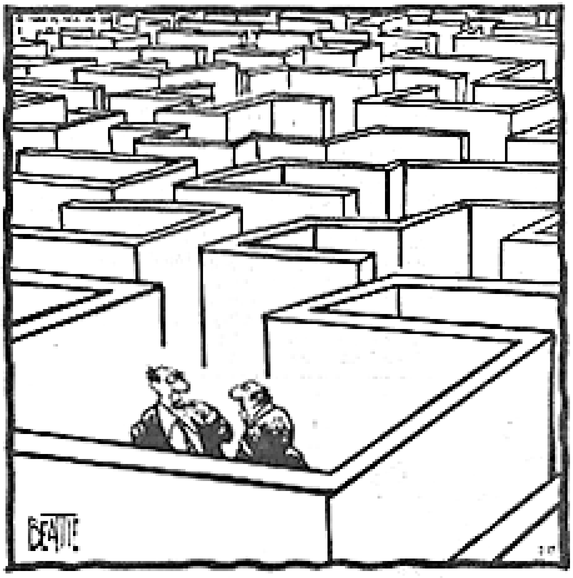
Chapter 9
Routing

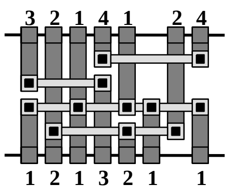


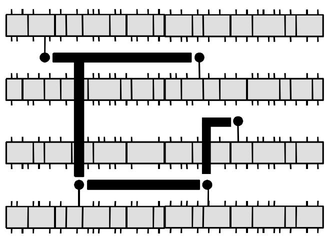
"The exit? Sure...take a right, then left, left again...no, wait...a *right*, then...no, wait...."

Local Routing

Local routing is the process of determining the exact patterns that interconnect sets of terminals in a given routing area.

Local routing is opposed to global routing, the process of determining through which routing areas a connection will run without fixing the wiring patterns within the routing areas.





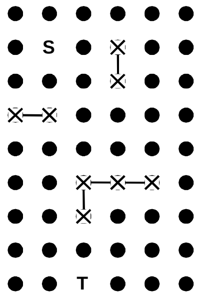
Characterization

Local routing is characterized by a number of parameters (each parameter setting defines a distinct problem type):

- the number of wiring layers,
- the orientation of wire segments in a layer: horizontal, vertical, diagonal or some combination of these,
- grided or gridless routing,
- presence or absence of obstacles in routing area,
- stretchable or fixed routing area,
- the constraints on the positions of the terminals: two parallel lines, along a rectangle, arbitrarily in an area, etc.
- terminals with a fixed or floating position.

The Lee Maze Routing Algorithm

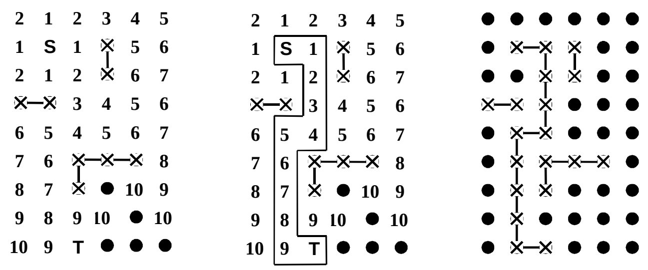
- The Lee algorithm is a classical routing technique; it is the basis of many routing programs.
- The one-layer version is presented.
- The routing area is a grid; terminals can be located anywhere in the area.
- The algorithm looks for connections in the presence of obstacles.



(a)

The Lee Algorithm (Continued)

A two-point connection is realized by propagating a wave front the source terminal outwards until the target terminal is reached and using backtracing for finding the shortest connection.



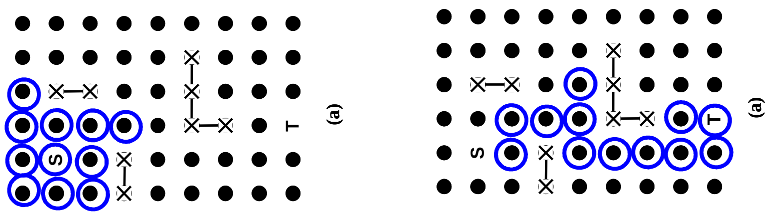
(b) (c) (d)

The Lee Algorithm (Continued)

- How to handle multi-terminal nets?
- Multi- terminal nets: first two terminals are connected, this connection is the target for the wave propagation from the third terminal, etc.
- A routed net is an obstacle for the next nets.
- The algorithm always finds a connection if a connection exists.
- For two- terminal nets, this connection is the shortest possible; for multi- terminal nets optimality is not guaranteed (remember: the “minimal rectilinear Steiner tree” problem is NP-complete).
- It can be generalized for multiple layers: wave front expansion in three dimensions.
- Its time complexity and space complexity is $O(n^2)$.
- The quality of its result strongly depends on the ordering of the nets.

Lee Routing Complexity

- Q: can we make it faster?
- A: Yes
- Q: How?
- A: A*
- Try to minimize the number of grid cells visited
- Informed search: try to search the most likely direction first



http://en.wikipedia.org/wiki/A*_search_algorithm

Channel Routing

Channel routing is characterized by:

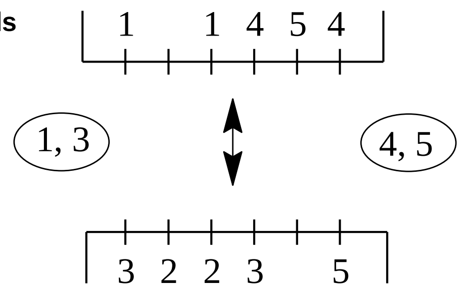
- a rectangular routing area;
- the top and bottom rectangle boundaries contain terminals with fixed positions;
- the left and right boundaries of the channel have terminals with floating positions;
- the goal is to minimize the height of the routing area.

Channel routing is not anymore mainstream

Replaced by area routing (we have much more metal levels now)

Remains good illustration of EDA way of thinking

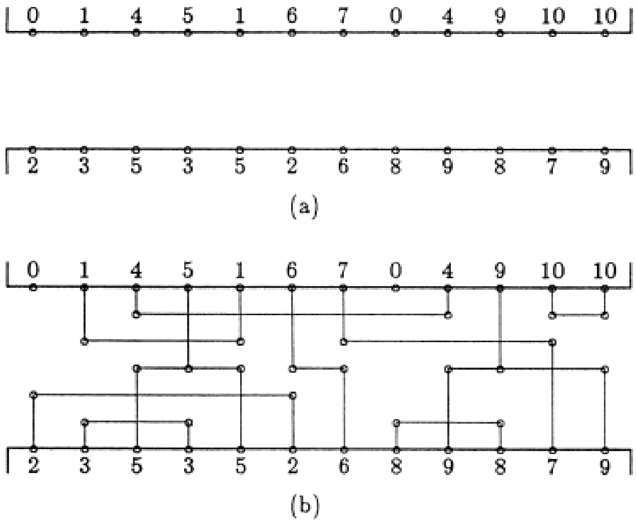
Related algorithms still relevant for register allocation



http://en.wikipedia.org/wiki/Channel_router

Channel Routing Example

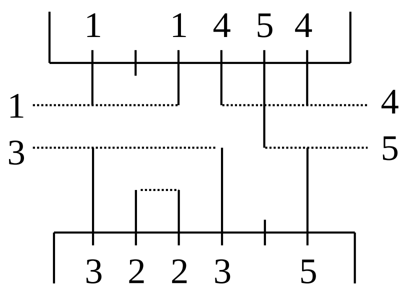
From Wong et al. Simulated Annealing for VLSI Design, p. 76, Kluwer Academic, 1988.



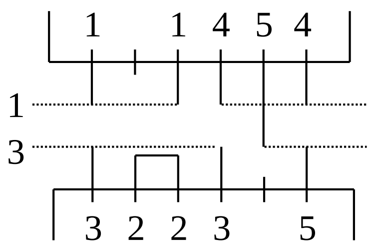
ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs 10/7/2010 9

Reserved and Non-Reserved Layers

Solution in a reserved-layer model



Solution in a nonreserved-layer model

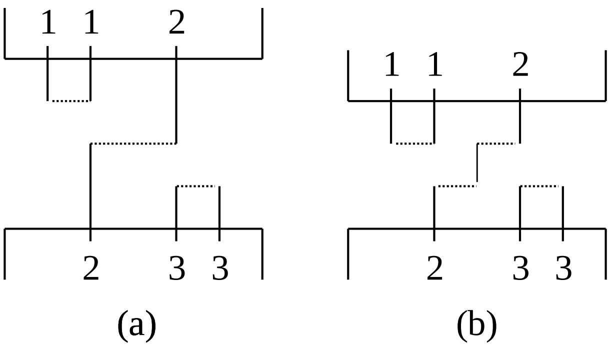


ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs 10/7/2010 10

Doglegging

Some routing models assume that each net only uses a single horizontal segment.

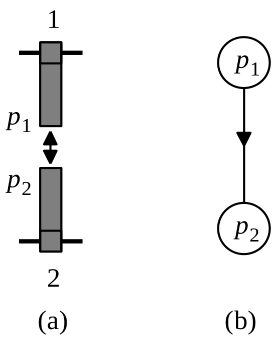
The use of multiple horizontal segments, doglegs, per net can lead to better results (but larger search space).



Vertical Constraints

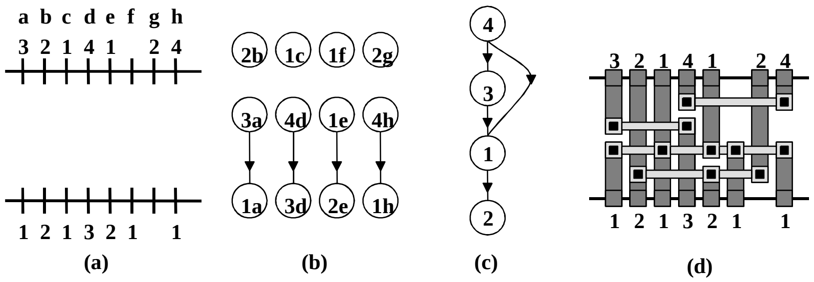
Two terminals located above each other give rise to a vertical constraint: the vertical segment connected to the top terminal cannot overlap with the vertical segment connected to the bottom terminal.

- p_1 above p_2
- (horizontal) track for p1 should be above the one for p_2



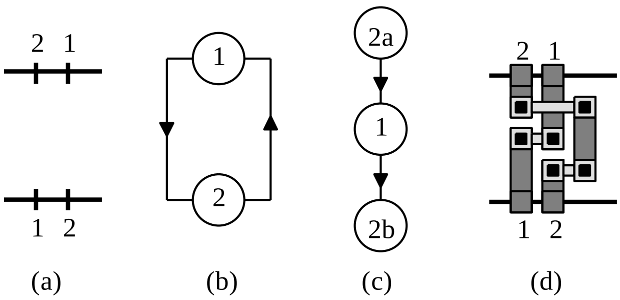
Vertical Constraints (Continued)

Vertical constraints can be combined into a vertical constraint graph under the assumption that each net will use one horizontal segment.



Vertical Constraints (Continued)

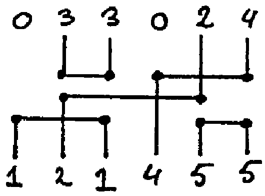
Cyclic constraints must be resolved by splitting horizontal segments.

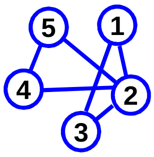


Horizontal Constraints

The horizontal segments of different nets cannot be located on the same track. This is called a horizontal constraint.

The combination of horizontal and vertical constraints makes the channel routing problem difficult.





- **Vertices corresponding to overlapping intervals are connected by an edge.**

Channel Density

- **Local density** $d(x)$**: the number of nets that cross the channel at position** x**.**
- **Channel density** $d_{max} : d_{max} = \max_x d(x)$.
- **The channel density is a lower bound on the number of rows (provided that all horizontal connections are made in the same layer – reserved layer model).**

The Left-Edge Algorithm

Channel routing problems without vertical constraints can be solved efficiently by the left-edge algorithm.

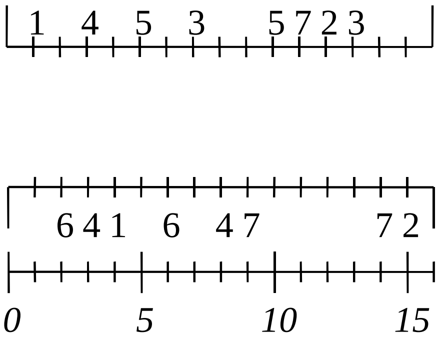
Problem: given a set of segments (intervals) $[x_{i_{\min}}, x_{i_{\max}}]$, put non-overlapping segments on the same track such that the number of tracks is minimal.

Solution: sort the intervals by their first coordinate and fill the tracks by the first fitting interval.

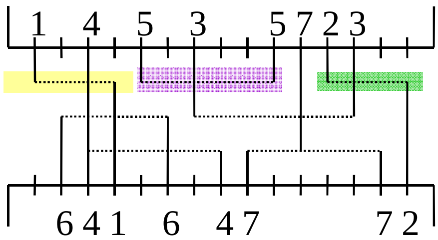
Remark: the simple left-edge algorithm solves the problem (i.e. the problem without VC's) optimally (in polynomial time).

The Left-Edge Algorithm (Ctd.)

An example problem:



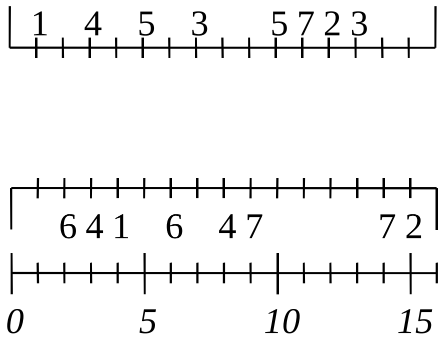
Problem solution:



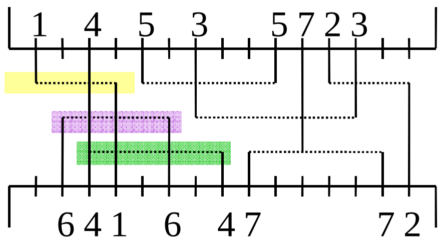
step	net	interval
1	1	[1,4]
4	6	[2,6]
6	4	[3,8]
2	5	[5,10]
5	3	[7,13]
7	7	[9,14]
3	2	[12,15]

The Left-Edge Algorithm (Ctd.)

An example problem:



Problem solution:

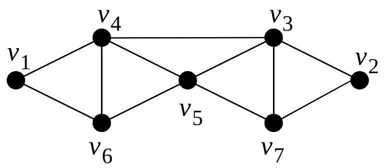


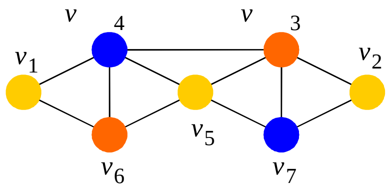
step	net	interval
1	1	[1,4]
2	6	[2,6]
3	4	[3,8]
4	5	[5,10]
5	3	[7,13]
6	7	[9,14]
7	2	[12,15]

Interval Graphs

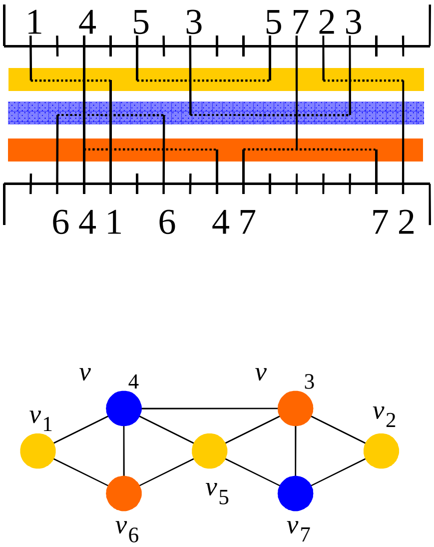
- Horizontal constraint graph is an interval graph
- Vertices corresponding to overlapping intervals are connected by an edge.
- Solving the track assignment problem is equivalent to finding a minimal vertex coloring of the graph.

net	interval
1	[1,4]
2	[12,15]
3	[7,13]
4	[3,8]
5	[5,10]
6	[2,6]
7	[9,14]





Track Assignment = Vertex Coloring



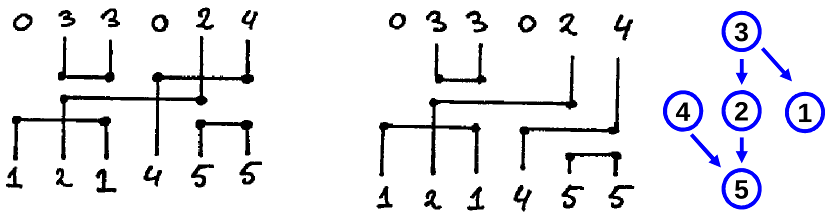
http://en.wikipedia.org/wiki/Vertex_colouring

Channel Routing Algorithms

- The channel routing problem (with VCs) is NP-hard.
- Heuristic: left-edge algorithm while trying to satisfy the vertical constraints.
- The Lee algorithm will not perform very well if used directly (there is too much freedom initially; wrong decisions are taken, blocking future connections). However, it can be used within an iterative algorithm or as a postprocessing step for other algorithms.
- There are many algorithms based on a multitude of heuristics.

Left Edge with VC's

- Optimal assignment of nets to tracks also depends on VC's
- In general, many choices matching VCG, only some (or one) of them is optimal
- Optimization is NP-hard
- Solution: Exhaustive search – not really
- Exhaustive search + branch and bound works better
- Use length of longest path in VC as lower bound for number of tracks



3 tracks (optimal) 4 tracks (sub-optimal)

Left Edge with Branch and Bound

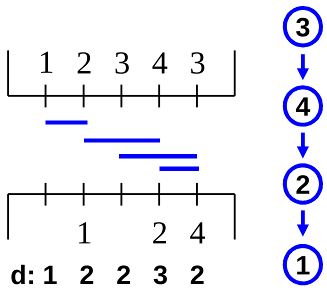
- Say, we have some solution having width *w*. From then on, we only search for better solutions. Set *w=ub*.
- Consider net *n* on track *t* during some step in algorithm.
- Assume longest directed path in VCG from *n* equals *l*.
- Width w would become at least *t+l*.
- Continue with current partial solution when $t+l<ub$
- Backtrack when $t+l \geq ub$.

Yoeli's Robust Router

- In principle, uses single horizontal segments for all nets.
- Fills tracks like the left-edge algorithm, but in a more sophisticated way.
- Alternates between top and bottom tracks until center is reached. The side on which the algorithm is working at the moment is called the current side.

Example of 'way of thinking' in developing heuristics

Yoeli's Net Weights

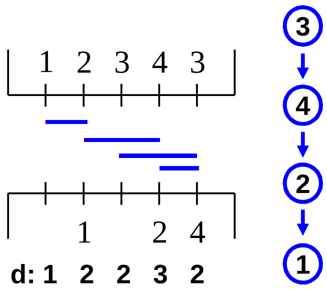


- Intervals, assign nets to tracks
- Q: which net should be in top-most track?
- A: net 3
- In general: not exclusively due to VCs
- Q: How to 'algorithmize' this?
- A: net weights

The choice for putting some segment in a track is based on net weights. The weights:

- favor nets that contribute to the channel density;
- favor nets with terminals at the current side;
- penalize nets whose routing at the current side would cause vertical constraint violations.

Weight Calculation (iteration 1)

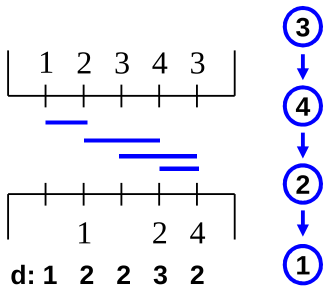


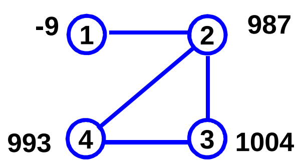
net (i)	(1)	(2)	(3)	w_i
1	0	1	-5x2	-9
2	1000	2	-5x3	987
3	1000	2+2	0	1004
4	1000	3	-5x2	993

1. Add large number *B* to weight of net that intersects max d column *[B=1000]*.
2. Add the local column density d to weight of net for each terminal at the current side *[top]*
3. Subtract *K·d* for each column with a VC violation *[K=5]*

ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs *10/7/2010* *27*

Max Weight Independent Set

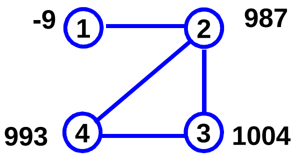




- Only one of 2, 3, 4 can go in top track
- Also 1 can go in top track
- Q: Which should go in top track?
- A: Only #3
- Q: Algorithm?
- A: Max weight independent set

ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs *10/7/2010* *28*

Max Weight Independent Set



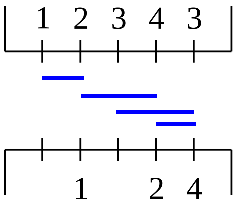
weight: how favorable to put net in topmost row

No HC violations!

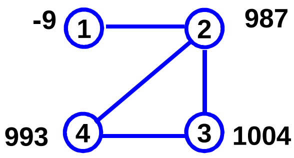
- **Independent set**: set of vertices w/o pairs of adjacent vertices
- Determine independent set with **maximum sum of weights**. Solution for this example?
- … v_3 → net 3 should go in top-most track
- Maximum weight independent set is NP-complete …
- … for general graphs
- Can be solved efficiently for interval graphs
- Dynamic programming

http://en.wikipedia.org/wiki/Independent_set_(graph_theory)

Dynamic Programming for MWIDS for Interval Graphs



- Scan from left to right
- Best solution in column i can be found by best solution in column i-1 and weights of nets ending in column i
- A net n with a rightmost terminal at position c is taken into the solution if:
  $w_n + total[x_{n\ min} - 1] > total[c - 1]$.
- Scan from r to l to determine solution

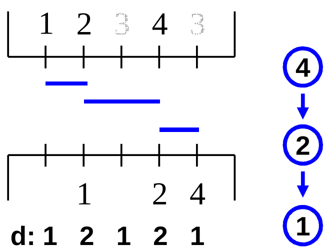


Column (i)	Total[i]	Selected[i]
1	0	0
2	Max(0, 0-9)=0	0
3	0	0
4	Max(0, 0+987)=987	2
5	Max(987,0+1004,0+983)=1004	3

Yoeli’s overall Flow

- Net 3 has been selected to go in the top row
- Do this, remove the net(s) for top row from the problem and solve a simpler problem for the bottom row
- Alternate between top and bottom row until done
- Resolve violations using rip-up and reroute (Lee Router)

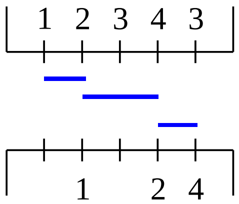
Weight Calculation (iteration 2)



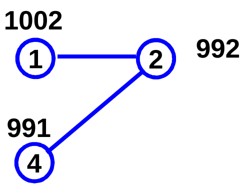
net (i)	(1)	(2)	(3)	w_i
1	1000	2	0	1002
2	1000	2	-5x2	992
4	1000	1	-5x2	991

1. Add large number *B* to weight of net that straddles max d column *[B=1000]*.
2. Add the local column density d to weight of net for terminal at the current side *[bottom]*
3. Subtract $K \cdot d$ for each VC violation *[K=5]*

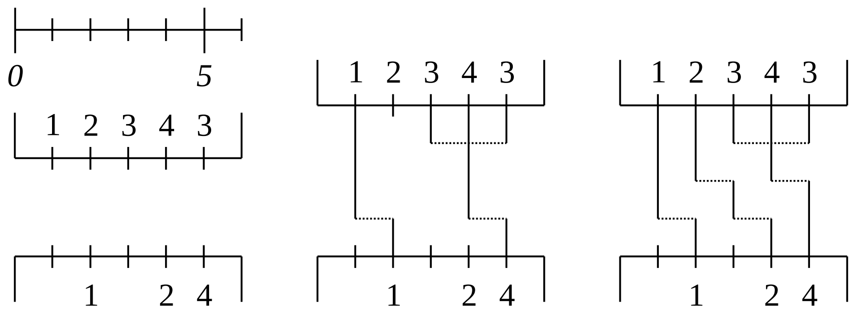
Dynamic Programming for MWIDS for Interval Graphs: Round 2



net (i)	(1)	(2)	(3)	w_i
1	1000	2	0	1002
2	1000	2	-5x2	992
4	1000	1	-5x2	991



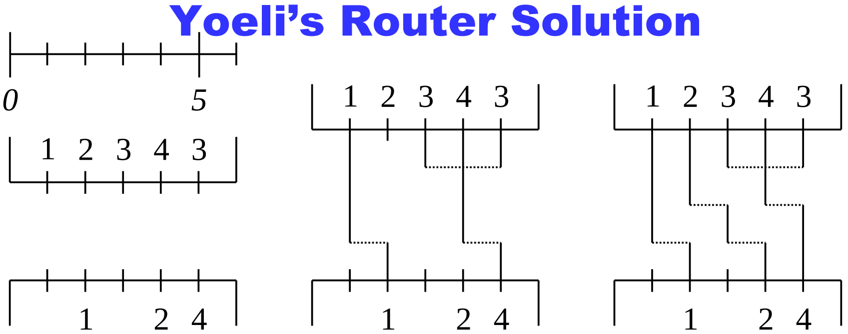
Yoeli's Router Solution

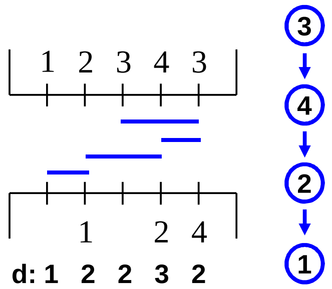


problem **After dyn. progr.** **After rip-up and reroute**

Net 2 can't be completed because of VC violation

Yoeli's Router Solution



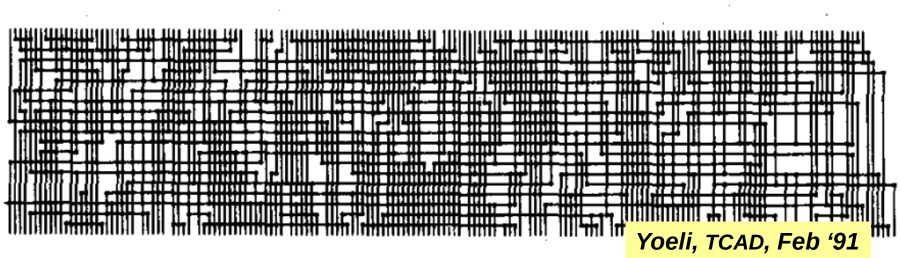


sub-opt solution satisfying VC from left-edge with B/B

better solution (3 tracks) from Yoeli router

ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs *10/7/2010* *35*

Deutsch' Difficult Example



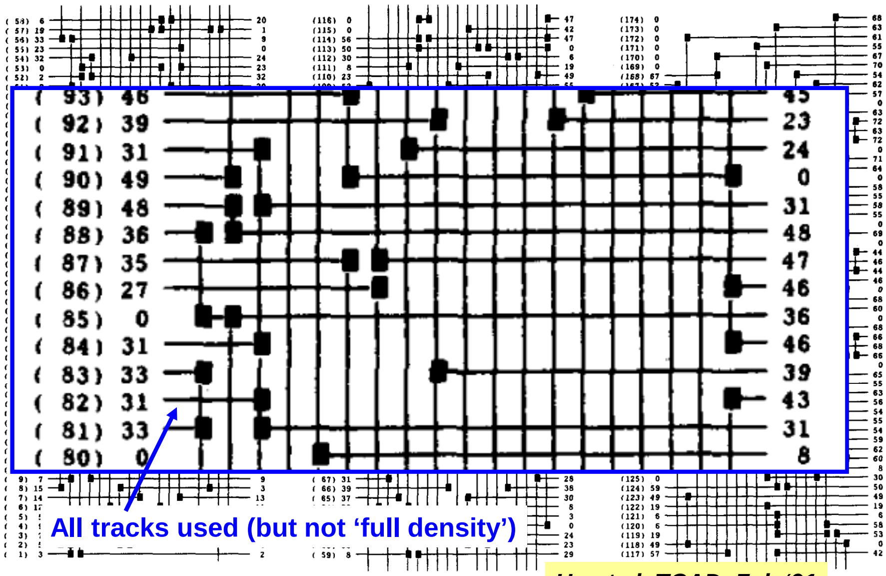
Yoeli, TCAD, Feb '91

- 'Standard' benchmark (but channel routing is not modern anymore)
  - 175 columns
  - Density: 19
  - Longest path in VCG: 28

Algorithm	# tracks	#vias, l
Left-edge with B&B	28	
Dogleg router	21	
Greedy router	20	
Hierarchical router	19	336, 5023
Yoeli's router	19	319, 4961
3-layer router	10	

ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs *10/7/2010* *36*

Deutsch Channel (Greedy Algorithm, d=19)



Ho et al, TCAD, Feb '91

Summary

- Routing is a very traditional EDA topic
- Has evolved quite a bit (a.o. driven by more layers, timing closure challenge, yield)
- But a lot of timing closure challenge hinges on placement and global routing
- Also for PCB, similar ideas but different details
- Good illustration of EDA way of thinking