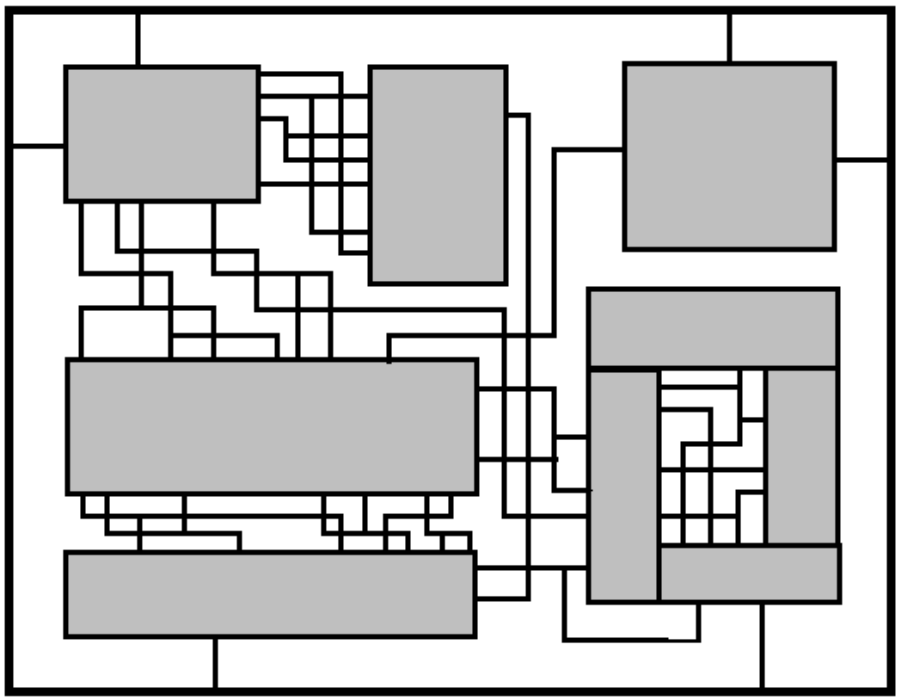
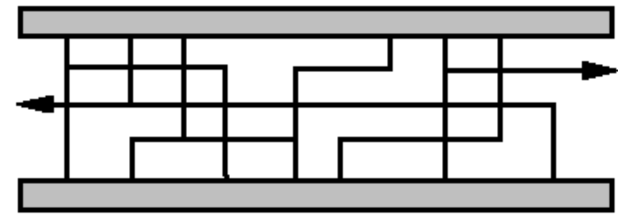
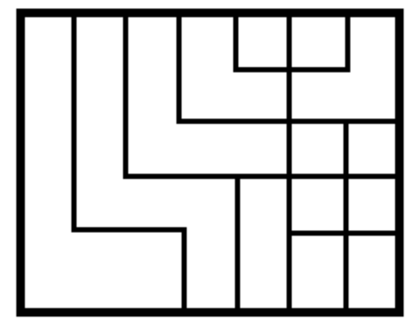
Detailed Routing



Detailed routing



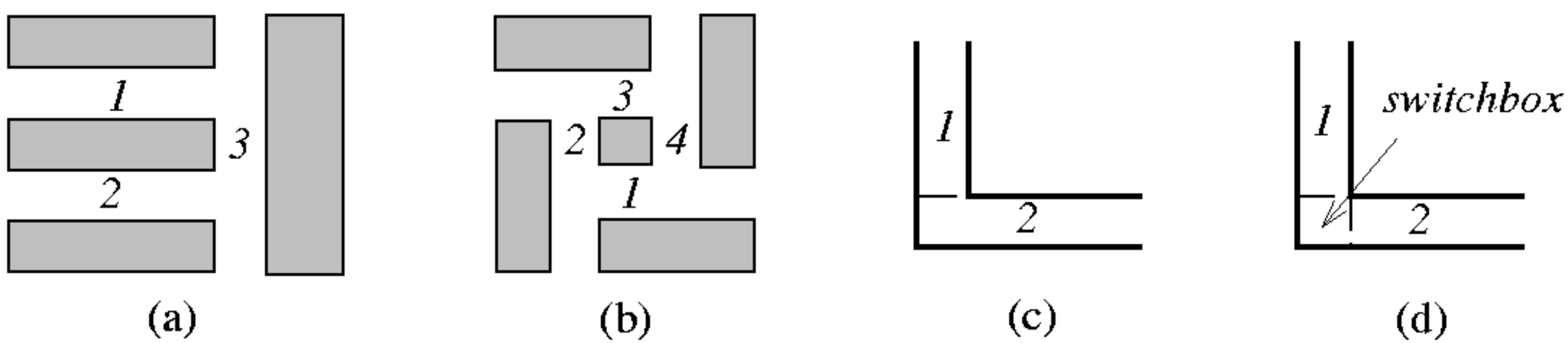
channel routing



switchbox routing

Order of Routing Regions

- (a) No conflicts in case of routing in the order of 1,2, and 3.
- (b) No ordering is possible to avoid conflicts.
- (c) The situation of (b) can be resolved by using L-channels.
- (d) An L-channel can be decomposed into two channels and a switchbox.

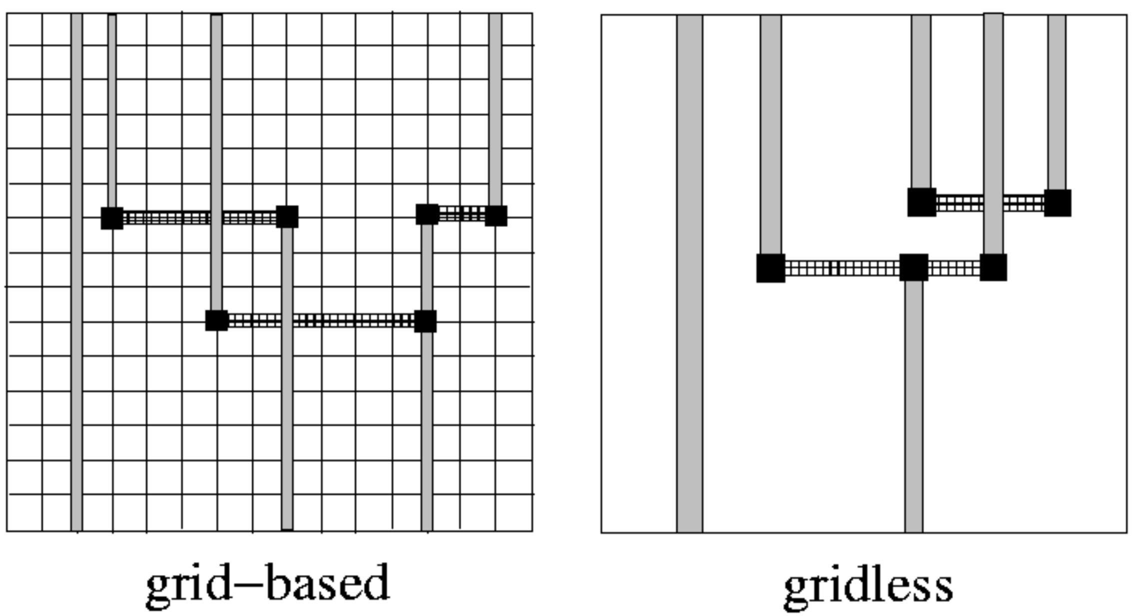


Routing Considerations

- Number of terminals (two-terminal vs. multi-terminal nets)
- Net widths (power and ground vs. signal nets)
- Via restrictions (stacked vs. conventional vias)
- Boundary types (regular vs. irregular)
- Number of layers (two vs. three, more layers?)
- Net types (critical vs. non-critical nets)

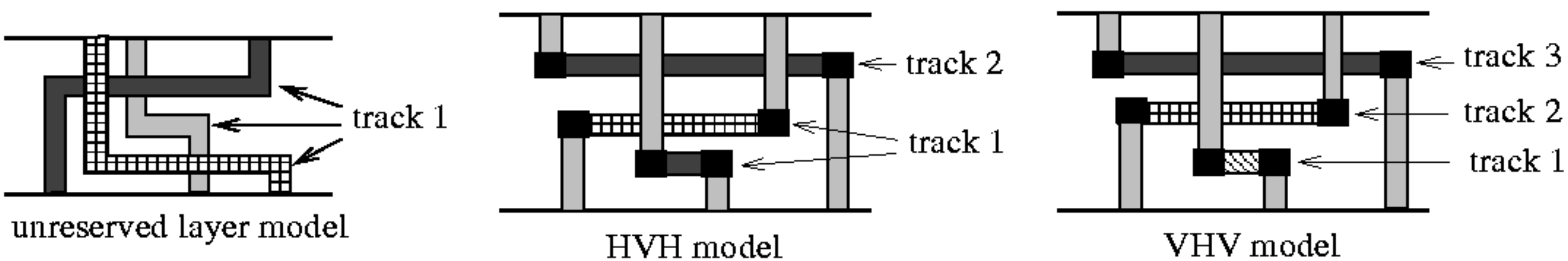
Routing Models

- **Grid-based model:**
  - A grid is super-imposed on the routing region.
  - Wires follow paths along the grid lines.
- **Gridless model:**



Models for Multi-Layer Routing

- **Unreserved layer model:** Any net segment is allowed to be placed in any layer.
- **Reserved layer model:** Certain type of segments are restricted to particular layer(s).
  - Two-layer: HV (horizontal-vertical), VH
  - Three-layer: HVH, VHV



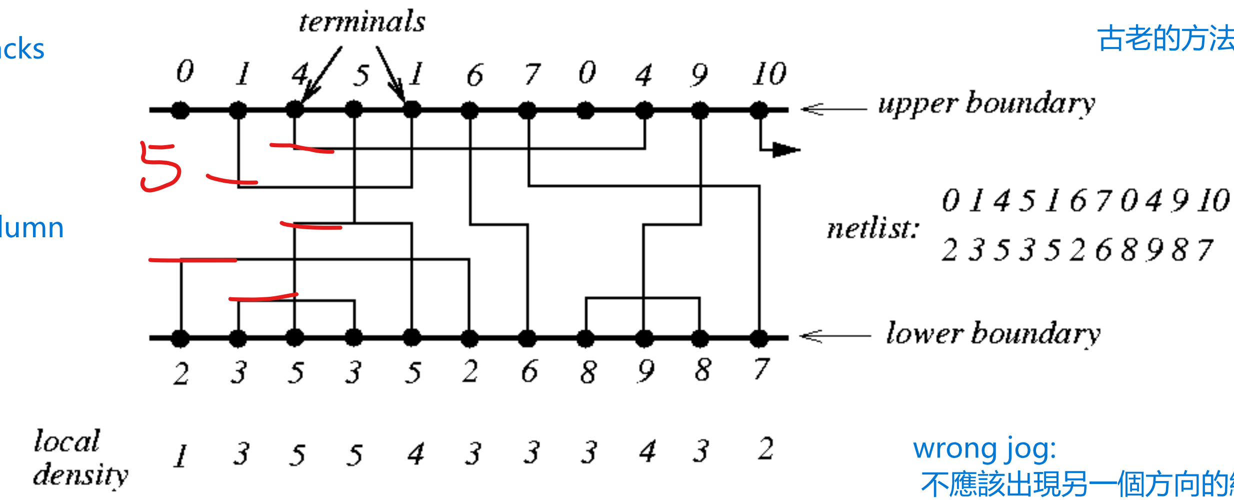
3 types of 3−layer models

Terminology for Channel Routing

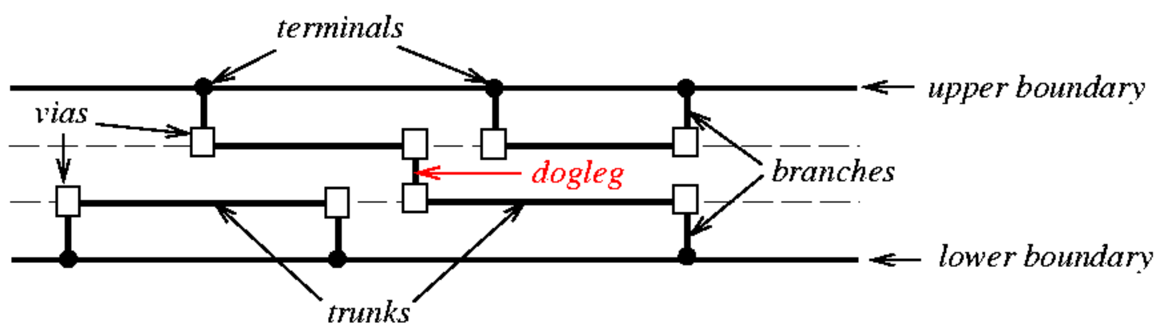
會用到5個tracks

古老的方法

column



wrong jog:
不應該出現另一個方向的線段



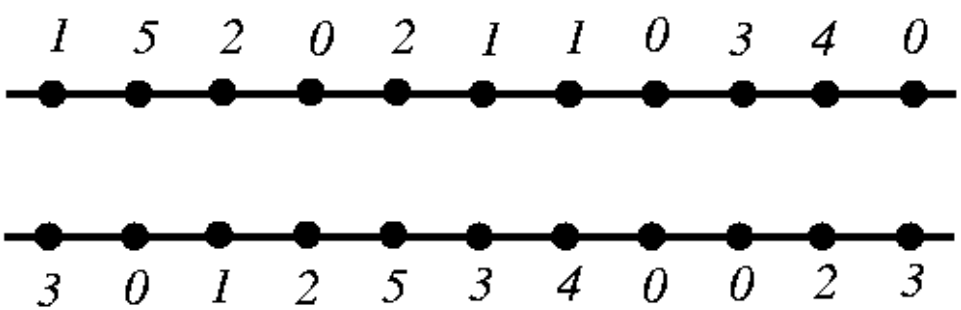
- Local density at column *i*: total # of nets that crosses column *i*.
- Channel density: maximum local density; # of horizontal tracks required ≧ channel density.

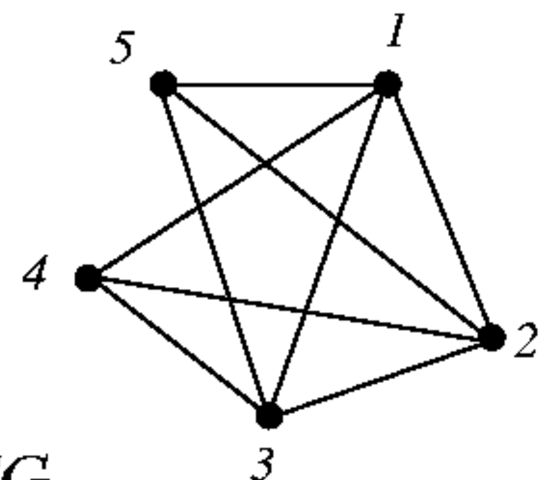
Channel Routing

- **Assignments of horizontal segments of nets to tracks.**
- **Assignments of vertical segments to connect**
  - horizontal segments of the same net in different tracks, and
  - the terminals of the net to horizontal segments of the net.
- **Horizontal and vertical constraints must not be violated.**
  - Horizontal constraints between two nets: The horizontal span of two nets overlaps each other.
  - Vertical constraints between two nets: There exists a column such that the terminal on top of the column belongs to one net and the terminal on bottom of the column belongs to the other net.
- **Objective: Channel height is minimized** (i.e., channel area is minimized).

Horizontal Constraint Graph (HCG)

- HCG $G = (V, E)$ is an **undirected** graph where
  - $V = \{v_i | v_i$ represents a net $n_i\}$
  - $E = \{(v_i, v_j)|$ a horizontal constraint exists between n_i and $n_j\}$.
- For graph G: vertices $\Leftrightarrow$ nets; edge $(i, j) \Leftrightarrow$ net i overlaps net j.

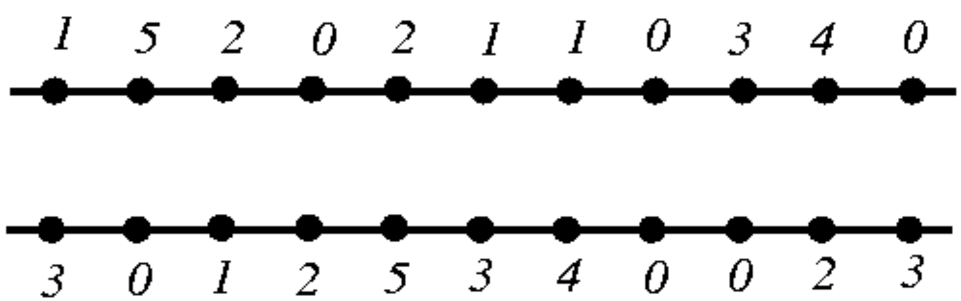


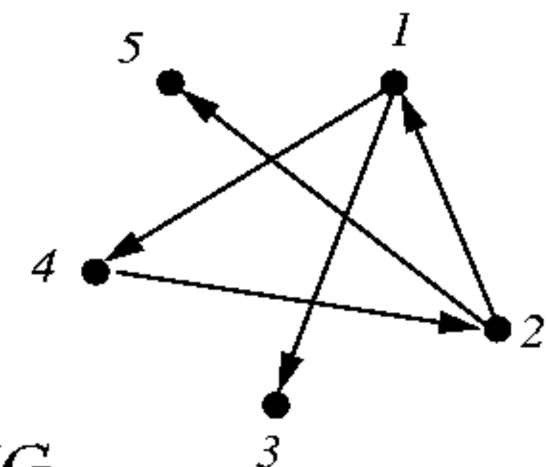


A routing problem and its HCG.

Vertical Constraint Graph (VCG)

- VCG $G = (V, E)$ is a **directed** graph where
  - $V = \{v_i | v_i$ represents a net $n_i\}$
  - $E = \{(v_i, v_j)|$ a vertical constraint exists between n_i and $n_j\}$.
- For graph G: vertices $\Leftrightarrow$ nets; edge $i \rightarrow j \Leftrightarrow$ net i must be above net j.





A routing problem and its VCG.

2-L Channel Routing: Basic Left-Edge Algorithm

- Hashimoto & Stevens, “Wire routing by optimizing channel assignment within large apertures,” DAC, 1971.
- **No vertical constraint.**
- HV-layer model is used.
- **Doglegs are not allowed.**
- Treat each net as an interval.
- Intervals are sorted according to their left-end x-coordinates.
- Intervals (nets) are routed one-by-one according to the order.
- For a net, tracks are scanned from top to bottom, and the first track that can accommodate the net is assigned to the net.
- Optimality: produces a routing solution with the minimum # of tracks (if no vertical constraint).

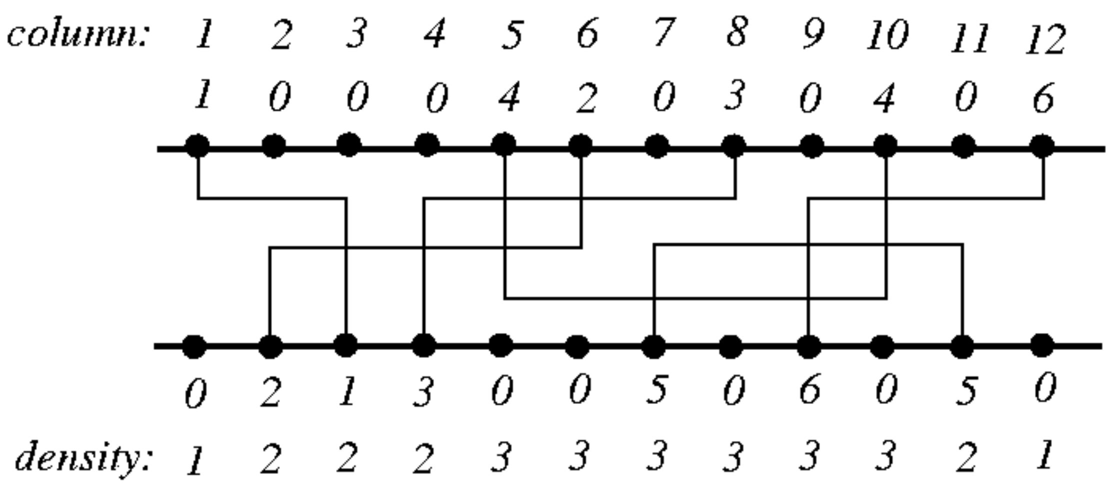
Basic Left-Edge Algorithm

Algorithm: Basic_Left-Edge($U, track[j]$**)**
U: set of unassigned intervals (nets) $I_1, \ldots, I_n$;
$I_j = [s_j, e_j]$: interval j with left-end x-coordinate s_j and right-end e_j;
$track[j]$: track to which net j is assigned.

begin
$U \leftarrow \{I_1, I_2, \ldots, I_n\}$;
$t \leftarrow 0$;
while $(U \neq \emptyset)$ **do**
    $t \leftarrow t + 1$;
    $watermark \leftarrow 0$;
    while (there is an $I_j \in U$ s.t. $s_j > watermark$) **do**
        Pick the interval $I_j \in U$ with $s_j > watermark$,
        nearest $watermark$;
        $track[j] \leftarrow t$;
        $watermark \leftarrow e_j$;
        $U \leftarrow U - \{I_j\}$;
end

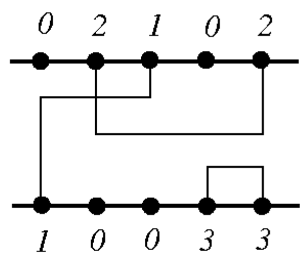
Example

- $U = \{I_1, I_2, \ldots, I_6\}$; $I_1 = [1,3]$, $I_2 = [2,6]$, $I_3 = [4,8]$, $I_4 = [5,10]$, $I_5 = [7,11]$, $I_6 = [9,12]$.
- $t = 1$:
  - Route I_1: watermark = 3;
  - Route I_3: watermark = 8;
  - Route I_6: watermark = 12;
- $t = 2$:
  - Route I_2: watermark = 6;
  - Route I_5: watermark = 11;
- $t = 3$: Route I_4

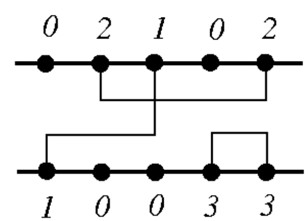


Basic Left-Edge Algorithm

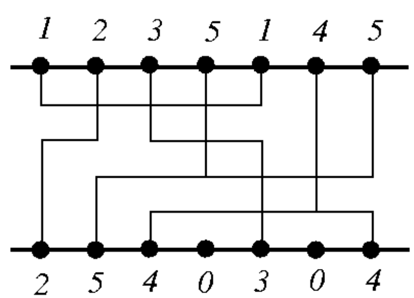
- If there is no vertical constraint, the basic left-edge algorithm is optimal.
- If there is any vertical constraint, the algorithm no longer guarantees optimal solution.

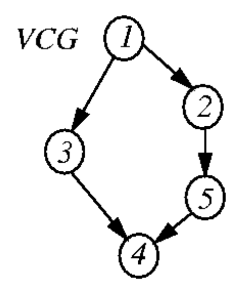


result from basic left–edge algorithm 3 tracks



optimal routing: 2 tracks





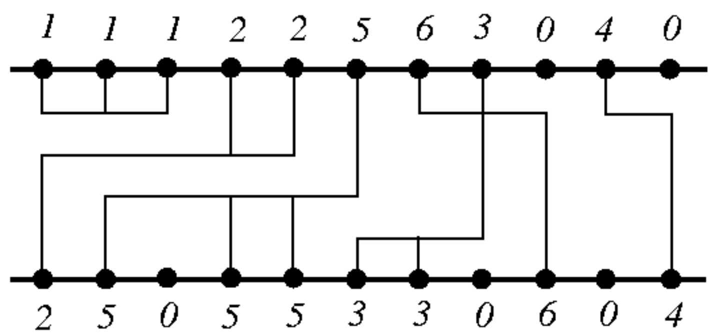
Constrained Left-Edge Algorithm

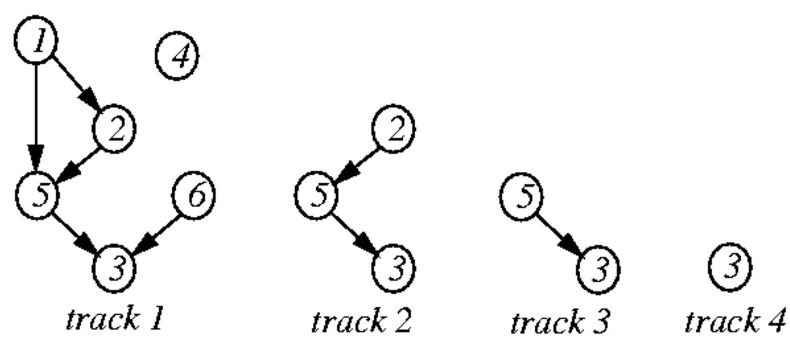
Algorithm: Constrained_Left-Edge($U, track[j]$)
U: set of unassigned intervals (nets) $I_1, \ldots, I_n$;
$I_j = [s_j, e_j]$: interval j with left-end x-coordinate s_j and right-end e_j;
$track[j]$: track to which net j is assigned.

```
begin
U ← {I_1, I_2, ..., I_n};
t ← 0;
while (U ≠ ∅) do
    t ← t + 1;
    watermark ← 0;
    while (there is an unconstrained I_j ∈ U s.t. s_j > watermark) do
        Pick the interval I_j ∈ U that is unconstrained,
        with s_j > watermark, nearest watermark;
        track[j] ← t;
        watermark ← e_j;
        U ← U − {I_j};
end
```

Constrained Left-Edge Example

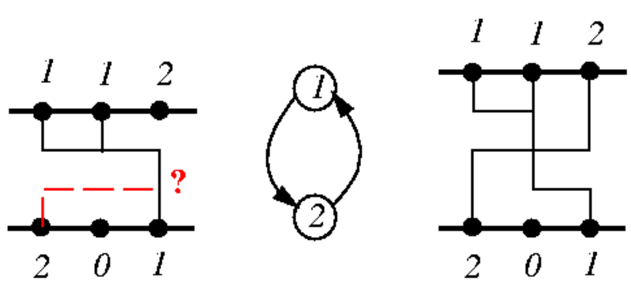
- $I_1 = [1,3]$, $I_2 = [1,5]$, $I_3 = [6,8]$, $I_4 = [10,11]$, $I_5 = [2,6]$, $I_6 = [7,9]$.
- Track 1: Route I_1 (cannot route I_3); Route I_6; Route I_4.
- Track 2: Route I_2; cannot route I_3.
- Track 3: Route I_5.
- Track 4: Route I_3.

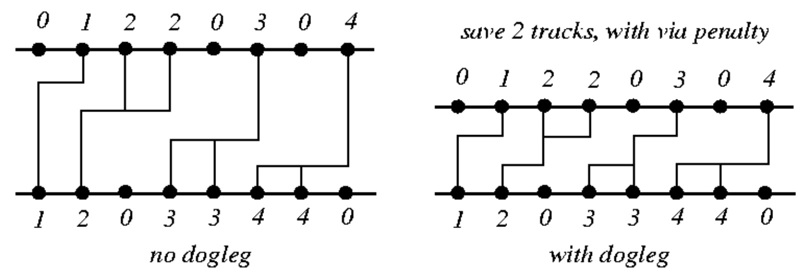




Dogleg Channel Router

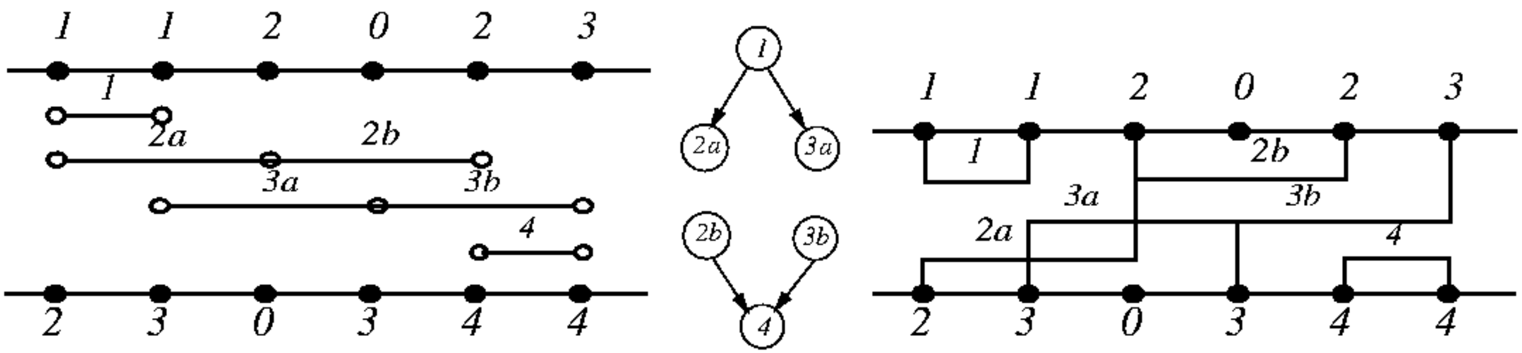
- Deutsch, “A dogleg channel router,” DAC, 1976.
- **Drawback of Left-Edge: cannot handle the cases with constraint cycles.**
  - **Doglegs** are used to resolve constraint cycle.
- **Drawback of Left-Edge: the entire net is on a single track.**
  - **Doglegs** are used to place parts of a net on different tracks to minimize channel height.
  - Might incur penalty for additional vias.





Dogleg Channel Router

- Each multi-terminal net is broken into a set of 2-terminal nets.
- Two parameters are used to control routing:
  - Range: Determine the # of consecutive 2-terminal subnets of the same net that can be placed on the same track.
  - Routing sequence: Specifies the starting position and the direction of routing along the channel.
- Modified Left-Edge Algorithm is applied to each subnet.



Over-the-Cell Routing

- Routing over the cell rows is possible due to the limited use of the 2nd (M2) mental layers within the cells.
- Divide the over-the-cell routing problem into 3 steps: (1) routing over the cell, (2) choosing the net segments, and (3) routing within the channel.