IIT KHARAGPUR

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Lecture 15: GRID ROUTING (PART 1)

PROF. INDRANIL SENGUPTA
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Introduction

- In the VLSI design cycle, *routing* follows *cell placement*.
- Once routing is completed, precise paths are defined on the layout surface, on which conductors carrying electrical signals are run.
- Routing takes up almost 30% of the design time, and a large percentage of layout area.
  – One main objective is to minimize the area required for routing.

Types of Routing?

- Given a set of blocks placed on a layout surface and defined pin locations:
  - Given a set of obstacles and a set of pins to connect, determine a solution to interconnect the pins on a single layer (**GRID ROUTING**).
  - Determine the approximate regions through which each interconnection net should pass (**GLOBAL ROUTING**).
  - For each routing region, complete the interconnection by assigning horizontal and vertical metal line segments on the layout surface (**DETAILED ROUTING**).

The General Routing Problem

- Given:
  - A set of blocks with pins on the boundaries.
  - A set of signal nets.
  - Locations of the blocks on the layout surface.
- Objective:
  - Find suitable paths on the available layout space, on which wires are run to connect the desired set of pins.
  - Minimize some given objective function, subject to given constraints.

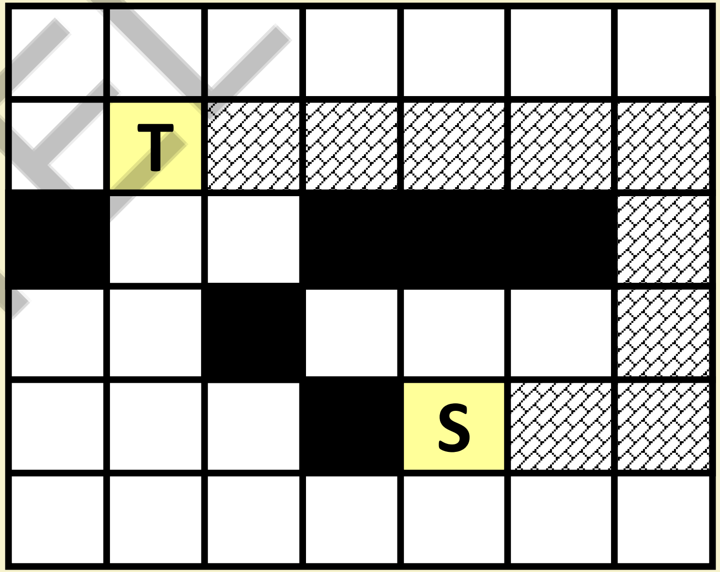
- Types of constraints:
  - Minimum width of routing wires.
  - Minimum separation between adjacent wires.
  - Number of routing layers available.
  - Timing constraints.

GRID ROUTING

Basic Concept

- The layout surface is assumed to be made up of a rectangular array of grid cells.
- Some of the grid cells act as obstacles.
  - Blocks that are placed on the surface.
  - Some nets that are already laid out.
- Objective is to find out a single-layer path (sequence of grid cells) for connecting two points belonging to the same net.

- Two broad classes of grid routing algorithms:
    1. Maze routing algorithms.
    2. Line search algorithms.



Grid Routing Algorithms

1. Maze running algorithm
   - Lee's algorithm
   - Hadlock's algorithm
2. Line search algorithm
   - Mikami-Tabuchi's algorithm
   - Hightower's algorithm
3. Steiner tree algorithm

Maze Running Algorithms

- The entire routing surface is represented by a 2-D array of grid cells.
  - All pins, wires and edges of bounding boxes that enclose the blocks are aligned with respect to the grid lines.
  - The segments on which wires run are also aligned.
  - The size of grid cells is appropriately defined.
    - Wires belonging to different nets can be routed through adjacent cells without violating the width and spacing rules.
- Maze routers connect a single pair of points at a time.
  - By finding a sequence of adjacent cells from one point to the other.

Lee's Algorithm

- The most common maze routing algorithm.
- Characteristics:
  - If a path exists between a pair of points S and T, it is definitely found.
  - It always finds the shortest path.
  - Uses breadth-first search.
- Time and space complexities are $O(N^2)$ for a grid of dimension $N \times N$.

Phase 1 of Lee's Algorithm

- Wave propagation phase
  - Iterative process.
  - During step i, non-blocking grid cells at Manhattan distance of i from grid cell S are all labeled with i.
  - Labeling continues until the target grid cell T is marked in step L.
    - L is the length of the shortest path.
  - The process fails if:
    - T is not reached and no new grid cells can be labeled during step i.
    - T is not reached and i equals M, some upper bound on the path length.

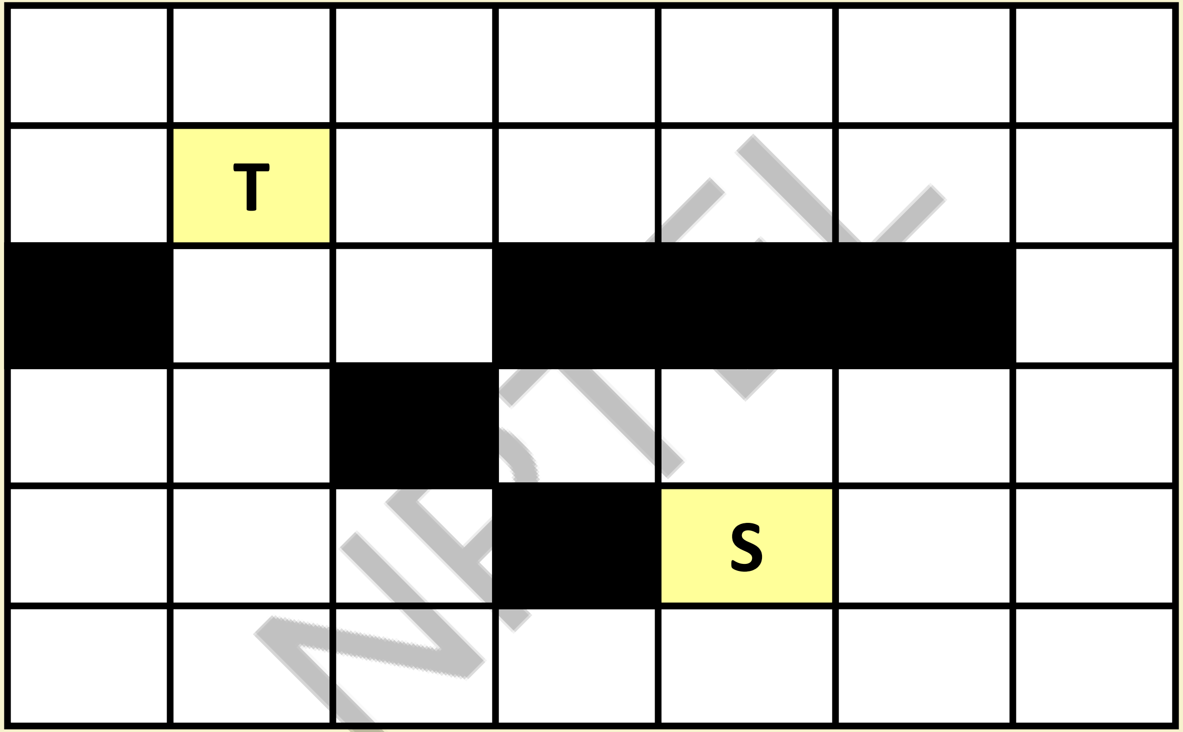
Phase 2 of Lee's Algorithm

- Retrace phase
  - Systematically backtrack from the target cell T back towards the source cell S.
  - If T was reached during step i, then at least one grid cell adjacent to it will be labeled i-1, and so on.
  - By tracing the numbered cells in descending order, we can reach S following the shortest path.
    - There is a choice of cells that can be made in general.
    - In practice, the rule of thumb is not to change the direction of retrace unless one has to do so.
    - Minimizes number of bends.

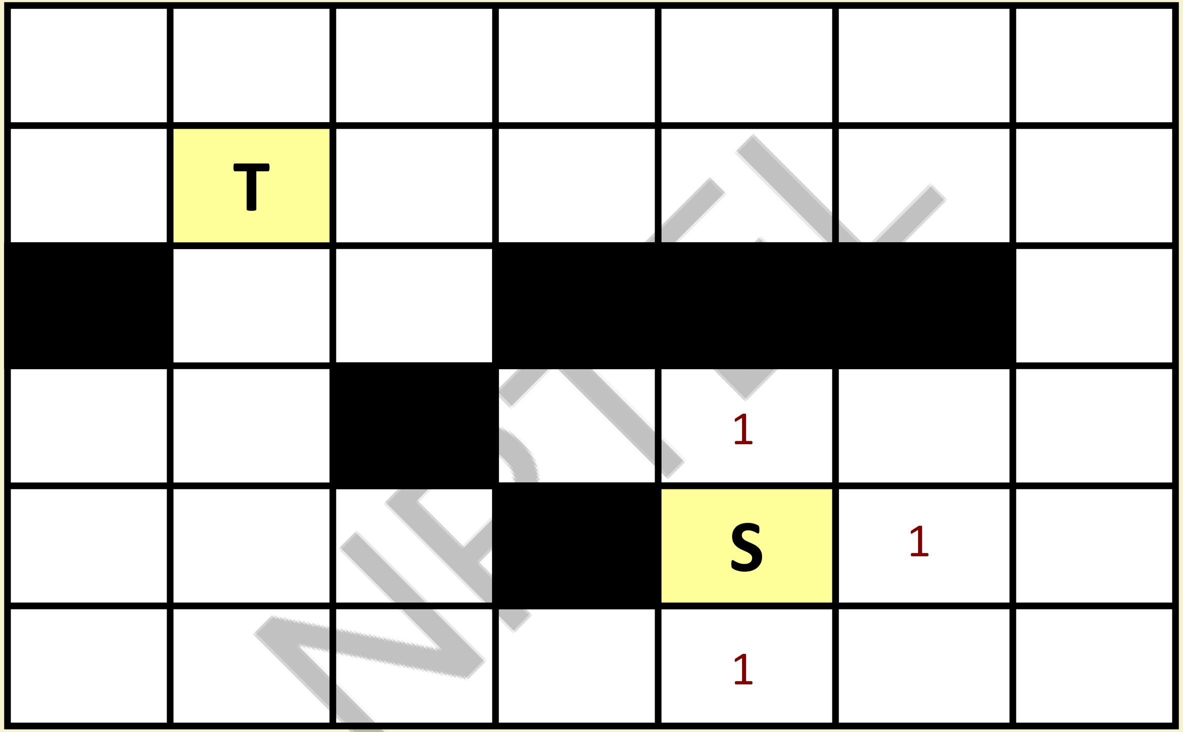
Phase 3 of Lee's Algorithm

- Label clearance
  - All labeled cells except those corresponding to the path just found are cleared.
  - Cells along the path are marked as obstacles.
  - Search complexity is as involved as the wave propagation step itself.

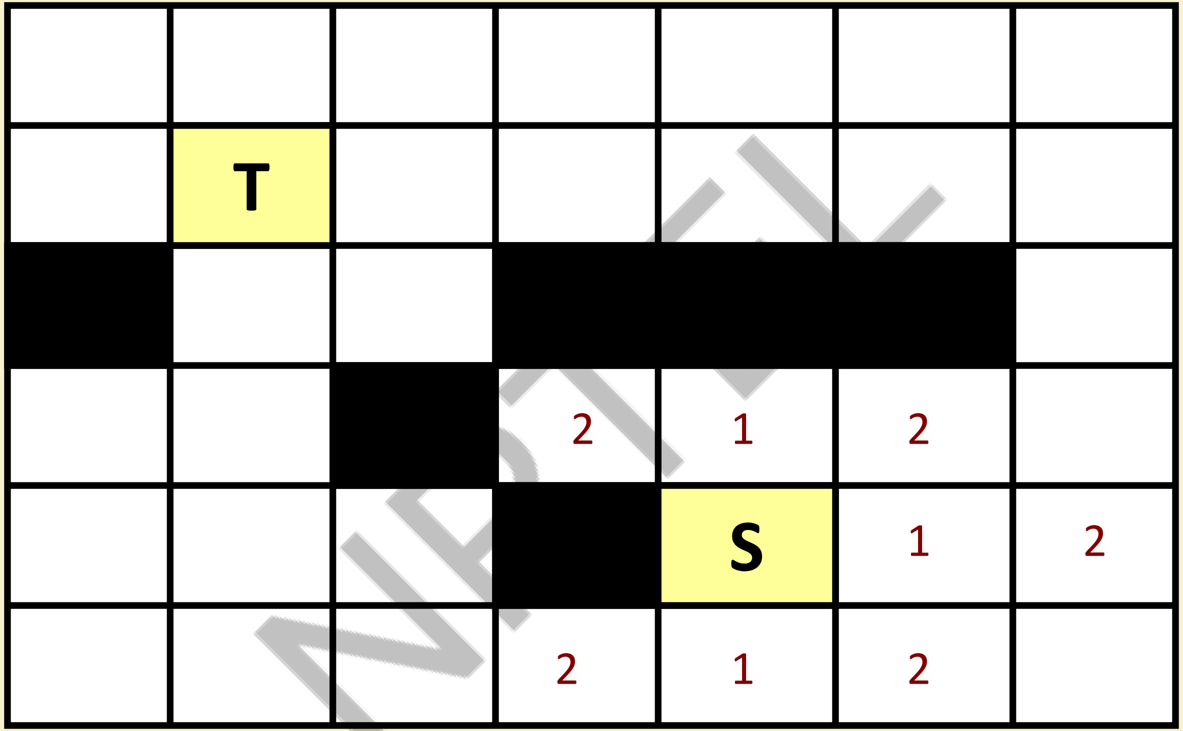
Initial routing problem



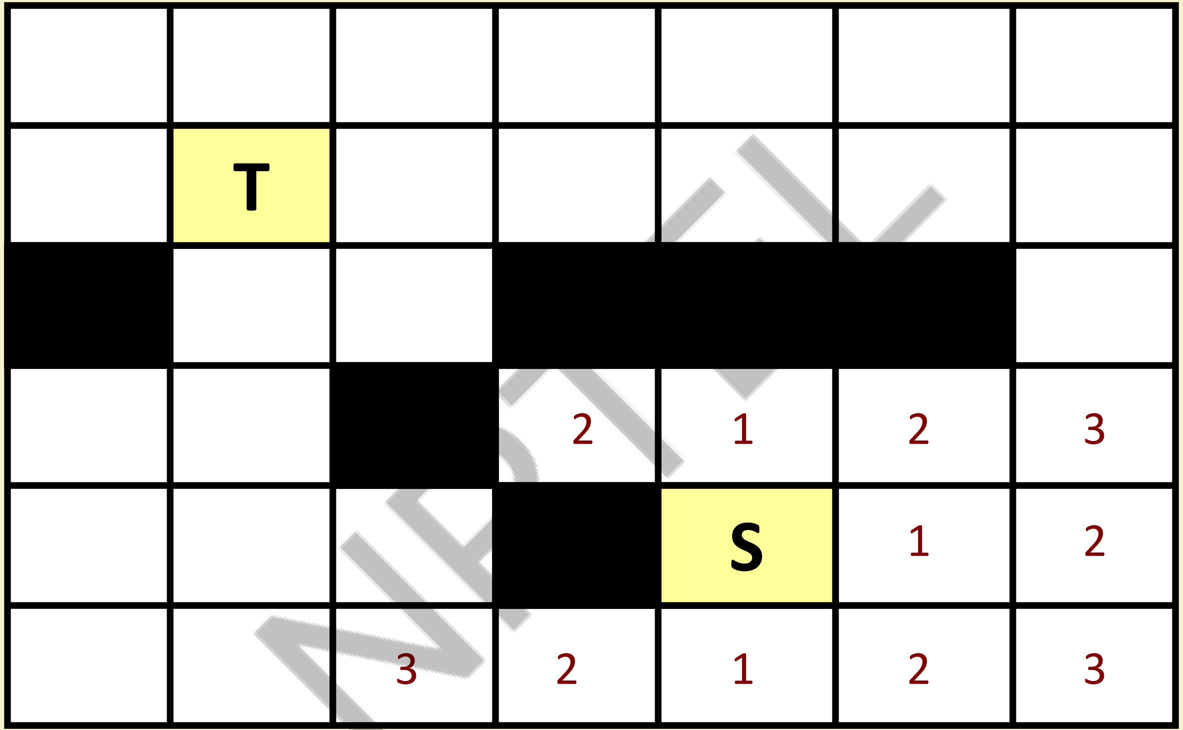
Phase 1
(i = 1)



Phase 1
(i = 2)



Phase 1
(i = 3)



Phase 1
(i = 4)

	T					
						4
			2	1	2	3
		4		S	1	2
	4	3	2	1	2	3

Phase 1
(i = 5)

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

Phase 1
(i = 6)

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

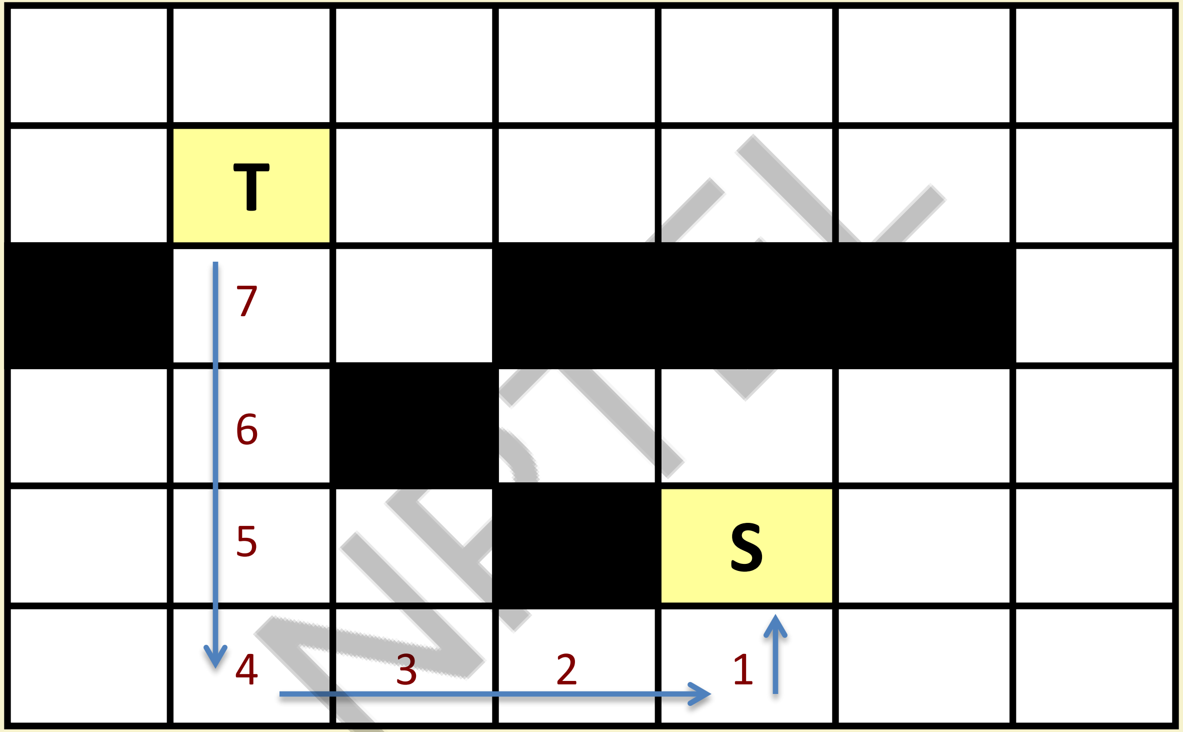
Phase 1
(i = 7)

					7	6
	T			7	6	5
	7					4
7	6		2	1	2	3
6	5	4		S	1	2
5	4	3	2	1	2	3

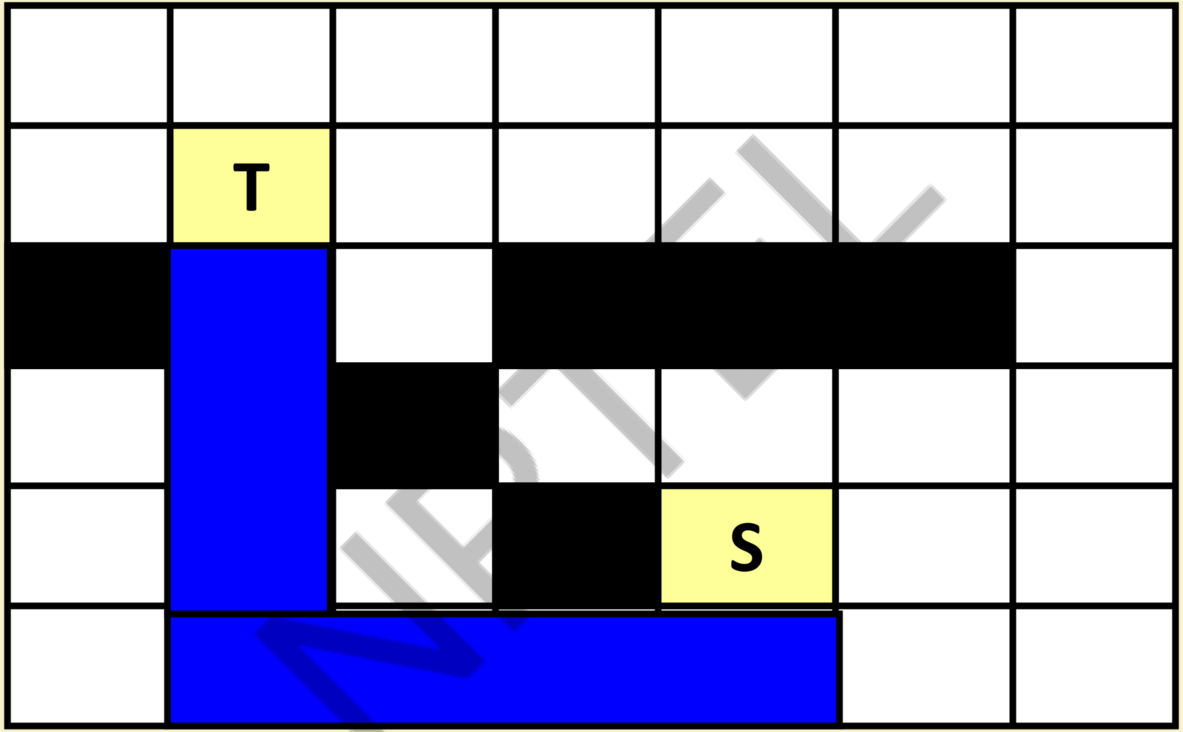
Phase 2 (RETRACE)

					7	6
	T			7	6	5
	7					4
7	6		2	1	2	3
6	5	4		S	1	2
5	4	3	2	1	2	3

Phase 3
(CLEAR)



Phase 3
(MARK)



END OF LECTURE 15

IIT KHARAGPUR

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Lecture 16: GRID ROUTING (PART 2)

PROF. INDRANIL SENGUPTA
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Lee’s Algorithm (contd.)

					7	6
	T			7	6	5
	7					4
7	6		2	1	2	3
6	5	4		**S**	1	2
5	4	3	2	1	2	3

- Memory Requirement
  - Each cell needs to store a number between 1 and L, where L is some bound on the maximum path length.
    - For M x N grid, L can be at most M+N-1.
  - One bit combination to denote empty cell.
  - One bit combination to denote obstacles.

$$\lceil \log_2(L+2) \rceil \text{ bits per cell}$$

- Examples:
  1. 2000 x 2000 grid
     - B = $\log_2 4001 = 12$
     - Memory required = 2000 x 2000 x 12 bits = 6 Mbytes
  2. 3000 x 3000 grid
     - B = $\log_2 6001 = 13$
     - Memory required = 3000 x 3000 x 13 bits = 14.6 Mbytes
  3. 4000 x 4000 grid
     - B = $\log_2 8001 = 13$
     - Memory required = 4000 x 4000 x 13 bits = 26 Mbytes

- <u>Improvements</u>:
  - Instead of using the sequence 1,2,3,4,5,….. for numbering the cells, the sequence 1,2,3,1,2,3,… is used.
    - For a cell, labels of predecessors and successors are different. So tracing back is easy.

      $\lceil \log_2(3+2) \rceil$ = 3 bits per cell.

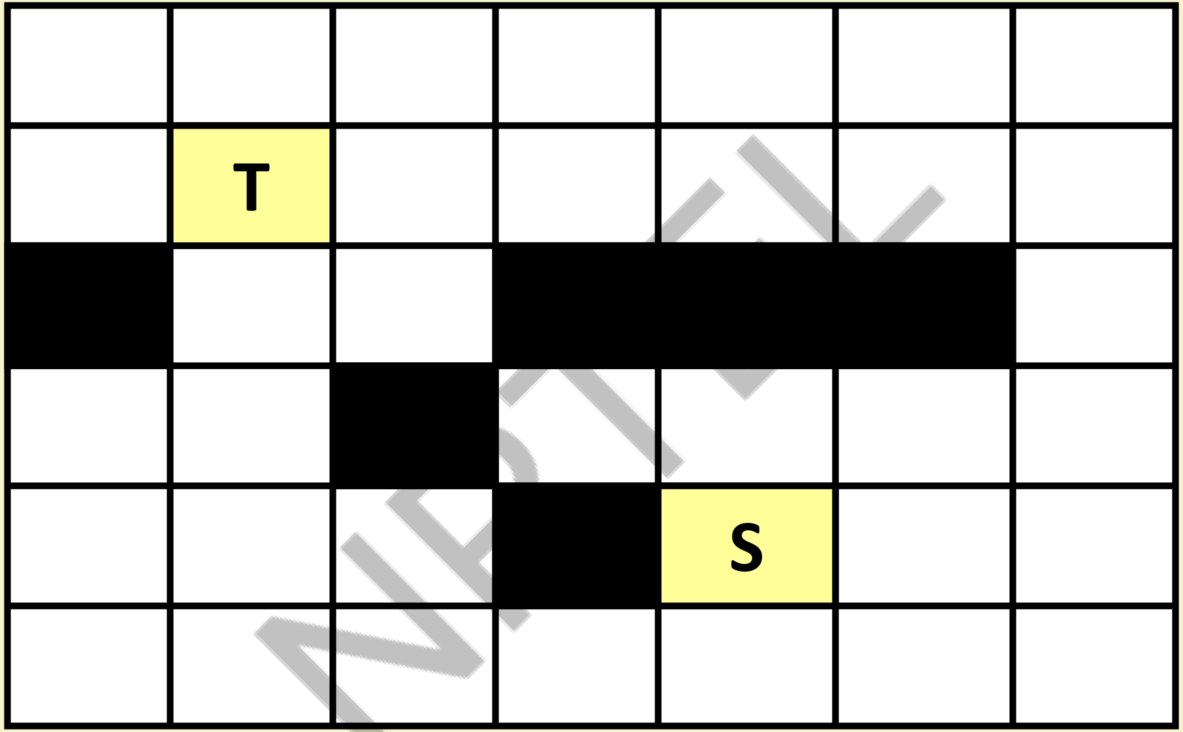
1.5 Mbytes for 2000 x 2000 grid

  - Use the sequence 0,0,1,1,0,0,1,1,…..
    - Predecessors and successors are again different.

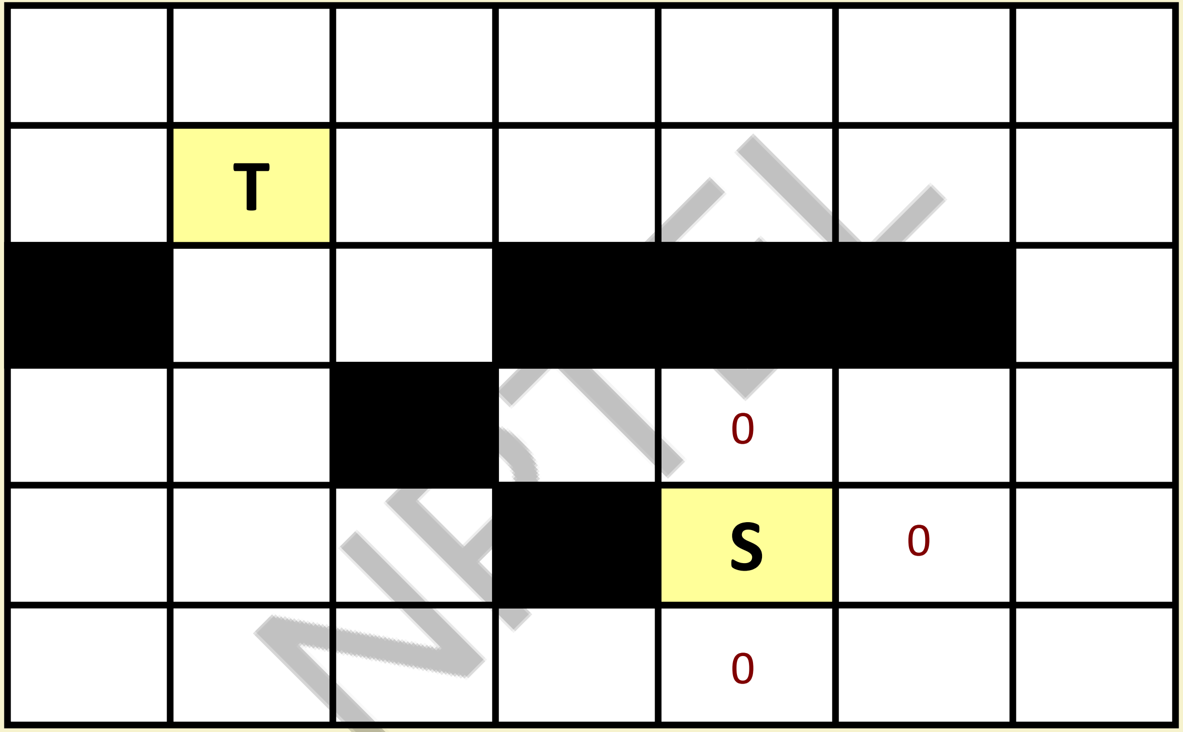
      $\lceil \log_2(2+2) \rceil$ = 2 bits per cell.

1.0 Mbyte for 2000 x 2000 grid

Initial routing problem

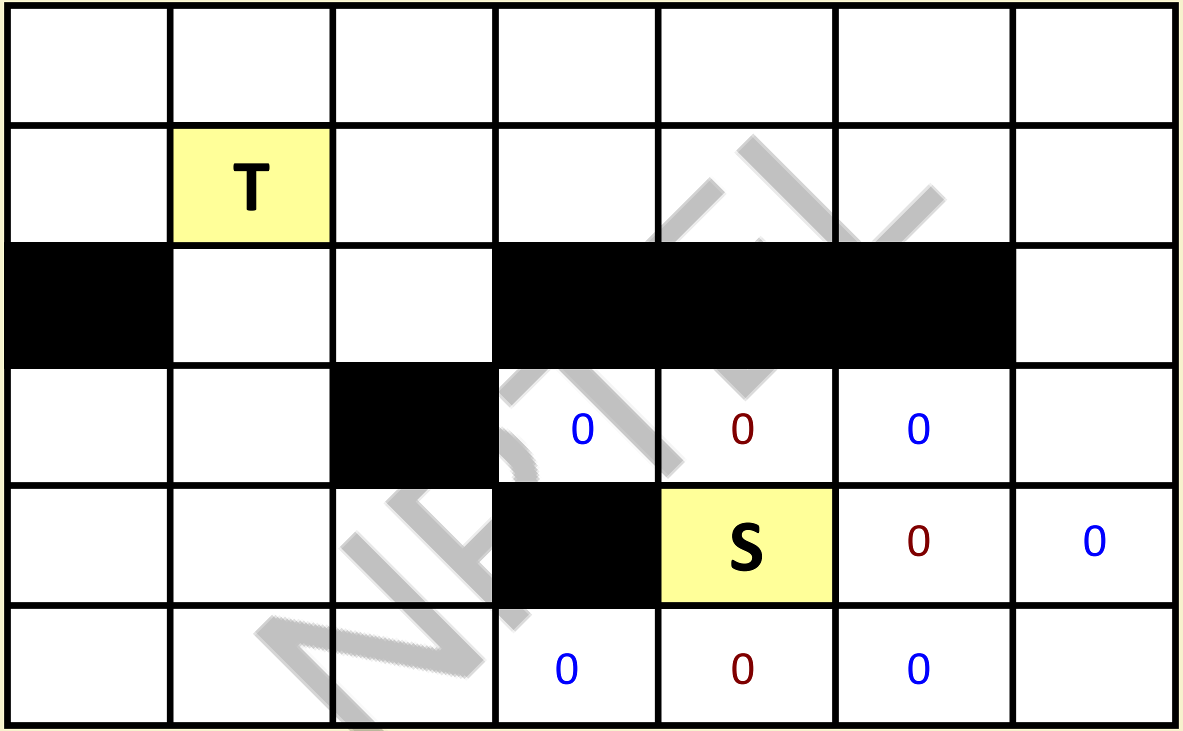


Label
0



Label

00



Label
001

	T					
			0	0	0	1
				S	0	0
		1	0	0	0	1

Label
0011

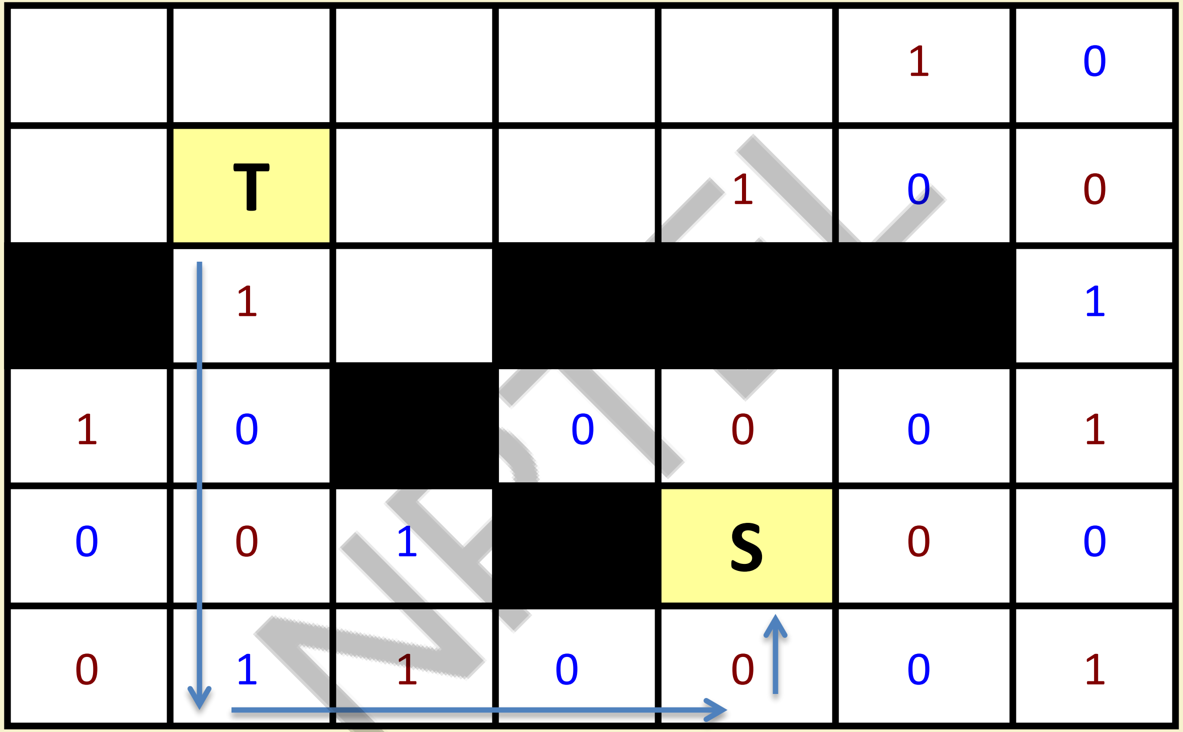
	T					
						1
			0	0	0	1
		1		S	0	0
	1	1	0	0	0	1

Label
0011001

					1	0
	T			1	0	0
	1					1
1	0		0	0	0	1
0	0	1		S	0	0
0	1	1	0	0	0	1

Retrace

0011001

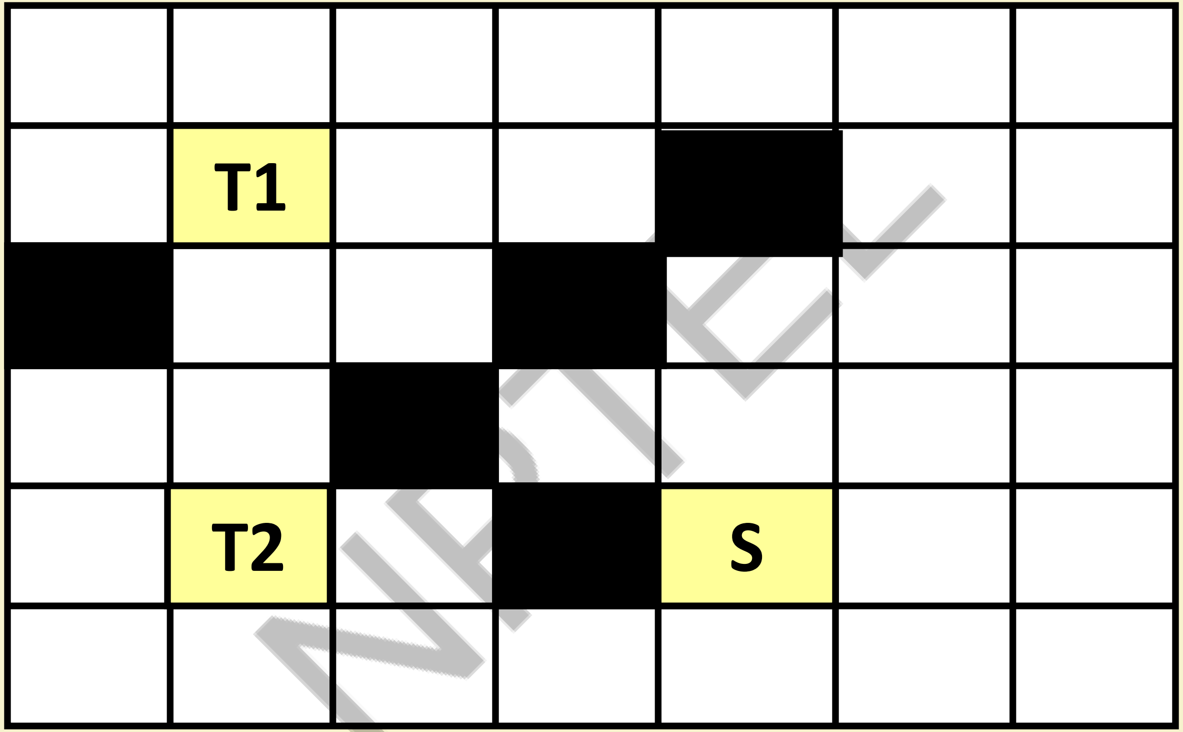


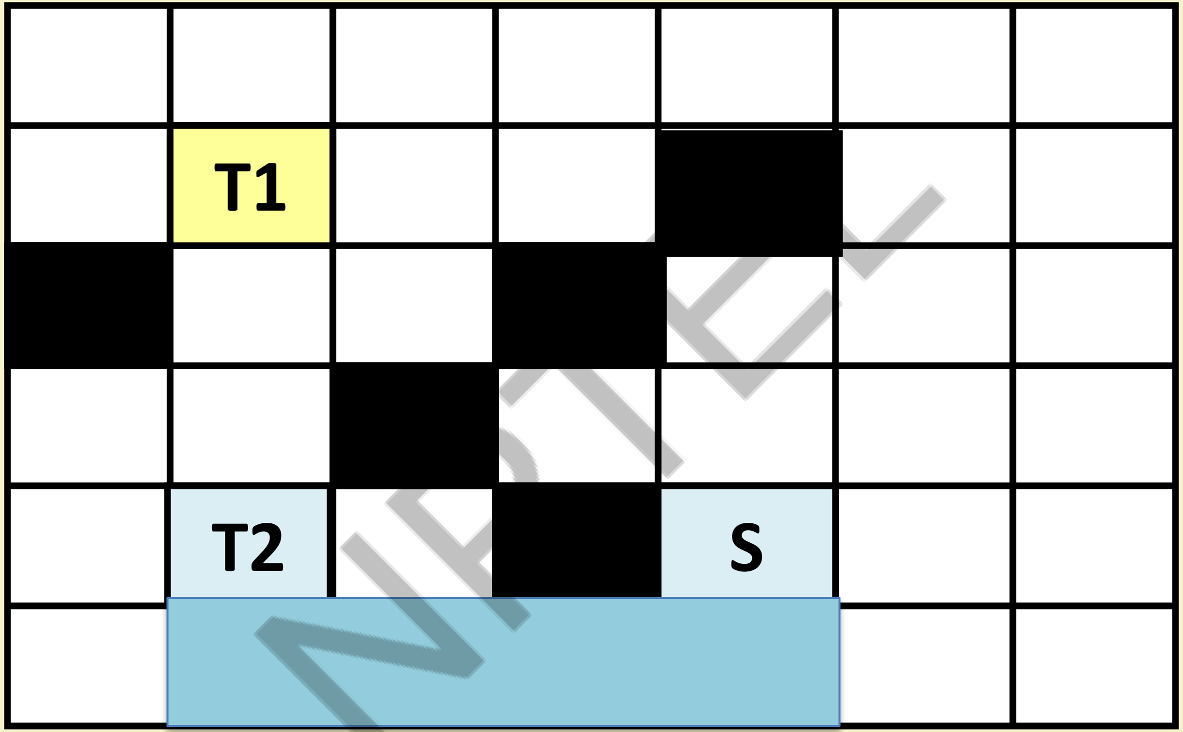
Reducing Running Time

- Starting point selection
  - Choose the starting point as the one that is farthest from the center of the grid.
- Double fan-out
  - Propagate waves from both the source and the target cells.
  - Labeling continues until the wavefronts touch.
- Framing
  - An artificial boundary is considered outside the terminal pairs to be connected.
  - 10-20% larger than the smallest bounding box.

Connecting Multi-point Nets

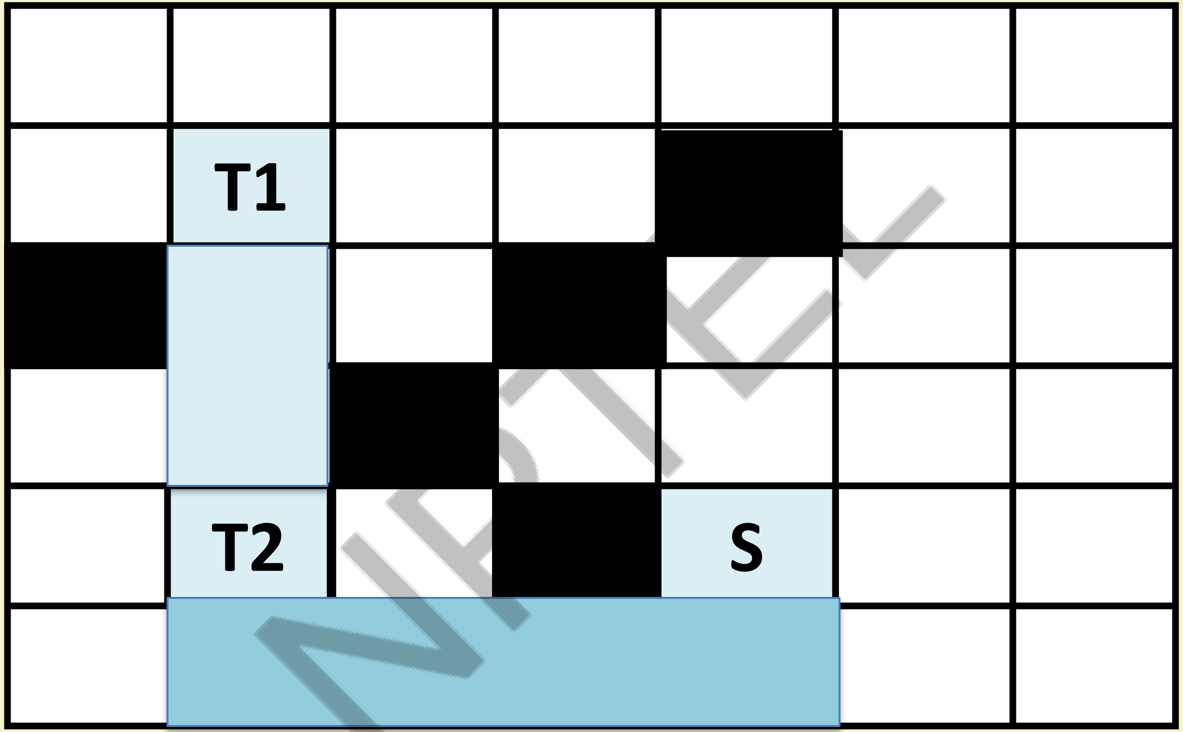
- A multi-pin net consists of three or more terminal points to be connected.
- Extension of Lee's algorithm:
  - One of the terminals of the net is treated as source, and the rest as targets.
  - A wave is propagated from the source until one of the targets is reached.
  - All the cells in the determined path are next labeled as source cells, and the remaining unconnected terminals as targets.
  - Process continues.


T1
T2
S


T1
T2
S

	T1					
	1			1		
1	T2	1		S	1	
1					1	

	T1					
	2			2		
2	1		2	1	2	
1	T2	1		S	1	2
1					1	2


T1
T2
S

END OF LECTURE 16

IIT KHARAGPUR

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Lecture 17: GRID ROUTING (PART 3)

PROF. INDRANIL SENGUPTA
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

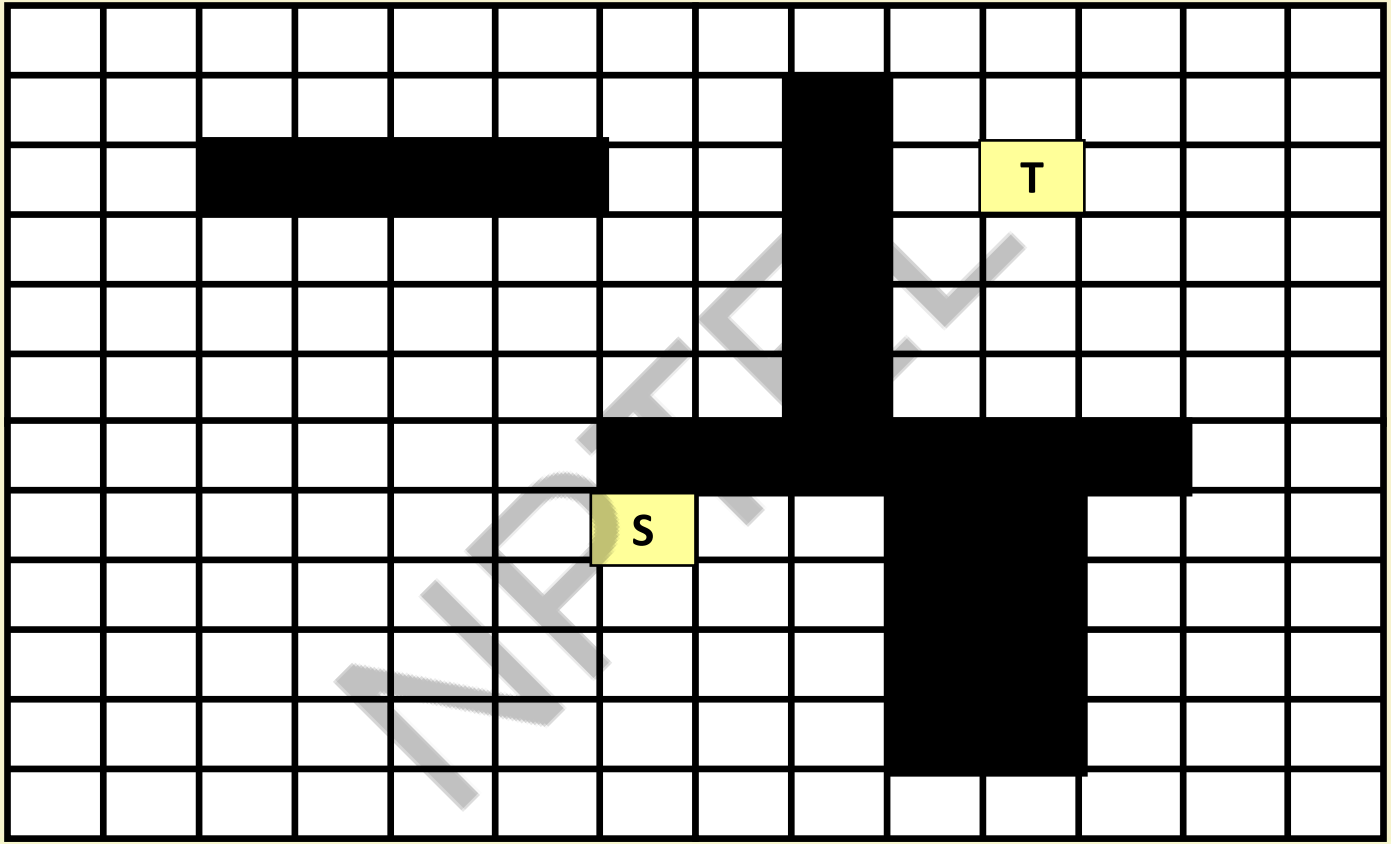
Hadlock's Algorithm

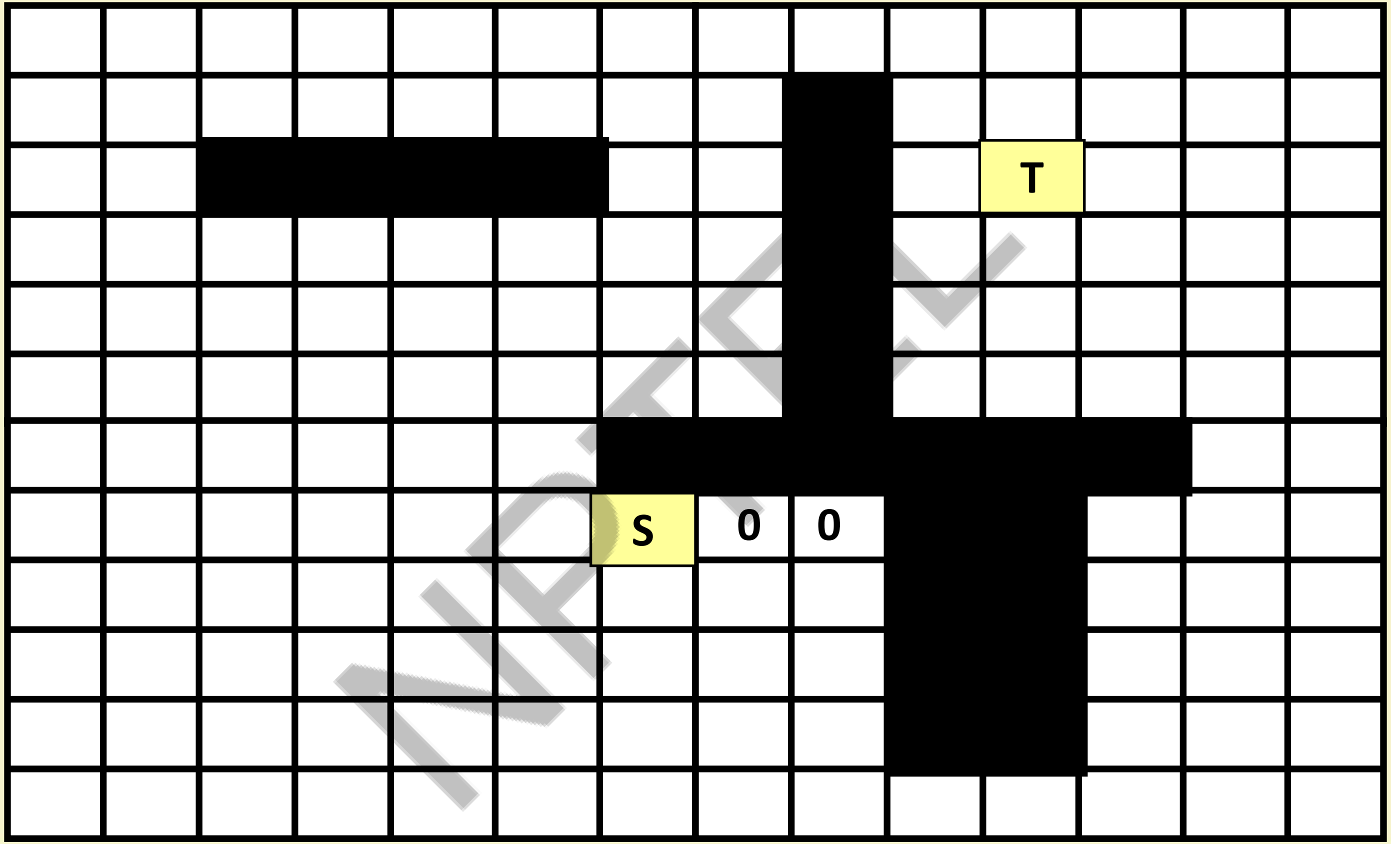
- Uses a new method for cell labeling called detour numbers.
  - A goal directed search method.
  - The detour number $d(P)$ of a path P connecting two cells S and T is defined as the number of grid cells directed away from its target T.
  - The length of the path P is given by

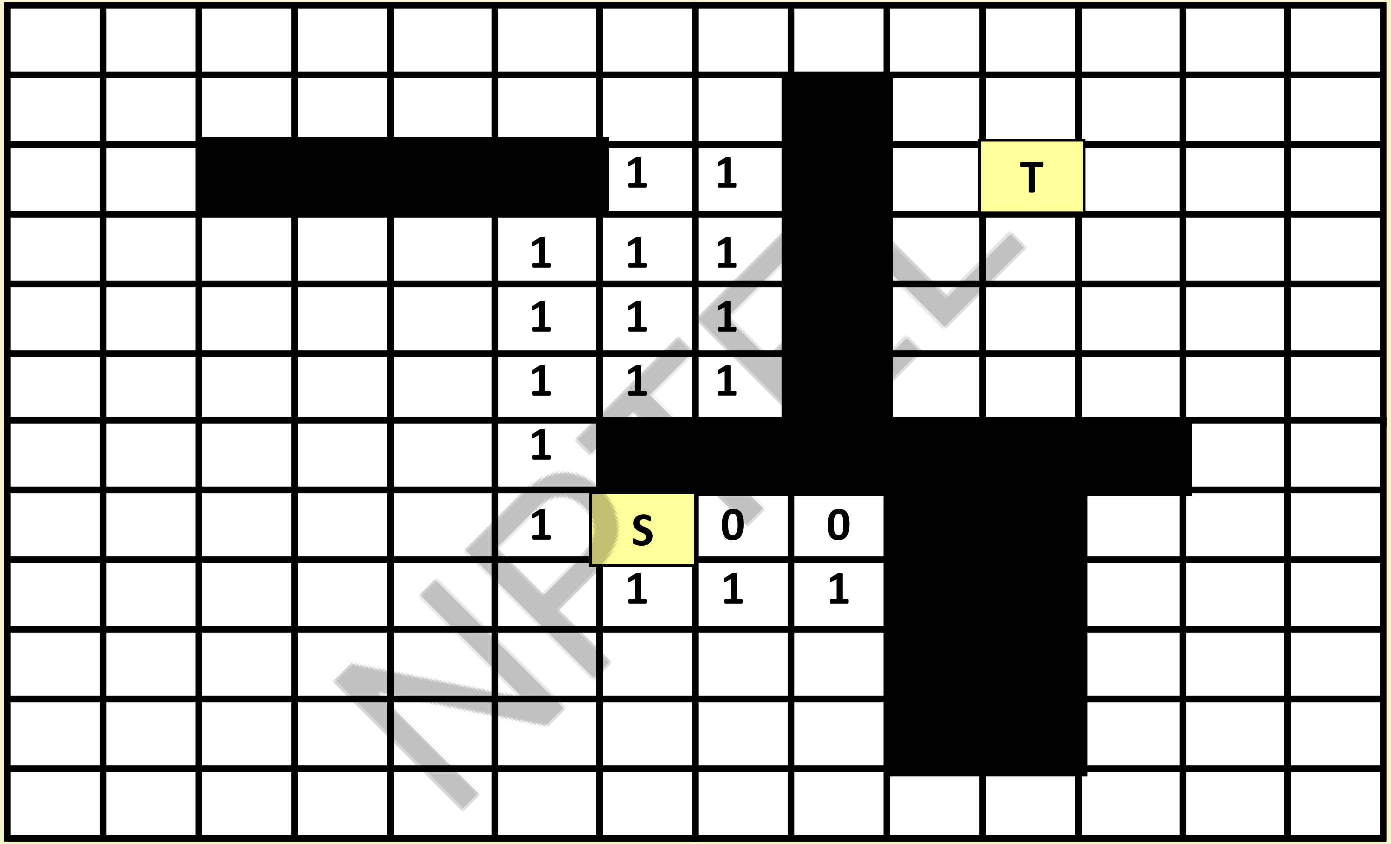
$$len(P) = MD\ (S,T) + 2\ d(P)$$

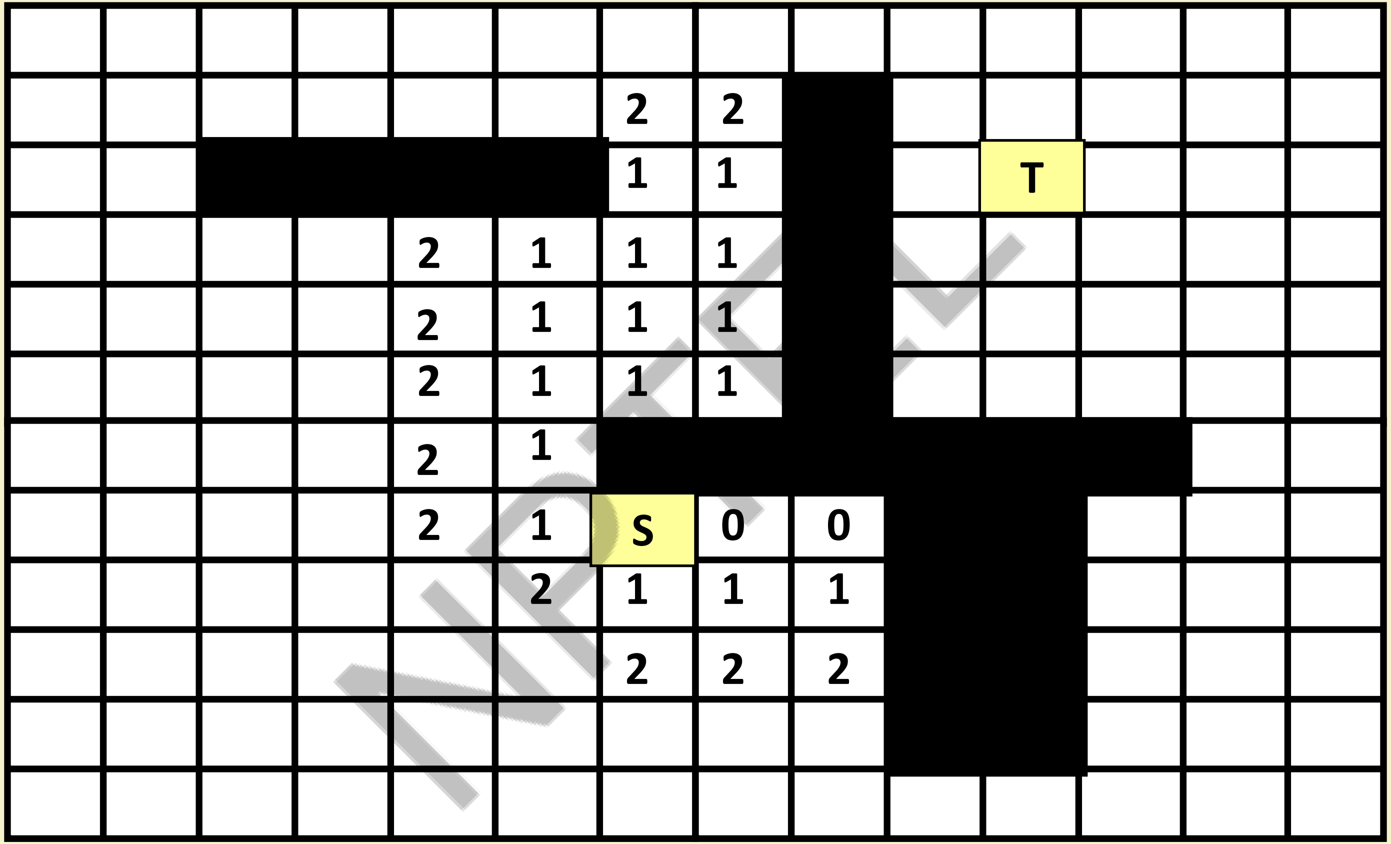
where $MD\ (S,T)$ is the Manhattan distance between S and T.

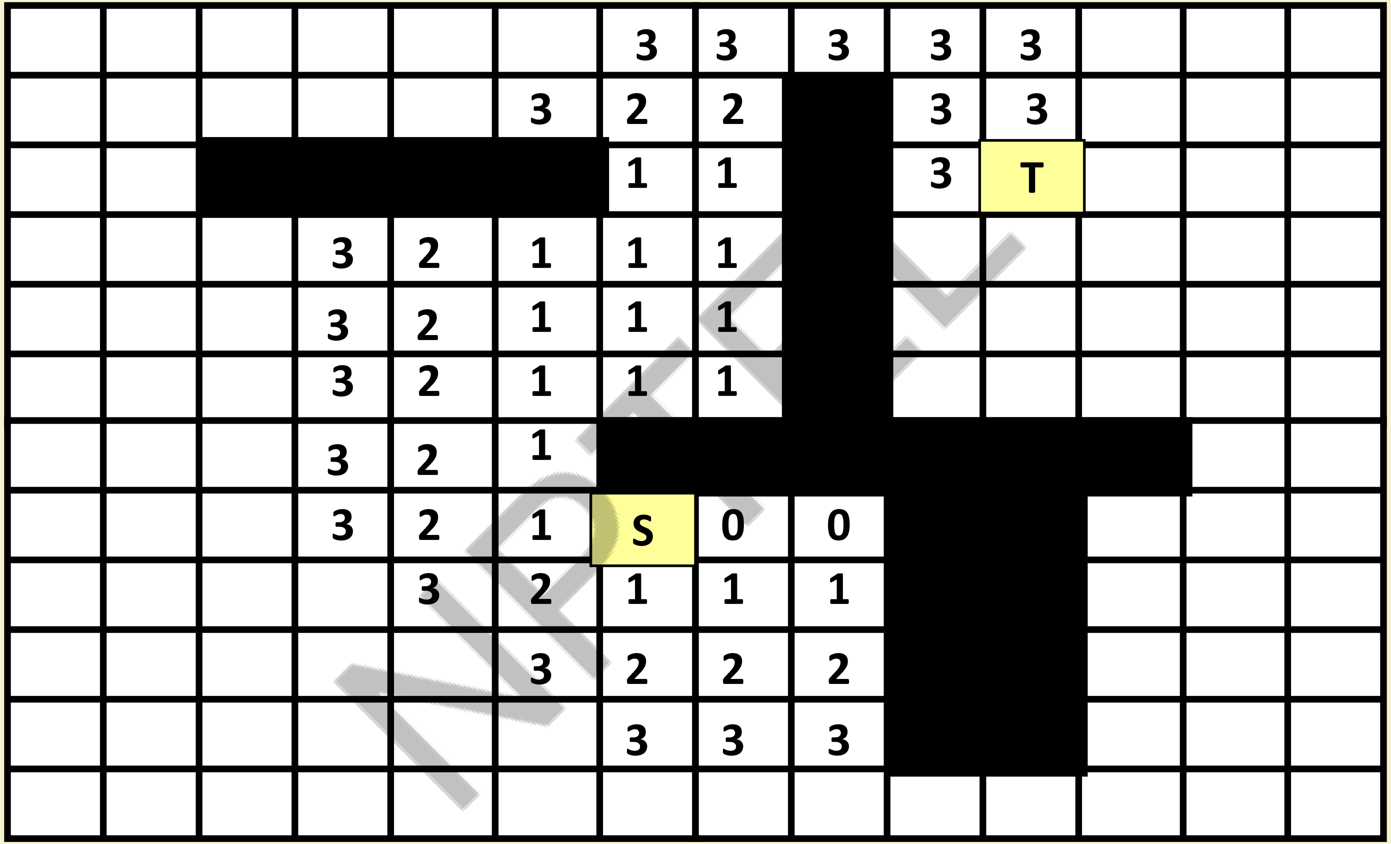
- The cell filling phase of Lee's algorithm can be modified as follows:
  - Fill a cell with the detour number with respect to a specified target T (not by its distance from source).
  - Cells with smaller detour numbers are expanded with higher priority.
- Path retracing is of course more complex, and requires some degree of searching.

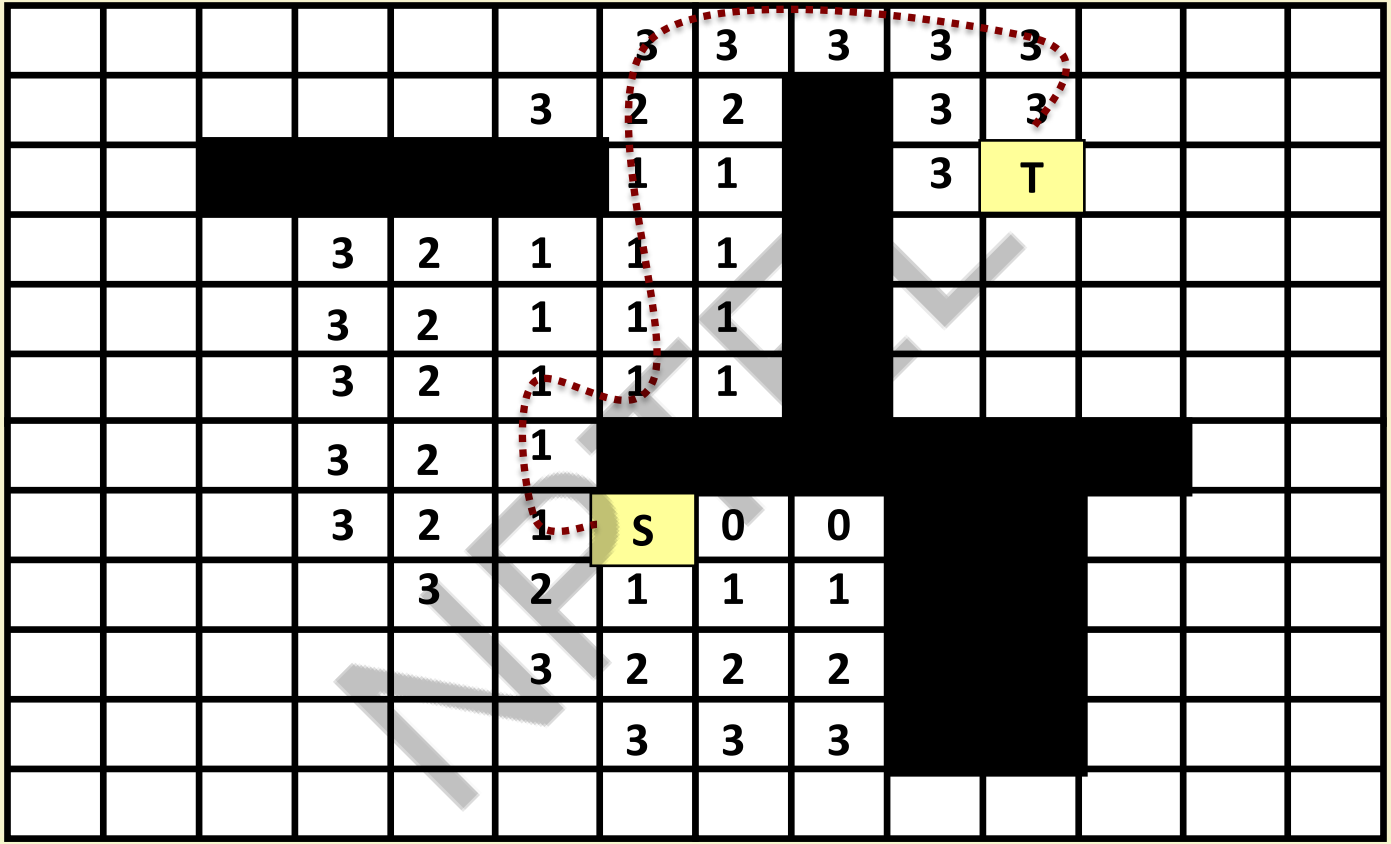

T
S


T
S
0
0


1 1 T
1 1 1
1 1 1
1 1 1
1
1 S 0 0
1 1 1


2 2
1 1 T
2 1 1 1
2 1 1 1
2 1 1 1
2 1
2 1 S 0 0
2 1 1 1
2 2 2


3 3 3 3 3
3 2 2 3 3
1 1 3 T
3 2 1 1 1
3 2 1 1 1
3 2 1 1 1
3 2 1
3 2 1 S 0 0
3 2 1 1 1
3 2 2 2
3 3 3


3 3 3 3 3
3 2 2 3 3
1 1 3 T
3 2 1 1 1
3 2 1 1 1
3 2 1 1 1
3 2 1
3 2 1 S 0 0
3 2 1 1 1
3 2 2 2
3 3 3

- Advantages:
  - Number of grid cells filled up is considerably less as compared to Lee's algorithm.
  - Running time for an NxN grid ranges from $O(N)$ to $O(N^2)$.
    - Depends on the obstructions.
    - Also locations of S and T.

Line Search Algorithm

- In maze running algorithms, the time and space complexities are too high.
- An alternative approach is called line searching, which overcomes this drawback.
- Basic idea:
  - Assume no obstacles for the time being.
  - A vertical line drawn through S and a horizontal line passing though T will intersect.
    - Manhattan path between S and T.
  - In the presence of obstacles, several such lines need to be drawn.

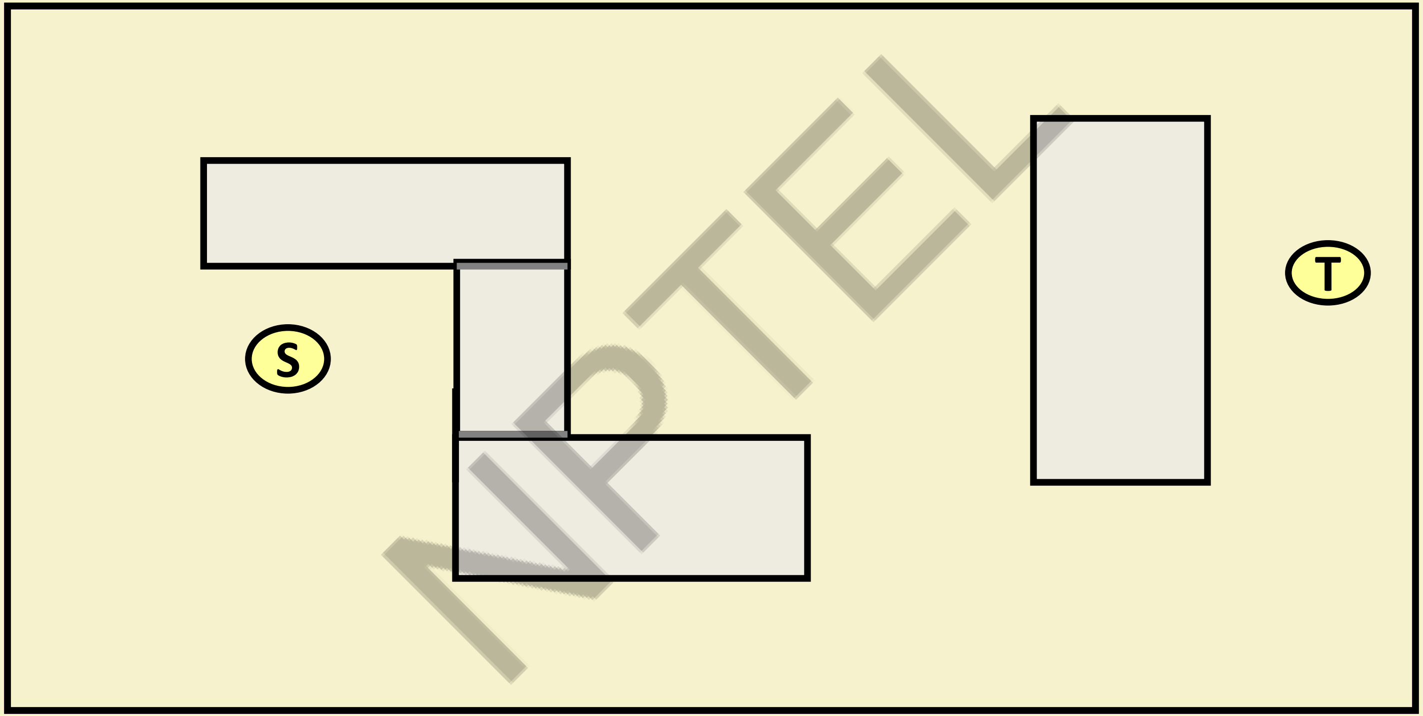
- Line search algorithms do not guarantee finding the optimal path.
  - May need several backtrackings.
  - Running time and memory requirements are significantly less.
  - Routing area and paths are represented by a set of line segments.
    - Not as a matrix as in Lee's or Hadlock's algorithm.

Mikami-Tabuchi's Algorithm

- Let S and T denote a pair of terminals to be connected.
- Step 0:
  - Generate four lines (two horizontal and two vertical) passing through S and T.
  - Extend these lines till they hit obstructions or the boundary of the layout.
  - If a line generated from S intersects a line generated from T, then a connecting path is found.
  - If they do not intersect, they are identified as trial lines of level zero.
    - Stored in temporary storage for further processing.

- <u>Step i of Iteration</u>: (i > 0)
  - Pick up trial lines of level i-1, one at a time.
    - Along the trial line, all its grid points are traced.
    - Starting from these grid points, new trial lines (of level i) are generated perpendicular to the trial line of level i-1.
  - If a trial line of level i intersects a trial line (of any level) from the other terminal point, the connecting path can be found.
    - By backtracing from the intersection point to S and T.
    - Otherwise, all trial lines of level i are added to temporary storage, and the procedure repeated.
- The algorithm guarantees to find a path if it exists.

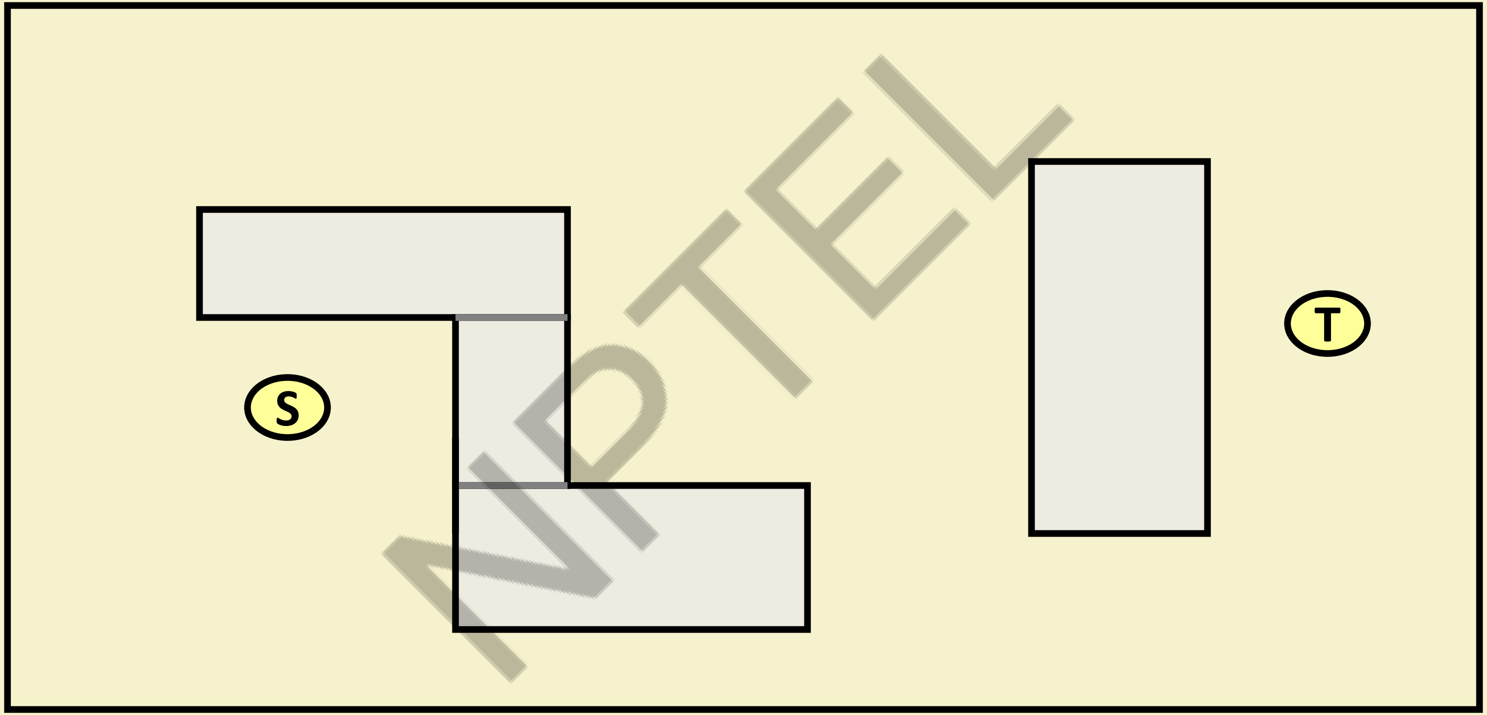
Illustration



Hightower's Algorithm

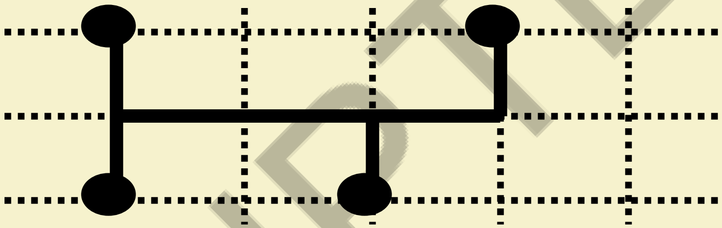
- Similar to Mikami-Tabuchi's algorithm.
  - Instead of generating all line segments perpendicular to a trial line, consider only those lines that can be extended beyond the obstacle which blocked the preceding trial line.
- Steps of the algorithm:
  - Pass a horizontal and a vertical line through source and target points (called first-level probes).
  - If the source and the target lines meet, a path is found.
  - Otherwise, pass a perpendicular line to the previous probe whenever it intersects an obstacle.
    - Concept of *escape point* and *escape line*.

Illustration



Steiner Trees

- A tree interconnecting a set $P=\{P_1,\ldots,P_n\}$ of specified points in the rectilinear plane and some arbitrary points is called a (rectilinear) Steiner tree of P.



- A Steiner tree with minimum total cost is called a Steiner minimal tree (SMT).
  - The general SMT problem is NP-hard.

Steiner Tree Based Algorithms

- Minimum length Steiner trees:
  - Goal is to minimize the sum of the length of the edges of the tree.
  - Both exact and approximate versions exist.
- Weigted Steiner trees:
  - Given a plane partitioned into a collection of weighted regions, an edge with length L in a region with weight W has cost LW.
- Steiner trees with arbitrary orientations:
  - Allows lines in non-rectilinear directions like $+45^o$ and -45^o.

END OF LECTURE 17

IIT KHARAGPUR

NPTEL ONLINE CERTIFICATION COURSES

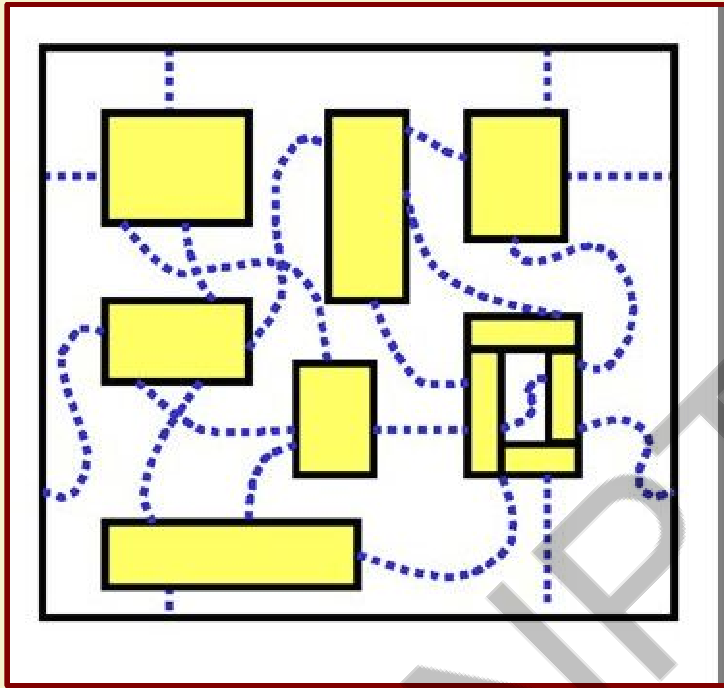
NPTEL

Lecture 18: GLOBAL ROUTING (PART 1)

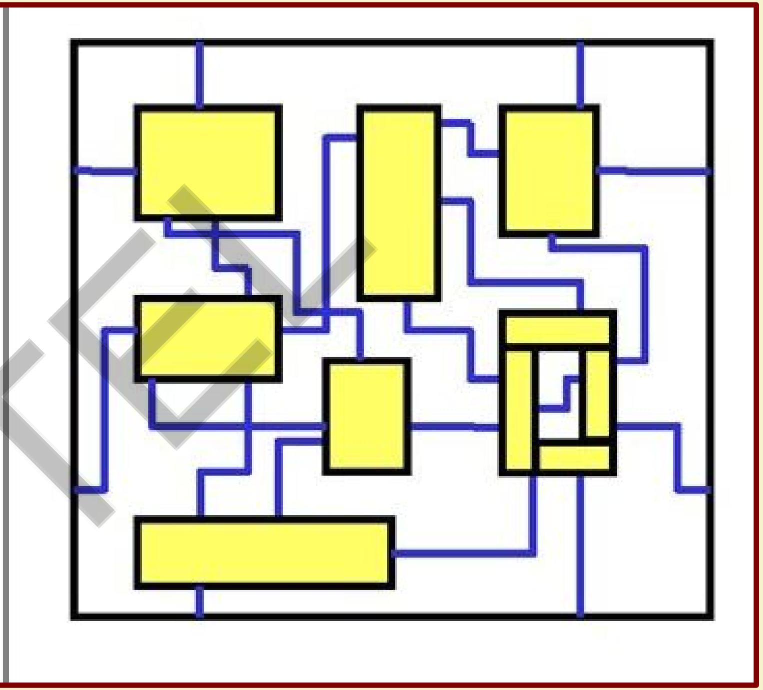
PROF. INDRANIL SENGUPTA
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Basic Idea

- The routing problem in ASIC is typically solved using a two-step approach:
  - Global Routing
    - Define the routing regions.
    - Generate a tentative route for each net.
    - Each net is assigned to a set of routing regions.
    - Does not specify the actual layout of wires.
  - Detailed Routing
    - For each routing region, each net passing through that region is assigned particular routing tracks.
    - Actual layout of wires gets fixed (channel routing and switchbox routing).



Global Routing



Detailed Routing

Routing Regions

- Regions through which interconnecting wires are laid out.
- How to define these regions?
  - Partition the routing area into a set of non-intersecting rectangular regions.
  - Types of routing regions:
    - Horizontal channel: parallel to the x-axis with pins at their top and bottom boundaries.
    - Vertical channel: parallel to the y-axis with pins at their left and right boundaries.
    - Switchbox: rectangular regions with pins on all four sides.

- Points to note:
  - Identification of routing regions is a crucial first step to global routing.
  - Routing regions often do not have pre-fixed capacities.
  - The order in which the routing regions are considered during detailed routing plays a vital part in determining overall routing quality.

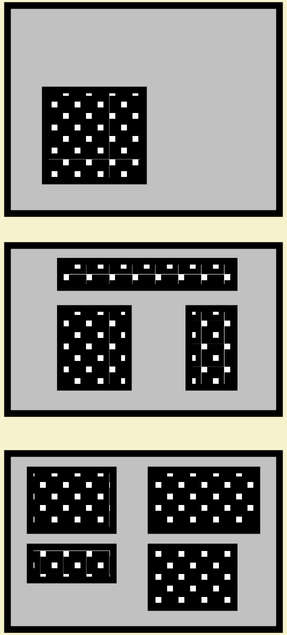
Types of Channel Junctions

- Three types of channel junctions may occur:

  L-type:
  - Occurs at the corners of the layout surface.
  - Ordering is not important during detailed routing.
  - Can be routed using channel routers.

  T-type:
  - The leg of the "T" must be routed before the shoulder.
  - Can be routed using channel routers.

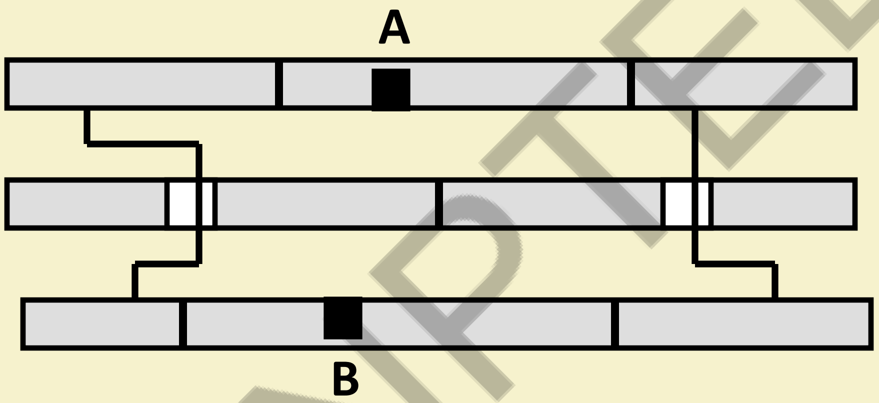
  +-type:
  - More complex and requires switchbox routers.
  - Advantageous to convert +-junctions to T-junctions.



Design Style Specific Issues

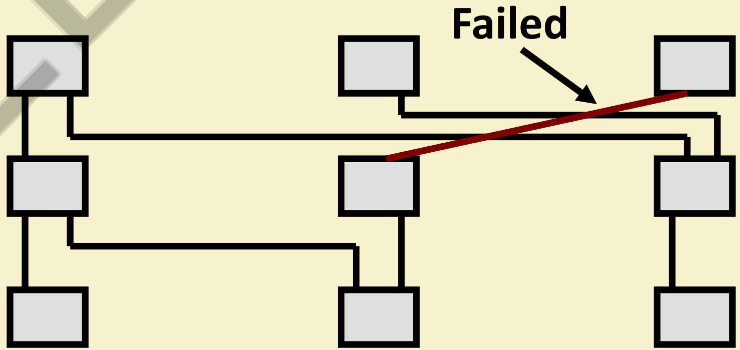
- Full Custom
  - The problem formulation is similar to the general formulation as discussed.
    - All the types of routing regions and channels junctions can occur.
  - Since channels can be expanded, some violation of capacity constraints are allowed.
  - Major violation in constraints are, however, not allowed.
    - May need significant changes in placement.

- Standard Cell
  - At the end of the placement phase
    - Location of each cell in a row is fixed.
    - Capacity and location of each feed-through is fixed.
    - Feed-throughs have predetermined capacity.
  - Only horizontal channels exist.
    - Channel heights are not fixed.
  - Insufficient feed-throughs may lead to failure.
  - Over-the-cell routing can reduce channel height, and change the global routing problem.



A cannot be connected to B

- Gate Array
  - The size and location of cells are fixed.
  - Routing channels & their capacities are also fixed.
  - Primary objective of global routing is to guarantee routability.
  - Secondary objective may be to minimize critical path delay.



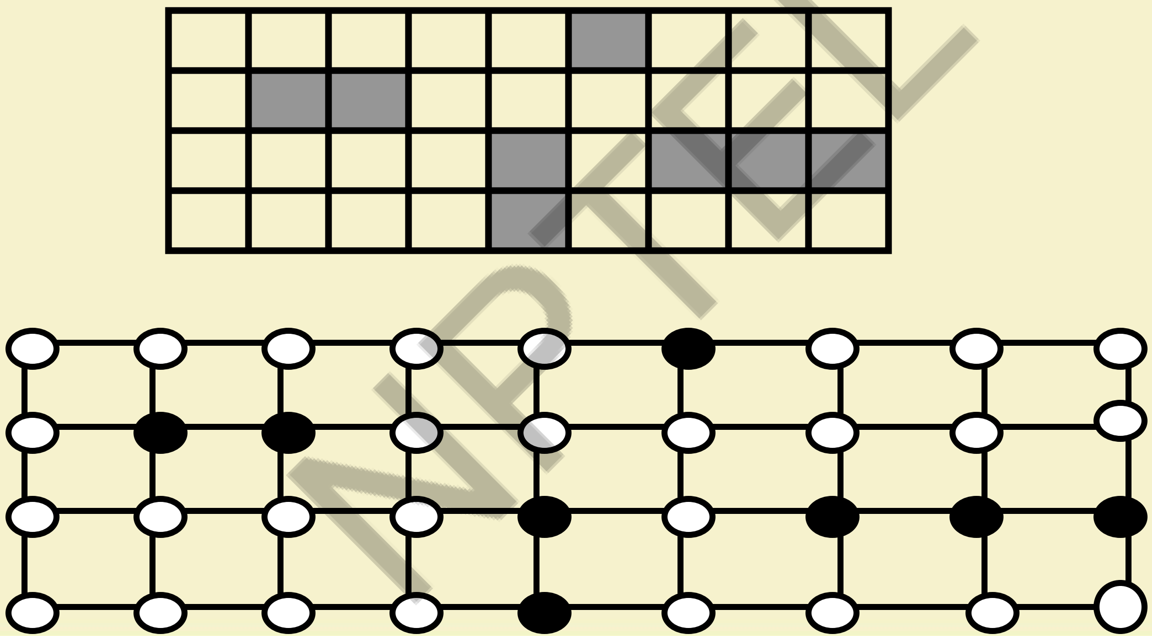
Graph Models used in Global Routing

- Global routing is typically studied as a graph problem.
  - Routing regions and their relationships modeled as graphs.
- Three important graph models:
  1. Grid Graph Model
     - Most suitable for area routing
  2. Checker Board Model
  3. Channel Intersection Graph Model
     - Most suitable for global routing

Grid Graph Model

- A layout is considered to be a collection of unit side square cells (grid).
- Define a graph:
  - Each cell c_i is represented as a vertex v_i.
  - Two vertices v_i and v_j are joined by an edge if the corresponding cells c_i and c_j are adjacent.
  - A terminal in cell c_i is assigned to the corresponding vertex v_i.
  - The occupied cells are represented as filled circles, whereas the others as clear circles.
  - The capacity and length of each edge is set to 1.
- Given a 2-terminal net, the routing problem is to find a path between the corresponding vertices in the grid graph.

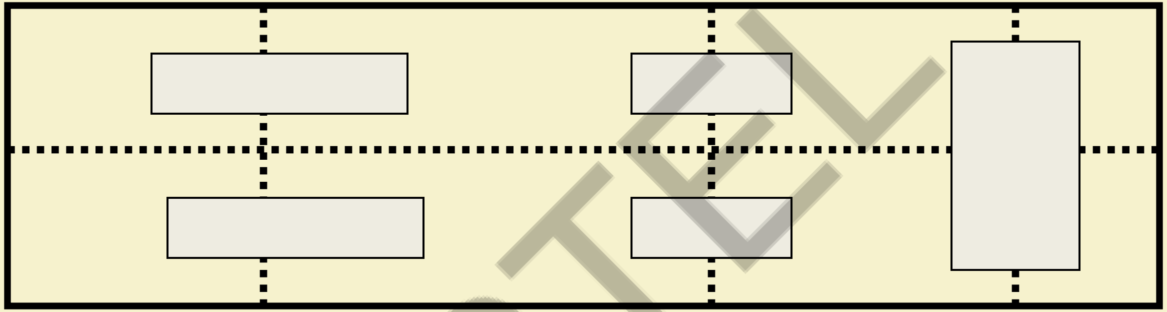
Grid Graph Model :: Illustration

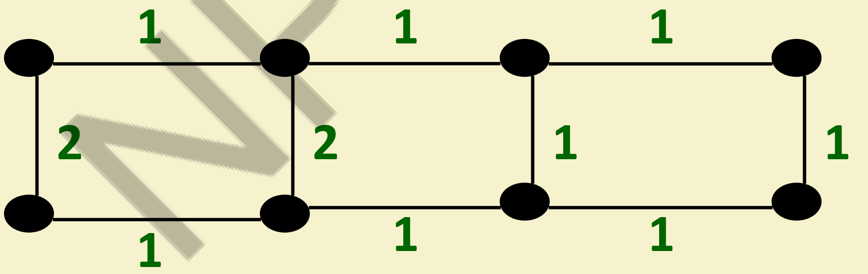


Checker Board Model

- More general than the grid graph model.
- Approximates the layout as a coarse grid.
- Checker board graph is generated in a manner similar to the grid graph.
- The edge capacities are computed based on the actual area available for routing on the cell boundary.
  - The partially blocked edges have a capacity of 1.
  - The unblocked edges have a capacity of 2.
- Given the cell numbers of all terminals of a net, the global routing problem is to find a path in the coarse grid graph.

Checker Board Model :: Illustration

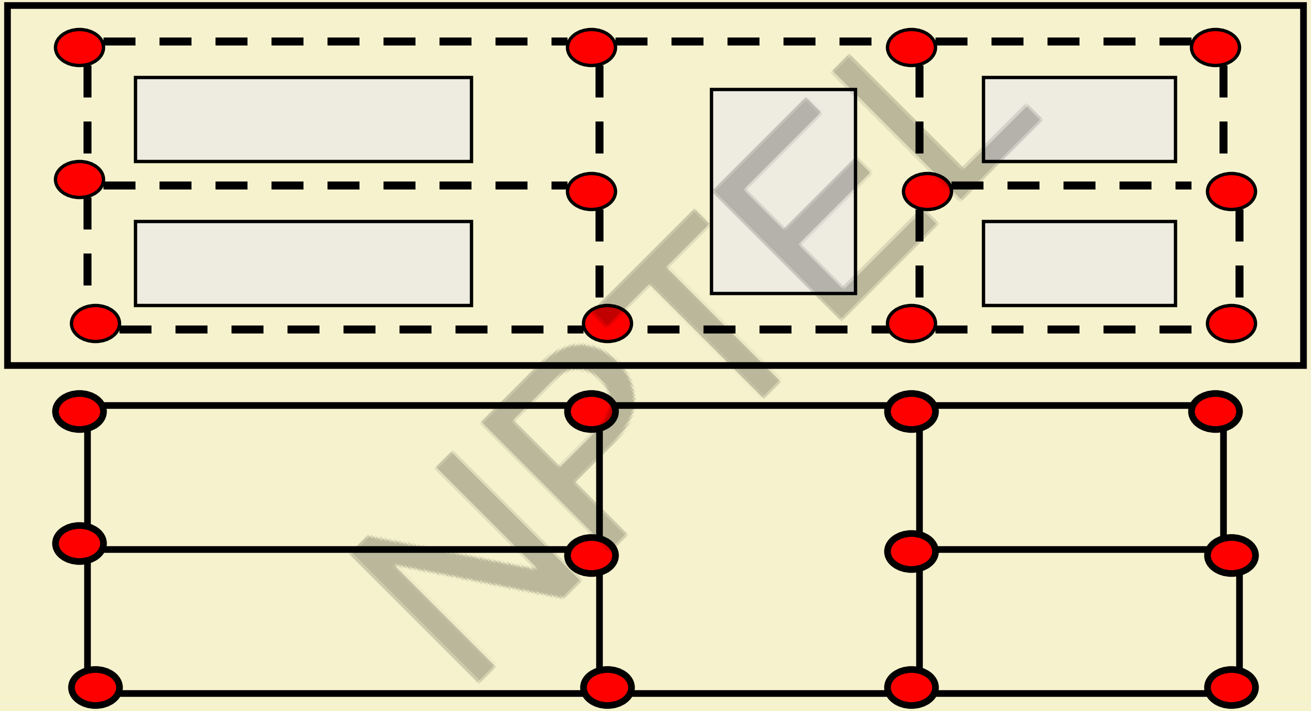




Channel Intersection Graph

- Most general and accurate model for global routing.
- Define a graph:
  - Each vertex v_i represents a channel intersection CI_i.
  - Channels are represented as edges.
  - Two vertices v_i and v_j are connected by an edge if there exists a channel between CI_i and CI_j.
  - Edge weight represents channel capacity.

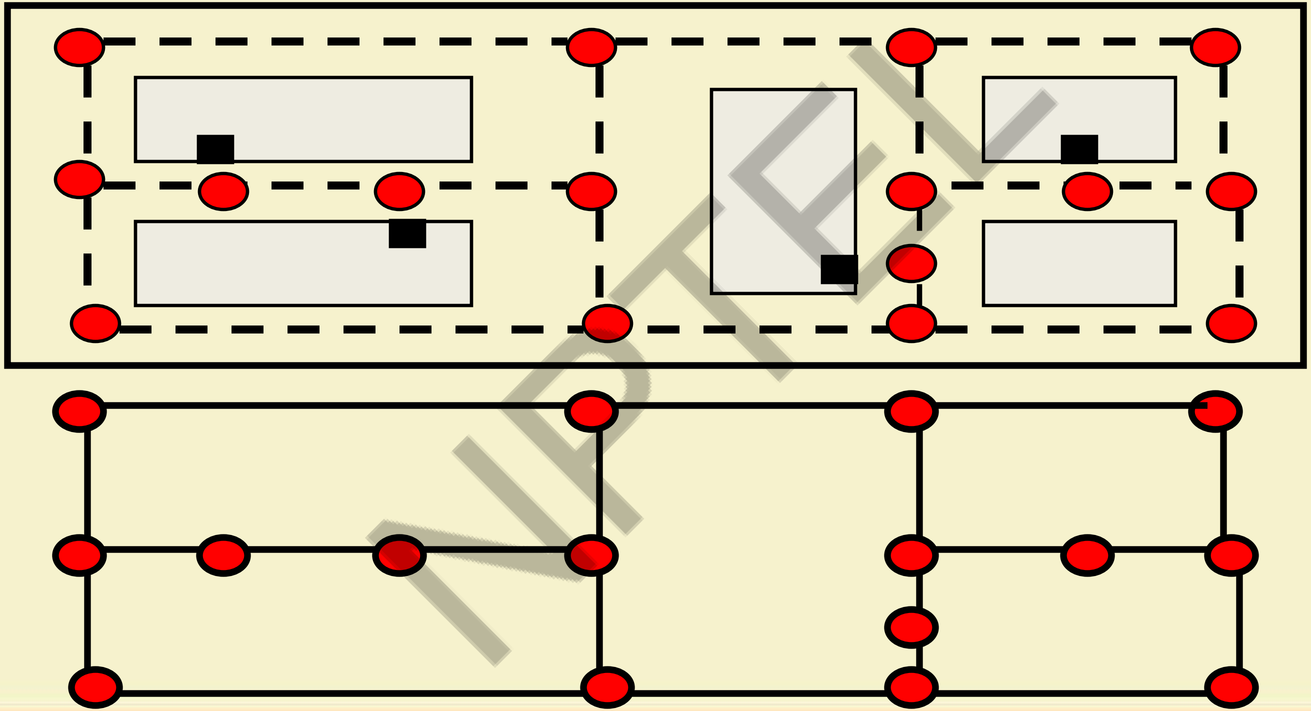
Illustration



Extended Channel Intersection Graph

- Extension of the channel intersection graph.
  - Includes the pins as vertices so that the connections between the pins can be considered.
- The global routing problem is simply to find a path in the channel intersection graph.
  - The capacities of the edges must not be violated.
  - For 2-terminal nets, we can consider the nets sequentially.
  - For multi-terminal nets, we can have an approximation to minimum Steiner tree.

Illustration



END OF LECTURE 18

IIT KHARAGPUR

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Lecture 19: GLOBAL ROUTING (PART 2)

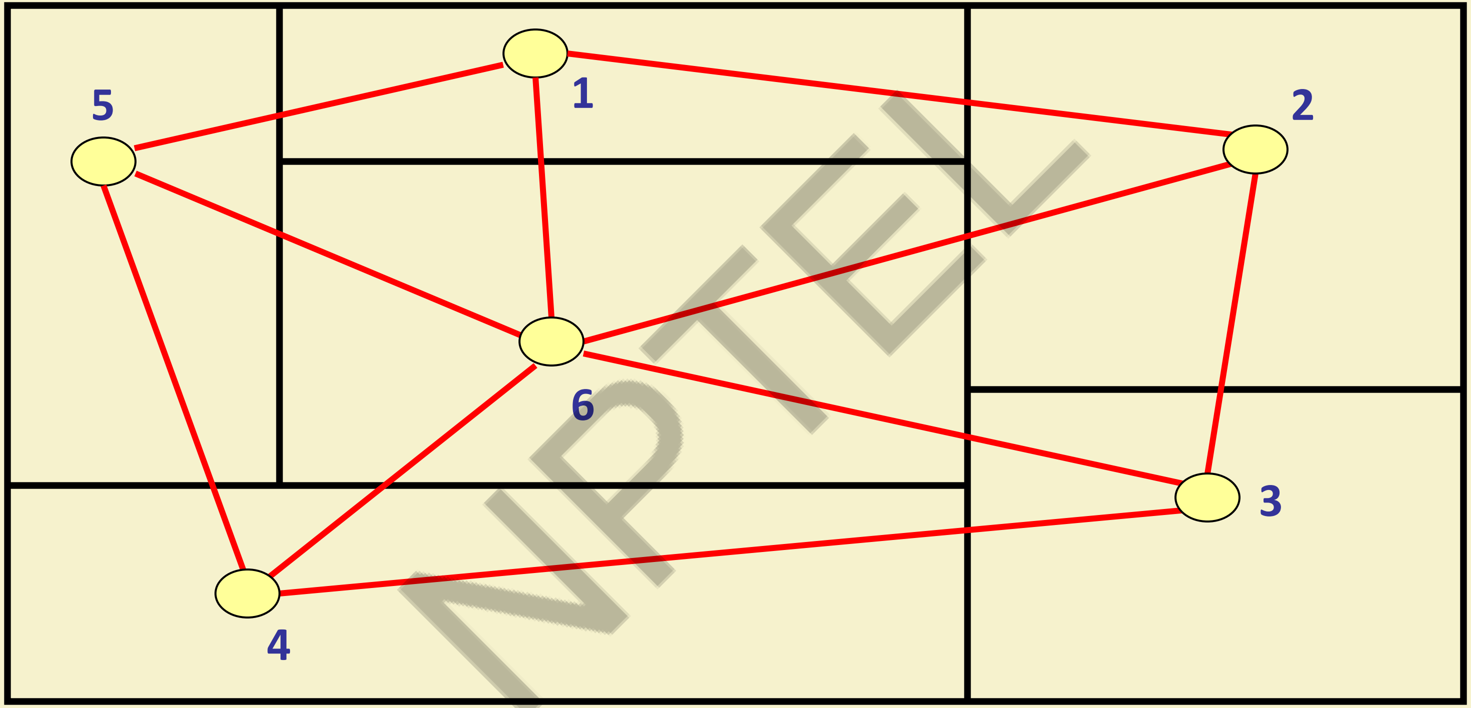
PROF. INDRANIL SENGUPTA
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Approaches to Global Routing

- What does a global router do?
  - It decomposes a large routing problem into small and manageable sub-problems
    - Called detailed routing
  - This is done by finding a rough path for each net.
    - Sequences of sub-regions it passes through

When Floorplan is Given

- The dual graph of the floorplan (shown in red) is used for global routing.
- Each edge is assigned with:
  - A weight w_{ij} representing the capacity of the boundary.
  - A value L_{ij} representing the edge length.
- Global routing of a two-terminal net
  - Terminals in rectangles r_1 and r_2.
  - Path connecting vertices v_1 and v_2 in G.


1
2
3
4
5
6

When Placement is Given

- The routing region is partitioned into simpler regions.
  - Typically rectangular in shape.
- A routing graph can be defined.
  - Vertices represent regions, and correspond to channels.
  - Edges represent adjacency between channels.
- Global routing of a two-terminal net
  - Terminals in regions r_1 and r_2.
  - Path connecting vertices v_1 and v_2 in G.

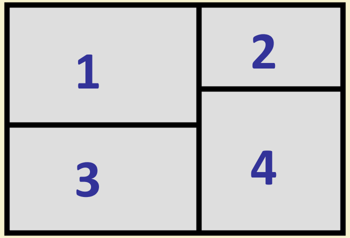
Sequential Approaches

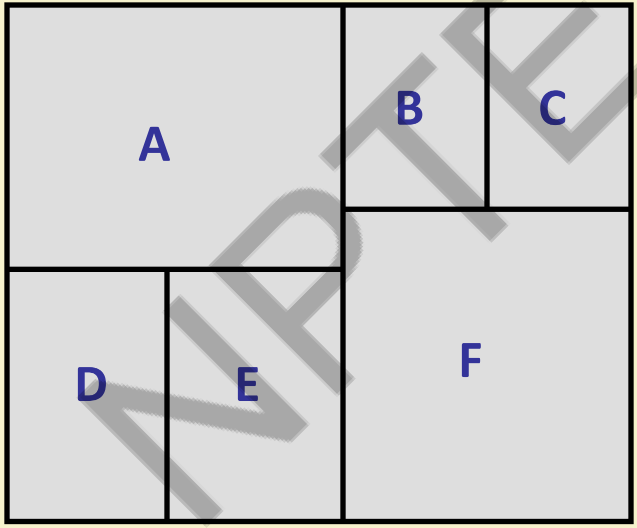
- Nets are routed sequentially, one at a time.
  - First an ordering of the nets is obtained based on: (a) Number of terminals, (b) Bounding box length, (c) Criticality.
  - Each net is then routed as dictated by the ordering.
- Most of these techniques use variations of maze running or line search methods.
- Very efficient at finding routes for nets as they employ well-known shortest path algorithms.
  - Rip up and reroute heuristic in case of conflict.

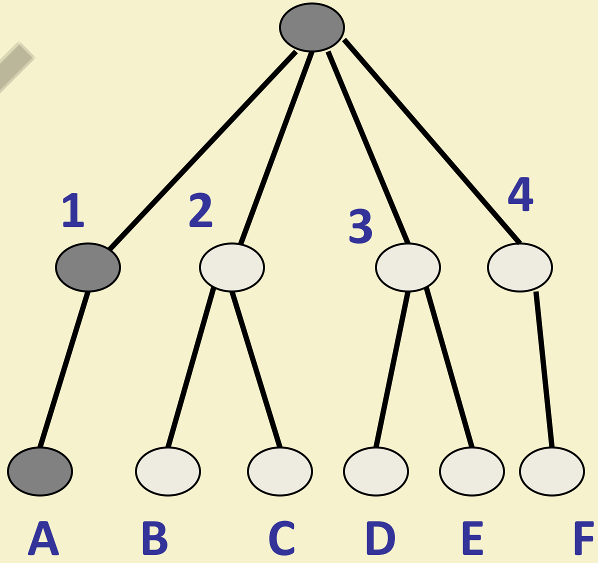
Hierarchical Approaches

- Use the hierarchy of the routing graph to decompose a large routing problem into sub-problems of manageable size.
  - The sub-problems are solved independently.
  - Sub-solutions are combined to get the total solution.
- A cut tree is defined on the routing graph.
  - Each interior node represents a primitive global routing problem.
  - Each problem is solved optimally by translating it into an integer programming problem.
  - The solutions are finally combined.

Hierarchical Approach :: Illustration







Hierarchical Routing :: Top-Down Approach

- Let the root of the cut tree T be at level 1, and the leaves of T at level h.
  - h is the height of T.
- The top-down approach traverses T from top to down, level by level.
  - I_i denotes the routing problem instance at level i.
- The solutions to all the problem instances are obtained using an integer programming formulation.

Algorithm

```
procedure  Hier_Top_Down
begin
    Compute solution R_i of the routing problem I_1;
    for i=2 to h do
    begin
        for all nodes n at level i-1 do
              Compute solution R_n of the routing problem I_n;
        Combine all solutions R_n for all nodes n, and R_{i-1} into solution R_i;
    end
end
```

Hierarchical Routing :: Bottom-up Approach

- In the first phase, the routing problem associated with each branch in T is solved by IP.
- The partial routings are then combined by processing internal tree nodes in a bottom-up manner.
- Main disadvantage of this approach:
  - A global picture is obtained only in the later stages of the process.

Algorithm

```
procedure  Hier_Bottom_Down
begin
    Compute solution R_h of the level-h abstraction of the problem;
    for i=h to 1 do
       begin
          for all nodes n at level i-1 do
              Compute solution R_n of the routing problem I_n by combining the
                  solution to the children of node n;
        end;
end;
```

Integer Linear Programming Approach

- The problem of concurrently routing the nets is computationally hard.
  - The only known technique uses integer programming.
- Global routing problem can be formulated as a 0/1 integer program.
- The layout is modeled as a grid graph.
  - N vertices: each vertex represents a grid cell.
  - M edges: an edge connects vertices i and j if the grid cells i and j are adjacent.
  - The edge weight represents the capacity of the boundary.

- For each net i, we identify the different ways of routing the net.
  - Suppose that there are n_i possible Steiner trees $t^i_1, t^i_2, \ldots, t^i_{ni}$ to route the net.
  - For each tree t^i_j, we associate a variable x_{ij} as:

$$x_{ij} = 1, \text{ if net i is routed using tree } t^i_j$$
$$= 0, \text{ otherwise.}$$

  - Only one tree must be selected for each net:

$$\sum_{j=1}^{n_i} x_{ij} = 1$$

- For a grid graph with M edges and $T = \Sigma n_i$ trees, we can represent the routing trees as a 0-1 matrix $A_{MxT} = [a_{ip}]$.

$$a_{ip} = 1, \text{ if edge i belongs to tree p}$$
$$= 0, \text{ otherwise.}$$

- Capacity of each arc (boundary) must not be exceeded:

$$\sum_{k=1}^{N} \sum_{l=1}^{n_k} a_{ip} \, x_{lk} \leq c_i$$

- If each tree t^j_i is assigned a cost g_{ij}, a possible objective function to minimize is:

$$F = \sum_{i=1}^{N} \sum_{j=1}^{n_k} g_{ij} \, x_{ij}$$

- <u>0-1 integer programming formulation:</u>

Minimize $\sum_{i=1}^{N} \sum_{j=1}^{n_k} g_{ij}\, x_{ij}$

Subject to:

$$\sum_{j=1}^{n_i} x_{ij} = 1, \qquad 1 \le i \le N$$

$$\sum_{k=1}^{N} \sum_{l=1}^{n_k} a_{ip}\, x_{lk} \le c_i, \qquad 1 \le i \le M$$

$$x_{kj} = 0,1 \qquad 1 \le k \le N,\ 1 \le j \le n_k$$

Performance Driven Routing

- Advent of deep sub-micron technology
  - Interconnect delay constitutes a significant part of the total net delay.
  - Reduction in feature sizes has resulted in increased wire resistance.
  - Increased proximity between the devices and interconnections results in increased cross-talk noise.
- Routers should model the cross-talk noise between adjacent nets.
- For routing high-performance circuits, techniques adopted:
  - Buffer insertion
  - Wire sizing
  - High-performance topology generation

END OF LECTURE 19