EE 531: ADVANCED VLSI DESIGN

Routing

Nishith N. Chakraborty

February, 2025

IMAGINE THE FOLLOWING

- You are planning the transportation (i.e., roads & highways) for a new city the size of Brooklyn
- Many dwellings need direct roads that cannot be used by anyone else
- You can affect the layout of houses and neighborhoods but architects will complain
- Also, the time along any path can’t be longer than some fixed amount

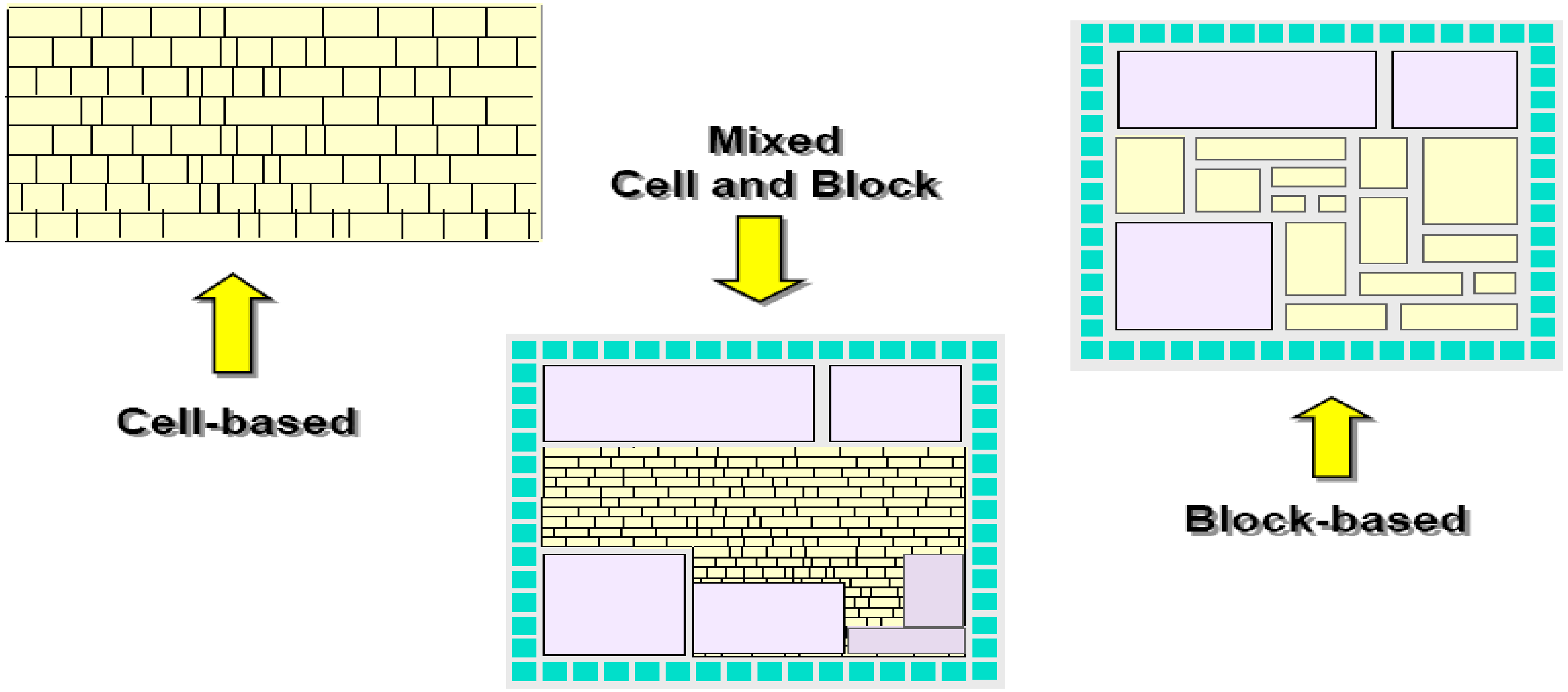
- What are the considerations?

SOME 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 a comparable complexity on every leading edge chip

ROUTING APPLICATIONS



ROUTING ALGORITHMS

- Hard to tackle high-level issues like congestion and wire-planning and low-level details simultaneously
- Global Routing
  - Identify routing resources to be used
  - Identify layers (and tracks) to be used
  - Assign particular nets to these resources
- Detail (Local) Routing
  - Define pin-to-pin connections
  - Must understand most or all design rules
  - May use a compactor to optimize results

ROUTING: THE PROBLEM

- Scale
  - Millions of wires
  - MUST connect them all
- Geometric Complexity
  - Basic starting point – grid representation.
  - But at nanoscale – Geometry rules are complex!
  - Also, many routing layers with different “costs”.
- Electrical Complexity
  - It’s not enough to just connect all the wires.
  - You also have to:
    - Ensure that the delays through the wires are small.
    - Ensure that wire-to-wire interactions (crosstalk) doesn’t mess up behavior.

PROBLEM DEFINITION

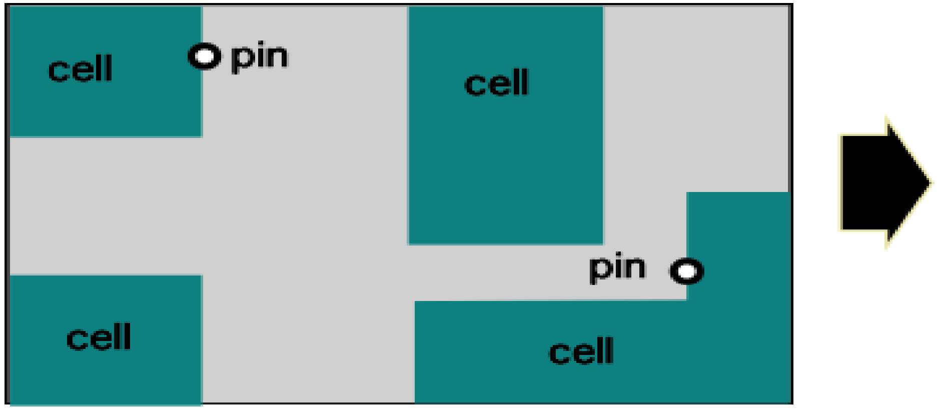
- Problem:
  - Given a placement, and a fixed number of metal layers, find a valid pattern of horizontal and vertical wires that connect the terminals of the nets.
- Input:
  - Cell locations, netlist
- Output:
  - Geometric layout of each net connecting various standard cells
- Two-step process
  - Global routing
  - Detailed routing

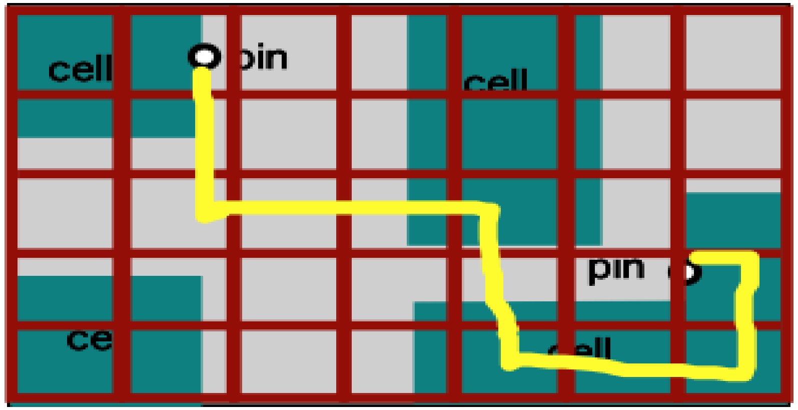
PROBLEM DEFINITION

- Objective
  - 100% connectivity of a system
  - Minimum area, wirelength
- Constraints
  - Number of routing layers
  - Design rules
  - Timing (delay)
  - Crosstalk
  - Process variations

DIVIDE AND CONQUER: GLOBAL ROUTING

- To deal with a big chip, we make our problem smaller
  - ➢ Divide the chip into big, coarse regions e.g., 200 X 200 tracks each.
  - ➢ These are called GBOXes.
- Now Maze Route through the GBOXes

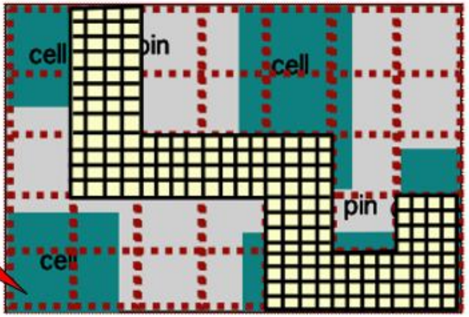




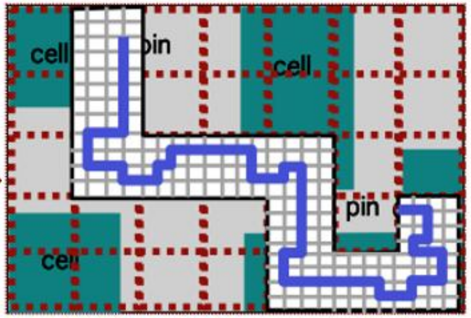
GLOBAL ROUTING

- Global routing takes care of basic congestion.
  - Balances supply vs. demand of routing resources.
  - Generates regions of confinement for the wires.
- Detailed routing decides on the exact path.

GBOXes have a *dynamic cost* according to how *congested* they are!



Global router tells us to search for detailed paths only here



Detailed router tells us exact final path in this region

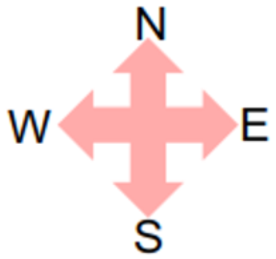
DETAILED ROUTING

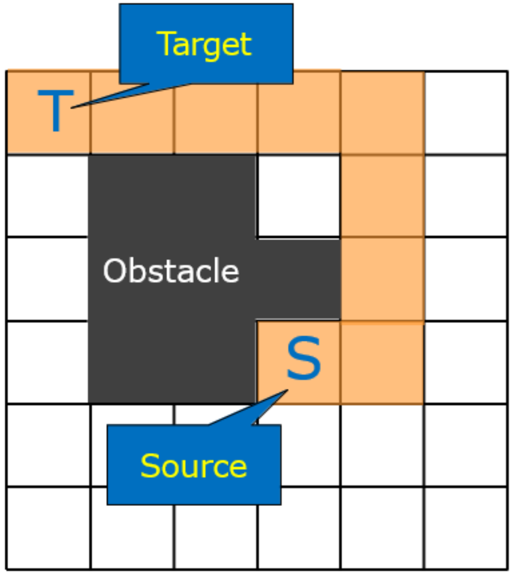
- The objective of detailed routing is to assign route segments of signal nets to specific routing tracks, vias, and metal layers in a manner consistent with given global routes of those nets
- Similar to global routing
  - Use physical wires to do connections
  - Estimating the wire resistance and capacitance, which determines whether the design meets timing requirements
- Detailed routing techniques are applied within routing regions
- Detailed routers must account for manufacturing rules and the impact of manufacturing faults

Routing Algorithm

GRID ASSUMPTION

- Despite the complexity of nano-scaled routing, we will use a grid assumption and add the complexity in later.
  - Layout is a grid of regular squares
  - A legal wire is a set of connected grid cells through unobstructed cells.
  - Obstacles (or blockages) are marked in the grid.
  - Wires are strictly horizontal and vertical (Manhattan Routing)
  - Paths go
    - North/South
    - East/West.



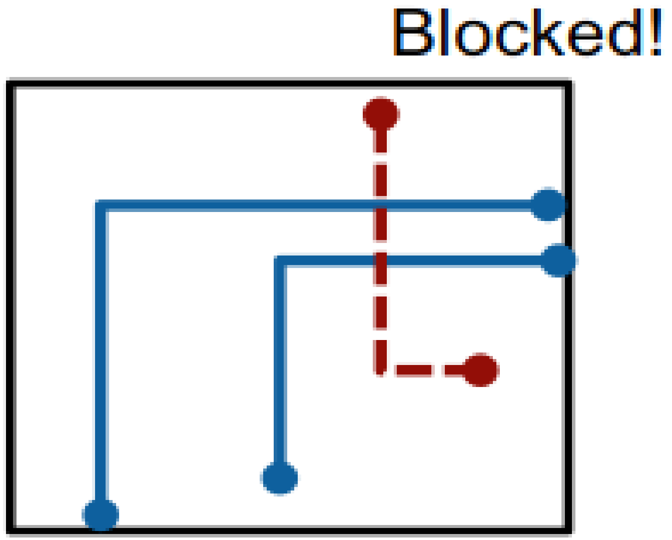


MAZE ROUTERS

- Also known as “Lee Routers”
  - C. Y. Lee, “An algorithm for path connections and its applications” 1961
- Strategy:
  - Route one net at a time.
  - Find the best wiring path for the current net.

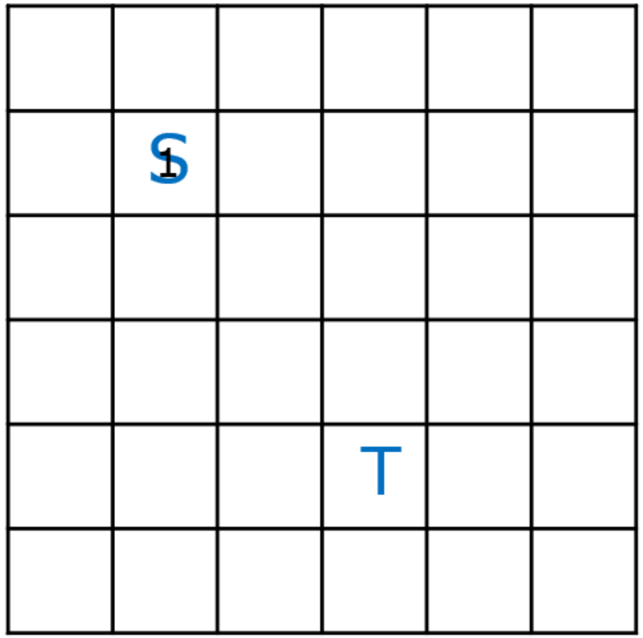
MAZE ROUTERS

- Also known as "Lee Routers"
  - C. Y. Lee, "An algorithm for path connections and its applications" 1961
- Strategy:
  - Route one net at a time.
  - Find the best wiring path for the current net.
- Problems:
  - Early wired nets may block path of later nets.
  - Optimal choice for one net may block others.
- Basic Idea:
  - Expand → Backtrace → Cleanup



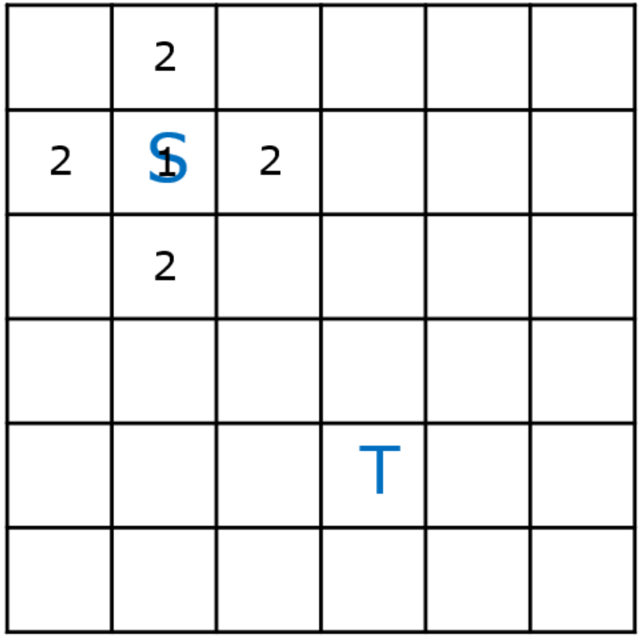
MAZE ROUTING: EXPANSION

- Start at the source.
- Find all paths 1 cell away.
- Continue until reaching the target.



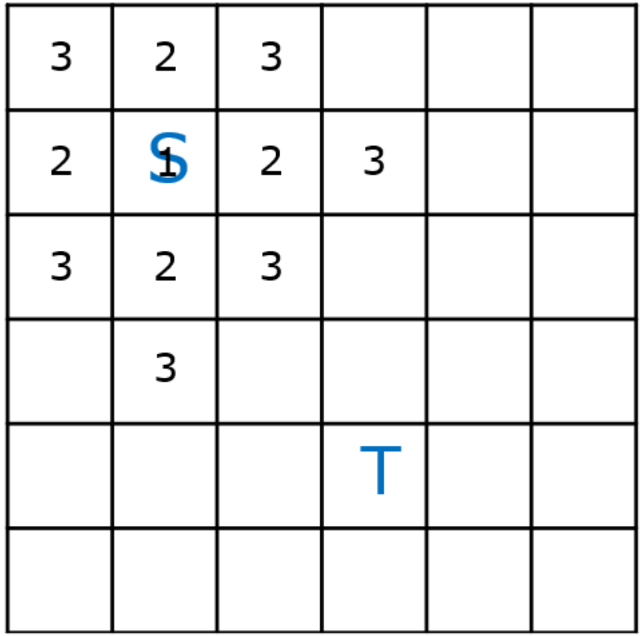
MAZE ROUTING: EXPANSION

- Start at the source.
- Find all paths 1 cell away.
- Continue until reaching the target.



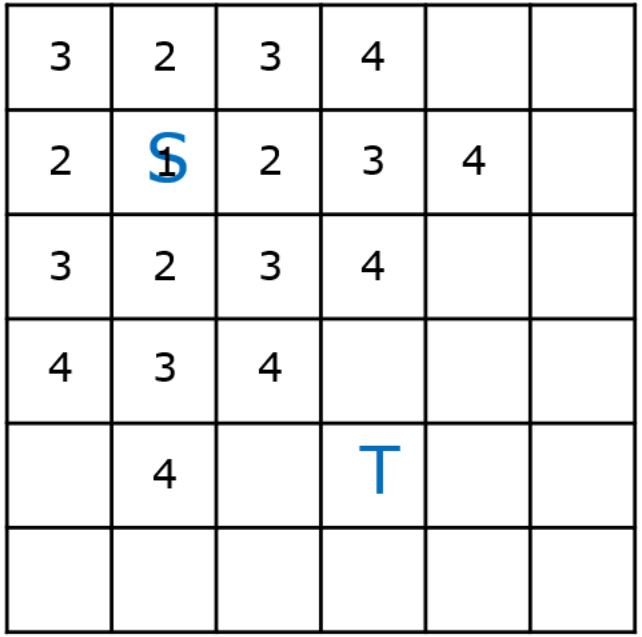
MAZE ROUTING: EXPANSION

- Start at the source.
- Find all paths 1 cell away.
- Continue until reaching the target.



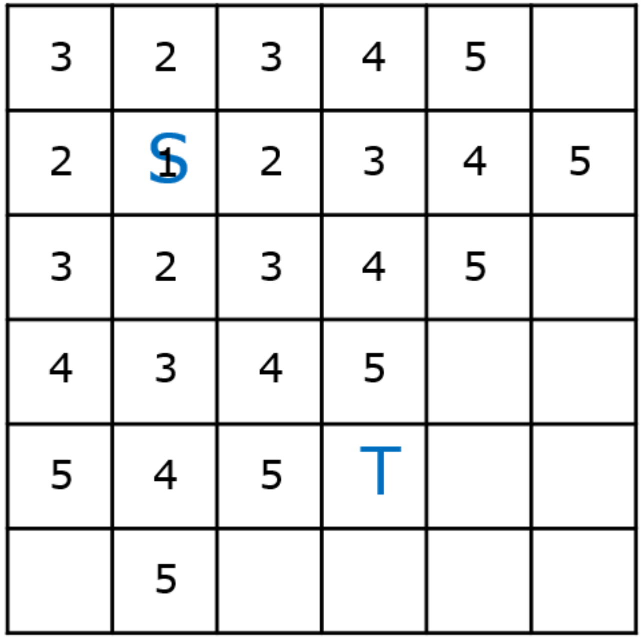
MAZE ROUTING: EXPANSION

- Start at the source.
- Find all paths 1 cell away.
- Continue until reaching the target.



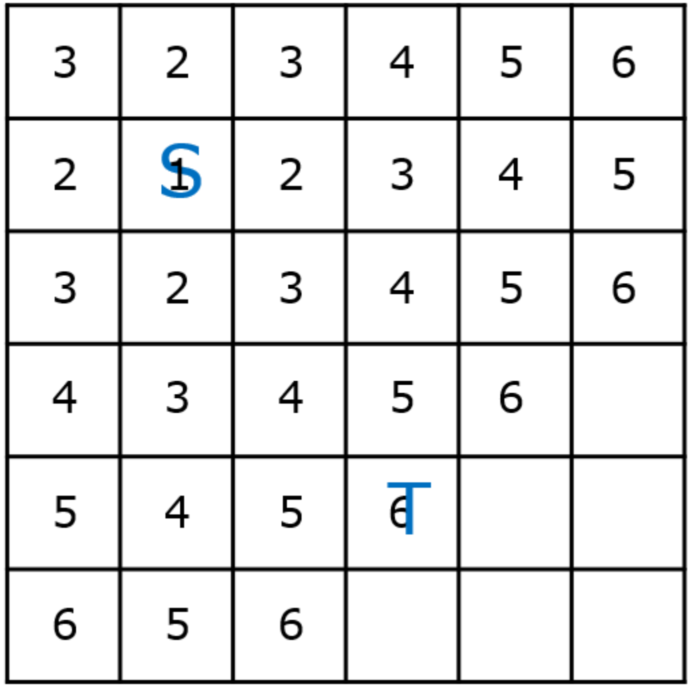
MAZE ROUTING: EXPANSION

- Start at the source.
- Find all paths 1 cell away.
- Continue until reaching the target.



MAZE ROUTING: EXPANSION

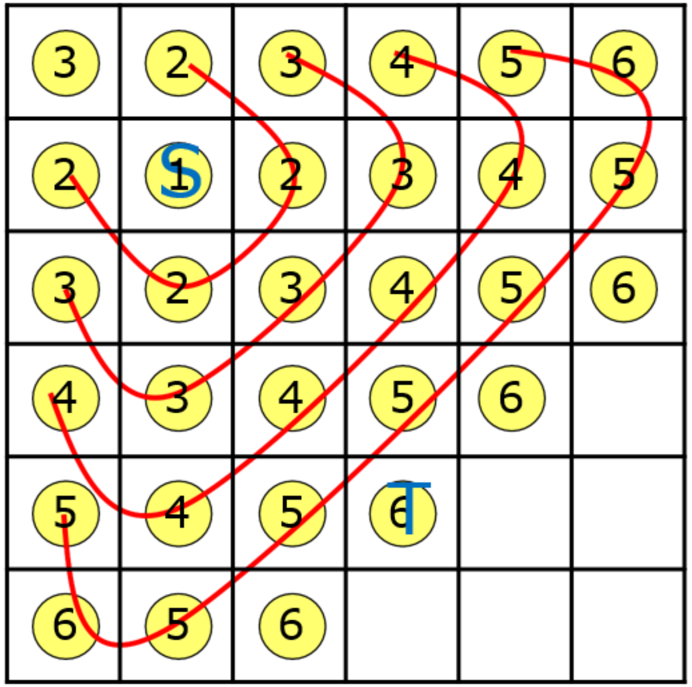
- Start at the source.
- Find all paths 1 cell away.
- Continue until reaching the target.



3	2	3	4	5	6
2	1 S	2	3	4	5
3	2	3	4	5	6
4	3	4	5	6	
5	4	5	6 T		
6	5	6			

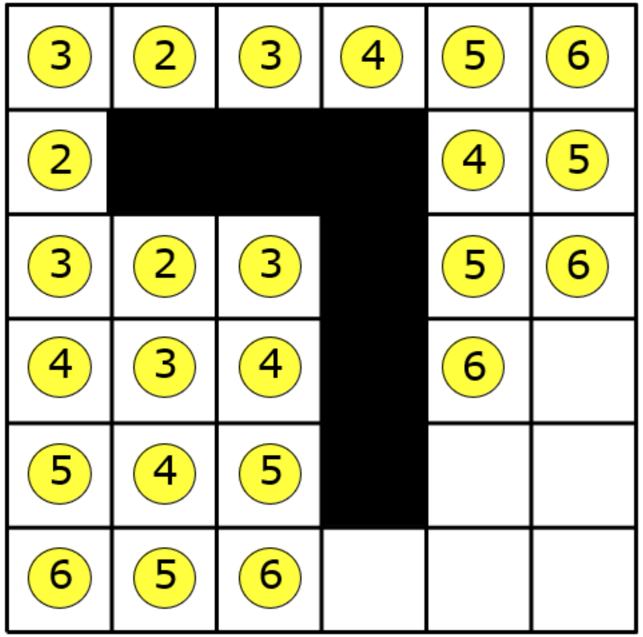
MAZE ROUTING: EXPANSION

- Start at the source.
- Find all paths 1 cell away.
- Continue until reaching the target.
  - ➢ We approach the target with a "wavefront"
  - ➢ We found that the shortest path to the target is 6 unit steps.



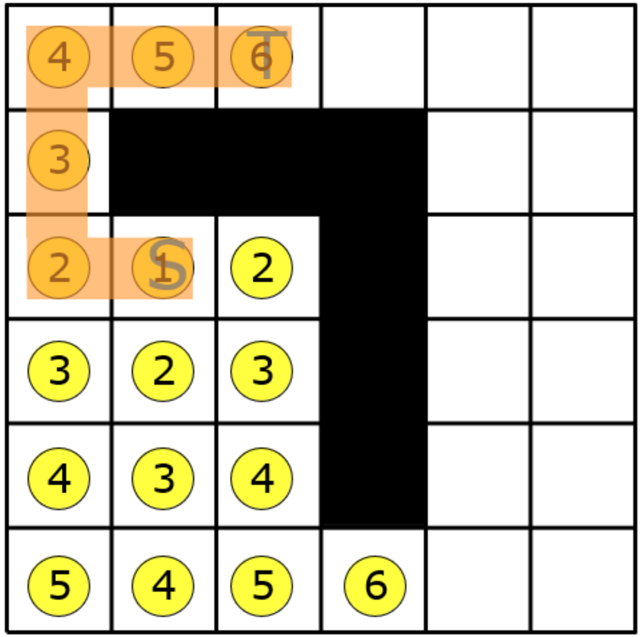
MAZE ROUTING: BACKTRACE & CLEANUP

- Backtrace:
  - Follow the path lengths backwards in descending order.
  - This will mark a shortest-path to the target.
  - However, there may be many shortest paths, so optimization can be used to select the best one.
- Cleanup
  - We have now routed the first net.
  - To ensure that future nets do not try to use the same resources, mark the net path from S to T as an obstacle.



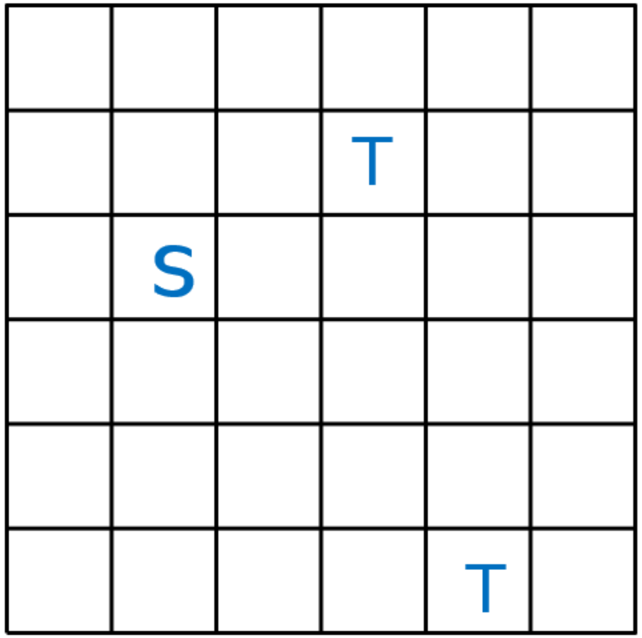
MAZE ROUTING: BLOCKAGES

- How do we deal with blockages?
  - Easy. Just "go around" them!
- To summarize:
  - Expand:
    - Breadth-first-search (BFS) to find all paths from S to T in path-length order.
  - Backtrace:
    - Walk shortest path back to source.
  - Cleanup:
    - Mark net as obstacle and erase distance markers.



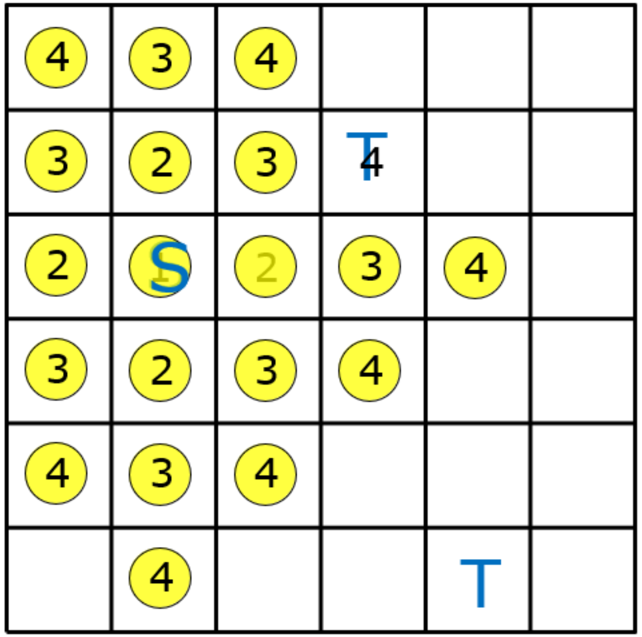
MULTI-POINT NETS

- How do we go about routing a net with multiple targets?
  - Actually, pretty straightforward.
  - Start with our regular maze routing algorithm to find the path to the nearest target.



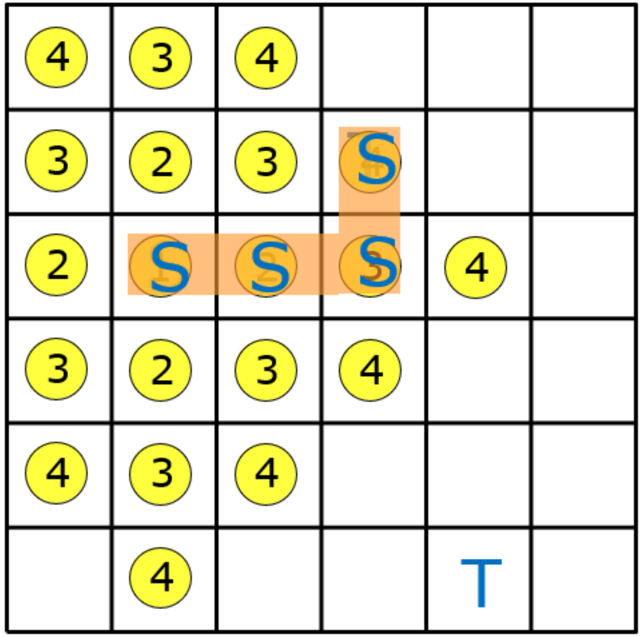
MULTI-POINT NETS

- How do we go about routing a net with multiple targets?
  - Actually, pretty straightforward.
  - Start with our regular maze routing algorithm to find the path to the nearest target.



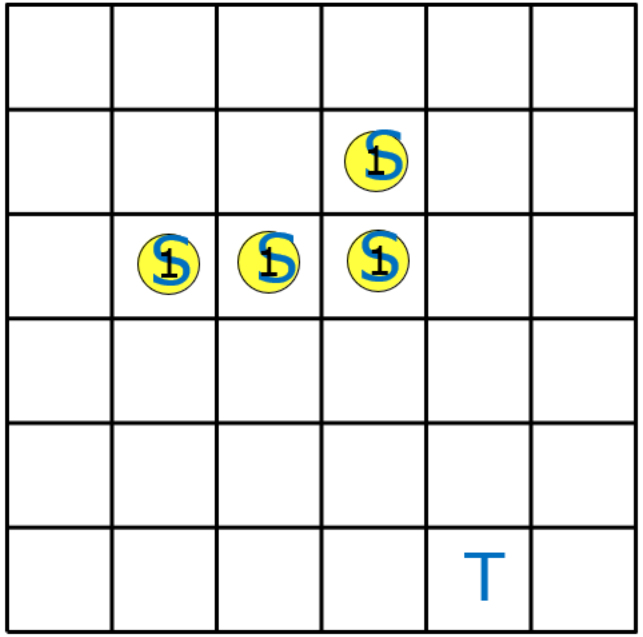
MULTI-POINT NETS

- How do we go about routing a net with multiple targets?
  - Actually, pretty straightforward.
  - Start with our regular maze routing algorithm to find the path to the nearest target.
  - Then re-label all cells on found path as sources, and re-run maze router using all sources simultaneously.



MULTI-POINT NETS

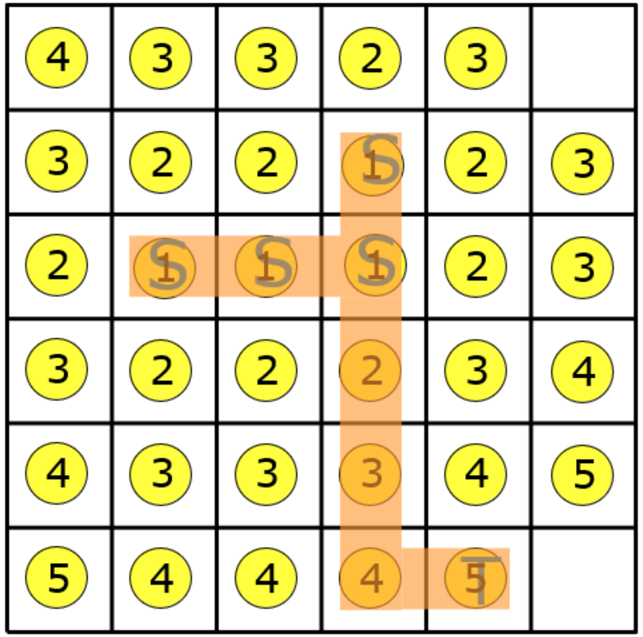
- How do we go about routing a net with multiple targets?
  - Actually, pretty straightforward.
  - Start with our regular maze routing algorithm to find the path to the nearest target.
  - Then re-label all cells on found path as sources, and re-run maze router using all sources simultaneously.



MULTI-POINT NETS

- How do we go about routing a net with multiple targets?
  - Actually, pretty straightforward.
  - Start with our regular maze routing algorithm to find the path to the nearest target.
  - Then re-label all cells on found path as sources, and re-run maze router using all sources simultaneously.
  - Repeat until reaching all target points.

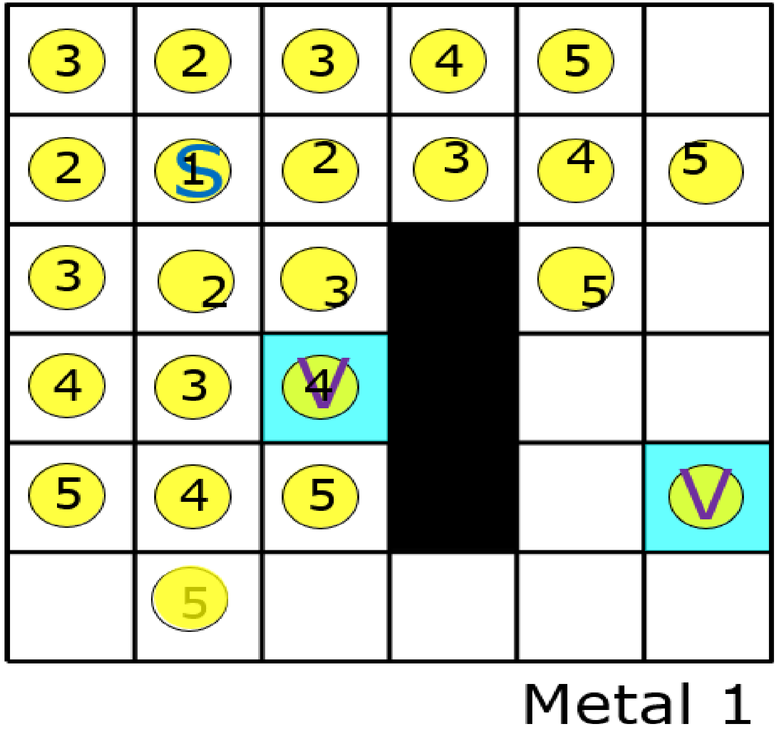
Note that this does not guarantee the shortest path (="Steiner Tree")

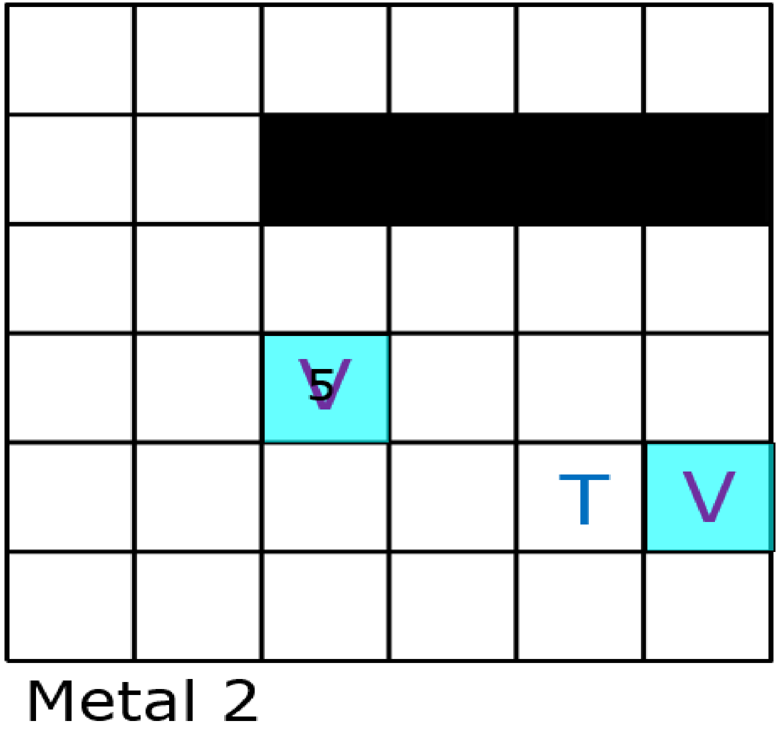


MULTI-LAYER ROUTING

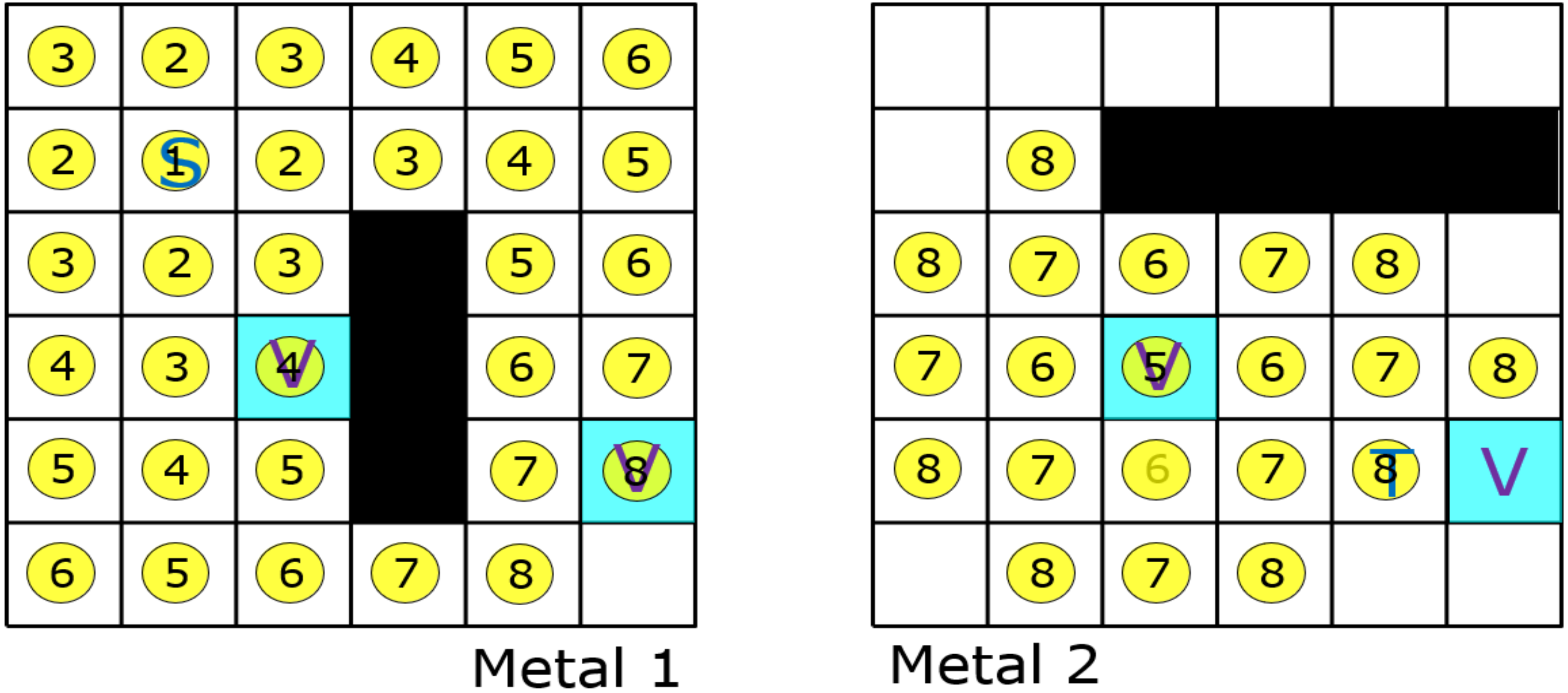
- Okay, so what about dealing with several routing layers?
  - Same basic idea of grid – one grid for each layer.
  - Each grid box can contain one via.
  - New expansion direction – up/down.

MULTI-LAYER ROUTING

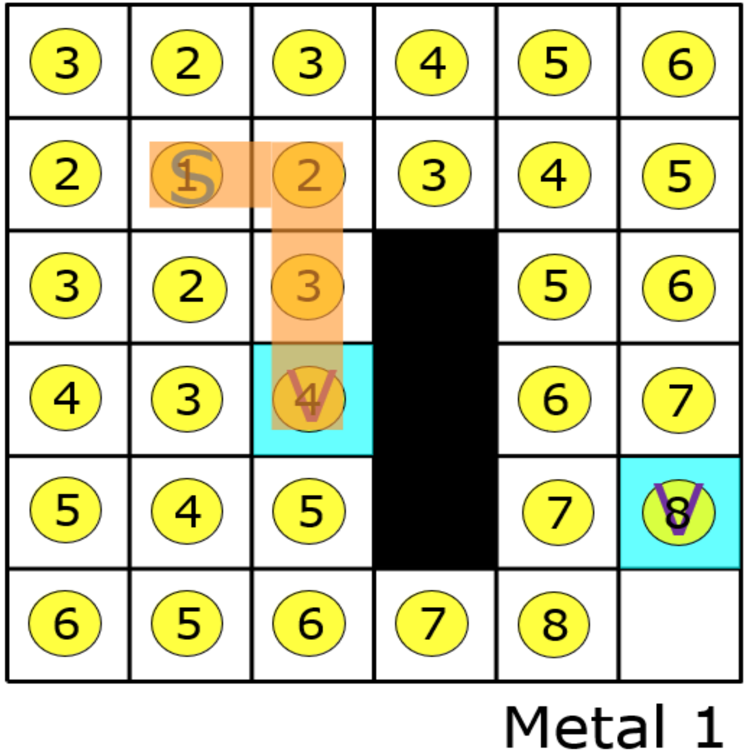


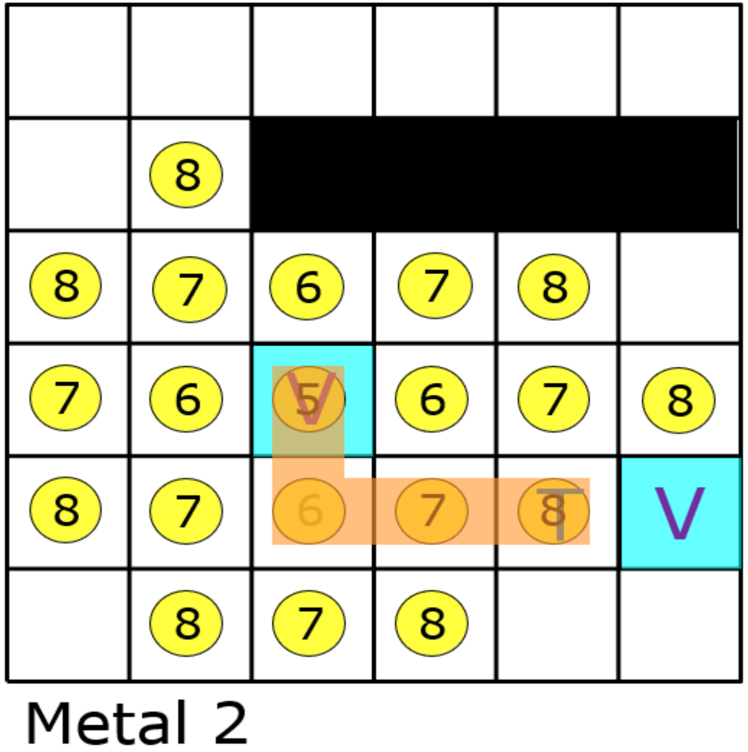


MULTI-LAYER ROUTING



MULTI-LAYER ROUTING

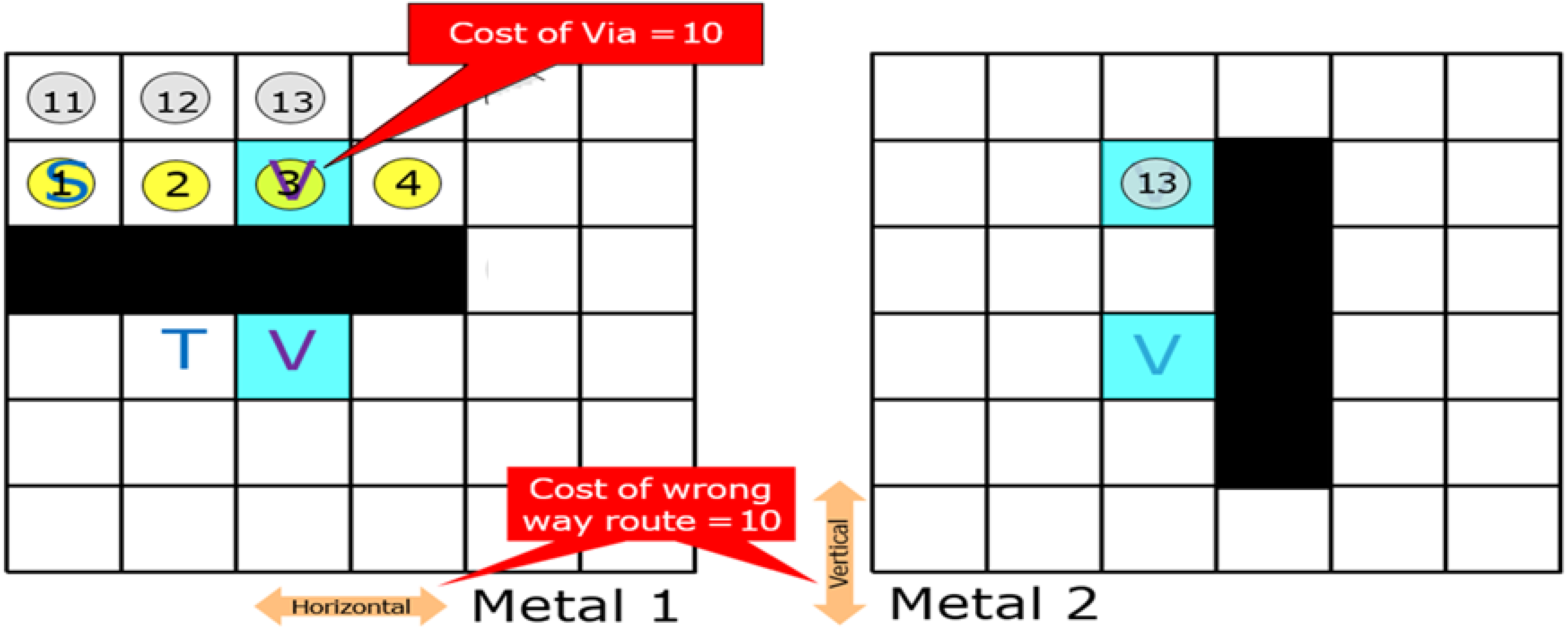




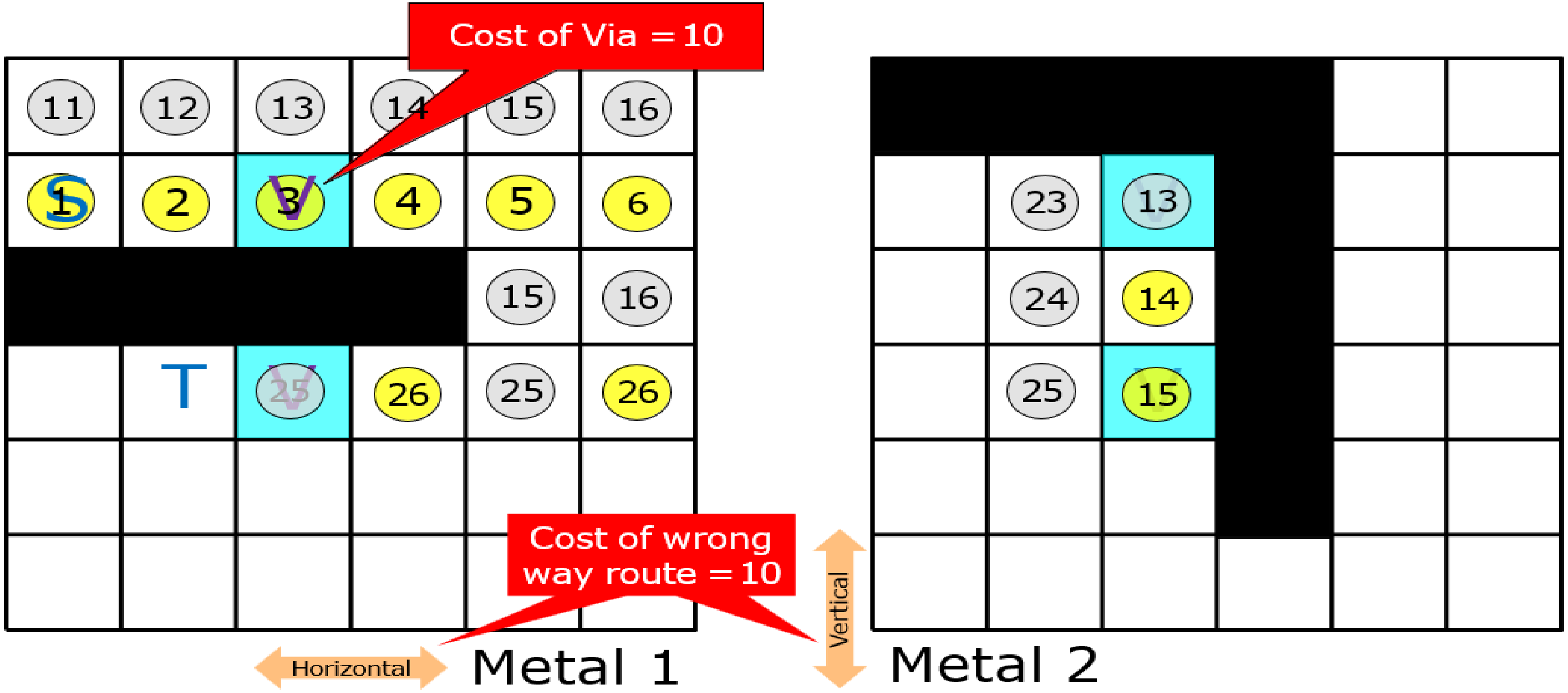
NON-UNIFORM GRID COSTS

- But we know that vias have (relatively) high resistance.
  - ➢ Shouldn't we prefer to stay on the same metal layer?
- We also prefer Manhattan Routing
  - ➢ Each layer is only routed in one direction.
  - ➢ A "turn" requires going through a via or a "jog" should be penalized.
- Is there a way to prefer routing in a certain layer/direction?
- Yes.
  - ➢ Let's introduce non-uniform grid costs.

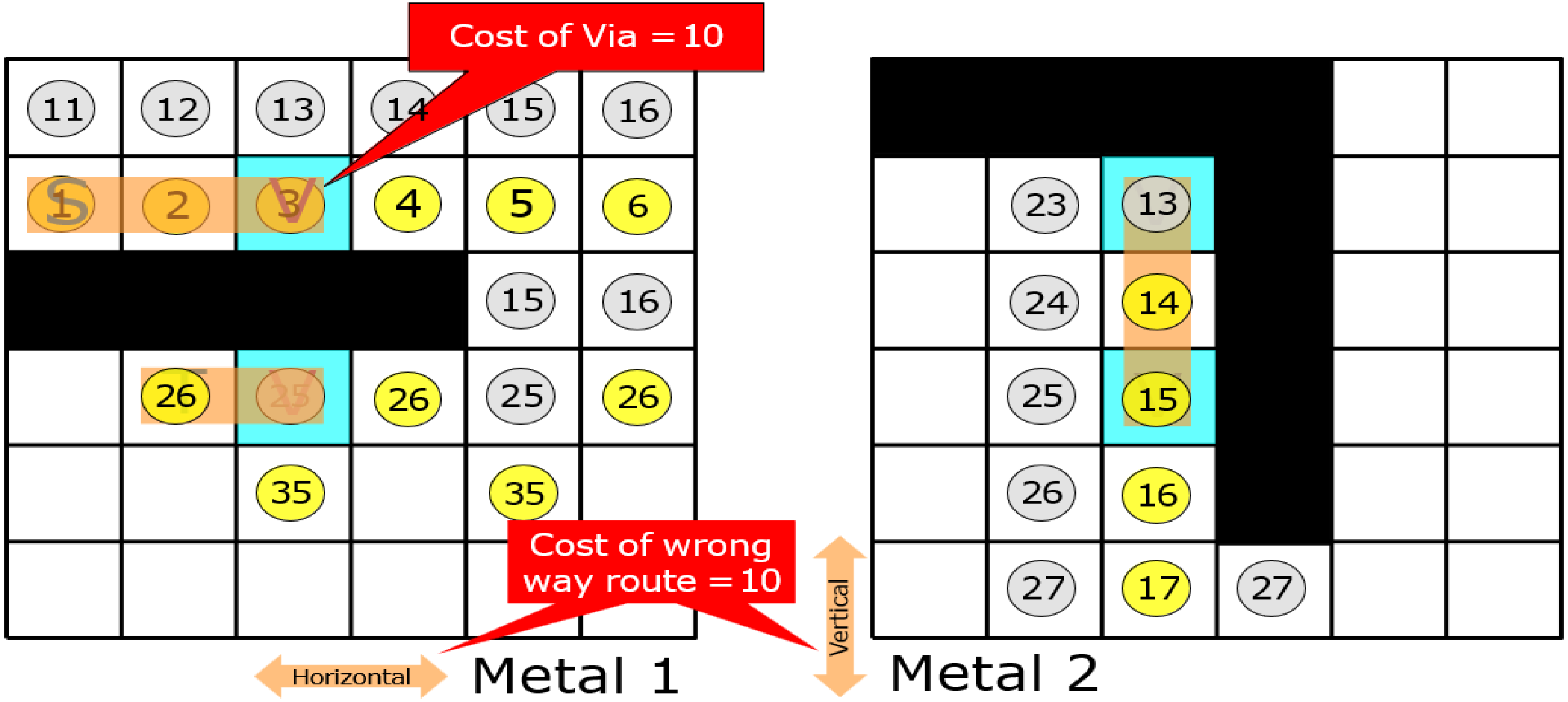
MULTI-LAYER ROUTING WITH NON-UNIFORM COST



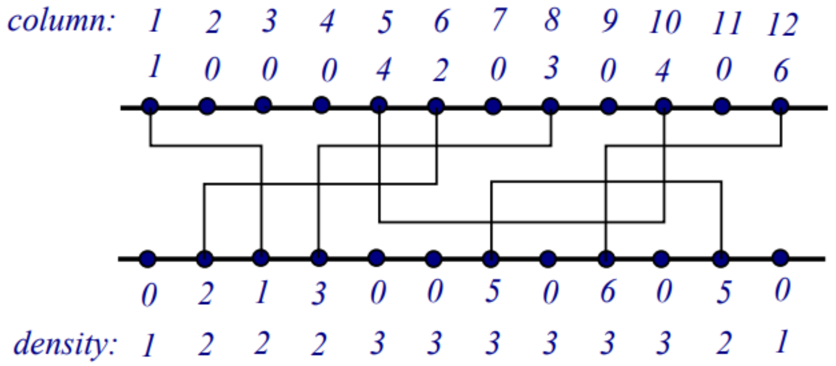
MULTI-LAYER ROUTING WITH NON-UNIFORM COST



MULTI-LAYER ROUTING WITH NON-UNIFORM COST



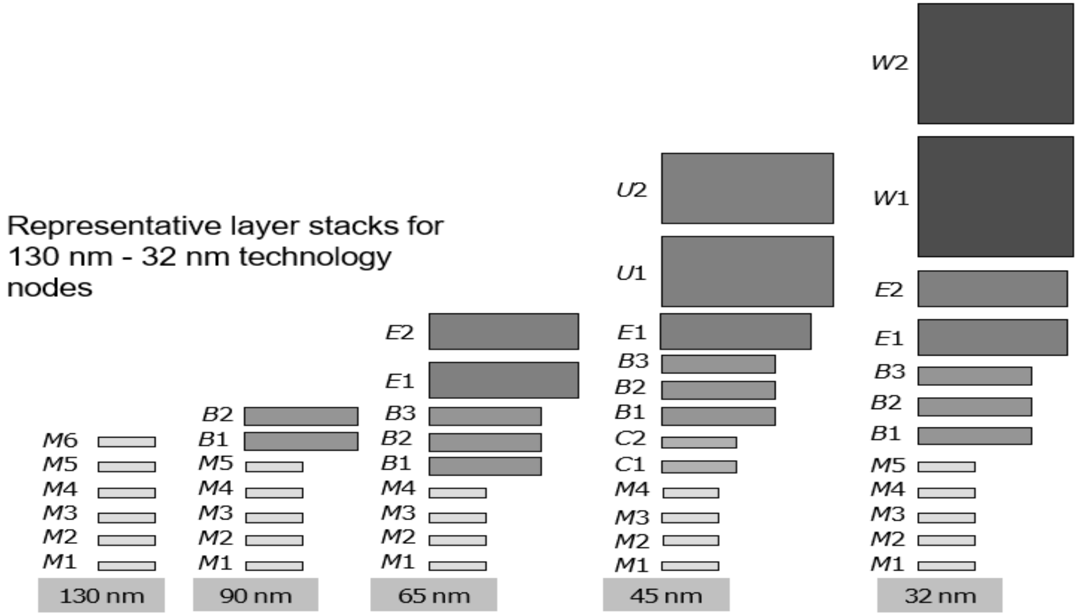
LEFT-EDGE ALGORITHM


column: 1 2 3 4 5 6 7 8 9 10 11 12
1 0 0 0 4 2 0 3 0 4 0 6
0 2 1 3 0 0 5 0 6 0 5 0
density: 1 2 2 2 3 3 3 3 3 3 2 1

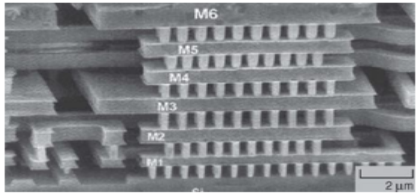
Routing in Practice

LAYER STACKS

- Metal stacks are changing (and growing)



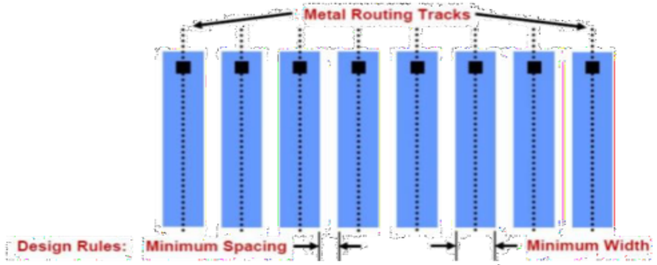
Intel 45nm 8 metal stack

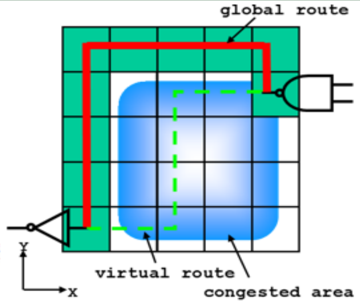


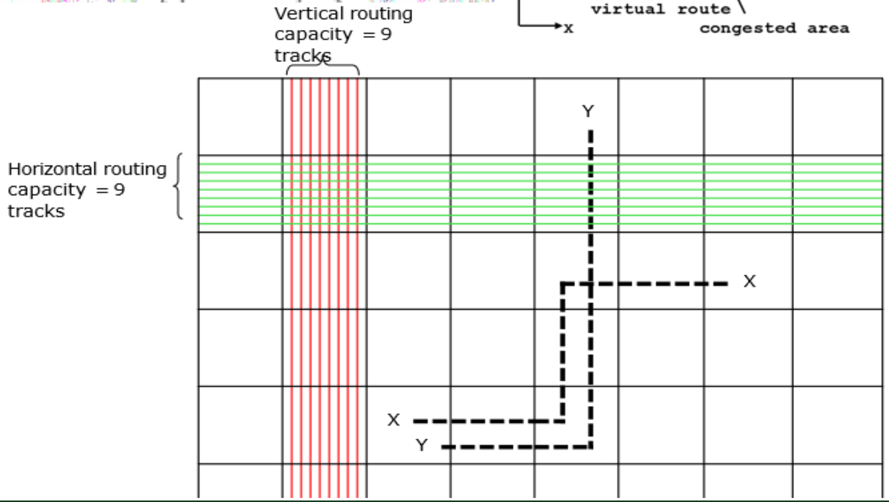
UMC 6 metal stack

GLOBAL ROUTE

- Divide floorplan into GCells
  - Approximately 10 tracks per layer each.
- Perform fast grid routing:
  - Minimize wire-length
  - Balance Congestion
  - Timing-driven
  - Noise/SI-driven
  - Keep buses together
- Also used for trial route
  - During earlier stages of the flow

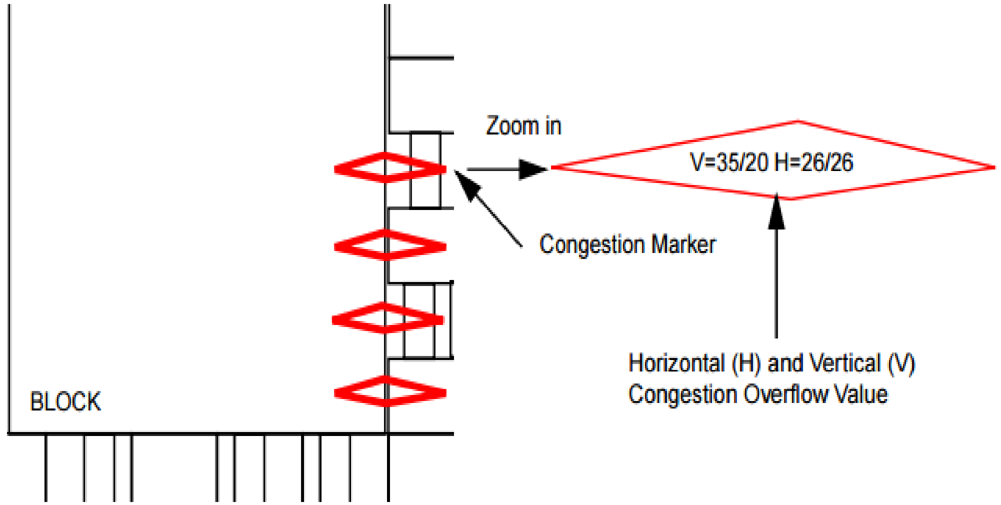




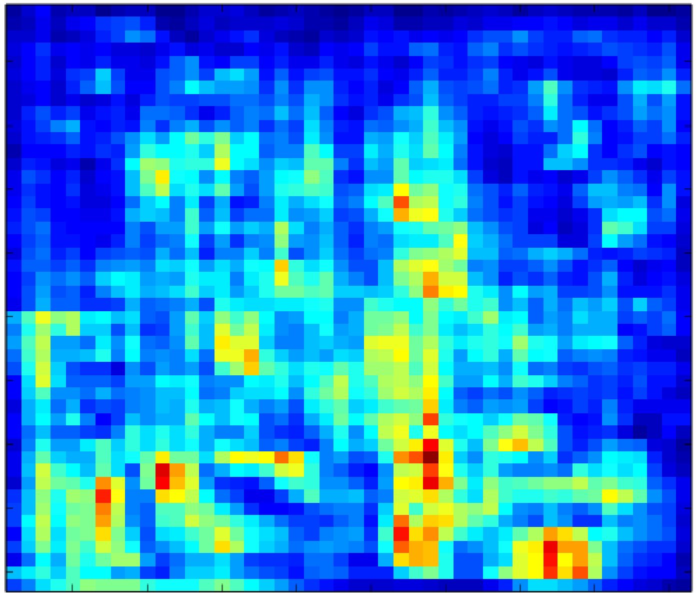

Vertical routing capacity = 9 tracks
Horizontal routing capacity = 9 tracks
Y
X
X
Y

CONGESTION MAP

- Use congestion map and report to examine design routability

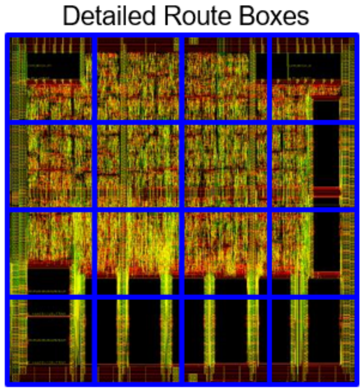


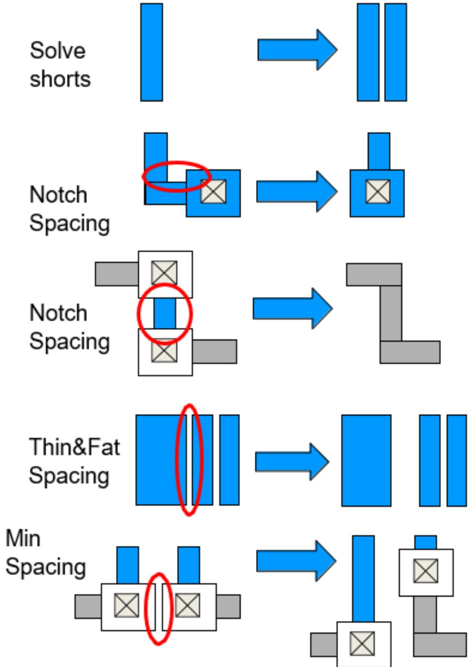
Congestion map



DETAILED ROUTE

- Using global route plan, within each global route cell
  - Assign nets to tracks
  - Lay down wires
  - Connect pins to nets
  - Solve DRC violations
  - Reduce cross couple cap
  - Apply special routing rules
- Flow:
  - Track Assignment (TA)
  - DRC fixing inside a Global Routing Cell (GRC)
  - Iterate to achieve a solution (default ~20 iterations)





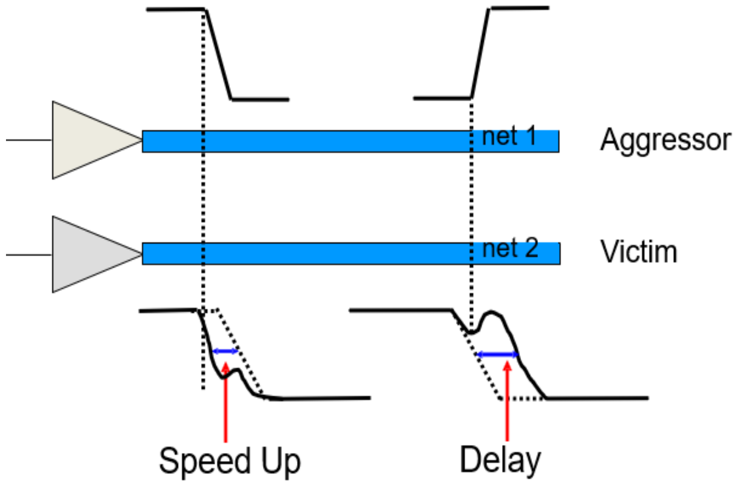
TIMING-DRIVEN ROUTING

- Optimize critical paths
- Route some nets first
  - ➢ Most routing freedom at start
  - ➢ Use shortest paths possible
- Net weights
  - ➢ Order of routing (priorities: e.g., Default : Clocks 50, others 2)
- Wire widening
  - ➢ Reduce resistance
- If you have a congested design, you may need to set the timing driven effort to “low”
- Beware when changing default options

CAL POLY
SAN LUIS OBISPO

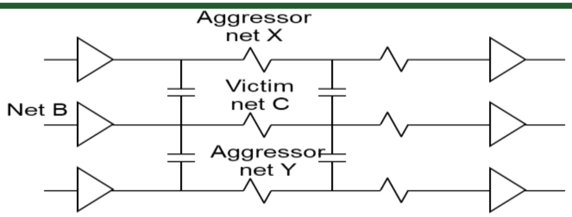
SIGNAL INTEGRITY (SI)

- Signal Integrity during routing is synonymous with Crosstalk.
    - A switching signal may affect a neighboring net.
    - The switching net is called the Aggressor.
    - The affected net is called the Victim.
- Two major effects:
    - Signal slow down
        - When the aggressor and victim switch in **opposite** directions.
    - Signal speed up
        - When the aggressor and victim switch in the **same** direction.



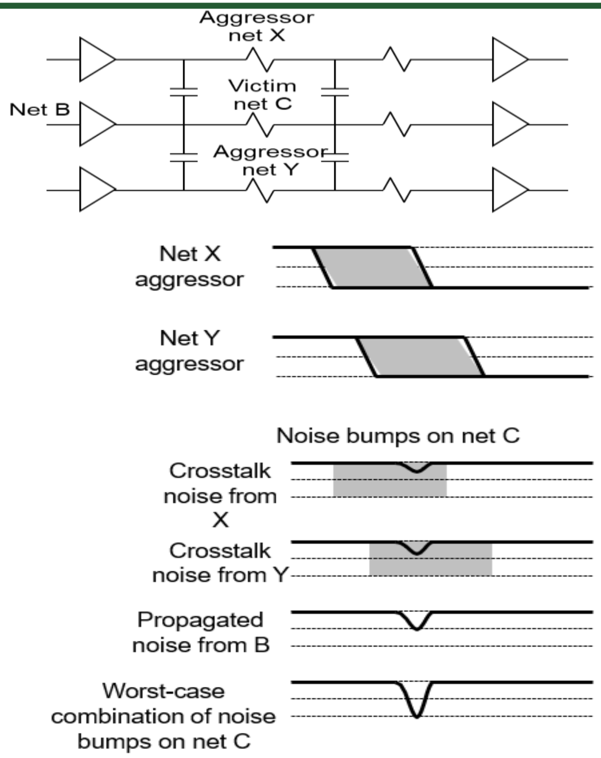
SI MULTI-AGGRESSOR TIMING ANALYSIS

- Infinite Window Analysis
  - An infinite noise window applies the maximum delay due to crosstalk during timing analysis.
  - This model was sufficient for older (pre-90nm) technologies but became too severe with the growing sidewall capacitances at scaled nodes.



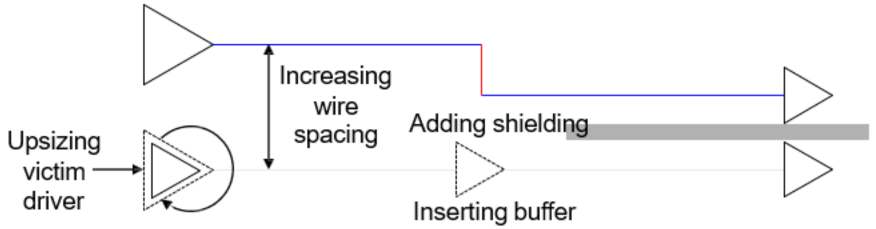
SI MULTI-AGGRESSOR TIMING ANALYSIS

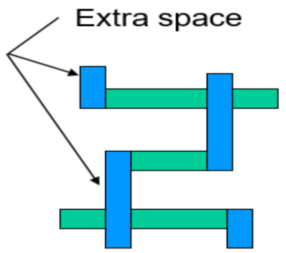
- Infinite Window Analysis
  - An infinite noise window applies the maximum delay due to crosstalk during timing analysis.
  - This model was sufficient for older (pre-90nm) technologies but became too severe with the growing sidewall capacitances at scaled nodes.
- Propagated Noise Analysis
  - Min/Max vectors are propagated through the design to create a transition window for all aggressors in relation to a certain victim.
  - Noise is only applied at the overlap of the two windows to determine the worst-case noise bump.



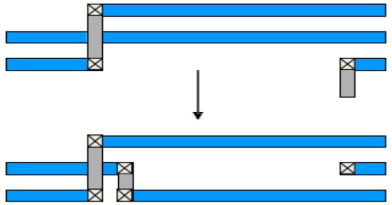
SIGNAL INTEGRITY - SOLUTIONS

- Crosstalk Prevention
  - Limit length of parallel nets
  - Wire spreading
  - Shield special nets
  - Upsize driver or buffer

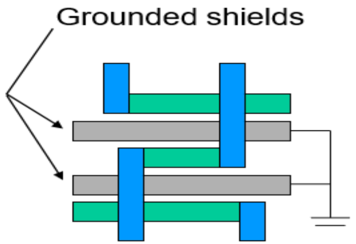




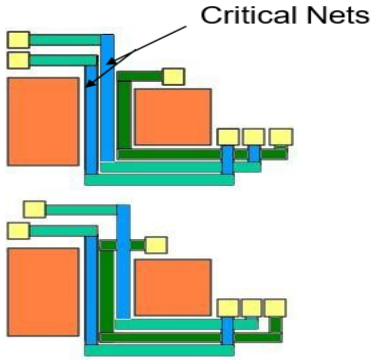
Spacing



Wire Spreading



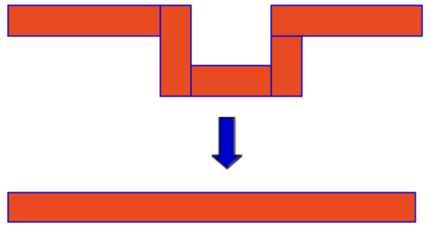
Shielding
Same layer (H)
Adjacent layers (V)

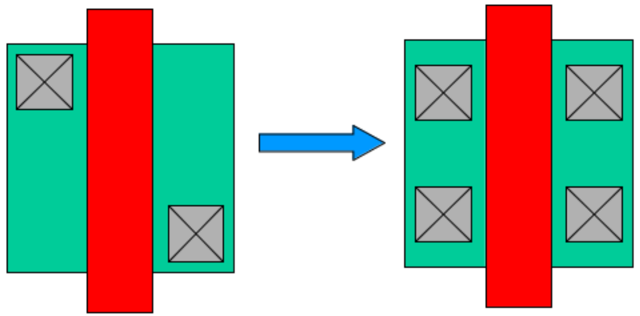


Net Ordering

DESIGN FOR MANUFACTURING

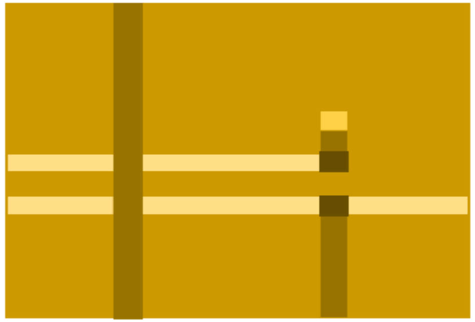
- During route, apply additional design for manufacturing (DFM) and/or design for yield (DFY) rules:
  - Via reduction
  - Redundant via insertion
  - Wire straightening
  - Wire spreading

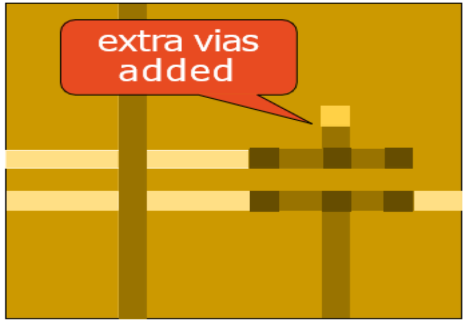

Wire straightening (reduce jogs)

Avoid asymmetrical contacts


DFM: VIA OPTIMIZATION

- Post-Route Via Optimization, includes:
  - Incremental routing for the minimization of vias.
  - Replacement of single vias with multi-cut vias.
- These operations are required for:
  - Reliability:
    - The ability to create reliable vias decreases with each process node. If a single via fails, it creates an open and the circuit is useless.
  - Electromigration:
    - Electromigration hazards are even more significant in vias, which are essentially long, narrow conductors.

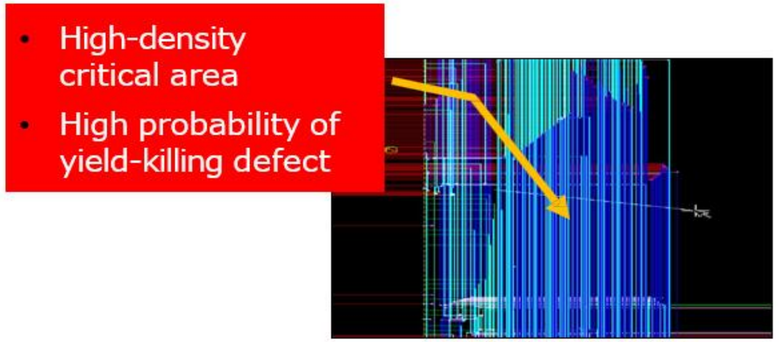


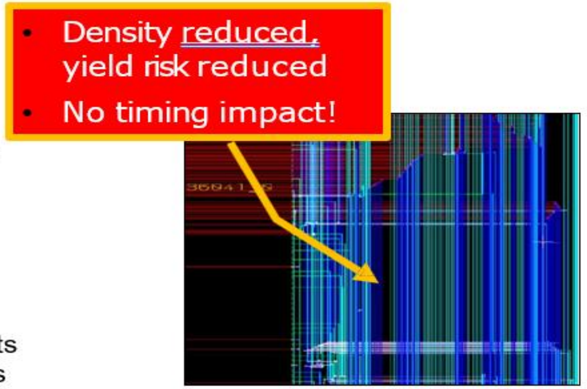


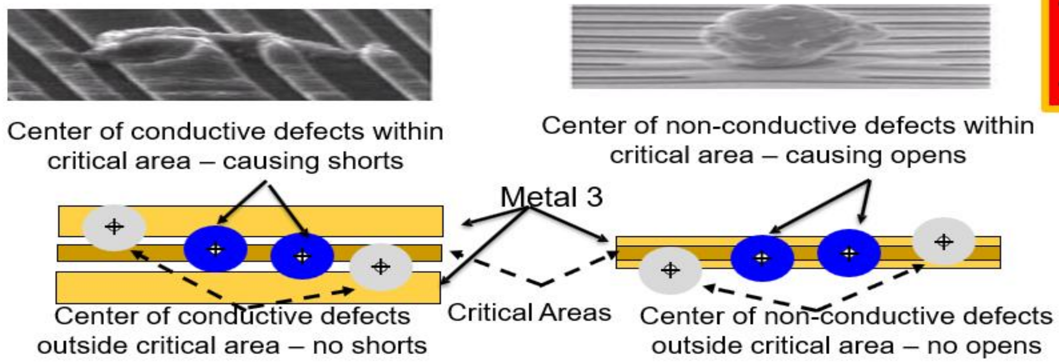
CAL POLY
SAN LUIS OBISPO

DFM: WIRE SPREADING

- Wire spreading achieves:
  - ➢ Lower capacitance and better signal integrity.
  - ➢ Lower susceptibility to shorts or opens due to random particle defects.


• High-density critical area
• High probability of yield-killing defect




Center of conductive defects within critical area – causing shorts
Center of non-conductive defects within critical area – causing opens
Metal 3
Critical Areas
Center of conductive defects outside critical area – no shorts
Center of non-conductive defects outside critical area – no opens

CAL POLY
SAN LUIS OBISPO

Thank you!