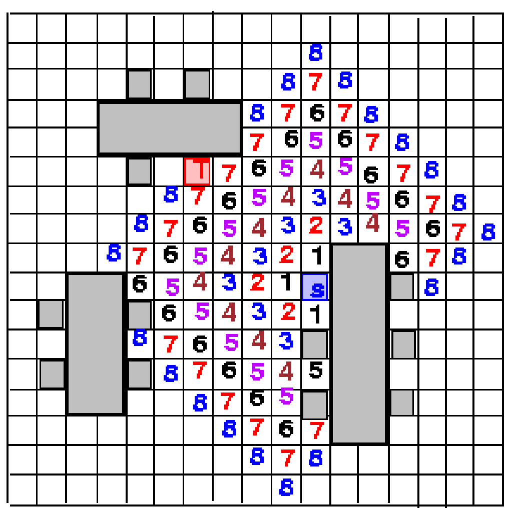
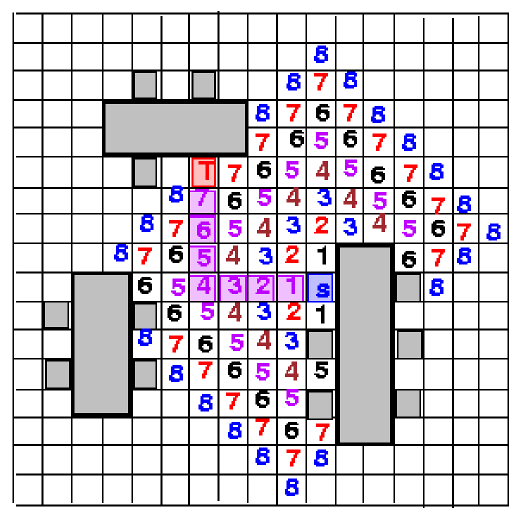
Unit 6: Maze (Area) and Global Routing

- Course contents
  - Routing basics
  - Maze (area) routing
  - Global routing



Filling



Retrace

Routing

placement

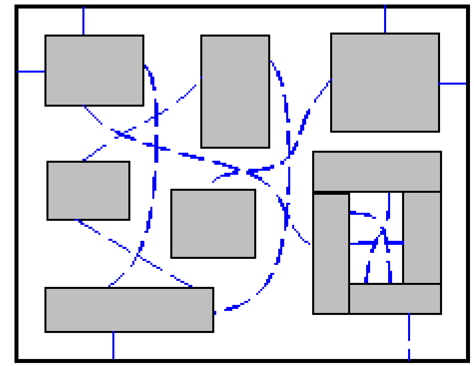
- Generates a "loose" route for each net.
- Assigns a list of routing regions to each net without specifying the actual layout of wires.

global routing

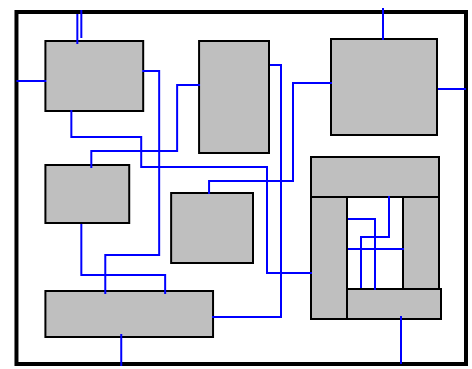
detailed routing

- Finds the actual geometric layout of each net within the assigned routing regions.

compaction



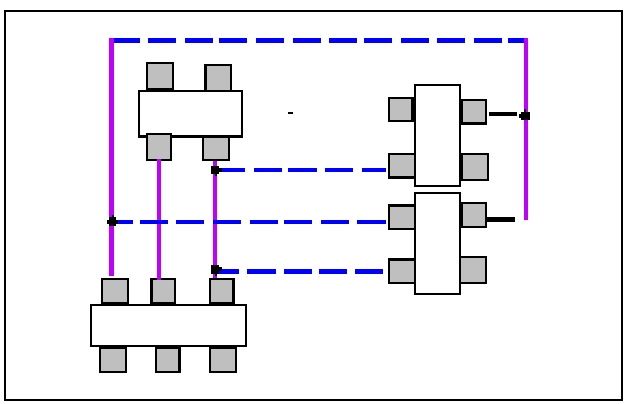
Global routing



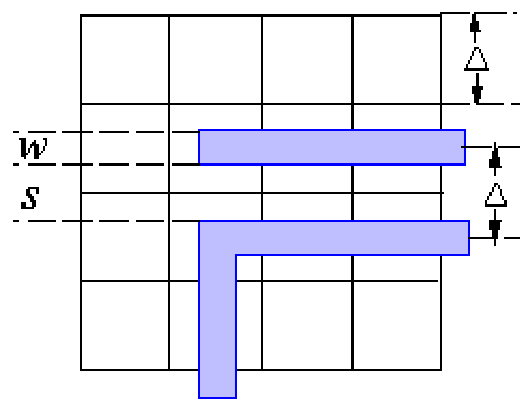
Detailed routing

Routing Constraints

- 100% routing completion + area minimization, under a set of constraints:
  - Placement constraint: usually based on fixed placement
  - Number of routing layers
  - Geometrical constraints: must satisfy design rules
  - Timing constraints (performance-driven routing): must satisfy delay constraints
  - Crosstalk?
  - Process variations?

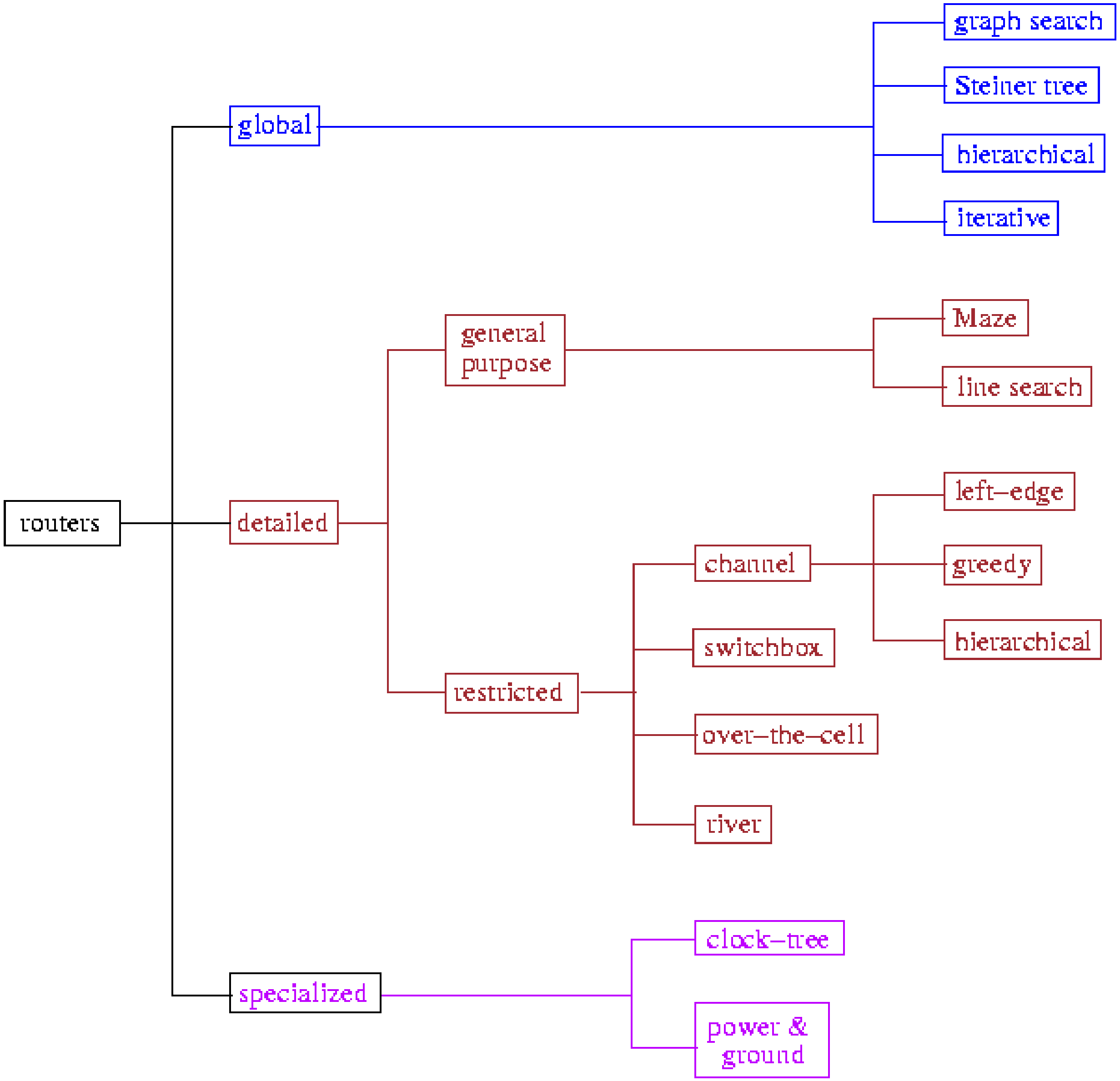


Two–layer routing



Geometrical constraint

Classification of Routing



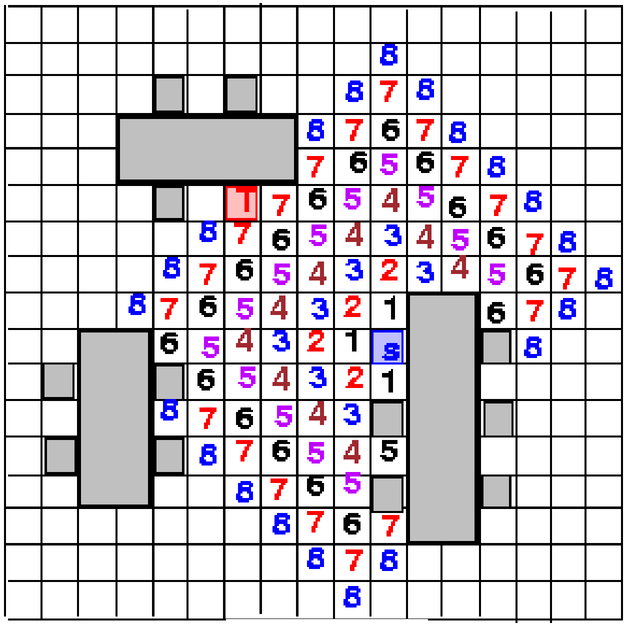
Slides Courtesy of Prof. H.-R. Jiang

Maze Router: Lee Algorithm

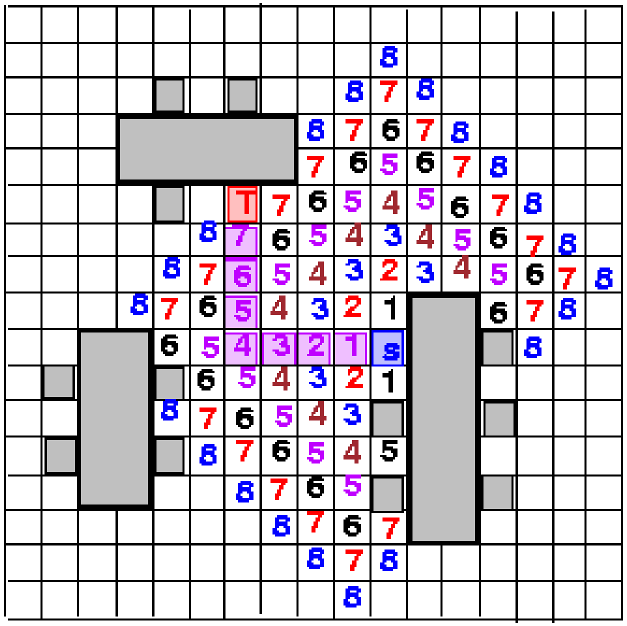
- Lee, "An algorithm for path connection and its application," *IRE Trans. Electronic Computer*, EC-10, 1961.
- Discussion mainly on single-layer routing
- **Strengths**
  - Guarantee to find connection between 2 terminals if it exists.
  - Guarantee minimum path.
- **Weaknesses**
  - Requires large memory for dense layout.
  - Slow.
- Applications: global routing, detailed routing

Lee Algorithm

- Find a path from *S* to *T* by "wave propagation".



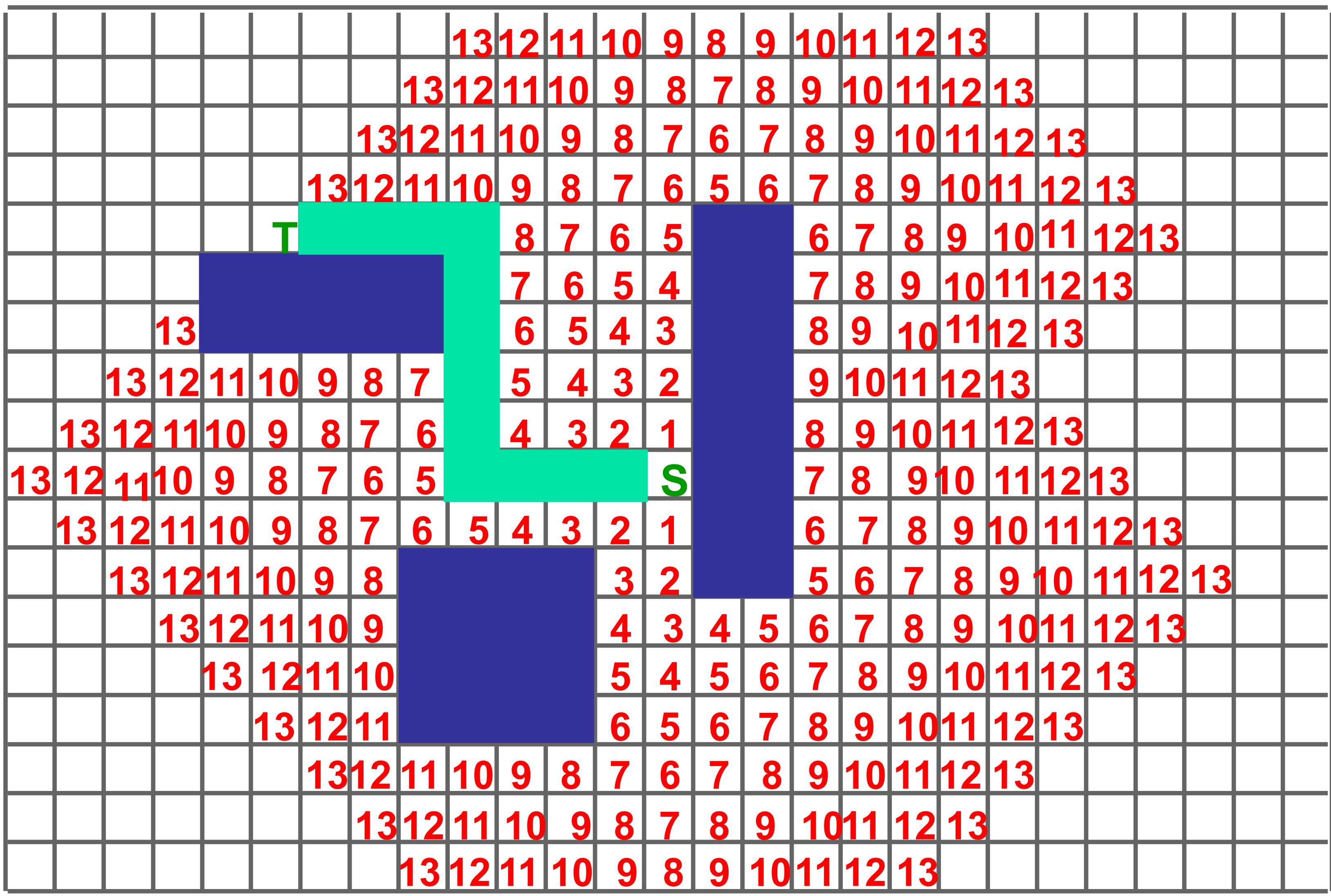
Filling



Retrace

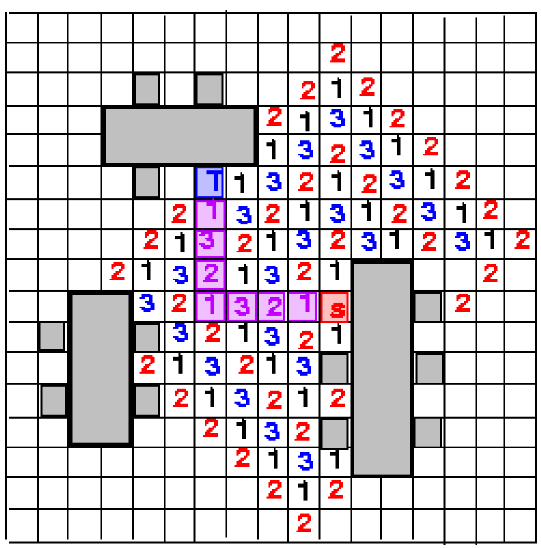
- Time & space complexity for an $M \times N$ grid: $O(MN)$ (**huge!**)

Example

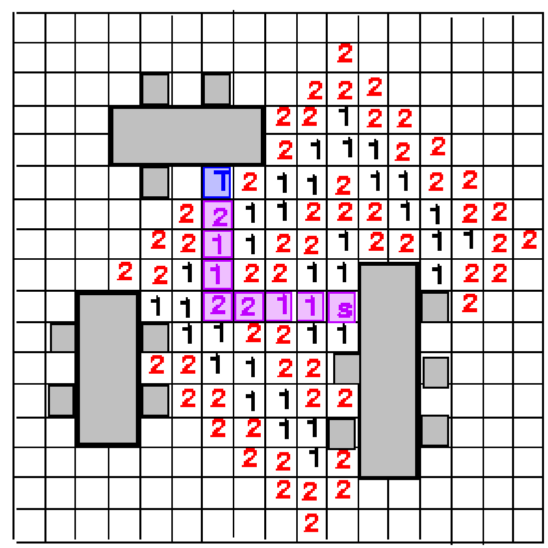


Reducing Memory Requirement

- Akers's Observations (1967)
  - Adjacent labels for *k* are either *k*-1 or *k*+1.
  - Want a labeling scheme such that each label has its preceding label different from its succeeding label.
- Way 1: coding sequence 1, 2, 3, 1, 2, 3, …; states: 1, 2, 3, *empty*, *blocked* (3 bits required)
- Way 2: coding sequence 1, 1, 2, 2, 1, 1, 2, 2, …; states: 1, 2, *empty*, *blocked* (need only 2 bits)



Sequence: 1, 2, 3, 1, 2, 3, ...

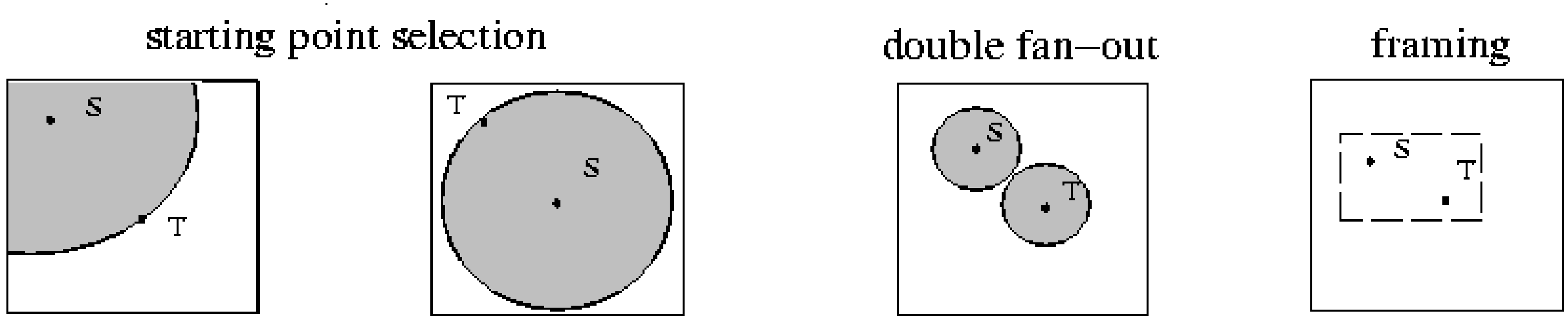


Sequence: 1, 1, 2, 2, 1, 1, 2, 2, ...

Slides Courtesy of Prof. H.-R. Jiang

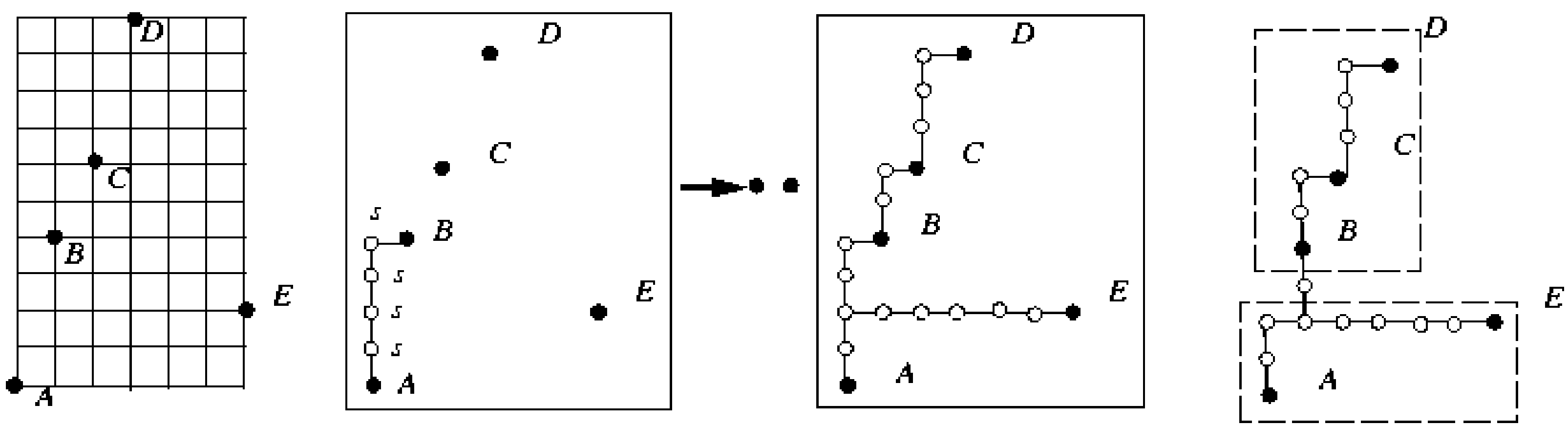
Reducing Running Time

- Starting point selection: Choose the point farthest from the center of the grid as the starting point.
- Double fan-out: Propagate waves from both the source and the target cells.
- Framing: Search inside a rectangle area 10--20% larger than the bounding box containing the source and target.
  - Need to enlarge the rectangle and redo if the search fails.



Connecting Multi-Terminal Nets

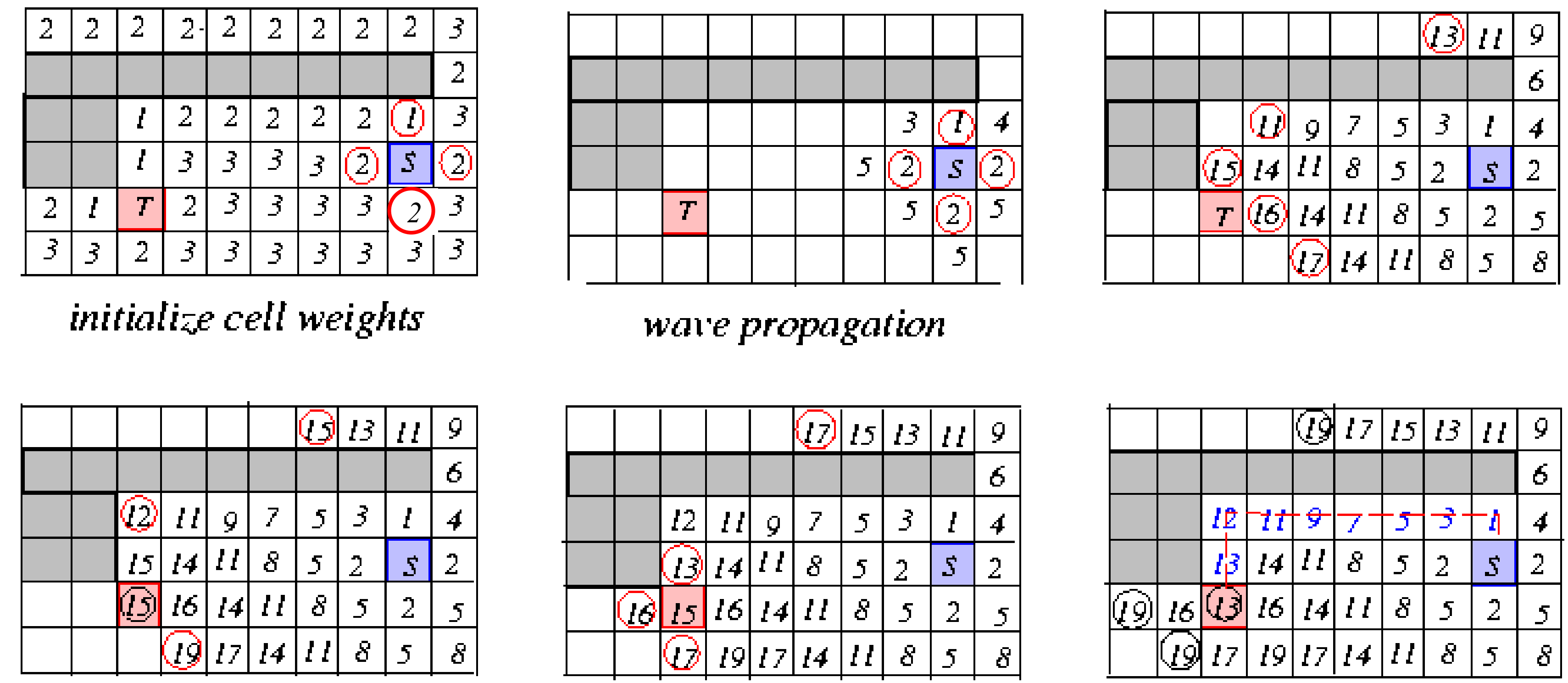
- Step 1: Propagate wave from the source *s* to the closet target.
- Step 2: Mark ALL cells on the path as *s*.
- Step 3: Propagate wave from ALL *s* cells to the other cells.
- Step 4: Continue until all cells are reached.
- Step 5: Apply heuristics to further reduce the tree cost.



Routing on a Weighted Grid

- Motivation: finding more desirable paths
- *weight*(*grid cell*) = *# of unblocked grid cell segments* - 1

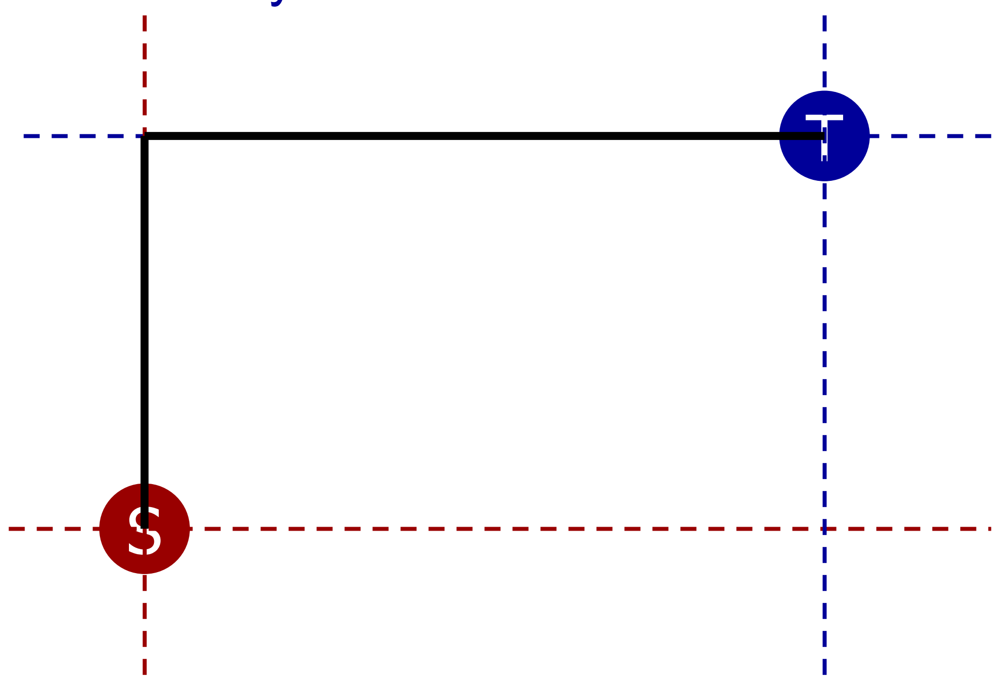
A Routing Example on a Weighted Grid



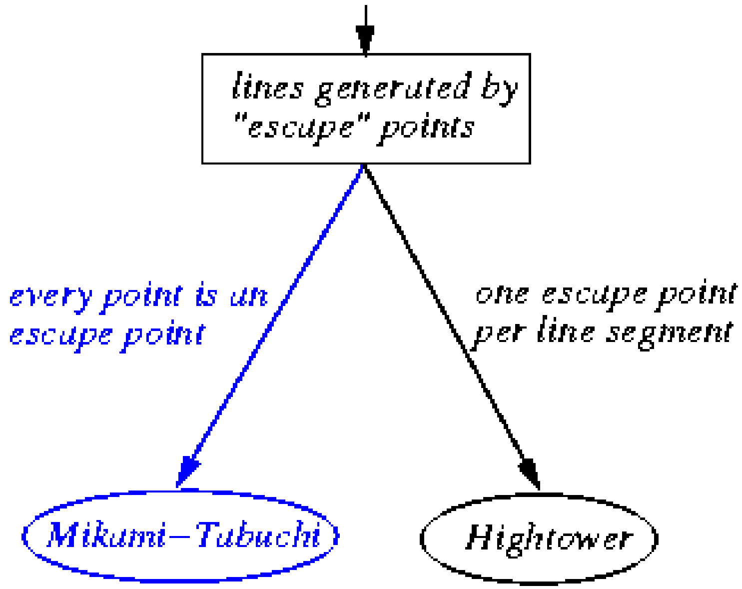
initialize cell weights · *wave propagation* · *first wave reaches the target* · *finding other paths* · *min−cost path found*

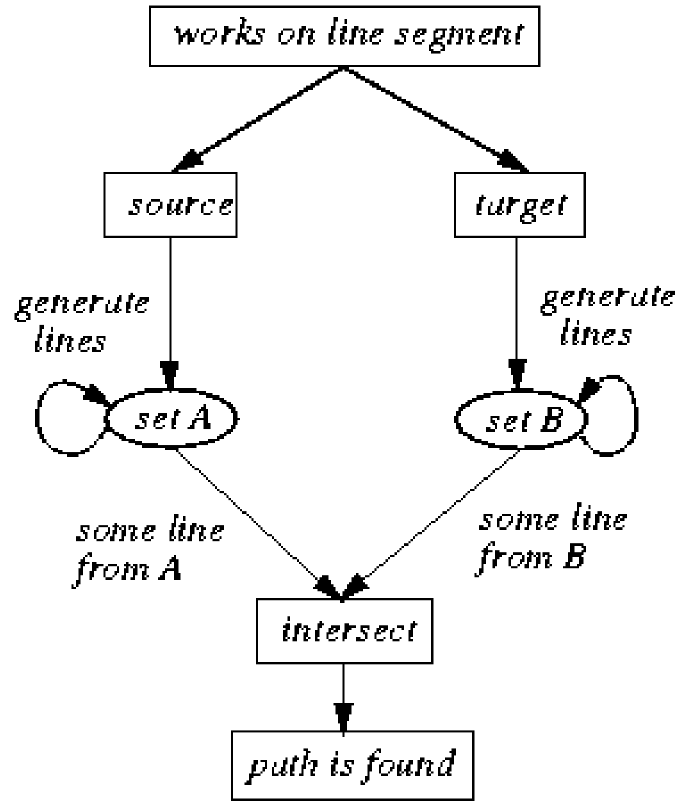
Line-Search Algorithms

- Overcome the drawback of grid representation used by Lee algorithms
  - Time & space complexity for an $M \times N$ grid: $O(MN)$
- Consider the base case
  - if no obstacles, two points S and T are to be connected, then a vertical line passing through S and a horizontal line passing through T naturally intersect.



Features of Line-Search Algorithms

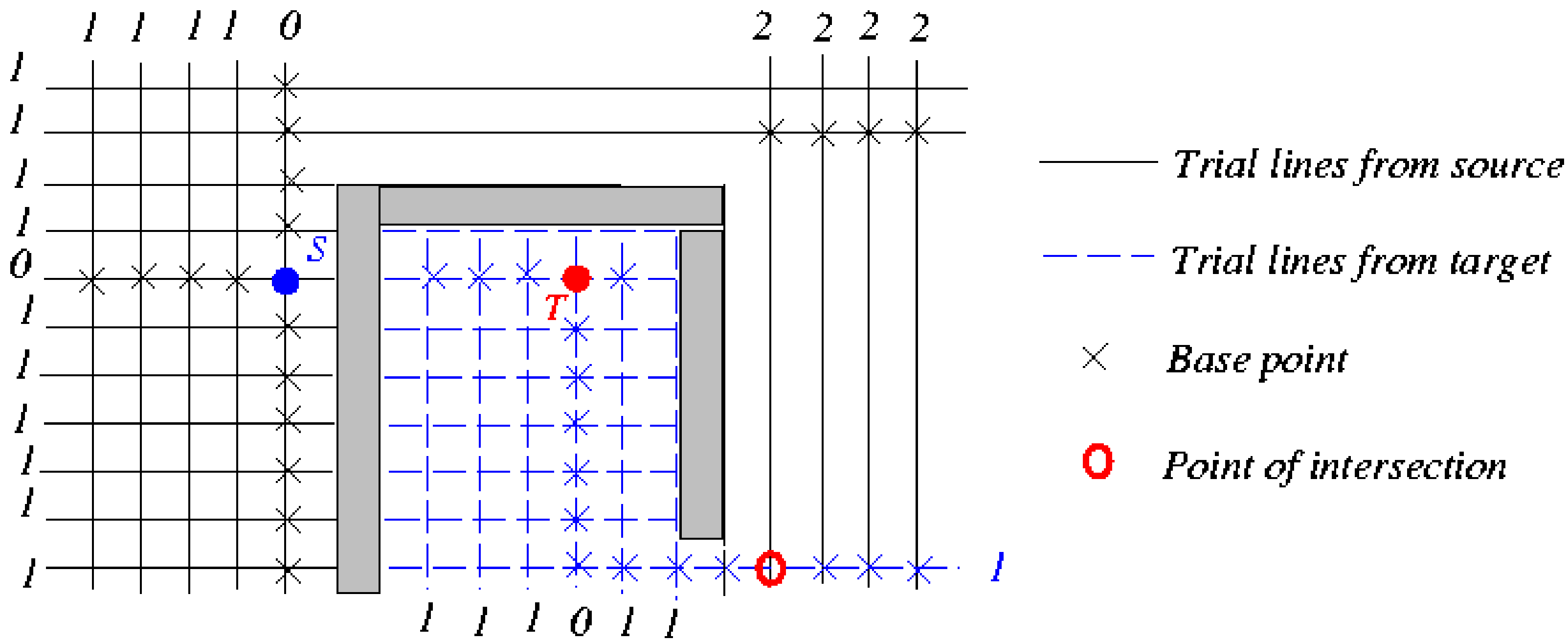




- Time and space complexities: $O(L)$, where L is the # of line segments generated.

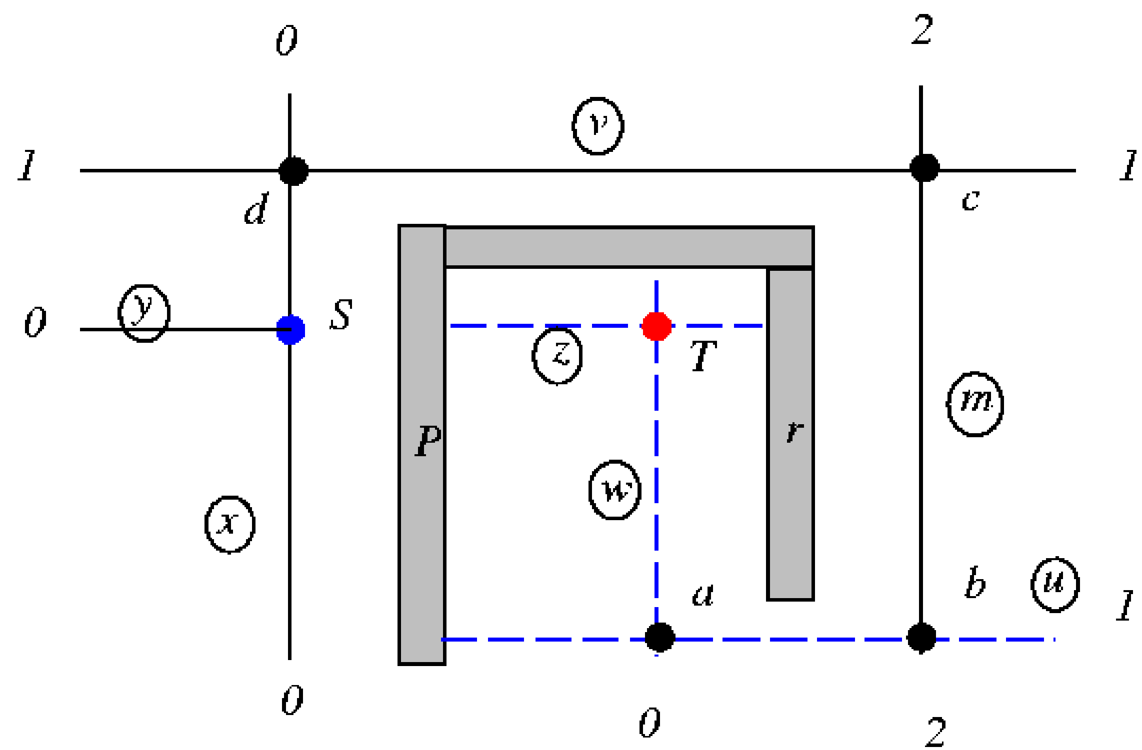
Mikami-Tabuchi's Algorithm

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



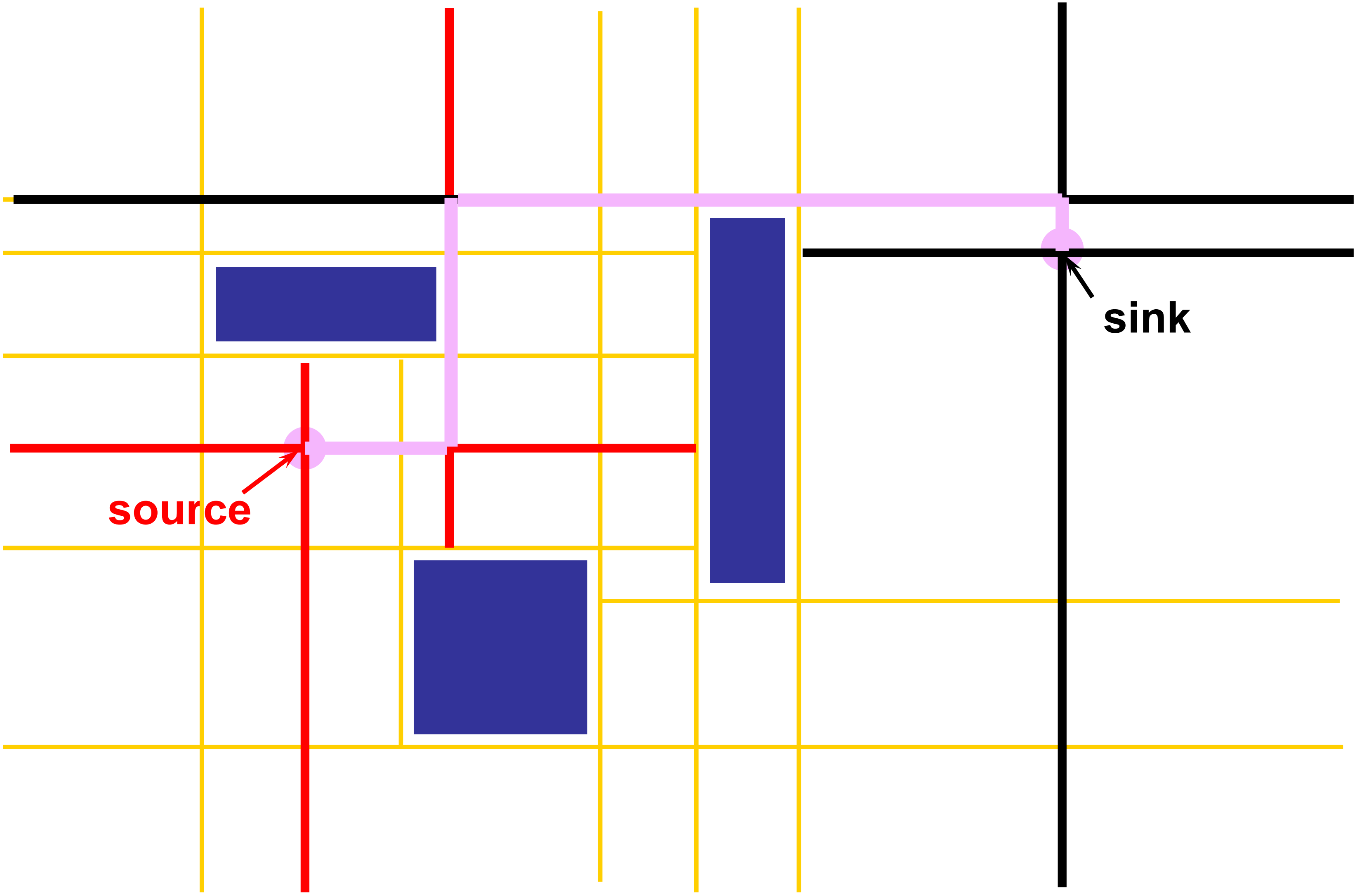
Hightower's Algorithm

- Hightower, "A solution to line-routing problem on the continuous plane," DAC-69.
- A single escape point on each line segment.
- If a line parallels to the blocked cells, the escape point is placed just past the endpoint of the segment.



Slides Courtesy of Prof. H.-R. Jiang

Example: Hightower's Algorithm



Comparison of Algorithms

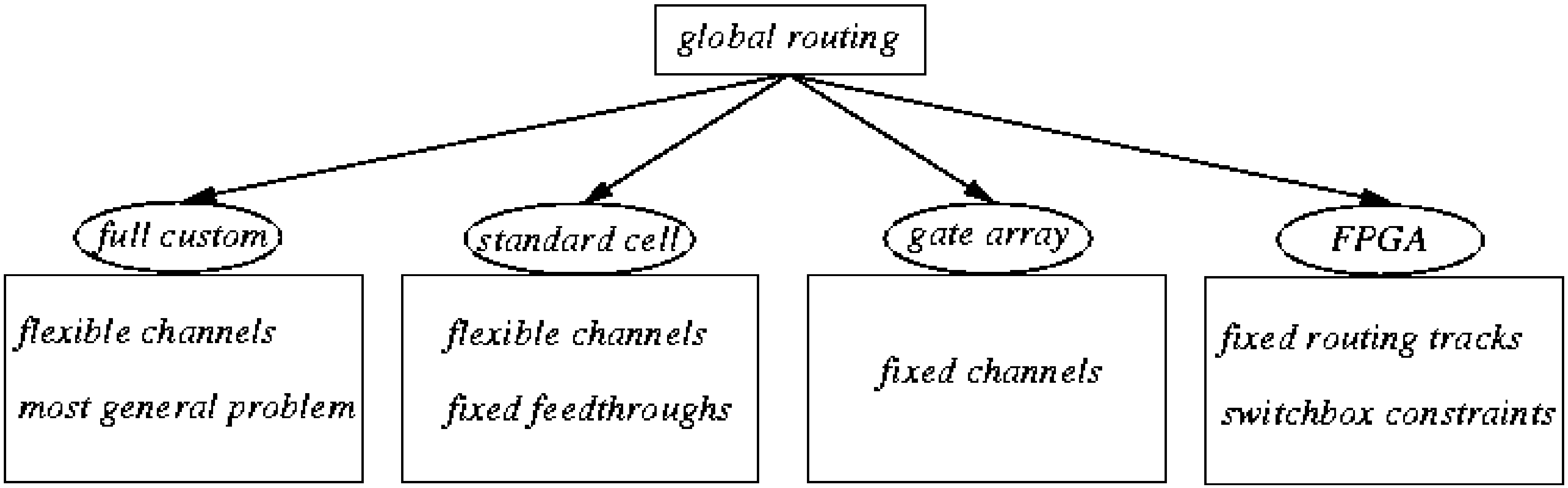
	Maze routing			Line search	
	Lee	Soukup	Hadlock	Mikami	Hightower
Time	$O(MN)$	$O(MN)$	$O(MN)$	$O(L)$	$O(L)$
Space	$O(MN)$	$O(MN)$	$O(MN)$	$O(L)$	$O(L)$
Finds path if one exists?	yes	yes	yes	yes	no
Is the path shortest?	yes	no	yes	no	no
Works on grids or lines?	grid	grid	grid	line	line

- Soukup, Mikami, and Hightower all adopt some sort of line-search operations $\Rightarrow$ cannot guarantee shortest paths.

Global-Routing Problem

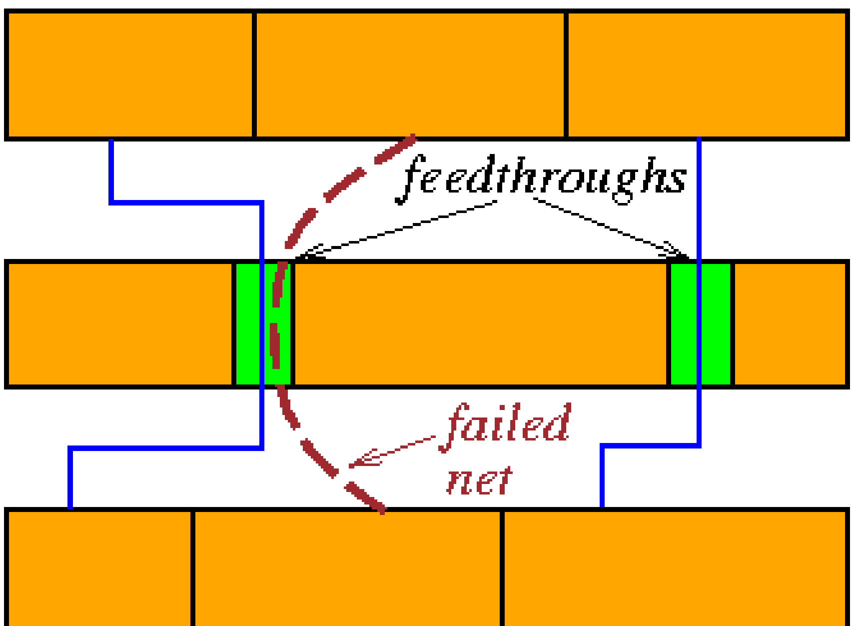
- Given a netlist N={N_1, N_2, …, N_n }, a routing graph $G = (V, E)$, find a Steiner tree T_i for each net N_i, $1 \leq i \leq n$, such that $U(e_j) \leq c(e_j)$, $\forall\ e_j \in E$ and $\sum_{i=1}^{n} L(T_i)$ is minimized, where
  - $c(e_j)$: capacity of edge e_j;
  - x_{ij}=1 if e_j is in T_i; x_{ij} = 0 otherwise;
  - $U(e_j) = \sum_{i=1}^{n} x_{ij}$: # of wires that pass through the channel corresponding to edge e_j;
  - $L(T_i)$: total wirelength of Steiner tree T_i.
- For high-performance, the maximum wirelength ($\max_{i=1}^{n} L(T_i)$) is minimized (or the longest path between two points in T_i is minimized).

Global Routing in different Design Styles



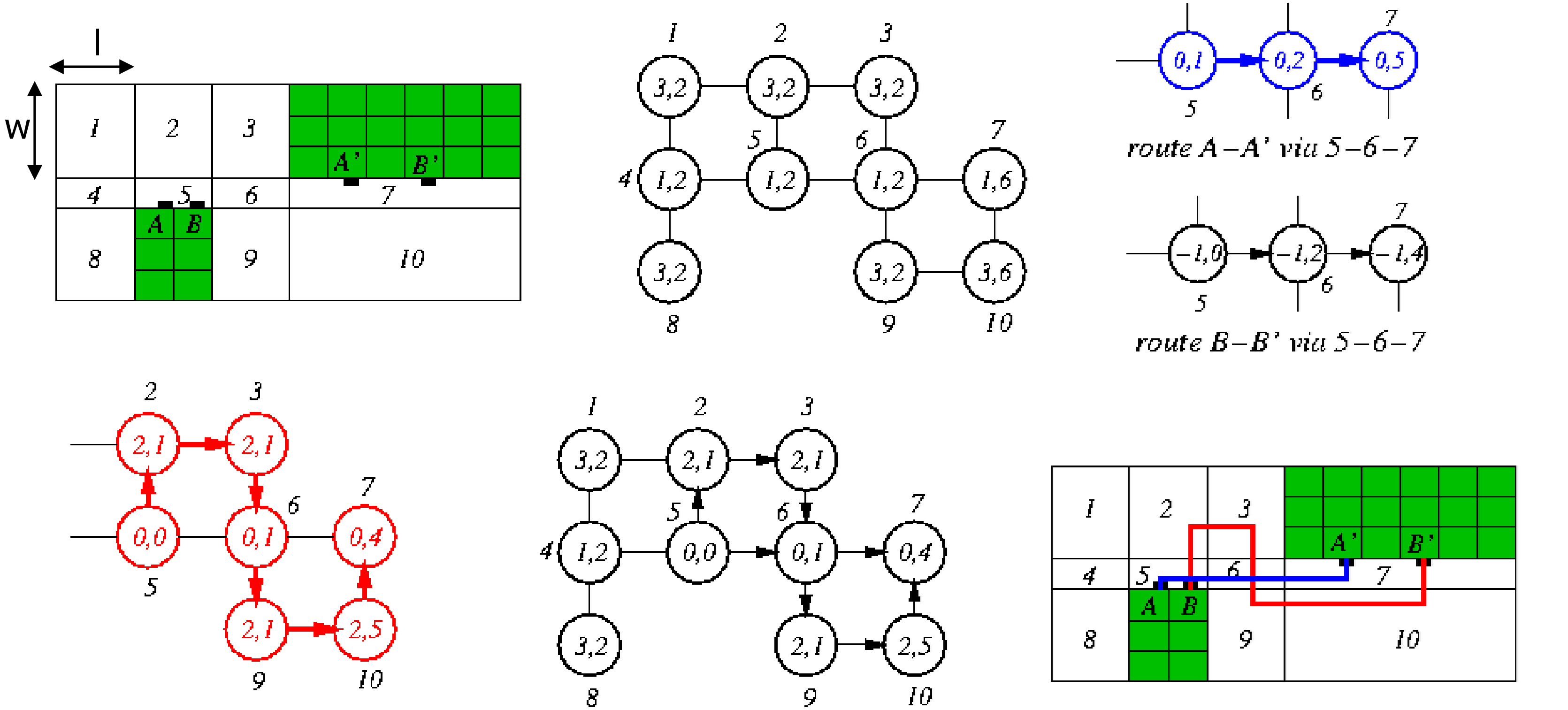
Global Routing in Standard Cell

- Objective
  - Minimize total channel height.
  - Assignment of **feedthrough**: Placement? Global routing?
- For high performance,
  - Minimize the maximum wire length.
  - Minimize the maximum path length.



Global-Routing: Maze Routing

- Routing channels may be modelled by a weighted undirected graph called **channel connectivity graph**.
  - Node ↔ channel; edge ↔ two adjacent channels; capacity: (*width*, *length*)



route B–B' via 5–2–3–6–9–10–7 *updated channel graph* *maze routing for nets A and B*

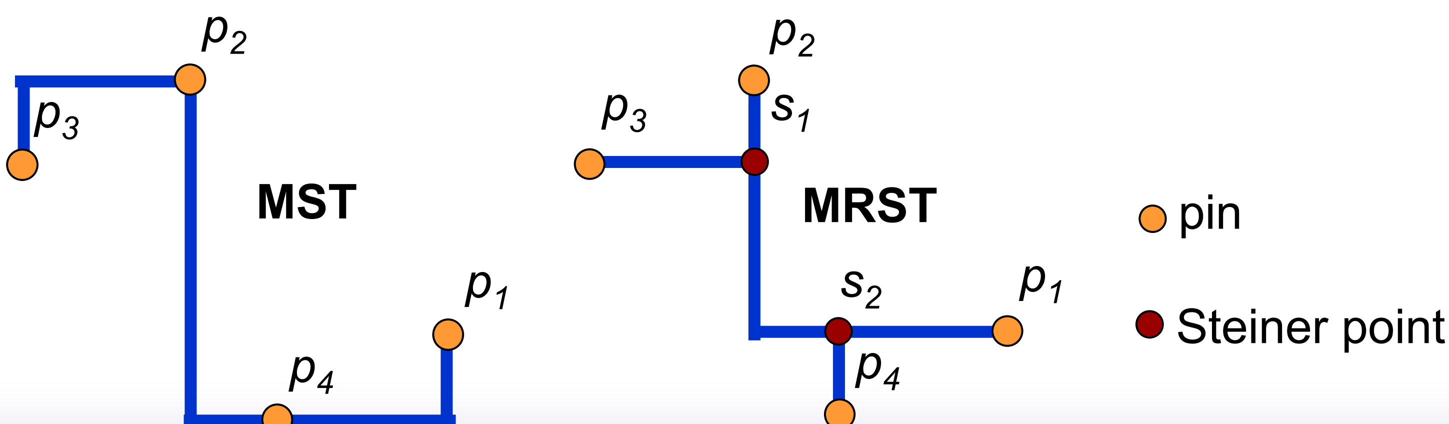
- Update:
  - If a path bends in a channel, both width and length are reduced
  - If a path goes only vertically/horizontally, only length/width is reduced

Routing Tree

- If all nets are two-pin ones, we can apply a general-purpose routing algorithm to handle the problem, such as maze, line-search, and A*-search routing.
- For three or more multi-pin nets, one approach is to *decompose* each net into a set of two-pin connections, and then routes the connections one-by-one.

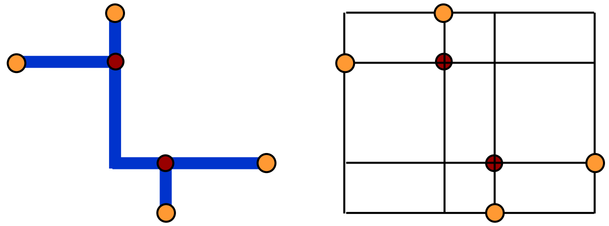
The Routing-Tree Problem

- **Problem:** Given a set of pins of a net, interconnect the pins by a "routing tree."
- **Minimum Spanning Tree (MST):** a minimum-length tree of edges connecting all the pins
- **Minimum Rectilinear Steiner Tree (MRST) Problem:** Given n points in the plane, find a minimum-length tree of rectilinear edges which connects the points.
- $MRST(P) = MST(P \cup S)$, where P and S are the sets of original points and Steiner points, respectively.



Theoretic Results for the MRST Problem

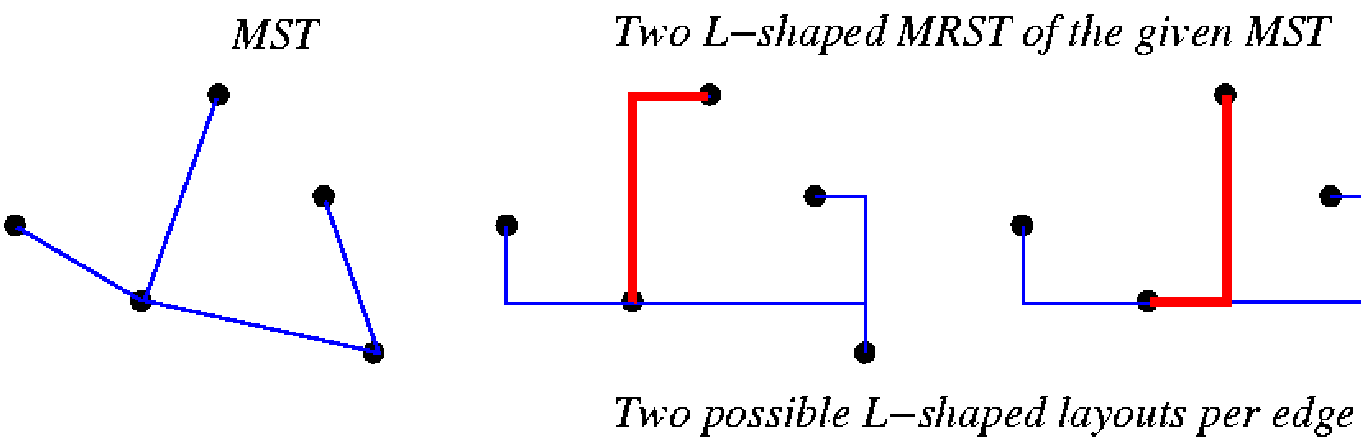
- **Hanan's Thm:** There exists an MRST with all Steiner points (set S) chosen from the intersection points of horizontal and vertical lines drawn points of P.
  - Hanan, "On Steiner's problem with rectilinear distance," *SIAM J. Applied Math.*, 1966.



- **Hwang's Theorem:** For any point set P, $\frac{Cost(MST(P))}{Cost(MRST(P))} \leq \frac{3}{2}$.
  - Hwang, "On Steiner minimal tree with rectilinear distance," *SIAM J. Applied Math.*, 1976.
- Better approximation algorithm with the performance bound 61/48
  - Foessmeier *et al*, "Fast approximation algorithm for the rectilinear Steiner problem," Wilhelm Schickard-Institut für Informatik, TR WSI-93-14, 93.

Coping with the MRST Problem

- Ho, Vijayan, and Wong, “New algorithms for the rectilinear Steiner problem,” TCAD-90.
  1. Construct an MRST from an MST.
  2. Each edge is straight or L-shaped.
  3. Maximize overlaps by dynamic programming.
- About 8% smaller than *Cost*(*MST*).

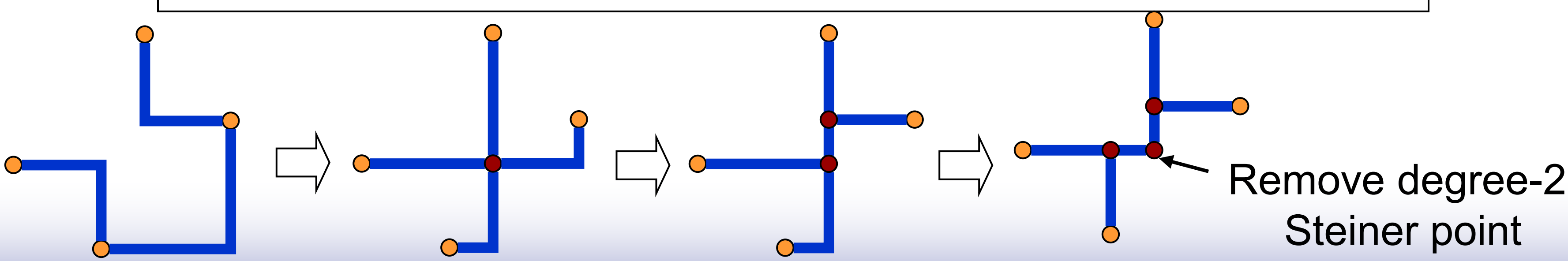


Iterated 1-Steiner Heuristic for MRST

- Kahng & Robins, "A new class of Steiner tree heuristics with good performance: the iterated 1-Steiner approach," *ICCAD*-90.

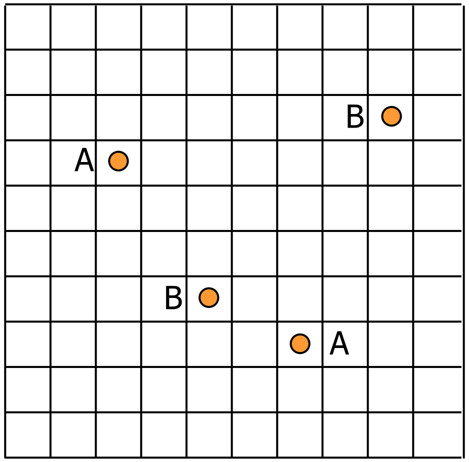
Algorithm: Iterated_1-Steiner(*P*)
P: set *P* of *n* points.

begin
$S \leftarrow \emptyset$;
  /* $H(P \cup S)$: set of Hanan points */
  /* $\Delta MST(A, B) = Cost(MST(A)) - Cost(MST(A \cup B))$ */
while ($Cand \leftarrow \{x \in H(P \cup S) \mid \Delta MST(P \cup S, \{x\}) > 0\} \neq \emptyset$) **do**
  Find $x \in Cand$ and which maximizes $\Delta MST(P \cup S, \{x\})$;
  $S \leftarrow S \cup \{x\}$;
  Remove points in S which have degree ≤ 2 in $MST(P \cup S)$;
Output $MST(P \cup S)$;
end

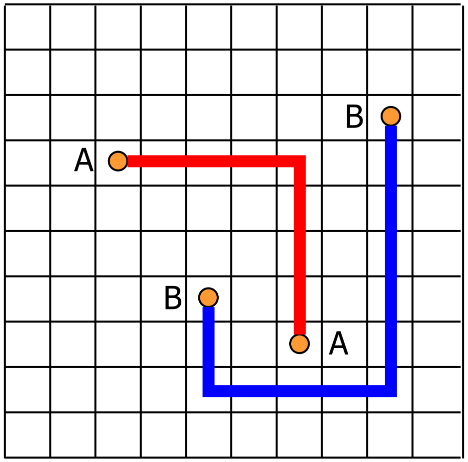


Net Ordering

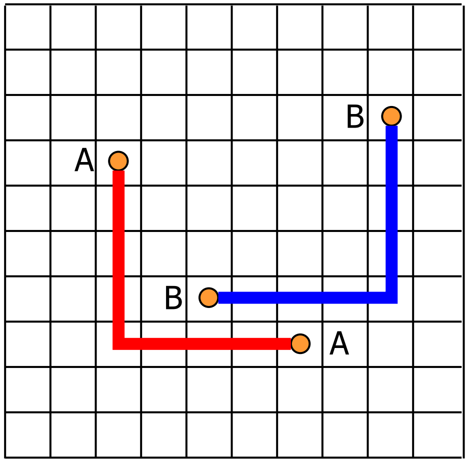
- Net ordering greatly affects routing solutions
- Finding the optimal net ordering is proven to be NP-hard
  - Abel, "On the ordering of connections for automatic wire routing," TC-1972.
- In the example, we should route net *b* before net *a*



A one-layer routing instance with nets *A* and *B*



Route *A* before *B*



Route *B* before *A*

Net Ordering (cont'd)

- Order the nets in the ascending order of the # of pins within their bounding boxes.
- Order the nets in the ascending (descending) order of their lengths if routability (timing) is the most critical metric
- Order the nets based on their timing criticality.

Rip-Up and Re-routing

- Rip-up and re-routing is required if a global or detailed router fails in routing all nets.
- Approaches: the manual approach? the automatic procedure?
- Two steps in rip-up and re-routing
  1. Identify bottleneck regions, rip off some already routed nets.
  2. Route the blocked connections, and re-route the ripped-up connections.
- Repeat the above steps until all connections are routed or a time limit is exceeded.