Routing in Integrated Circuits
A. Kahng
K. Keutzer
A. R. Newton
Class News

- Wednesday:
  - Presentation of research topics in placement, routing, and timing
  - Each group will do a short presentation at the whiteboard:
    - Introduce group members
    - Describe your research problem
    - Name mentors
    - Name resources you will need to do project and where you will get them
    - Describe your approach – methodologically and content
Today - Imagine

- You have to plan transportation for a *new* city the size of Chicago
- Many dwellings need direct roads that can’t be used by anyone else
- You can affect the layout of houses and neighborhoods but the architects and planners will complain
- And … you’re told that the time along any path can’t be longer than a fixed amount
- What are some of your considerations?
What are some of your considerations?

- How many levels do my roads need to go? Remember: Higher is more expensive.

- How do I avoid congestion?

- What basic structure do I want for my roads?
  - Manhattan?
  - Chicago?
  - Boston?

- Automated route tools have to solve problems of comparable complexity on every leading edge chip.
RTL Design Flow

Library

HDL

RTL Synthesis

Manual Design

Module Generators

netlist

logic optimization

netlist

physical design

layout

Courtesy K. Keutzer et al. UCB

Kahng/Keutzer/Newton
Physical Design Flow

Input
- Read Netlist

Floorplanning
- Floorplanning

Placement
- Initial Placement
- Routing Region Definition
- Global Routing
- Cost Estimation
- Routing Region Ordering
- Detailed Routing
- Cost Estimation

Routing

Output
- Compaction/clean-up
- Write Layout Database

Placement Improvement
- Placement Improvement

Routing Improvement

ECE 260B – CSE 241A /UCB EECS 244 6

Courtesy K. Keutzer et al. UCB
Kahng/Keutzer/Newton
Routing Applications

- Cell-based
- Mixed Cell and Block
- Block-based
Cell based placement leaves us with …

(a) Global placement with 1 region  
(b) Global placement with 4 region  
(c) Final placements

D. Pan – U of Texas
Routing Algorithms

Hard to tackle high-level issues like congestion and wire-planning and low level details of pin-connection at the same time

- **Global routing**
  - Identify routing resources to be used
  - Identify layers (and tracks) to be used
  - Assign particular nets to these resources
  - Also used in floorplanning and placement

- **Detail routing**
  - Actually define pin-to-pin connections
  - Must understand most or all design rules
  - May use a compactor to optimize result
  - Necessary in all applications
Basic Rules of Routing - 1

- Wiring/routing performed in layers – 5-9 (-11), typically only in “Manhattan” N/S E/W directions
  - E.g. layer 1 – N/S
  - Layer 2 – E/W

- A segment cannot cross another segment on the same wiring layer or …?

- Wire segments can cross wires on other layers

- Power and ground may have their own layers

Photo courtesy:
Jan M. Rabaey
Anantha Chandrakasan
Borivoje Nikolic
Basic Rules of Routing – Part 2

- Routing can be on a fixed grid –

- Case 1: Detailed routing only in channels
  - Wiring can only go over a row of cells when there is a free track – can be inserted with a “feedthrough”
  - Design may use of metal-1, metal-2
  - Cells *must* bring signals (i.e. inputs, outputs) out to the channel through “ports” or “pins”
Basic Rules of Routing – Part 3

Routing can be on a fixed or gridless (aka area routing)

Case 1: Detailed routing over cells
- Wiring can go over cells
- Design of cells must try to minimize obstacles to routing – i.e. minimize use of metal-1, metal-2
- Cells *do not* need to bring signals (i.e. inputs, outputs) out to the channel – the route will come to them
Taxonomy of VLSI Routers

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

Courtesy K. Keutzer et al. UCB
Global Routing

- Objectives
  - Minimize wire length
  - Balance congestion
  - Timing driven \(\rightarrow\) maintain timing constraints
  - Noise driven \(\rightarrow\) minimize cross-coupled capacitance
  - Keep buses together
Global Routing Formulation

Given
(i) Placement of blocks/cells
(ii) channel capacities

Determine
Routing topology of each net

Optimize
(i) max # nets routed
(ii) min routing area
(iii) min total wirelength

Classic terminology: In general cell design or standard cell design, we are able to move blocks or cell rows, so we can guarantee connections of all the nets (“variable-die” + channel routers).

Classic terminology: In gate-array design, exceeding channel capacity is not allowed (“fixed-die” + area routers).

Since Tangent’s Tancell (~1986), and > 3LM processes, we use largely use area routers for cell-based layout.
Global Routing

- Provide guidance to detailed routing (why?)
- Objective function is application-dependent
Graph Models for Global Routing

- Global routing problem is a graph problem
- Model routing regions, their adjacencies and capacities as graph vertices, edges and weights
- Choice of model depends on algorithm
- Grid graph model
  - Grid graph represents layout as a $h \times w$ array, vertices are layout cells, edges capture cell adjacencies, zero-capacity edges represent blocked cells
- Channel intersection graph model for block-based design
Channel Intersection Graph

- Edges are channels, vertices are channel intersections (CI), v1 and v2 are adjacent if there exists a channel between (CI₁ and CI₂). Graph can be extended to include pins.

Courtesy K. Keutzer et al. UCB
Global Routing Approaches

- Can route nets:
  - Sequentially, e.g. one at a time – e.g. using maze routing but just assign to a routing resource
  - Concurrently, e.g. simultaneously all nets

- Sequential approaches
  - Sensitive to ordering
  - Usually sequenced by
    - Criticality
    - Number of terminals

- Concurrent approaches
  - Computationally hard
  - Hierarchical methods used
Taxonomy of VLSI Routers

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

Courtesy K. Keutzer et al. UCB
Clock Routing Structures

Balanced Tree

H-Tree

Or in low performance ASIC designs a clock may be “just another net”
Clock Routing

- Multiple Clock Domains

Trunk or Grid

Clock Mesh
Power Routing

- Power Mesh
- Power Ring
- Star Routing

In cases where an entire layer is devoted to power, both N/S, E/W directions may be used.
Taxonomy of VLSI Routers

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

X \{ gridded, gridless \}

Courtesy K. Keutzer et al. UCB
One Layer Routing: General River-Routing

- For clock, power, ground still may need to solve single-layer routing
- Two possible paths per net along boundary
  - Path = alternating sequence of horizontal and vertical segments connecting two terminals of a net
- Consider starting terminals and ending terminals
- Assume every path counter-clockwise around boundary
One Layer Routing: General River-Routing

- Create circular list of all terminals ordered counterclockwise according to position on boundary
One Layer Routing: General River-Routing

- Boundary-packed solution
- Flip corners to minimize wire length

![Diagram of general river-routing network with labeled nodes and wires]

Courtesy K. Keutzer et al. UCB
Taxonomy of VLSI Routers

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

Courtesy K. Keutzer et al. UCB
Routing Layer Models

1 layer

2 layers

3 layers

VH model

HV model

VHV model

HVH model

Layer 1

Layer 2

Layer 3

Via
Channel vs. Switchbox

**Channel**
- Channel may have exits at left and right sides, but exit positions are not fixed.
- We may map exits to either lower or upper edge of a channel.

**Switchbox**
- Terminal positions on all four sides of a switchbox are fixed.
- Two dimensional problem.
- Switchbox routing is more difficult.

---

![Diagram showing channel and switchbox comparisons](image-url)
Channel Routing Problem

**Input:** Pins on the lower and upper edge

**Output:** Connection of each net

**Constraints (Assumption)**
- (i) grid structure
- (ii) two routing layers. One for horizontal wires, the other for vertical wires
- (iii) vias for connecting wires in two layers

**Minimize:**
- (i) # tracks (channel height)
- (ii) total wire length
- (iii) # vias
Channel Routing

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

Diagram:
- Tracks
- Pins
- Trunk
- Branch
- Via
- Net
- Channel
- End
Channel/Switch Box Routing Algorithms

- Graph theory based algorithm
  Yoshimura and Kuh

- Greedy algorithm
  Rivest and Fiduccia

- Maze routing and its variations
  Lee, Robin, Soukup, Ohtsuki

- Hierarchical wire routing
  Burstein and Pelavin

Channel routing

Channel / switchbox and general area routing
Greedy Channel Router


- A simple linear time algorithm
- Guarantee the completion of all the nets (may extend to right-hand side of the channel)
- Produce both restricted doglegs and unrestricted doglegs
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
Overview of Greedy Router

Left-to-right, Column-by-column scan

\[
c := 0; \\
\text{while (not done) do} \\
\quad \text{begin} \\
\qquad c := c + 1; \\
\qquad \text{complete wiring at column } c; \\
\quad \text{end;}
\]

In general, at a point in a net may be

1. empty (net 5)
2. unsplit (nets 1, 4)
3. split (net 3)
4. completed (net 2)
Parameters to Greedy Router

• Initial-channel-width icw
• Minimum-jog-length mjl (Why?)
• Steady-net-constant snc
• Usually start icw as d. the density
• Mjl controls the number of vias, use a large mjl for fewer vias
• Snc also controls # of vias (typical value=10 Why?)
Operations at Each Column

At each column, the greedy router tries to maximize the utility of the wiring produced:

A: Make minimal feasible top/bottom connections;
B: Collapse (connect) split nets;
C: Move split nets closer to one another;
D: Raise rising nets/lower falling nets, i.e. bring nets closer to destination terminal;
E: Widen channel when necessary;
F: Extend to next column;
(A) Make Minimal Feasible top/bottom Connections
(B) Collapse/(Connect) Split Nets

G                  G*

H                  H*

I                  I*

J                  J*
(C) Move Split Nets Closer
(D) Rising/Falling
(E) Insert New Track
(F) Extend to Next Column
Greedy Routing Example
Comments on Greedy Router (Rivest & Fiduccia 1982)

- Always succeeds (even if cyclic conflict is present);
- Allows unrestricted doglegs;
- Allows a net to occupy more than 1 track at a given column;
- May use a few columns off the edge;

Net 1 occupies two track at this column; it also "wraps-around"

Unrestricted dogleg

A column off the edge
What's wrong with channel routing?

- We need to be able to route over cells and get to pins
- We want to minimize routing area

The X Architecture: not your father's diagonal wiring

Steven L. Teig
Simplex Solutions, Inc.
Sunnyvale, CA
steve@simplex.com
Two-Terminal Routing: Maze Routing

- Maze routing finds a path between source (s) and target (t) in a planar graph on a fixed grid.
- Grid graph model is used to represent block placement.
- Available routing areas are unblocked vertices, obstacles are blocked vertices.
- Finds an optimal path.

Courtesy K. Keutzer et al. UCB
Maze Routing

- Point to point routing of nets
- Route from source to sink
- Basic idea = wave propagation (Lee, 1961)
  - Breadth-first search + back-tracing after finding shortest path
  - Guarantees to find the shortest path
- Objective = route all nets according to some cost function that minimizes congestion, route length, coupling, etc.

Maze Routing

E. Young
Basic Idea

- A Breadth-First Search (BFS) of the grid graph.
- Always find the shortest path possible.
- Consists of two phases:
  - Wave propagation
  - Retrace
An Illustration

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

E. Young
Wave Propagation

- At step $k$, all vertices at Manhattan-distance $k$ from $S$ are labeled with $k$.
- A Propagation List (FIFO) is used to keep track of the vertices to be considered next.
Retrace

- Trace back the actual route.
- Starting from $T$.
- At vertex with $k$, go to any vertex with label $k-1$.

Final labeling

E. Young
How many grids visited using Lee’s algorithm?
Time and Space Complexity

- For a grid structure of size $w \times h$:
  - Time per net = $O(wh)$
  - Space = $O(wh \log wh)$ (O(log $wh$) bits are needed to store each label.)

- For a $4000 \times 4000$ grid structure:
  - 24 bits per label
  - Total 48 Mbytes of memory!

- Generally, time and space complexity per net $O(\text{height} \times \text{width})$

E. Young
Lee’s Algorithm

- **Strengths:**
  - Guarantee to find a connection if one exists.
  - Guarantee shortest path.

- **Weaknesses:**
  - Large memory requirements.
  - Slow

- **Applications:**
  - Global routing and detailed routing
Improvement to Lee’s Algorithm

- Improvement on memory:
  - Aker’s Coding Scheme

- Improvement on run time:
  - Starting point selection
  - Double fan-out
  - Framing
  - Hadlock’s Algorithm
  - Soukup’s Algorithm
Limiting the Search Region

- Since majority of nets are routed within the bounding box defined by S and T, can limit points searched by maze router to those within bounding box
  - Allows maze router to finish sooner with little or no negative impact on final routing cost
  - Router will not consider points that are unlikely to be on the route path

S denotes the source point
T is the sink point

Bounding box of S and T

Normally, the search region is restricted to the bounding box + X

In this example, X = 2

The points outside of blue area are not considered by the maze router
Problems With Maze Routing

Slow: for each net, we have to search $N \times N$ grid
Memory: total layout grid needs to be kept $N \times N$

**Improvements**

- Simple speed-up
- Minimum detour algorithm (Hadlock, 1977)
- Fast maze algorithm (Soukup, 1978)
  - depth-first search until obstacle
  - breadth-first at obstacle
  - until target is reached
- Will find a path if it exists, may be suboptimal
- Typical speed-up 10-50x

**Further improvements**

- Maze routing infeasible for large chips
- Line search (Mikami & Tabuchi, 1968; Hightower, 1969)
- Pattern routing
Grid Problem with Maze Routing

- What’s the grid?
- Easy if:
  - Global router does layer assignment or
  - TSMC style - sets layer width the same for layers 1-6 (8)
- Hard if maze router presumed to be multilayer and different wire widths for each layer:
  - M1 105
  - M2 105
  - M3 110
  - M4 140
  - M5 165
  - M6 240
  - M7 360
  - M8 540

Photo courtesy:
Jan M. Rabaey
Anantha Chandrakasan
Borivoje Nikolic
Line-Probe Algorithm


Mikami+Tabuchi’s algorithm
- Generate search lines from both source and target (level-0 lines)
- From every point on the level-i search lines, generate perpendicular level-(i+1) search lines
- Proceed until a search line from the source meets a search line from a target
- Will find the path if it exists, but not guaranteed to find the shortest path

Time and space complexity: $O(L)$, where $L$ is the number of line segments
Line-Probe Summary

- Fast, handles large nets / distances / designs
- Routing may be incomplete
Problems with Sequential Routing Algorithms

Net ordering

- Must route net by net, but difficult to determine best net ordering!
- Difficult to predict/avoid congestion

What can be done

- Use other routers
  - Channel/switchbox routers
  - Hierarchical routers
- Rip-up and reroute
A Potpourri of Other Routing Issues

- More than any other topic that we will study, routing and compaction face a large number of issues arising from idiosyncrasies of semiconductor processes.
Detailed Routing Issues

- Routing completion
- Width and spacing rule
  - Minimum width and spacing
  - Variable width and spacing
    - Connection
    - Net
    - Class of nets
  - Tapering

![Diagram](image-url)
Detailed Routing Issues

- **Width and spacing rule**

  - Minimum spacing
  - Width-based Spacing

  ![Diagram showing detailed routing issues with width and spacing rules.](image)
Detailed Routing Issues

- Noise-driven – i.e. noise minimization driven

Noisy region

Extra space

Grounded Shields

Quiet region

Spacing

Shielding

Segregation
Detailed Routing Issues

- Shielding
  - Same-layer shielding
  - Adjacent-layer shielding
Detailed Routing Issues

- Shielding
  - Bus shielding
  - Bus interleaving

Bus Shielding
Detailed Routing Issues

- Differential pair routing
- Balanced length or capacitance

Differential
Balanced length
Detailed Routing Issues

Bus Routing
Current Status

- Routing 100,000 nets is routine
- Fundamental problems
  - Timing driven routing
- Tactical problems
  - Increasing complexity/restrictions of design rules
  - Routing even one net DRC (design rule checked) correctly is a challenge (Jason Cong paper)
- But... routing is not a "showstopper" in current VLSI designs
  - Routing is not a bottleneck, and therefore there's at least one new start-up every year

Focus of start-ups is:

- Chip-level routing for assembly
- Special purpose niche routing – e.g. die to bonding pads to package
Routing Summary

Routing requires an especially large toolbox of techniques
Every router mentioned here is used
Summary

- Large toolset → many algorithms
- Fortunately, most of the algorithms are simple
- Be sure you’re familiar with:
  - Maze Routing – Lee
  - Line routing - Hightower
  - Single-layer switchbox routing
  - Greedy channel routing - Rivest
- Be familiar with the flow and relationship between routers
Extras

- Aker’s Coding Scheme for Maze Routing
- What’s there to do in channel routing?
- Pattern-based routing
- Steiner Trees
- Concurrent global routing
- Yoshimura-Kuh
- Compaction
- Other Physical Issues
Aker’s Coding Scheme

- For the Lee’s algorithm, labels are needed during the retrace phase.

- But there are only two possible labels for the neighbors of each vertex labeled $i$, which are, $i-1$ and $i+1$.

- Want a labeling scheme such that each label has its preceding label different from its succeeding label.
Extras

- What’s there to do in channel routing?
- Pattern-based routing
- Steiner Trees
- Concurrent global routing
- Yoshimura-Kuh
- Compaction
- Other Physical Issues
Trivial Channel Routing

- Assign every net its own track

Dehon - Caltech
Trivial Channel Routing

- Assign every net its own track
  - Channel width > N (single output functions)
  - Chip bisection $\propto N \rightarrow \text{chip area } N^2$

Dehon- Caltech
Sharing Tracks

- Want to Minimize tracks used
- Trick is to share tracks

Dehon - Caltech
Not that Easy

- With Two sides
  - Even assigning one track/signal may not be enough

Dehon - Caltech
Not that Easy

- With Two sides
  - Even assigning one track/signal may not be enough

Bad assignment
Overlap:
A, B
B, C
Not that Easy

- With Two sides
  - Even assigning one track/signal may not be enough

Valid assignment avoids overlap
Not that Easy

- With Two sides
  - Even assigning one track/signal may not be enough

There are vertical constraints on ordering
Extras

- What’s there to do in channel routing?
- Pattern-based routing
- Steiner Trees
- Concurrent global routing
- Yoshimura-Kuh
- Compaction
- Other Physical Issues
Pattern-Based Routing

- Restrict routing of net to certain basic templates
- Basic templates are L-shaped (1 bend) or Z-shaped (2 bends) routes between a source and sink
- Templates allow fast routing of nets since only certain edges and points are considered
In general, maze and line-probe routing are not well-suited to multi-terminal nets

Several attempts made to extend to multi-terminal nets

- Connect one terminal at a time
- Use the entire connected subtrees as sources or targets during expansion
- Ripup/Reroute to improve solution quality (remove a segment and re-connect)

• Results are sub-optimal
• Inherit time and memory cost of maze and line-probe algorithms
Multi-terminal Nets: Different Routing Options

(a) Steiner Tree (14)  
(b) Steiner Tree with Trunk (15)  
(c) Minimum Spanning Tree (16)  
(d) Chain (17)  
(e) Complete Graph (42)

Cost is determined by routing model

Kahng/Keutzer/Newton
Steiner Tree Based Algorithms

- Tree interconnecting a set of points (demand points, D) and some other (intermediate) points (Steiner points, S)
- If S is empty, Steiner Minimum Tree (SMT) equivalent to Minimum Spanning Tree (MST)
- Finding SMT is NP-complete; many good heuristics
  - SMT typically 88% of MST cost; best heuristics are within ½ % of optimal on average
- Underlying Grid Graph defined by intersection of horizontal and vertical lines through demand points (Hanan grid) → Rectilinear SMT and MST problems
- Can modify MST to approximate RMST, e.g., build MST and rectilinearize each edge
Minimum Spanning Tree (Prim’s construction)

Given a weighted graph
Find a spanning tree whose weight is minimum

Prim’s algorithm
start with an arbitrary node s
T ← {s}
while T is not a spanning tree
  find the closest pair x ∈ V - T, y ∈ T
  add (x, y) to T

runs in $O(n^2)$ time
very simple to implement
always gives a tree of minimum cost
Applying Spanning and Steiner Tree Algorithms

- General cell/block design: channel intersection graphs

- Standard-cell or gate-array design: RSMT or RMST in geometry or grid-graph
Concurrent Global Routing
Can formulate routing problem as *integer programming*, solve simultaneously for all nets

**Given**
(i) Set of Steiner trees for each net  
(ii) Placement of blocks/cells  
(iii) Channel capacities

**Determine**
Select a Steiner tree for each net w/o violating channel capacities

**Optimize**
Min total wirelength
Horizontal Constraint Graph (HCG)

1. Node $v_i$: represents a horizontal interval spanned by net $i$

2. There is an edge between $v_i$ and $v_j$ if horizontal intervals overlap

3. No two nets with a horizontal constraint may be assigned to the same track

4. Maximum clique of HCG establishes lower bound on # of tracks: # tracks $\geq$ size of maximum clique of HCG

Local density at column $C$, $ld(C) =$ # nets split by column $C$

Channel Density $d =$ max $ld(C)$ over all $C$

Each net spans over an interval

Horizontal Constraint Graph (HCG) is an undirected graph with:
  - vertex: net
  - edge: $<n_j, n_k>$, if intervals $I_j, I_k$ intersect

Kahng/Keutzer/Newton
Vertical Constraint Graph (VCG)

1. Node: represents a net

2. Edge \((a_1 \rightarrow a_2)\) exists if at some column:
   - Net \(a_1\) has a terminal on the upper edge
   - Net \(a_2\) has a terminal on the lower edge
   - Edge \(a_1 \rightarrow a_2\) means that Net \(a_1\) must be above Net \(a_2\)

3. Establishes lower bound: \# tracks \(\geq\) longest path in VCG

4. VCG may have a cycle!
Doglegs in Channel Routing

Doglegs may reduce the longest path in VCG

Doglegs break cycles in VCG
Characterizing the Channel Routing Problem

Channel routing problem is completely characterized by the vertical constraint graph and the horizontal constraint graph.

Vertical constraint graph $G_v$

Horizontal constraint graph
**Interval Packing**

**Theorem**  A set of intervals with density d can be packed into d tracks.

Proof: \( I_1 = (a, b) \) \( I_2 = (c, d) \)

Define: \( I_1 < I_2 \) if \( b < c \) or \( I_1 = I_2 \)

1. reflexive: \( I_1 < I_1 \)
2. anti-symmetric: \( I_1 < I_2, I_2 < I_1 \) \( \rightarrow I_1 = I_2 \)
3. transitive: \( I_1 < I_2, I_2 < I_3 \) \( \rightarrow I_1 < I_3 \)

Set of intervals with binary relation < forms a partially ordered set (POSET)

Intervals in a single track \( \rightarrow \) form a chain

Intervals intersecting a common column \( \rightarrow \) form an antichain

Dilworth’s theorem (1950): If the maximum antichain of a POSET is of size d, then the POSET can be partitioned into d chains

\[ \text{Courtesy K. Keutzer et al. UCB} \]
Left-Edge Algorithm for Interval Packing

Repeat
create a new track \( t \)
Repeat
put leftmost feasible interval to \( t \)
until no more feasible interval
until no more interval

Intervals are sorted according to their left endpoints

\[ I_6 \quad I_5 \quad I_3 \quad I_4 \quad I_2 \quad I_1 \]

\[ I_6 \quad I_5 \quad I_4 \quad I_3 \quad I_2 \quad I_1 \]

\( O(n \log n) \) time algorithm. Greedy algorithm works!
Horizontal Constraint Graph (HCG)

- Node $v_i$: represents a horizontal interval spanned by net $I_i$
- There is an edge b/w $v_i$ and $v_j$ if horizontal intervals overlap
- No two nets with a horizontal constraint may be assigned to the same track
- Maximum clique of HCG establishes a lower bound on # of tracks:
  \[ \# \text{ tracks} \geq \text{maximum clique of HCG} \]
Vertical Constraint Graph (VCG)

- **Node**: represents a net
- **Edge** (a₁ → a₂): if at some column, net a₁ has a terminal on the upper edge, net a₂ has a terminal on the lower edge. a₁ → a₂ means that net a₁ has to be above a₂.
- **Establishes a lower bound**: # tracks ≥ longest path in VCG.
- **VCG may have a cycle**!
Doglegs in Channel Routing

- Doglegs may reduce the longest path in VCG

- Doglegs break cycles in VCG
Restricted Dogleg vs unrestricted dogleg
Constraint Graph Based Algorithm: “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.
- Nets can be organized into zones, further simplifying the problem.
The channel routing problem is completely characterized by the vertical constraint graph and the horizontal constraint graph.

Vertical constraint graph $G_v$

Horizontal constraint graph
Zone Representation of Horizontal Segments

Zones are maximum cliques in the horizontal interval graph. Each net must be routed on a different track.
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 \( j \)
Merging of Nets: Example

merge (5,6)
Updating of Zone Representation

Zone

1 2 3 4 5

1 2 3 4 5

1 2 3 4 5

1 2 3 4 5

1 2 3 4 5
Merging of Nets: Example

1. Merge (5, 6)
2. Merge (1, 7)
3. Merge (4, 10)
4. Merge (5, 6, 9)
5. Merge (3, 8)
Final Routing
Routing Examples by Y-K’s Algorithm

Example 3c
- number of tracks = 18
- maximum density = 18

Example 4b
- number of tracks = 17
- maximum density = 17
Routing Examples by Y-K's Algorithm (Cont'd)

Example 5

Deutsch's Difficult example without dogleg

number of tracks = 20
maximum density = 20

number of tracks = 28
maximum density = 19
Yoshimura and Kuh’s Method

Source:

“Efficient Algorithms for Channel Routing”

by T. Yoshimura and E. Kuh


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

- Channel Compaction (one dimension)
Compaction

- **Area Compaction (1.5 or 2 dimension)**
  - May need a lot of constraints to get desired results

[Diagram showing X-compaction and Y-compaction]
Shape-based Routing

- Evolve from maze routing
- Gridless: look at actual size of each shape
- Each shape may have its spacing rule
- Good for designs with multiple width/spacing rules and other complex rules
- Slower than gridded router
Other Physical Issues
Detailed Routing Objectives

- Via selection
  - Via array based on wire size or resistance
  - Rectangular via rotation and offset

Rotate and offset horizontal vias

No rotation for a “cross” via
Detailed Routing Objectives

* Understand complex pin & equivalent pin modeling
Maze Routing

- Initialize priority queue Q, source S, and sink T
- Place S in Q
- Get lowest cost point X from Q, put neighbors of X in Q
- Repeat last step until lowest-cost point X is equal to the sink T
- Rip and reroute nets, i.e., select a number of nets based on a cost function (e.g., congestion of regions through which net travels), then remove the net and reroute it
- Main objective: reduce overflow
  - Edge overflow = 0 if num_nets less than or equal to the capacity
  - Edge overflow = num_nets – capacity if num_nets is greater than capacity
  - Overflow = \( \sum \) (edge overflows) over all edges

Courtesy K. Keutzer et al. UCB
Mikami & Tabuchi’s Algorithm


- Every grid point is an escape point.

D. Pan – University of Texas
Points can be popped from queue according to a multivariable cost function

Cost = function(overflow, coupling, wire length, ...)

Add <distance to sink> to cost function \( \rightarrow \) directed search

- Allows maze router to explore points around the direct path from source to sink first