Elective II: VLSI Design

Code: CISM 402

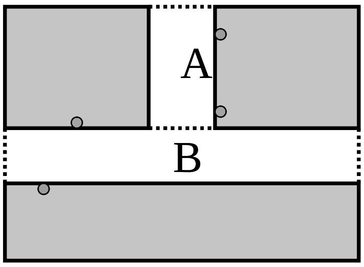
Pritha Banerjee

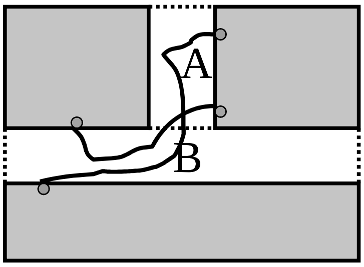
Courtsey for slides: Debasis Mitra, NIT Durgapur

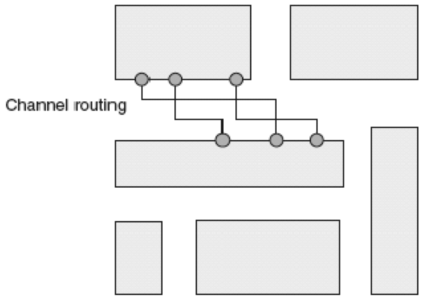
Detailed Routing

- Books:
  - Chapter 7 of Naveed A. Sherwani, *Algorithms for VLSI Physical Design Automation*, Kluwer Academic Publishers

Detailed Routing





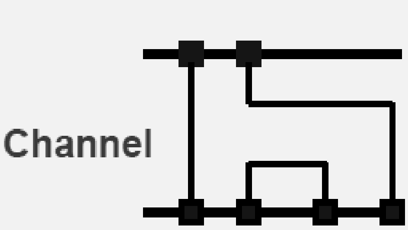


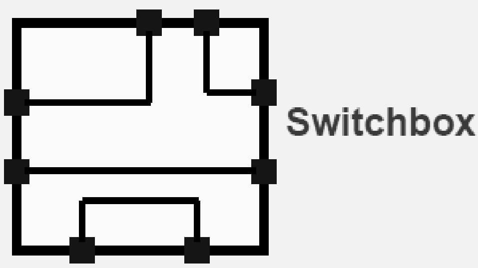
- Global routing : wire paths are constructed throgh a subset of routing regions connecting terminals of each net
- The routing regions are divided into channels and switchboxes
- So only need to consider the ***channel routing problem*** and the ***switchbox routing problem***

Detailed Routing

Channels and Switchboxes

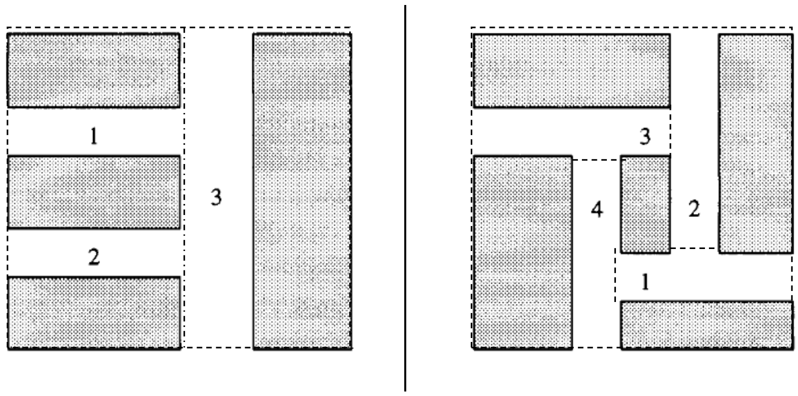
- **There are normally two kinds of rectilinear regions.**
  - **Channels:** routing regions having two parallel rows of fixed terminals.
  - **Switchboxes:** generalizations of channels that allow fixed terminals on all four sides of the region.





Channel Routing

- For Standard-cell and Full-custom design, channels are expandable
- The goal is to route all nets using the minimum channel width
- Incremental method : route one region at a time
- Ordering of routing region is important
- Slicing floorplan: route channel 1, 2, and 3, no rerouting required; non-slicing: 1,2, 3, 4; 1 may have to be re-routed



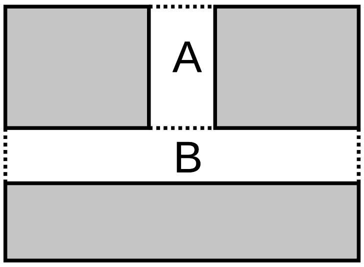
Channel routing for different design styles

- For Gate-array design, channel widths are fixed. The goal is to finish routing of all the nets.

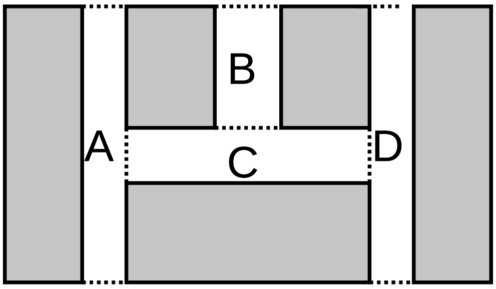
- For Standard-cell and Full-custom design, channels are expandable. The goal is to route all nets using the <u>minimum channel height</u>

- We will consider the case when the channels are expandable.

Channel Ordering



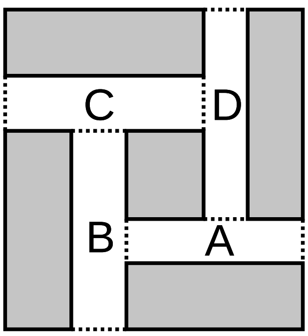
The width of A is not known until A is routed, we must route A first.



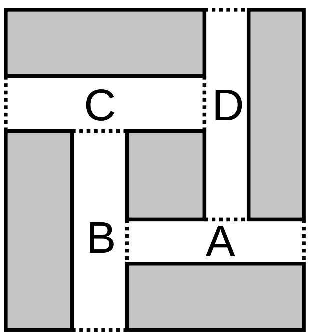
What should be the routing order for this example?

Channel Ordering

No feasible channel order!



Need to use switchbox



1. Fix the terminals between A & B
2. Route B, C, then D (channel)
3. Route A (switchbox)

Routing Considerations

- Number of terminals
  - Majority of nets are two-terminal ones.
  - For some nets like clock and power, number of terminals can be very large.
  - Each multi-terminal net can be decomposed into several two-terminal nets.
- Net width
  - Power and ground nets have greater width.
  - Signal nets have less width.

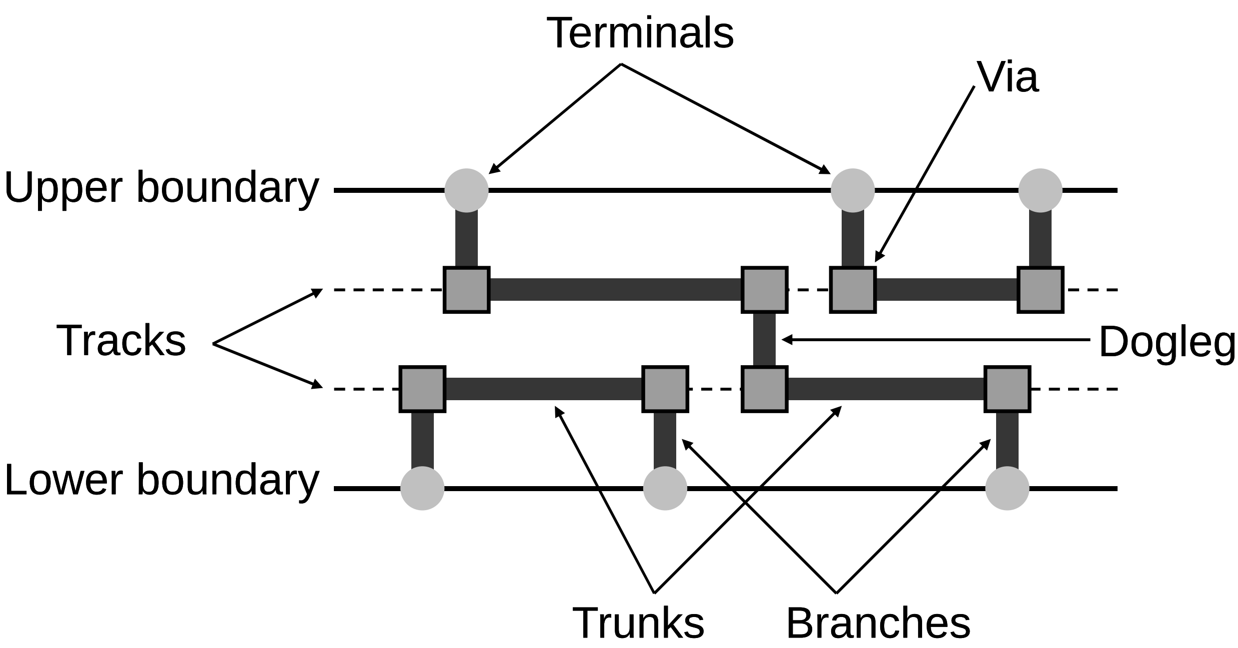
Contd.

- Via restrictions
  - Regular: only between adjacent layers.
  - Stacked: passing through more than two layers.
- Boundary type
  - Regular: straight border of routing region
  - Irregular
- Number of layers
  - Modern fabrication technology allows at least five layers of routing.
- Net types
  - Critical: power, ground, clock nets
  - Non-critical: signal nets

Terminologies

- A channel is a routing region bounded by two parallel rows of terminals
  - Without loss of generality, it is assumed that the two rows are horizontal
  - The top and bottom rows are called **top boundary** and **bottom boundary** respectively
- The horizontal and vertical dimension of a channel is called ***channel length*** and ***channel height*** respectively
- The horizontal segment of a net is called a ***trunk***
- The vertical segment of a net that connect the trunk to the terminals are called its ***branches***
- The horizontal line along which a trunk is placed is called a ***track***
- A ***dogleg*** is a vertical segment that is used to maintain the connectivity of the two trunks of a net on two different tracks

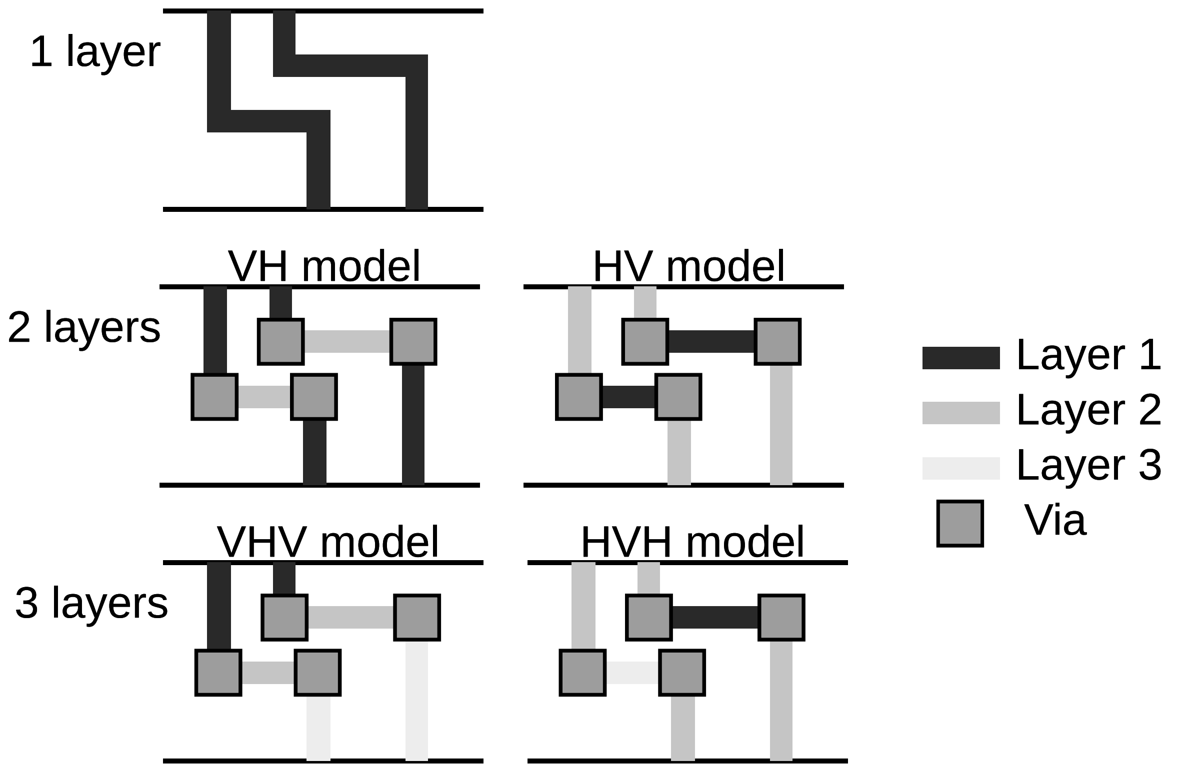
Terminologies



Terminologies

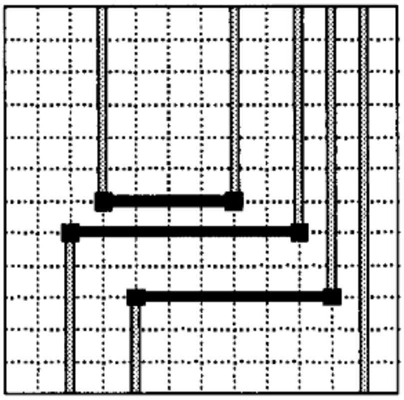
- Routing region consists of one or more layers
- Single –layer routing problem is NP-complete
- Multi-layer routing regions: wires can switch adjacent layers at certain location using vias (electrical connection)
- Multi-layer routing problem is NP-complete
- Restricted layer/ reserved layer model: layers are restricted to one type of wire segment (vertical or horizontal)
- Unreserved layer: No restrictions

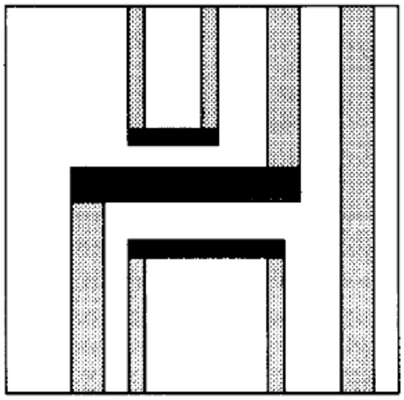
Modeling of Routing Layers



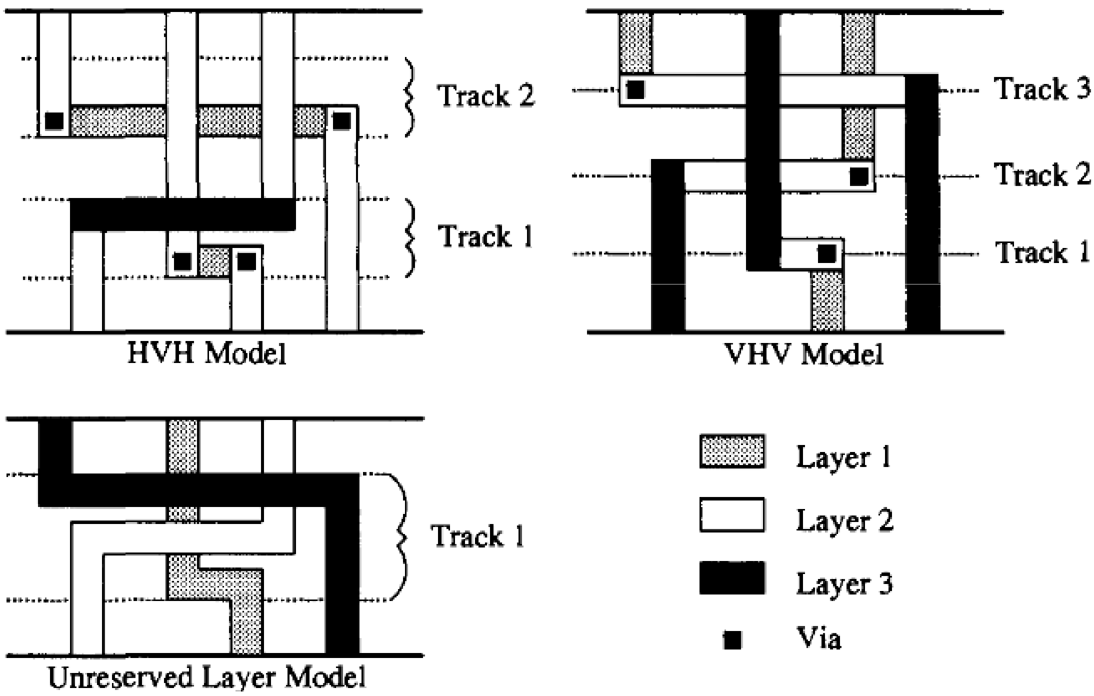
Routing Models

- Grid based Model
  - Huge space requirement
  - Wires follow paths along the grid lines
    - Horizontal grid line : track
    - Vertical grid line : column
- Grid less Model





Routing Models

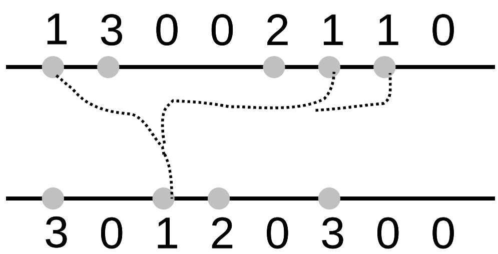


Channel Routing problem

- Input:
  - Two vectors of the same length to represent the pins on two sides of the channel
    - Each terminal (pin) is assigned a number which represents the net to which that terminals belongs to
    - Terminals numbered zero (**vacant terminals**) does not belong to any net and therefore requires no electrical connection
  - Number of layers and layer model used
- Output:
  - Connect pins of the same net together
  - Minimize the channel width
  - Minimize the number of vias

Channel Routing problem

- Input:



(13002110)
(30120300)

- Channel length, terminal list, connection list
- Number of layers and layer model used

- Output:
  - Connect pins of the same net together
  - Minimize the channel width
  - Minimize the number of vias

Problem Formulation

- The channel is defined by a rectangular region with two rows of terminals along its top and bottom sides.
  – Each terminal is assigned a number between 0 and N.
  – Terminals having the same label i belong to the same net i.
  – A '0' indicates no connection.
- The netlist is usually represented by two vectors TOP and BOT.
  – TOP(k) and BOT(k) represents the labels on the grid points on the top and bottom sides of the channel in column k, respectively.

Contd.

- The task of the channel router is to:
  – Assign horizontal segments of nets to tracks.
  – Assign vertical segments to connect
    - Horizontal segments of the same net in different tracks.
    - The terminals of the net to horizontal segments of the net.
- Channel height should be minimized.
- Horizontal and vertical constraints must not be violated.

Horizontal Constraints

- There is a horizontal constraint between two nets if the trunks of these two nets overlaps when placed on the same track

Horizontal Constraints

Horizontal Constraint Graph:

A undirected graph $G_h = (V, E_h)$, where

$V = \{ v_i \mid v_i$ represents interval I_i corresponding to net $N_i\}$

$E_h = \{ (v_i, v_j) \mid I_i$ and I_j have non-empty intersection$\}$

For a net N_i, the interval spanned by the net (denoted by I_i) is defined by (l_i , r_i) where l_i , r_i are the left most and right most terminals of the net N_i respectively

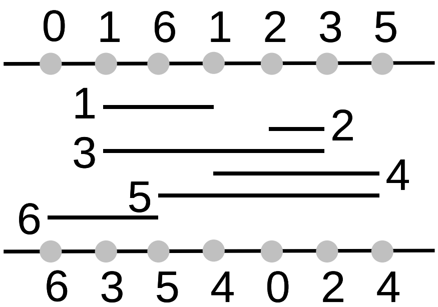
Horizontal Constraints

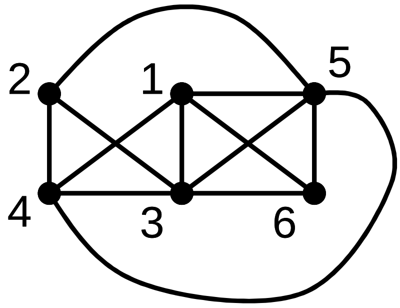
Horizontal Constraint Graph:

A undirected graph $G_h = (V, E_h)$, where

$V = \{ v_i \mid v_i$ represents interval I_i corresponding to net $N_i\}$

$E_h = \{ (v_i, v_j) \mid I_i$ and I_j have non-empty intersection$\}$





Horizontal constraint graph

Horizontal Constraints

Horizontal Constraint Graph:

A undirected graph $G_h = (V, E_h)$, where

$V = \{ v_i \mid v_i$ represents interval I_i corresponding to net $N_i\}$

$E_h = \{ (v_i, v_j) \mid I_i$ and I_j have non-empty intersection$\}$

Usage:

- HCG plays a major role in determining the channel height
- In a grid based two-layer model, no two nets which have a horizontal constraint may be assigned to the same track
  - The maximum clique in HCG forms the lower bound for channel height: channel density
- In the two-layer gridless model, the summation of widths of nets involved in the maximum clique determine the lower bound

Vertical Constraints

- There is a vertical constraint between two nets N_i and N_j if there exists a column such that the top terminal of the column belongs to N_i and the bottom terminal belongs to N_j and $i \neq j$

Vertical Constraints

Vertical Constraint Graph:

A directed graph $G_v = (V, E_v)$, where

$V = \{ v_i \mid v_i \text{ represents net } N_i \}$

$E_v = \{ (v_i, v_j) \mid N_i \text{ has a vertical constraint with } N_j \}$

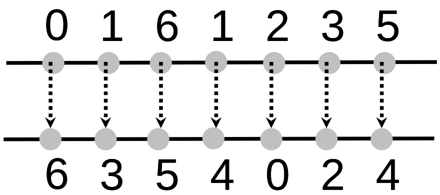
Vertical Constraints

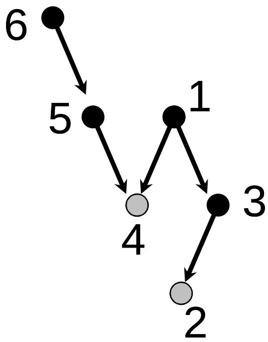
Vertical Constraint Graph:

A directed graph $G_h = (V, E_v)$, where

$V = \{ v_i \mid v_i \text{ represents net } N_i \}$

$E_v = \{ (v_i, v_j) \mid N_i \text{ has a vertical constraint with } N_j \}$





Vertical constraint graph

Vertical Constraints

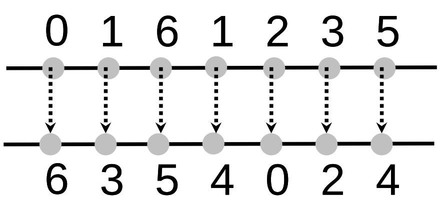
Vertical Constraint Graph:

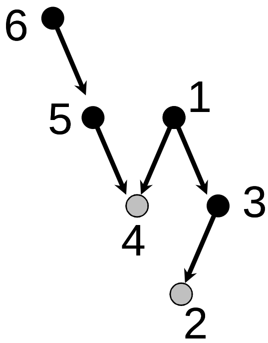
A directed graph $G_h = (V, E_v)$, where

$V = \{ v_i \mid v_i \text{ represents net } N_i \}$

$E_v = \{ (v_i, v_j) \mid N_i \text{ has a vertical constraint with } N_j \}$

Note: A vertical constraint implies a horizontal constraint, but the converse is not true





Vertical constraint graph

Vertical Constraints

Vertical Constraint Graph:

A directed graph $G_h = (V, E_v)$, where

$V = \{ v_i \mid v_i \text{ represents net } N_i \}$

$E_v = \{ (v_i, v_j) \mid N_i \text{ has a vertical constraint with } N_j \}$

Usage:

- VCG also helps in determining the channel height
- In a grid based two-layer model, no two nets in a directed path in the VCG may be assigned to the same track
  - If doglegs are not allowed, then the length of the longest path in VCG forms a lower bound for channel height : channel density
- In the two-layer gridless model, the summation of widths of nets involved in the longest path of VCG determine the lower bound

Vertical Constraints

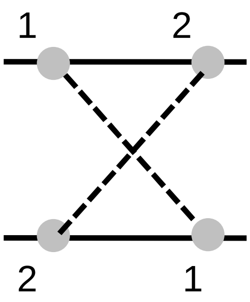
Vertical constraint cycle and Dogleg:

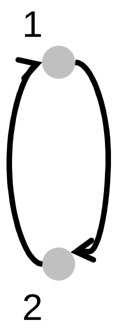
- If VCG is not acyclic then some nets must be doglegged to break the cycle

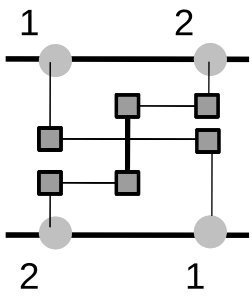
Vertical Constraints

Vertical constraint cycle and Dogleg:

- If VCG is not acyclic then some nets must be doglegged to break the cycle







Combined Constraint Graph

A mixed graph $G_m = (V, E_m)$, where

$V = \{ v_i \mid v_i \text{ represents net } N_i \}$

$E_m = \{ E_h \cup E_v \}$

Lower bound for channel width* = *maximum (The maximum clique in HCG, The longest path in VCG)

Two-layer Channel Routing

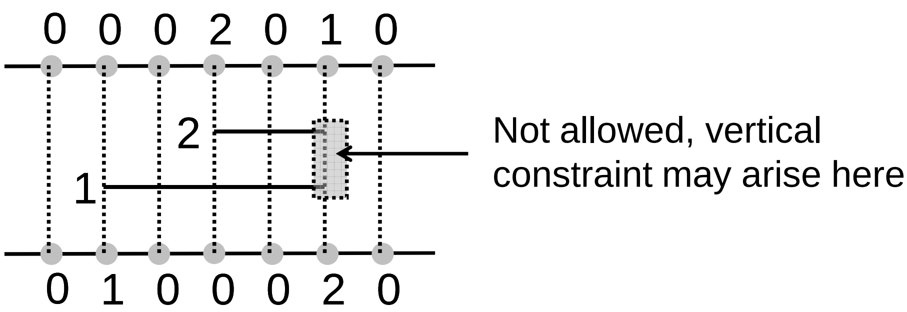
- Left-Edge Algorithms (LEA)
  - Basic Left-Edge Algorithm
  - Left-Edge Algorithm with Vertical Constraints
  - Dogleg Router
- Constraint-Graph Based Algorithm
  - Net Merge Channel Router
  - Gridless Channel Router
- Greedy Channel Router
- Hierarchical Channel Router

Left-Edge Algorithm

"Wire Routing by Optimizing Channel Assignment within Large Apertures", A. Hashimoto and J. Stevens, DAC 1971, pages 155-169.

Left-Edge Algorithm

- Assumptions:
  - 2-layers routing
    - One horizontal routing layer
  - No doglegs, no vertical constraint
  - Two terminal nets
- Always gives a solution with channel width equal to channel density, i.e., optimal solution.



Left-Edge Algorithm: Ignore VC

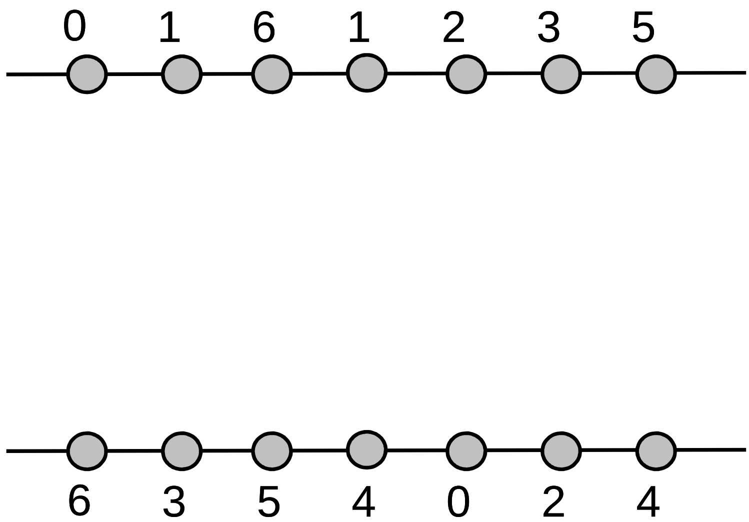
- Idea:
  - Sorts the horizontal segments of the nets in increasing order of their **left** end points
  - Place them one by one greedily starting from the bottommost (or topmost) available track
    - Scans through the tracks from bottom to top (or top to bottom), and assigns the net to the first track that can accommodate the net

      [A net cannot be placed on a track if it has a horizontal constraint with any one of the already placed net at that track]

Time complexity : O(n log n), n : number of intervals

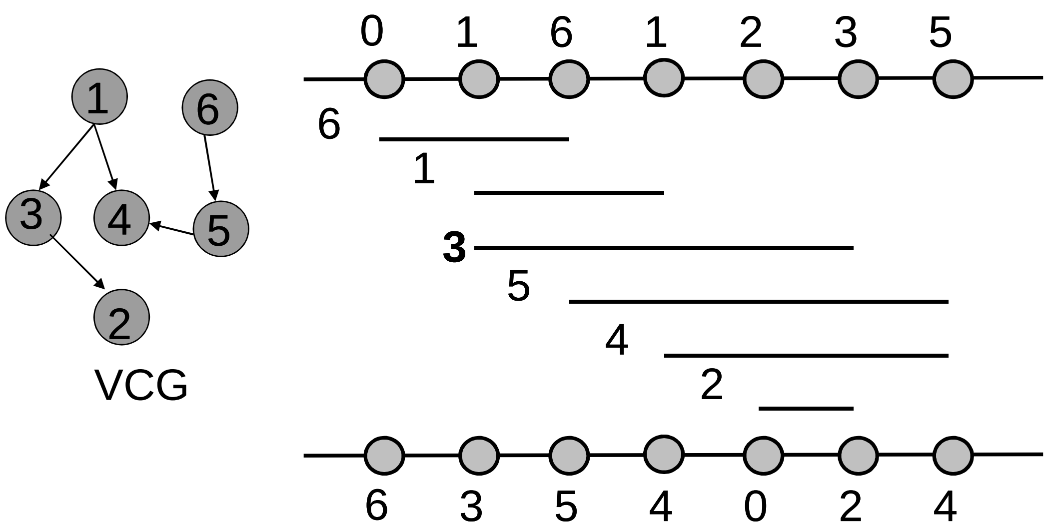
Left-Edge Algorithm : Ignore VC

- Illustration:



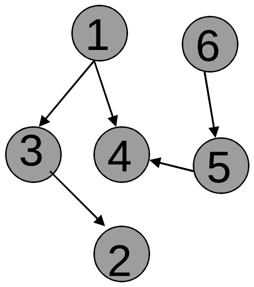
Left-Edge Algorithm: Extension

- Illustration:

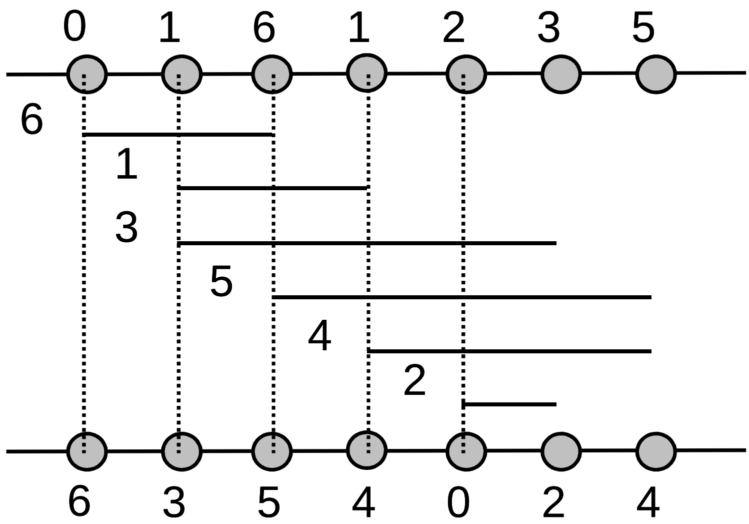


Left-Edge Algorithm

- Illustration:



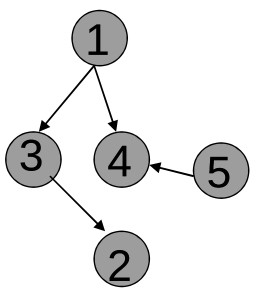
VCG



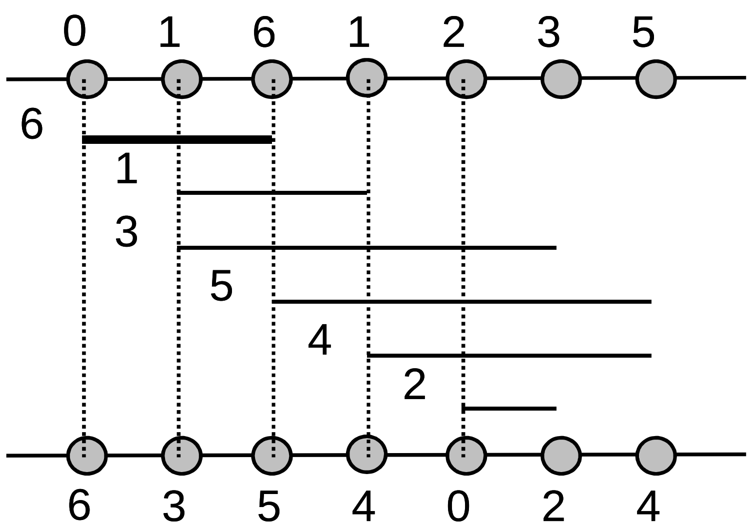
No edge
incident on 6

Left-Edge Algorithm

- Illustration:



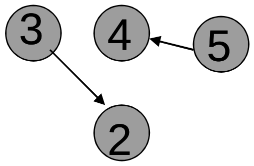
VCG



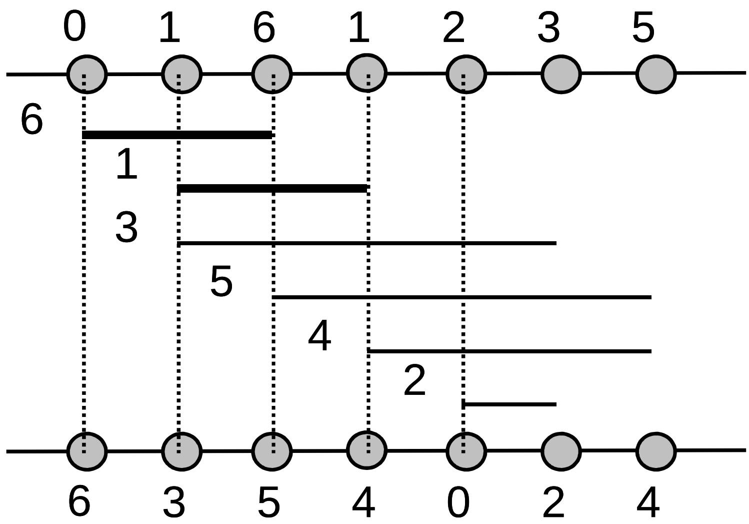
No edge
incident on 1

Left-Edge Algorithm

- Illustration:



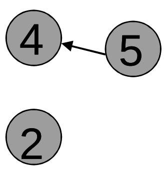
VCG



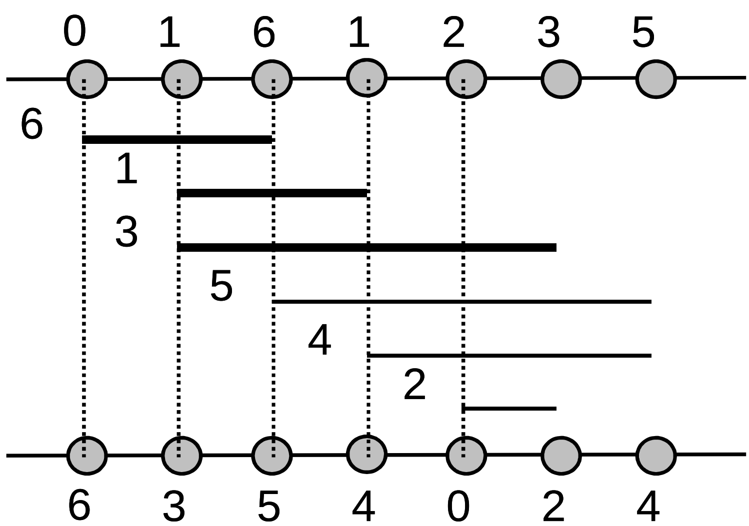
No edge
incident on 3

Left-Edge Algorithm

- Illustration:



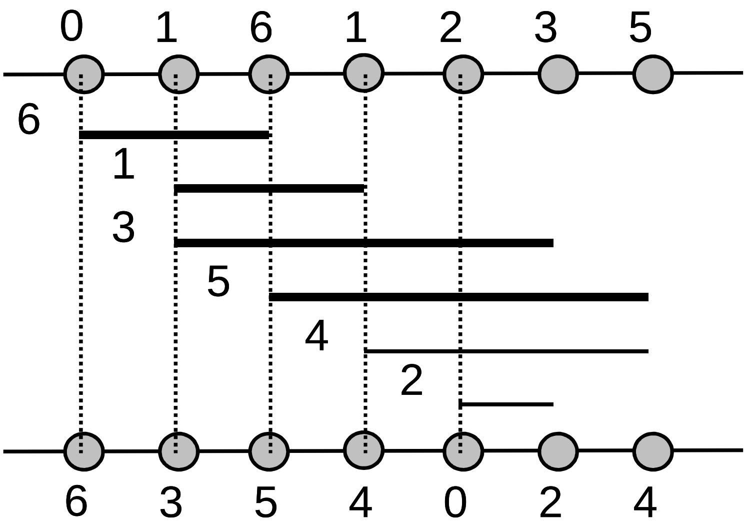
VCG

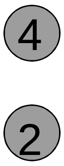


No edge
incident on 5

Left-Edge Algorithm

- Illustration:



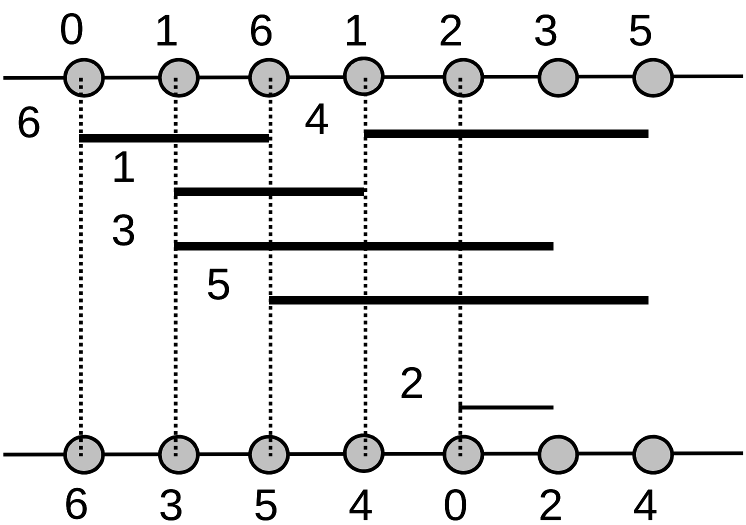


VCG

No edge
incident on 4

Left-Edge Algorithm

- Illustration:

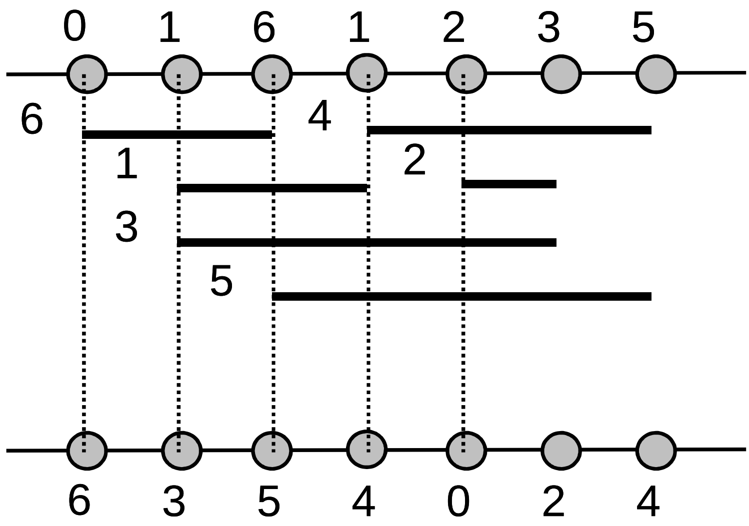


VCG

No edge
incident on 2

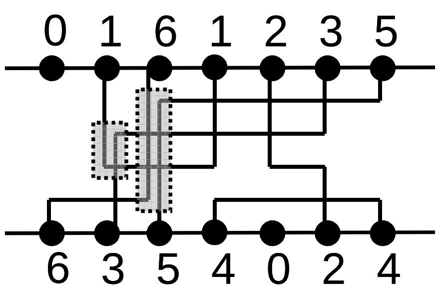
Left-Edge Algorithm

- Illustration:



Vertical constraint consideration

- The Left-edge algorithm ignores vertical constraints.
- When there is only one vertical layer, the algorithm will produce overlapping of vertical wire segments.



If vertical constraint exists

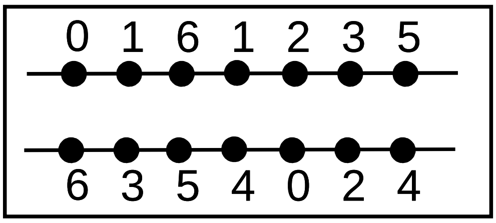
- **Extension to Left-Edge Algorithm**
  - Vertical constraints may exist, but there are no directed cycles in the VCG.
  - Select a net for routing if
    - The x-coordinate of the leftmost terminal is the least.
    - There is no edge incident on the vertex corresponding to that net in the VCG.
  - After routing a net, the corresponding vertex and the incident edges are deleted from the VCG.
  - Other considerations same as the basic left-edge algorithm.

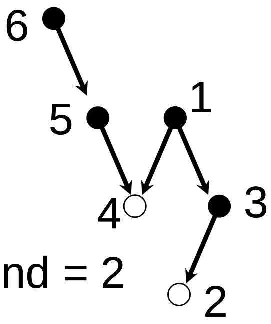
Left-Edge Algorithm

Note:

Given a two-layer channel routing problem with no vertical constraints, if doglegs are not allowed, LEA produces a routing solution with minimum number of tracks

Lower bound on channel width





Lower bound = 2

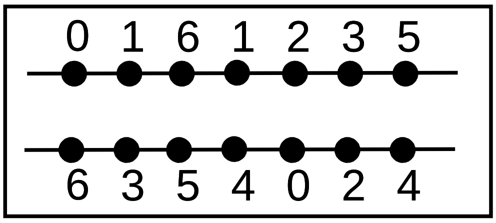
Length of the longest path in the vertical constraint graph

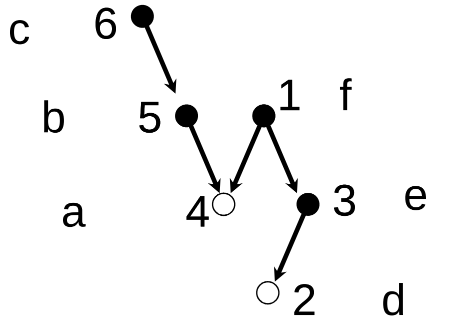
Lower bound on channel width : max{ max clique in HCG (channel density), longest path in VCG}

Constrained LEA

- Consider vertical constraints.
- Similar to the Left-edge algorithm.
- Modifications: Place a horizontal segment only if it does not have any unplaced descendants in the vertical constraint graph G_v. Place it on the bottommost available track above all its descendents in G_v.

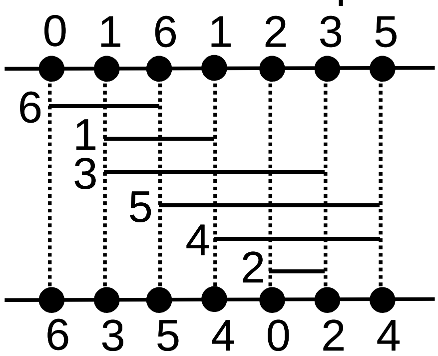
Constrained LEA : Example



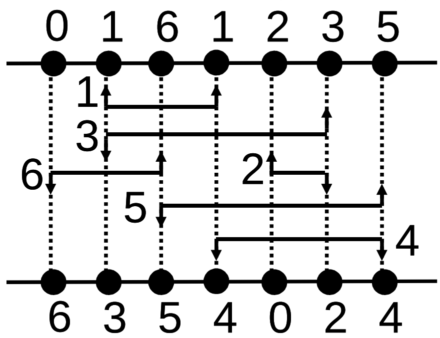


Vertical constraint graph

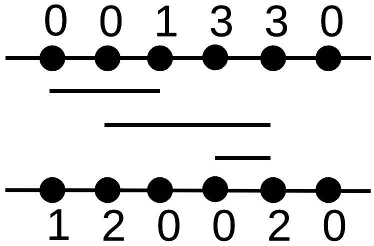
1. Sort the left end points.



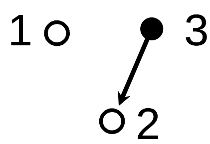
2. Place nets greedily.



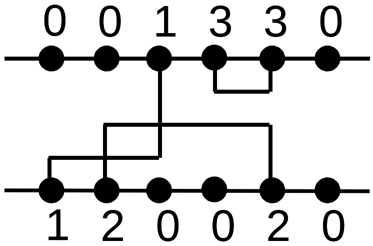
Drawback of constrained LEA



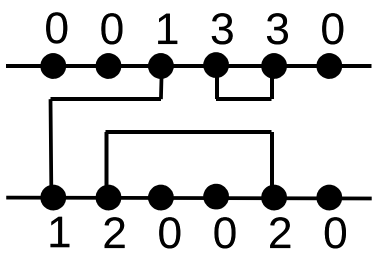
Vertical constraint graph



By Constrained Left-edge algorithm



There is a better solution...

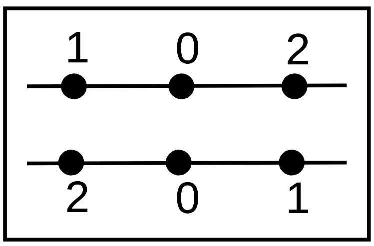


Drawback of constrained LEA

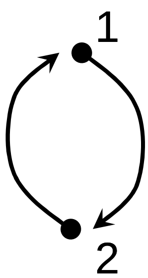
The Constrained Left-edge algorithm does not take care of the vertical and horizontal constraints together optimally.

Cycles in VCG: Dogleg Router

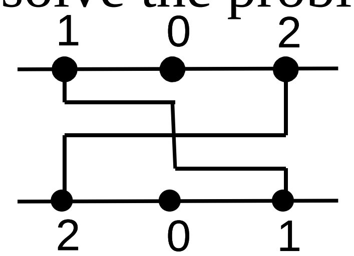
- If there is cycle in the vertical constraint graph, the channel is not routable.



Vertical constraint graph



- Dogleg can solve the problem.

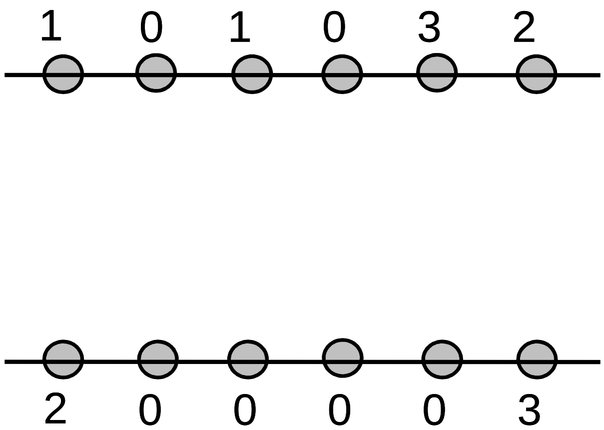


Reduce Channel height: Dogleg Router

- Since doglegs are not allowed, LEA places an entire net on a single track
- Doglegs can be used to reduce the channel width

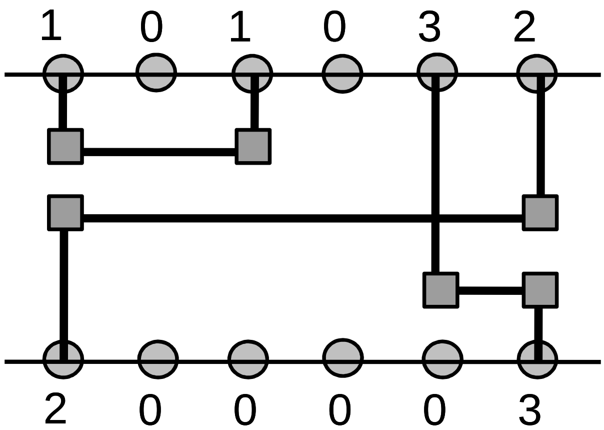
Reduce channel height: Dogleg Router

- Since doglegs are not allowed, LEA places an entire net on a single track
- Doglegs can be used to reduce the channel width



Reduce channel height: Dogleg Router

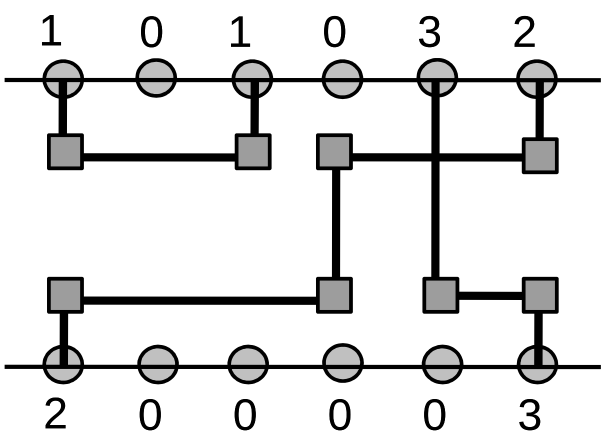
- Since doglegs are not allowed, LEA places an entire net on a single track
- Doglegs can be used to reduce the channel width



LEA
(Without dogleg)
3 tracks

Reduce channel height: Dogleg Router

- Since doglegs are not allowed, LEA places an entire net on a single track
- Doglegs can be used to reduce the channel width



With dogleg
2 tracks

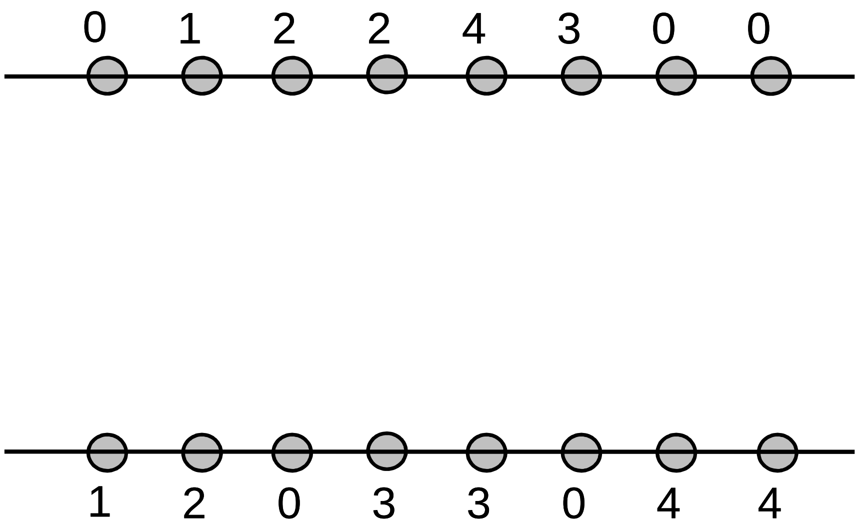
Dogleg Router

- D. N. Deutsch, "A Dogleg Channel Router", DAC 1976, pages 425-433
- Allows multi-terminal nets and vertical constraints
- Long multi-terminal nets (e.g., power, clock) can be broken into a series of 2-terminal subnets and each subnet can be routed on a different track using doglegs
- Cyclic vertical constraint can not be handled

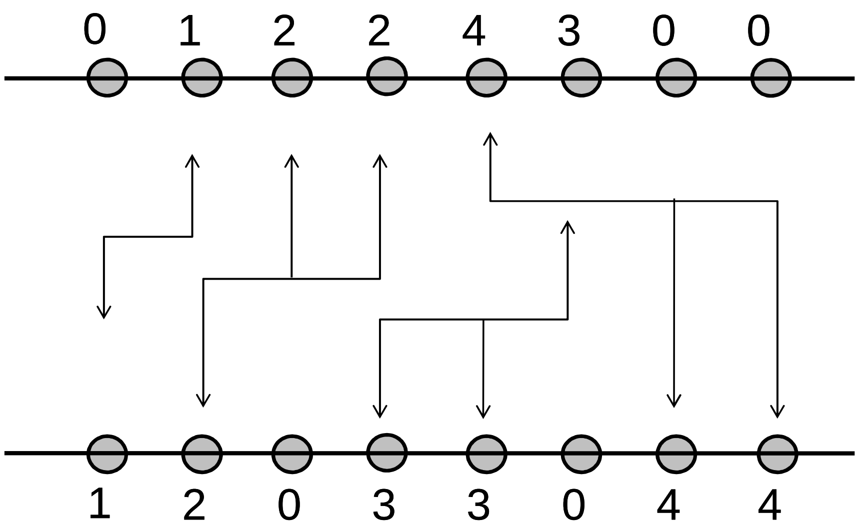
Dogleg Router: Algorithm

- **Step 1**:
  – If cycle exists in the VCG, return with failure.
- **Step 2**:
  – Split each multi-terminal net into a sequence of 2-terminal nets.
    - A net 2 .. 2 .. 2 will get broken as 2a .. 2a 2b .. 2b.
  – HCG and VCG gets modified accordingly.
- **Step 3**:
  – Apply the extended left-edge algorithm to the modified problem.

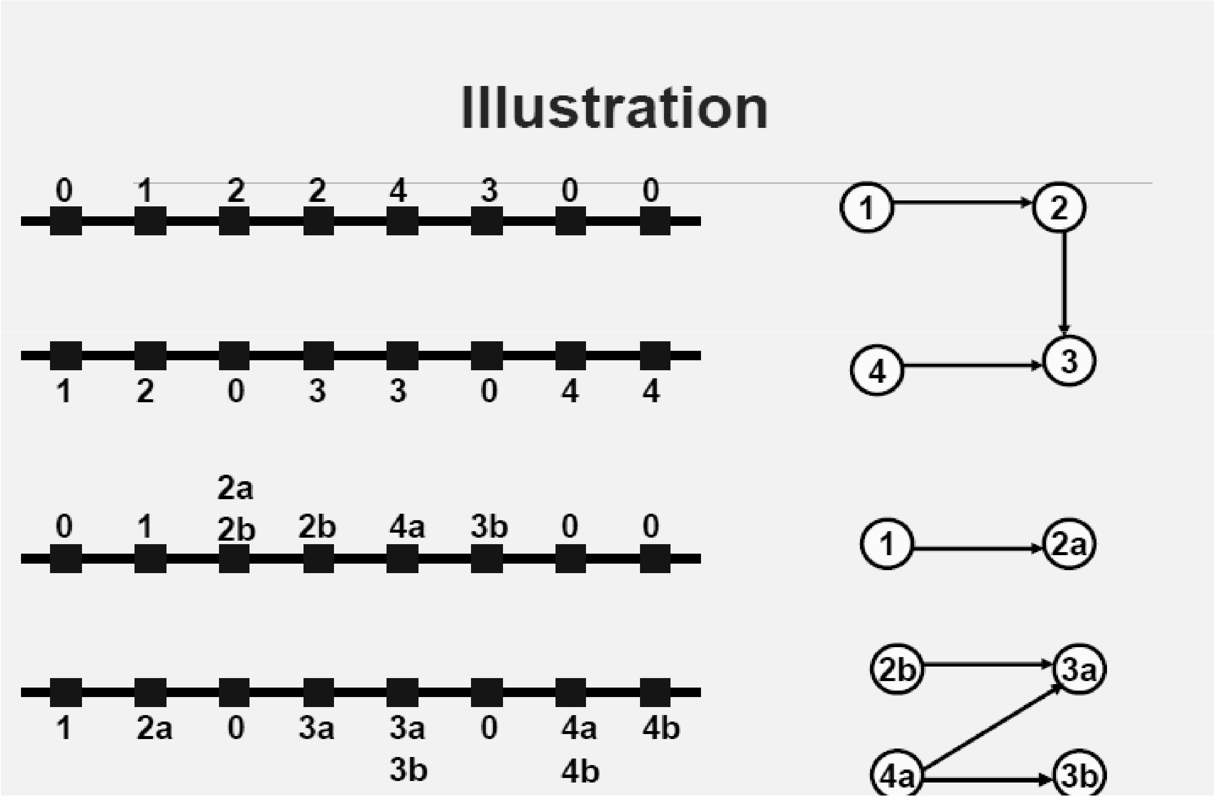
Dogleg Router



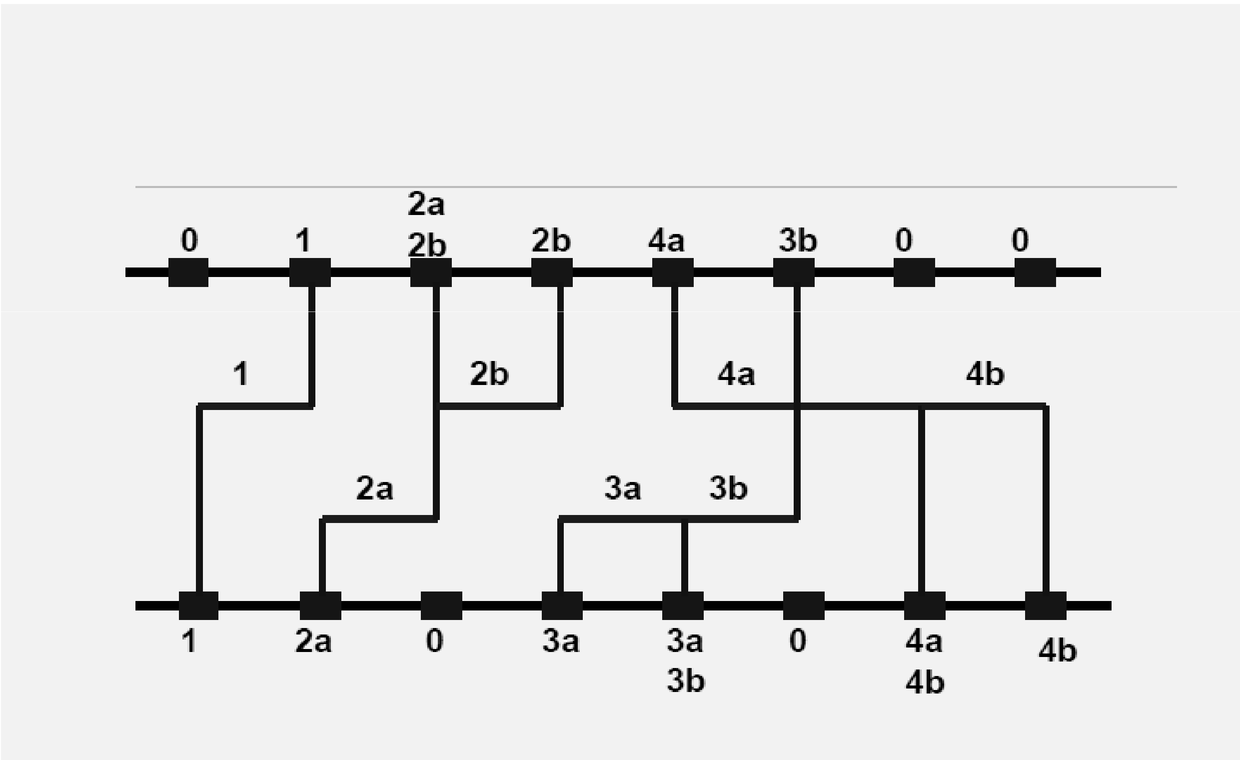
Dogleg Router



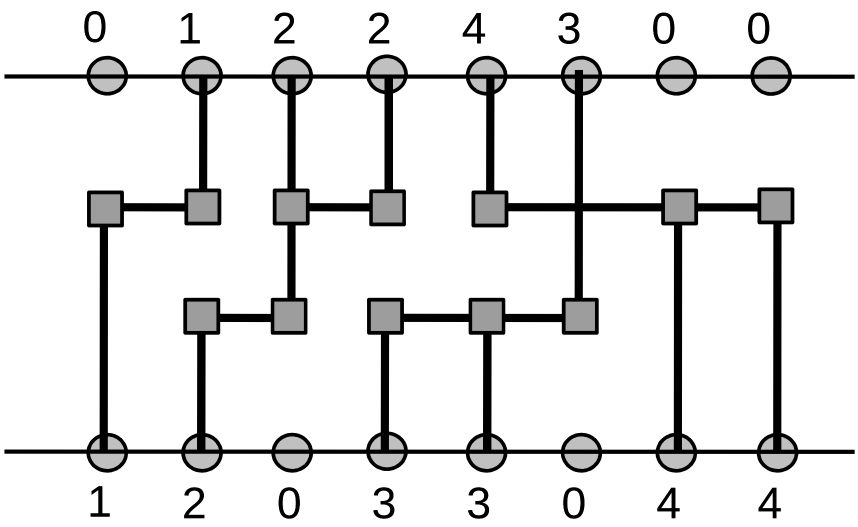
Dogleg Router



Dogleg Router



Dogleg Router



Net Merge Channel Router

Source:

"Efficient Algorithms for Channel Routing"

by T. Yoshimura and E. Kuh

IEEE Trans. On Computer-Aided Design of Integrated Circuits and Systems.

Vol. CAD-1, pp25-35, Jan 1982

Yoshimara and Kuh's Method

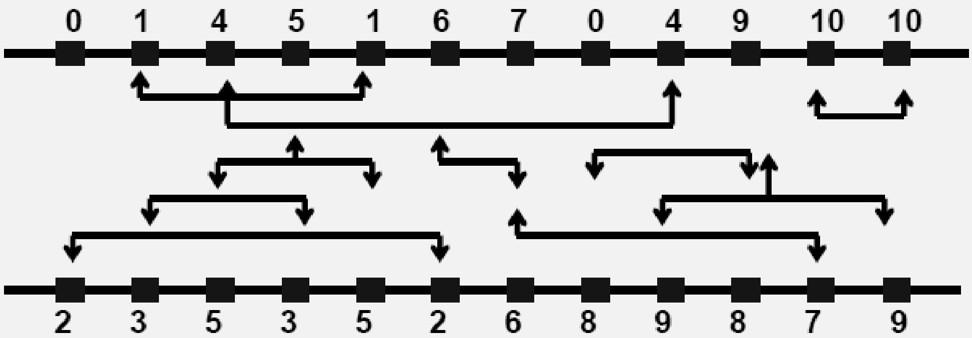
- Nets are partitioned into different zones based on horizontal segments of different nets and their constraints
- Proceeds from left to right of the channel and merges nets from adjacent zones
- Does not allow doglegs and cannot handle vertical constraint cycles

Net Merge Channel Router

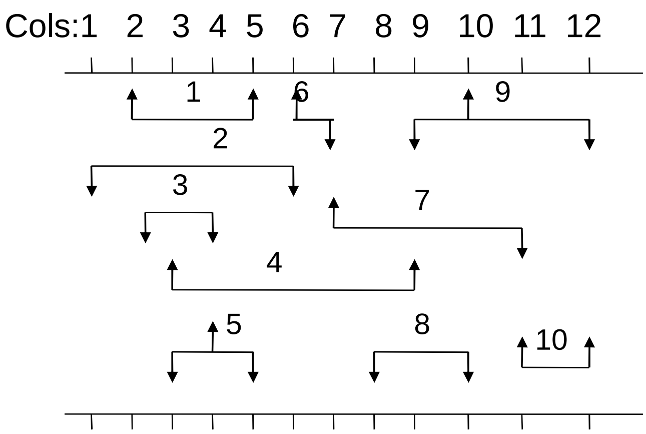
- Due to Yoshimura and Kuh.
- Basic idea:
  - If there is a path of length *p* in the VCG, at least *p* horizontal tracks are required to route the channel.
  - Try to minimize the longest path in the VCG.
  - Merge nodes of VCG to achieve this goal.
- Does not allow doglegs or cycles in the VCG.
- How does it work?
  - Partition the routing channel into a number of regions called "zones".
  - Nets from adjacent zones are merged.
    - Merged nets are treated as a "composite net" and assigned to a single track.

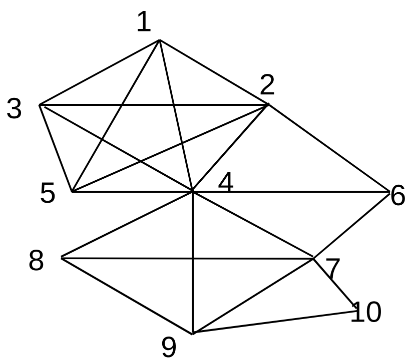
Contd.

- Key steps of the algorithm:
  a) Zone representation
  b) Net merging
  c) Track assignment
- An example:



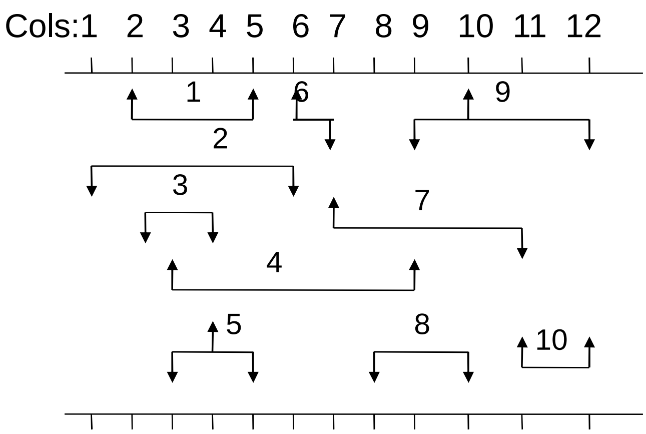
Yoshimara and Kuh's Method

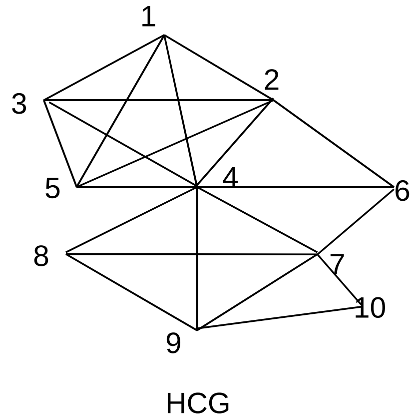


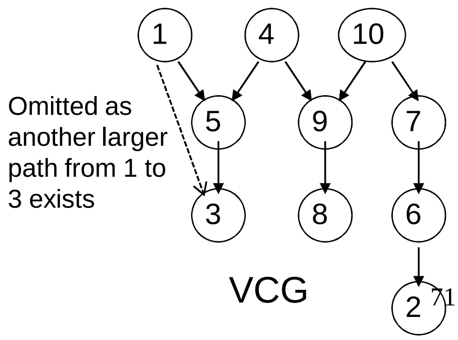


HCG

Yoshimara and Kuh's Method



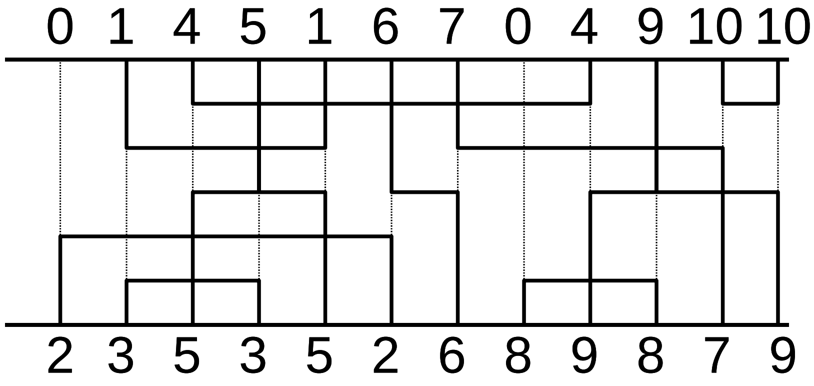




Step 1: Zone Representation

- Let *S(i)* denote the set of nets whose horizontal segments intersect column *i*.
- Take only those *S(i)* which are maximal, that is, not a proper subset of some other *S(j).*
- Define a zone for each of the maximal sets.
- In terms of HCG / interval graph, a zone corresponds to a maximal clique in the graph.

Yoshimara and Kuh's Method



Scan columnwise, add nets in the current zone whose horizontal segments intersect current column

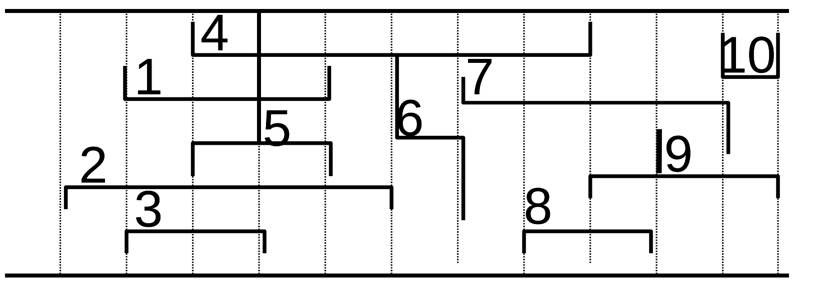
2	1	1	1	1	2	4	4	4	7	7	9
	2	2	2	2	4	6	7	7	8	9	10
	3	3	3	4	6	7	8	8	9	10	
		4	4	5				9			
		5	5								

Zone: 1 2 3 4 5

A new zone appears when some intervals begin after some intervals end

Yoshimara and Kuh's Method

Cols: 1 2 3 4 5 6 7 8 9 10 11 12



Scan columnwise, add nets in the current zone whose horizontal segments intersect current column

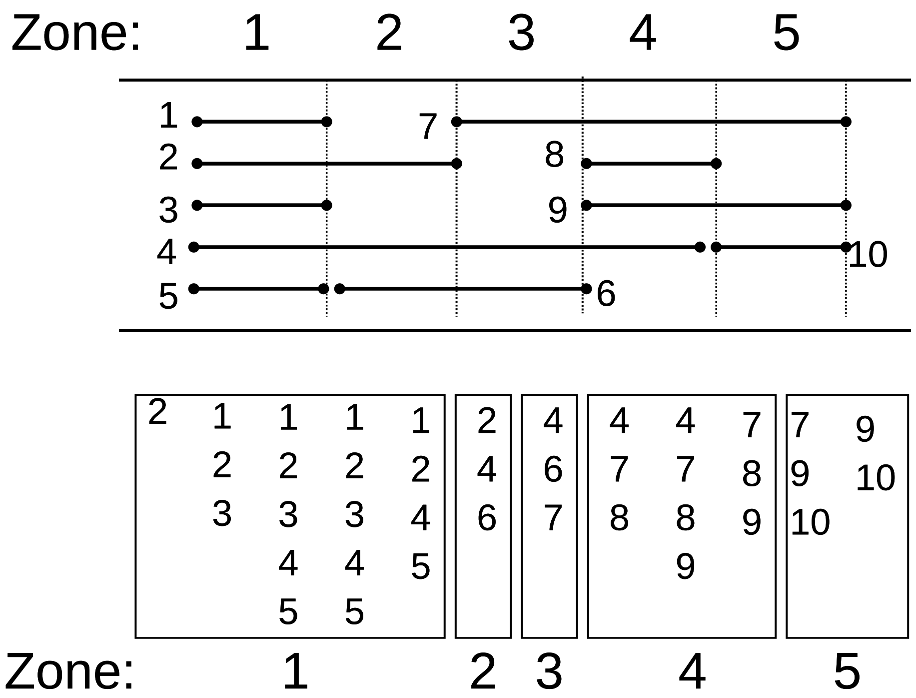
2	1	1	1	1	2	4	4	4	7	7	9
	2	2	2	2	4	6	7	7	8	9	10
	3	3	3	4	6	7	8	8	9	10	
		4	4	5				9			
		5	5								

Zone: 1 2 3 4 5

Components of a zone is either superset or subset of the set so far constructed

A new zone appears when some intervals begin after some intervals end

Yoshimara and Kuh's Method

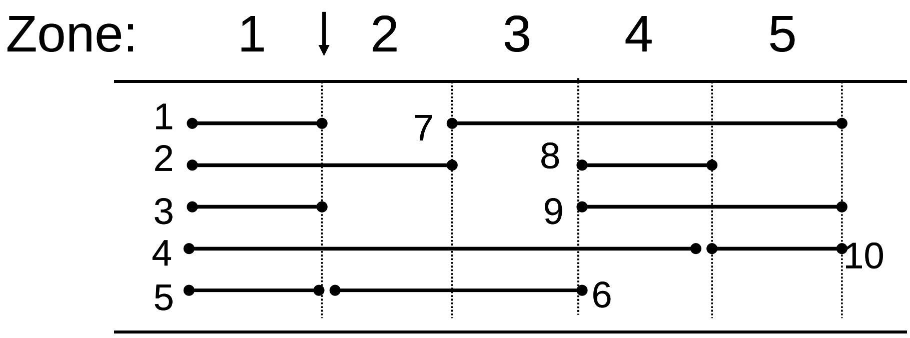


A new zone appears when some intervals begin after some intervals end

Yoshimara and Kuh's Method

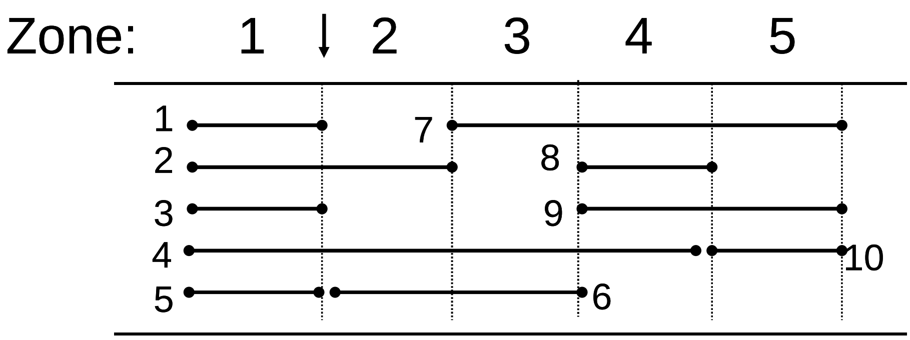
- Merges nets as long as two nets from different zones can be merged
- In each iteration, the nets ending in zone z_i are appended to the list Left and the nets starting in z_{i+1} are kept in list Right
- Nets in Left and Right are merged so as to minimize the increase in the longest path length in VCG
  - Make sure VCG remains acyclic

Yoshimara and Kuh’s Method



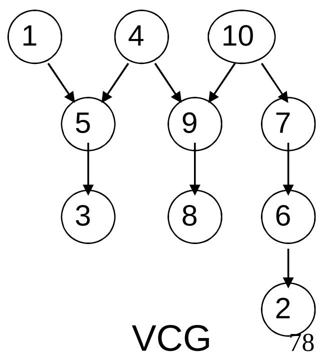
Left = {1, 3, 5}
Right = {6}

Yoshimara and Kuh’s Method



Left = {1, 3, 5}
Right = {6}

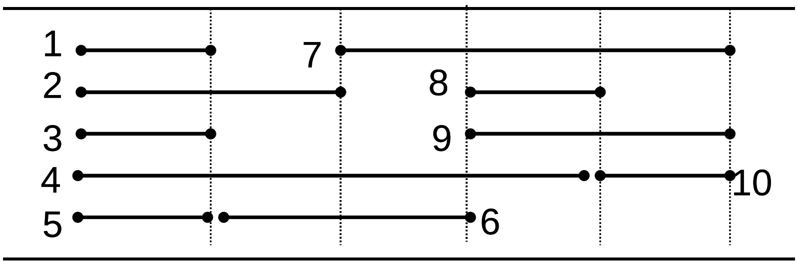
{1,6}, {3,6} or {5,6} ?



VCG

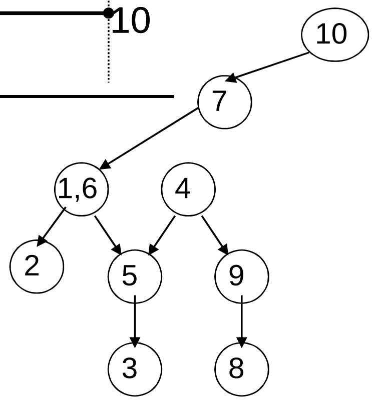
Yoshimara and Kuh's Method

Zone: 1 2 3 4 5



Left = {1, 3, 5}
Right = {6}

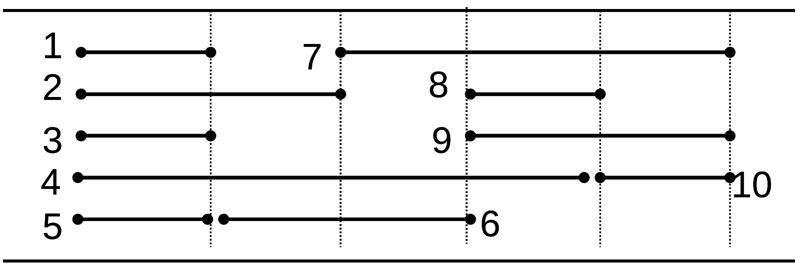
Length of the longest path: 4



VCG

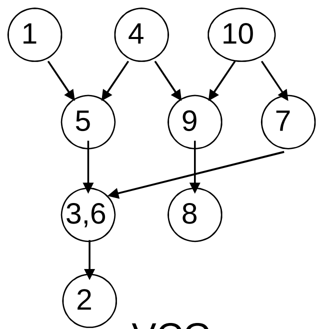
Yoshimara and Kuh's Method

Zone: 1 2 3 4 5



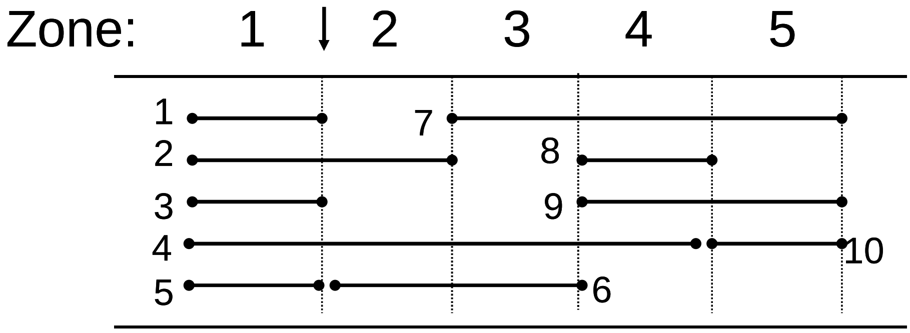
Left = {1, 3, 5}
Right = {6}

Length of the longest path: 3



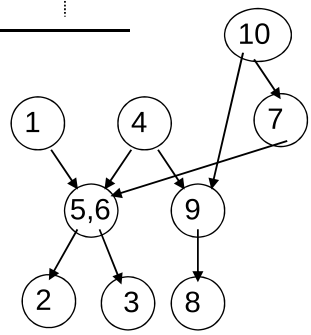
VCG

Yoshimara and Kuh’s Method



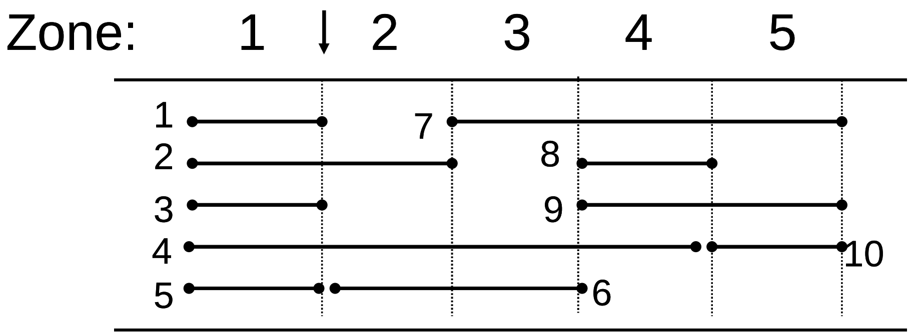
Left = {1, 3, 5}
Right = {6}

Length of the longest path: 3



VCG

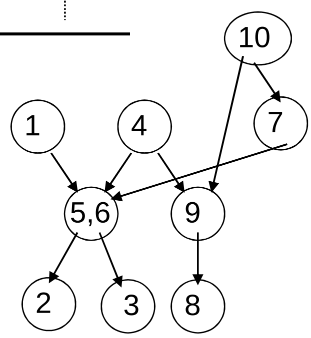
Yoshimara and Kuh’s Method



Left = {1, 3, 5}
Right = {6}

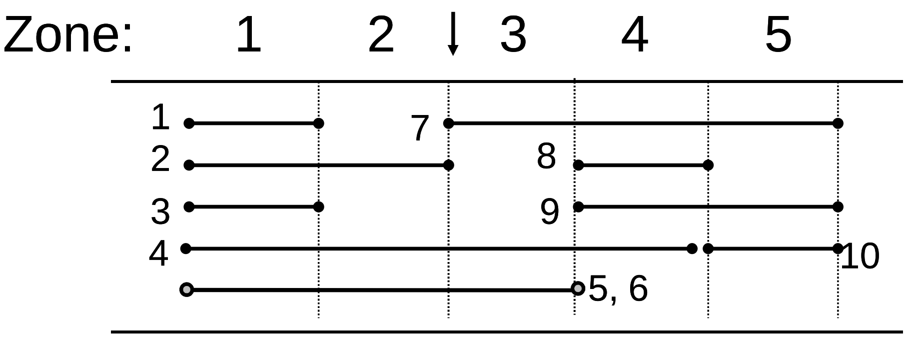
Hence, either 3,6 or 5,6 can be merged

Let us merge 5 and 6

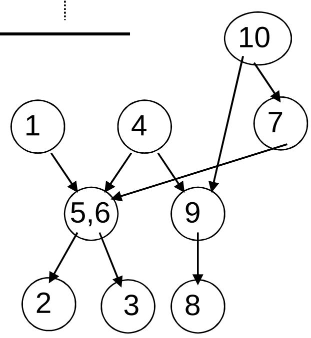


VCG

Yoshimara and Kuh's Method

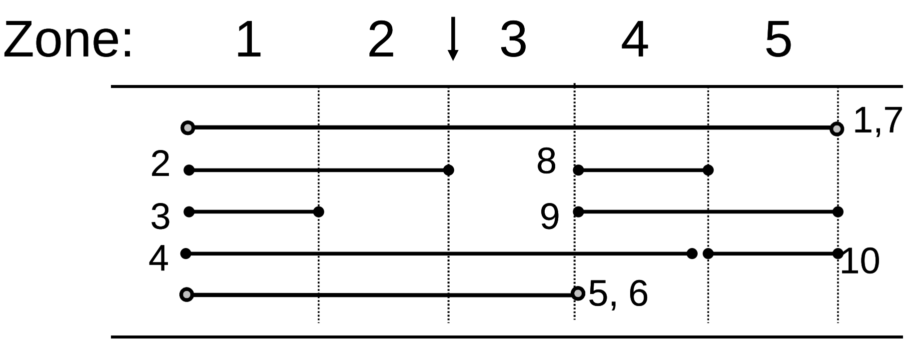


Left = {1, 2, 3}
Right = {7}

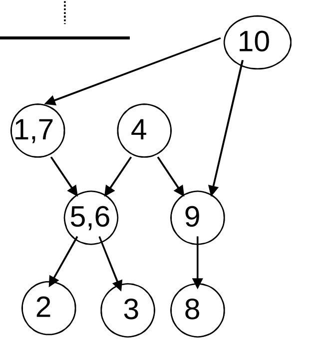


VCG

Yoshimara and Kuh's Method

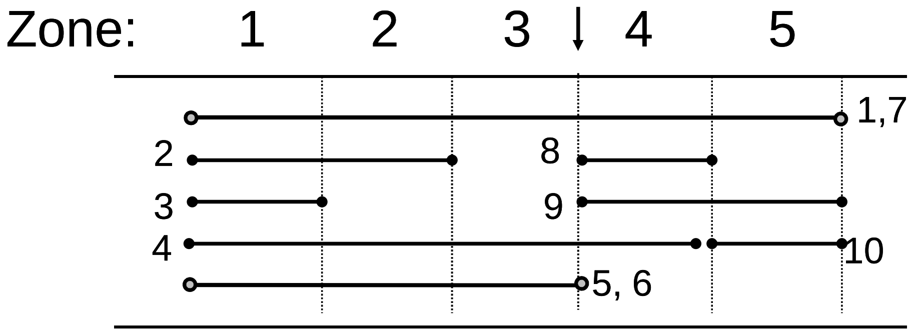


Left = {1, 2, 3}
Right = {7}

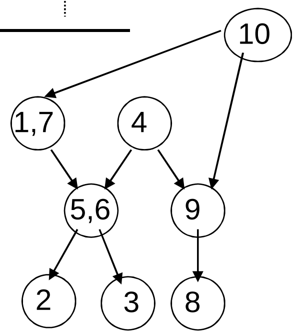


VCG

Yoshimara and Kuh's Method

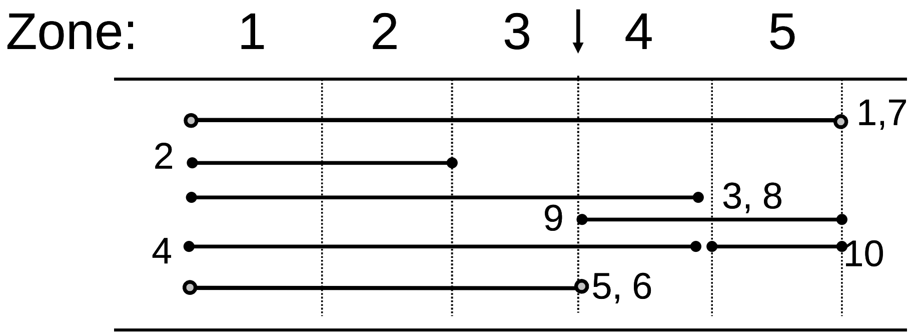


Left = {2, 3, 5.6}
Right = {8, 9}

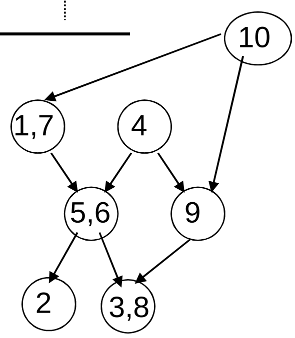


VCG

Yoshimara and Kuh's Method

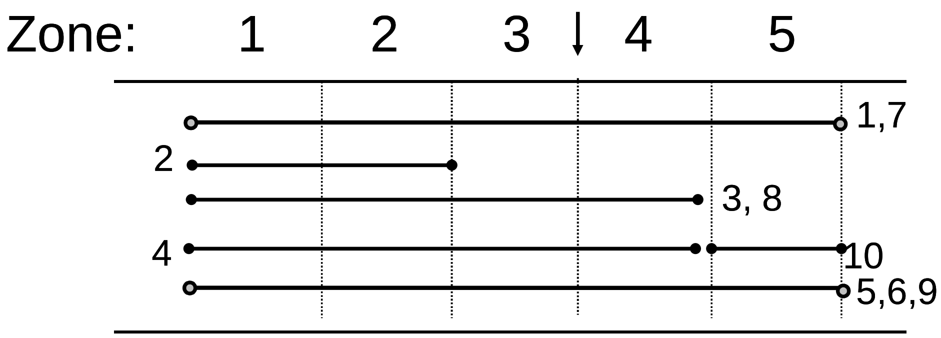


Left = {2, 3, 5.6}
Right = {8, 9}

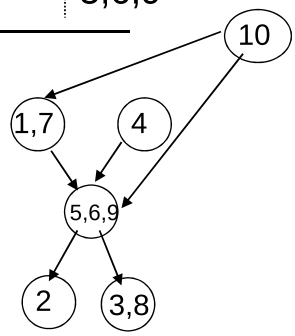


VCG

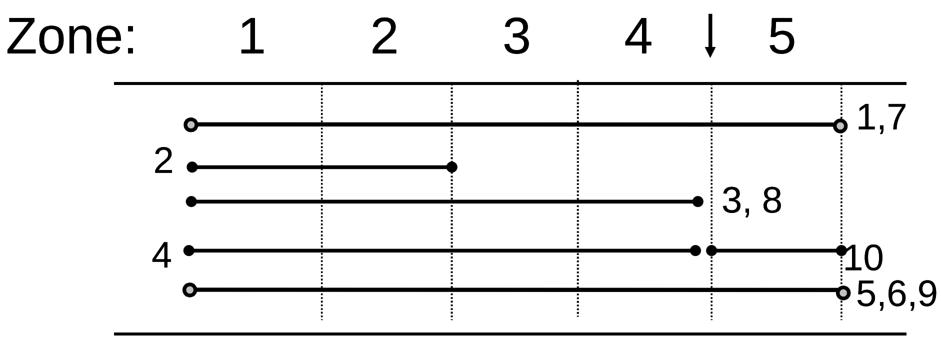
Yoshimara and Kuh's Method



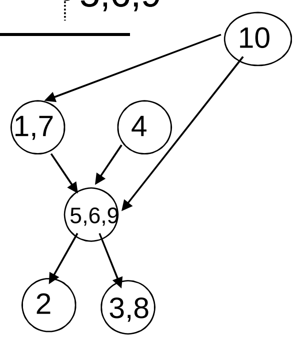
Left = {2, 3, 5.6}
Right = {8, 9}



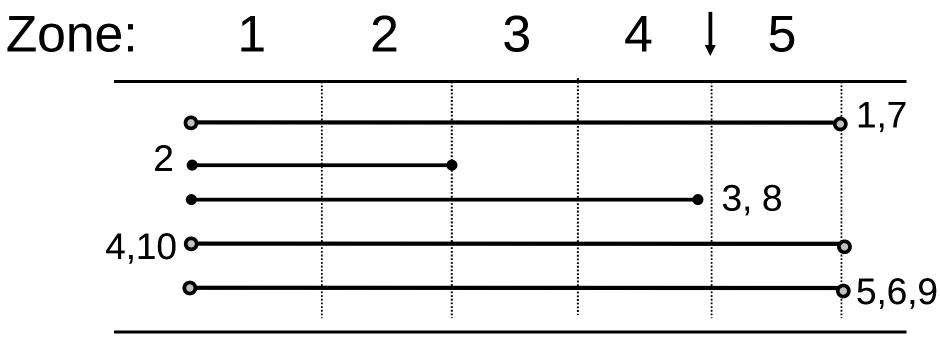
Yoshimara and Kuh's Method



Left = {2, 3.8, 4}
Right = {10}

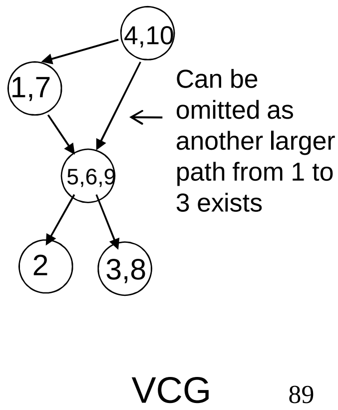


Yoshimara and Kuh’s Method



Left = {2, 3.8, 4}

Right = {10}



Yoshimara and Kuh’s Method