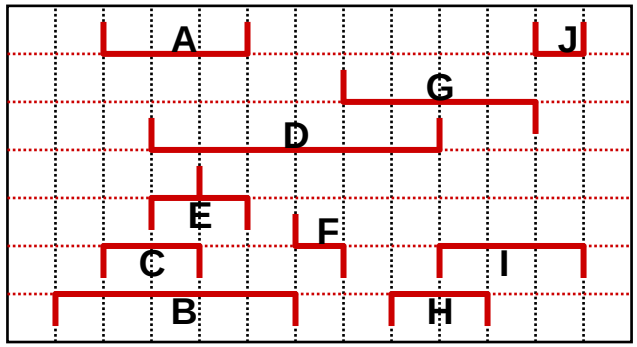



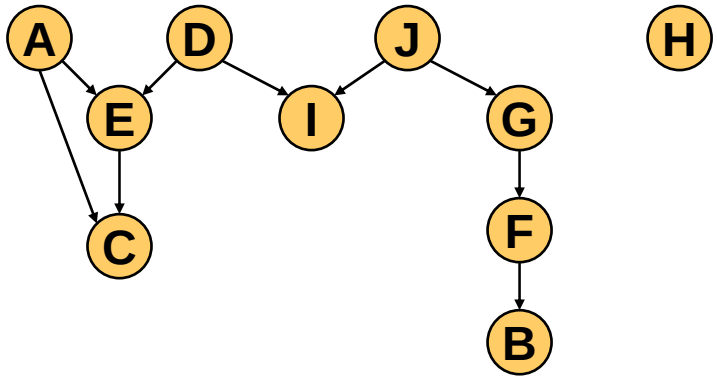



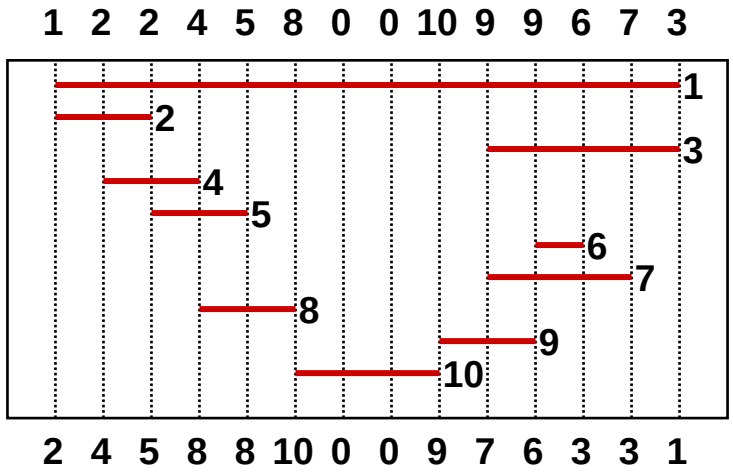



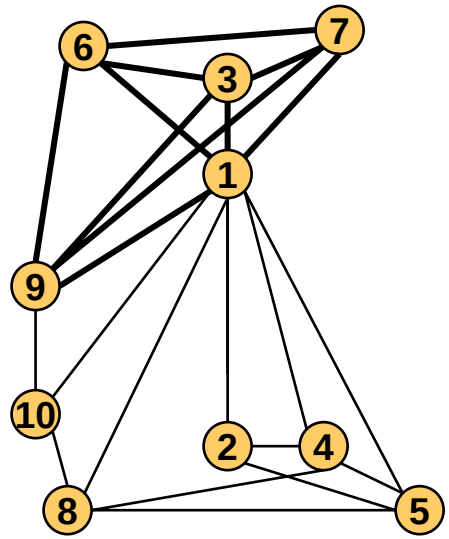
- **Constrained left edge algorithm**
  - **VCG (Vertical Constraint Graph)**
    - **Vertex: net**
    - **Edge (directed): from a net connecting a top terminal to a net connecting a bottom terminal on the same column**



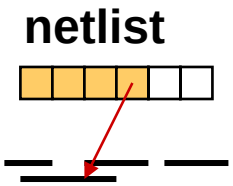



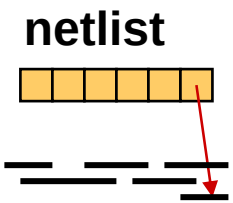
- **HCG (Horizontal Constraint Graph)**
  - **Vertex: Net**
  - **Edge: Intersection between two intervals**
  - **Interval: Leftmost and rightmost column of a net**
  - **Density: Maximum number of intervals crossing a column**
    **--> size of the largest clique in HCG**
    - **Density can be used as a lower bound on number of tracks**
  - **Interval graph**



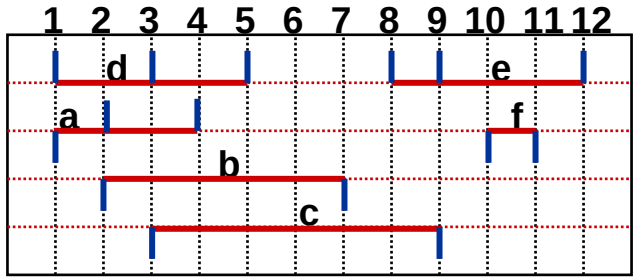
- **Branch and bound**
  - **B.W.Kernighan, D.G.Schweikert, and G.Persky, "An optimum channel-routing algorithm for polycell layouts of integrated circuits," *Proc. 10th Design Automation Workshop*, 1973**
  - **Select a leftmost net that satisfies the vertical constraint**
  - **If the current track t is filled, compute the lower bound b**
  - **If t+b ≥ V (V is the best solution seen so far), replace the most recently placed net by the next unplaced net**

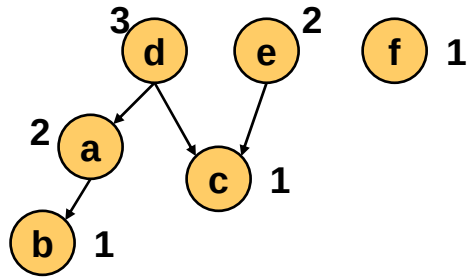



```
s=0; t=1; V=n;
place: do while track t is not full & last net is not placed
   place next leftmost legal net on track t;
   s=s+1; netlist(s)=net placed;
if track t is full
   compute b=lower bound on tracks needed for remaining nets;
   if t+b < V
      t=t+1; goto place;
   unplace netlist(s); goto place;
if n-th net is placed
   record new solution; V=t;
   unplace netlist(s); unplace netlist(s);
   goto place;
```

- **Computation of lower bound**
  - **In the vertical constraint graph, static lower bound s of a net is computed by (largest lower bound of the net's children) + 1**



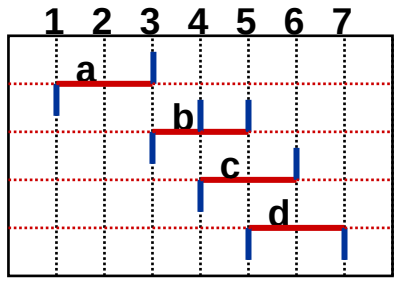

  - **Dynamic lower bound b is computed considering the static bound and overlap of unassigned nets (matrix is used)**
  - **If a net spanning from column L to R has static bound of s, then add 1 to matrix components M(s,L),...,M(s,R)**
  - **Algorithm**

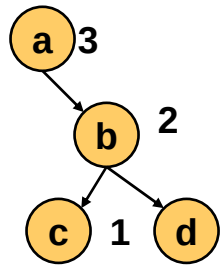
```
for j = 1, max_column
    b(j) = 0;
    for i = 1, max_s
        b(j) = max(i, b(j)+M(i,j));
    end for;
end for;
b = max_j b(j);
```

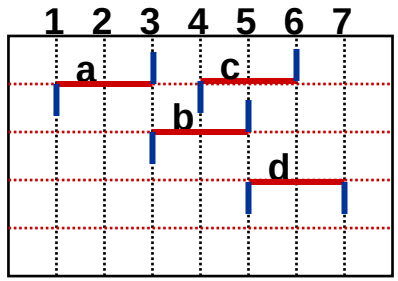
column

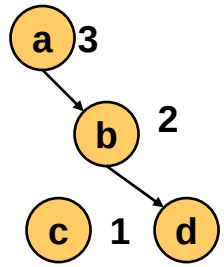
| s | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 3 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 |
| 1 | 0 | 1 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 1 | 0 |

– **The dynamic lower bound is not exact since it is computed column-wise.**



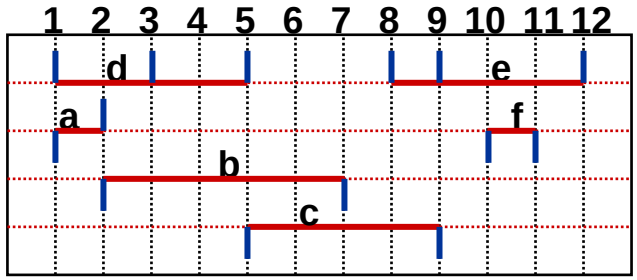



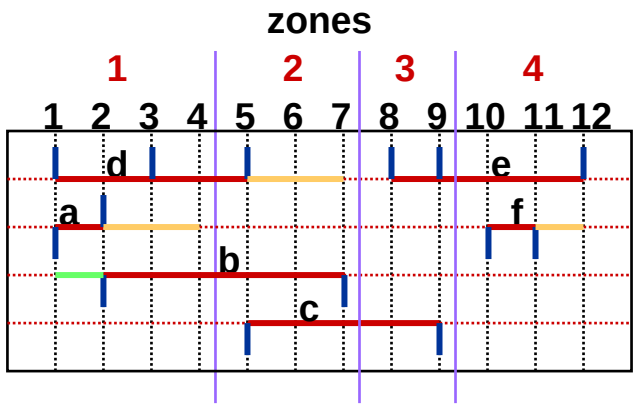



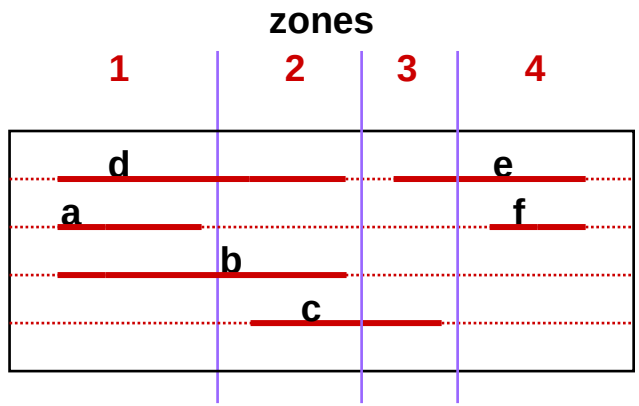




| | column | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|---|
| | 3 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| s | 2 | 0 | 0 | 1 | 1 | 1 | 0 | 0 |
| | 1 | 0 | 0 | 0 | 1 | 2 | 2 | 1 |

– **Zone**


1 2 3 4 5 6 7 8 9 10 11 12
d
e
a
f
b
c


zones
1
2
3
4
1 2 3 4 5 6 7 8 9 10 11 12
d
e
a
f
b
c


zones
1
2
3
4
d
e
a
f
b
c

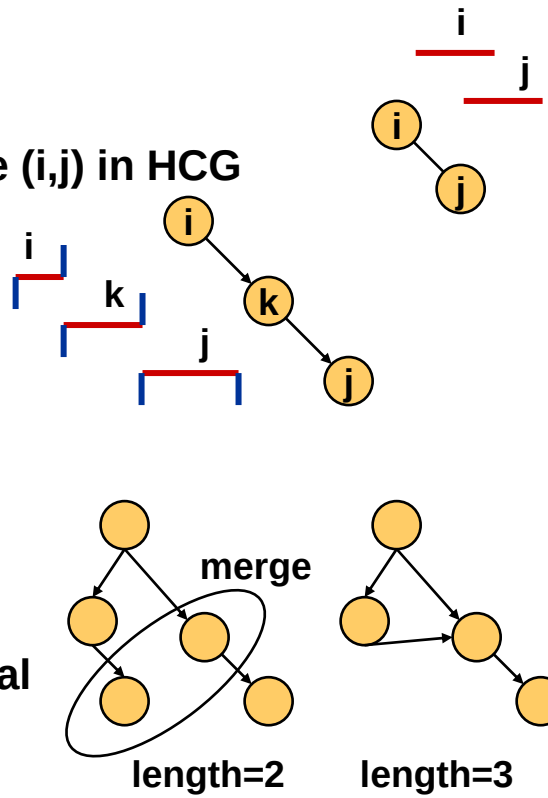
- **Speed improvement to branch and bound algorithm**
  - **T.Yoshimura and E.S.Kuh, "Efficient algorithms for channel routing," *IEEE Trans. on CAD of ICAS*, Jan. 1982**
  - **Branch and bound technique gives the optimum solution to the restricted (no dogleg) channel routing problem which is**

    **NP-complete --> long runtime**
  - **Net merging (merged nets --> same track) simplifies the problem (smaller VCG)**
  - **Candidate nets (i,j) for merging**
    - **No overlap between i and j --> No edge (i,j) in HCG**
    - **There is no path from i to j in VCG**
  - **Algorithm:**

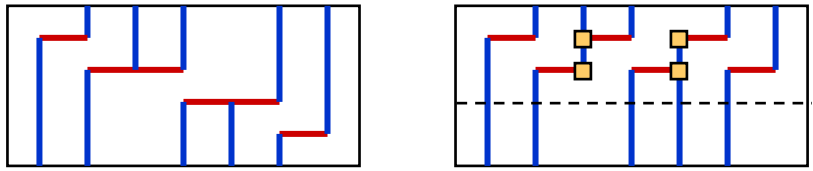
    **Select and merge nets iteratively**

    **Assign merged nets to tracks**

    **Separate nets**
  - **Heuristic**
    - **Select nets to be merged so that longest path in reduced VCG is minimal**



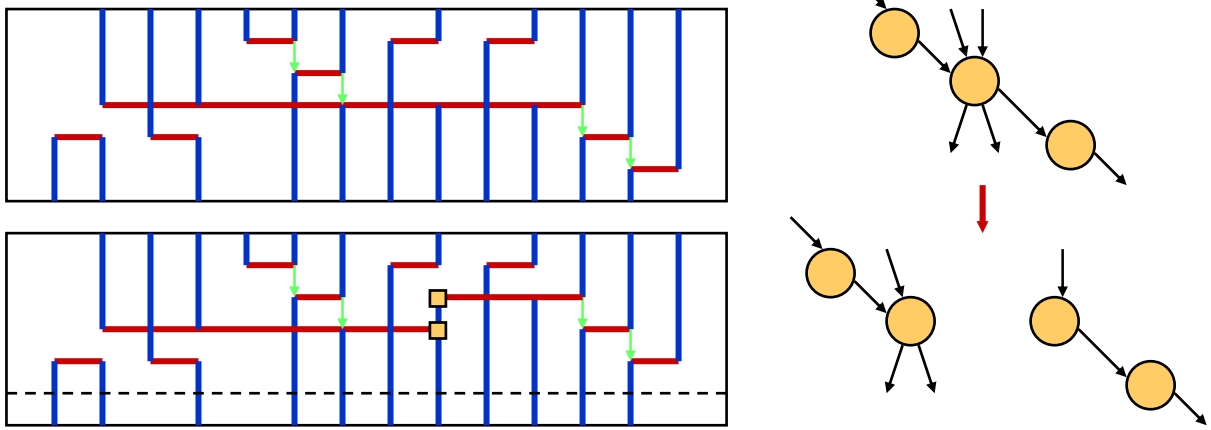
- **Dogleg channel router**
  - **D.N.Deutsch, "A 'dogleg' channel router," *13th Design Automation Conf.*, June 1976**
  - **Doglegs are allowed**



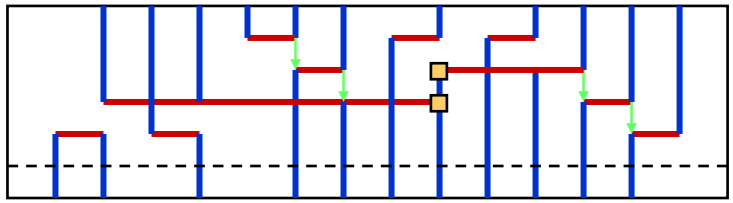
  - **Observations**
    - **Usually a few crucial nets such as clock lines are heavily connected to both sides of the channel**

      **--> long constraint chain**

      **--> require many tracks**



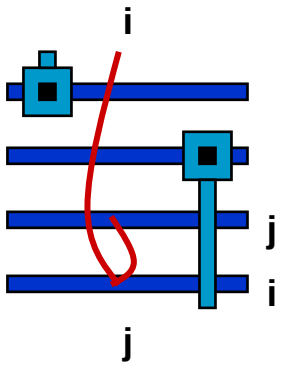
- **Doglegs are introduced only at terminal positions of the net**
- **Divide long connections into a series of two-pin connections**
- **To avoid generating too many doglegs, 'range' concept is used**
  - **Range: Number of consecutive trunk segments that must be assigned to the same track**



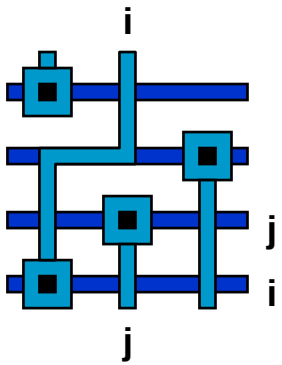
- **Jog insertion**
  - **J.Reed, A.Sangiovanni-Vincentelli, and M.Santomauro,"A new symbolic channel router: YACR2," *IEEE Trans. on CAD*, July 1985**
  - **Allows horizontal jogs on vertical layer and vertical jogs on horizontal layer**
  - **Algorithm**
    - **Horizontal track assignment**

      **--> Uses modified LEA to assign nets to tracks minimizing number of vertical constraint violations (allows vertical constraint violations)**
    - **Maze routing**
    - **If routing fails, add a track and restart**
  - **Preprocessor translates pin locations into a symbolic grid.**
  - **Postprocessor translates the symbolic routing to geometry.**

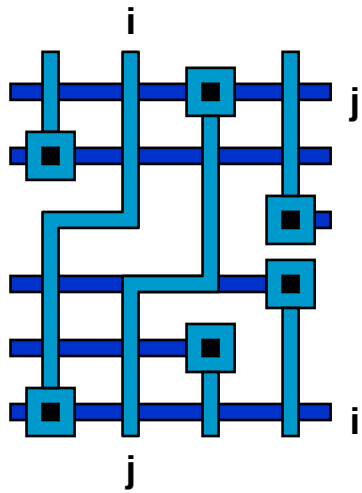
- **Maze1 routing**




**vertical constraint violation**

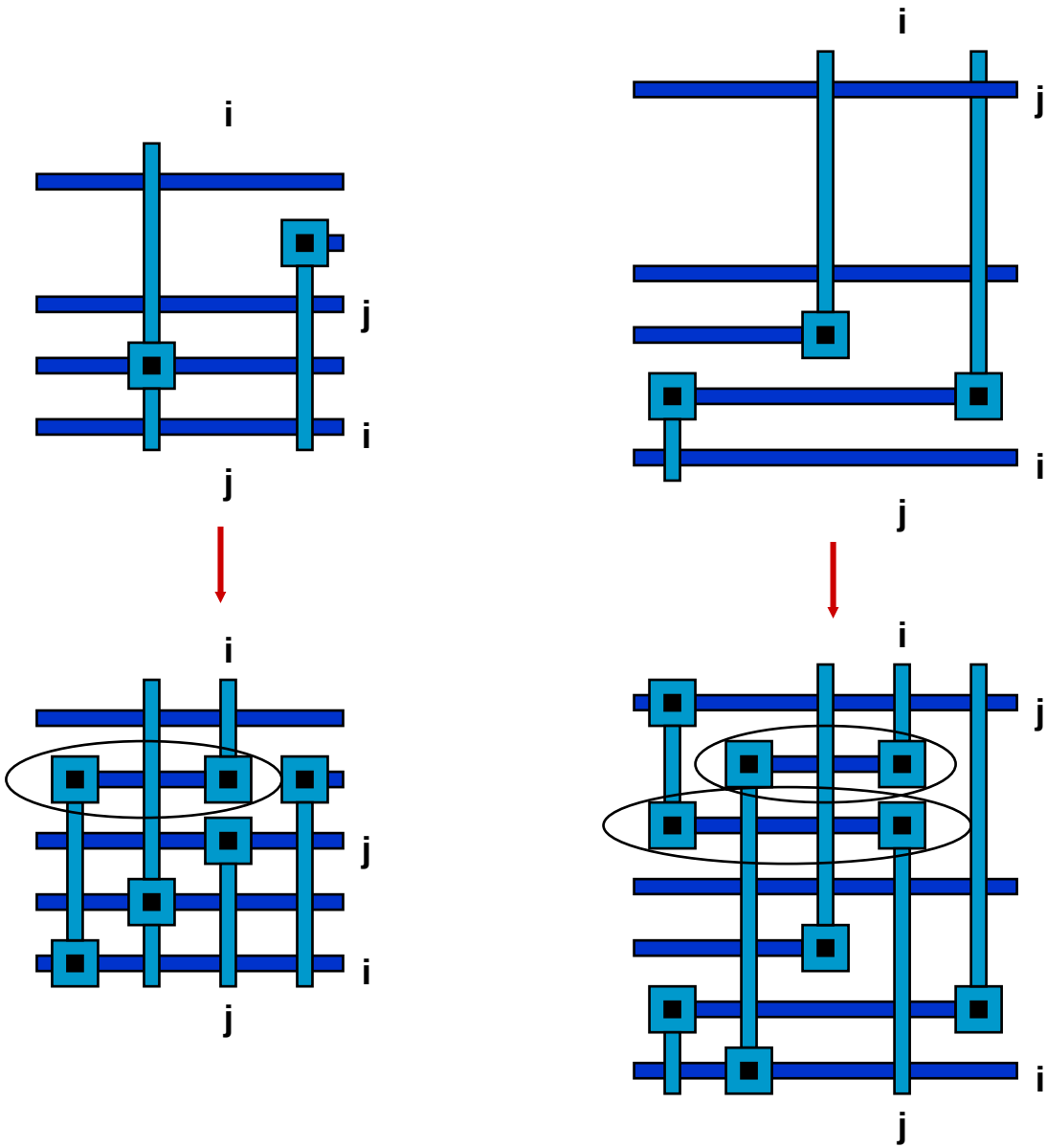



**after maze routing**




**more complex example**

- **Maze2, Maze3 routing**

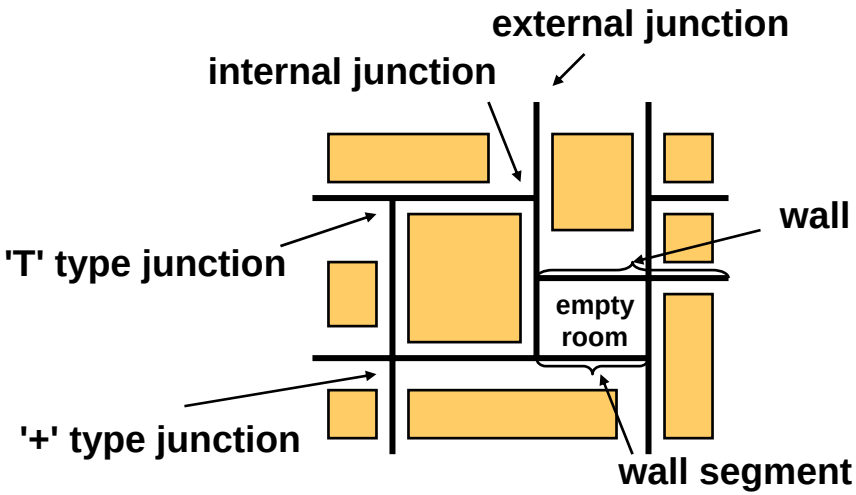


# Channel Definition and Ordering

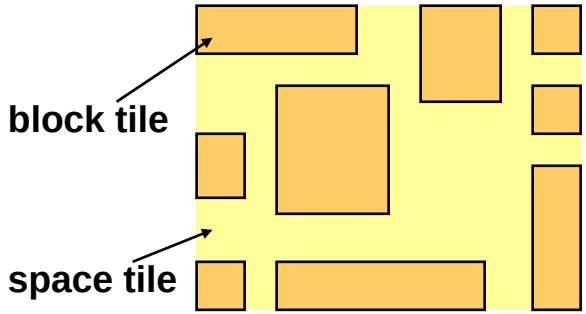
- **Routing Region Definition and Ordering**
  - **W.M.Dai, T.Asano, and E.S.Kuh, "Routing region definition and ordering scheme for building-block layout," *IEEE Trans. on CAD*, July, 1985**
  - **Channel routers are the most effective detailed routers**
  - **Tasks**
    - **Partition layout area into module area and routing area**
    - **Identify channels for routing**
    - **Determine order of channels to be routed**

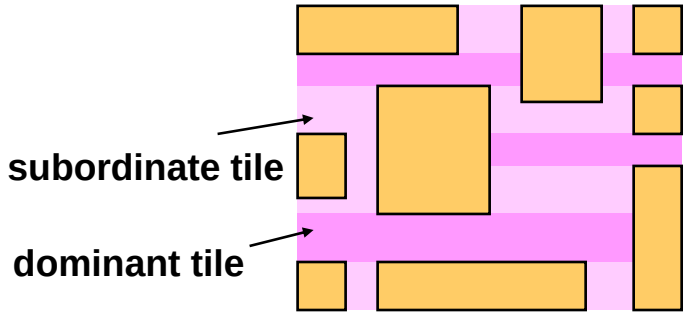
- Supports non-slicing structures
- Define tiles
  - Block tiles (for modules)
  - Space tiles (for routing)
- Define maximal space tiles
  - Horizontal tile plane
  - Vertical tile plane
- Replace dominant tiles by walls

  --> Floor plan graph

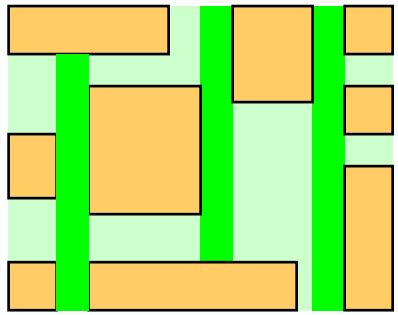



Floor plan graph



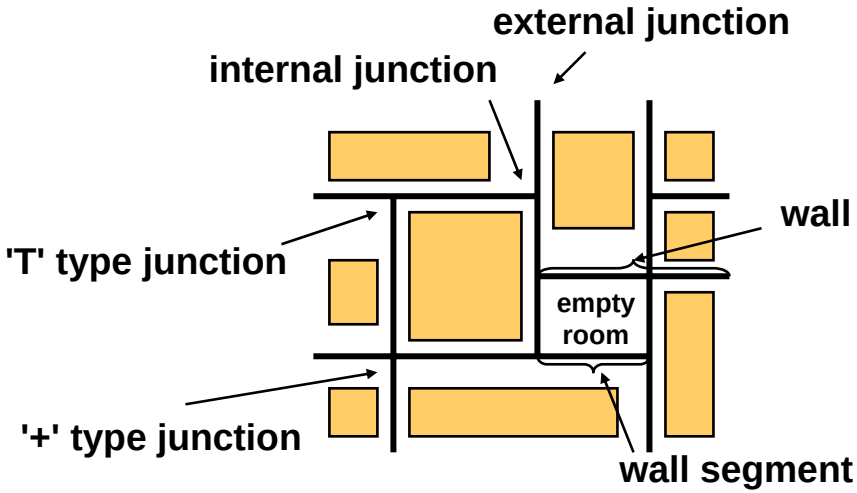

Horizontal tile plane



Vertical tile plane

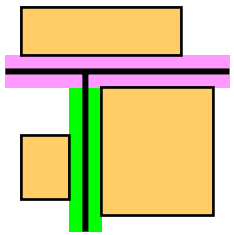
- **Wall segments: channels**
- **Wall precedence relation**

  **--> ordering**

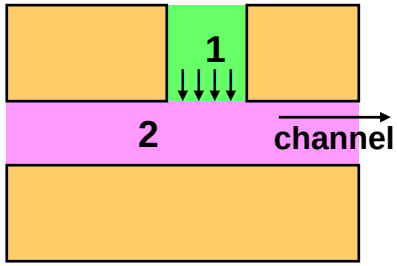



Floor plan graph

- **T junction: vertical segment before horizontal segment**



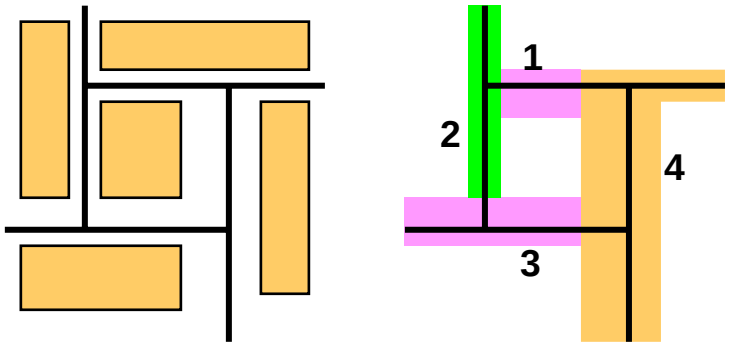
- **Requirements for channel routing**
  - **Pin definition requirement: Positions of all pins along the two edges of a channel must be fixed**
  - **Rigidity requirement: A channel already routed cannot be altered in its channel direction**



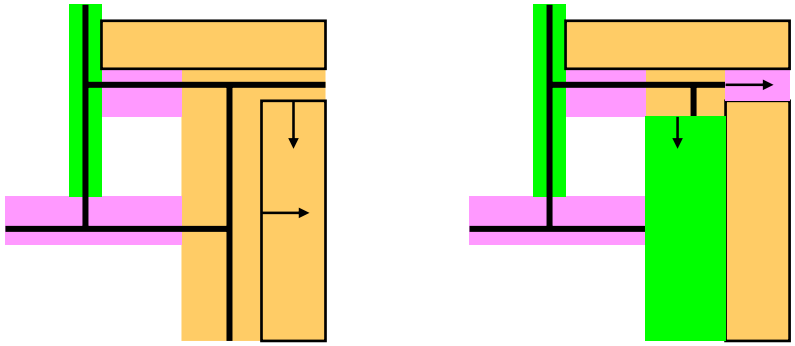

**Channel 1 must be routed first**

- **Cycle in the precedence relation**

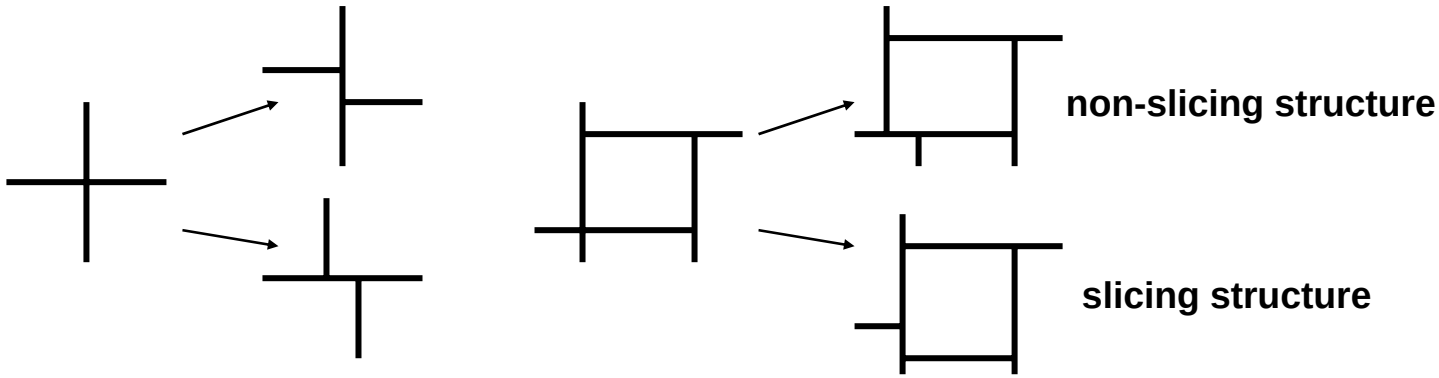
  **--> L-shaped channel to break the cycle**



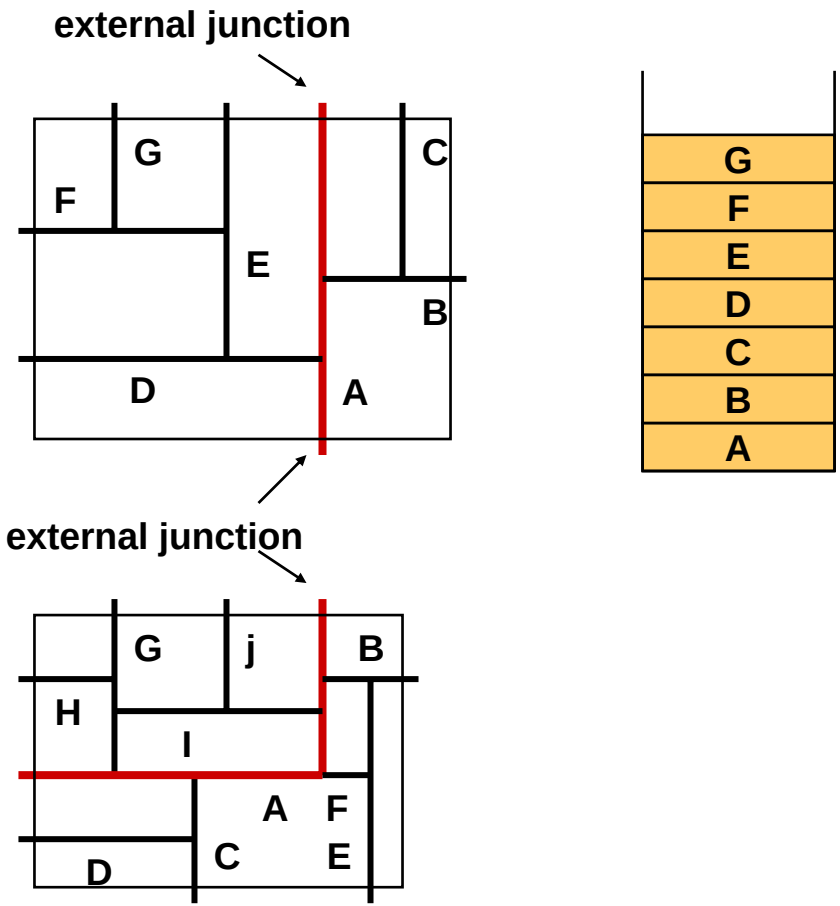
– **Width of L-shaped channels can be adjusted without destroying previously routed channels (how about switchbox?)**



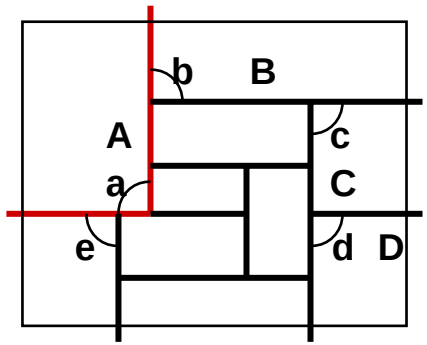
– **'+' type junction**
  - **Needs normalization**
  - **Slicing structure provides acyclic precedence relation**

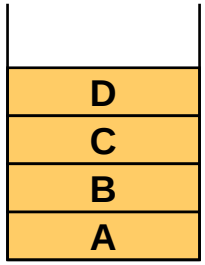


- **Algorithm**
  - **Wall slicing and Corner cutting**
  - **Channel definition stack**

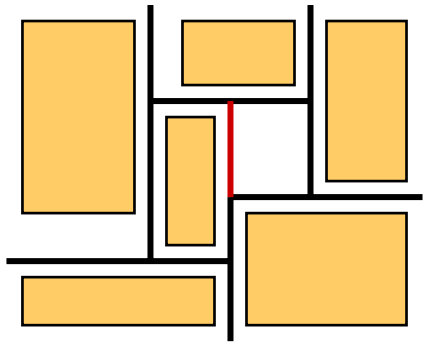


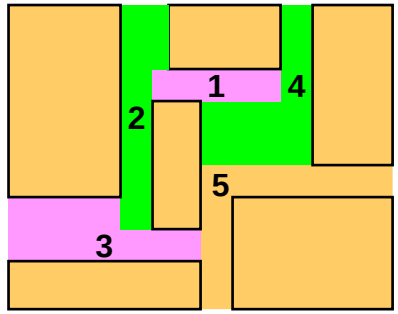
- **Corner dependency**




- **Empty room**
  - **One of the four wall segments adjacent to an empty room is ignored**





# Global Routing

- **Goal**
  - **Distribute nets over channels**
  - **Minimize**
    - **Total net length**
    - **Critical net lengths**
    - **Congestion**
- **Approaches**
  - **One net at a time**
    - **Depends on order**
    - **Cost function**

      $C = aL + b / c^{T+1}$

      **L: channel length**

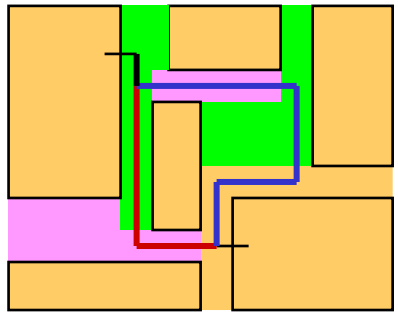
      **T: Tracks available**
  - **All nets at once**
    - **0-1 linear program**
    - **Simulated annealing**
    - **Hierarchical routing**

- **Integer (0-1) linear programming**
  - **Assume two-terminal nets (can be generalized)**

$P_n$ : Set of paths for net n

$$a_{np}^{c} = \begin{cases} 1 \text{ if path p in } P_n \text{ uses channel c} \\ 0 \text{ otherwise} \end{cases}$$

$$x_{np} = \begin{cases} 1 \text{ if net n uses path p in } P_n \\ 0 \text{ otherwise} \end{cases}$$



  - **Minimize cost**

$$\sum_{c=1}^{C} L_c \sum_{n=1}^{N} \sum_{p=1}^{|P_n|} a_{np}^{c} \, x_{np}$$

  - **Subject to constraints**

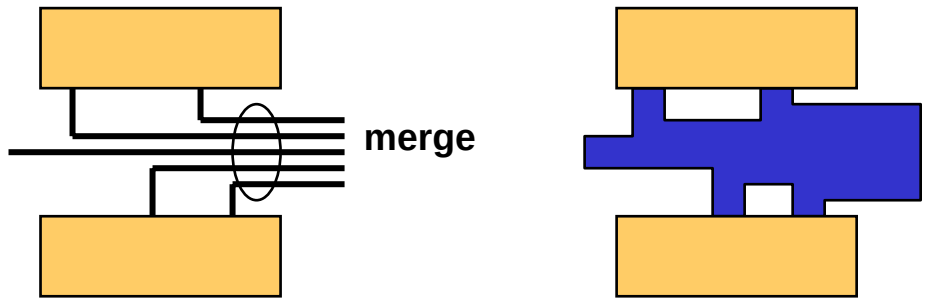
$$x_{np} \geq 0, n = 1,2,...,N, p = 1,2,...,|P_n|$$

$$\sum_{p=1}^{|P_n|} x_{np} = 1, n = 1,2,...,N$$

$$\sum_{n=1}^{N} \sum_{p=1}^{|P_n|} a_{np}^{c} \, x_{np} \leq T_c, c = 1,2,...,C$$

# Power and Ground Routing

- **Special Routing**
  - **Usually on a single layer (no vias)**
    **--> planar routing**
  - **Usually much wider (more current)**
- **MOSAICO**
  - **Macrocell place and route system**
  - **Assume power and ground rings around the chip**
  - **Before placement, decompose power and ground nets into smaller nets**
  - **Merge the nets after global routing**
  - **Find power requirement**
  - **After symbolic detailed routing, find the sub-net (power and ground) widths**



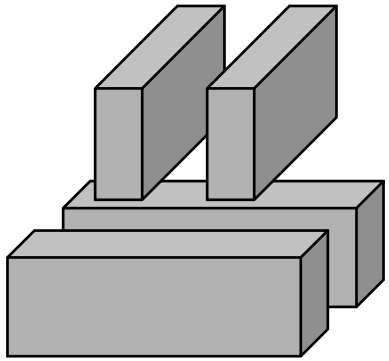
# Cross-Coupling

- **Problem**
  - **Routing in deep sub-micron design**
    - **Thin wire**
      - **Large interconnect delay**
      - **delay $\propto$ (wire length)$^2$**

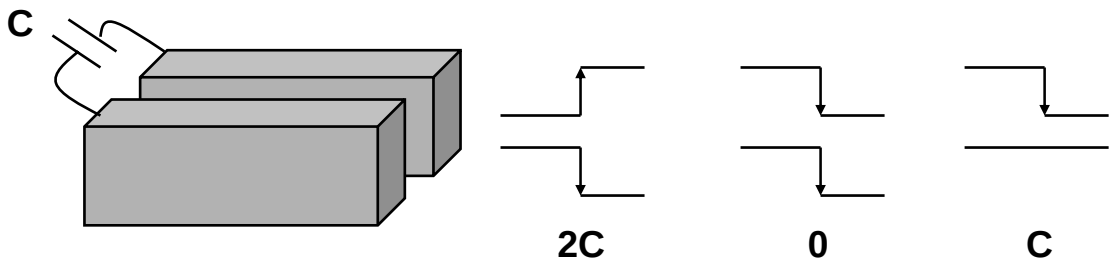
      **--> Al --> Cu**
    - **H > W**
      - **Delay due to larger edge (horizontal) capacitance**
      - **Cross-talk with neighboring wires**

      **--> Permutation of signal wires or**

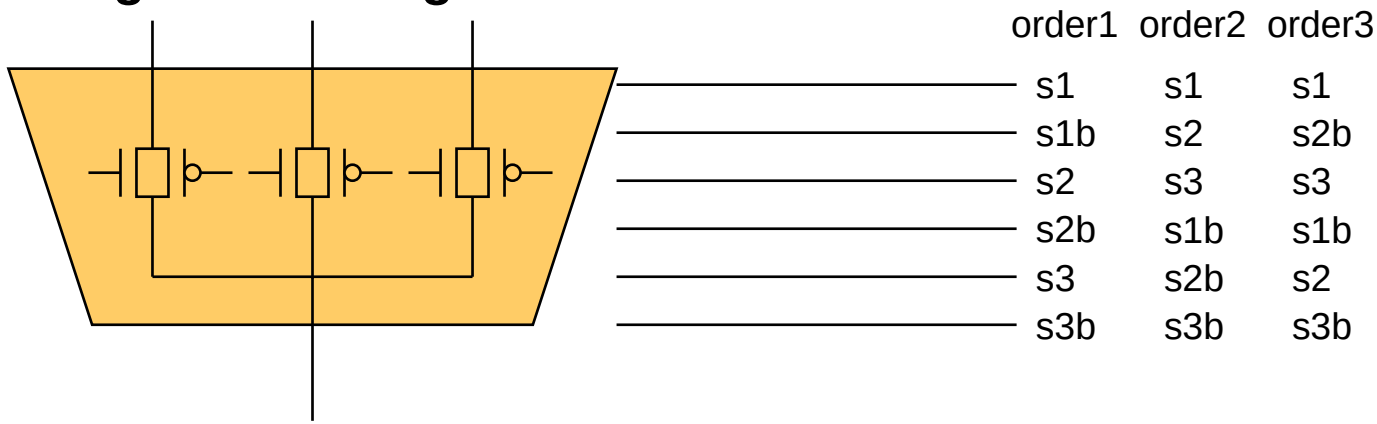
      **Interleaving with power and ground wires**



- **Reduction of Cross-Coupling**
  - **Joon-Seo Yim and Chong-Min Kyung, "Reducing cross-coupling among interconnect wires in deep-submicron datapath design," *Proc. 36th Design Automation Conf.*, June 1999**
  - **Miller effect**




  - **Control signal ordering**



Cross−Coupling

| | selection signal change | | | | | | total # of opposite transitions |
|---|---|---|---|---|---|---|---|
| | 1->2 | 1->3 | 2->1 | 2->3 | 3->1 | 3->2 | |
| s1 | ↓ | ↓ | ↑ | 0 | ↑ | 0 | 12 |
| s1b | ↑ | ↑ | ↓ | 1 | ↓ | 1 | |
| s2 | ↑ | 0 | ↓ | ↓ | 0 | ↑ | |
| s2b | ↓ | 1 | ↑ | ↑ | 1 | ↓ | |
| s3 | 0 | ↑ | 0 | ↑ | ↓ | ↓ | |
| s3b | 1 | ↓ | 1 | ↓ | ↑ | ↑ | |
| s1 | ↓ | ↓ | ↑ | 0 | ↑ | 0 | 8 |
| s2 | ↑ | 0 | ↓ | ↓ | 0 | ↑ | |
| s3 | 0 | ↑ | 0 | ↑ | ↓ | ↓ | |
| s1b | ↑ | ↑ | ↓ | 1 | ↓ | 1 | |
| s2b | ↓ | 1 | ↑ | ↑ | 1 | ↓ | |
| s3b | 1 | ↓ | 1 | ↓ | ↑ | ↑ | |
| s1 | ↓ | ↓ | ↑ | 0 | ↑ | 0 | 0 |
| s2b | ↓ | 1 | ↑ | ↑ | 1 | ↓ | |
| s3 | 0 | ↑ | 0 | ↑ | ↓ | ↓ | |
| s1b | ↑ | ↑ | ↓ | 1 | ↓ | 1 | |
| s2 | ↑ | 0 | ↓ | ↓ | 0 | ↑ | |
| s3b | 1 | ↓ | 1 | ↓ | ↑ | ↑ | |