2-L Channel Routing: Basic Left-Edge Algorithm

- Hashimoto & Stevens, "Wire routing by optimizing channel assignment within large apertures," DAC-71.
- **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) $I_1, \ldots, I_n$;

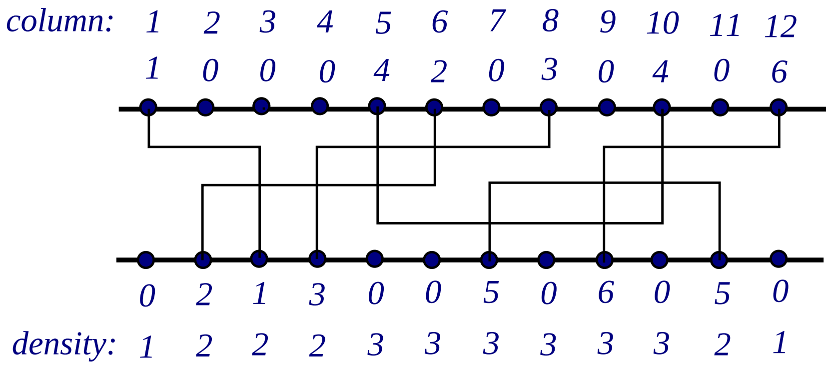
$I_j = [s_j, e_j]$: interval j with left-end x-coordinate s_j and right-end e_j;

$track[j]$: 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

Basic Left-Edge 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



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