CSE460: VLSI Design

Lecture 15+16: VLSI Physical Design
Routing Algorithm

Outline

- Introduction
- Routing Approaches
- Lee’s Maze Algorithm
- Extension of Lee’s Algorithm
- Memory Requirements

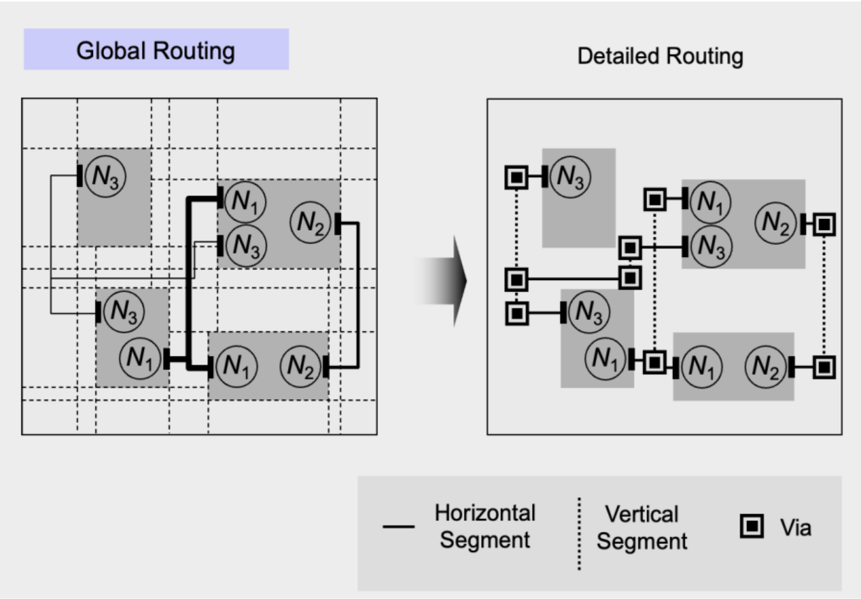
Introduction

- Forming physical connection between pins. First, approx. path (**Global Routing**), then, placing metal line segment (**Detailed Routing**)

- After **placement** phase, the exact **locations** of **circuit blocks** and **pins** are determined.

- Space not occupied by the blocks are used for routing and are called as **routing regions**.

- Two kinds of approaches to solve global routing problem: **Sequential** & **Concurrent** Approach.



Routing Approaches

Sequential Approach:

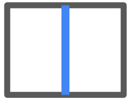
- In this approach, as the name suggests, pins are routed **one by one**.
- Order is important: once a region has been routed it may block other nets which are yet to be routed.
- Eg.
    1. **Maze routing** algorithms
    2. Line-probe algorithms
    3. Shortest path based algorithms

Concurrent Approach:

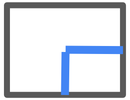
- This approach avoids the ordering problem by considering routing of all the nets simultaneously.
- Computationally sophisticated.

Lee’s Maze Routing algorithms

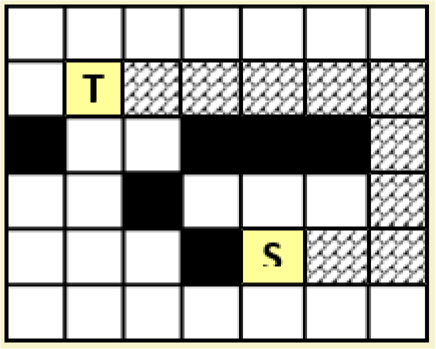
- The layout surface is assumed to be made up of a rectangular array of **grid cells**.
- Each ***Grid cell*** represents a square cell where ***one*** wire can cross.
- Objective is to find out the ***shortest*** path (**sequence of grid cells**) for <u>connecting two points</u> (Source, **S** to Target, **T**) .
- When using cells, a wire can either ***cross*** or ***bend***.



cross



Bend



- Some of the grid cells act as **obstacles (Black Cells)**.

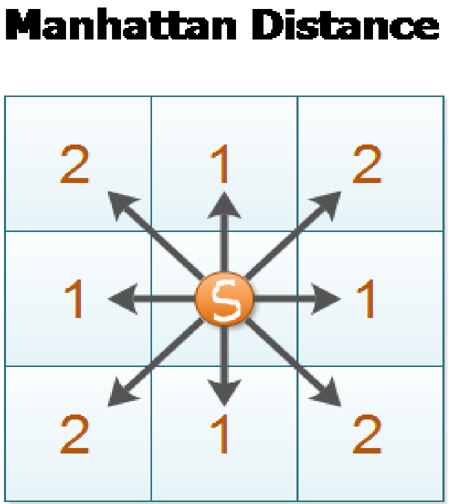
  -Blocks that are placed on the surface.

  -Some nets that are already laid out.

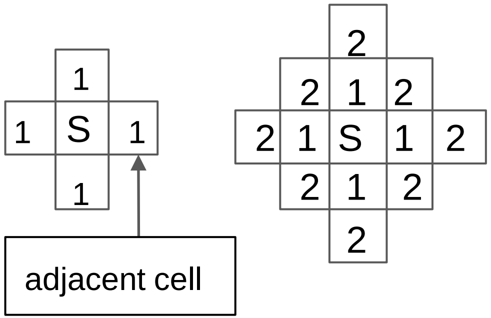
Lee’s Algorithm : *Expansion*

Step 1: Wave propagation -

- Iterative process
- Starts at the most adjacent cell of the Source.
- Need to find all new cells/grid that are reachable at ***pathlength 1***(i.e, all paths that are just 1 unit in total length(just 1 cell)).
- Using the ***pathlength 1*** cells, all new cells which are reachable at ***pathlength 2*** can be found.
- Process is repeated until the target, **T** is reached.
- During i^{th} iteration, non-blocking grid cells at Manhattan distance of i from grid cell **S** are all labeled with **i**



Note: Manhattan distance is the number of strides (horizontal and/or vertical) required to reach a cell.



Lee’s Algorithm

Step 1: Wave propagation (contd)-

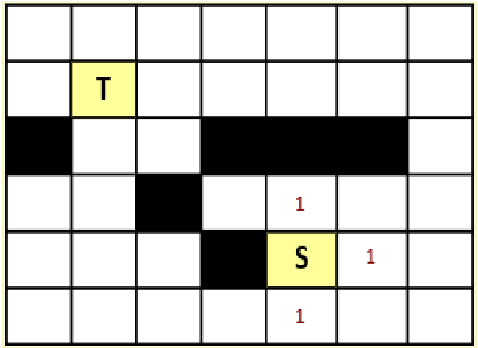
- Labeling continues until the target grid cell **T** is marked in iteration L (i.e. when i=L0).
- L0 is the length of the shortest path.

- The process fails if:

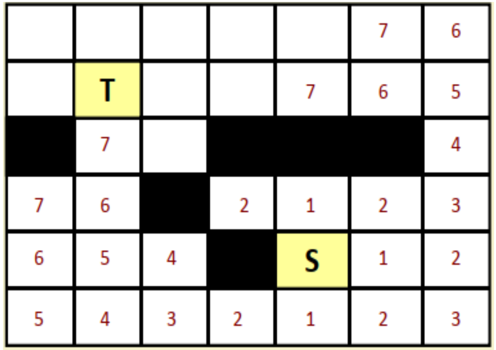
   - T is not reached and no new grid cells can be labeled during step i.

   - T is not reached and i equals M, some upper bound on the path length.

i = 1



i = 7

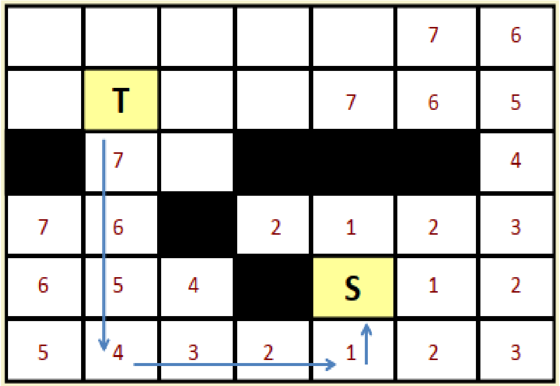


Lee’s Algorithm

Step 2: Retrace-

- 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.

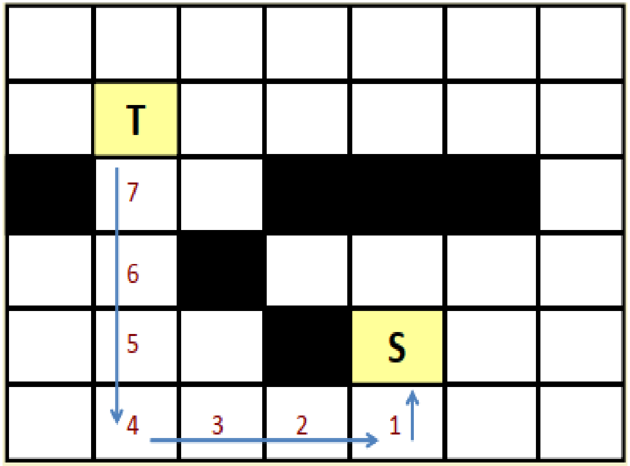
- In practice, the **rule of thumb** is not to change the direction of retrace unless one has to do so.
- **Minimizes** number of **bends.**



Lee’s Algorithm

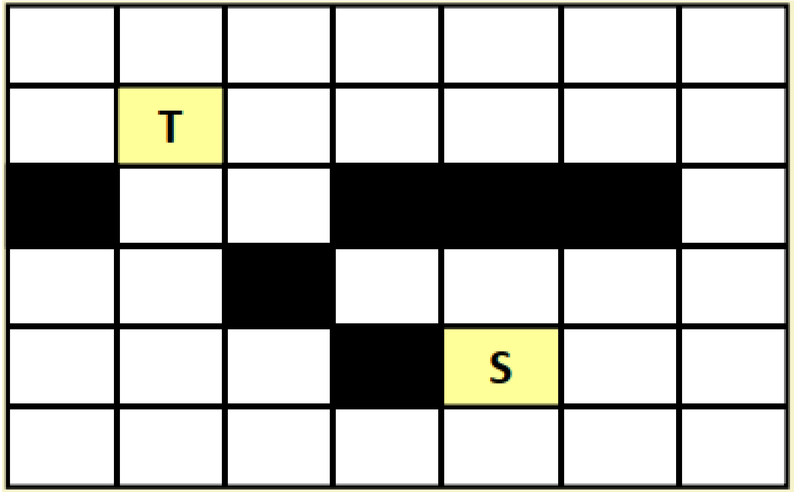
Step 3: Label clearance-

- 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.



Example 1

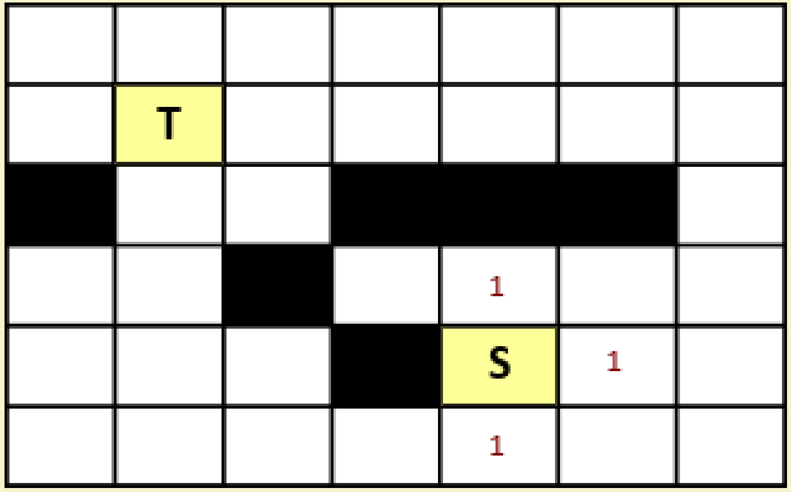
Initial routing problem:



Example 1

Step 1: Wave propagation

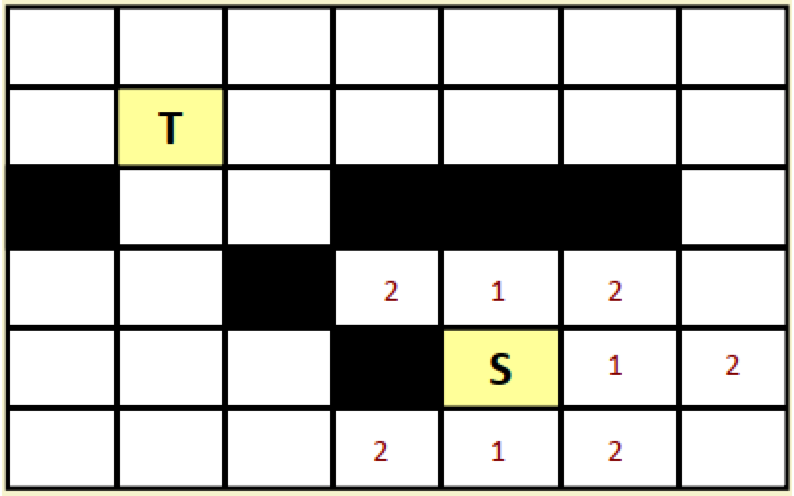
i = 1



Example 1

Step 1: Wave propagation

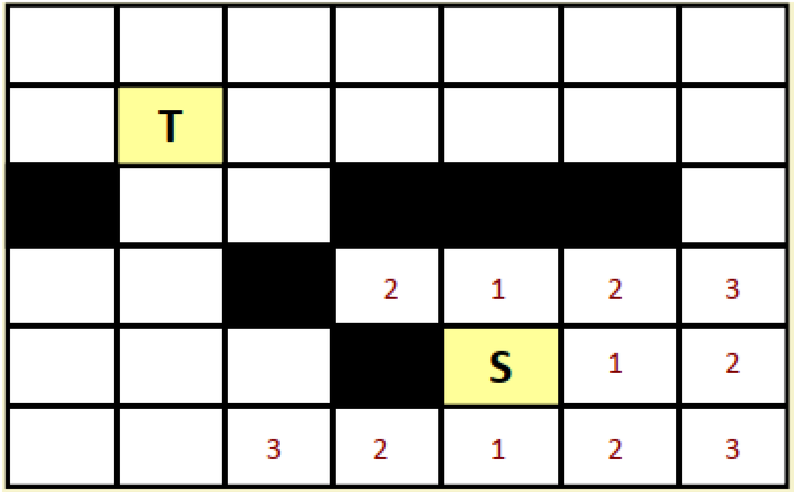
i = 2



Example 1

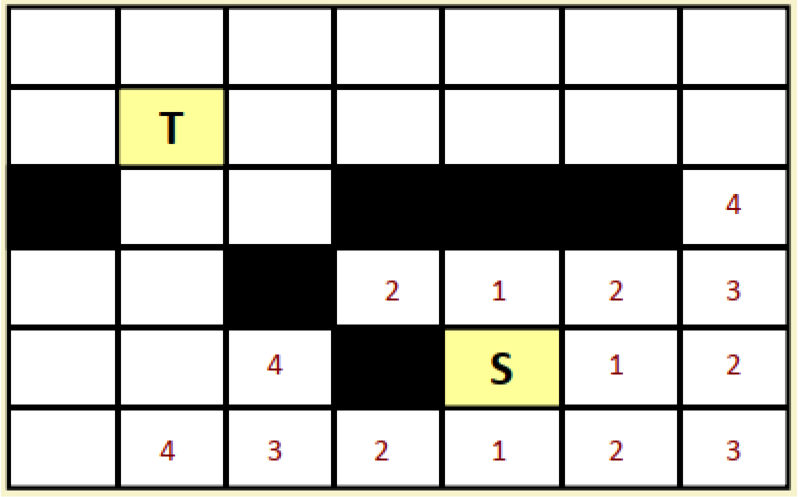
Step 1: Wave propagation

i = 3

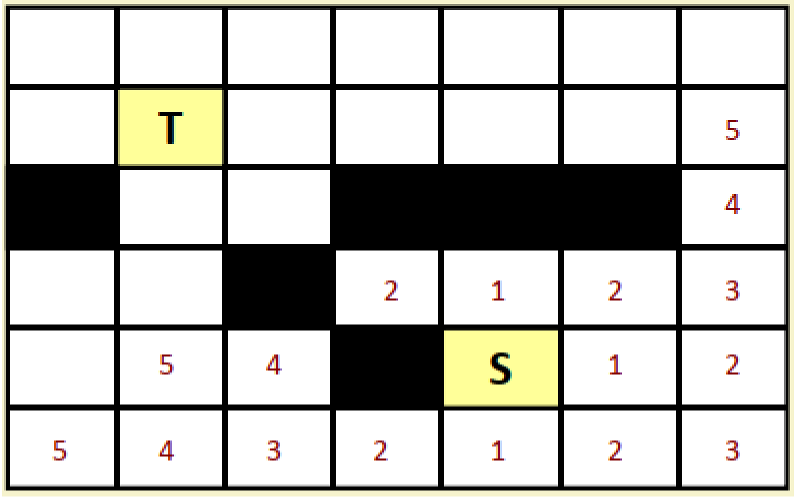


Example 1

Step 1: Wave propagation



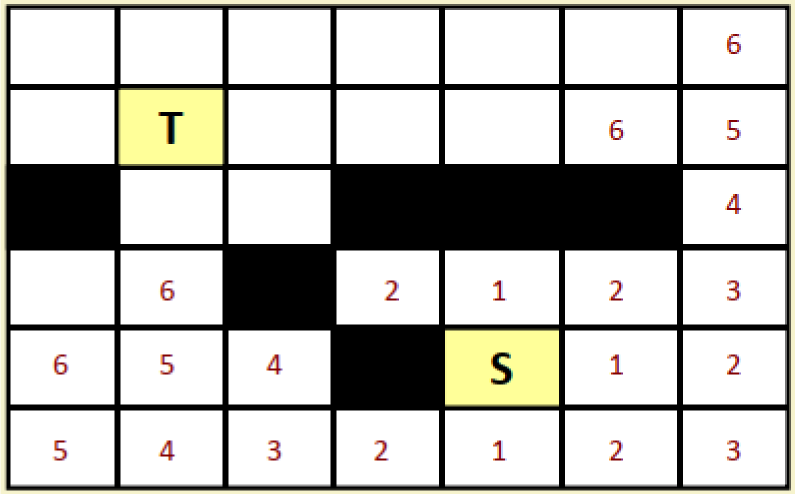
i = 4

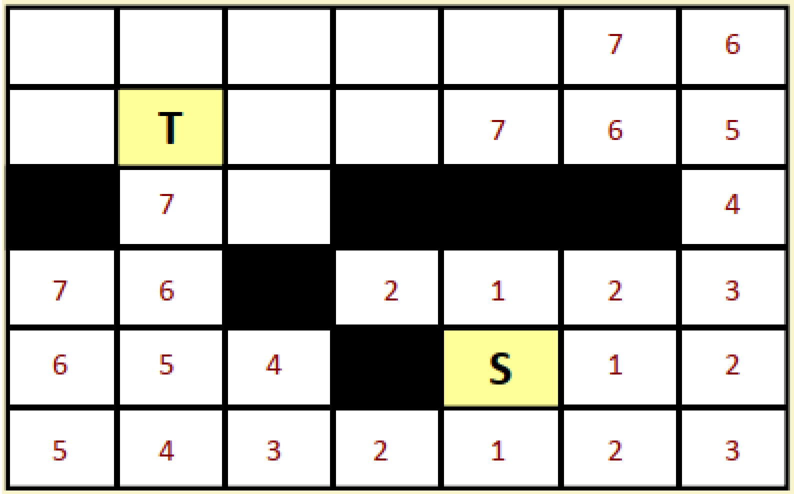


i = 5

Example 1

Step 1: Wave propagation



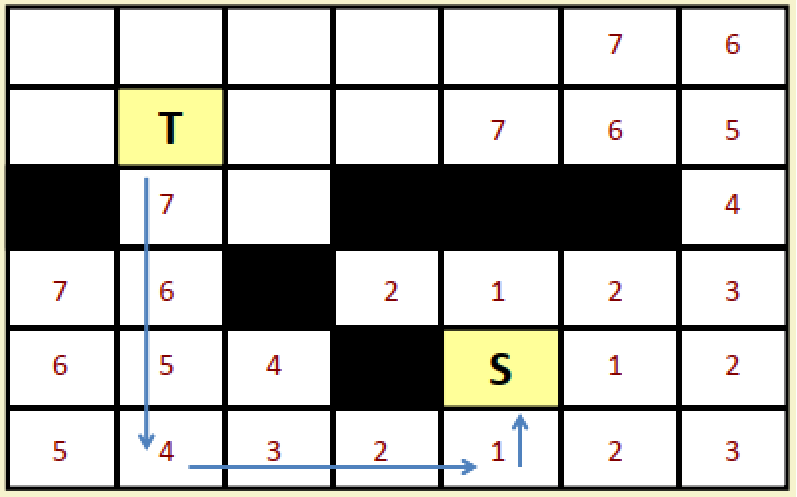


i = 6

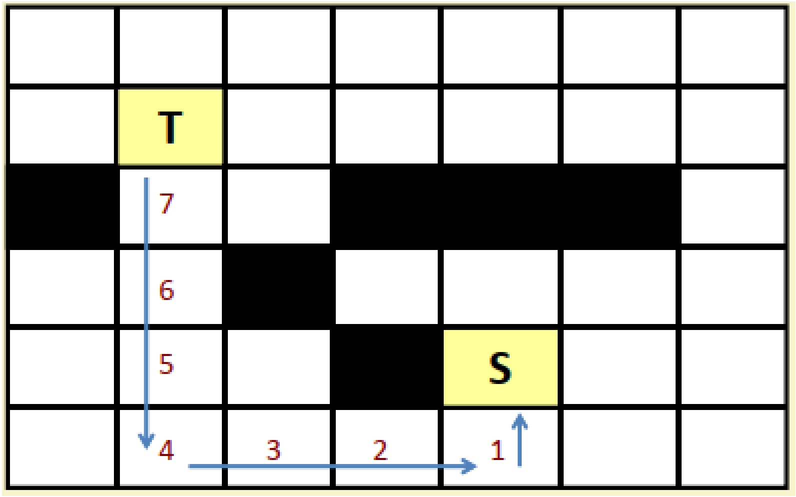
i = 7

Example 1

Step 2: Backtrace

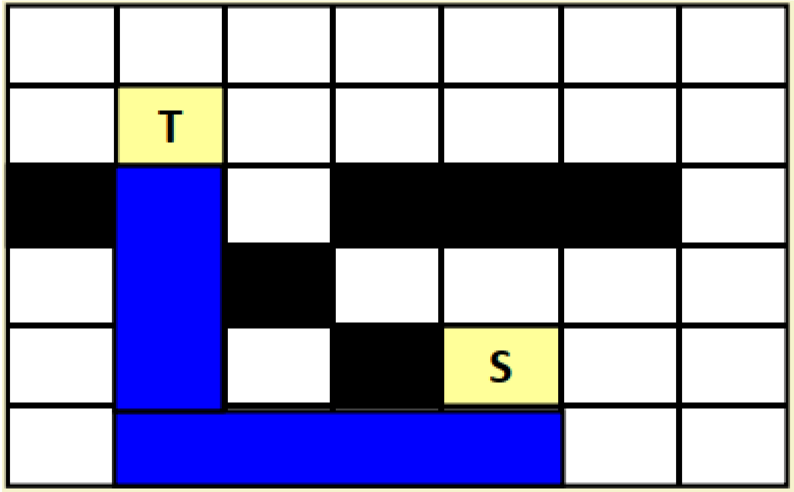


Step 3: Clear



Example 1

Final output (Mark):



- Marked region is now blocked.
- New wave cannot propagate through this merged region unless the same S has another target T.
- If the source has another target T, then whole marked (blue) region will be considered as Source, S which we will explore in Example 2.

Multi-Terminal Nets

- Step 1: Propagate wave from the source s to the closet target.
  - One of the terminals (**A**; out of 5) 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.
- Step 2: <u>All the cells in the marked path are next labeled as source cells</u>,(total path from A-B is considered as 1 source) and the remaining unconnected terminals as targets.
- 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

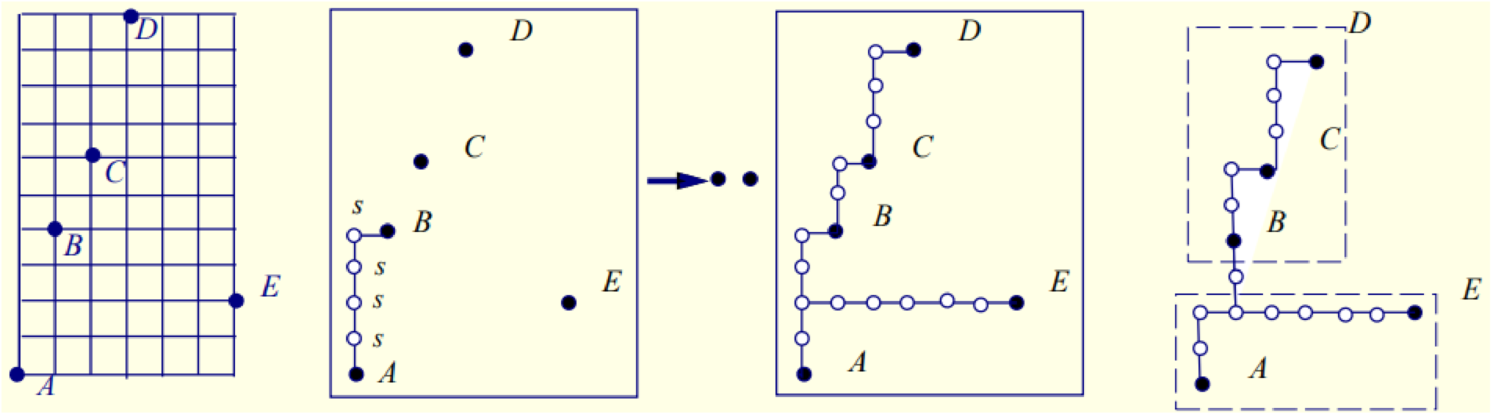
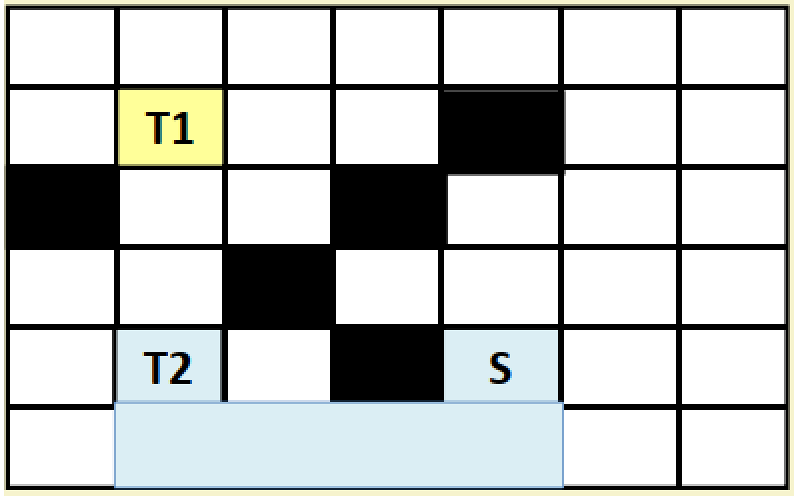


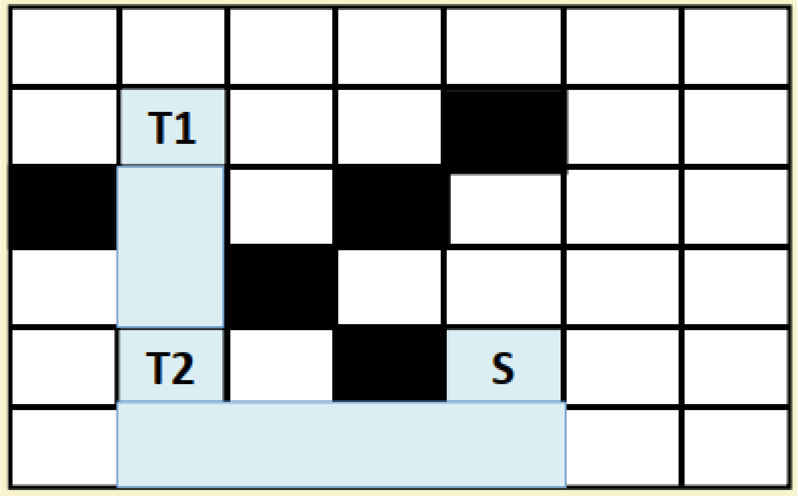
Fig: (a) A as source; B-E as targets (b) A-B as source; C-E as Targets. (c) all paths from source to target were found (d) optimized path from A-B (*low bend*)

Example 2

Step 1 to 3:

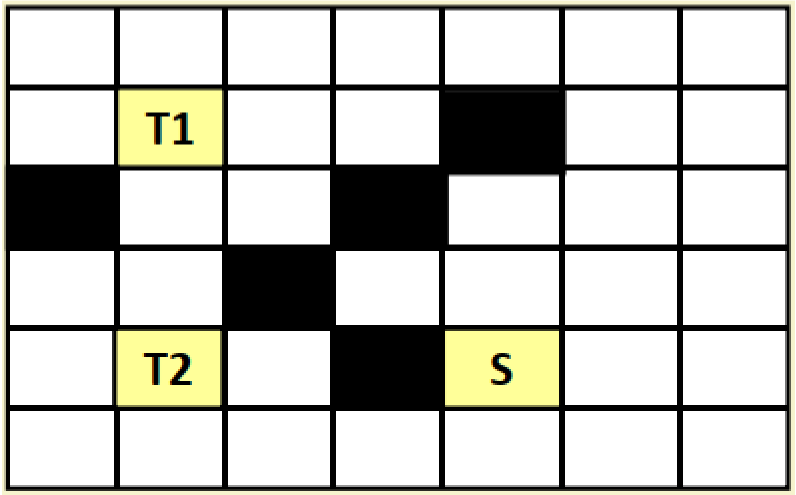


Step 1 to 3 (Again):

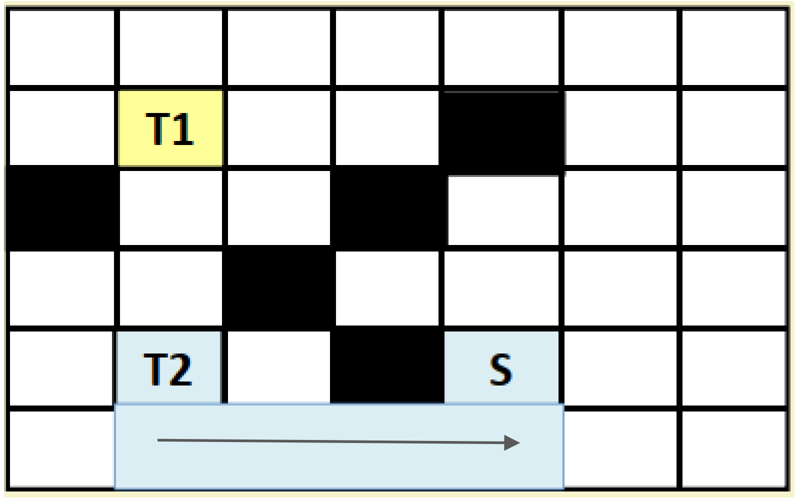


Ref. [1]

Example 2: Extension of Lee's algorithm[Multi point nets/ Target]

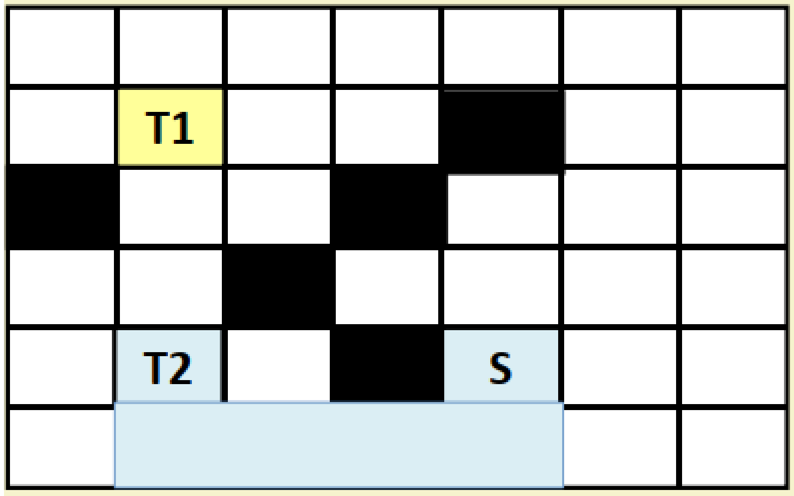


(a) Initial state

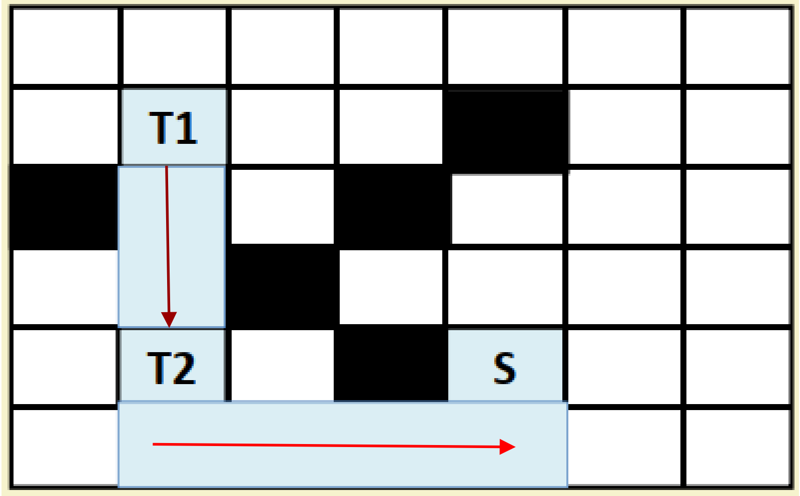


(b) Finding Path-1; From S to T2

Example 2 : [Multi point nets/ Target]

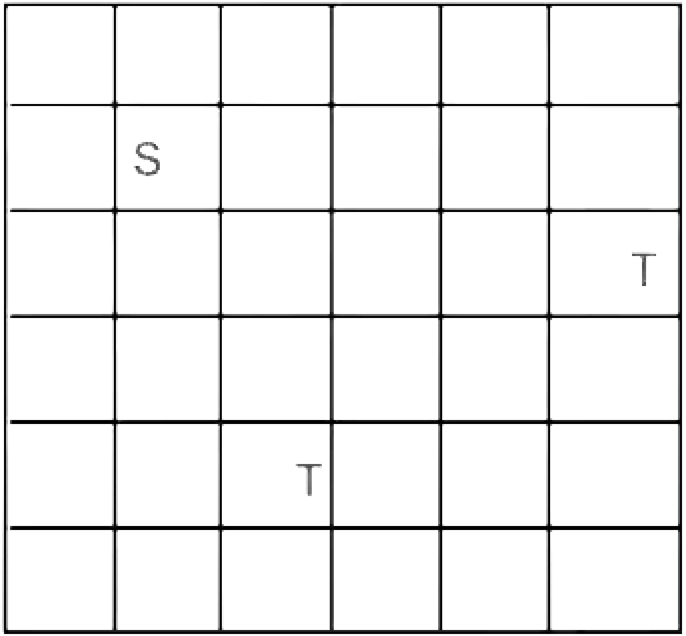


(c) S-T2 path as a source, ***S1*** for T1



(d) Finding Path-2; From ***S1*** to T2

Example 3: CW/HW



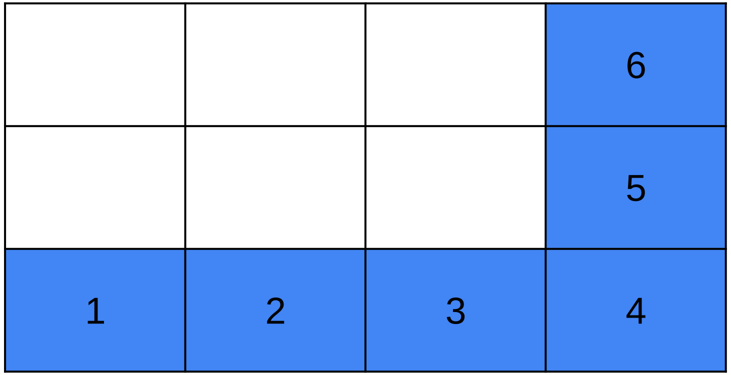
- Find the shortest path for each Target.
- Calculate the memory usage.

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***.
- **Two things** yet to be denoted: empty cell/obstacle.

So, $n = ceil(\log_2(L+2))$ bits per cell

Total memory = ($M \times N \times n$) bits



Here, 3 x 4 grid, 3 bits are required per cell
Total memory = 36 bits

Memory Requirement

Examples:

1. **2000 x 2000** grid
   $n = \log_2 (4001) = 12$
   Memory required = 2000 x 2000 x 12 bits = **6** Mbytes

2. **3000 x 3000** grid
   $n = \log_2 (6001) = 13$
   Memory required = 3000 x 3000 x 13 bits = **14.6** Mbytes

Note: For memory requirement calculations, the maximum path length (**L**) is always considered to be (**M+N-1**) if the sequence is a series of natural numbers (1,2,3,4,5,....). Actual path length can be lower or equal to this value. If we use a repeating sequence, L is total number of unique entity.

Memory Optimization

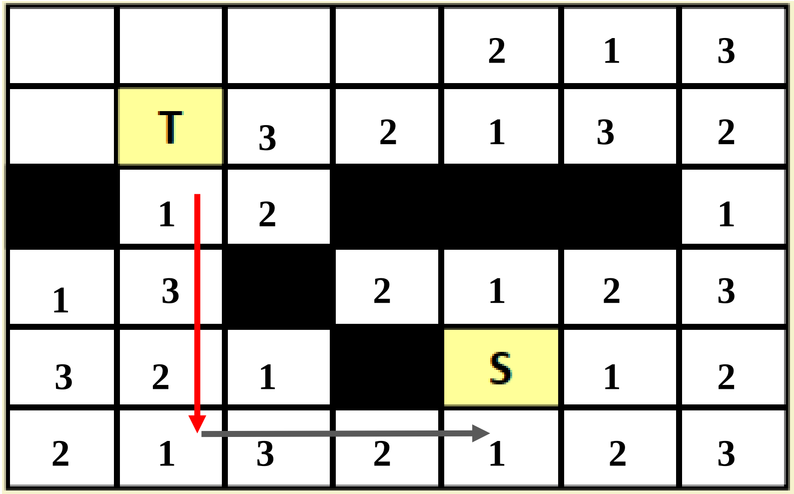
Akers's Observations (1967)

- Adjacent labels for k are either k − 1 or k + 1. Need a labeling scheme such that each label has its preceding label different from its succeeding label.

WAY 1

- Instead of using the sequence 1,2,3,4,5,….. for numbering the cells, the sequence 1,2,3,1,2,3,… is used.(***L=3***)
- For a cell, labels of predecessors and successors are different. So tracing back is easy.
- ***ceil(log$_2$(3+2)) = 3*** bits per cell.

1.5 Mbytes for 2000 x 2000 grid



Memory Optimization [*Minimum Cost*]

WAY 2

- Use the sequence 0,0,1,1,0,0,1,1,….., (***L=2***)
- 0-> First 0; 1-> First 1.
- 0-> Second 0; 1-> Second 1.
- Predecessors and successors are again different. 0, ,1
- ***ceil(log$_2$(2+2)) = 2*** bits per cell.

1.0 Mbytes for 2000 x 2000 grid

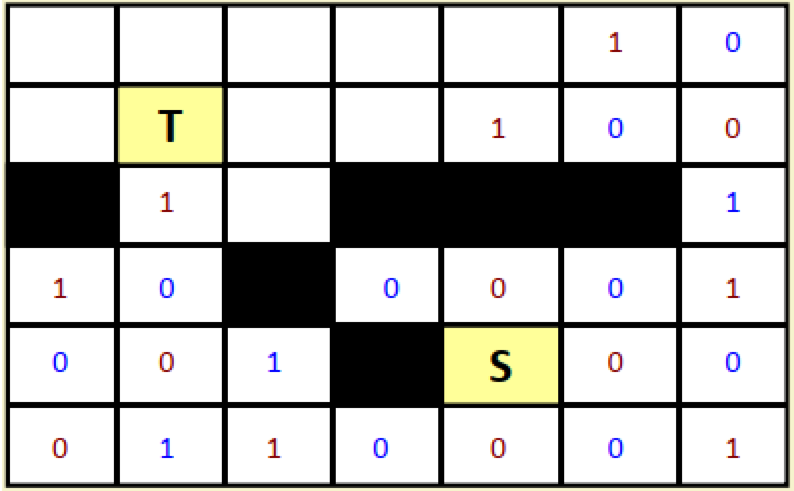


Fig: Minimum Bend Path

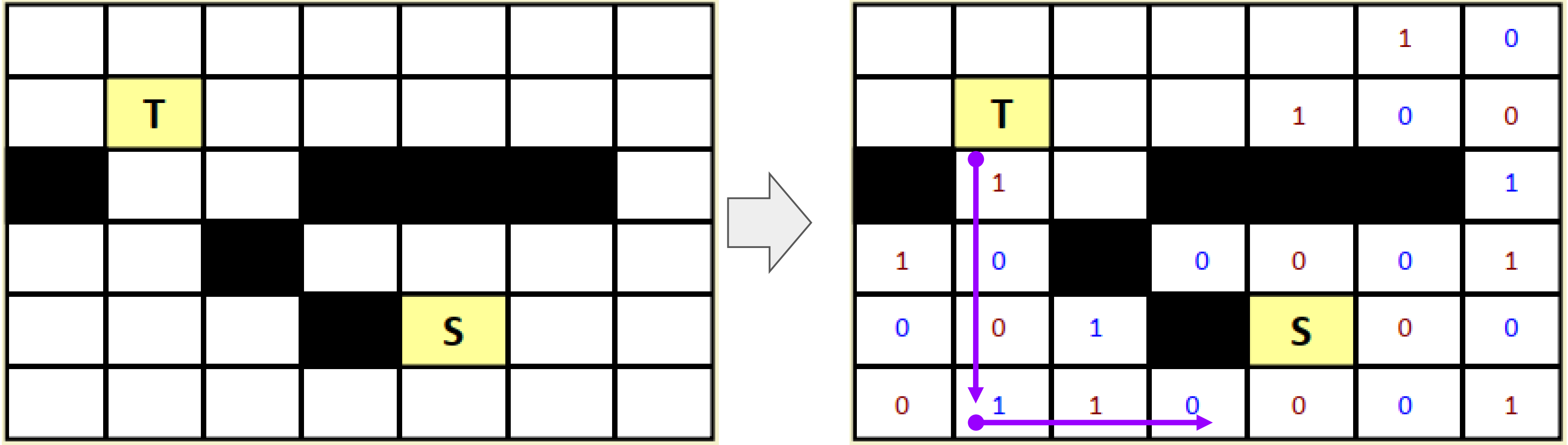
Memory Requirements - Summary

For memory requirement calculations, the critical path length (L) is always considered to be (M+N-1) if the sequence is a series of natural numbers (1,2,3,4,5,....). Actual path length can be lower or equal to this value.

If we use a repeating sequence, L is the total number of unique entities. For example, if we use 1,2,3,1,2,3... repeating sequence, L = 3. If the sequence is 0,1,0,1...., L = 2. So, L does not depend on the actual path length for memory calculation.

If you are asked to calculate the memory used by a definite path, that's a different question. Generally, memory requirements denote the maximum memory required to run the algorithm.

Example 4: Using 001100… Sequence



Label: 0011001

Retrace

References

1. Grid Routing- Lecture 16, *NPTEL Online certification Courses*, IIT Kharagpur.
2. M. A. Breuer, M. Sarrafzadeh and F. Somenzi, "Fundamental CAD algorithms," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 19, no. 12, pp. 1449-1475, Dec. 2000, doi: 10.1109/43.898826.
3. http://users.eecs.northwestern.edu/~haizhou/357/lec6.pdf

Thank You