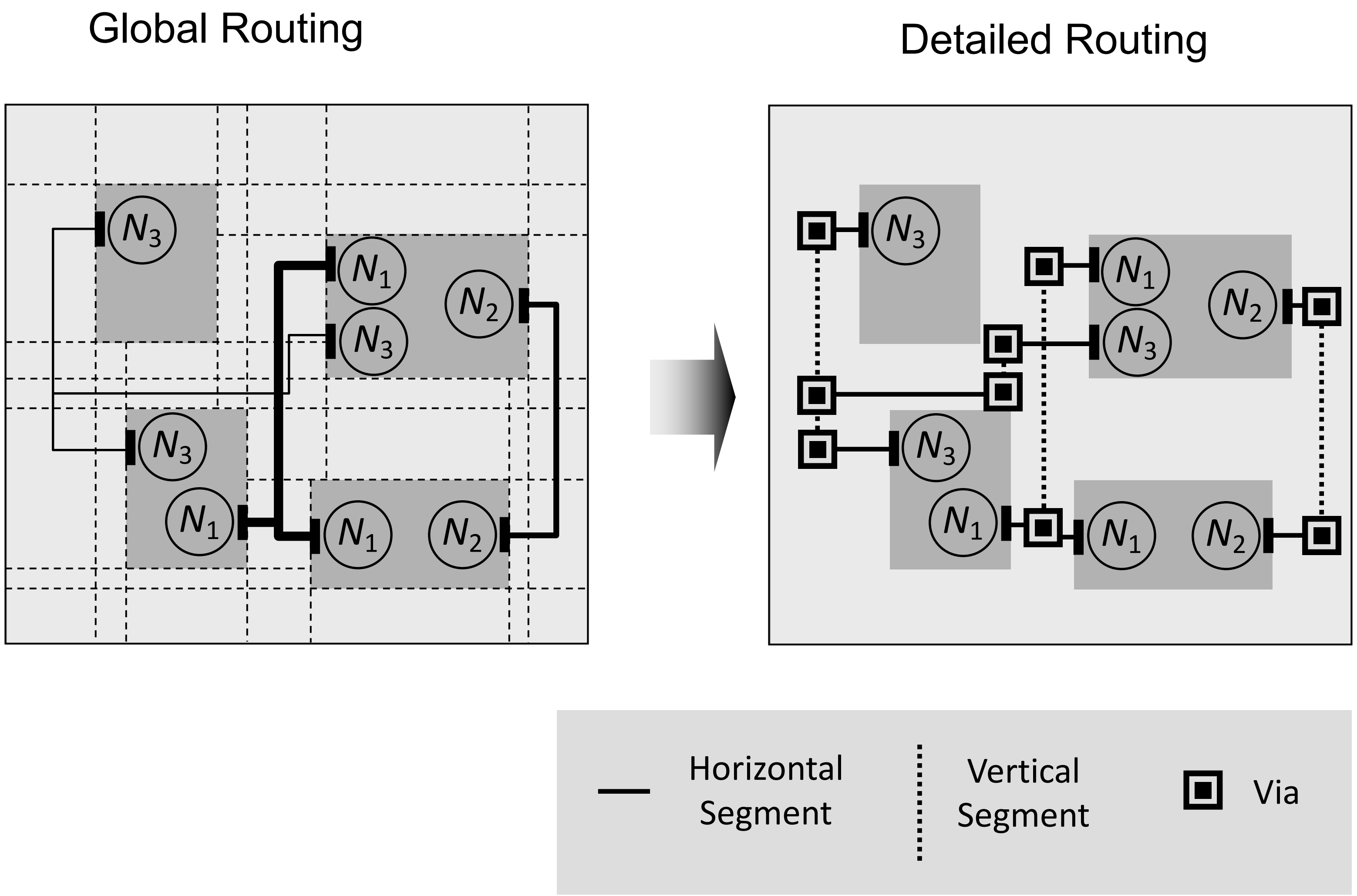
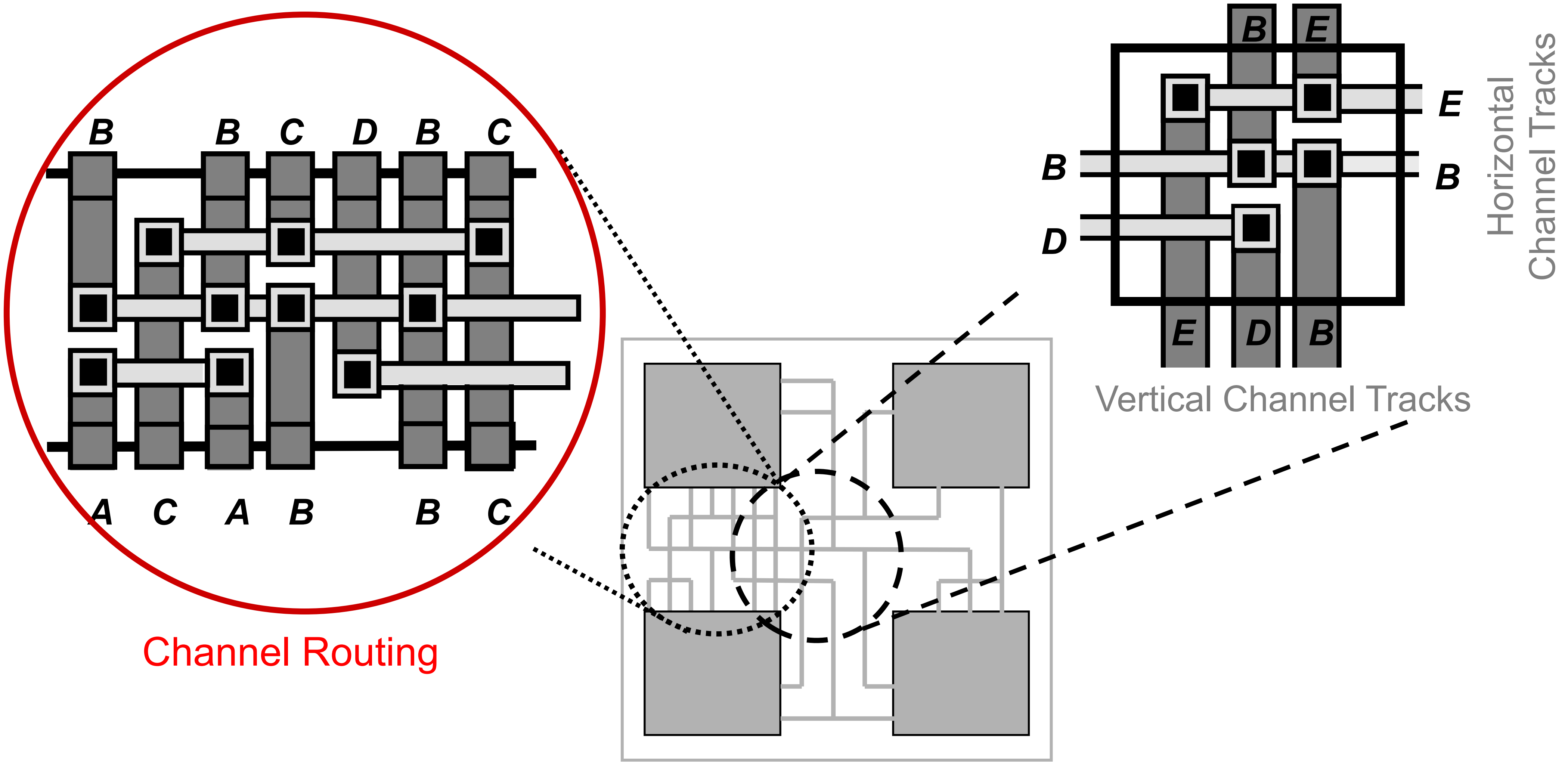
Lecture 18: Routing – III

Tsung-Wei (TW) Huang
Department of Electrical and Computer Engineering
University of Utah, Salt Lake City, UT

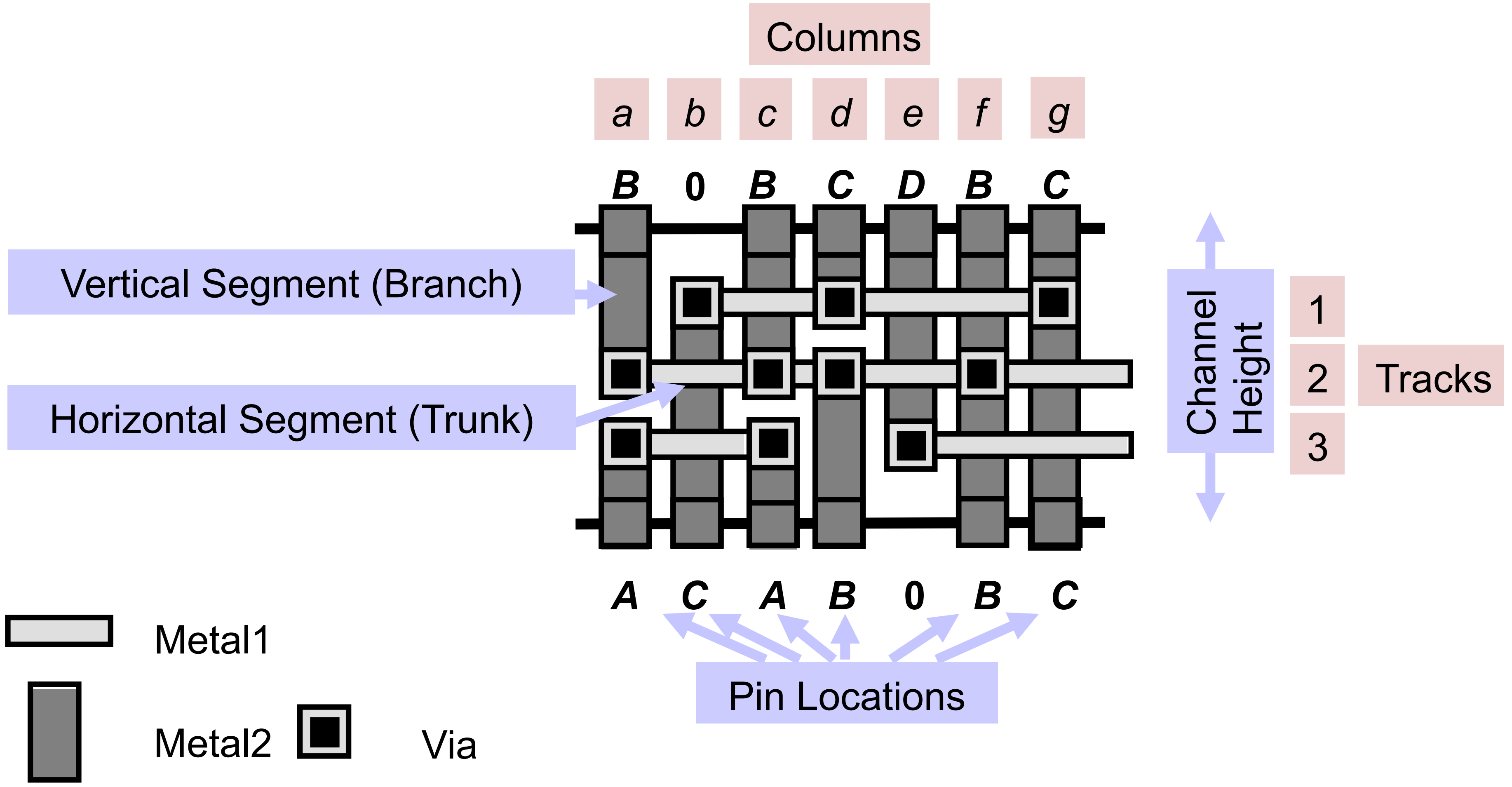
Recap: Global and Detailed Routing



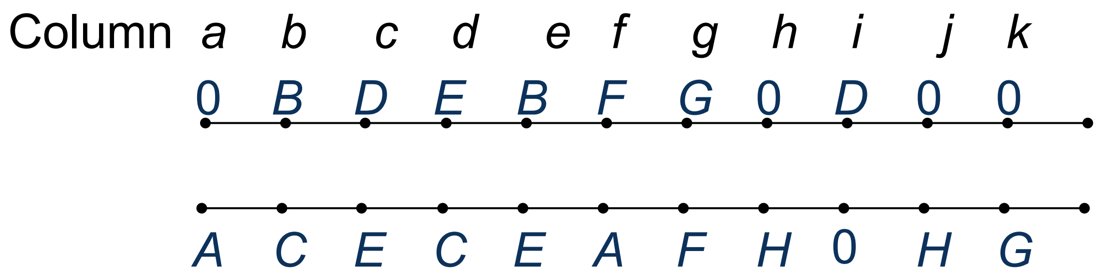
Recap: Channel Routing



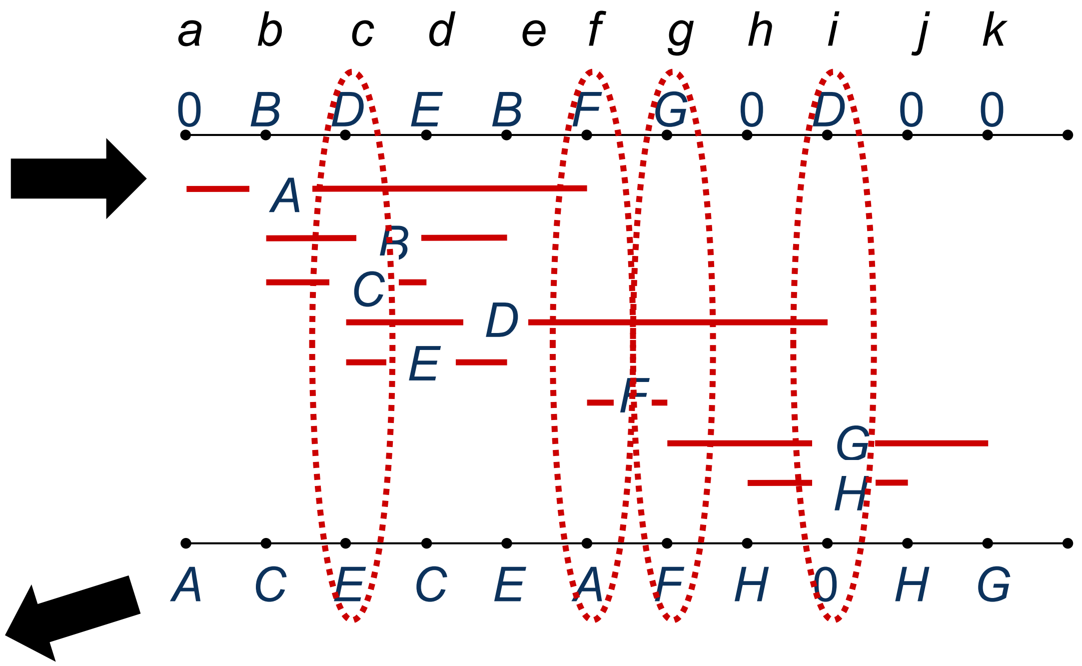
Recap: Two-Layer Channel Routing

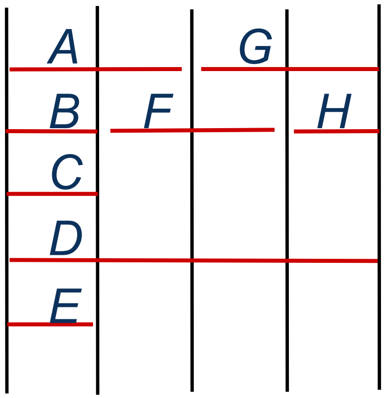


Recap: Horizontal and Vertical Constraints



$S(a) = \{A\}$
$S(b) = \{A,B,C\}$
$S(c) = \{A,B,C,D,E\}$
$S(d) = \{A,B,C,D,E\}$
$S(e) = \{A,B,D,E\}$
$S(f) = \{A,D,F\}$
$S(g) = \{D,F,G\}$
$S(h) = \{D,G,H\}$
$S(i) = \{D,G,H\}$
$S(j) = \{G,H\}$
$S(k) = \{G\}$



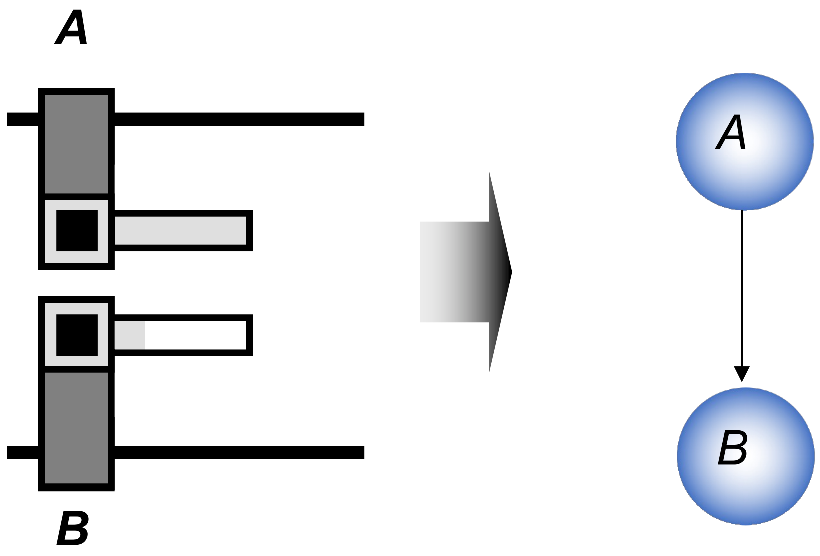


Zone representation

Lower bound on the number of tracks = 5

Recap: Vertical Constraint Graph

- A directed edge $e(i,j)$ connects nodes i and j if the horizontal segment of net i must be located above net j

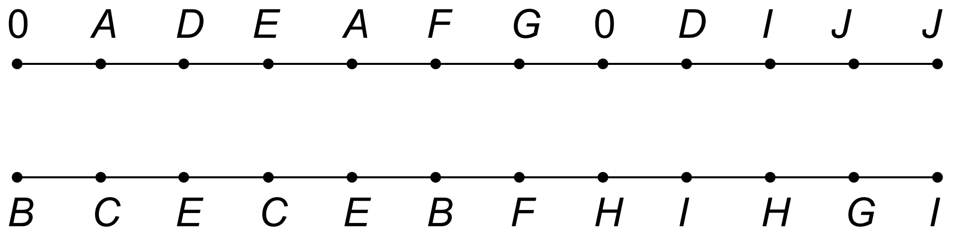


Recap: Left-Edge Algorithm

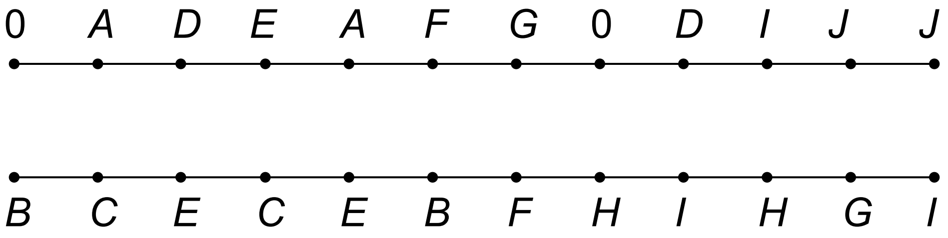
```
Input:          channel routing instance CR
Output:         track assignments for each net
curr_track = 1                                      // start with topmost track
nets_unassigned = Netlist
while (nets_unassigned != Ø)                        // while nets still unassigned
    VCG = VCG(CR)                                   // generate VCG and zone
    ZR = ZONE_REP(CR)                               // representation
    SORT(nets_unassigned,start column)              // find left-to-right ordering
                                                    // of all unassigned nets

    for (i =1 to |nets_unassigned|)
       curr_net = nets_unassigned[i]
       if (PARENTS(curr_net) == Ø &&                // if curr_net has no parent
            (TRY_ASSIGN(curr_net,curr_track))       //   and does not cause
                                                    //   conflicts on curr_track,
            ASSIGN(curr_net,curr_track)             //   assign curr_net
            REMOVE(nets_unassigned,curr_net)
    curr_track = curr_track + 1                     // consider next track
```

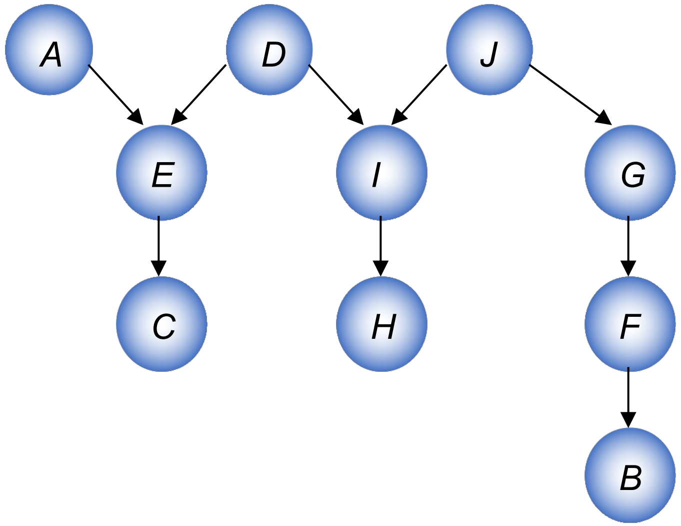
Recap: Walkthrough – I

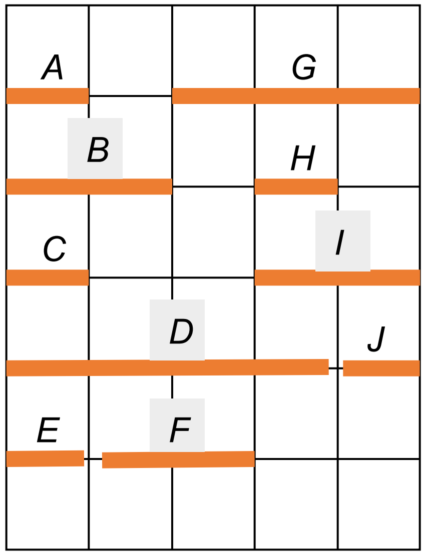


Recap: Walkthrough – II



1. Generate VCG and zone representation





$S(a) = \{A\}$

$S(b) = \{A,B,C\}$

$S(c) = \{A,B,C,D,E\}$

$S(d) = \{A,B,C,D,E\}$

$S(e) = \{A,B,D,E\}$

$S(f) = \{A,D,F\}$

$S(g) = \{D,F,G\}$

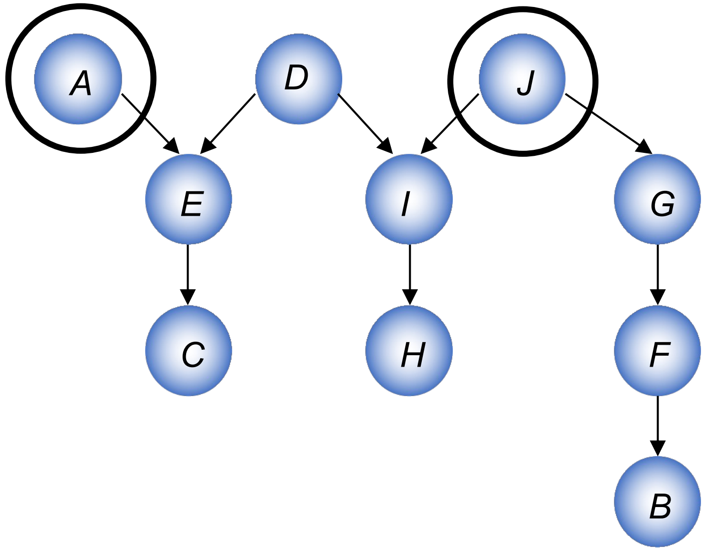
$S(h) = \{D,G,H\}$

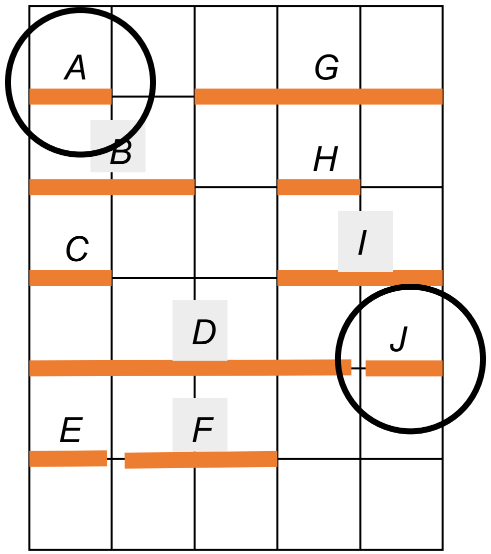
$S(i) = \{D,G,H\}$

$S(j) = \{G,H\}$

$S(k) = \{G\}$

Recap: Walkthrough – III





$S(a) = \{A\}$
$S(b) = \{A,B,C\}$
$S(c) = \{A,B,C,D,E\}$
$S(d) = \{A,B,C,D,E\}$
$S(e) = \{A,B,D,E\}$
$S(f) = \{A,D,F\}$
$S(g) = \{D,F,G\}$
$S(h) = \{D,G,H\}$
$S(i) = \{D,G,H\}$
$S(j) = \{G,H\}$
$S(k) = \{G\}$

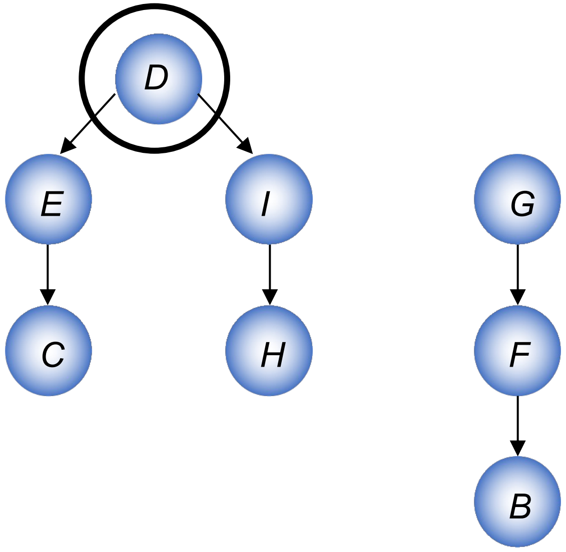
2. Consider next track
3. Find left-to-right ordering of all unassigned nets
   If *curr_net* has no parents and does not cause conflicts on *curr_track*
   assign *curr_net*

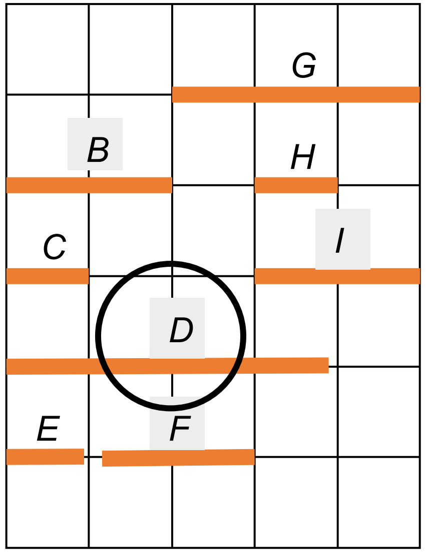
curr_track = 1: Net *A*	Net *J*

4. Delete placed nets (*A*, *J*) in VCG and zone represenation

Recap: Walkthrough – IV

Recap: Walkthrough – V





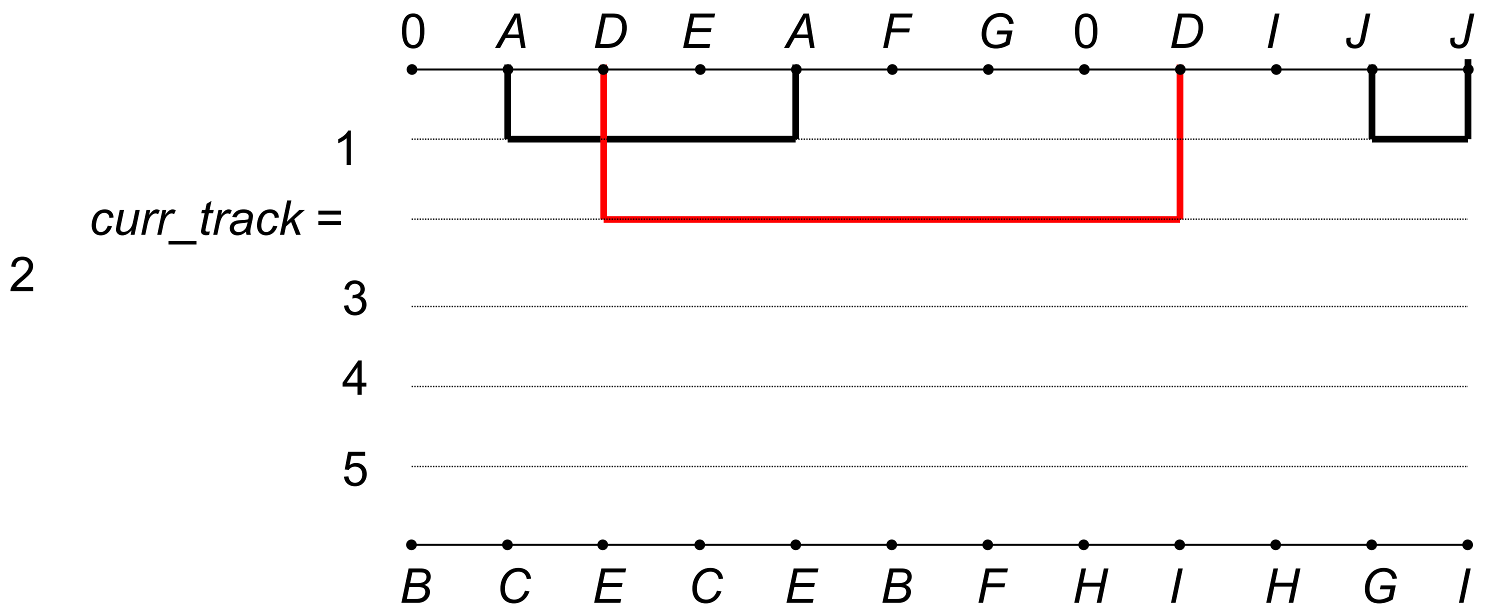
$S(a) = \{A\}$
$S(b) = \{A,B,C\}$
$S(c) = \{A,B,C,D,E\}$
$S(d) = \{A,B,C,D,E\}$
$S(e) = \{A,B,D,E\}$
$S(f) = \{A,D,F\}$
$S(g) = \{D,F,G\}$
$S(h) = \{D,G,H\}$
$S(i) = \{D,G,H\}$
$S(j) = \{G,H\}$
$S(k) = \{G\}$

2. Consider next track
3. Find left-to-right ordering of all unassigned nets
   If *curr_net* has no parents and does not cause conflicts on *curr_track* assign *curr_net*

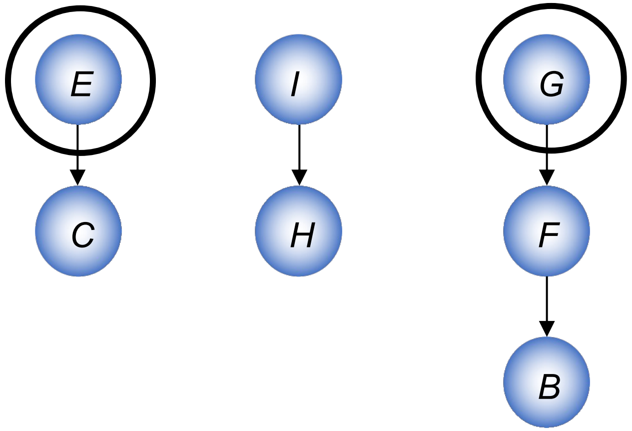
curr_track = 2: Net *D*

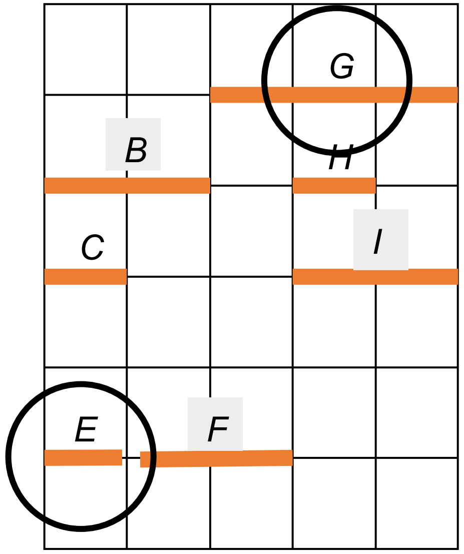
4. Delete placed nets (*D*) in VCG and zone representation

Recap: Walkthrough – VI



Recap: Walkthrough – VII





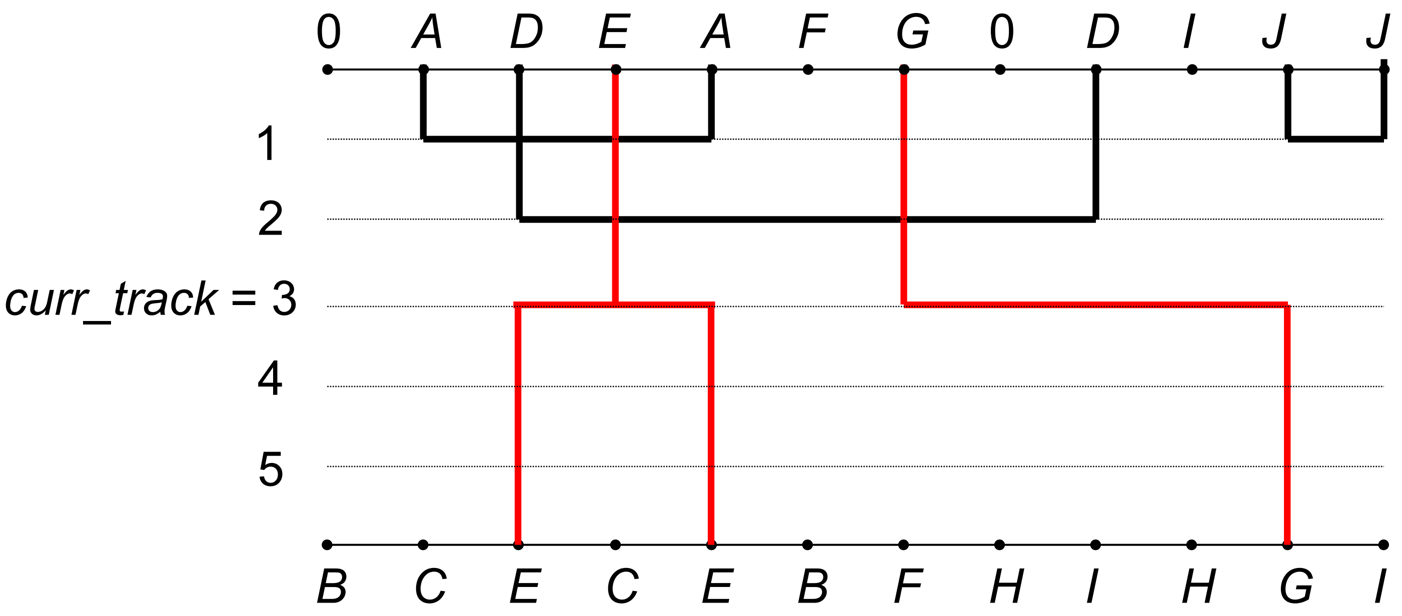
$S(a) = \{A\}$
$S(b) = \{A,B,C\}$
$S(c) = \{A,B,C,D,E\}$
$S(d) = \{A,B,C,D,E\}$
$S(e) = \{A,B,D,E\}$
$S(f) \ = \{A,D,F\}$
$S(g) = \{D,F,G\}$
$S(h) = \{D,G,H\}$
$S(i) \ = \{D,G,H\}$
$S(j) \ = \{G,H\}$
$S(k) = \{G\}$

2. Consider next track
3. Find left-to-right ordering of all unassigned nets
   If *curr_net* has no parents and does not cause conflicts on *curr_track* assign *curr_net*

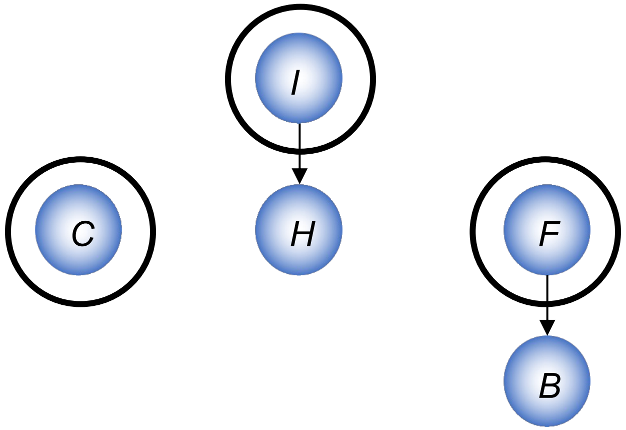
curr_track = 3: Net *E* Net *G*

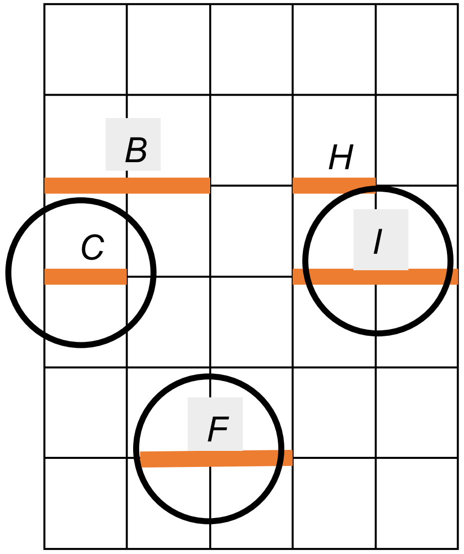
4. Delete placed nets (*E*, *G*) in VCG and zone representation

Recap: Walkthrough – VIII



Recap: Walkthrough – IX





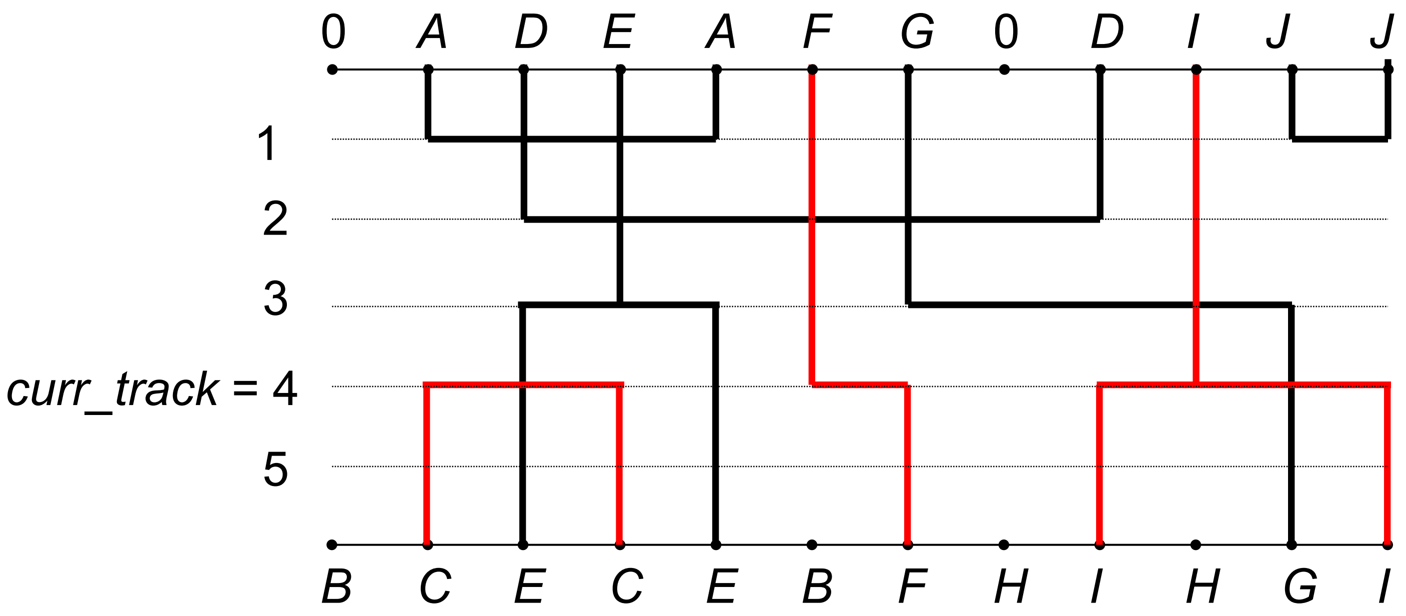
$S(a) = \{A\}$
$S(b) = \{A,B,C\}$
$S(c) = \{A,B,C,D,E\}$
$S(d) = \{A,B,C,D,E\}$
$S(e) = \{A,B,D,E\}$
$S(f) = \{A,D,F\}$
$S(g) = \{D,F,G\}$
$S(h) = \{D,G,H\}$
$S(i) = \{D,G,H\}$
$S(j) = \{G,H\}$
$S(k) = \{G\}$

2. Consider next track
3. Find left-to-right ordering of all unassigned nets
   If *curr_net* has no parents and does not cause conflicts on *curr_track* assign *curr_net*

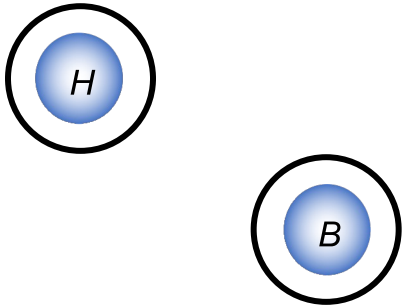
curr_track = 4: Net *C* | Net *F* | Net *I*

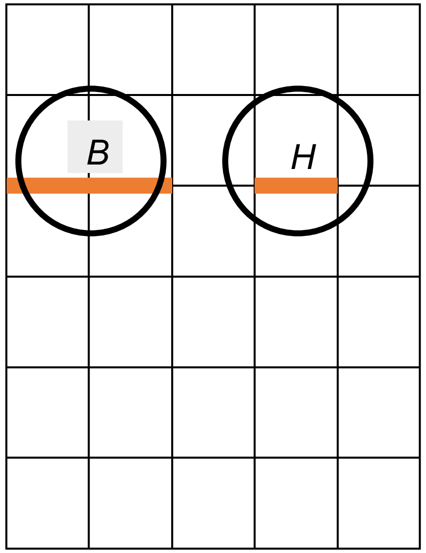
4. Delete placed nets (*C*, *F*, *I*) in VCG and zone representation

Recap: Walkthrough – X



Recap: Walkthrough – XI





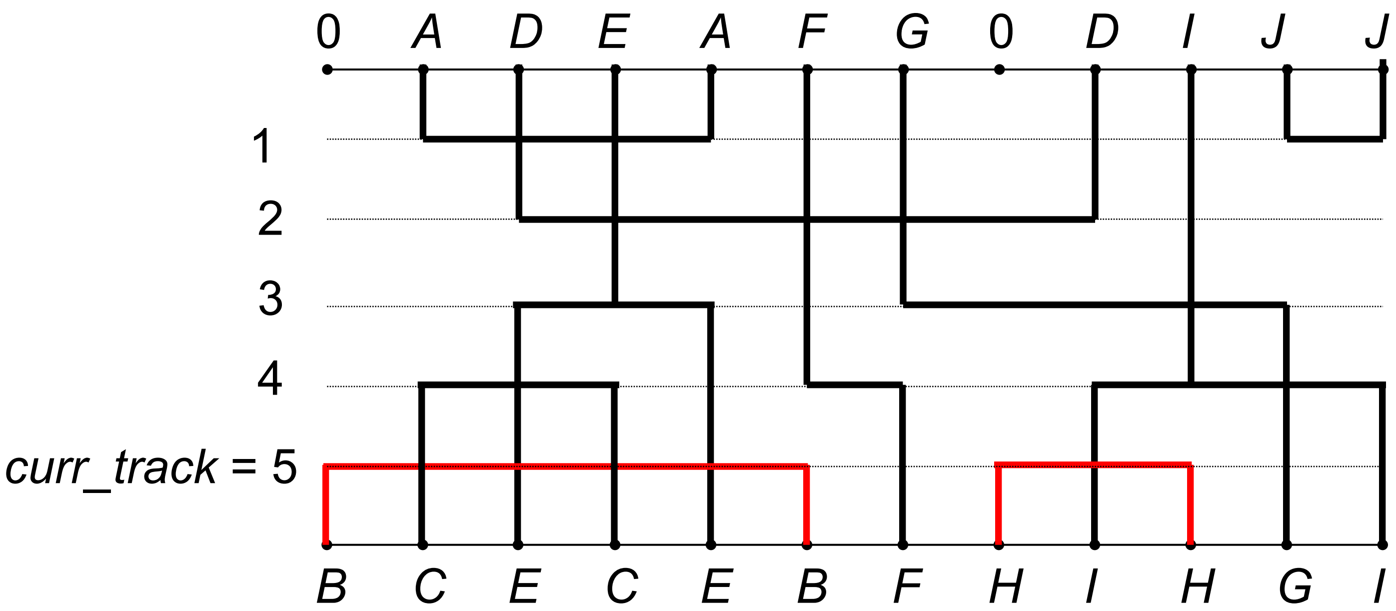
S(*a*) = {*A*}
S(*b*) = {*A*,*B*,*C*}
S(*c*) = {*A*,*B*,*C*,*D*,*E*}
S(*d*) = {*A*,*B*,*C*,*D*,*E*}
S(*e*) = {*A*,*B*,*D*,*E*}
S(*f*) = {*A*,*D*,*F*}
S(*g*) = {*D*,*F*,*G*}
S(*h*) = {*D*,*G*,*H*}
S(*i*) = {*D*,*G*,*H*}
S(*j*) = {*G*,*H*}
S(*k*) = {*G*}

2. Consider next track
3. Find left-to-right ordering of all unassigned nets
   If *curr_net* has no parents and does not cause conflicts on *curr_track* assign *curr_net*

curr_track = 5: Net *B*	Net *H*

4. Delete placed nets (*B*, *H*) in VCG and zone representation

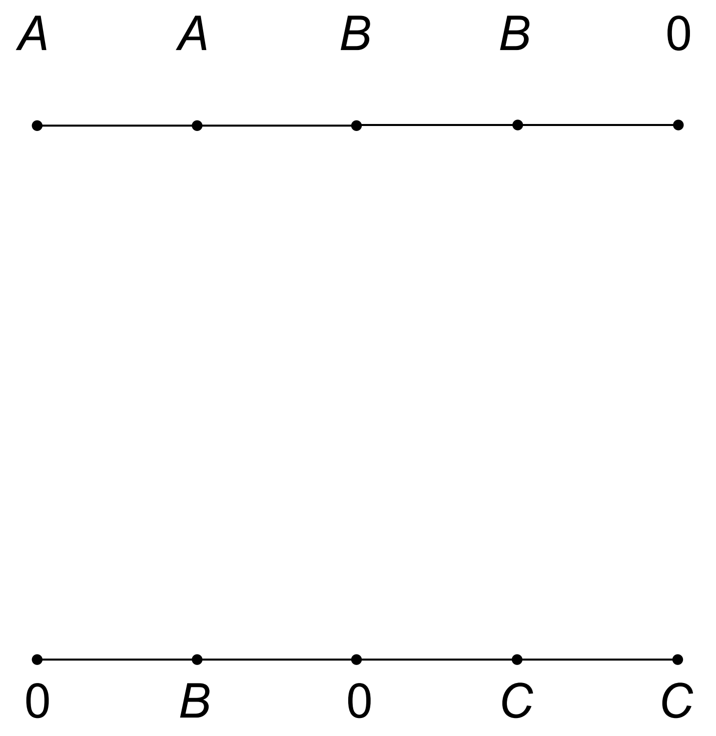
Recap: Walkthrough – XII



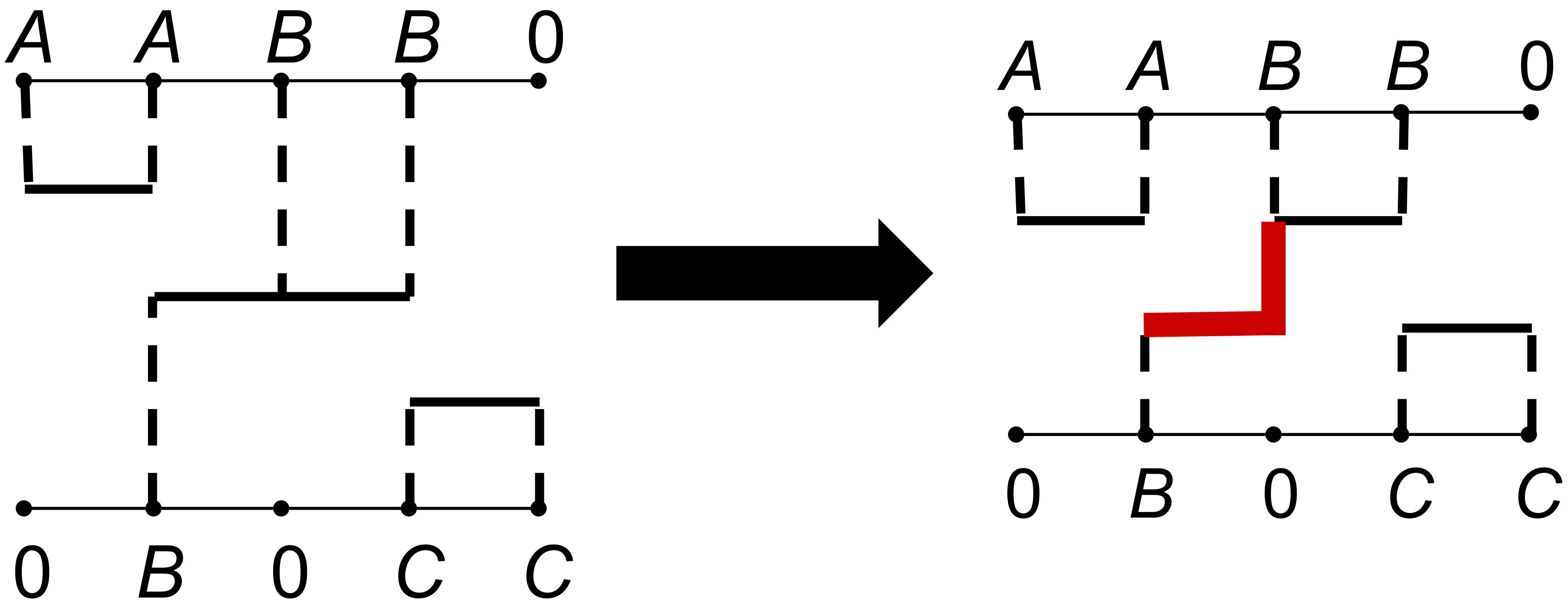
Routing result

Practice Time

- Solve this channel routing problem using the left-edge algorithm

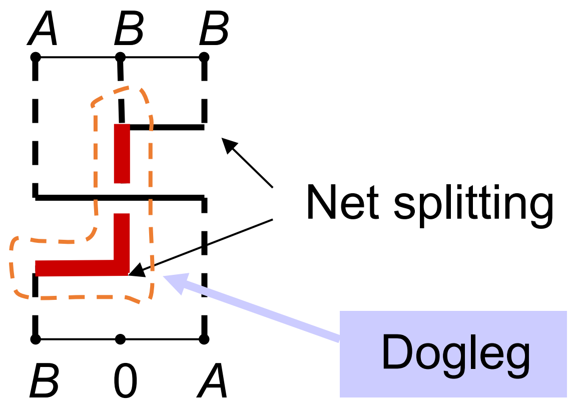


Plain Left-Edge Algorithm is NOT Optimal

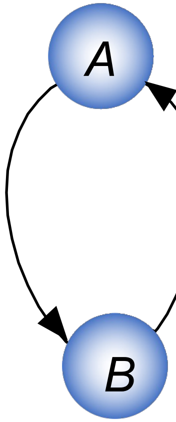


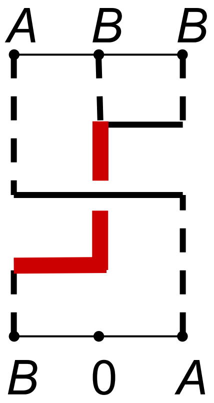
Dogleg Routing Refinement

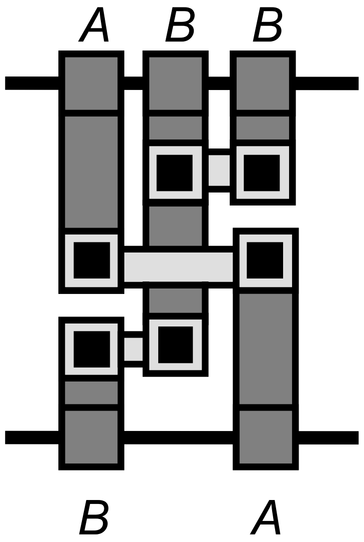
- **Refine left-edge algorithm by net splitting**
  - Ex: B → B1 and B2
- **Two advantages**
  - Alleviate the conflict (break cycles) in the VCG
  - Number of tracks can typically be reduced



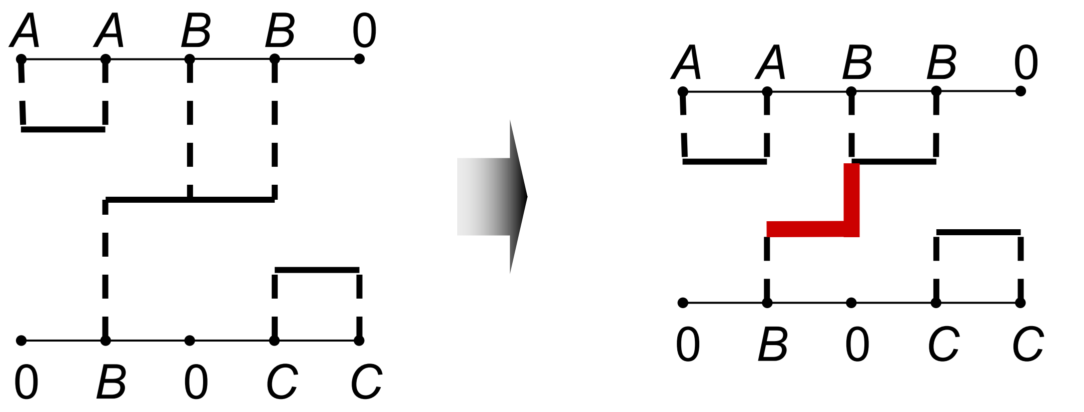
Conflict Alleviation using a Dogleg





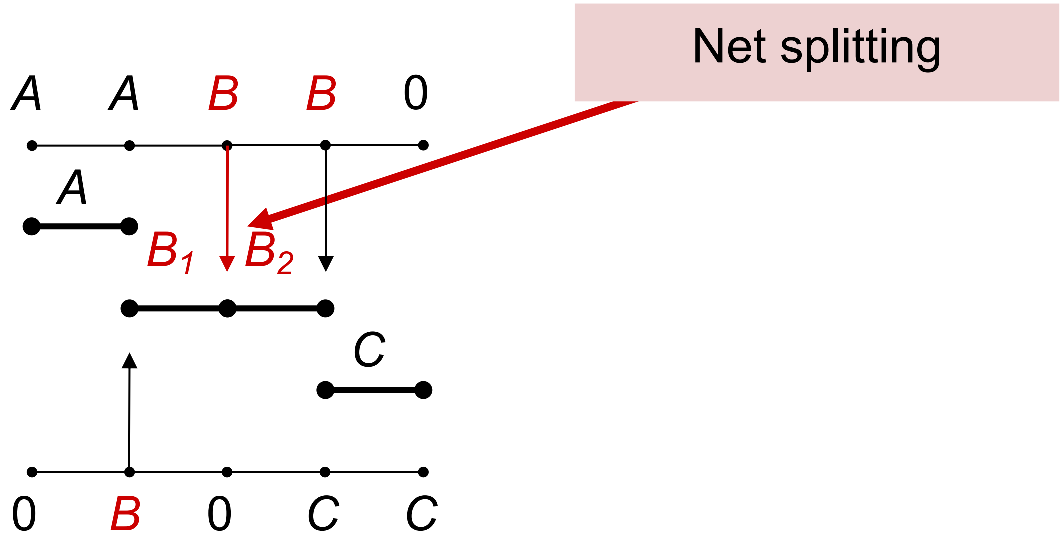


Track Reduction using a Dogleg

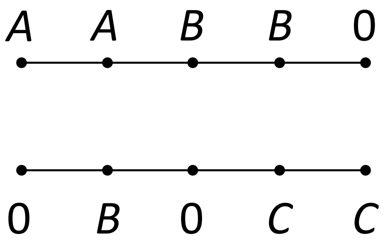


Dogleg Refinement Algorithm

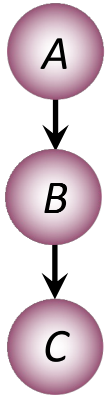
- Splitting *p*-pin nets (p > 2) into p - 1 horizontal segments
- Net splitting occurs only in columns that contain a pin of the given net
- After net splitting, the algorithm follows the left-edge algorithm



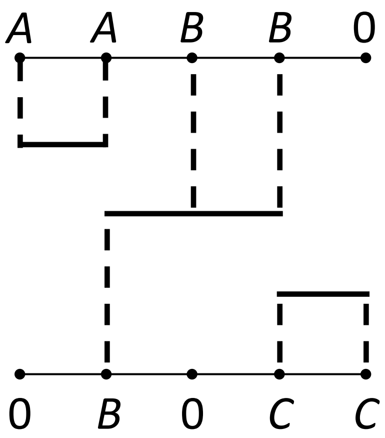
Dogleg Routing Example



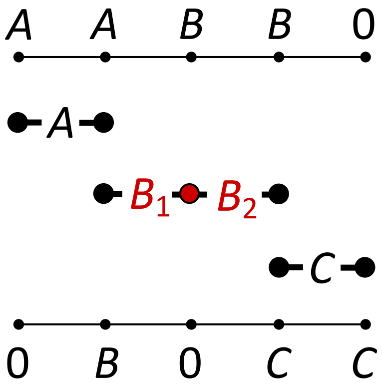
Channel routing problem



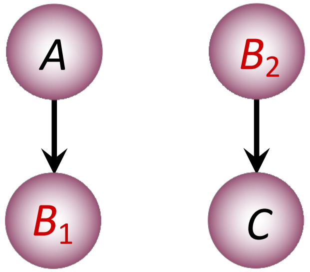
VCG without net splitting



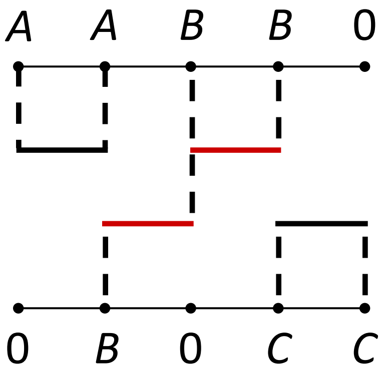
Channel routing solution



Net splitting

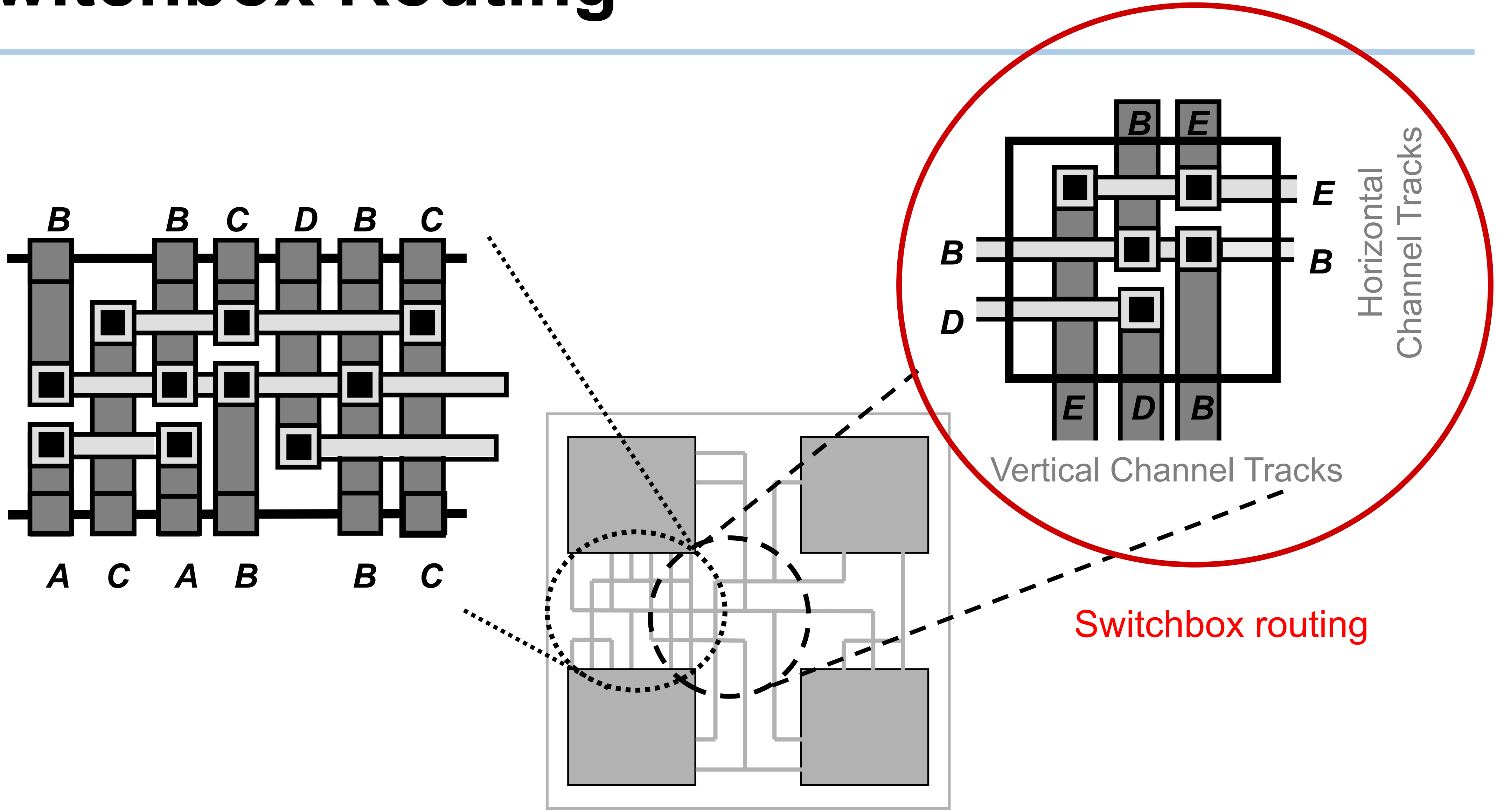


VCG with net splitting



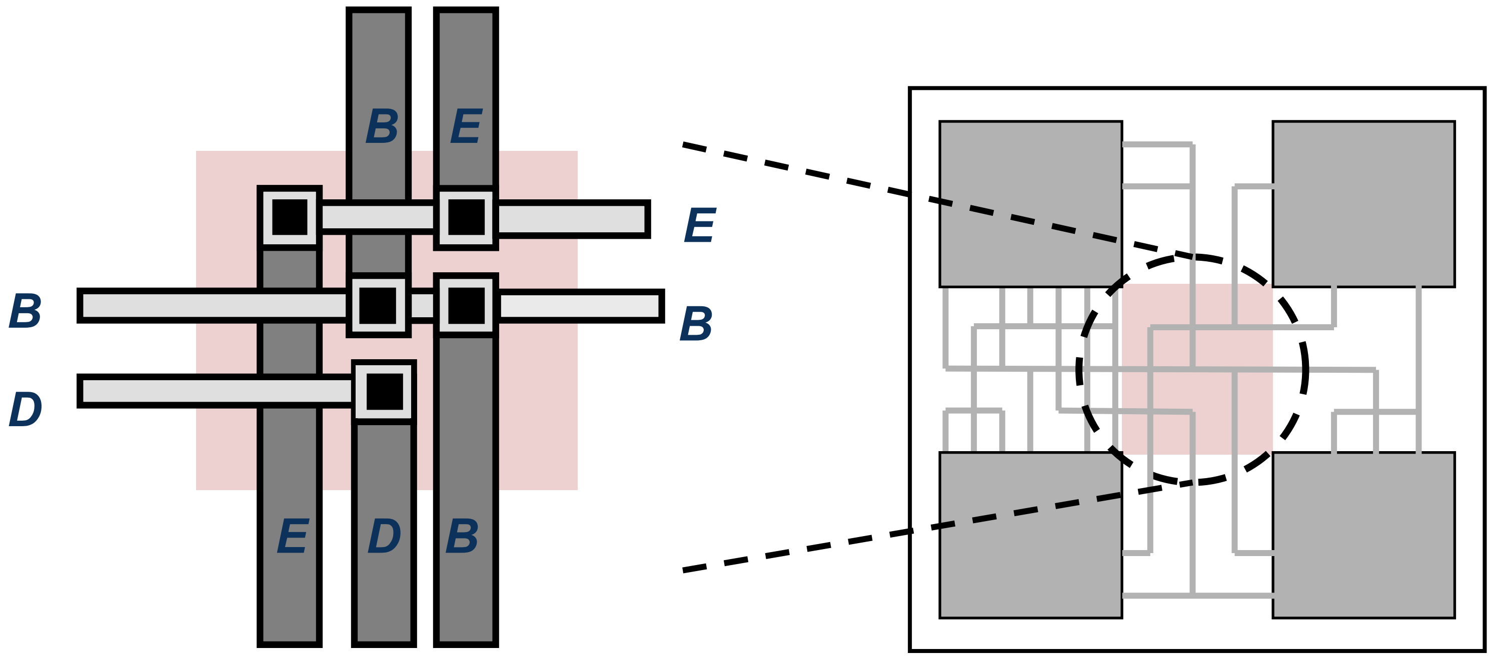
Channel routing solution

Switchbox Routing



Switchbox Routing

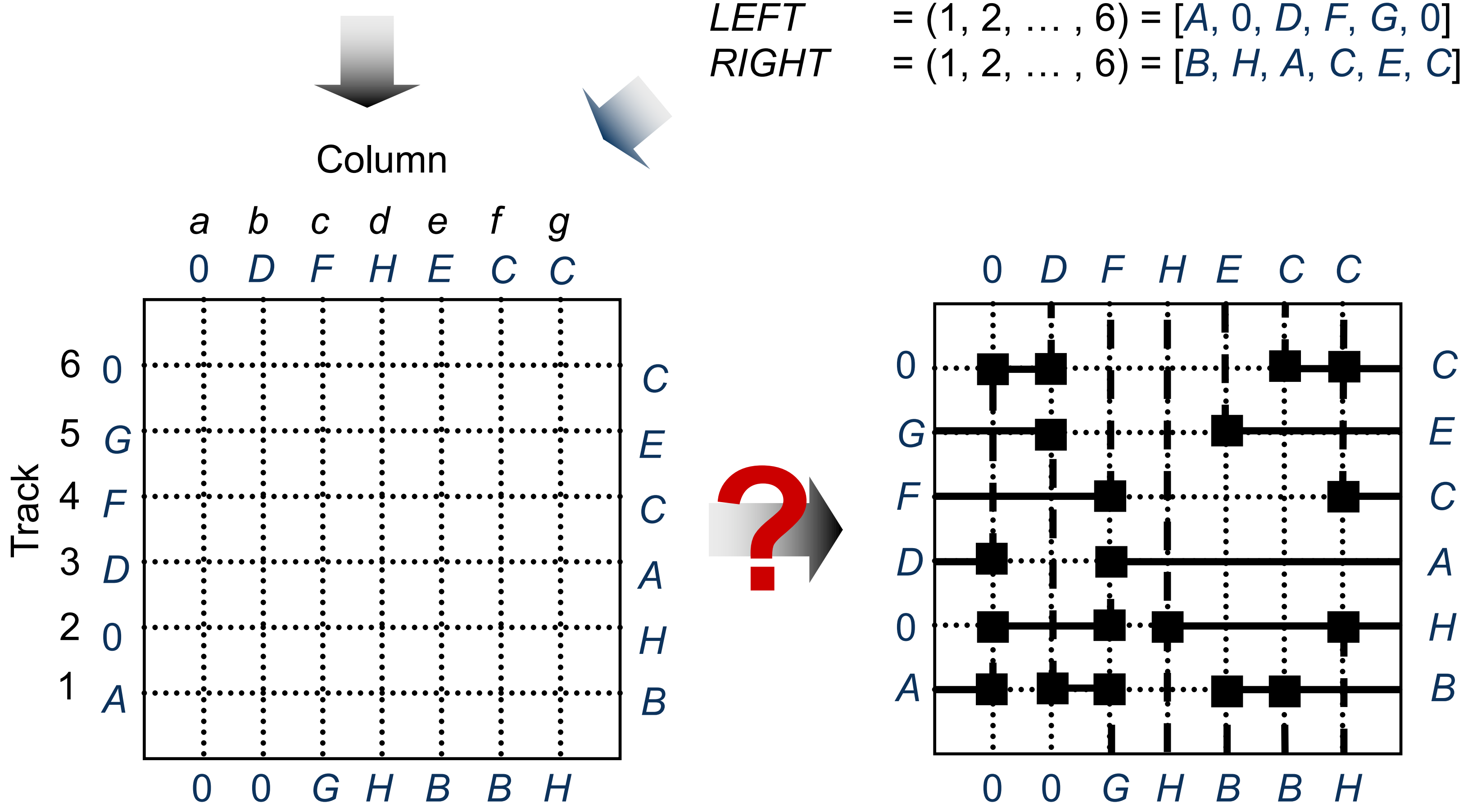
- Fixed dimensions and pin connections on all four sides
- Defined by four vectors *TOP*, *BOT*, *LEFT*, *RIGHT*
- Switchbox routing algorithms are usually derived from (greedy) channel routing algorithms



Switchbox Routing Problem Formulation

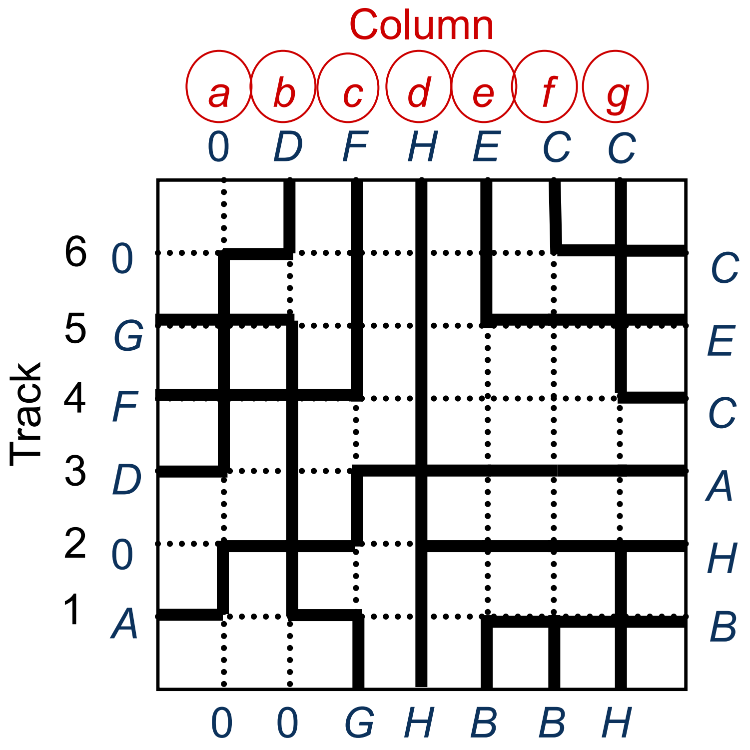
R = {0, 1, 2, …, 8} x {0, 1, 2, … , 7}

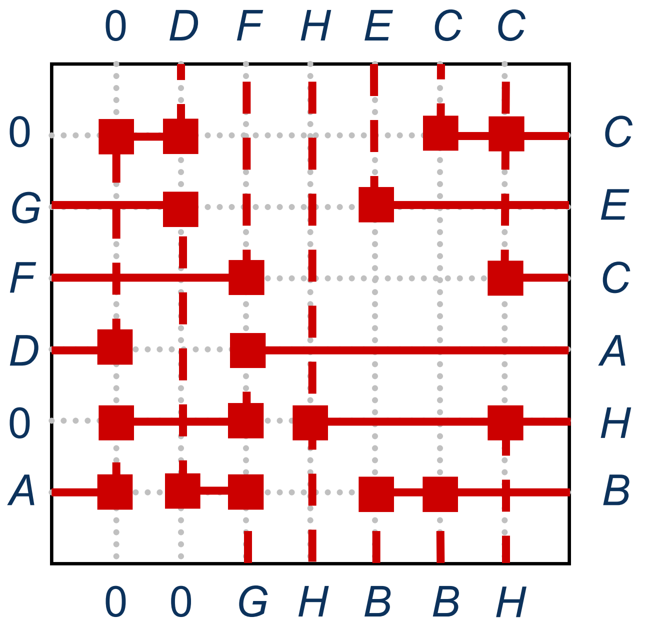
TOP = (1, 2, … , 7) = [0, *D*, *F*, *H*, *E*, *C*, *C*]
BOT = (1, 2, … , 7) = [0, 0, *G*, *H*, *B*, *B*, *H*]
LEFT = (1, 2, … , 6) = [*A*, 0, *D*, *F*, *G*, 0]
RIGHT = (1, 2, … , 6) = [*B*, *H*, *A*, *C*, *E*, *C*]



Switchbox Routing: Example

TOP = (1, 2, … , 7) = [0, *D*, *F*, *H*, *E*, *C*, *C*]
BOT = (1, 2, … , 7) = [0, 0, *G*, *H*, *B*, *B*, *H*]
LEFT = (1, 2, … , 6) = [*A*, 0, *D*, *F*, *G*, 0]
RIGHT = (1, 2, … , 6) = [*B*, *H*, *A*, *C*, *E*, *C*]



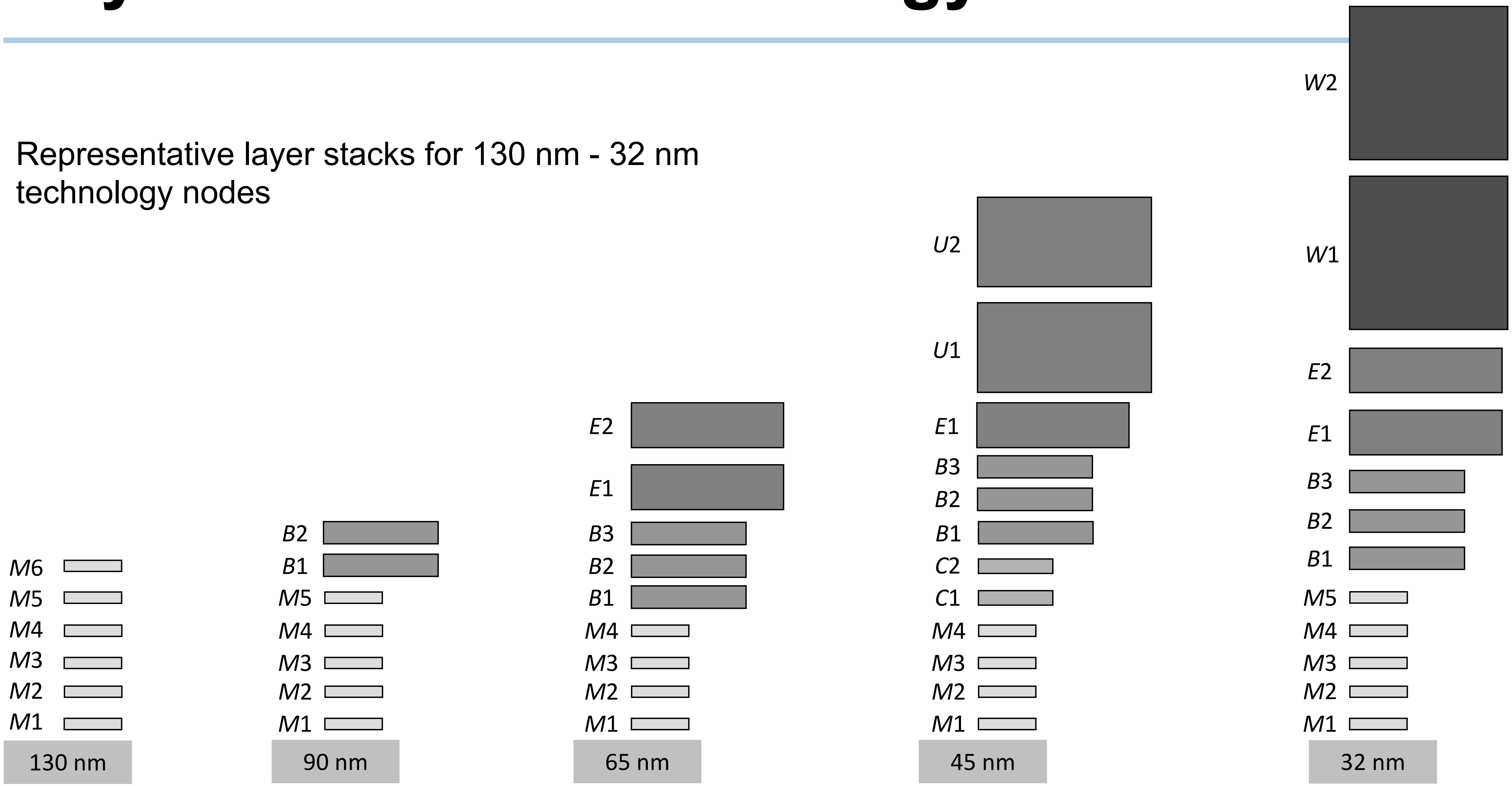


Modern Challenges in Detailed Routing

- **Manufacturers today use different configurations**
  - Many metal layers and widths for high-performance designs
- **Detailed routing is becoming more challenging**
  - Vias connecting wires of different widths inevitably block additional routing resources on the layer with the smaller wire pitch
  - Advanced lithography techniques used in manufacturing require stricter enforcement of preferred routing direction on each layer

Layer Stacks vs Technology Nodes

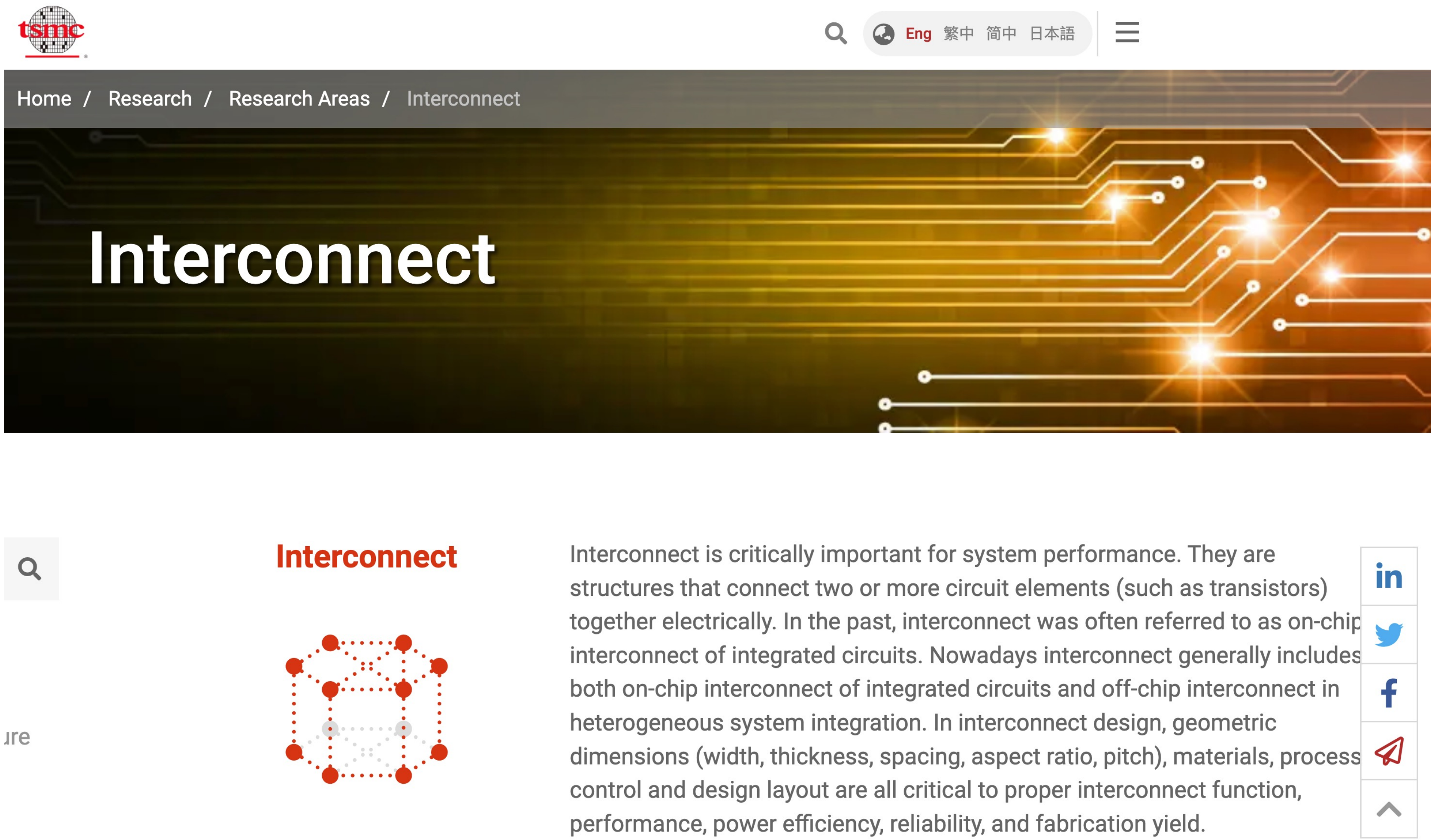
Representative layer stacks for 130 nm - 32 nm technology nodes



Cannot Ignore Manufacturing Constraints

- **Chip manufacturing yield is a key concern in detailed routing**
  - Redundant vias and wiring segments as backups
    (via doubling and non-tree routing)
  - Manufacturability constraints (design rules) become more restrictive
  - Forbidden pitch rules prohibit routing wires at certain distances apart, but allows smaller or greater spacings
- **Detailed routers must account for manufacturing rules and the impact of manufacturing faults**
  - Via defects: via doubling during or after detailed routing
  - Interconnect defects: add redundant wires to already routed nets
  - Antenna-induced defects: detailed routers limit the ratio of metal to gate area on each metal layer

Perspective from Manufacturers



Eng 繁中 简中 日本語

Home / Research / Research Areas / Interconnect

Interconnect

Interconnect

Interconnect is critically important for system performance. They are structures that connect two or more circuit elements (such as transistors) together electrically. In the past, interconnect was often referred to as on-chip interconnect of integrated circuits. Nowadays interconnect generally includes both on-chip interconnect of integrated circuits and off-chip interconnect in heterogeneous system integration. In interconnect design, geometric dimensions (width, thickness, spacing, aspect ratio, pitch), materials, process control and design layout are all critical to proper interconnect function, performance, power efficiency, reliability, and fabrication yield.

https://research.tsmc.com/english/research/interconnect/publish-time-1.html

ACM ISPD Detailed Routing Contest

Announcements

Abstract

Benchmarks

Evaluation

Submission

Schedule

Organizers

FAQ

Contact

ISPD 2018 Contest

Initial Detailed Routing

sponsored by cādence

ISPD 2018 Contest on Initial Detailed Routing

Because the contest already finish, innovus license is expired.

For academic research, Caden

You can apply for the license th

https://www.cadence.com/conte

https://www.ispd.cc/contests/

Announcements Abstract Benchmarks Evaluation Submission Final Results Schedule Registration

Organizers FAQ Contact

ISPD 2019 Contest on Initial Detailed Routing

Because the contest already finish, innovus license is expired.

For academic research, Cadence provides academic innovus license for use.

You can apply for the license through the following webpage:

https://www.cadence.com/content/cadence-www/global/en_US/home/services/cadence-academic-network/university-software-program.html

Summary

- **We have discussed two key detailed routing problems**
  - Channel routing
  - Switchbox routing
- **We have discussed left-edge algorithm**
  - Simple to implement efficiently, in spite of non-optimality
- **We have discussed dogleg refinement algorithm**
  - Split net after an initial left-edge solution
  - Reduce track and alleviate conflict
- **We have discussed modern detailed routing challenges**
  - Must consider the impact from manufacturing process