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

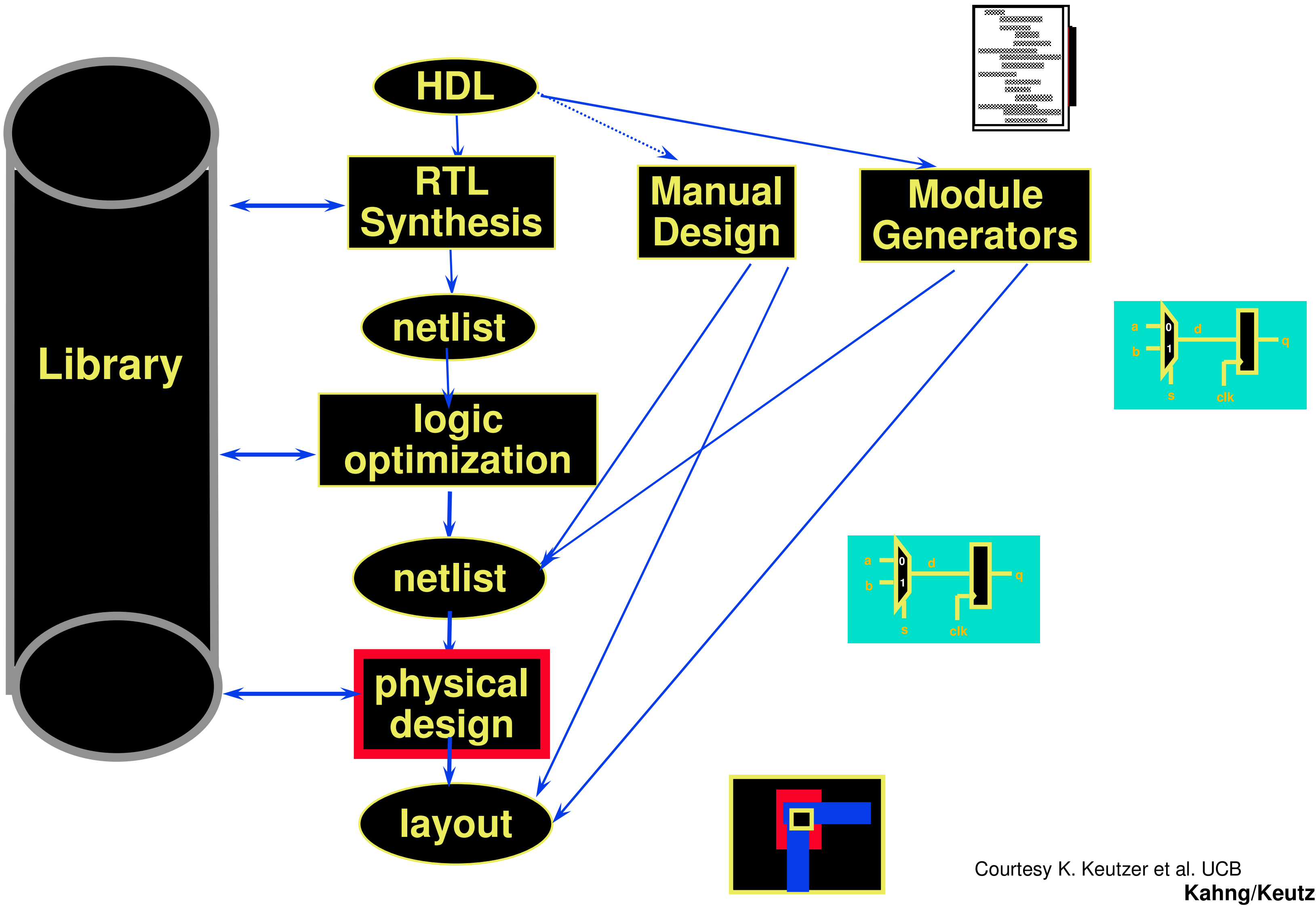
- What are some of your considerations?

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

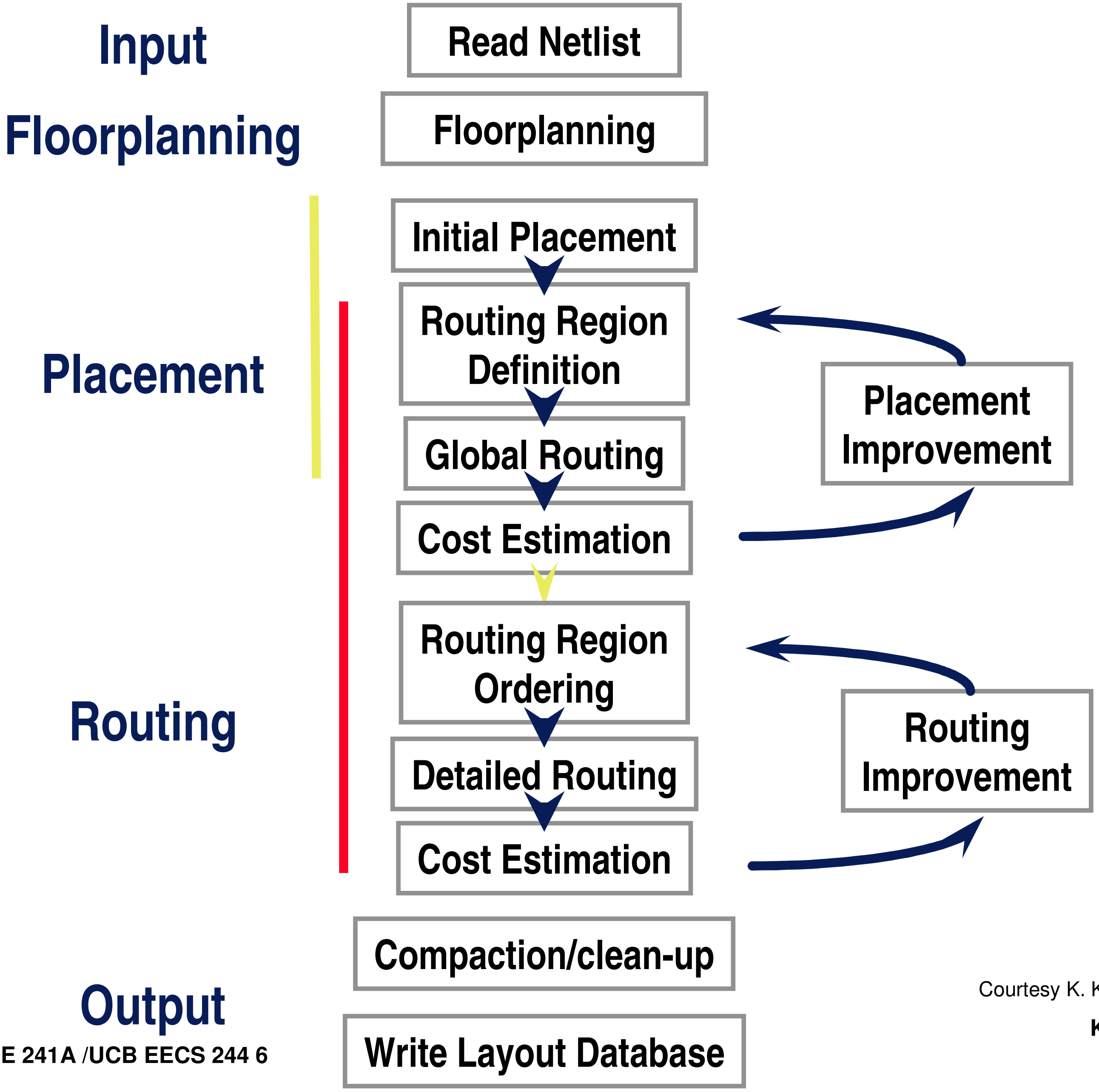
RTL Design Flow



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

Placement Improvement

Routing

Routing Region Ordering

Detailed Routing

Cost Estimation

Routing Improvement

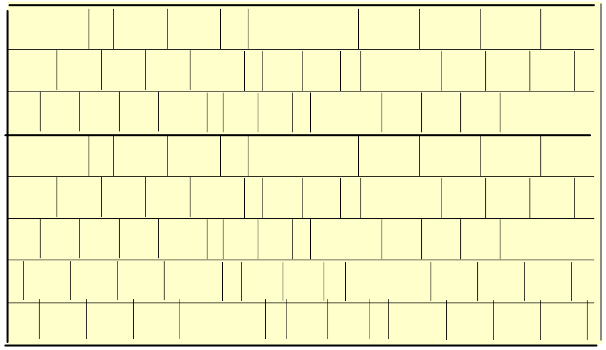
Output

Compaction/clean-up

Write Layout Database

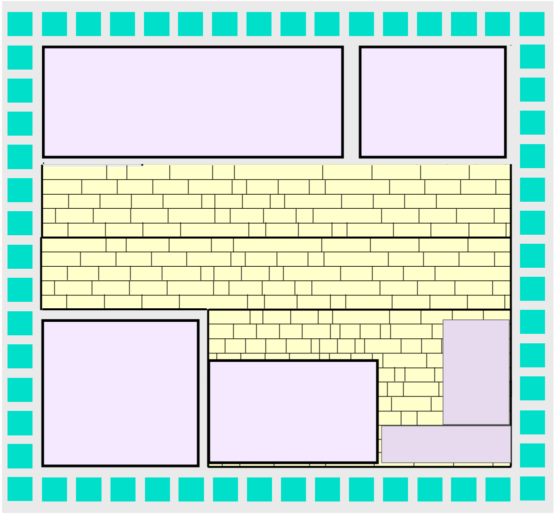
Courtesy K. Keutzer et al. UCB

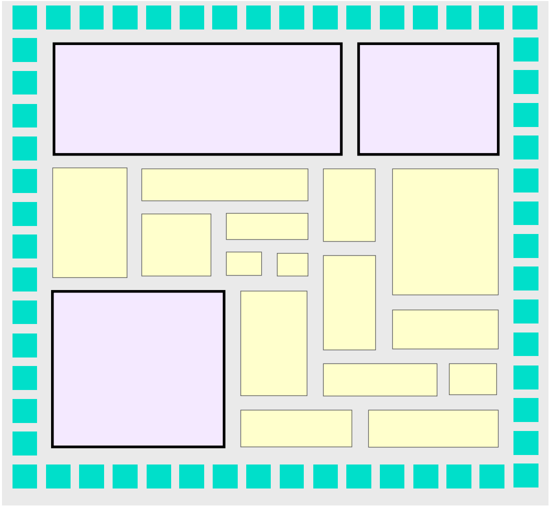
Routing Applications



Cell-based

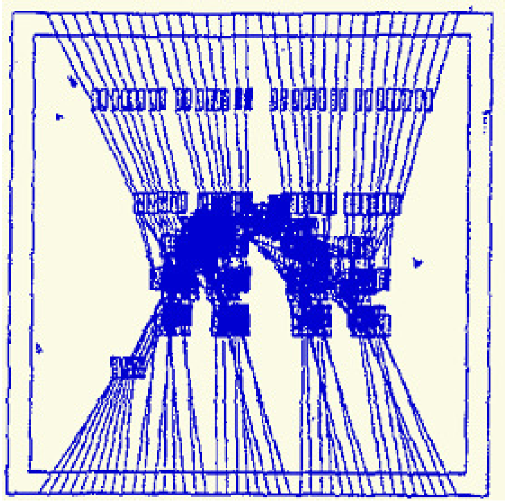
Mixed
Cell and Block



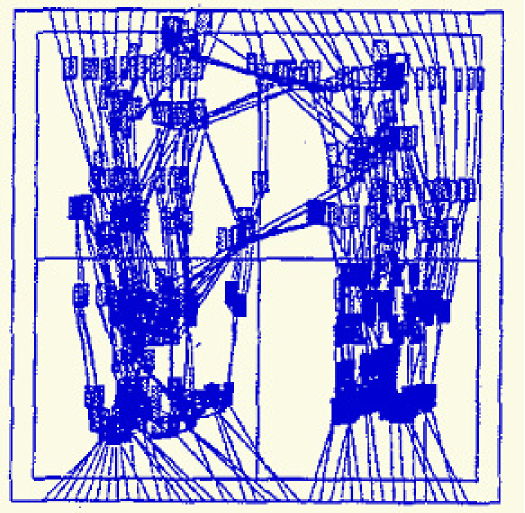


Block-based

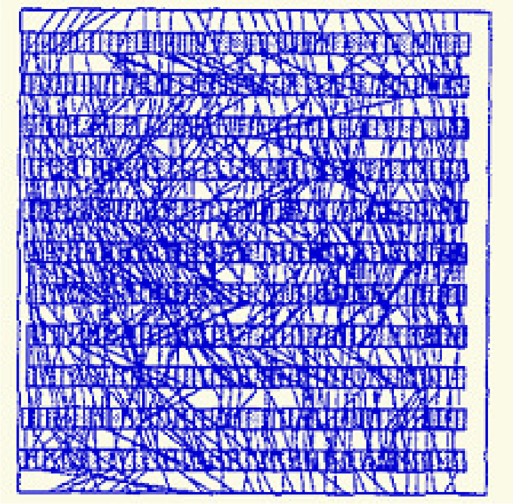
Cell based placement leaves us with …



(a) Global placement with 1 region



(b) Global placement with 4 region



(c) Final placements

D. Pan – U of Texas

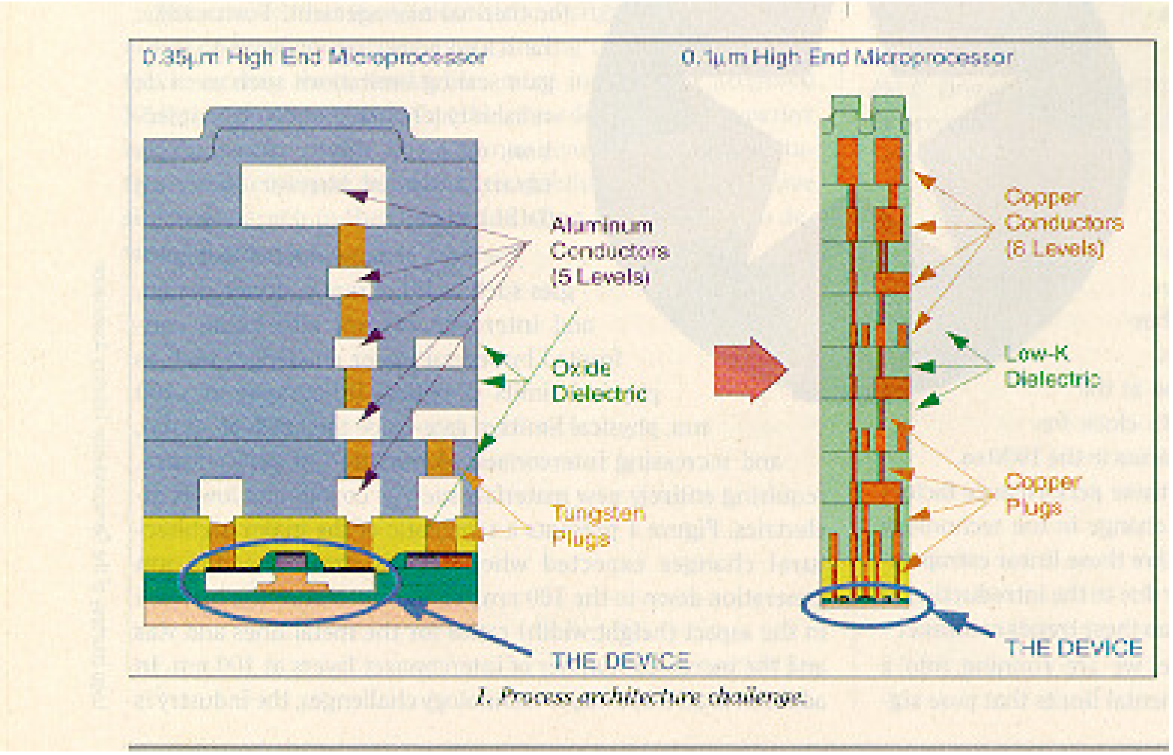
Routing Algorithms

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

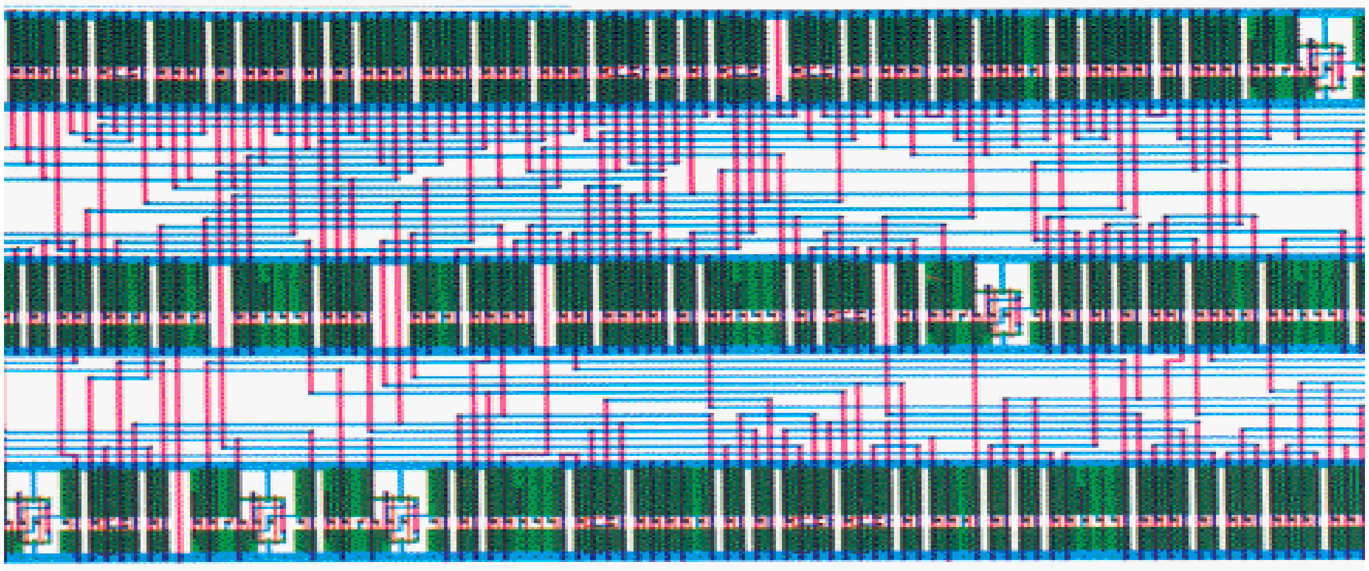

0.35µm High End Microprocessor
Aluminum Conductors (5 Levels)
Oxide Dielectric
Tungsten Plugs
THE DEVICE
0.1µm High End Microprocessor
Copper Conductors (6 Levels)
Low-K Dielectric
Copper Plugs
THE DEVICE

1. Process architecture challenge.

Photo courtesy:
Jan M. Rabaey
Anantha Chandrakasan
Borivoje Nikolic

- 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

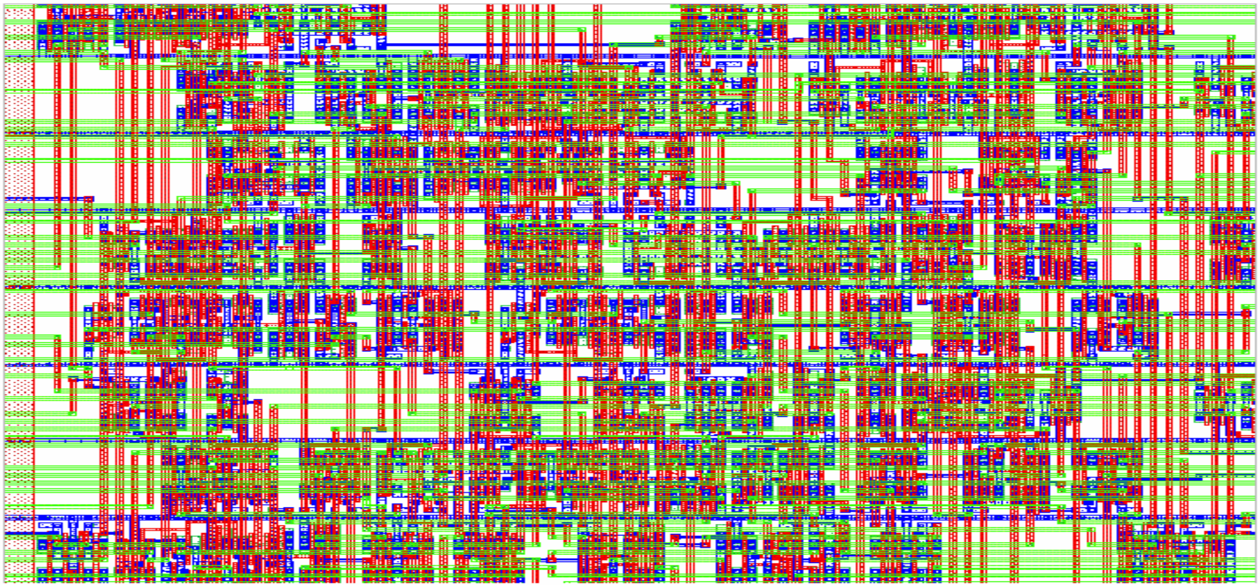
Basic Rules of Routing – Part 2



- 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”

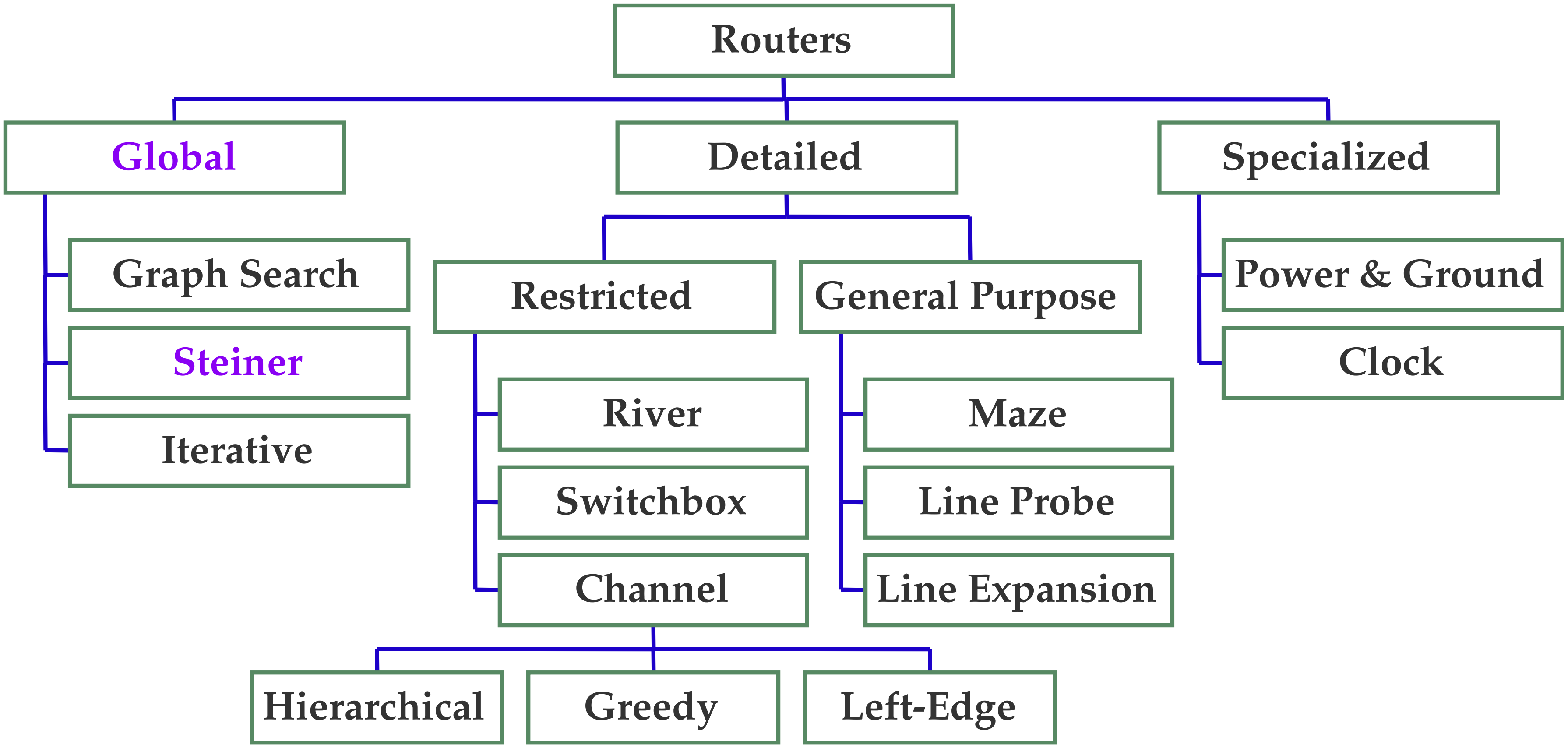
Kahng/Keutzer/Newton

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



Courtesy K. Keutzer et al. UCB

Global Routing

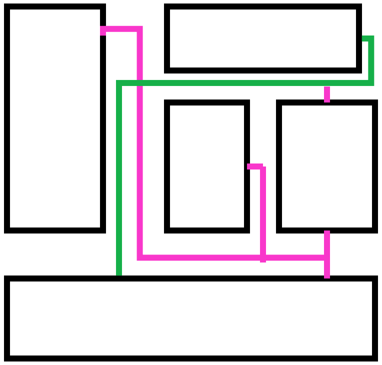
- Objectives
  - Minimize wire length
  - Balance congestion
  - Timing driven → maintain timing constraints
  - Noise driven → minimize cross-coupled capacitance
  - Keep buses together

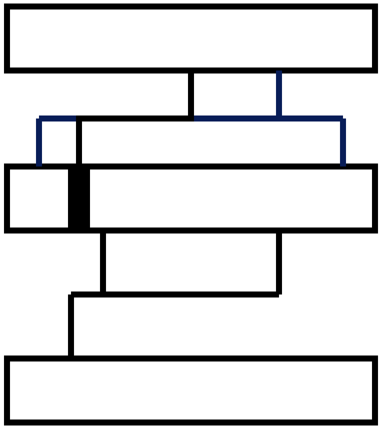
Global Routing Formulation

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).

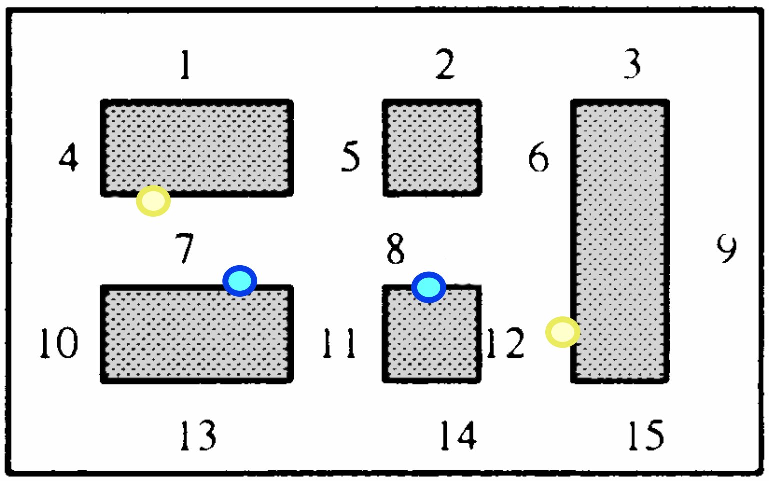
Since Tangent’s Tancell (~1986), and > 3LM processes, we use largely use area routers for cell-based layout

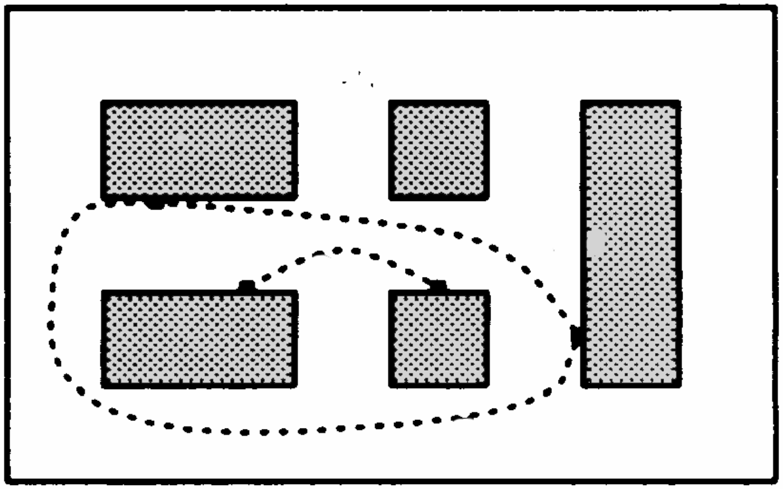
Courtesy K. Keutzer et al. UCB

Kahng/Keutzer/Newton

Global Routing

- Provide guidance to detailed routing (why?)
- Objective function is application-dependent





Courtesy K. Keutzer et al. UCB

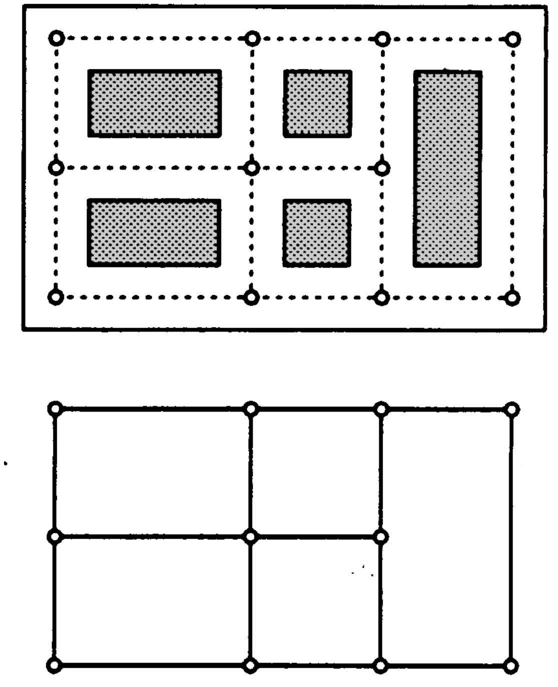
Graph Models for Global Routing

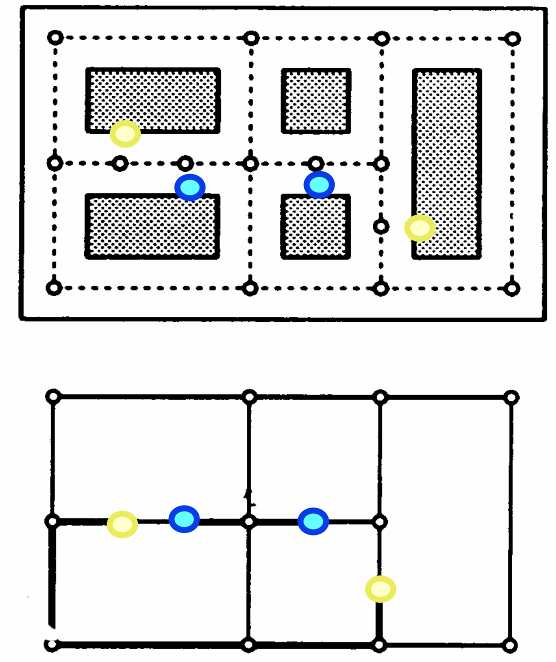
- Global routing problem is a graph problem
- Model routing regions, their adjacencies and capacities as graph vertices, edges and weights
- Choice of model depends on algorithm
- Grid graph model
  - Grid graph represents layout as a $h \times w$ array, vertices are layout cells, edges capture cell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

Courtesy K. Keutzer et al. UCB

Channel Intersection Graph

- Edges are channels, vertices are channel intersections (CI), v1 and v2 are adjacent if there exists a channel between (CI_1 and CI_2). Graph can be extended to include pins.





Courtesy K. Keutzer et al. UCB

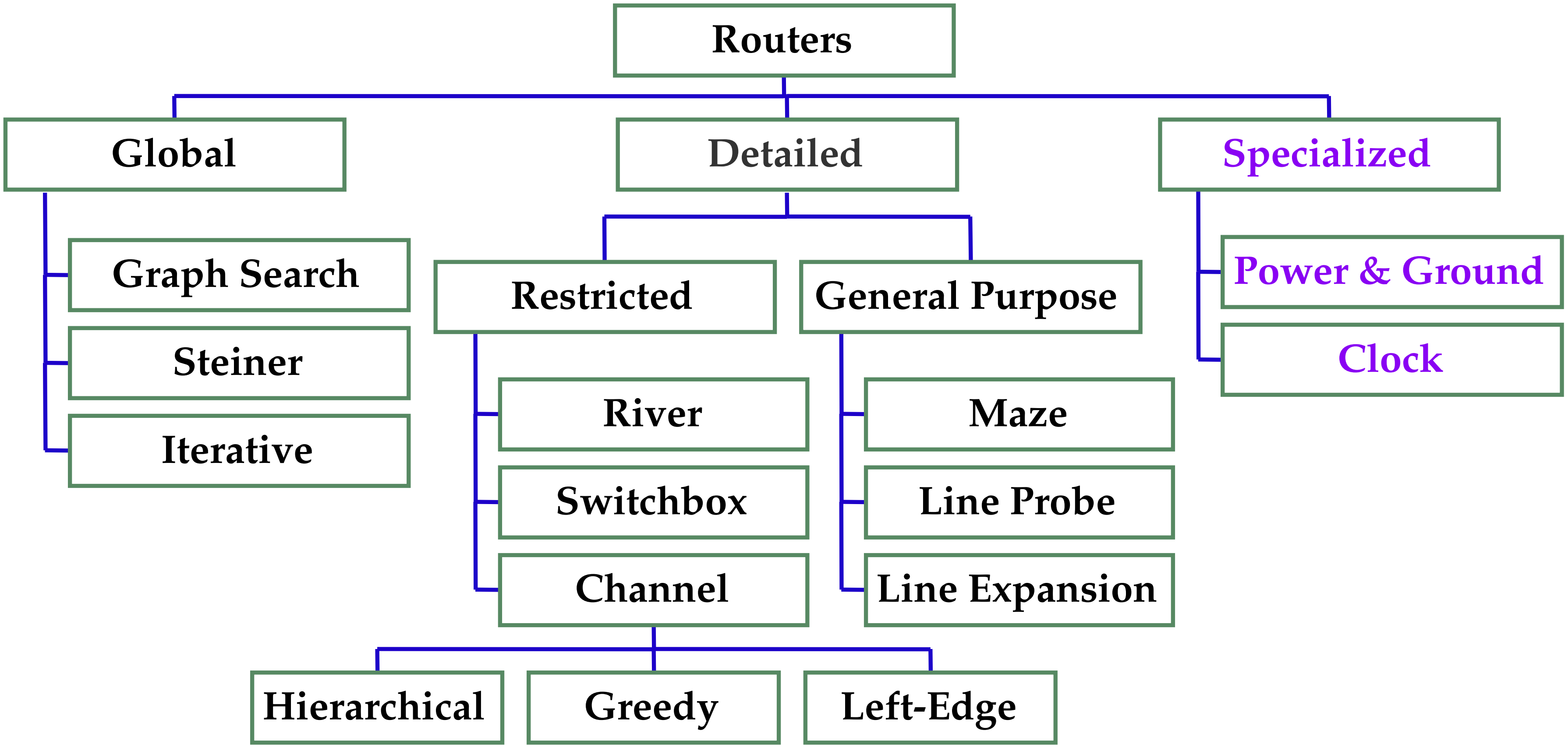
Kahng/Keutzer/Newton

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

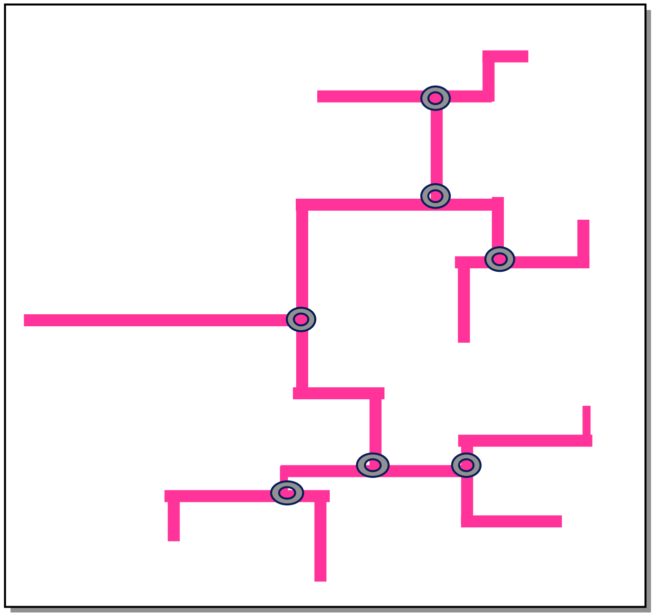
Courtesy K. Keutzer et al. UCB

Taxonomy of VLSI Routers

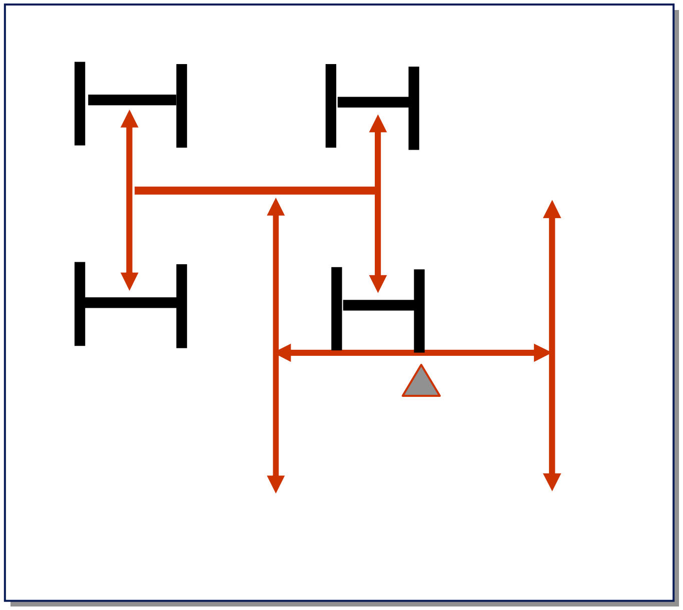


Courtesy K. Keutzer et al. UCB

Clock RoutingStructures



Balanced Tree



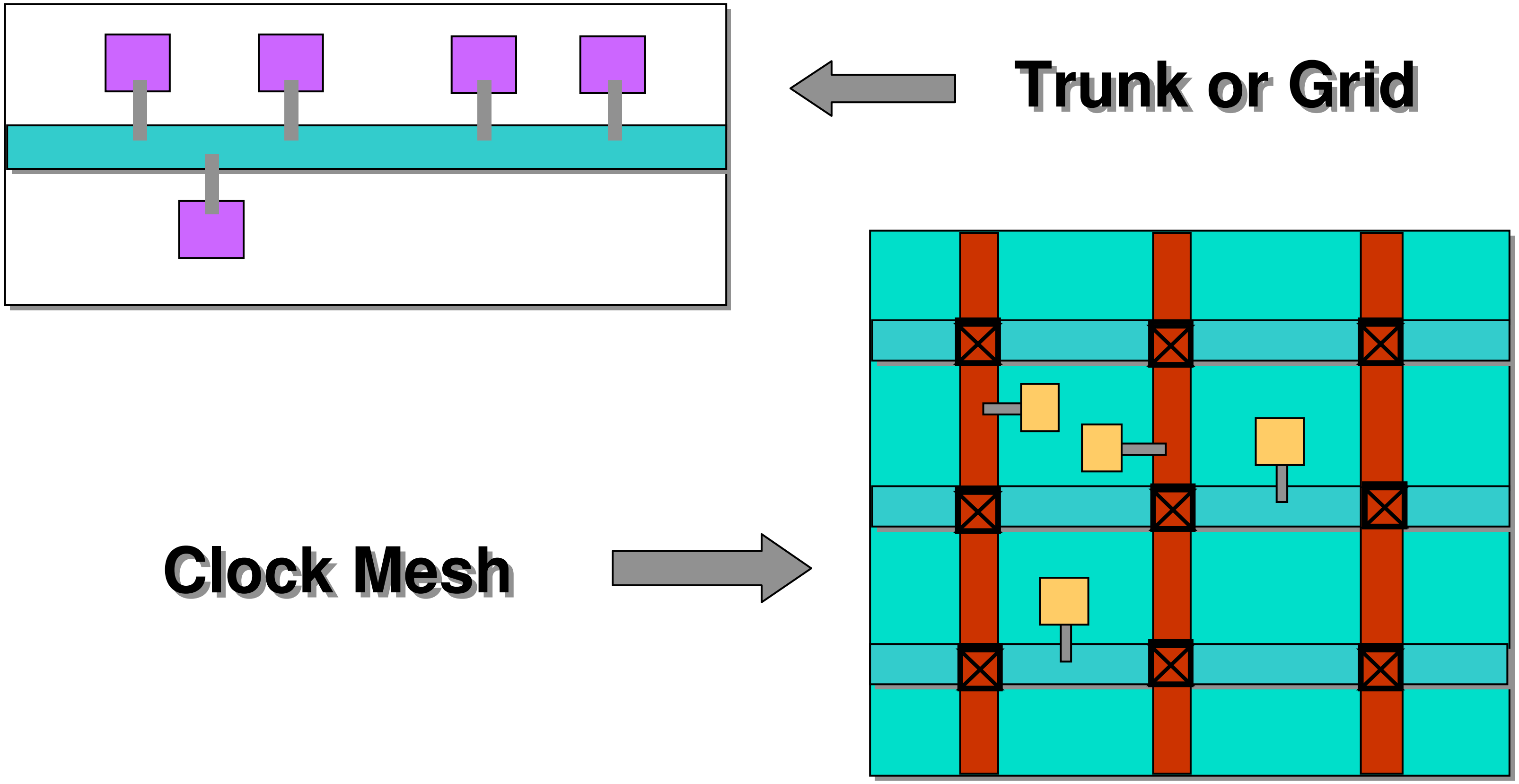
H-Tree

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

Kahng/Keutzer/Newton

Clock Routing

- **Multiple Clock Domains**



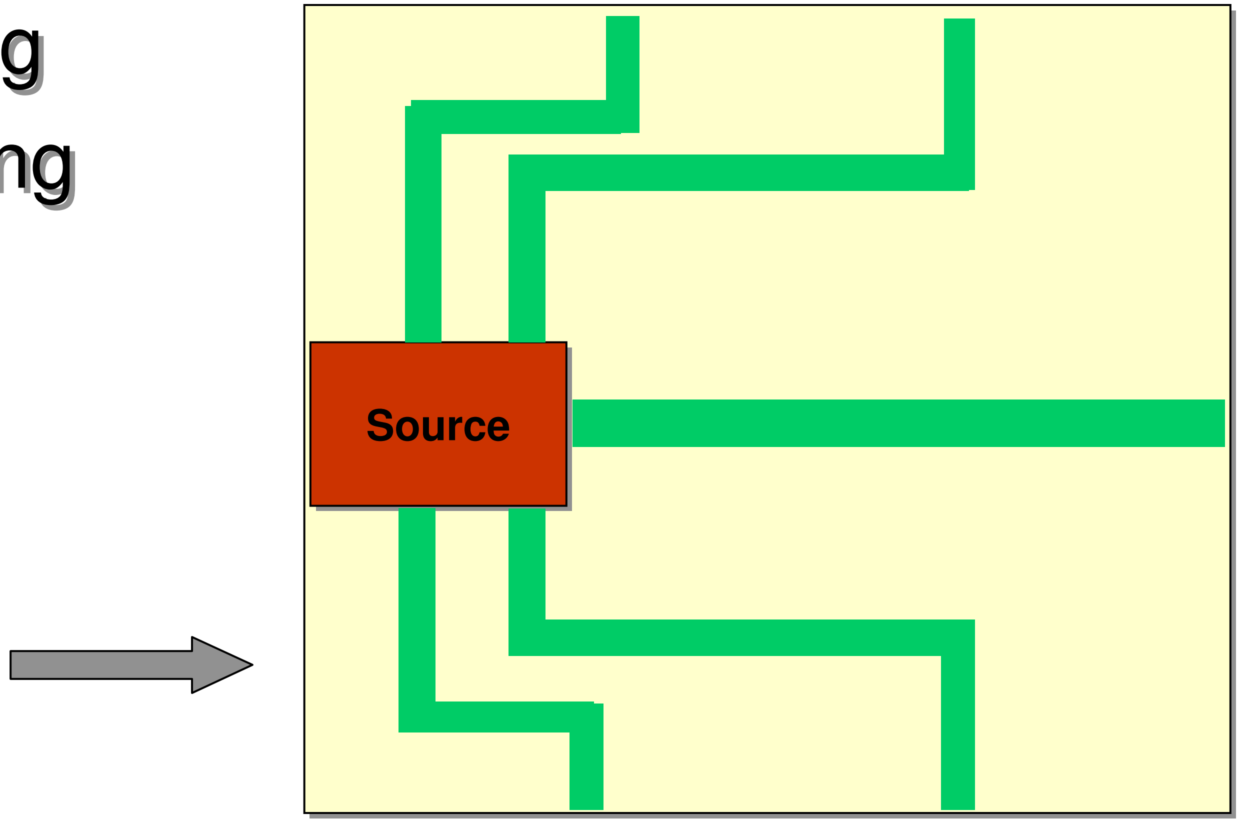
Trunk or Grid

Clock Mesh

Power Routing

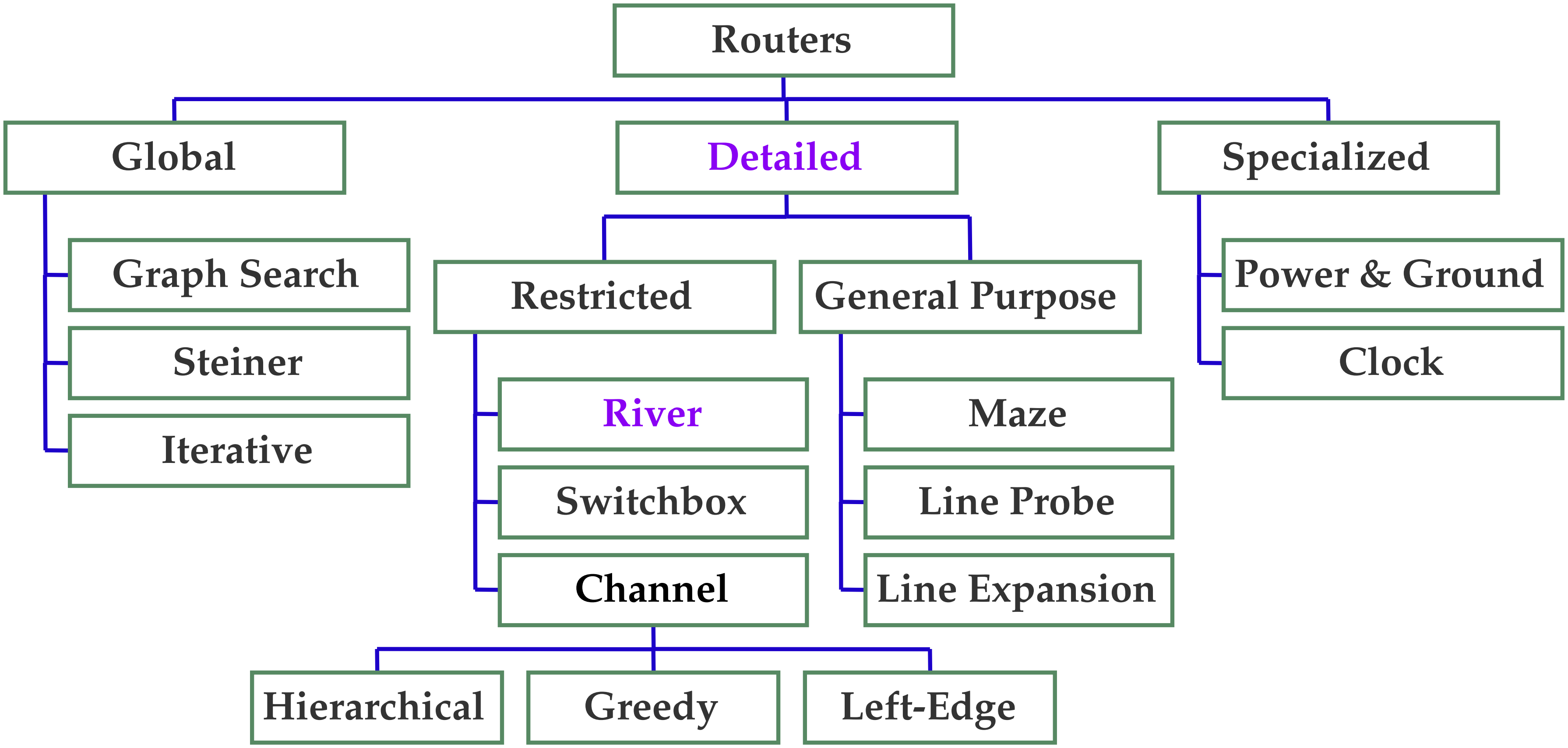
- Power Mesh
- Power Ring
- Star Routing

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



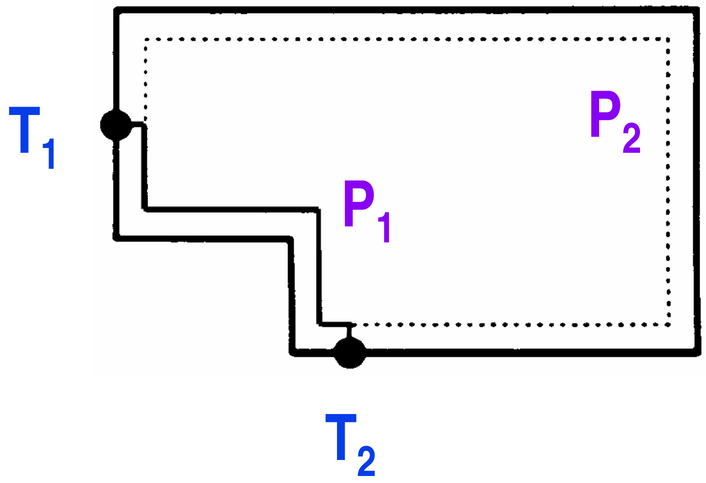
- Routers
  - Global
    - Graph Search
    - Steiner
    - Iterative
  - Detailed
    - Restricted
      - River
      - Switchbox
      - Channel
        - Hierarchical
        - Greedy
        - Left-Edge
    - General Purpose
      - Maze
      - Line Probe
      - Line Expansion
  - Specialized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

X { gridded, gridless}

Courtesy K. Keutzer et al. UCB

One Layer Routing: General River-Routing

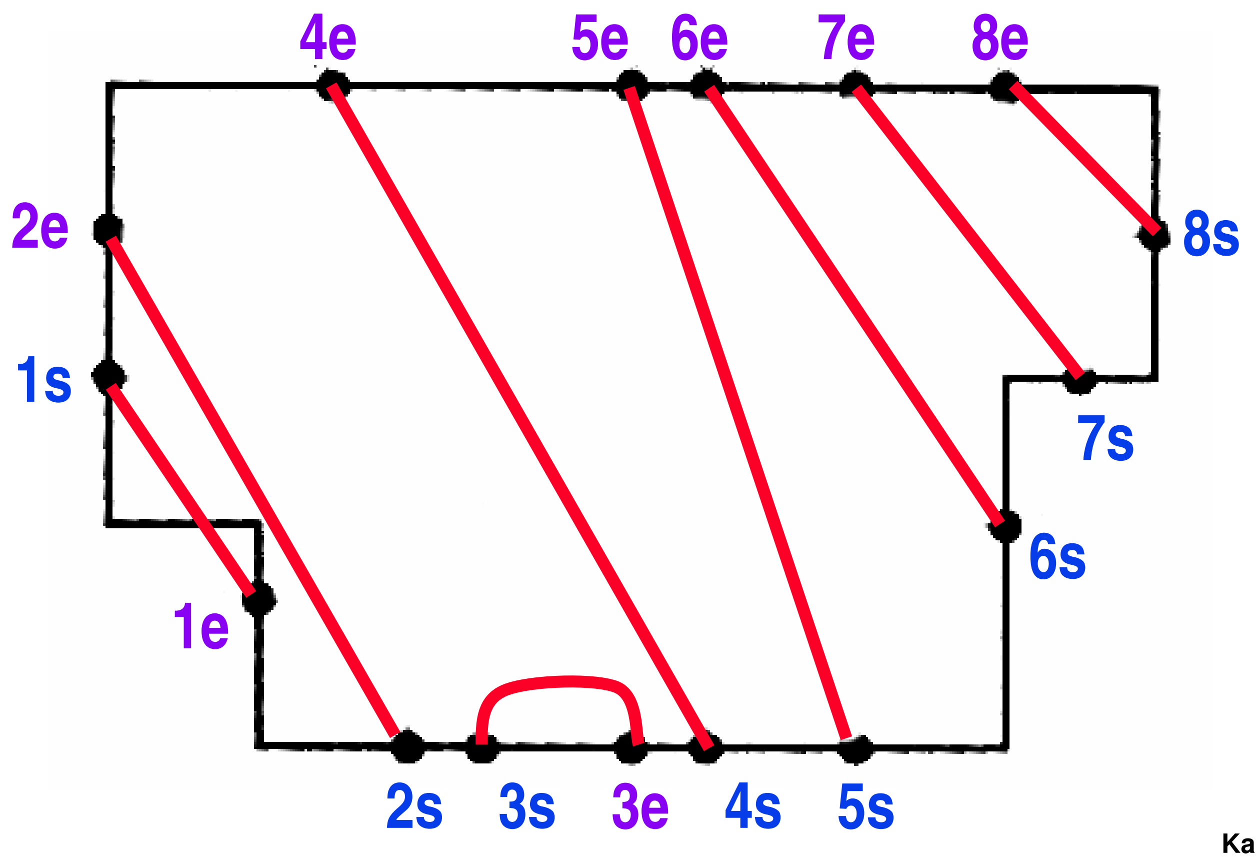
- For clock, power, ground still may need to solve single-layer routing
- Two possible paths per net along boundary
  - Path = alternating sequence of horizontal and vertical segments connecting two terminals of a net
- Consider starting terminals and ending terminals
- Assume every path counter-clockwise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



Courtesy K. Keutzer et al. UCB

One Layer Routing: General River-Routing

- Create circular list of all terminals ordered counterclockwise according to position on boundary

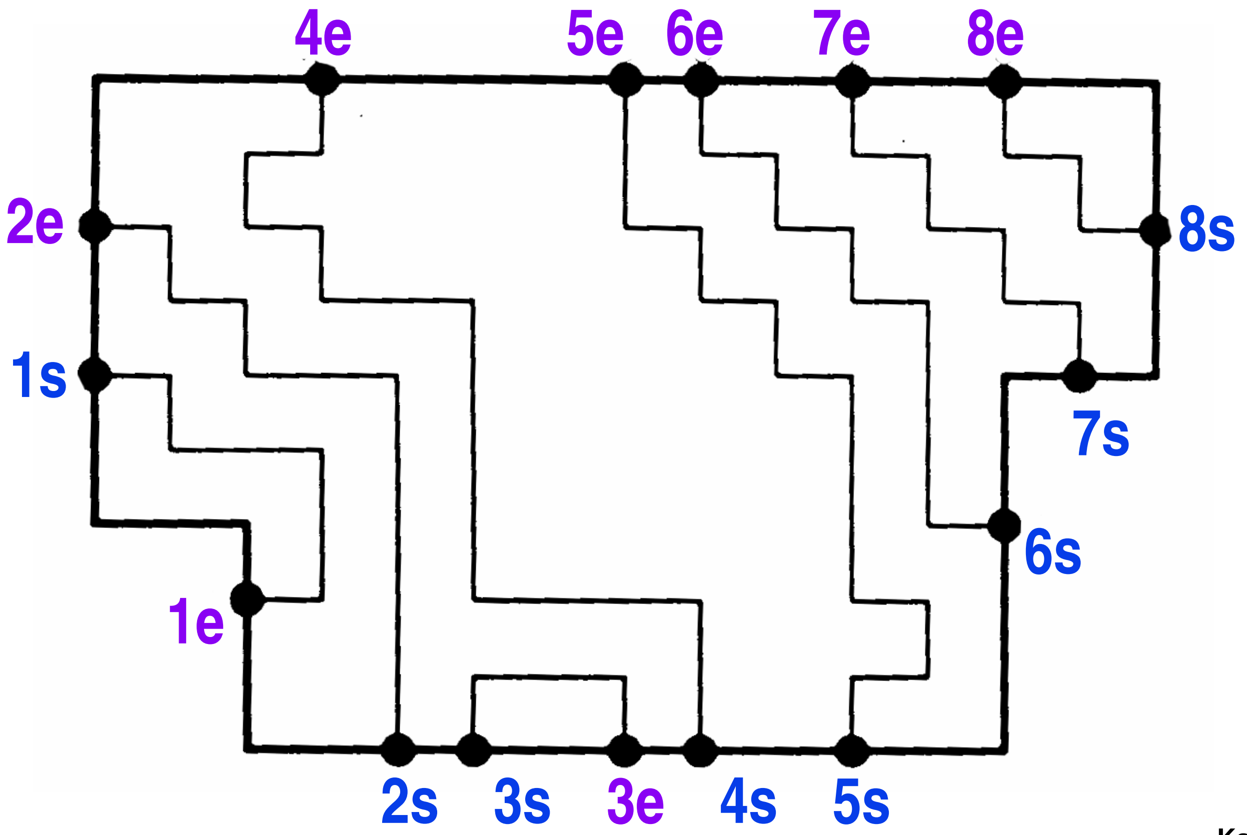


Courtesy K. Keutzer et al. UCB

Kahng/Keutzer/Newton

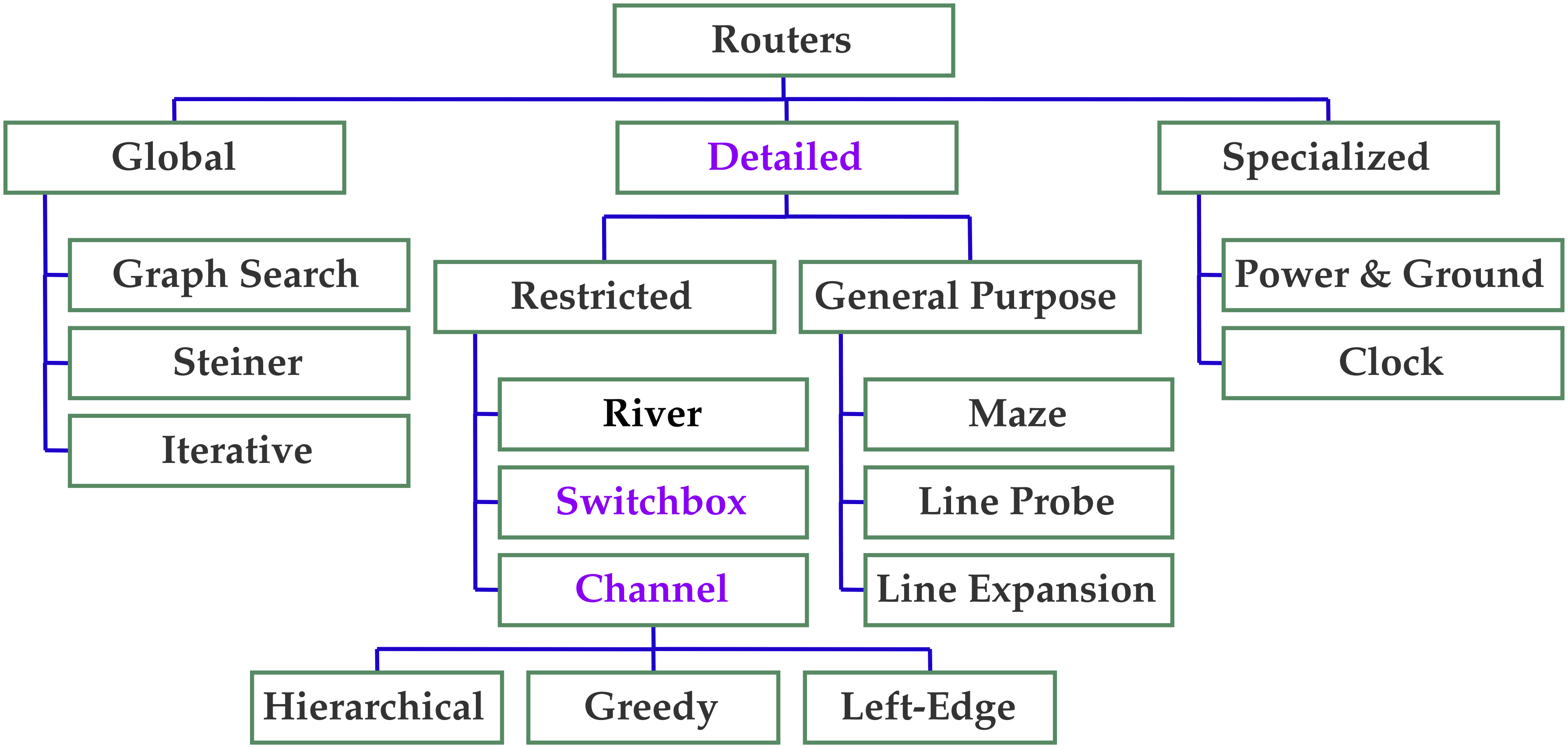
One Layer Routing: General River-Routing

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



Courtesy K. Keutzer et al. UCB

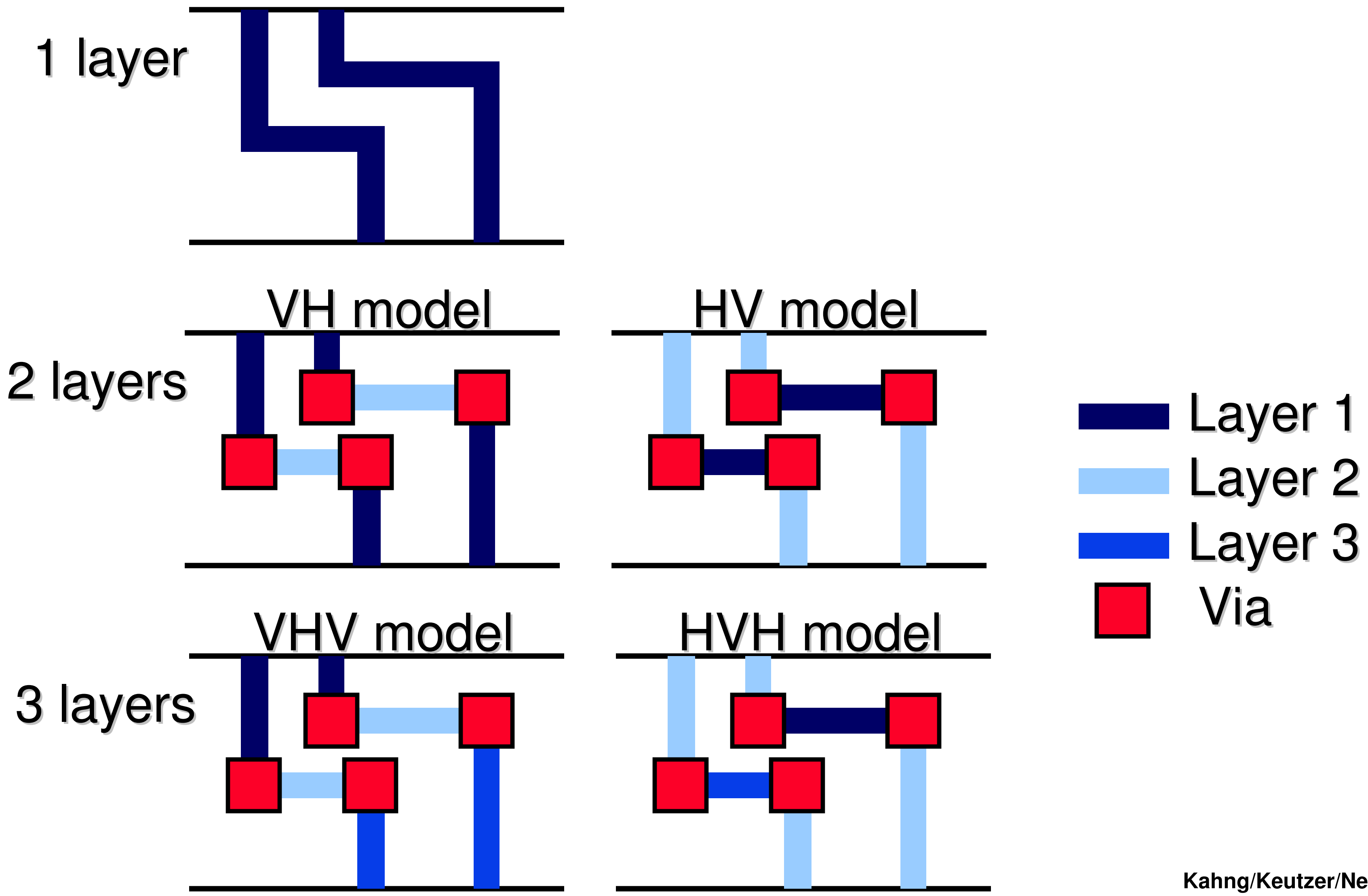
Taxonomy of VLSI Routers



Kahng/Keutzer/Newton

Courtesy K. Keutzer et al. UCB

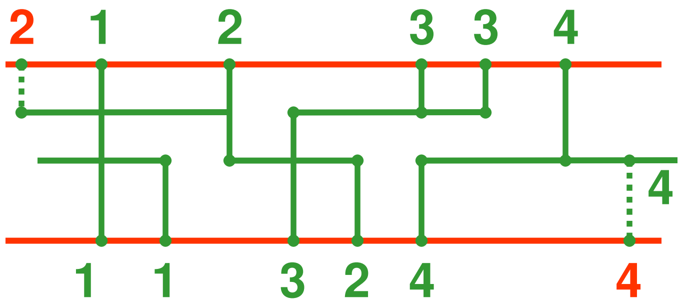
Routing Layer Models



Channel vs. Switchbox

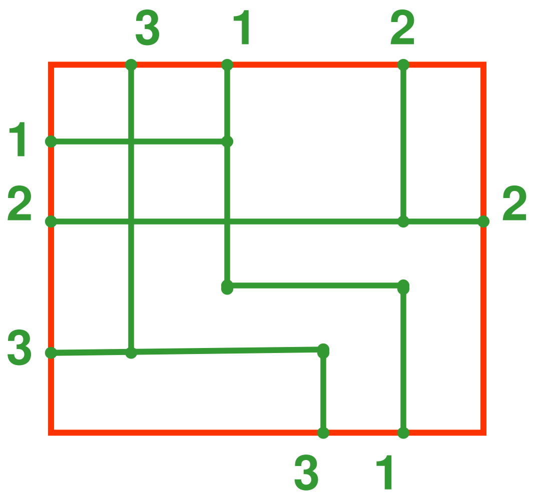
Channel

- 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
- One dimensional problem



Switch box

- Terminal positions on all four sides of a switchbox are fixed
- Two dimensional problem



Switchbox routing is more difficult

Courtesy K. Keutzer et al. UCB

Channel Routing Problem

Input: Pins on the lower and upper edge

Output: Connection of each net

Constraints (Assumption)

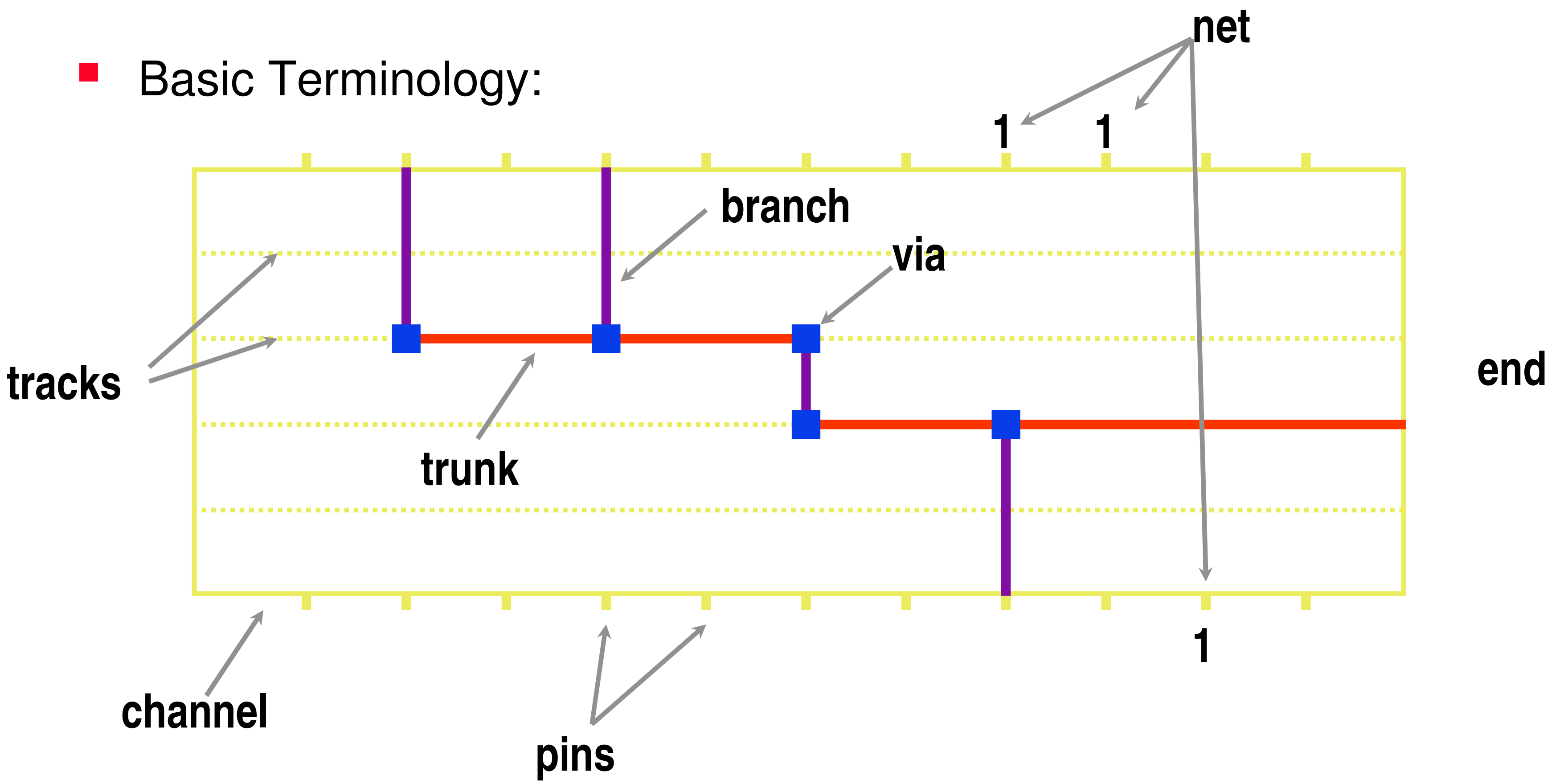
(i) grid structure
(ii) two routing layers. One for horizontal wires, the other for vertical wires
(iii) vias for connecting wires in two layers

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

Kahng/Keutzer/Newton

Channel/Switch Box Routing Algorithms

- Graph theory based algorithm
  Yoshimura and Kuh
- Greedy algorithm
  Rivest and Fiduccia

Channel routing

- Maze routing and its variations
  Lee, Robin, Soukup, Ohtsuki
- Hierarchical wire routing
  Burstein and Pelavin

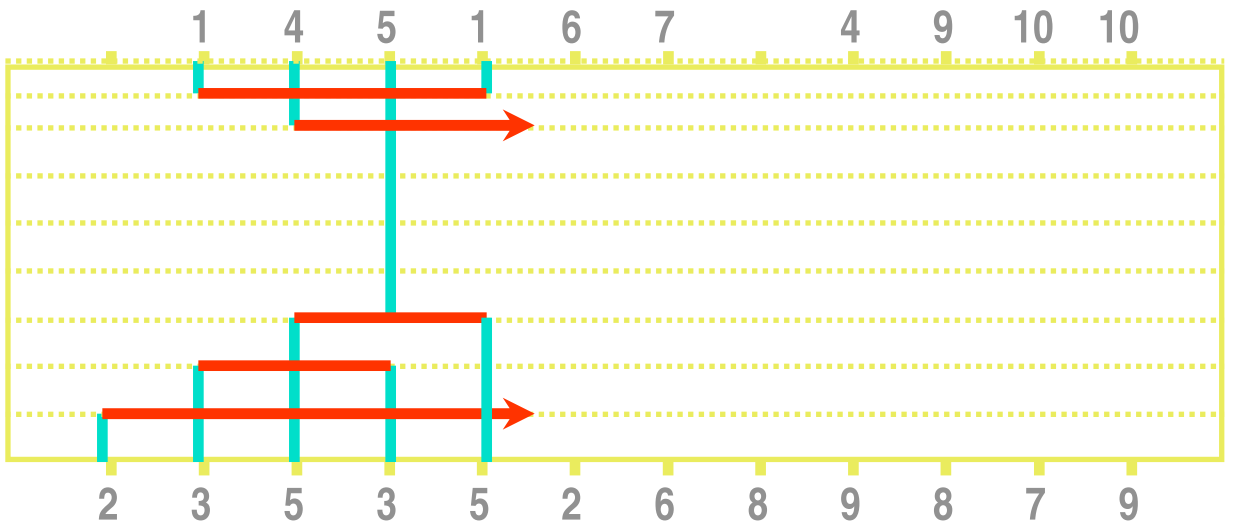
Channel / switchbox and general area routing

Greedy Channel Router

R.L. Rivest and C.M. Fiduccia " A Greedy Channel Router", 19th DAC, 1982 P418-424

- 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;
while (not done) do
    begin
        c:=c+1;
        complete wiring at column c;
    end;
```

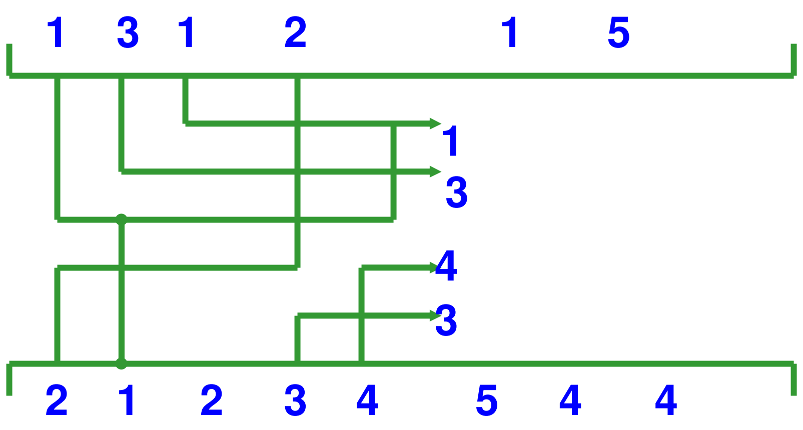
In general, at a point in a net may be

(1) empty (net 5)

(2) unsplit (nets 1,4)

(3) split (net 3)

(4) completed (net2)



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;

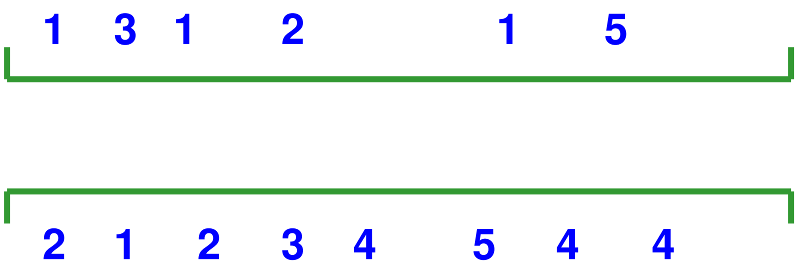
B: Collapse (connect) split nets;

C: Move split nets closer to one another;

D: Raise rising nets/lower falling nets, I.e. bring nets closer to destination terminal;

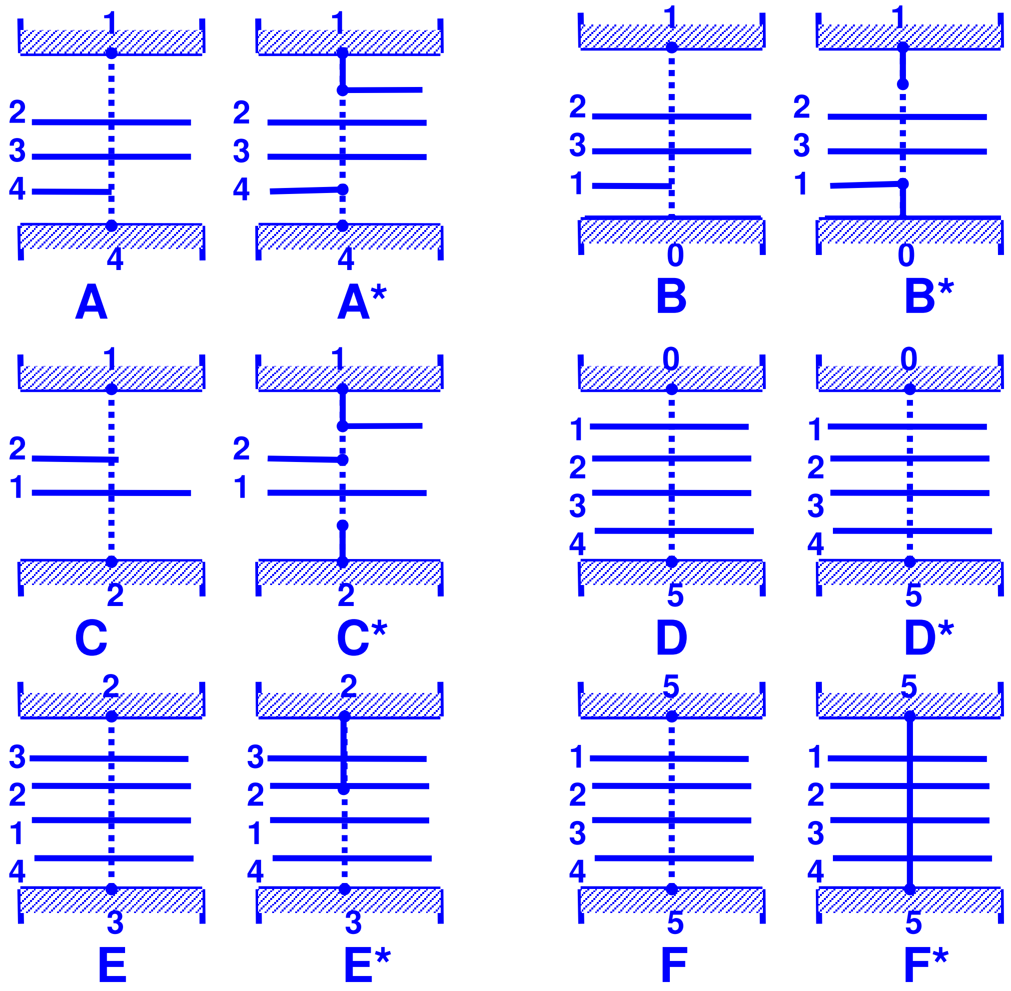
E: Widen channel when necessary;

F: Extend to next column;



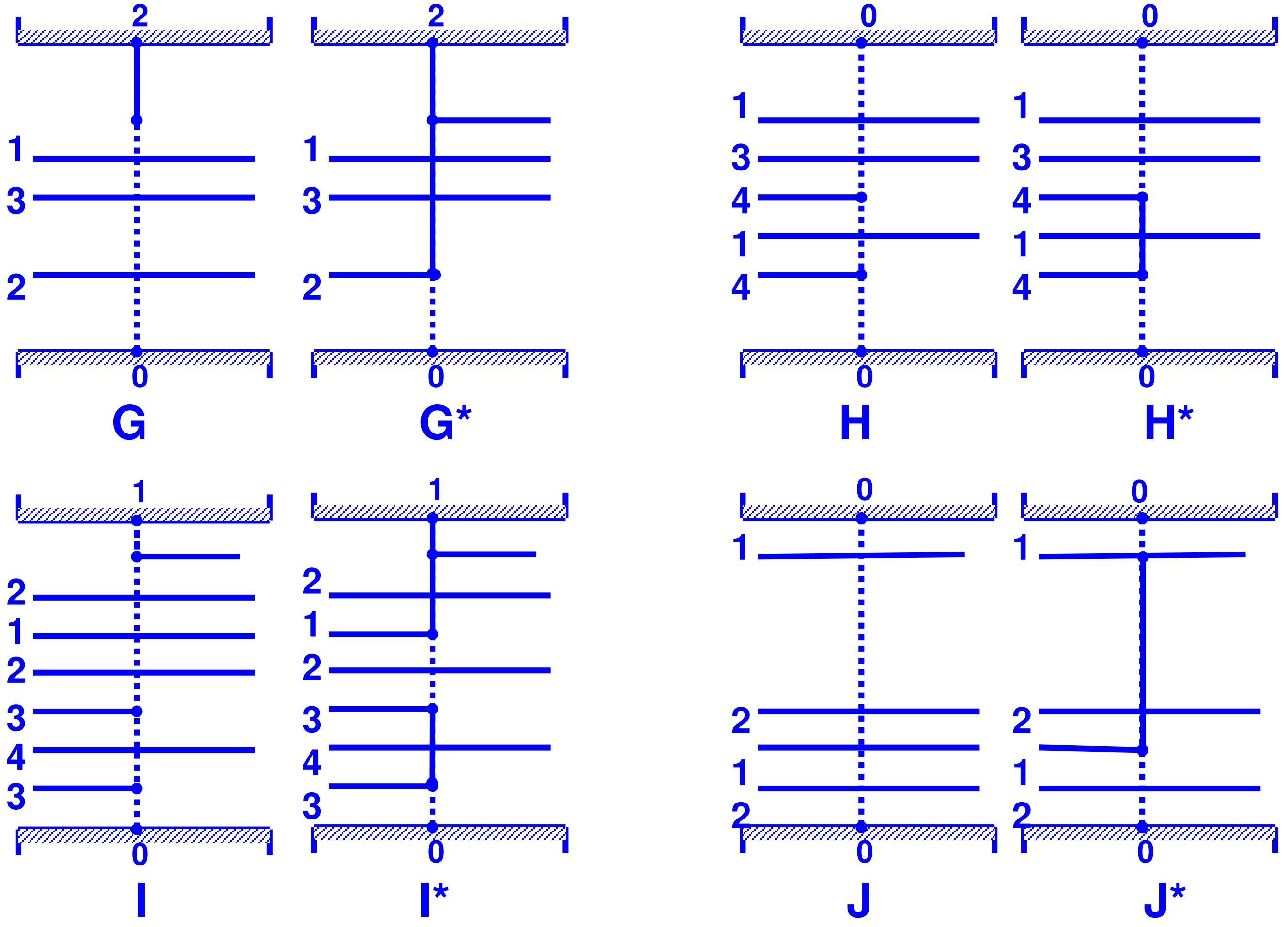
Kahng/Keutzer/Newton

(A) Make Minimal Feasible top/bottom Connetions

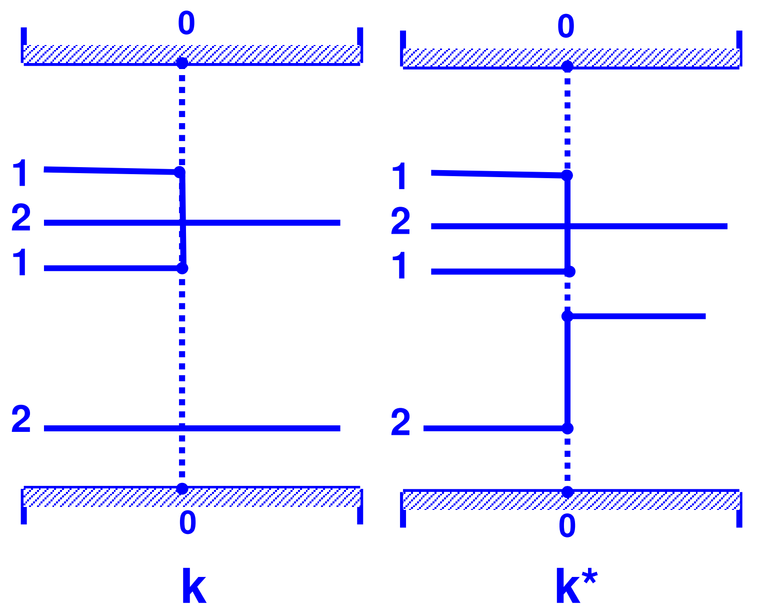


Kahng/Keutzer/Newton

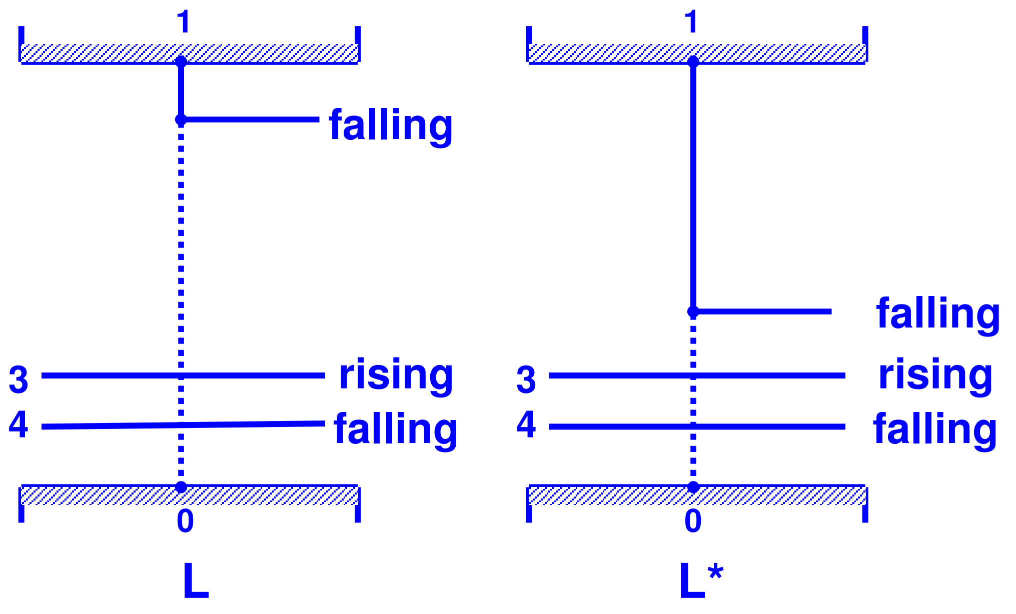
(B) Collapse/(Connect) Split Nets



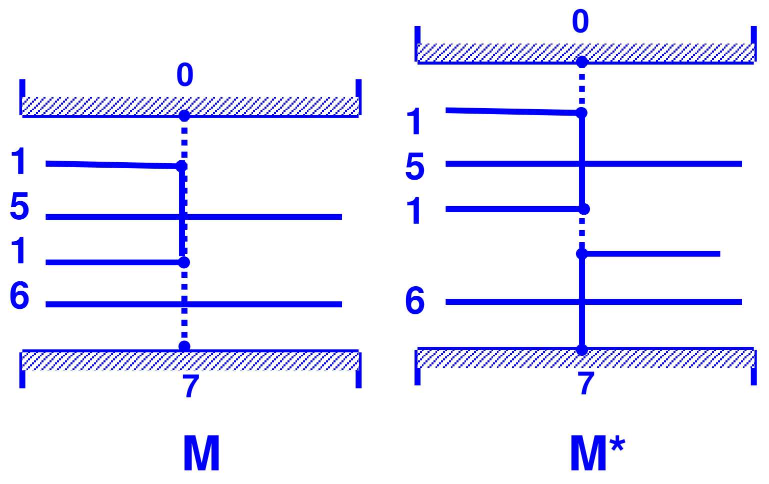
(C) Move Split Nets Closer



(D) Rising/Falling

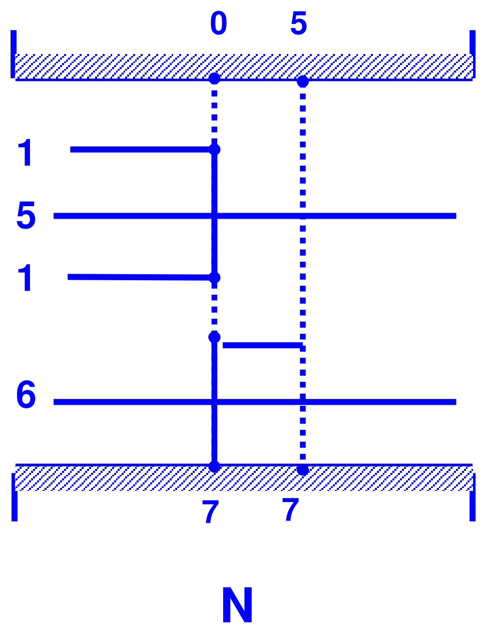


(E) Insert New Track

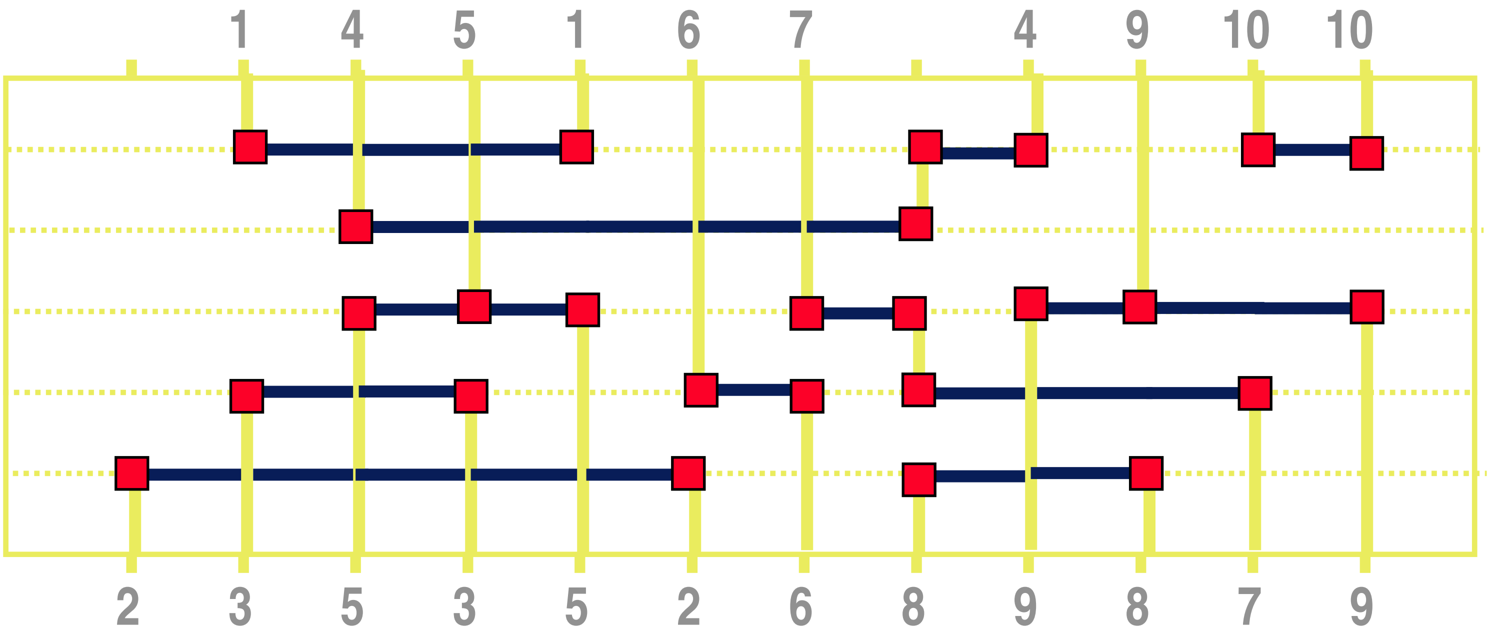


Kahng/Keutzer/Newton

(F) Extend to Next Column

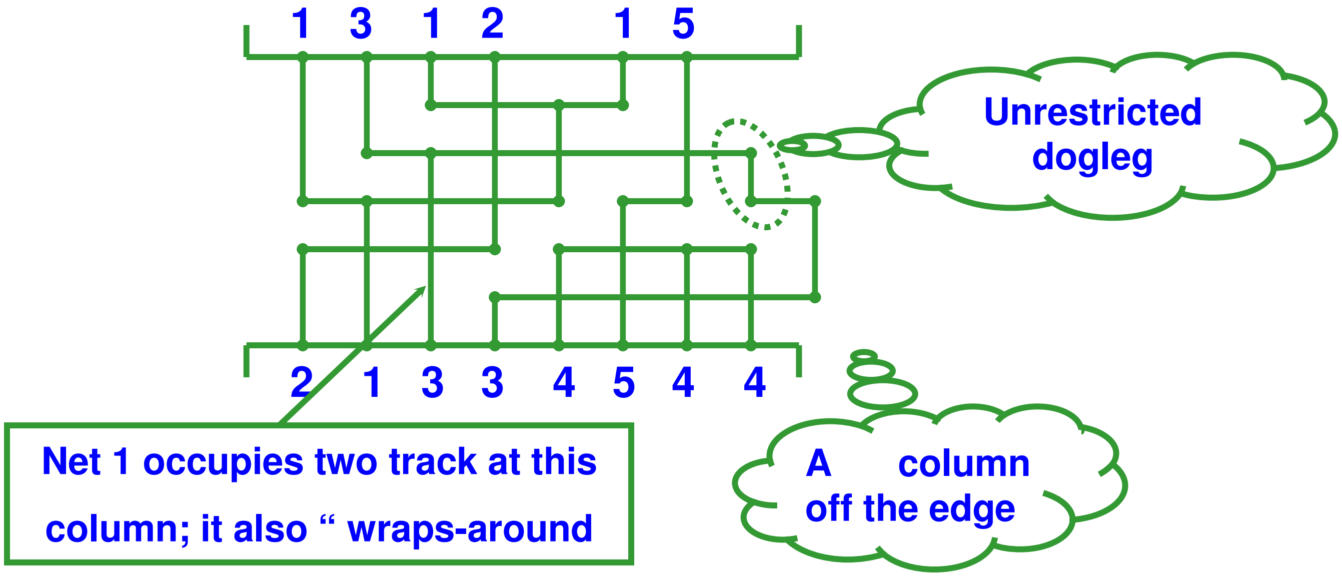


Greedy Routing Example



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;



Unrestricted dogleg

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

A column off the edge

Kahng/Keutzer/Newton

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

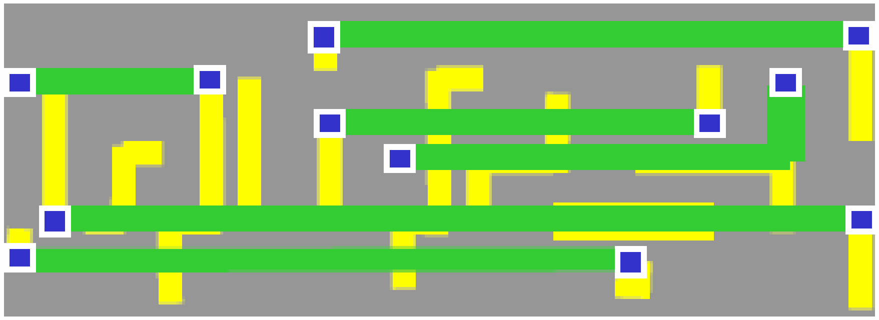


Figure 5. Channel with preferred-direction Manhattan routing

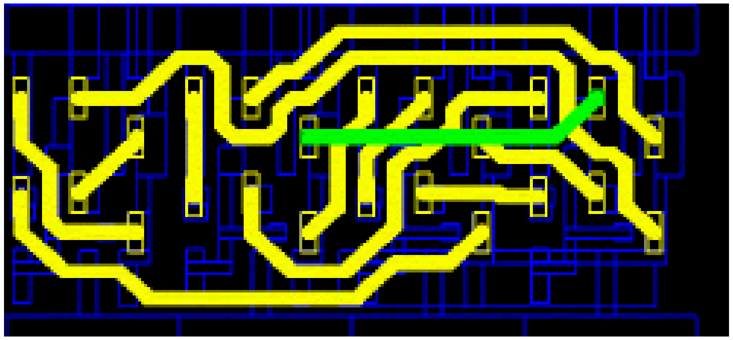
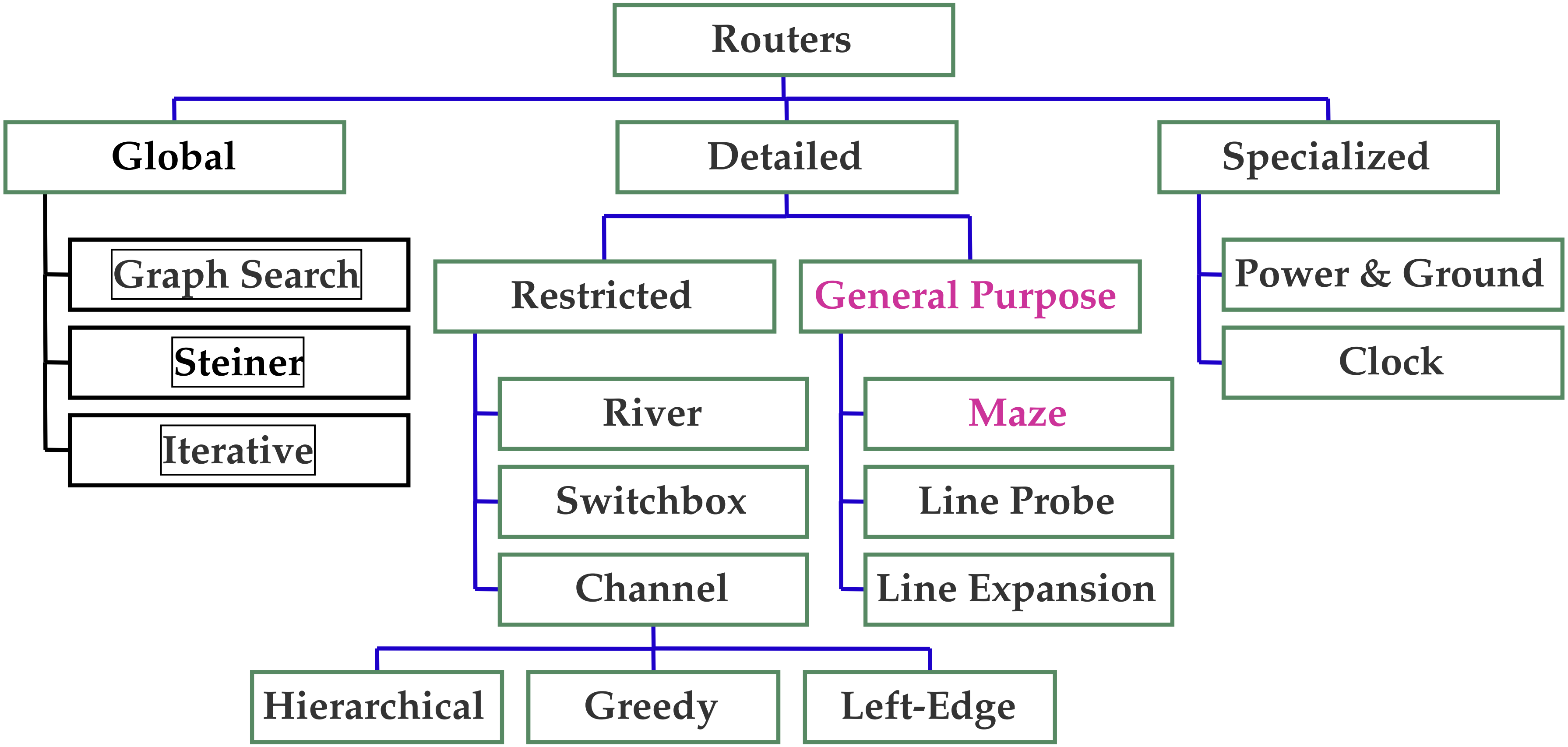


Figure 6. Channel with Liquid Routing

The X Architecture: not your father’s diagonal wiring

Steven L. Teig
Simplex Solutions, Inc.
Sunnyvale, CA
steve@simplex.com

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

Two-Terminal Routing: Maze Routing

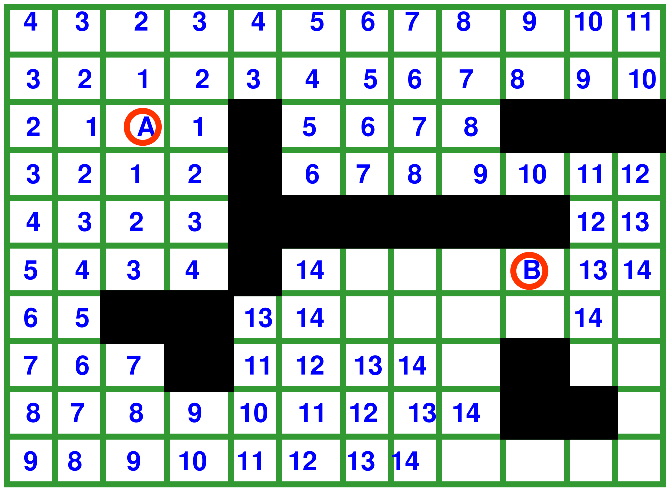
- Maze routing finds a path between source (s) and target (t) in a planar graph on a fixed grid
- Grid graph model is used to represent block placement
- Available routing areas are unblocked vertices, obstacles are blocked vertices
- Finds an optimal path

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.

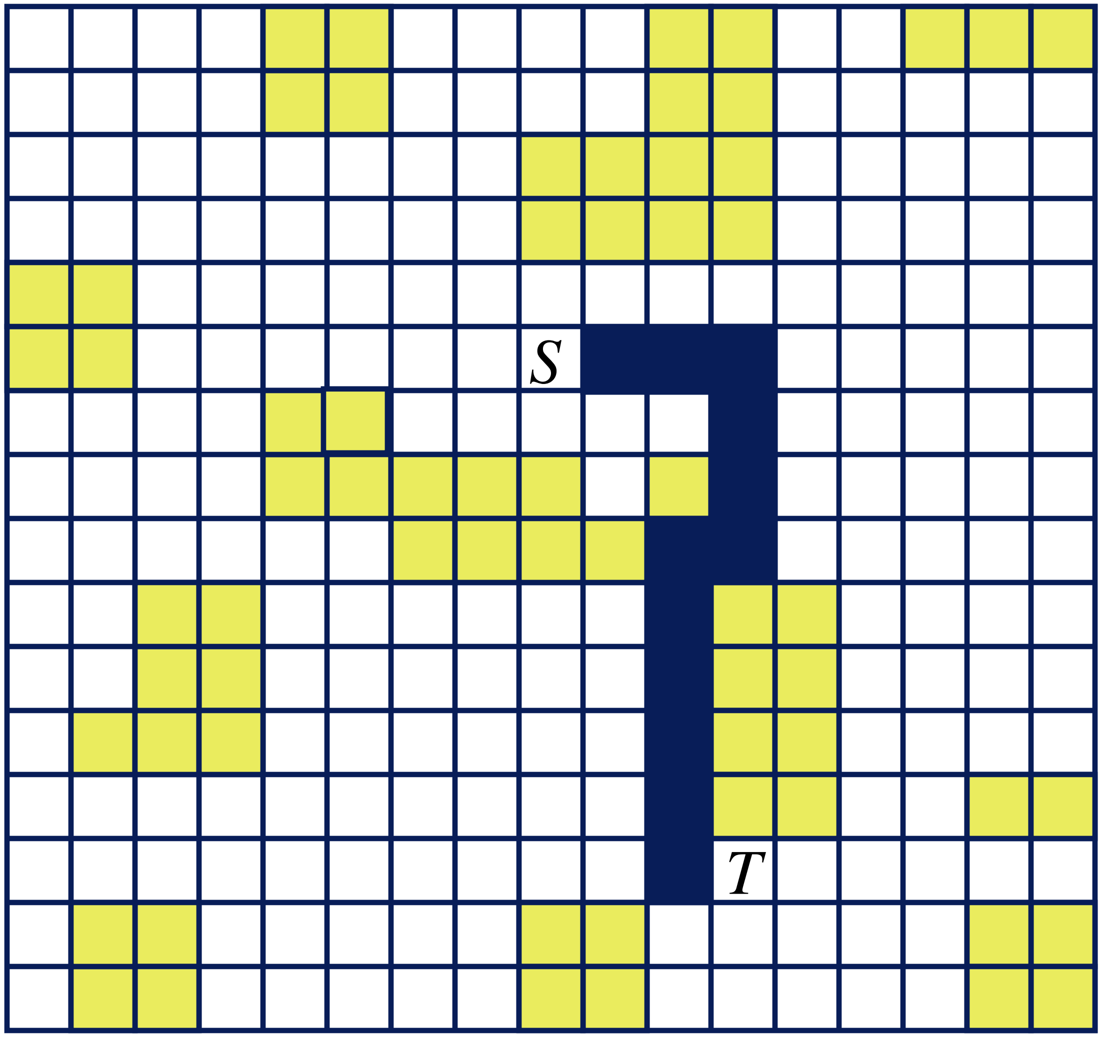
"An Algorithm for Path Connection and its Application", C.Y. Lee, IRE Transactions on Electronic Computers, 1961.



4	3	2	3	4	5	6	7	8	9	10	11
3	2	1	2	3	4	5	6	7	8	9	10
2	1	A	1	■	5	6	7	8	■	■	■
3	2	1	2	■	6	7	8	9	10	11	12
4	3	2	3	■	■	■	■	■	■	12	13
5	4	3	4	■	14				B	13	14
6	5	■	■	13	14					14	
7	6	7	■	11	12	13	14		■		
8	7	8	9	10	11	12	13	14	■	■	
9	8	9	10	11	12	13	14				

Courtesy K. Keutzer et al. UCB

Maze Routing



E. Young

Kahng/Keutzer/Newton

Basic Idea

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

E. Young

An Illustration

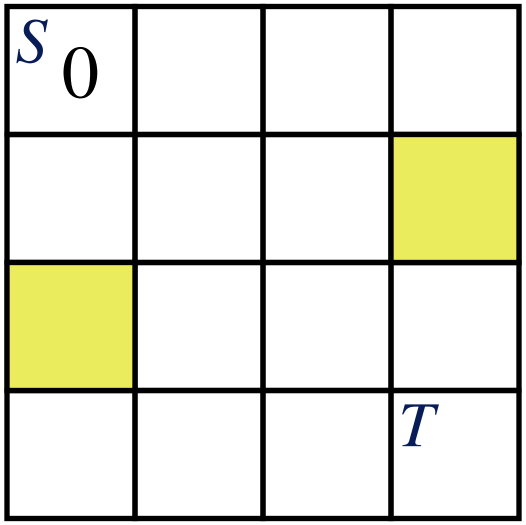
S 0	1	2	3
1	2	3	🟨
🟨	3	4	5
5	4	5	T 6

E. Young

Wave Propagation

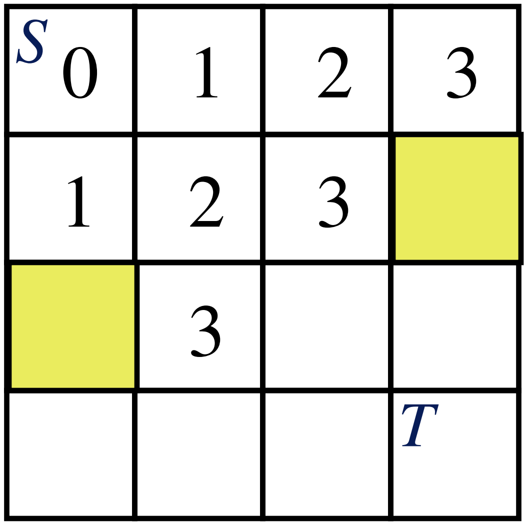
- At step *k*, all vertices at Manhattan-distance *k* from *S* are labeled with *k*.
- A Propagation List (FIFO) is used to keep track of the vertices to be considered next.

E. Young



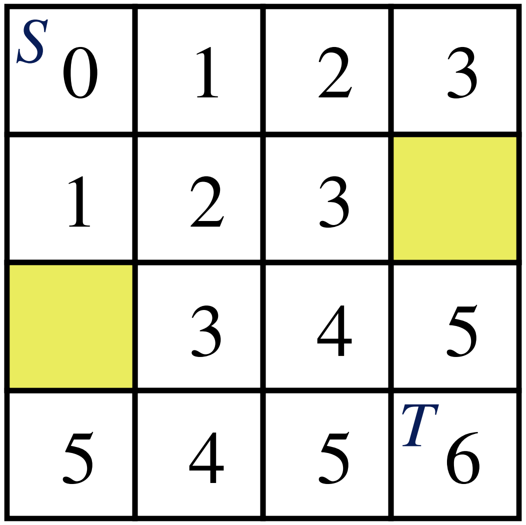
S 0			
			▨
▨			
			T

After Step 0



S 0	1	2	3
1	2	3	▨
▨	3		
			T

After Step 3

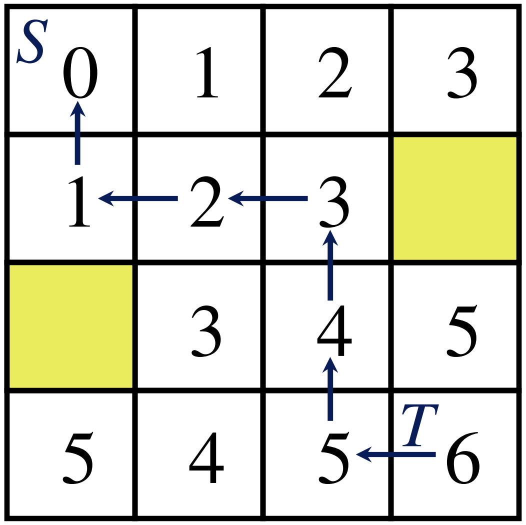


S 0	1	2	3
1	2	3	▨
▨	3	4	5
5	4	5	*T* 6

After Step 6

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

Kahng/Keutzer/Newton

How many grids visited using Lee’s algorithm?

13	12	11	10	🟨	🟨	7	6	7	7	🟨	🟨	9	10	🟨	🟨	🟨
12	11	10	9	🟨	🟨	6	5	6	7	🟨	🟨	8	9	10	11	12
11	10	9	8	7	6	5	4	🟨	🟨	🟨	🟨	7	8	9	10	11
10	9	8	7	6	5	4	3	🟨	🟨	🟨	🟨	6	7	8	9	10
🟨	🟨	7	6	5	4	3	2	1	2	3	4	5	6	7	8	9
🟨	🟨	6	5	4	3	2	1	S	1	2	3	4	5	6	7	8
9	8	7	6	🟨	🟨	3	2	1	2	3	4	5	6	7	8	9
10	9	8	7	🟨	🟨	🟨	🟨	🟨	3	🟨	5	6	7	8	9	10
11	10	9	8	9	10	🟨	🟨	🟨	🟨	7	6	7	8	9	10	11
12	11	🟨	🟨	10	11	12	11	10	9	8	🟨	🟨	9	10	11	12
13	12	🟨	🟨	11	12	13	12	11	10	9	🟨	🟨	10	11	12	13
	🟨	🟨	🟨	12	13		13	12	11	10	🟨	🟨	11	12	13	
				13				13	12	11	🟨	🟨	12	13	🟨	🟨
									13	12	T		13			
	🟨	🟨						🟨	🟨	13					🟨	🟨
	🟨	🟨						🟨	🟨						🟨	🟨

E. Young

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 × 4000 grid structure:
  - 24 bits per label
  - Total 48 Mbytes of memory!
- Generally, time and space complexity per net O($height \times width$)

E. Young

Lee’s Algorithm

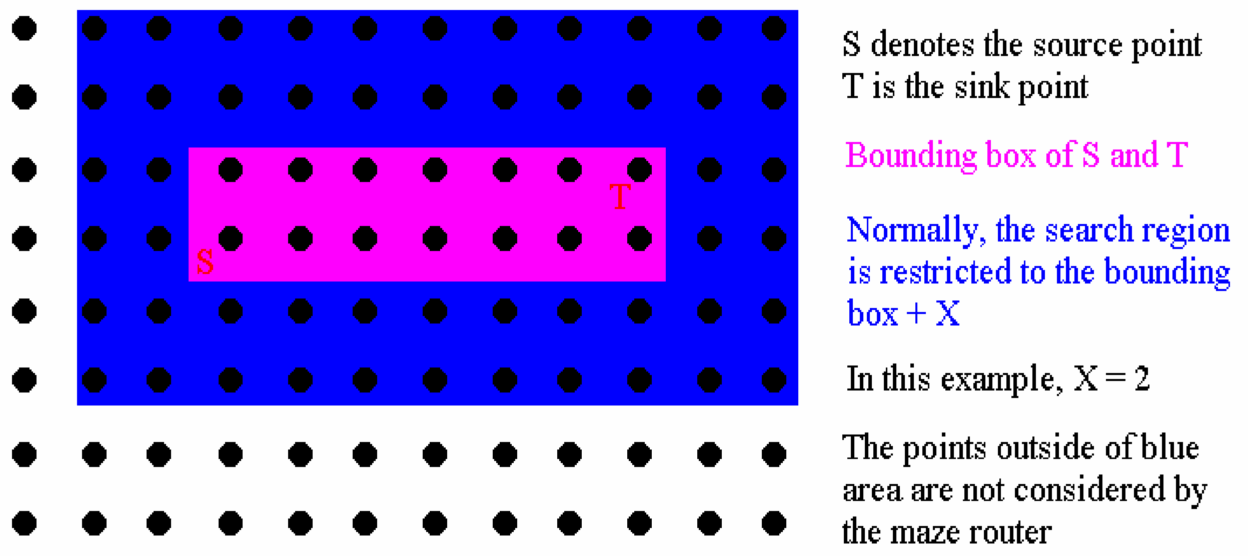
- Strengths:
  - Guarantee to find a connection if one exists.
  - Guarantee shortest path.
- Weaknesses:
  - Large memory requirements.
  - Slow
- Applications:
  - Global routing and detailed routing

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×N grid

Memory: total layout grid needs to be kept NxN

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

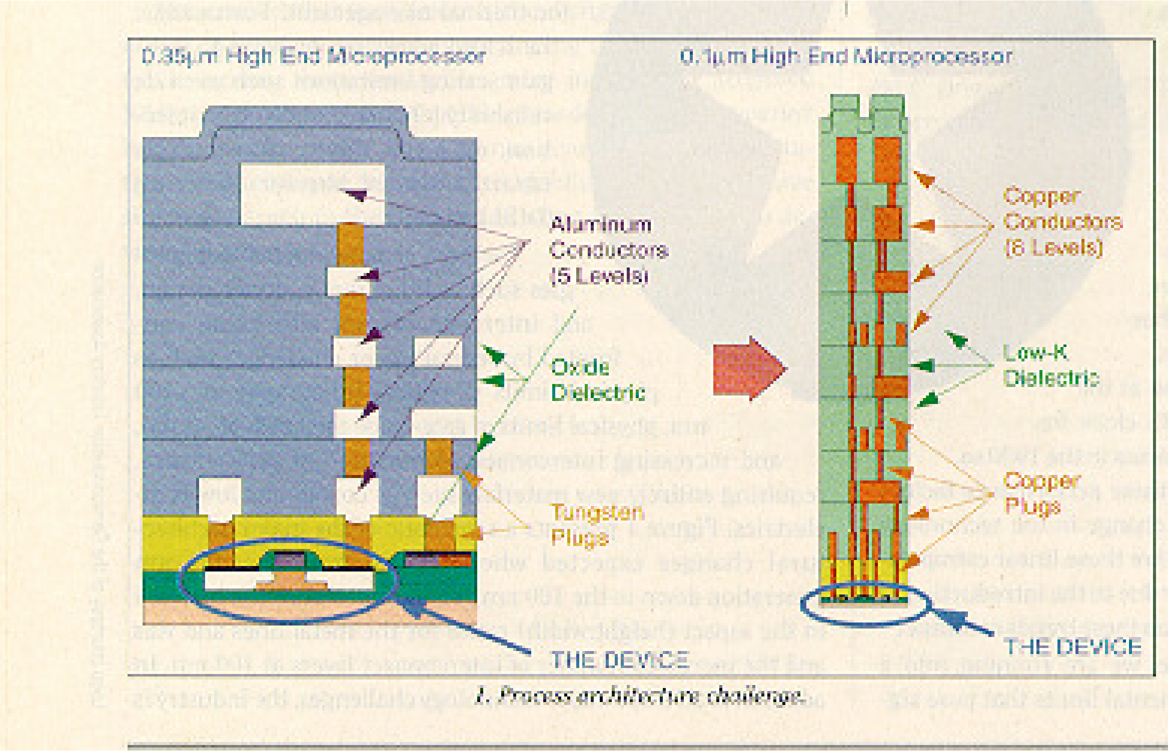
Further improvements

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

Kahng/Keutzer/Newton

Courtesy K. Keutzer et al. UCB

Grid Problem with Maze Routing


0.35µm High End Microprocessor
Aluminum Conductors (5 Levels)
Oxide Dielectric
Tungsten Plugs
THE DEVICE
0.1µm High End Microprocessor
Copper Conductors (8 Levels)
Low-K Dielectric
Copper Plugs
THE DEVICE

1. Process architecture challenge.

Photo courtesy:
Jan M. Rabaey
Anantha Chandrakasan
Borivoje Nikolic

- 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

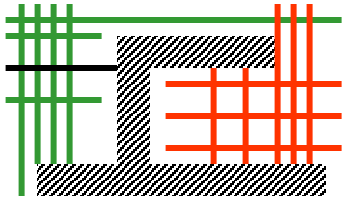
Line-Probe Algorithm

Mikami&Tabuchi IFIPS Proc, Vol H47, pp 1475-1478, 1968

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

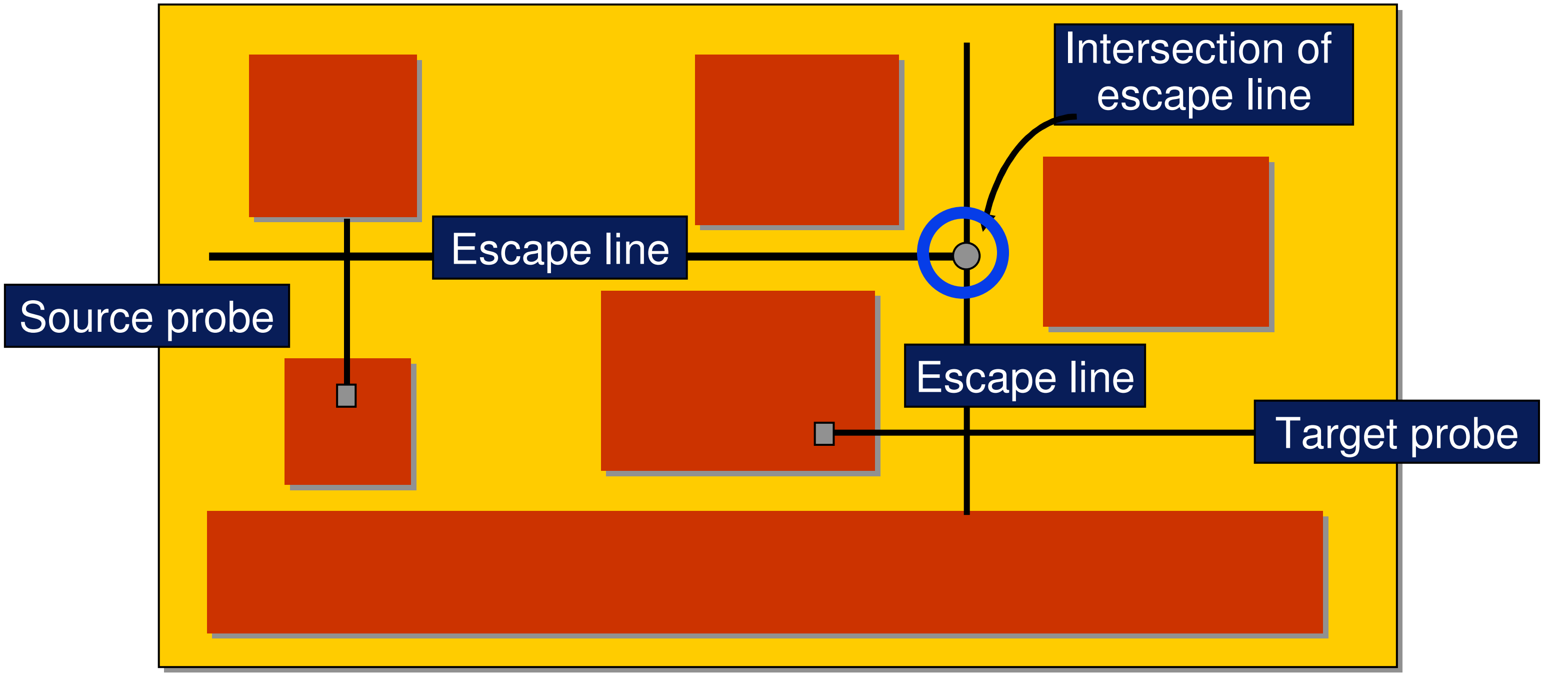


Kahng/Keutzer/Newton

Courtesy K. Keutzer et al. UCB

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

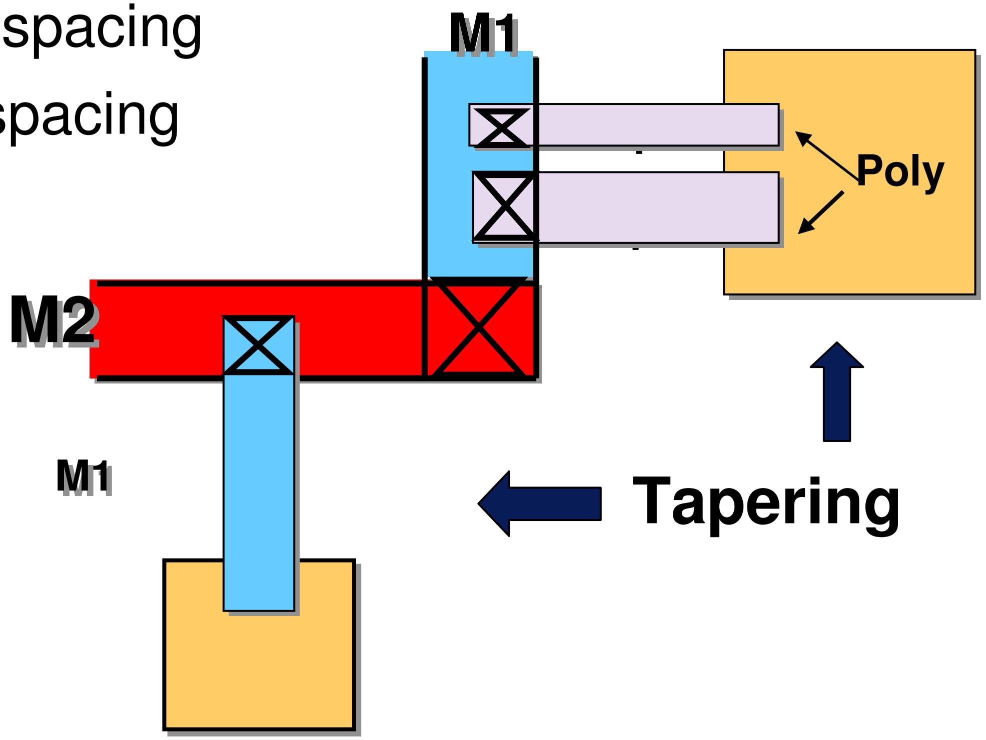
Courtesy K. Keutzer et al. UCB

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

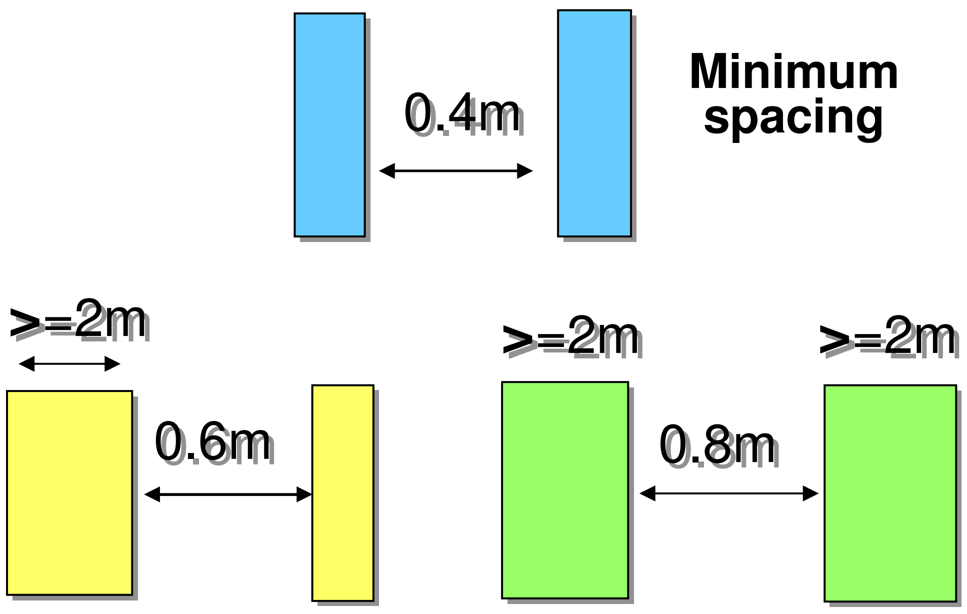
Detailed Routing Issues

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



Detailed Routing Issues

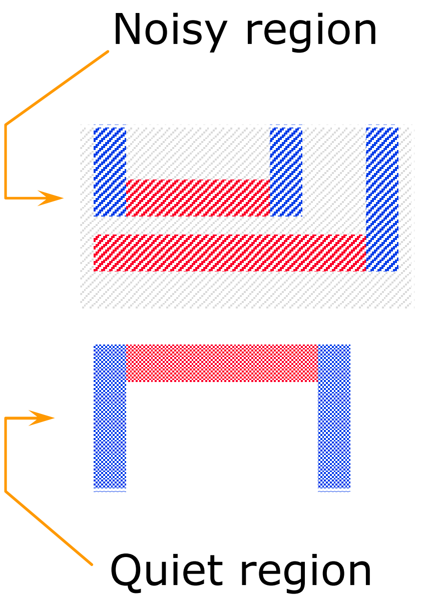
- Width and spacing rule



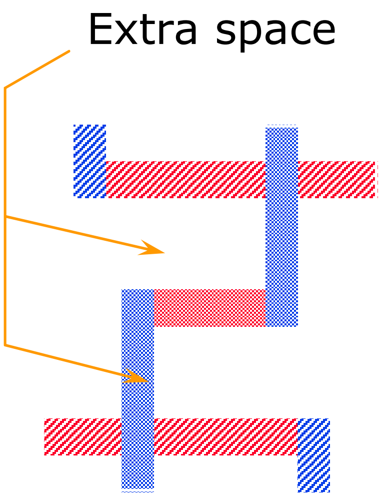
Width-based Spacing

Detailed Routing Issues

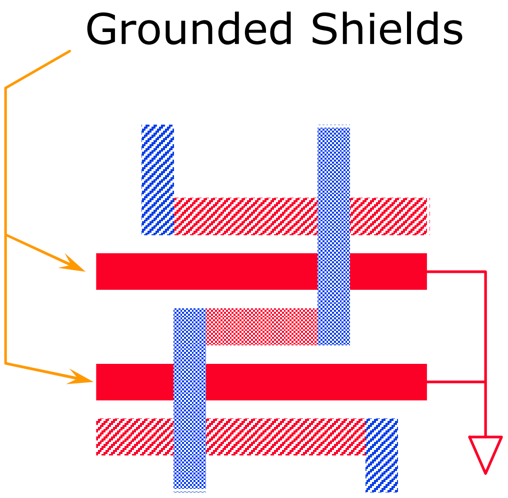
- Noise-driven – i.e. noise minimization driven



Segregation



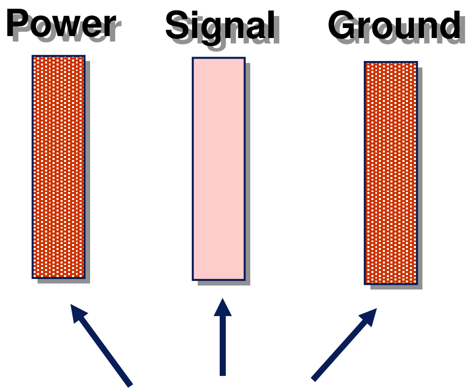
Spacing



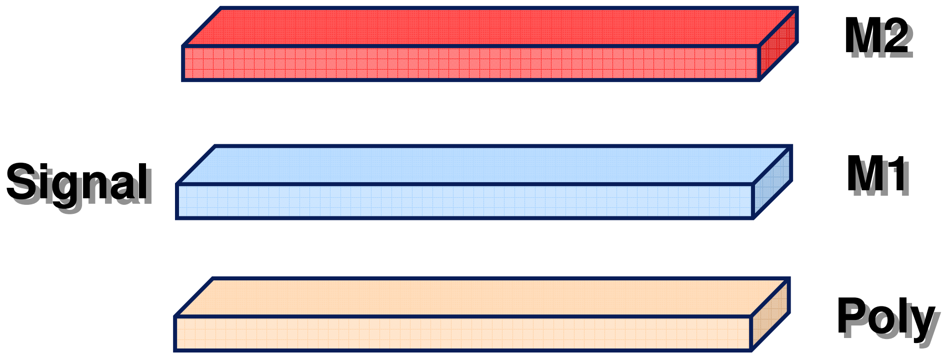
Shielding

Detailed Routing Issues

- Shielding
  - Same-layer shielding
  - Adjacent-layer shielding



Same-Layer Shielding

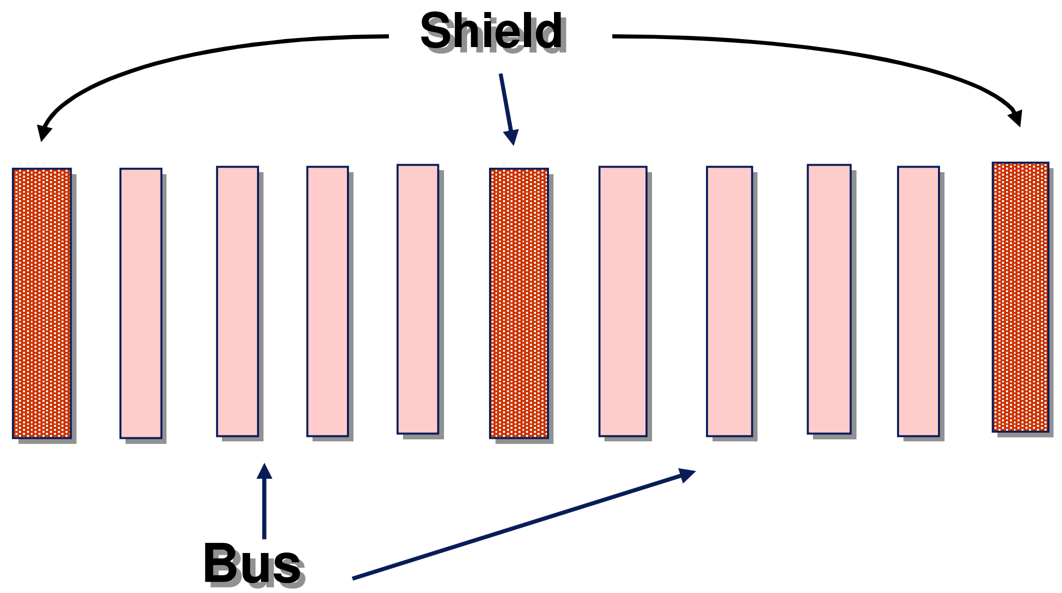


Adjacent-Layer Shielding

Kahng/Keutzer/Newton

Detailed Routing Issues

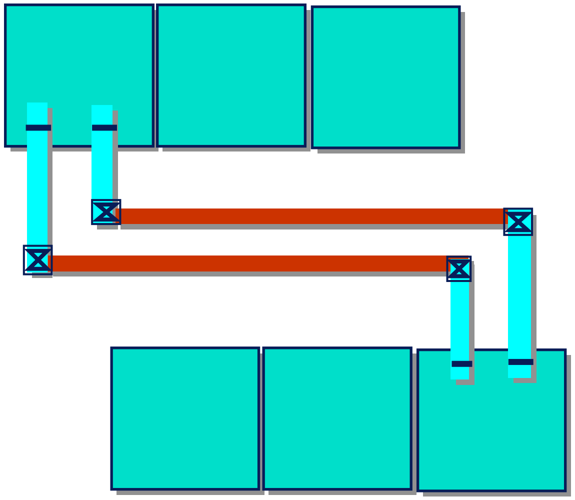
- Shielding
  - Bus shielding
  - Bus interleaving



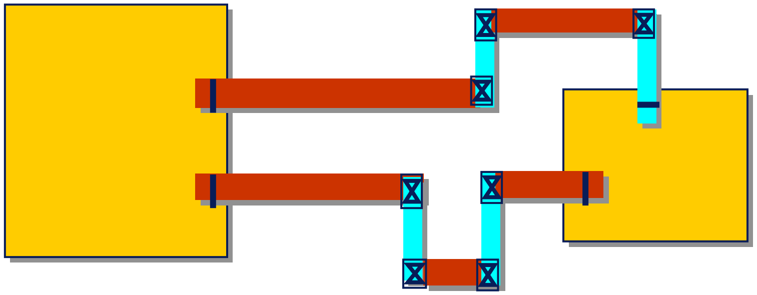
Bus Shielding

Detailed Routing Issues

- Differential pair routing
- Balanced length or capacitance



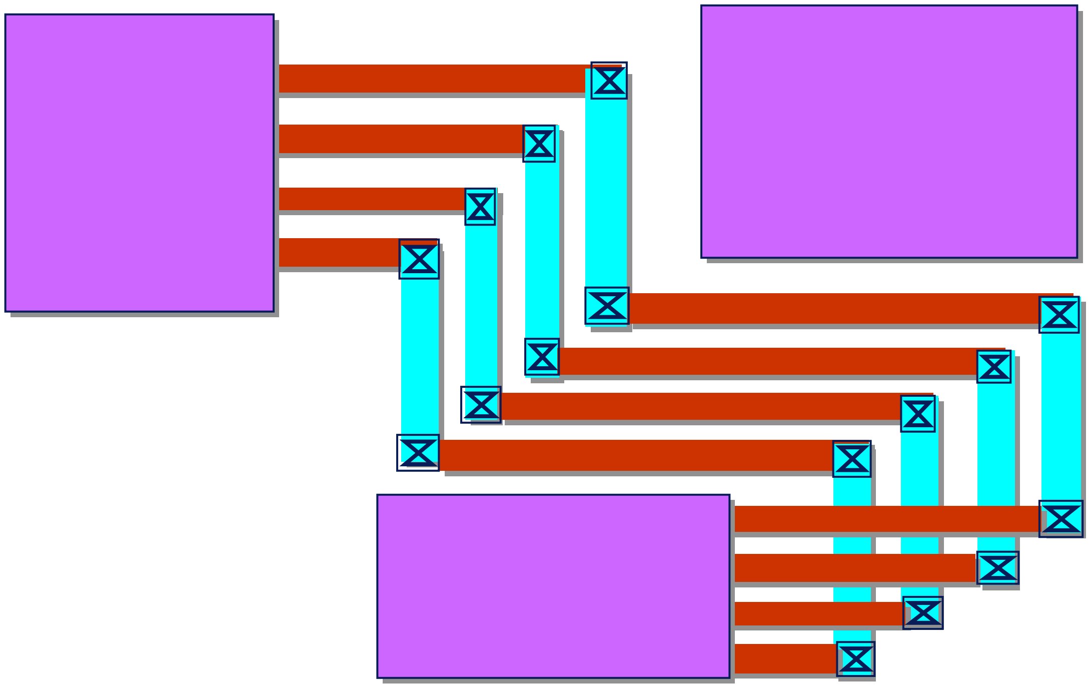
Differential



Balanced length

Detailed Routing Issues

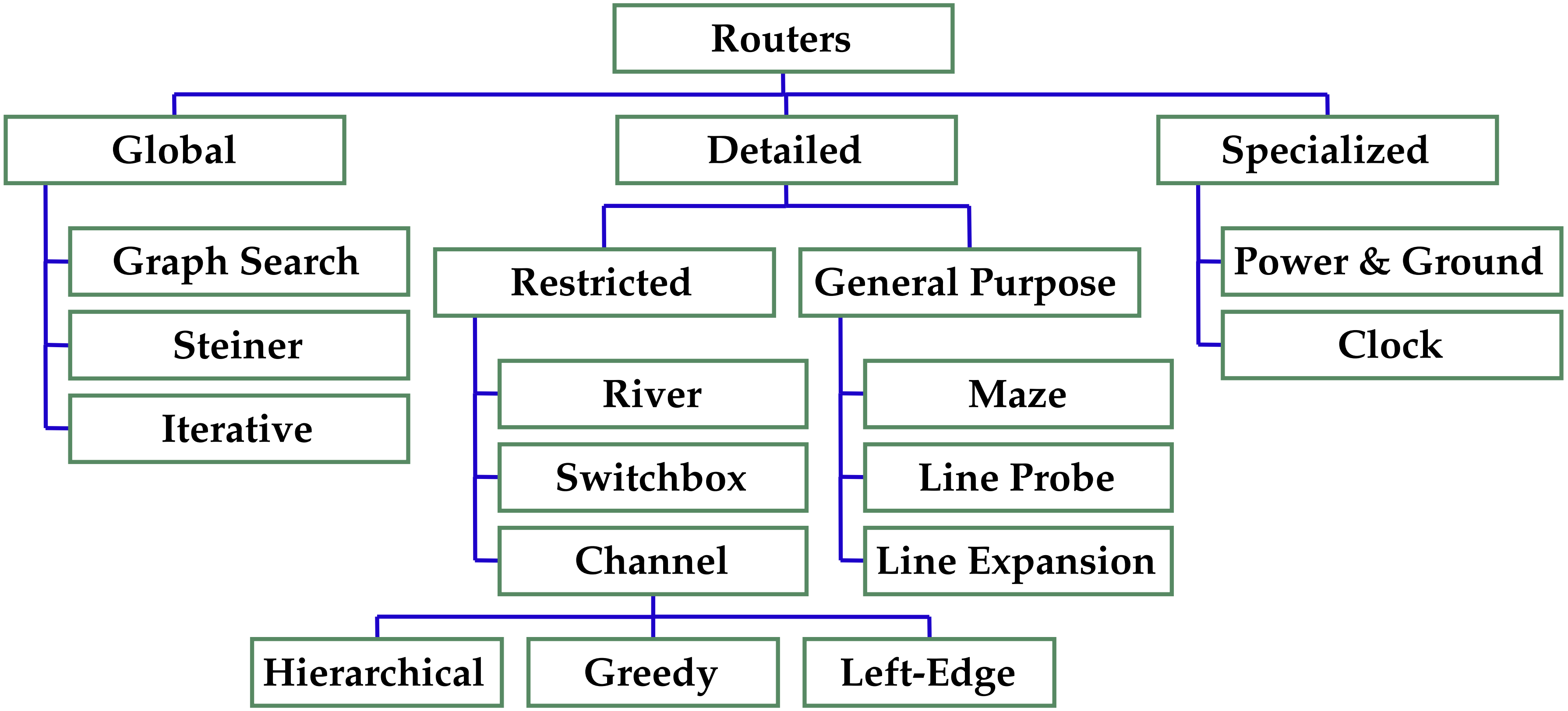
- **Bus Routing**



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)
- Routing is not a “showstopper” in current VLSI designs
- But … routing is a perennial 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

Courtesy K. Keutzer et al. UCB

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

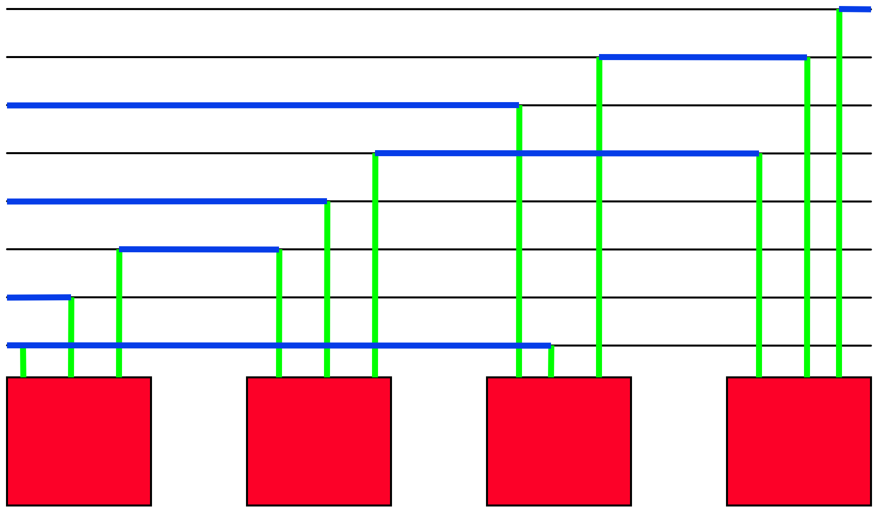
- For the Lee’s algorithm, labels are needed during the retrace phase.
- But there are only two possible labels for the neighbors of each vertex labeled *i*, which are, *i*-1 and *i*+1.
- Want a labeling scheme such that each label has its preceding label different from its succeeding label.

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

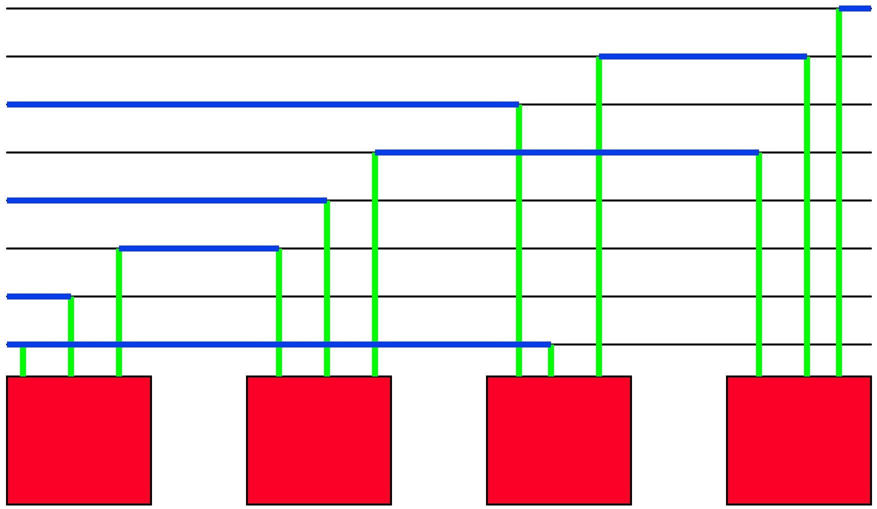


Dehon- Caltech

Kahng/Keutzer/Newton

Trivial Channel Routing

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

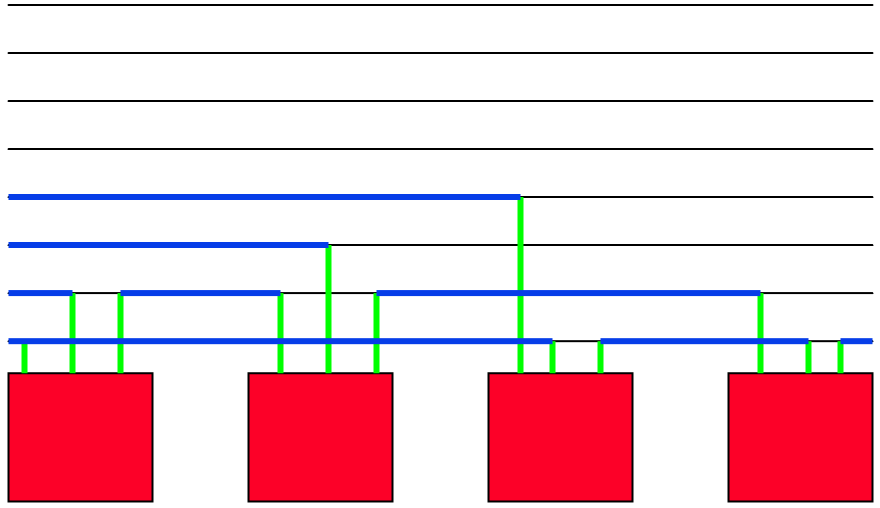


Dehon- Caltech

Kahng/Keutzer/Newton

Sharing Tracks

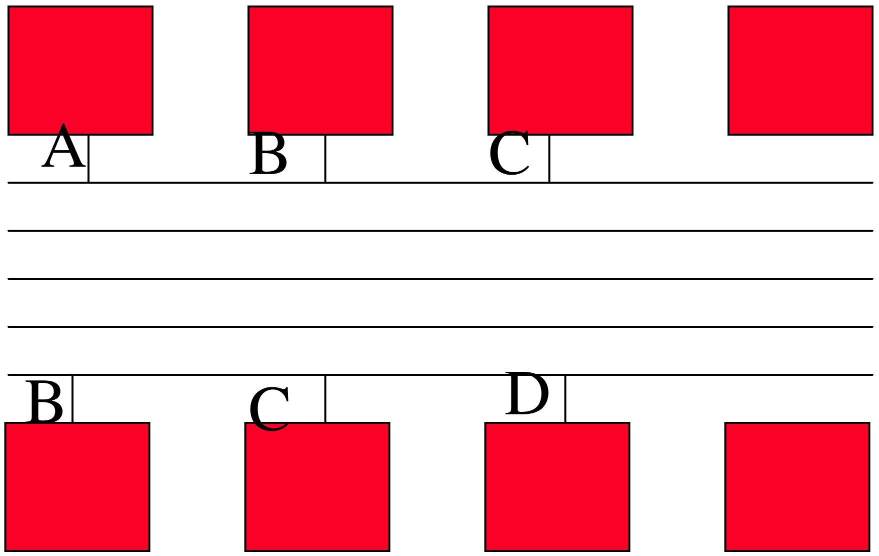
- Want to Minimize tracks used
- Trick is to share tracks



Dehon- Caltech

Not that Easy

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

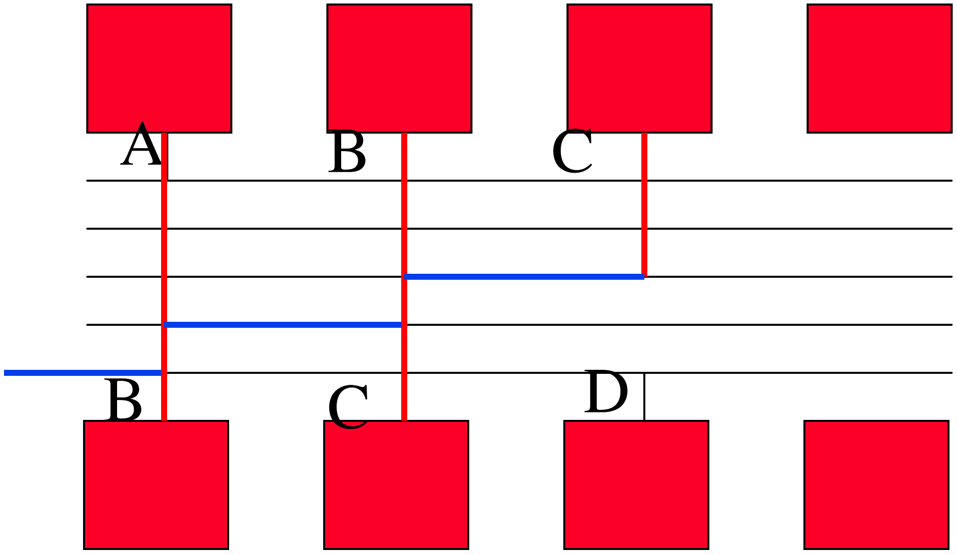


Dehon- Caltech

Not that Easy

Dehon- Caltech

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

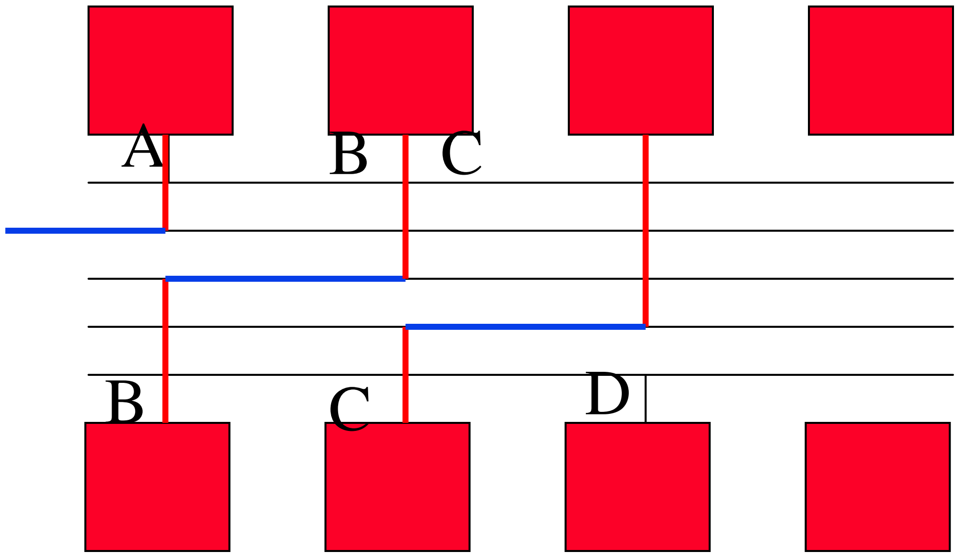


Bad assignment
Overlap:
A,B
B,C

Not that Easy

Dehon- Caltech

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

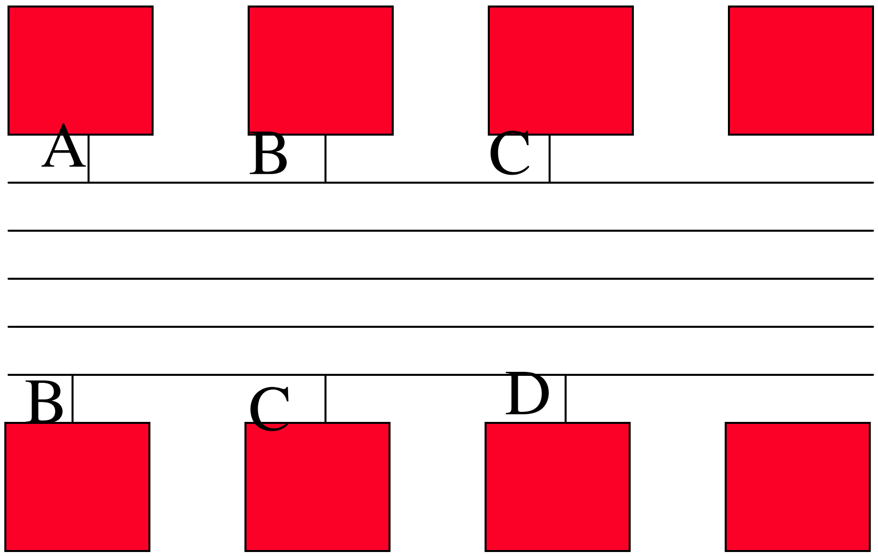


Valid assignment avoids overlap

Not that Easy

Dehon- Caltech

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



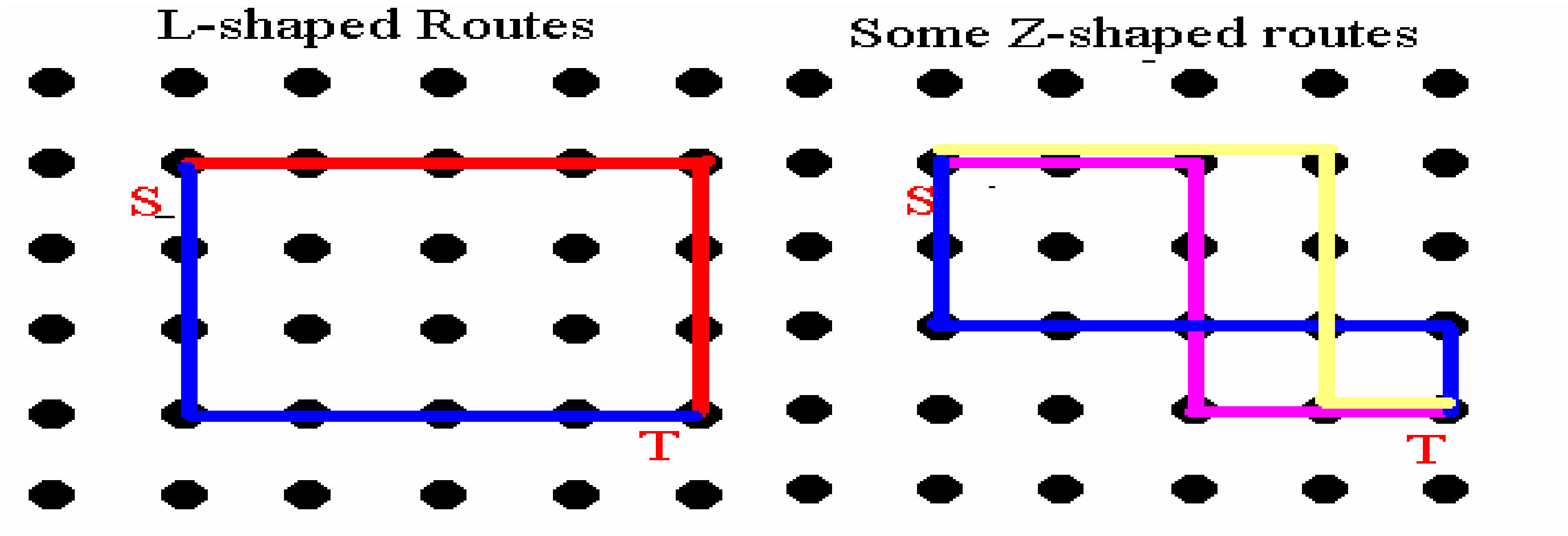
There are vertical constraints on ordering

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



Courtesy K. Keutzer et al. UCB

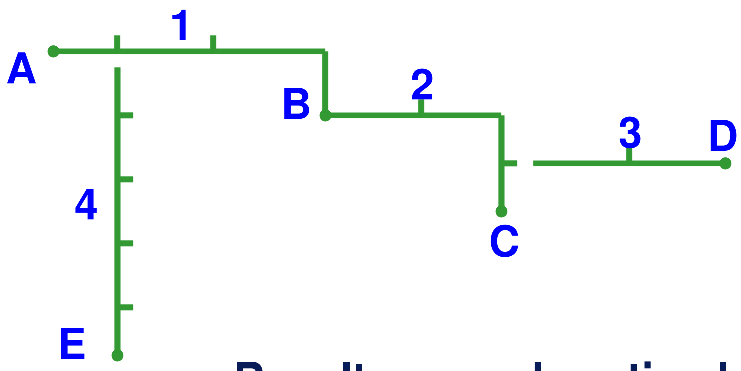
Steiner Trees

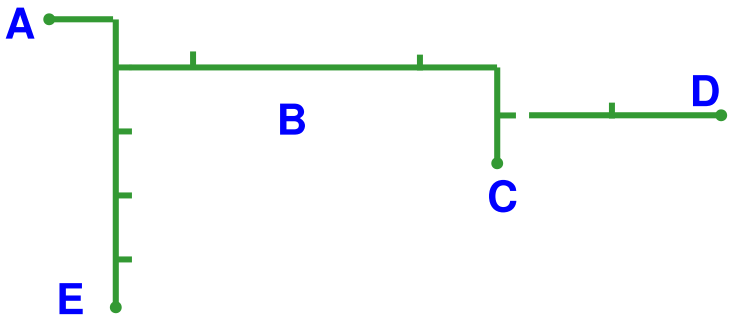
Connecting Multi-Terminal Nets

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)

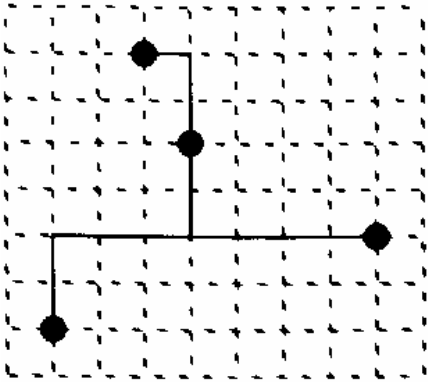




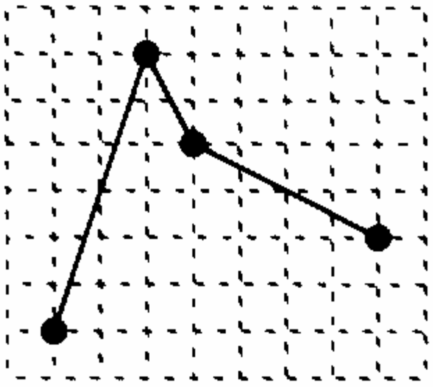
- Results are sub-optimal
- Inherit time and memory cost of maze and line-probe algorithms

Courtesy K. Keutzer et al. UCB

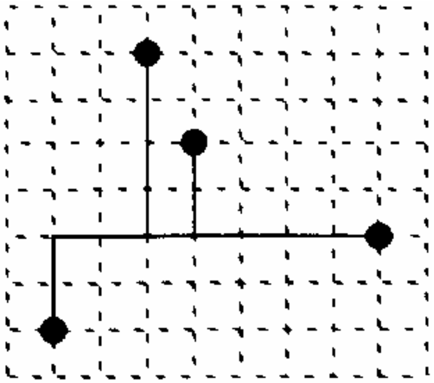
Multi-terminal Nets: Different Routing Options



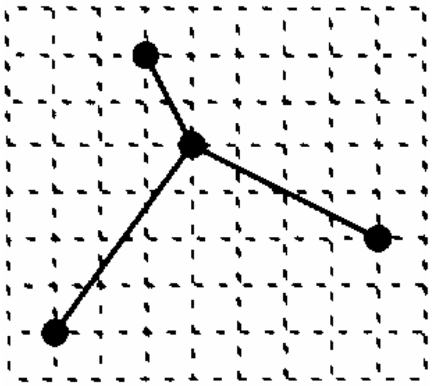
(a) Steiner Tree (14)



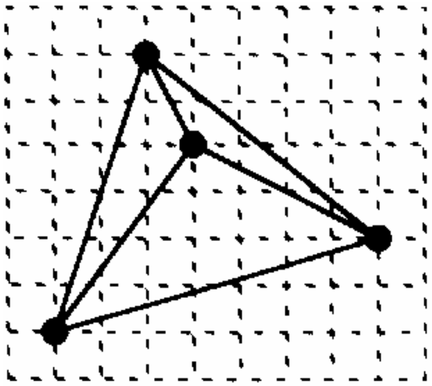
(d) Chain (17)



(b) Steiner Tree with Trunk (15)



(c) Minimum Spanning Tree (16)



(e) Complete Graph (42)

Cost is determined by routing model

Courtesy K. Keutzer et al. UCB

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

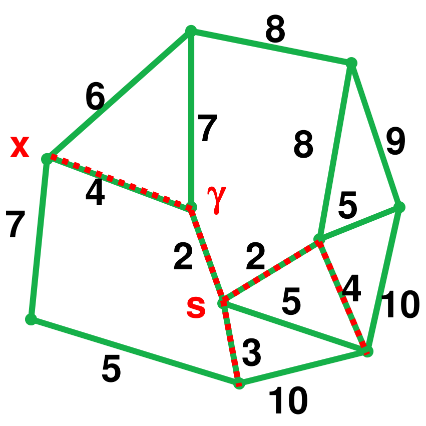
Minimum Spanning Tree (Prim's construction)

Given a weighted graph

Find a spanning tree whose weight is minimum

Prim's algorithm

```
start with an arbitrary node s
$T \leftarrow \{s\}$
while T is not a spanning tree
    find the closest pair $x \in V-T$, $y \in T$
    add (x,y) to T
```



runs in $O(n^2)$ time

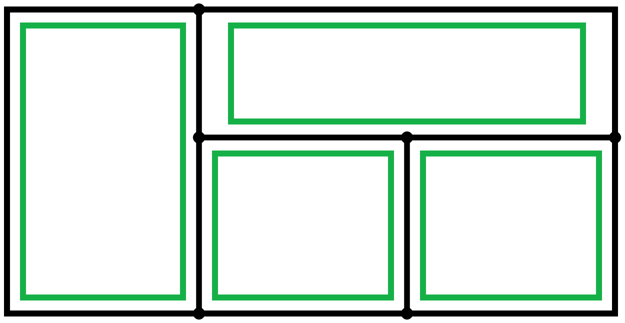
very simple to implement

always gives a tree of minimum cost

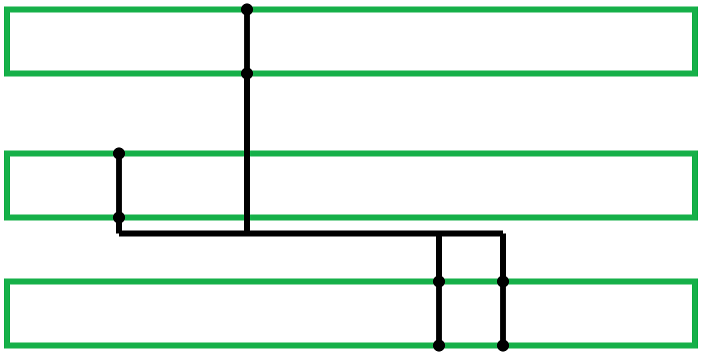
Courtesy K. Keutzer et al. UCB

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



Courtesy K. Keutzer et al. UCB

Concurrent Global Routing

Global Routing: Concurrent Approaches

- 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

Courtesy K. Keutzer et al. UCB

Yoshimura and Kuh

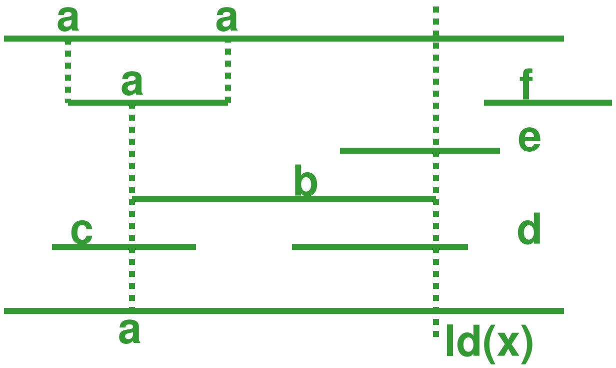
Horizontal Constraint Graph (HCG)

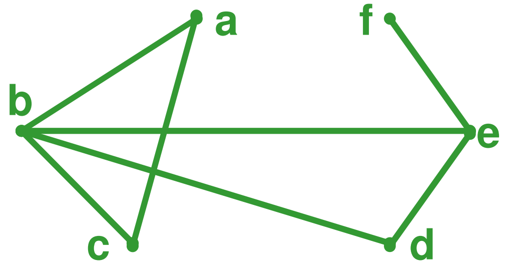
1. Node v_i: represents a horizontal interval spanned by net i

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 ≥ 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

Courtesy K. Keutzer et al. UCB

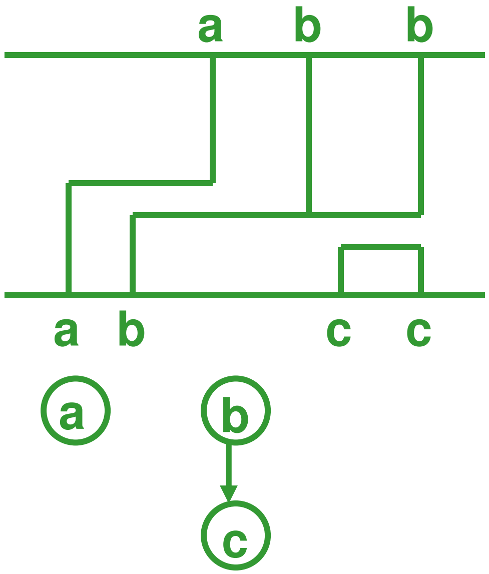
Vertical Constraint Graph (VCG)

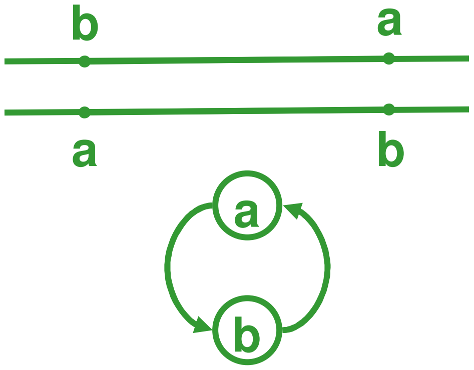
1. Node: represents a net

2. Edge (a1→a2) exists if at some column:
   - Net a1 has a terminal on the upper edge
   - Net a2 has a terminal on the lower edge
   - Edge a1→a2 means that Net a1 must be above Net a2

3. Establishes lower bound: # tracks ≥ longest path in VCG

4. VCG may have a cycle !

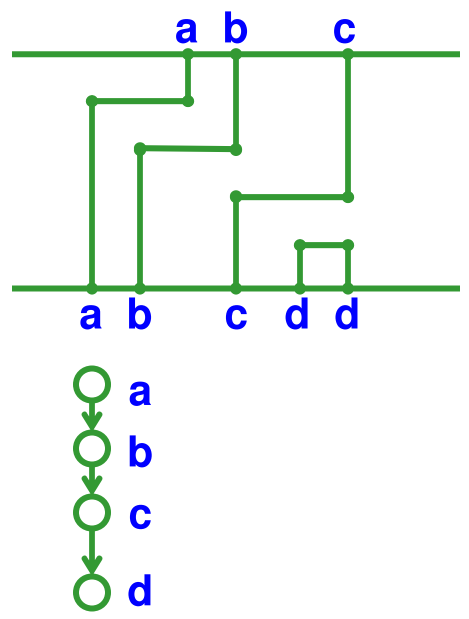


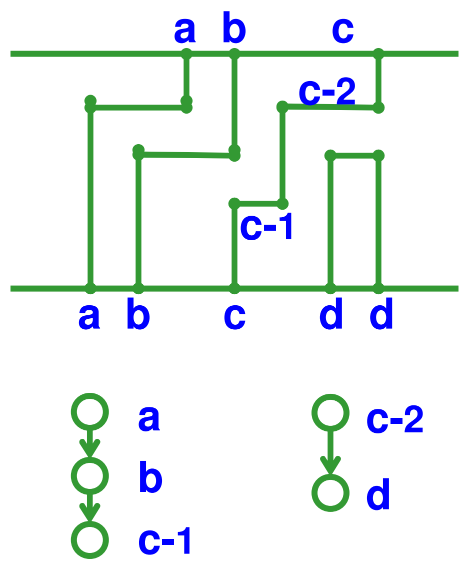


Courtesy K. Keutzer et al. UCB

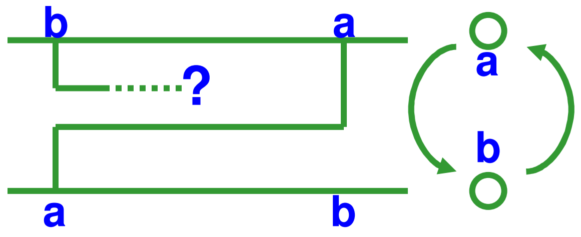
Doglegs in Channel Routing

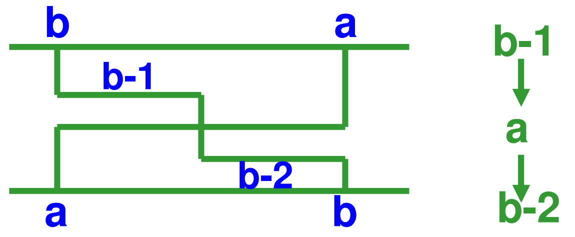
Doglegs may reduce the longest path in VCG





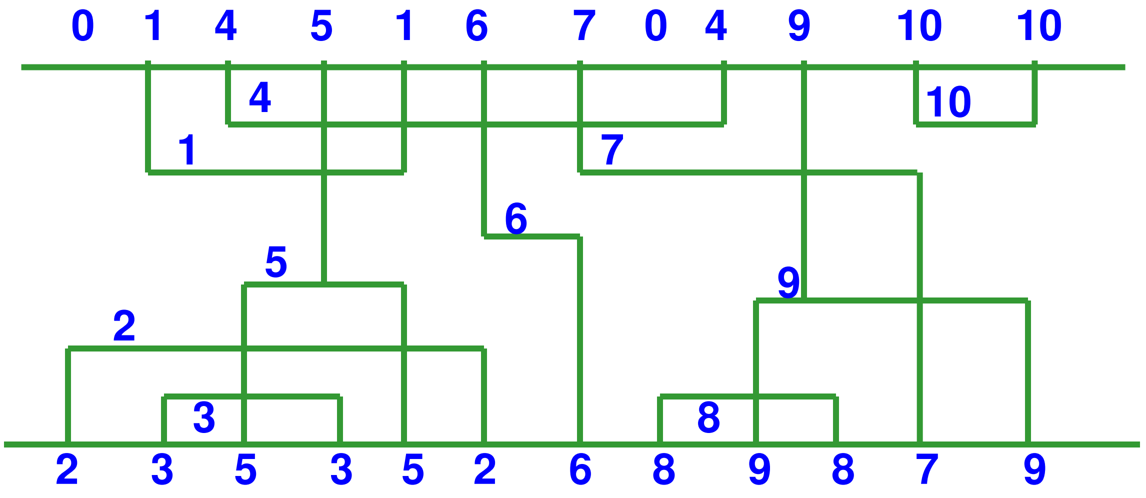
Doglegs break cycles in VCG

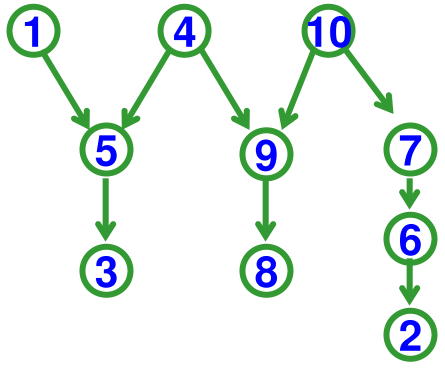




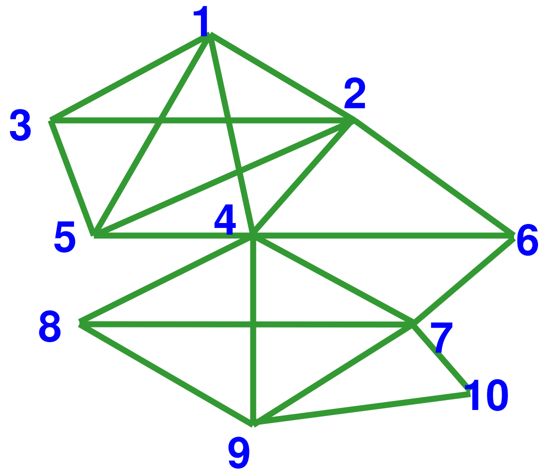
Courtesy K. Keutzer et al. UCB

Characterizing the Channel Routing Problem





Vertical constraint graph G_v



Horizontal constraint graph

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

Courtesy K. Keutzer et al. UCB

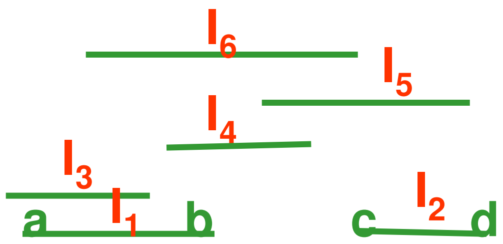
Interval Packing

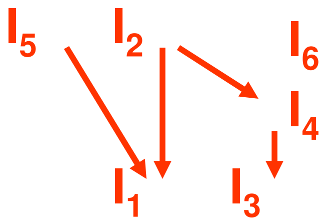
Theorem A set of intervals with density d can be packed into d tracks.

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

Define: $I_1<I_2$ iff $b<c$ or $I_1=I_2$

1. reflexive: $I_1<I_1$
2. anti-symmetric: $I_1<I_2$, $I_2<I_1$ → $I_1=I_2$
3. transitive: $I_1<I_2$, $I_2<I_3$ → $I_1<I_3$





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

Intervals in a single track→ form a chain

Intervals intersecting a common column → form an antichain

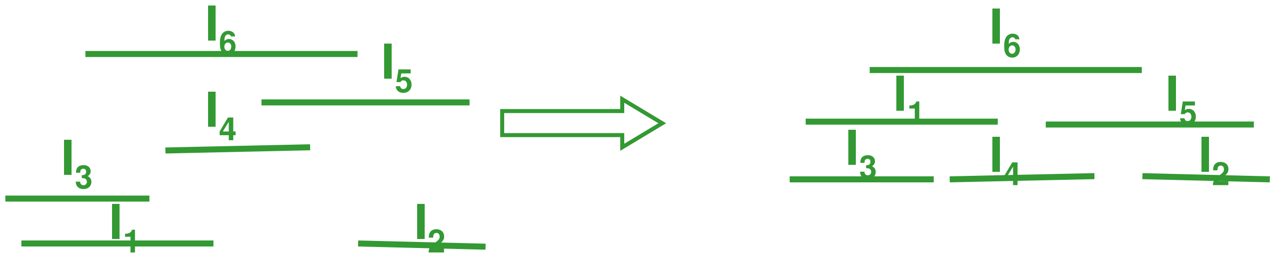
Dilworth's theorem (1950): If the maximum antichain of a POSET is of size d, then the POSET can be partitioned into d chains

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



O(nlogn) time algorithm. Greedy algorithm works!

Kahng/Keutzer/Newton

Courtesy K. Keutzer et al. UCB

Horizontal Constraint Graph (HCG)

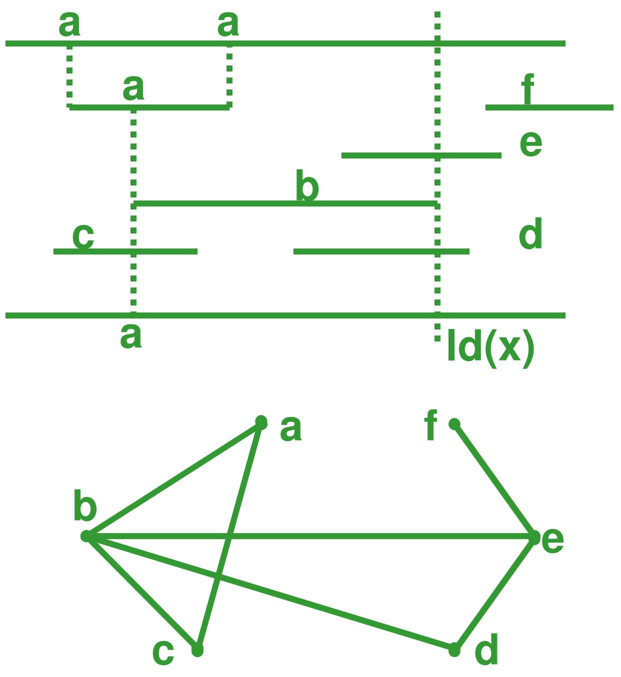
- ♋ Node v_i: represents a horizontal interval spanned by net I

- ♋ 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:

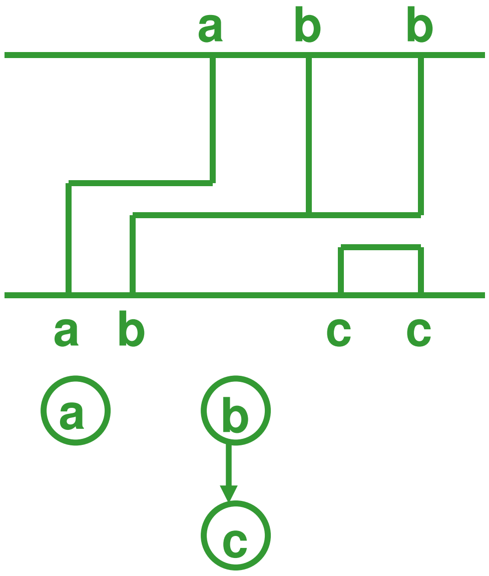
  # tracks ≥ maximum clique of HCG

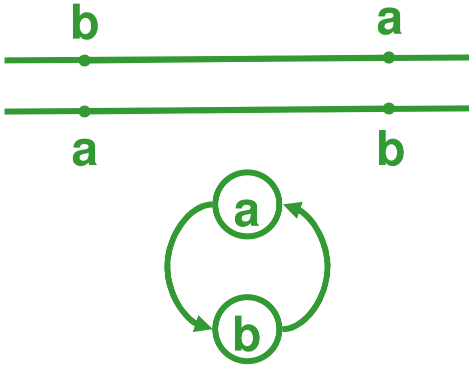


Vertical Constraint Graph (VCG)

- 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

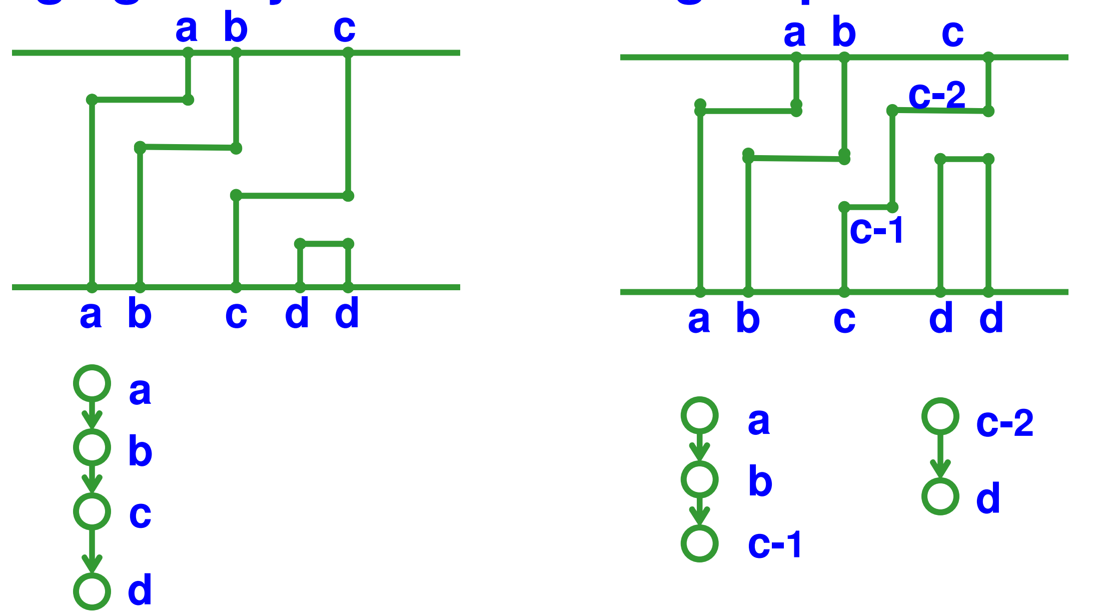
  a1→a2 means that net a1 has to be above a2
- Establishes a lower bound:
  # tracks ≥ longest path in VCG
- VCG may have a cycle !



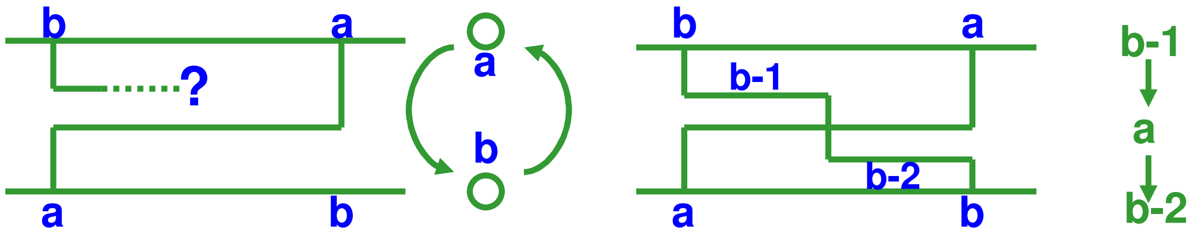


Doglegs in Channel Routing

- Doglegs may reduce the longest path in VCG

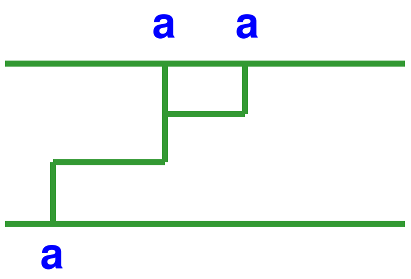


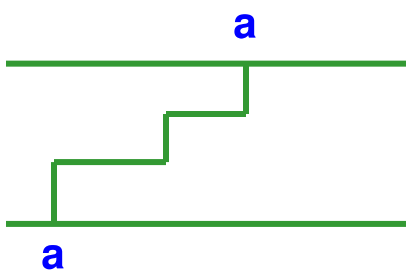
- Doglegs break cycles in VCG



Doglegs in Channel Routing(Cont’d)

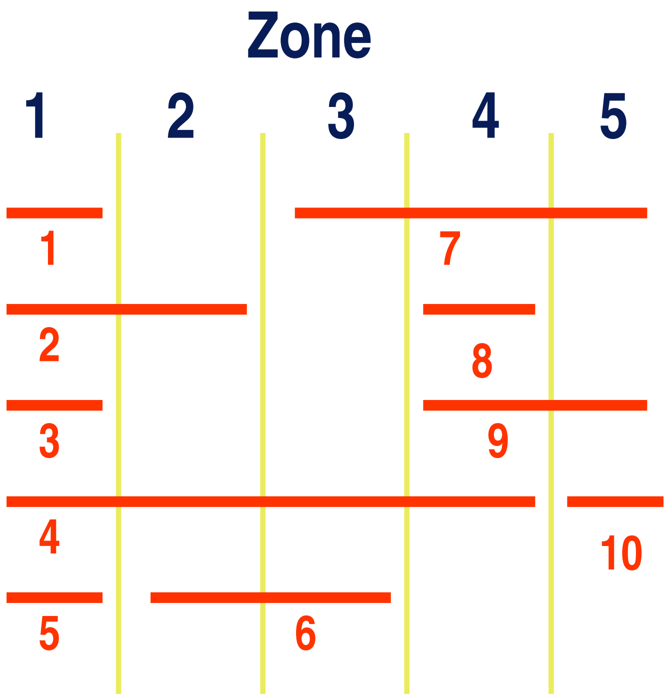
- Restricted Dogleg vs unrestricted dogleg

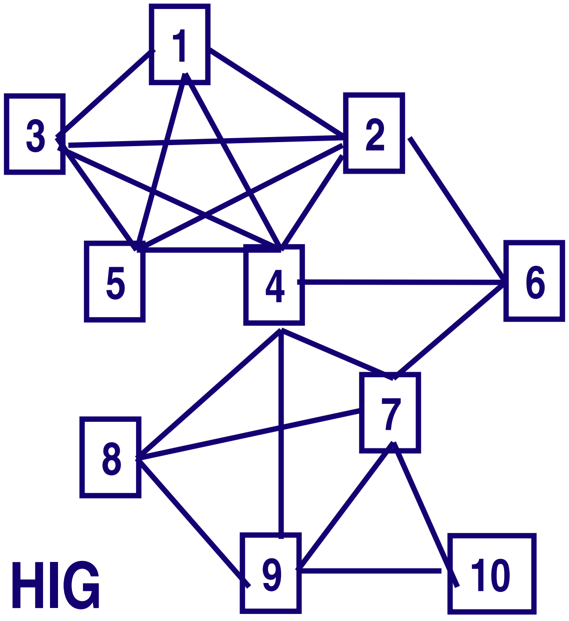




Constraint Graph Based Algorithm: "Merging of Nets" (Yoshimura & Kuh)

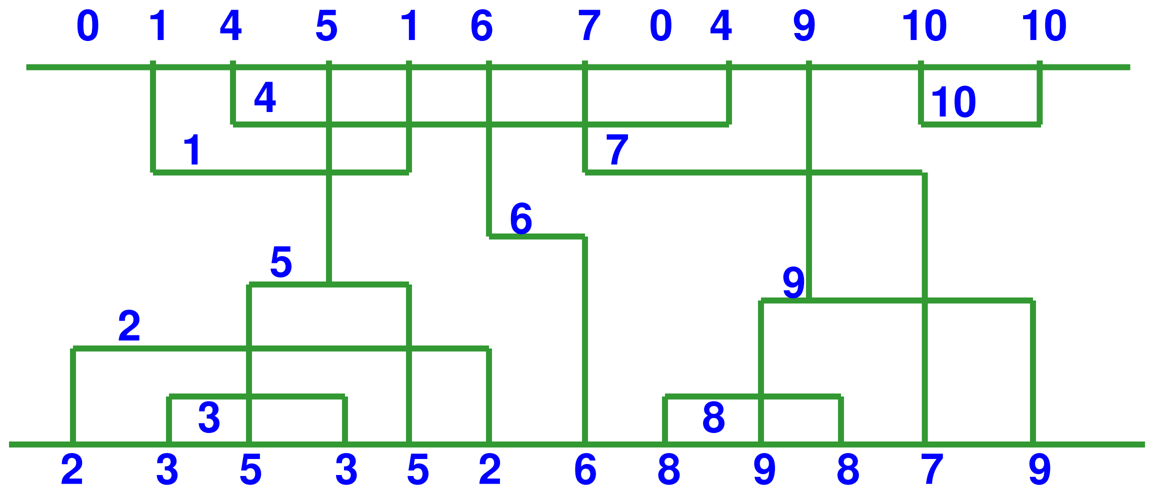
- On the assumption of no cyclic constraints, nets that can be placed on the same track can be merged in the VCG, simplifying the VCG.
- Nets can be organized into zones, further simplifying the problem

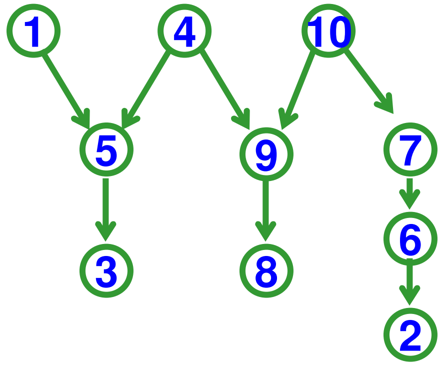




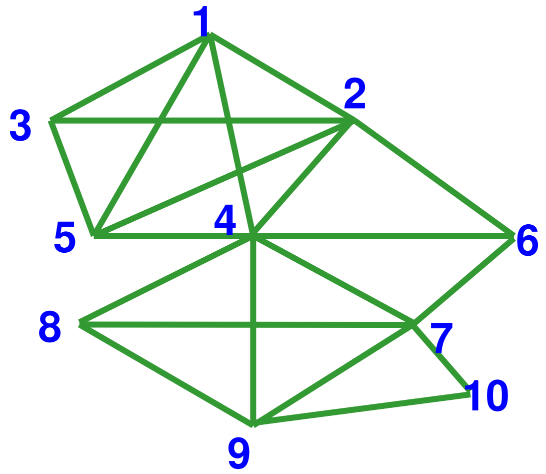
Kahng/Keutzer/Newton

Characterizing Channel Routing Problem





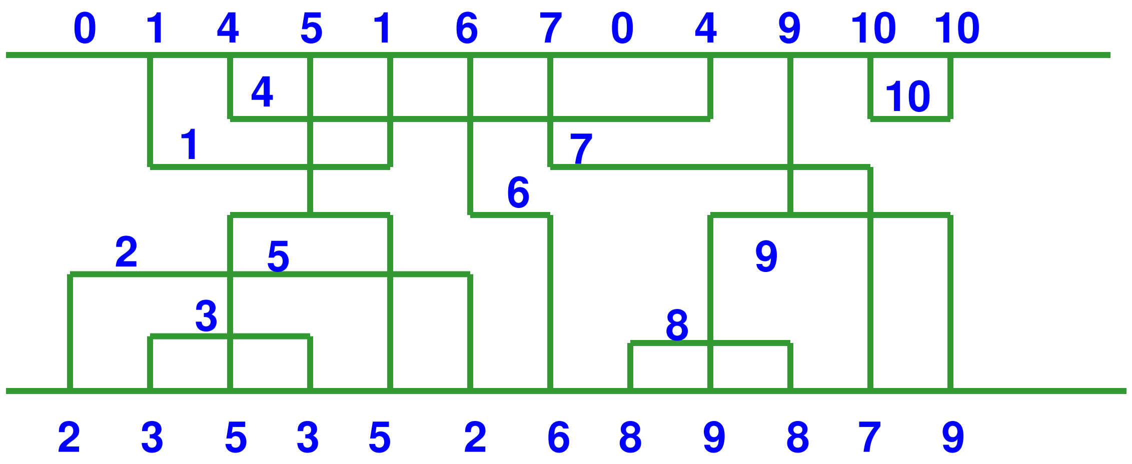
Vertical constraint graph G_v



Horizontal constraint graph

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

Zone Representation of Horizontal Segments



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

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

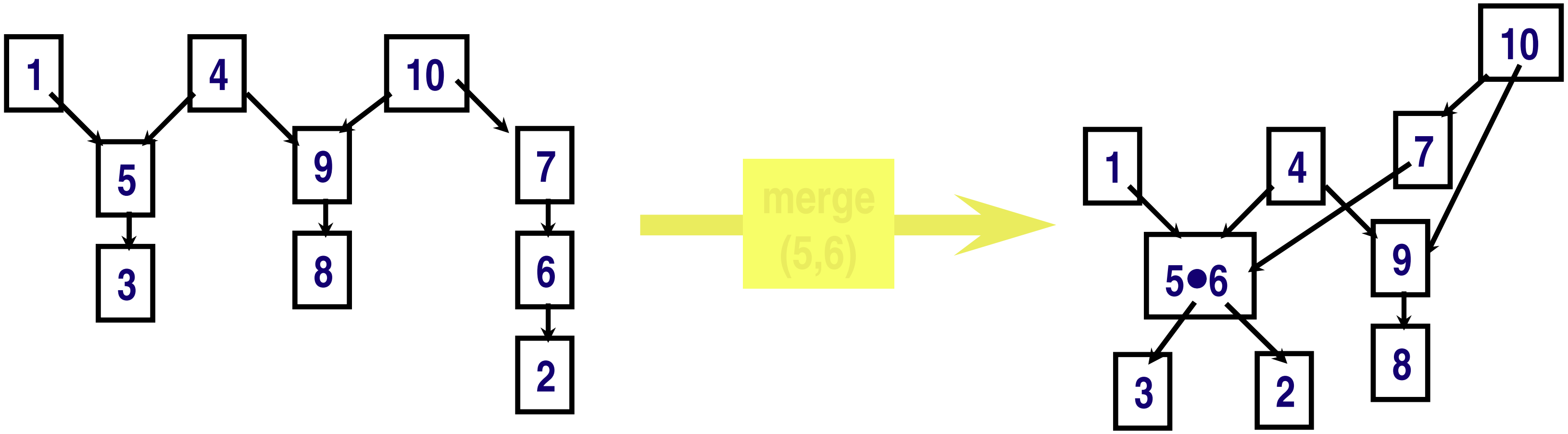
  Then these nets can be assigned to the same track and hence they can be merged in the VCG

- Merging Operation:

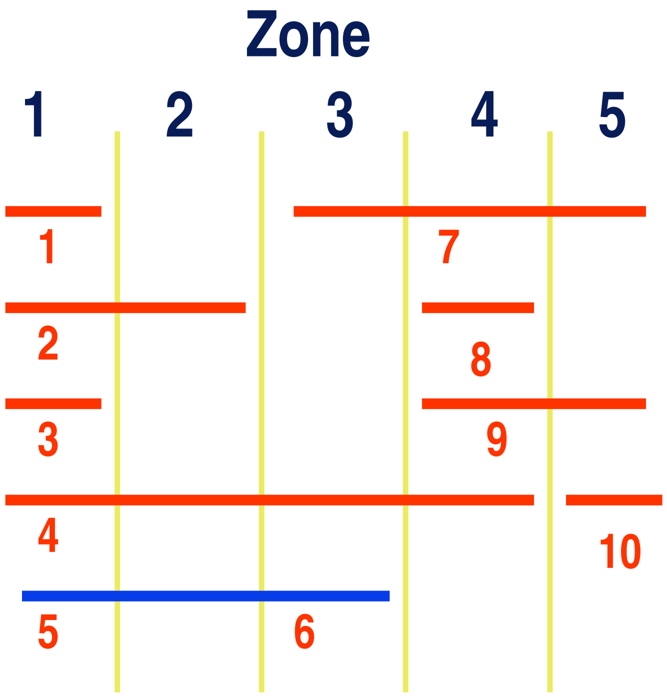
  (1) Combine nodes *i* and *j* into node *i*• j in VCG

  (2) Update zone representation such that *i*•*j* occupies zones of *i* and *j*

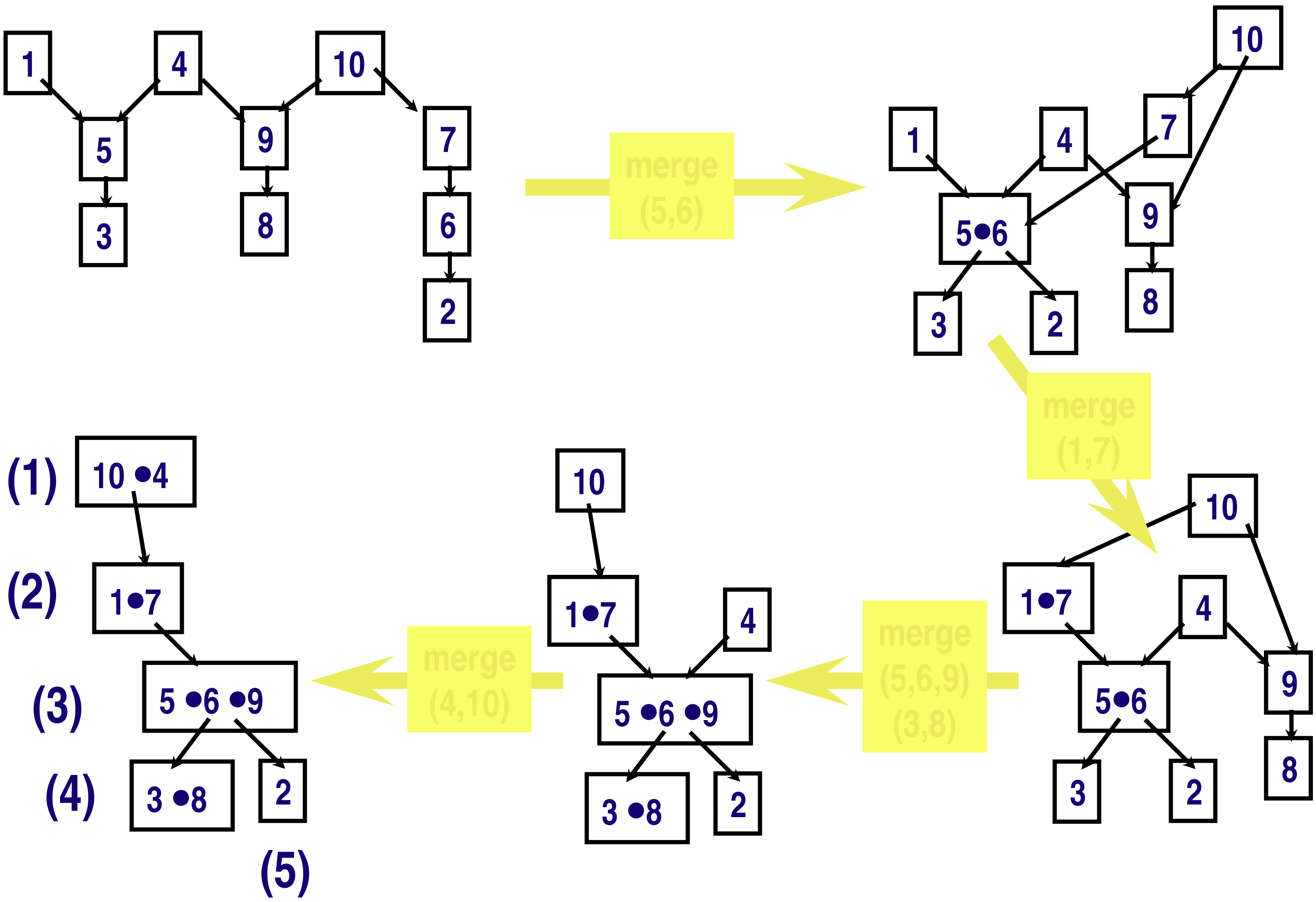
Merging of Nets: Example



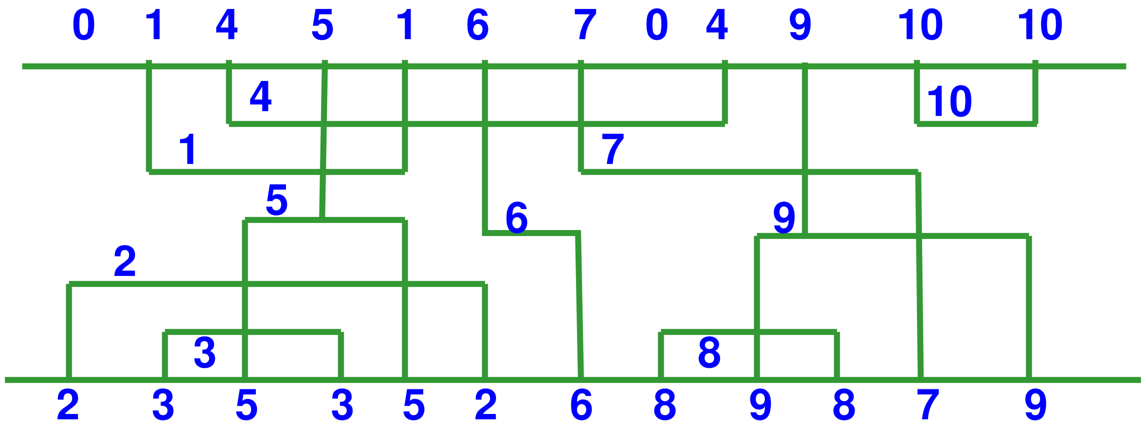
Updating of Zone Representation



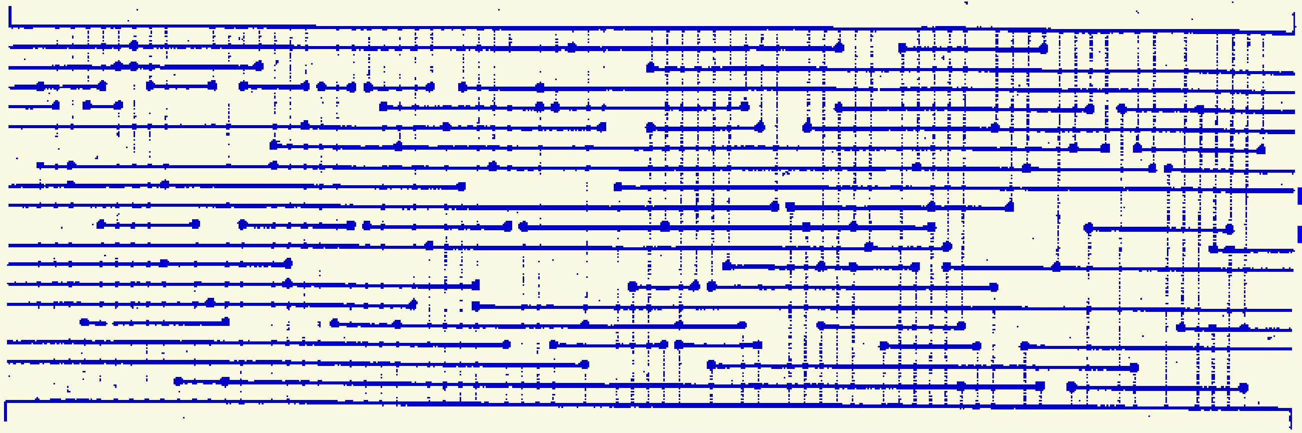
Merging of Nets: Example



Final Routing

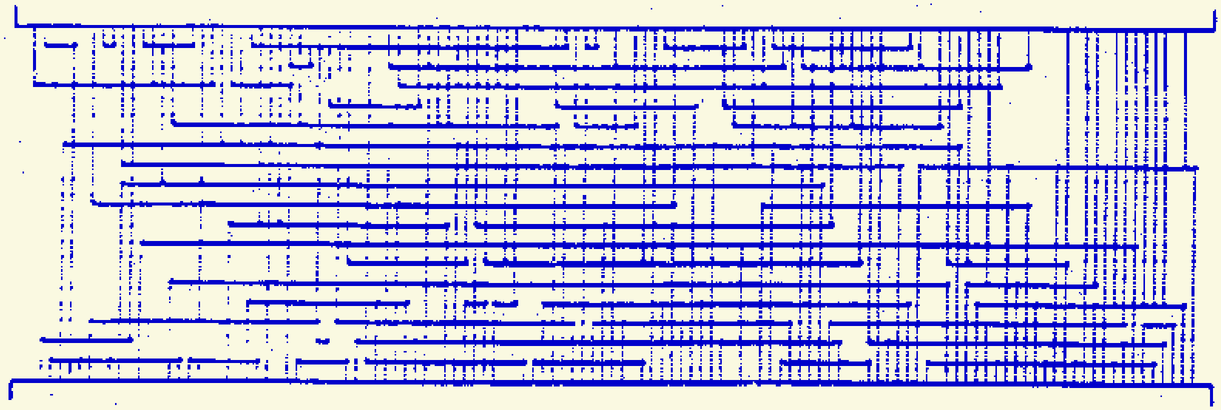


Routing Examples by Y-K's Algorithm



number of tracks=18
maximum density =18

Example 3c

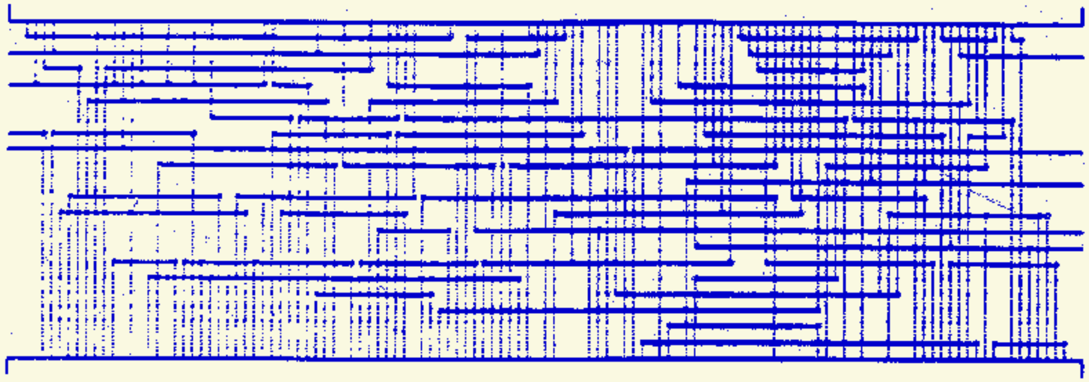


number of tracks=17

maximum density =17

Example 4b

Routing Examples by Y-K's Algorithm (Cont'd)

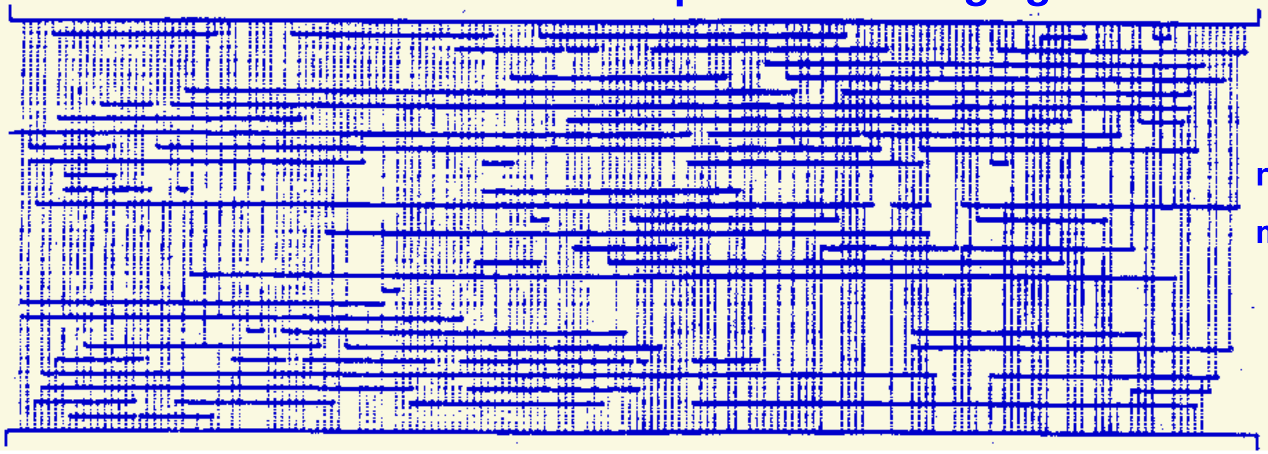


number of tracks=20

maximum density =20

Example 5

Deutsch's Difficult example without dogleg



number of tracks=28

maximum density =19

Yoshimura and Kuh’s Method

Source:

“Efficient Algorithms for Channel Routing”

by T. Yoshimura and E. Kuh

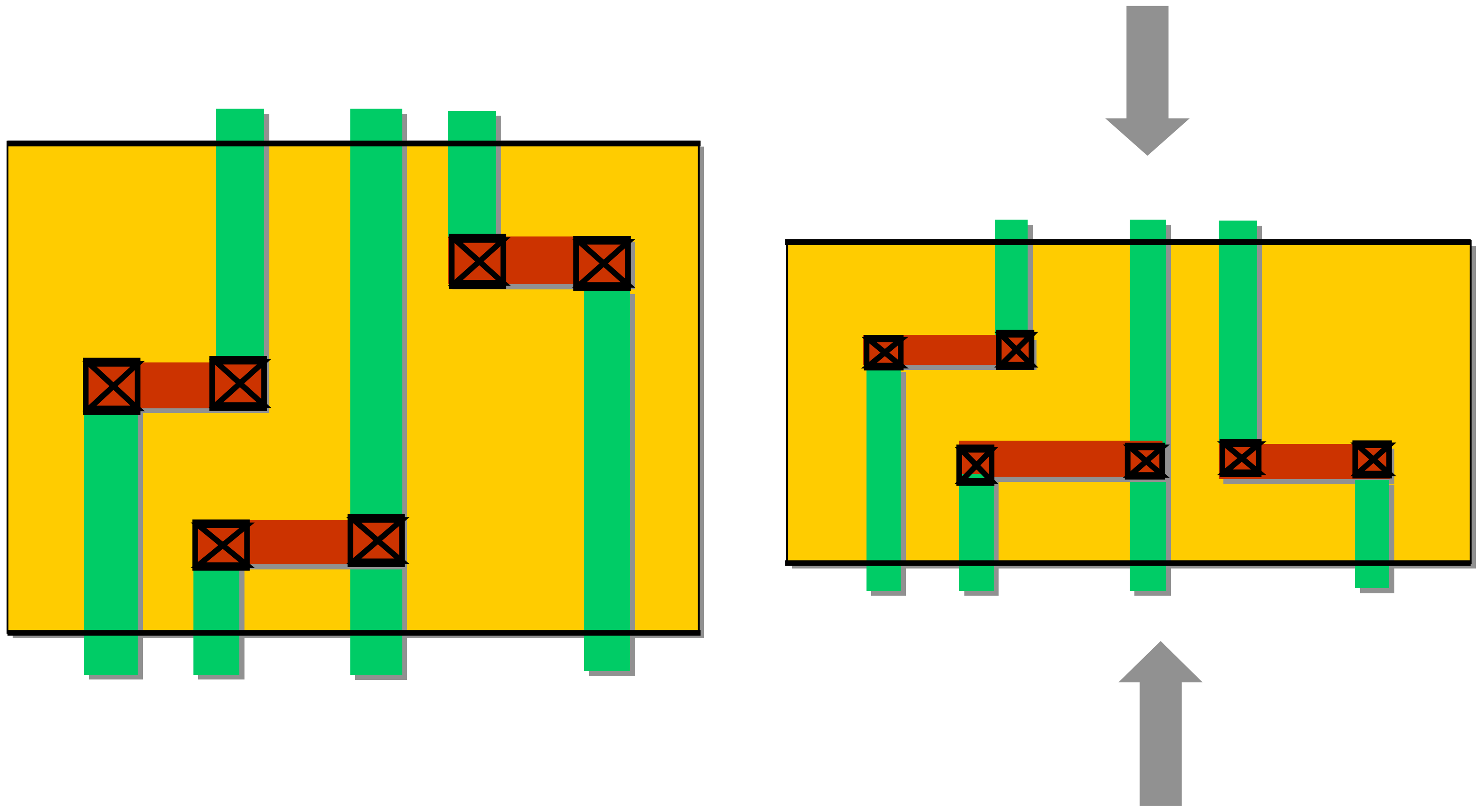
IEEE Trans. On Computer-Aided Design of Integrated Circuits and Systems.

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

Compaction

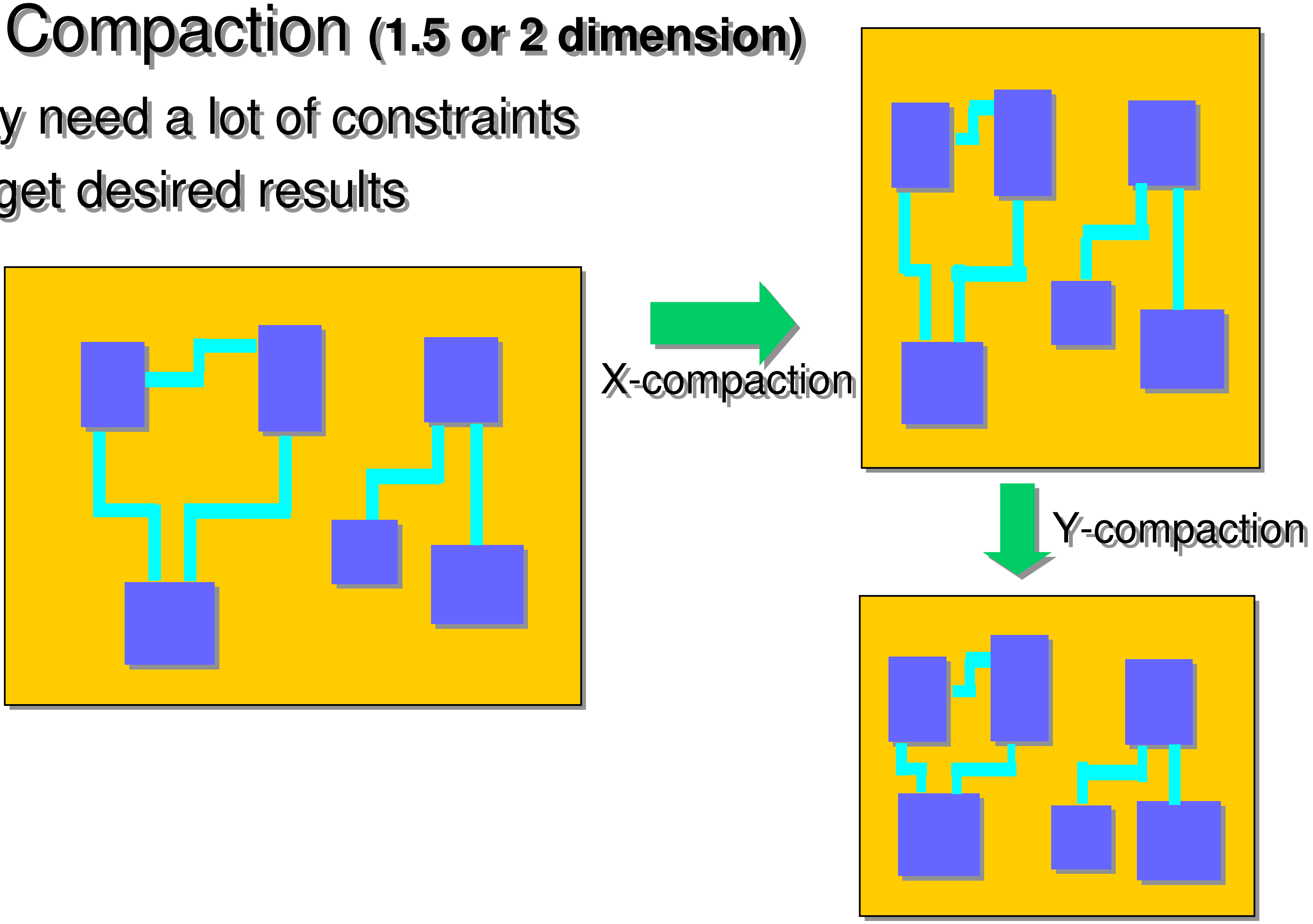
Compaction

- **Channel Compaction** (one dimension)



Compaction

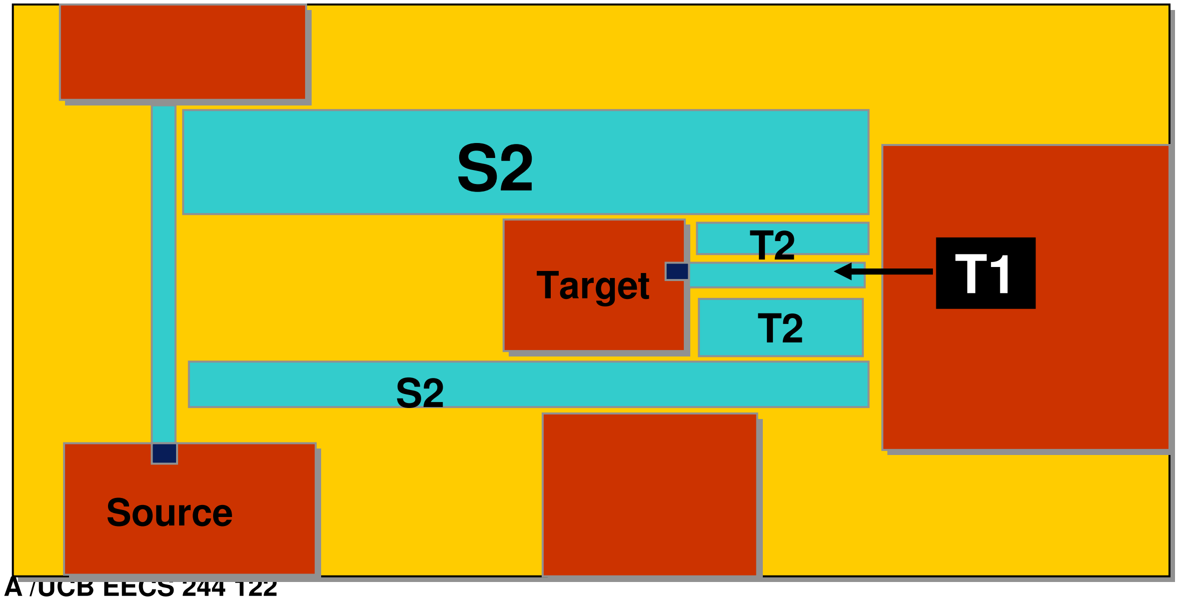
- Area Compaction **(1.5 or 2 dimension)**
  - May need a lot of constraints to get desired results



Kahng/Keutzer/Newton

Shape-based Routing

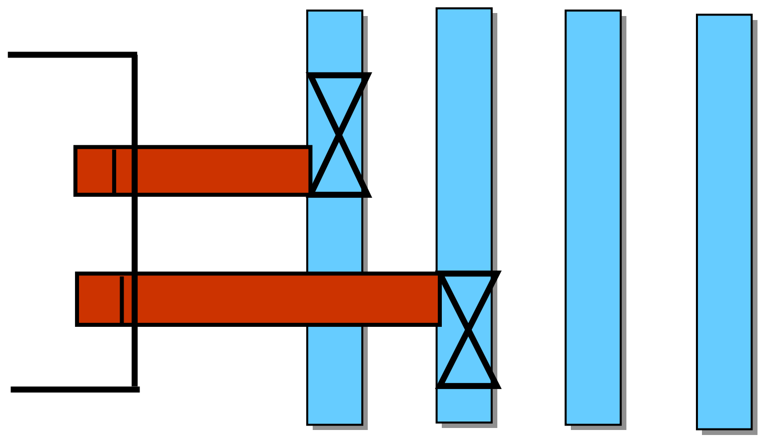
- Evolve from maze routing
- Gridless: look at actual size of each shape
- Each shape may have its spacing rule
- Good for designs with multiple width/spacing rules and other complex rules
- Slower than gridded router



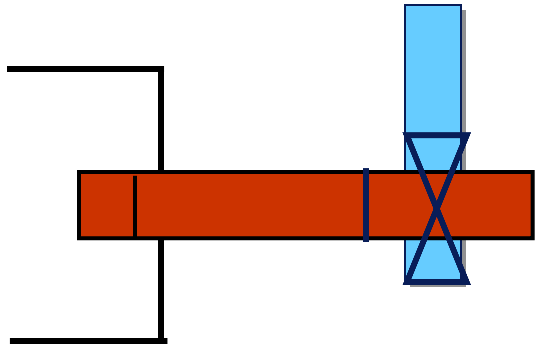
Other Physical Issues

Detailed Routing Objectives

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



Rotate and offset horizontal vias

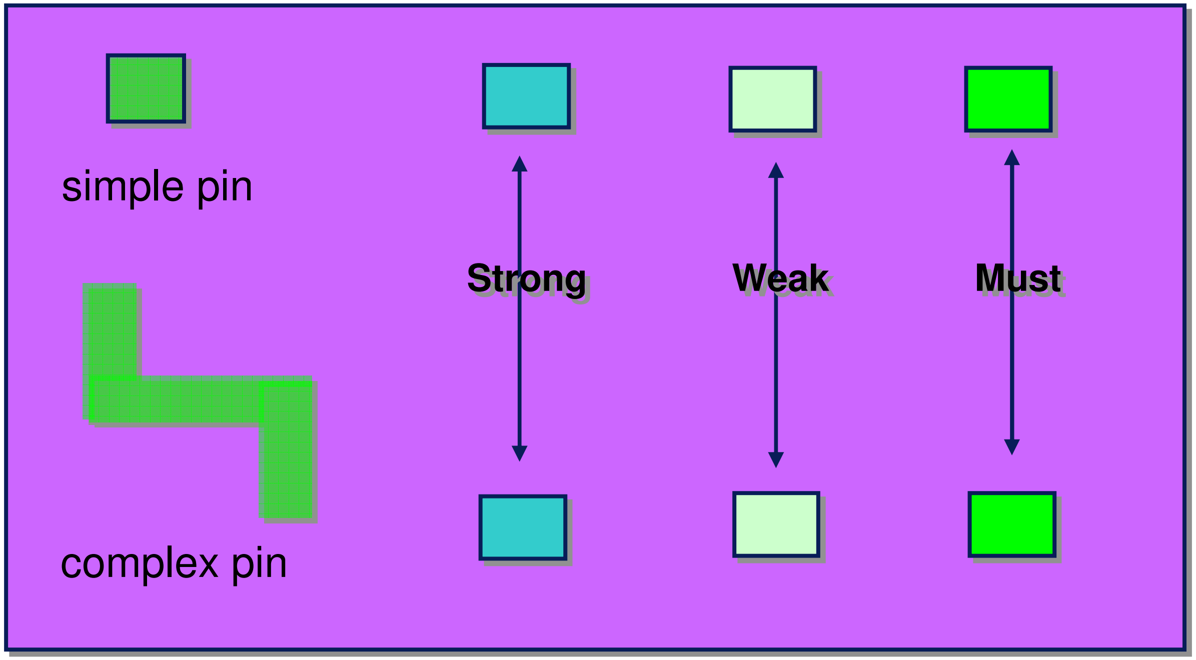


No rotation for a “cross” via

Kahng/Keutzer/Newton

Detailed Routing Objectives

- Understand complex pin & equivalent pin modeling



Extra extra

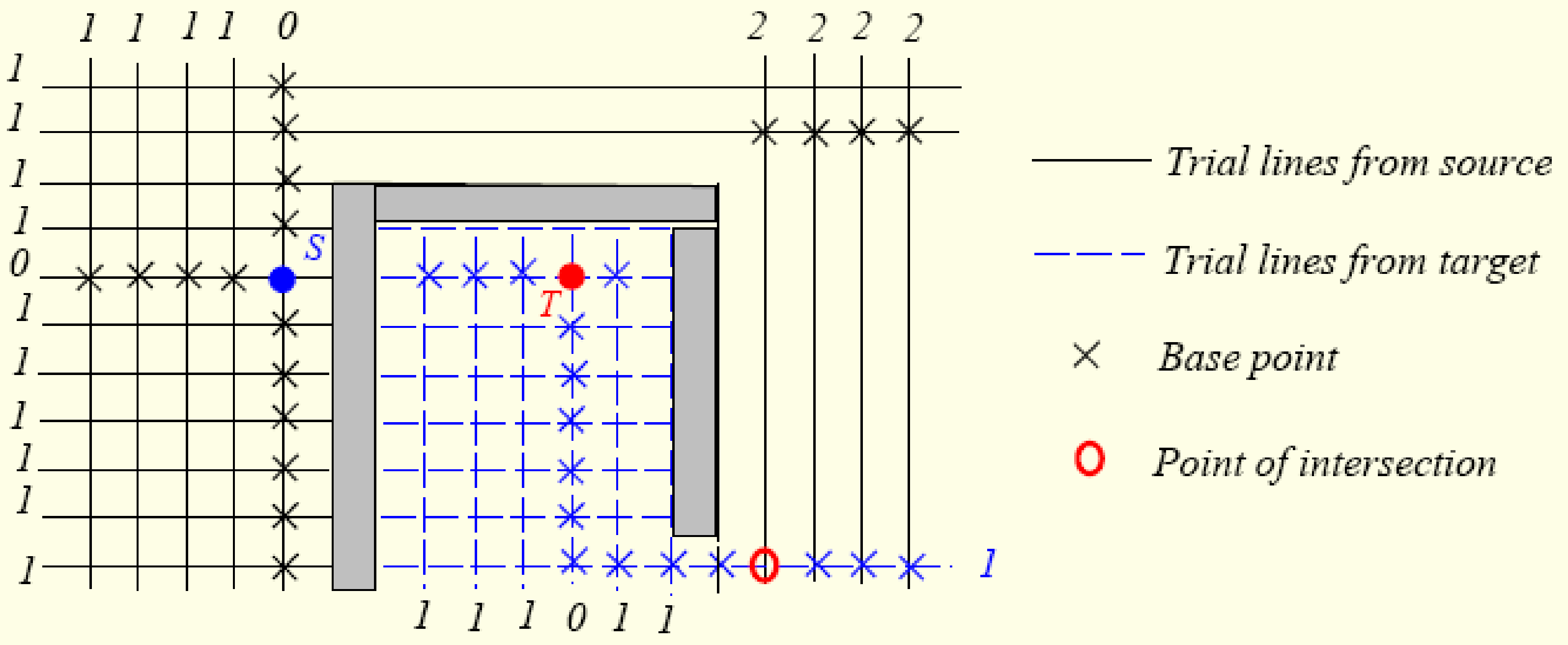
Maze Routing

- 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 = Σ (edge overflows) over all edges

Courtesy K. Keutzer et al. UCB

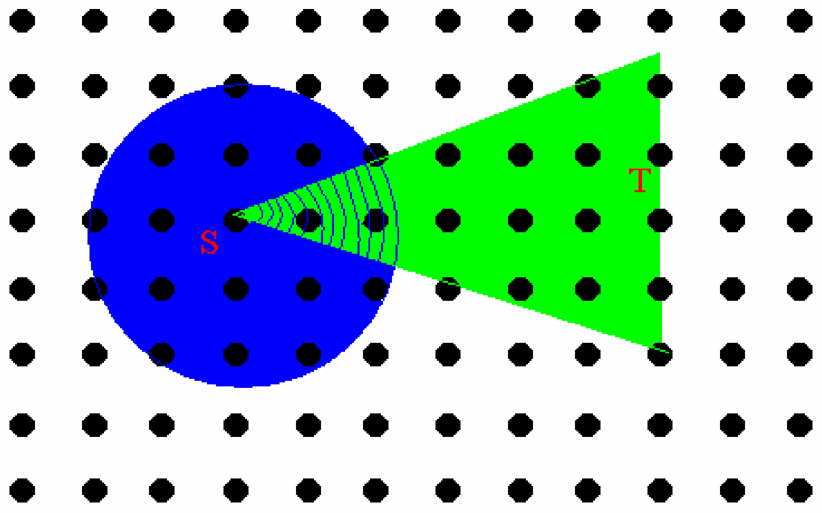
Mikami & Tabuchi’s Algorithm

- Mikami & Tabuchi, “A computer program for optimal routing of printed circuit connectors,” IFIP, H47, 1968.
- Every grid point is an escape point.



Maze Routing Cost Function and Directed Search

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



S denotes the source point
T is the sink point

Directed search limits the search space when all other cost variables are equal

Non directed search expands in a circular fashion from S

Directed search expands in a conical fashion from S to T