Taxonomy of VLSI Routers

- Routers
  - Global
    - Graph Search
      - Steiner
      - Iterative
  - Detailed
    - Restricted
      - River
      - Switchbox
      - Channel
    - Greedy
    - Left-Edge
  - General Purpose
    - Maze
    - Line Probe
    - Line Expansion
  - Specialized
    - Power & Ground
    - Clock

EE244 Fall 97

1.2.1

1.2.2
Maze Route: Lee Path Connection Algorithm

Point-to-point, single layer

Wavefront propagation

Backtracking

Prioritizing the search: A*

Line Probe: Hightower & Tabuchi

Also called “Ray”, “Line search”, “Gridless router”

Usually implemented on a grid
Line Search: Hightower & Tabuchi

- Note order-dependence A-B versus B-A

Pattern Route: Soukup

Use line-search to blockage and then revert to maze-router
Channel Routing

- Basic Terminology:
  - Fixed pin positions on top and bottom edges
  - Classical channel: no nets leave channel
  - Three-sided channel possible

Routing Region Definition

- Use of Slicing Structure for ordering (Otten)
Graph representations:
- Vertical Constraint Graph (VCG)
- Horizontal Interval Graph (HIG)
**Vertical Constraint Graph**

1. Proceed column by column (left to right)
2. Make connections to all pins in that column
3. Free up tracks by collapsing as many tracks as possible to collapse nets
4. Shrink range of rows occupied by a net by using doglegs
5. If a pin cannot enter a channel, add a track

**Greedy Router: Rivest & Fiduccia**

- Proceed column by column (left to right)
- Make connections to all pins in that column
- Free up tracks by collapsing as many tracks as possible to collapse nets
- Shrink range of rows occupied by a net by using doglegs
- If a pin cannot enter a channel, add a track
- $O(pins)$ time
Greedy Router

- Three Parameters Control Algorithm:
  - Initial_Channel_Width, Minimum_Jog_length, Steady_Net_Constant

- Rule-Based Approach
  - Rule 1: Make feasible top and bottom connections in minimal manner
  - Rule 2: Free up as many tracks as possible by collapsing split nets
  - Rule 3: Add jogs to minimize distance between split nets (but no shorter than minimum_jog_length)
  - Rule 4: Add jogs where possible to raise rising nets and lower falling nets
  - Rule 5: Widen channel if needed to make top or bottom connections. Add tracks in the middle of the channel
  - Rule 6: Extend the channel to a new column to complete unconnected net segments.

Greedy Routing Example

![Diagram of a greedy router example]
**Left-Edge Algorithm (LEA)**

*(Hashimoto & Stevens)*

- Process horizontal tracks in order from top to bottom of channel and assign nets to tracks at each stage (assuming no cyclic constraints in VCG)
  
  For track $i$, choose the “left-most net” of those who have no ancestors in the VCG
  
  Repeat process until no additional nets can be assigned to the track
  
  Move to track $i+1$

- Guarantees a solution since there are no cyclic constraints permitted in VCG

- In example, VCG permits $(1,4,10)$ in first track, but only $(1,10)$, $(4,10)$ based on HIG; choose $(1,10)$
"Merging of Nets"
(Yoshimura & Kuh)

◆ On the assumption of no cyclic constraints, nets that can be placed on the same track can be merged in the VCG, simplifying the VCG.

Merging of Nets

◆ Definition: Let $i$ and $j$ be nets for which the following holds:

(a) $i$ and $j$ are not adjacent in the HIG

(b) There is no direct path between $i$ and $j$ in the VCG

Then these nets can be assigned to the same track and hence they can be merged in the VCG

◆ Merging Operation:

1. Combine nodes $i$ and $j$ into node $i\bullet j$ in VCG
2. Update zone representation such that $i\bullet j$ occupies zones of $i$ and $j$ and all zones in between
Merging of Nets: Example

Introducing Doglegs

Finding the optimal locations for doglegs is an NP-Complete problem
Cyclic Constraints

- VCG may contain a “cyclic constraint” and therefore cannot be routed with algorithms above.
- The use of “doglegs” (a net using two or more tracks) will permit breaking these constraints.

YACR: Pattern Routing

- Performs normal LEA, track assignment for nets that do not violate vertical constraints.
- Connect remaining trunks using “maze router” (actually pattern router).
- Three patterns: Maze_1 (route under), Maze_2 (one dogleg), Maze_3 (two doglegs).
- If patterns fail, add track.
- Will add end columns, if needed, to resolve otherwise impossible cyclic constraints in VCG.