IIT KHARAGPUR

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Lecture 07: PARTITIONING

PROF. INDRANIL SENGUPTA
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

What is Partitioning?

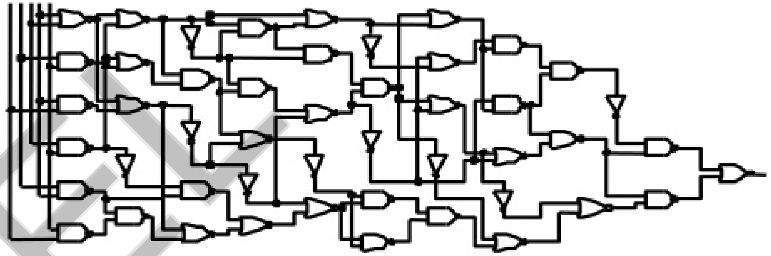
System Design

- Decomposition of a complex system into smaller subsystems.
- Each subsystem can be designed independently.
- Decomposition scheme has to minimize the interconnections between the subsystems.
- Decomposition is carried out hierarchically until each subsystem is of manageable size.

Module 1 | Module 2 | Module n | Interface Information

An Example of Partitioning

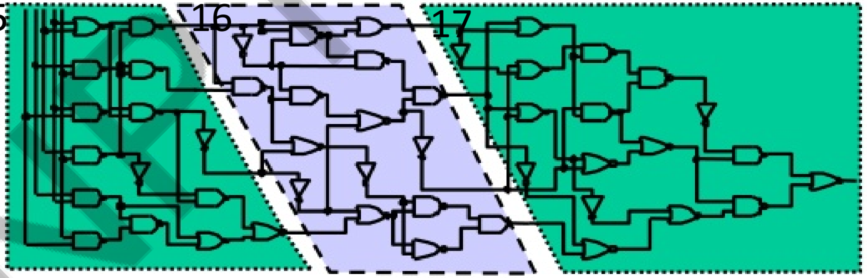
48 gates



Size-1 = 15

Size-2 = 16

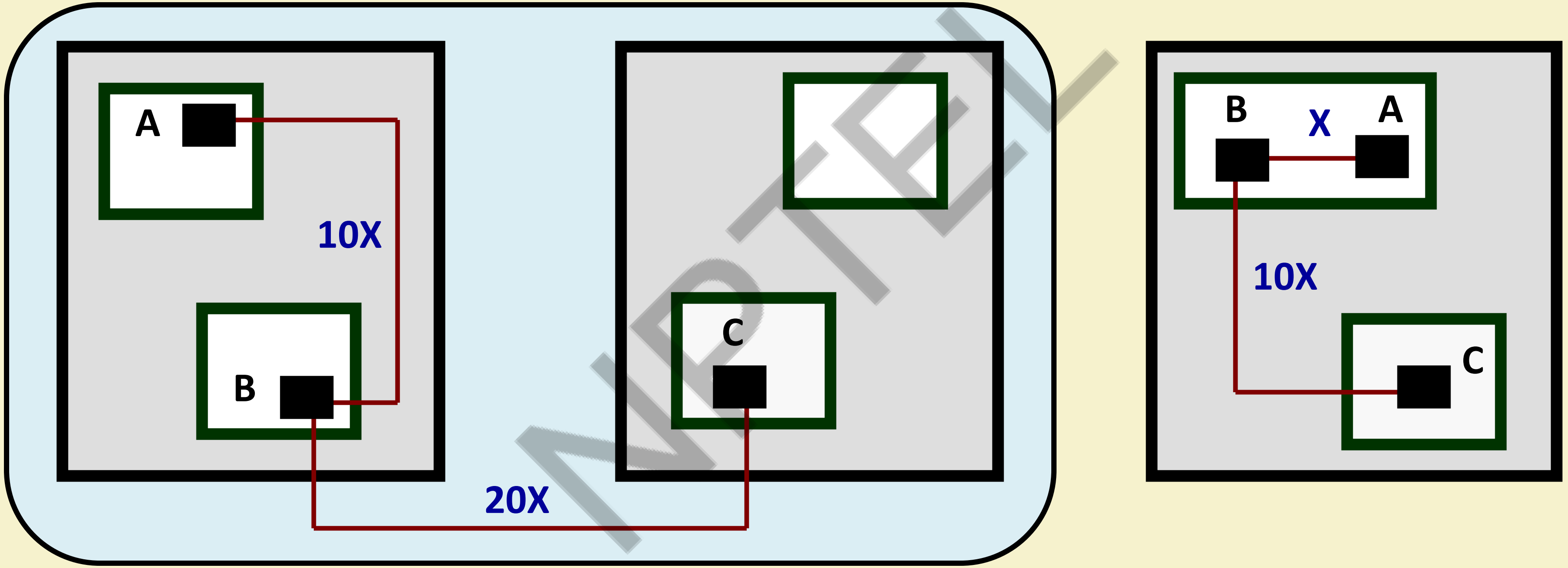
Size-3 = 17



Partitioning at Different Levels

- Can be done at multiple levels:
  - System level
  - Board level
  - Chip level
- Delay implications are different:
  - Intrachip : X
  - Intraboard : 10X
  - Interboard : 20X

Different Delays in a Chip



Problem Formulation

- Partition a given netlist into smaller netlists such that:
  1. Interconnection between partitions is minimized.
  2. Delay due to partitioning is minimized.
  3. Number of terminals is less than a predetermined maximum value.
  4. The area of each partition remains within specified bounds.
  5. The number of partitions also remains within specified bounds.

Partitioning Techniques

- Broadly two classes of algorithms:
  1. Constructive
     - Random selection
     - Cluster growth
     - Hierarchical clustering
  2. Iterative-improvement
     - Min-cut
     - Simulated annealing

Random Selection

- Randomly select nodes one at a time and place them into clusters of fixed size, until the proper size is reached.
- Repeat above procedure until all the nodes have been placed.
- Quality/Performance:
  - Fast and easy to implement.
  - Generally produces poor results.
  - Usually used to generate the initial partitions for iterative placement algorithms.

Cluster Growth

- Starting with a single node, add other nodes to form the partitions based on connectivity.
- Number of clusters n can be an input parameter.

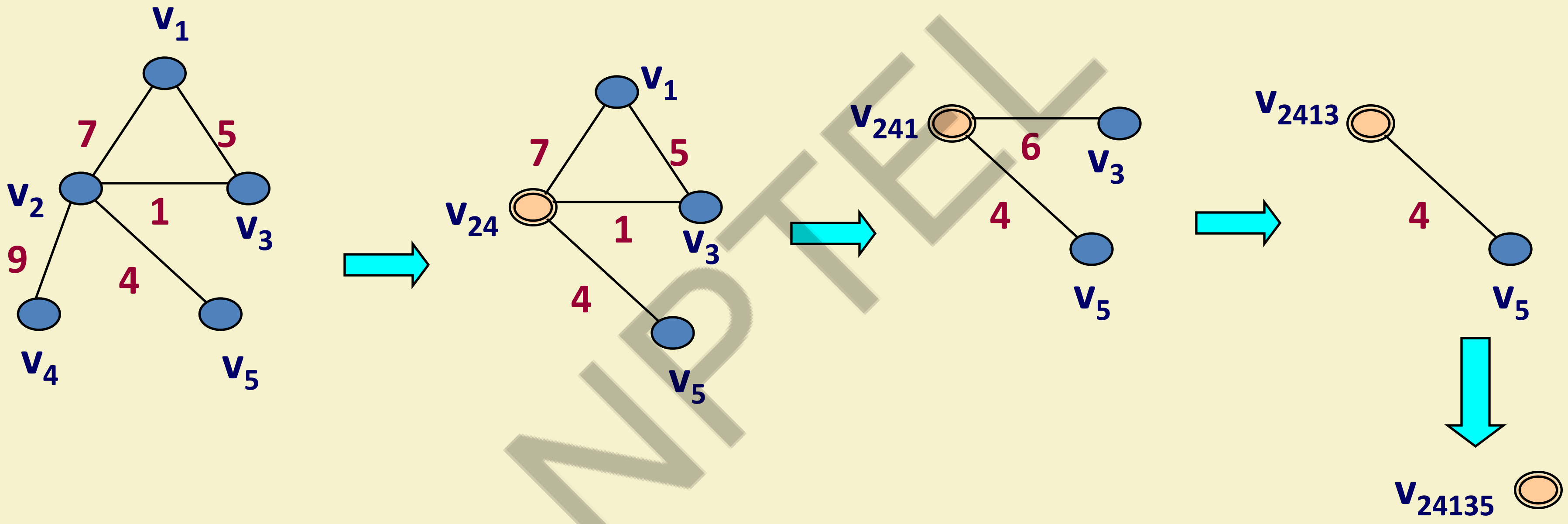
m : size of each cluster, V : set of nodes

```
n = |V| / m ;
for (i=1; i<=n; i++)
    seed = vertex in V with maximum degree;
    $V_i$ = {seed};
    V = V – {seed};
    for (j=1; j<m; j++)
        t = vertex in V maximally connected to $V_i$;
        $V_i$ = $V_i$ U {t};
        V = V – {t};
    endfor
endfor
```

Hierarchical Clustering

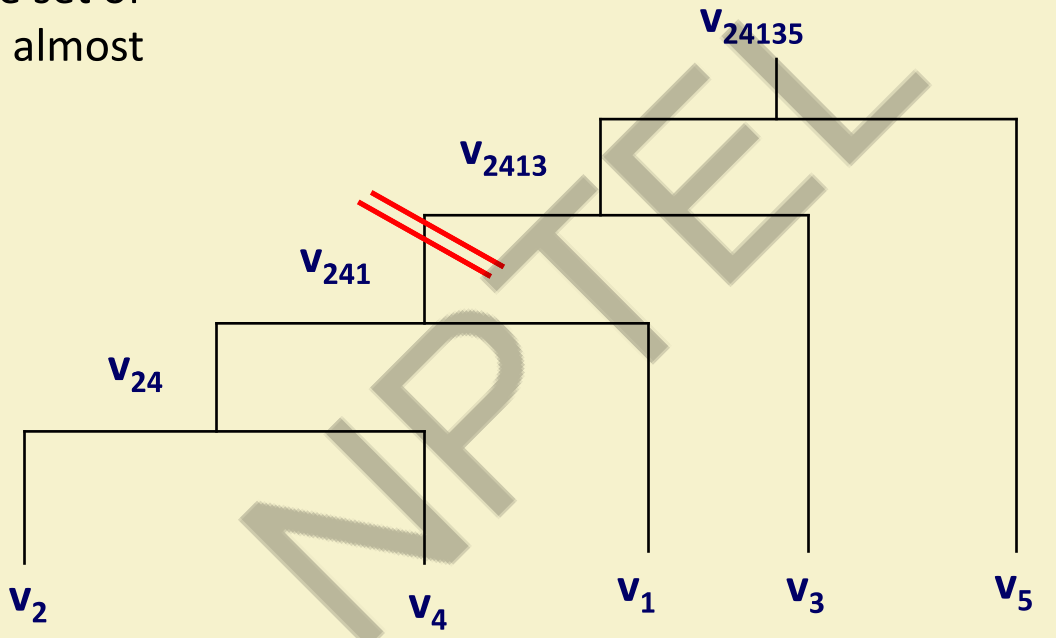
- Consider a set of objects and group them depending on some measure of closeness.
  - The two closest objects are clustered first, and considered to be a single object for further partitioning.
  - The process continues by grouping two individual objects, or an object or cluster with another cluster.
  - We stop when a single cluster is generated and a hierarchical cluster tree has been formed.
    - The tree can be cut in any way to get clusters.

Example



v_1
7
5
v_2
1
v_3
9
4
v_4
v_5
v_1
7
5
v_{24}
1
v_3
4
v_5
v_{241}
6
v_3
4
v_5
v_{2413}
4
v_5
v_{24135}

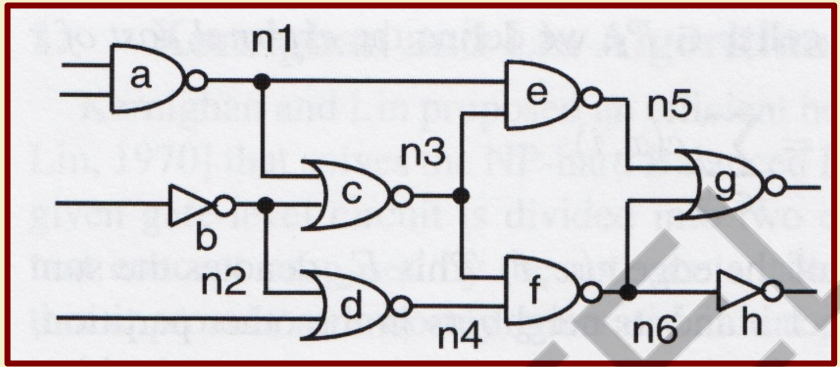
Find an edge whose removal splits the set of vertices into two almost equal partitions.

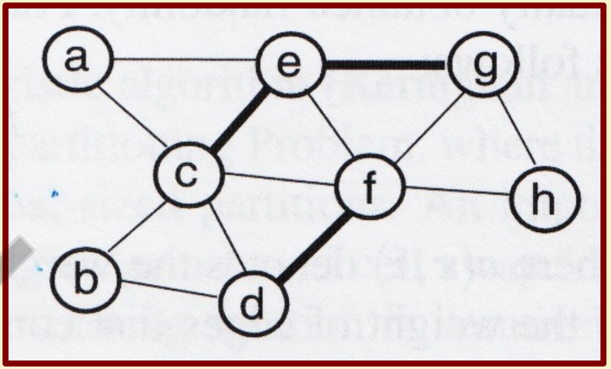


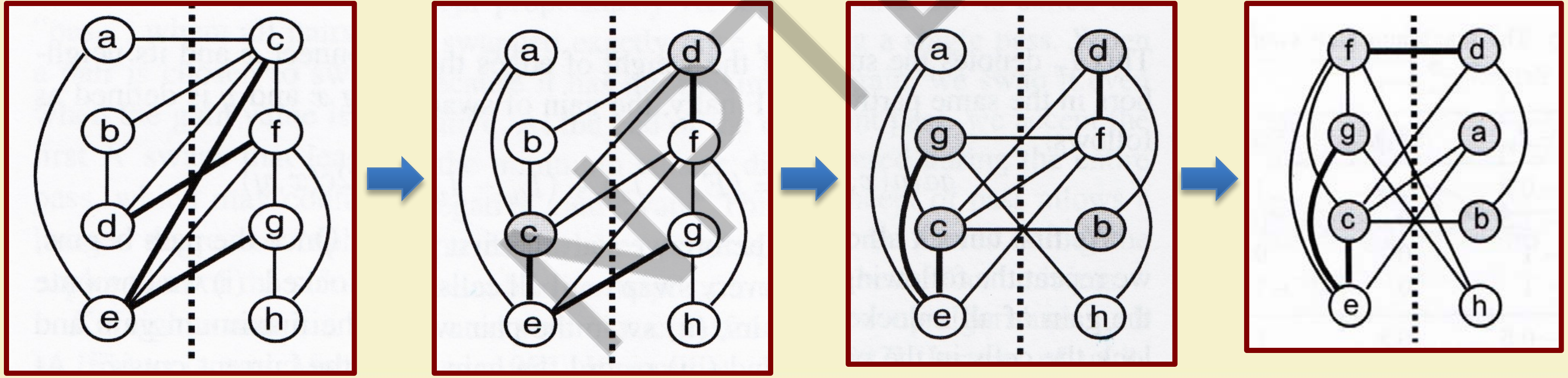
Min-Cut Algorithm (Kernighan-Lin)

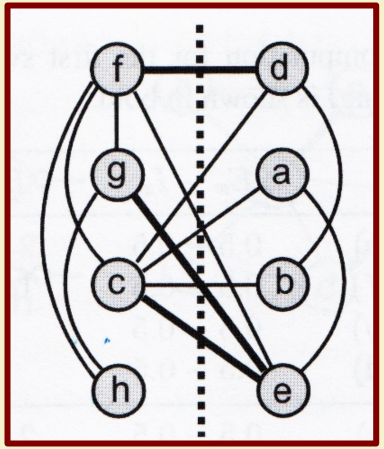
- Basically a bisection algorithm.
  - The input graph is partitioned into two subsets of equal sizes.
- Till the cutsets keep improving:
  - Vertex pairs which give the ***largest decrease*** in cutsize are exchanged.
  - These vertices are then locked.
  - If no improvement is possible and some vertices are still unlocked, the vertices which give the ***smallest increase*** are exchanged.

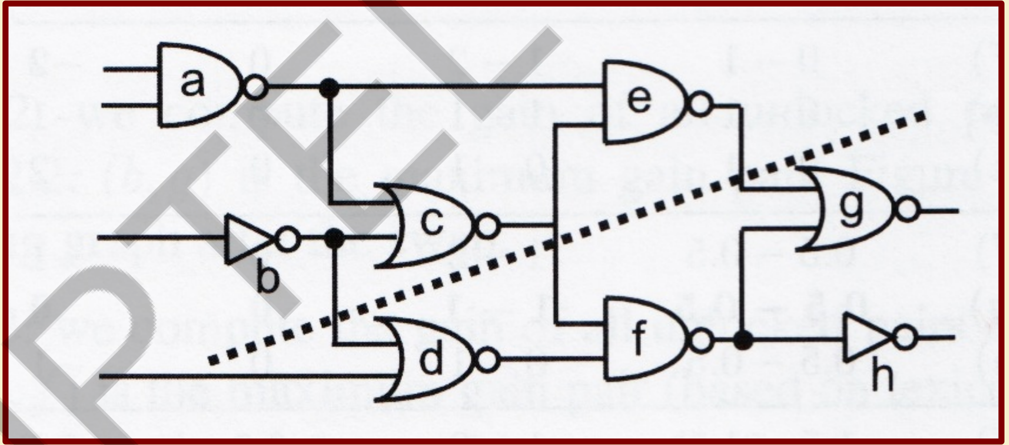
Example


n1
a
e
n5
n3
c
g
b
n2
d
f
h
n4
n6


a
e
g
c
f
h
b
d


a
c
b
f
d
g
e
h
a
d
b
f
c
g
e
h
a
d
g
f
c
b
e
h
f
d
g
a
c
b
e
h


f
d
g
a
c
b
h
e


a
e
c
b
g
d
f
h

- Drawbacks of K-L Algorithm
  - It is not applicable for hyper-graphs.
    - It considers edges instead of hyper-edges.
    - It cannot handle arbitrarily weighted graphs.
    - Partition sizes have to be specified a priori.
  - Time complexity is high :: $O(n^3)$.
  - It considers balanced partitions only.

Extension of K-L Algorithm

- Unequal sized blocks
  - To partition a graph with 2n vertices into two subgraphs of unequal sizes n1 and n2:
    - Divide the nodes into two subsets A and B, containing MIN(n1,n2) and MAX(n1,n2) vertices respectively.
    - Apply K-L algorithm, but restrict the maximum number of vertices that can be interchanged in one pass to MIN(n1,n2).

- Unequal sized elements
  - To generate a two-way partition of a graph whose vertices have unequal sizes:
    - Assume that the smallest element has unit size.
    - Replace each element of size s with s vertices which are fully connected (***s-clique***) with edges of infinite weight.
    - Apply K-L algorithm to the modified graph.

Performance Driven Partitioning

- On-board delay is much larger than on-chip delay.
  - On-chip delay is of the order of nanoseconds.
  - On-board delay can be in the order of milliseconds.
- If a critical path is cut many times by the partition, the delay in the path may be too large to meet the goals of high-performance systems.
- Goal of partitioning in high-performance systems:
  1. Reduce the cut-size.
  2. Minimize the delay in critical paths.
  3. Timing constraints have to be satisfied.

END OF LECTURE 07

IIT KHARAGPUR

NPTEL ONLINE
CERTIFICATION COURSES

NPTEL

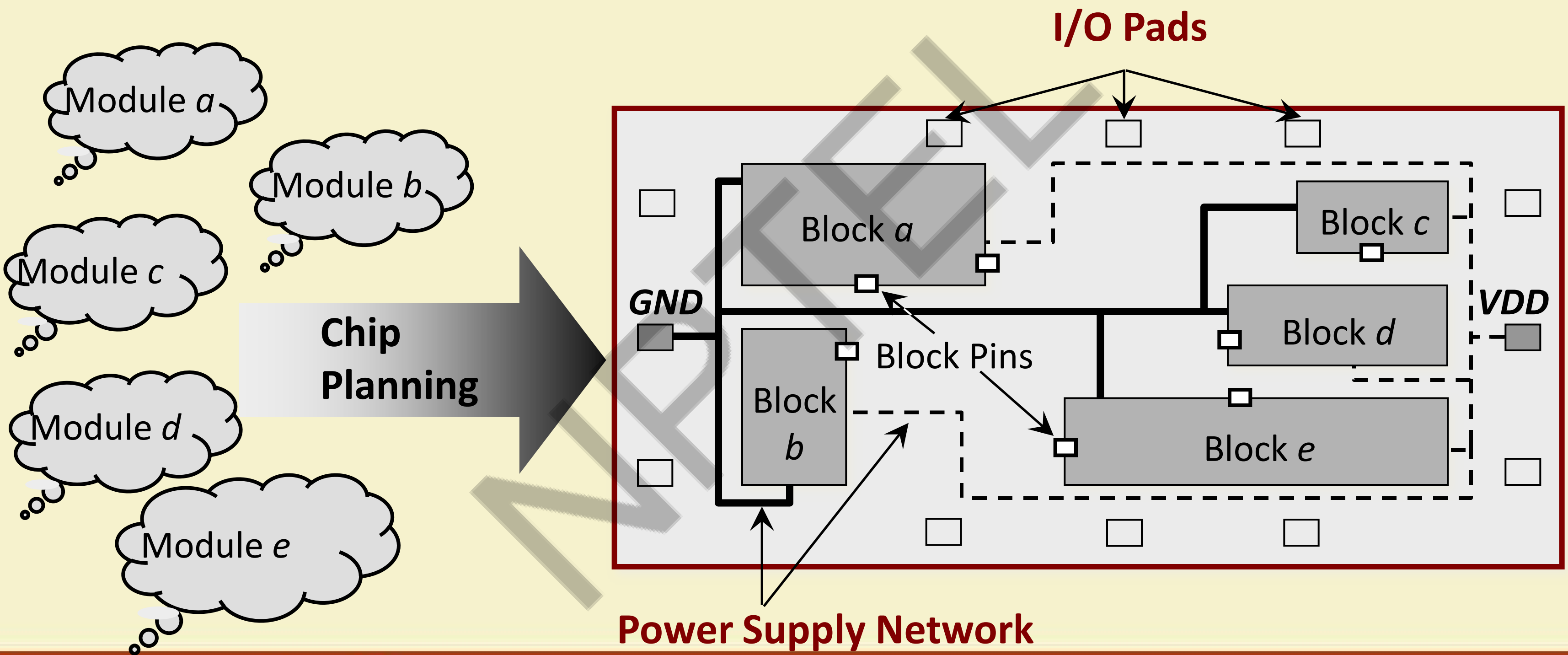
Lecture 08: FLOORPLANNING

PROF. INDRANIL SENGUPTA
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

What is Floorplanning?

- Find approximate locations of a set of modules that need to be placed on a layout surface.
  - Available region typically considered rectangular.
  - Modules are also typically rectangular in shape, but there can be exceptions (e.g. L-shaped modules).

An Example Floorplan



Problem Definition

- Input:
  - n Blocks with areas $A_1, A_2, \dots, A_n$
  - Bounds r_i and s_i on the aspect ratio of block B_i
- Output:
  - Coordinates (x_i, y_i), width w_i and height h_i for each block such that $h_i w_i = A_i$ and $r_i \le h_i/w_i \le s_i$
- Objective:
  - Minimize area; reduce wire length for critical lengths.

Floorplanning and Placement: Differences

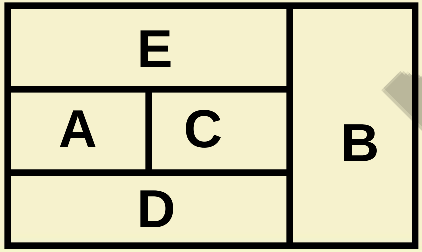
- The problems are similar in nature.
- Main differences:
  - Floorplanning:- Some of the blocks may be flexible, and the exact locations of the pins are not yet fixed.
  - Placement:- All blocks have well-defined geometrical shapes, with defined pin locations. We keep separate space for routing.

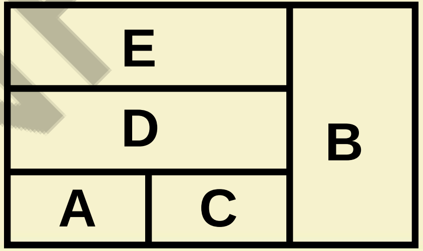
- Points to note:
  - Floorplanning problem is more difficult as compared to placement.
    - Multiple choice for the shape of a block.
  - In some of the VLSI design styles, the two problems are identical.

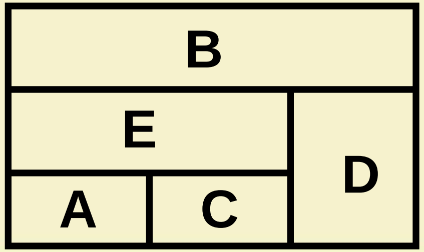
An Example for Rigid Blocks

Module	Width	Height
A	1	1
B	1	3
C	1	1
D	1	2
E	2	1

Some of the Feasible Floorplans







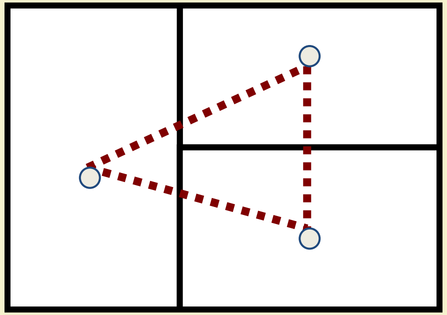
Design Style Specific Issues

- <u>Full Custom</u>
  - All the steps required for general cells.
- <u>Standard Cell</u>
  - Dimensions of all cells are fixed.
  - Floorplanning problem is simply the placement problem.
  - For large netlists, two steps:
    - First do global partitioning.
    - Next carry out placement for the individual regions.
- <u>Gate Array</u>
  - Floorplanning problem same as placement problem.

Estimating Cost of a Floorplan

- The number of feasible solutions of a floorplanning problem is very large.
  - Finding the best solution is NP-hard.
- Several criteria used to measure the quality of floorplans:
  a) Minimize area
  b) Minimize total length of wire
  c) Maximize routability
  d) Minimize delays
  e) Any combination of above

- How to determine area?
  - Not difficult.
  - Can be easily estimated because the dimensions of each block is known.
  - Area *A* computed for each candidate floorplan.



- How to determine wire length?
  - A coarse measure is used.
  - Based on a model where all I/O pins of the blocks are merged and assumed to reside at its center.
  - Overall wiring length $L = \Sigma_{i,j} (c_{ij} * d_{ij})$

    where c_{ij} : number of connections between blocks i and j

    d_{ij} : Manhattan distances between the centres of rectangles of blocks i and j

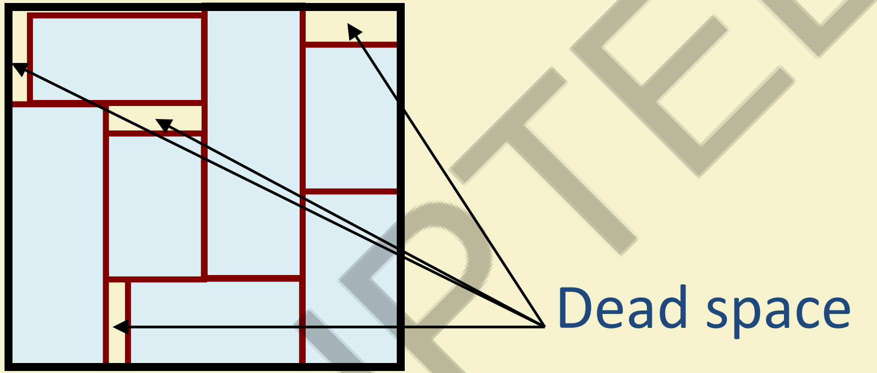
- Typical cost function used:

$$\text{Cost} = w1 * A + w2 * L$$

where w1 and w2 are user-specified parameters.

Dead Space

- Dead space is the space that is wasted:

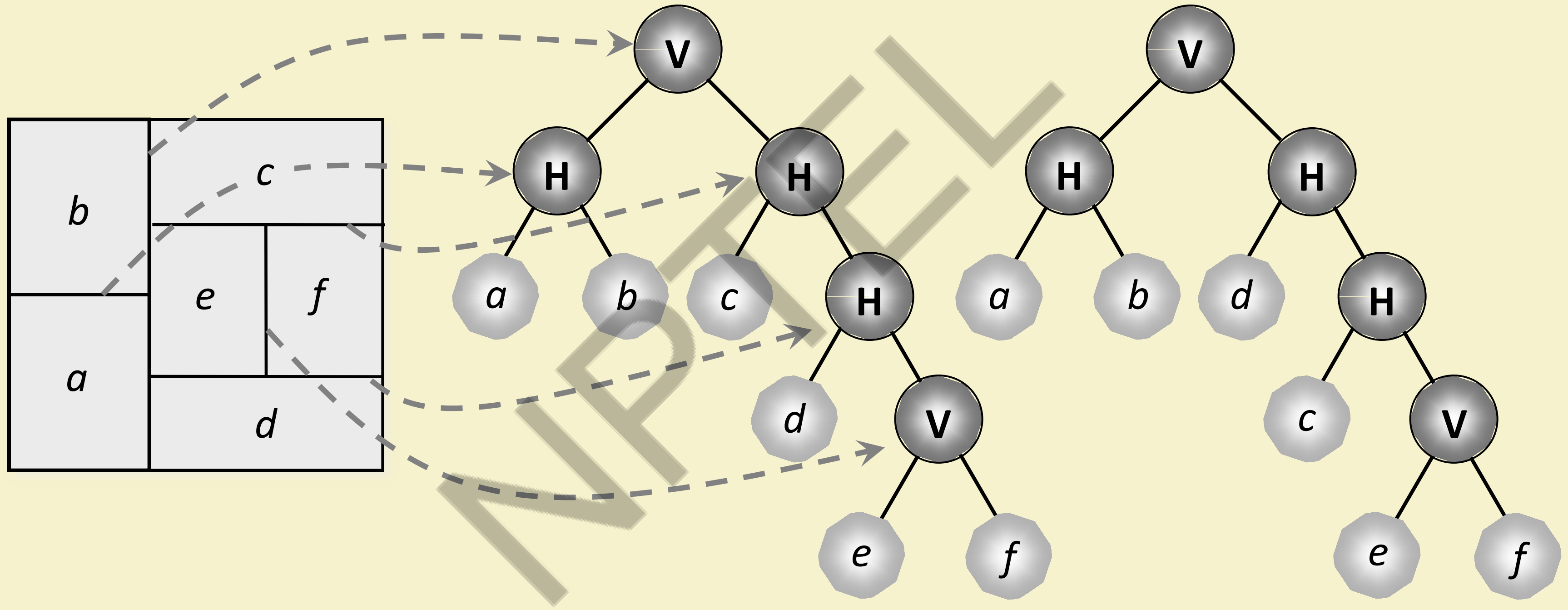


- Minimizing area is the same as minimizing deadspace.
- Dead space percentage is computed as: $(A - \Sigma_i A_i) / A \times 100\%$

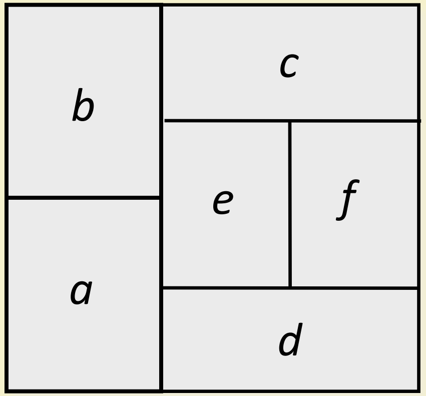
Slicing Structure

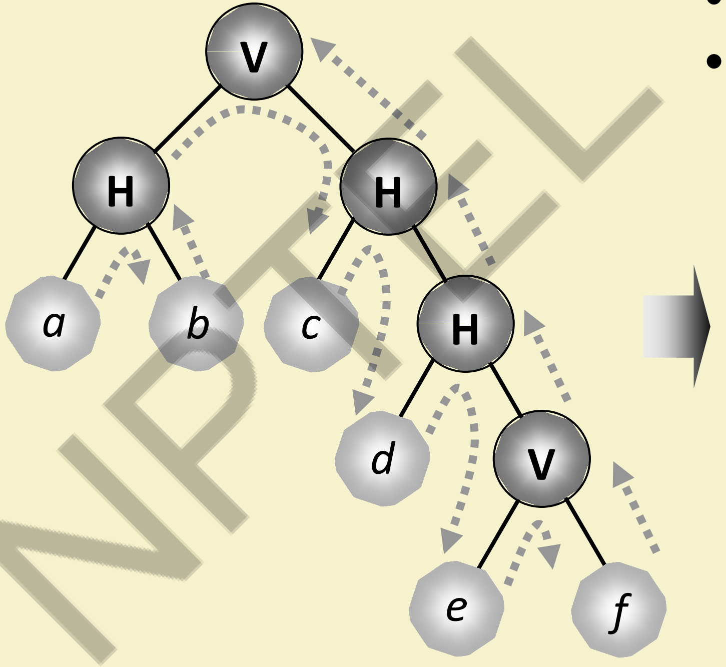
- A rectangular dissection that can be obtained by repeatedly splitting rectangles by horizontal and vertical lines into smaller rectangles.
- Slicing Tree:
  - A binary tree that models a slicing structure.
  - Each node represents a *vertical cut line* (V), or a *horizontal cut line* (H).
    - A third kind of node called *Wheel (W)* appears for non-sliceable floorplans (discussed later).
  - Each leaf is a basic block (rectangle).

A Slicing Floorplan and Two Slicing Trees



Polish Expression



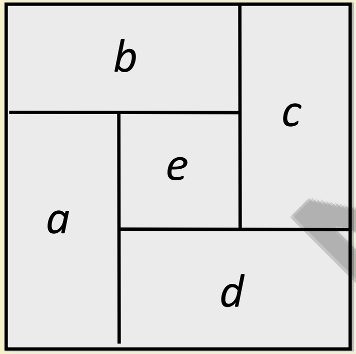


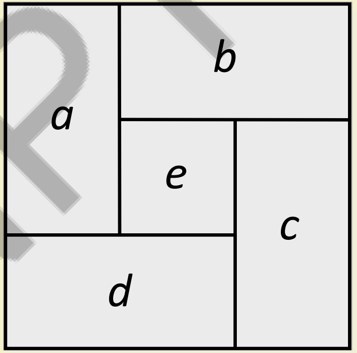
- V: | H: –
- For n number of leaves, length of the expression is 2n-1

A B – C D E F | – – |

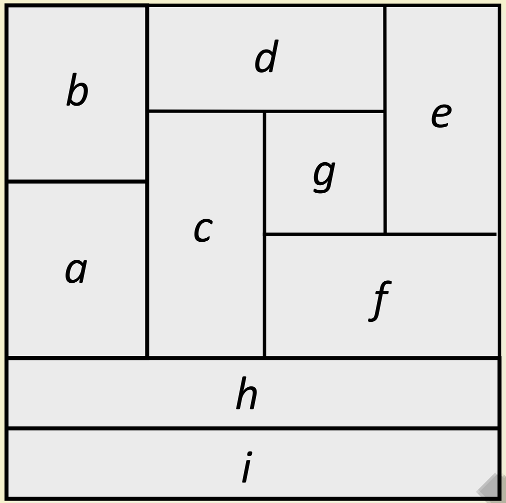
A Non-Slicing Floorplan

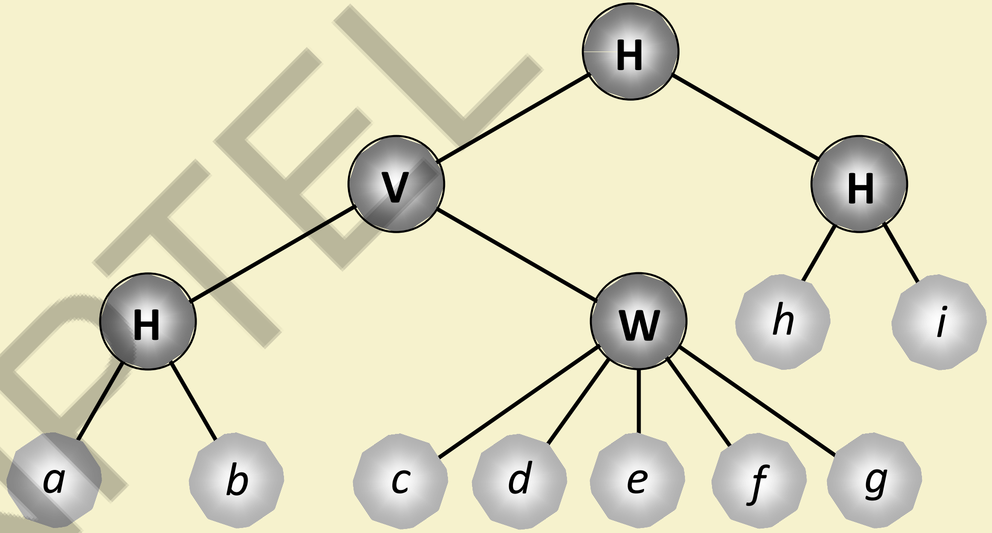
- One that may not be obtained by repetitively subdividing alone.
  - Also called a WHEEL.





A Hierarchical Floorplan

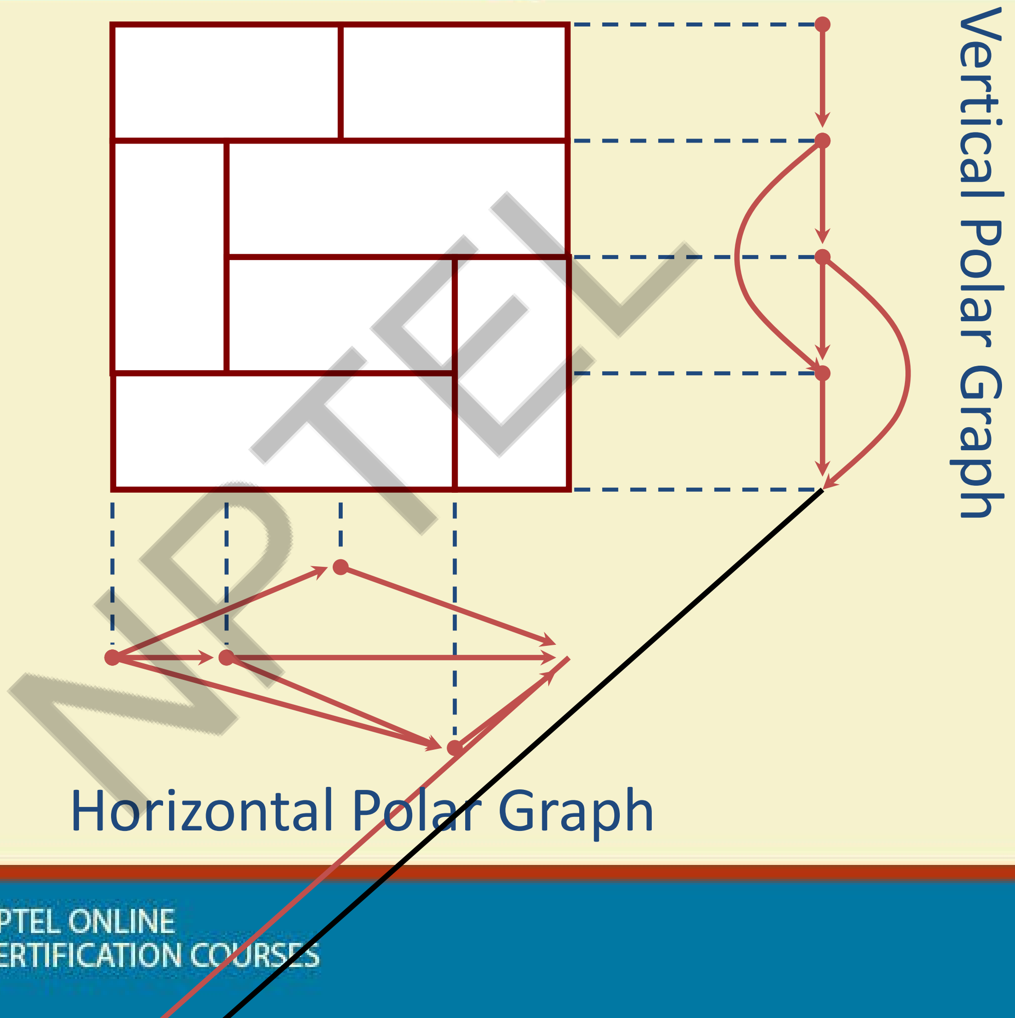




Polar Graph Representation

- An alternate graph representation of floorplan.
- Each floorplan is modeled by a pair of directed acyclic graphs:
  - Horizontal polar graph
  - Vertical polar graph
- For horizontal (vertical) polar graph,
  - Vertex represents vertical (horizontal) channel
  - Edge indicates that the 2 channels are on 2 sides of a block
  - Edge weight indicates the width (height) of the block

Polar Graph Example



END OF LECTURE 08

IIT KHARAGPUR

NPTEL ONLINE
CERTIFICATION COURSES

NPTEL

Lecture 09: FLOORPLANNING ALGORITHMS

PROF. INDRANIL SENGUPTA
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Floorplanning Algorithms

- Several broad classes of algorithms:
  - Integer programming based
  - Rectangular dual graph based
  - Hierarchical tree based
  - Simulated annealing based
  - Other variations

Integer Linear Programming Formulation

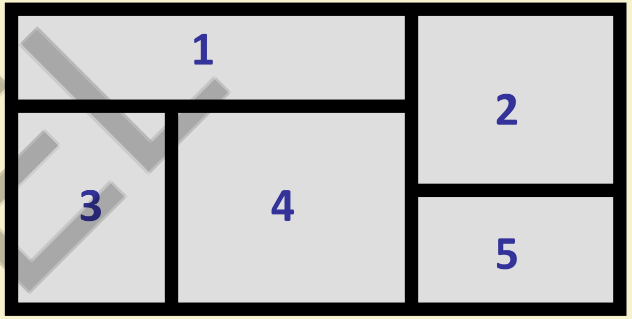
- The problem is modeled as a set of linear equations using 0/1 integer variables.
- ILP solver used to obtain optimal solution based on a defined cost function.
- Can be used only for small problem instances.
  - Very high computational complexity.

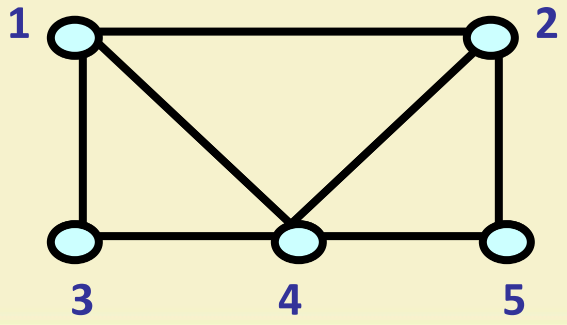
Rectangular Dual-Graph Approach

- Basic Concept:
  - Output of partitioning algorithms represented by a graph.
  - Floorplan obtained by converting the graph into its rectangular dual.
- The rectangular dual of a graph satisfies the following properties:
  - Each vertex corresponds to a distinct rectangle.
  - For every edge, the corresponding rectangles are adjacent.

A Rectangular Floorplan & its Dual Graph

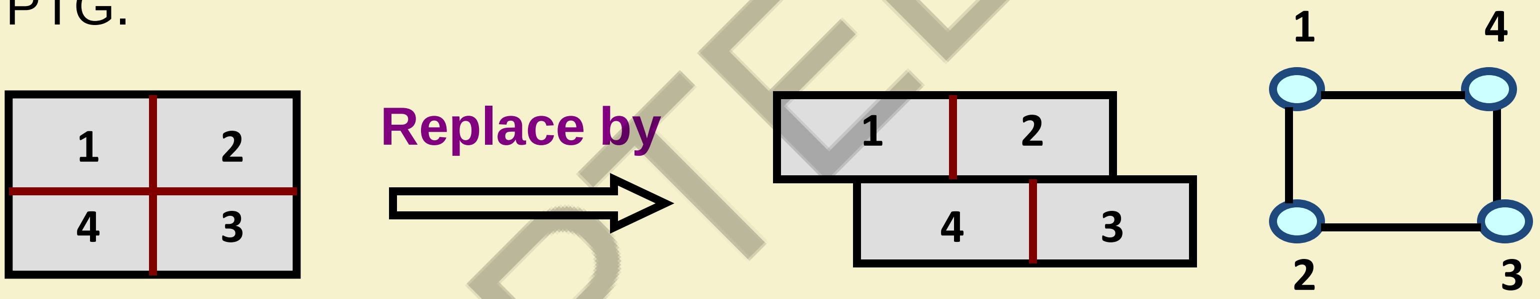
- Without loss of generality, we assume that a rectangular floorplan contains no cross junctions.
- Under this assumption, the dual graph of a rectangular floorplan is a *planar triangulated graph (PTG).*



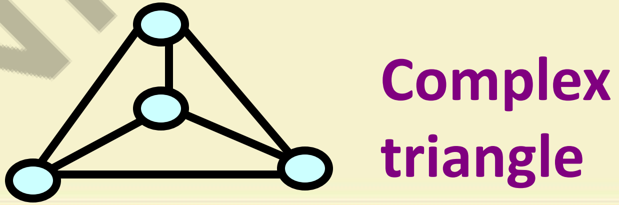


Contd.

- Every dual graph of a rectangular floorplan (without cross junction) is a PTG.



- However, not every PTG corresponds to a rectangular floorplan.

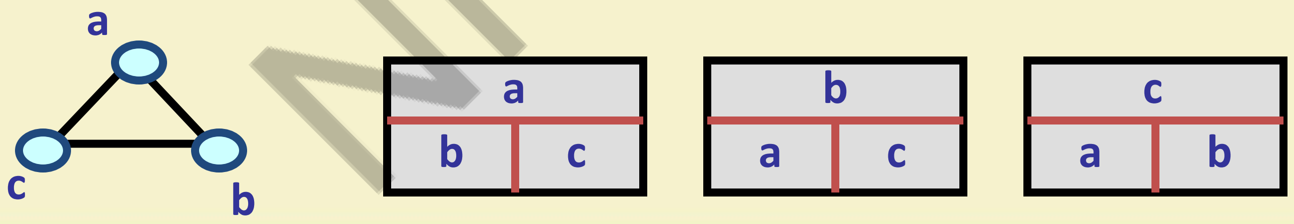


Drawbacks

- An interesting approach to floorplanning.
- The main problem concerns the existence of the rectangular dual, i.e. the elimination of complex triangles.
  - Select a minimum set E of edges such that each complex triangle has at least one edge in E.
  - A vertex can be added to each edge of E to eliminate all complex triangles.
  - The weighted complex triangle elimination problem has been shown to be NP-complete.
    - Some heuristics are available.

Hierarchical Approach

- Widely used approach to floorplanning.
  - Based on a divide-and-conquer paradigm.
  - At each level of the hierarchy, only a small number of rectangles are considered.
- A small graph, and all possible floorplans.

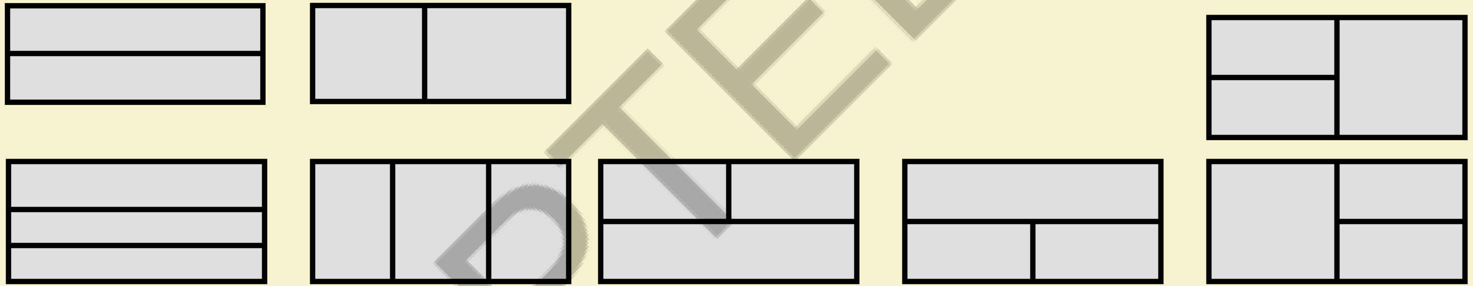


- After an optimal configuration for the three modules has been determined, they are merged into a larger module.
- The vertices 'a', 'b', 'c' are merged into a super vertex at the next level.
- The number of floorplans increases exponentially with the number of modules 'd' considered at each level.
  - 'd' is thus limited to a small number (typically $d < 6$).

- All possible floorplans for:

d = 2

d = 3



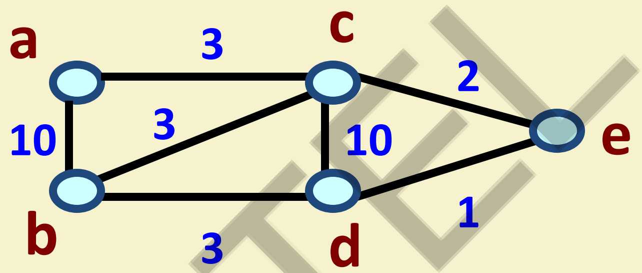
Hierarchical Approach :: Bottom-Up

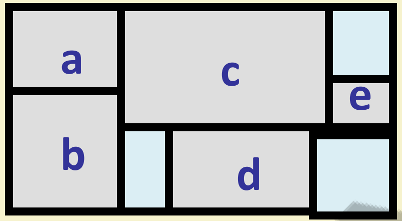
- Hierarchical approach works best in bottom-up fashion.
- Modules are represented as vertices of a graph, while edges represent connectivity.
  - Modules with high connectivity are clustered together.
    - Number of modules in each cluster $\leq$ d.
  - An optimal floorplan for each cluster is determined by exhaustive enumeration.
  - The cluster is merged into a larger module for high-level processing.

- A Greedy Procedure
  - Sort the edges in decreasing weights.
  - The heaviest edge is chosen, and the two modules of the edge are clustered in a greedy fashion.
    - Restriction: number of modules in each cluster $\leq$ d.
  - In the next higher level, vertices in a cluster are merged, and edge weights are summed up accordingly.

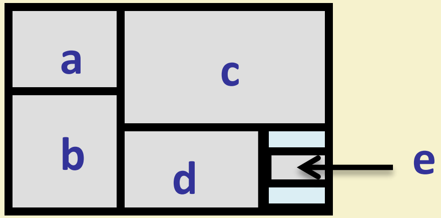
- <u>Problem</u>
  - Some lightweight edges may be chosen at higher levels in the hierarchy, resulting in adjacency of two clusters of highly incompatible areas.
- <u>Possible solution</u>
  - Arbitrarily assign a small cluster to a neighboring cluster when their sizes will be too small for processing at a higher level of the hierarchy.

Example





Greedy clustering



Merging small clusters

Hierarchical Approach :: Top-Down

- The fundamental step is the partitioning of modules.
  - Each partition is assigned to a child floorplan.
  - Partitioning is recursively applied to the child floorplans.
- Major issue here is to obtain balanced graph partitioning.
  - k-way partitioning, in general.
- Not very widely used due to the difficulty of obtaining balanced partitions.

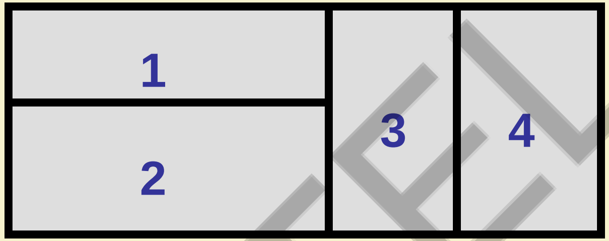
Simulated Annealing

- Important issues in the design of a simulated annealing optimization problem:
    1. The solution space.
    2. The movement from one solution to another.
    3. The cost evaluation function.
- A solution by Wong is applicable to sliceable floorplans only.
    - Floorplan can be represented by a tree.
    - Postfix notations used for easy representation and manipulation.

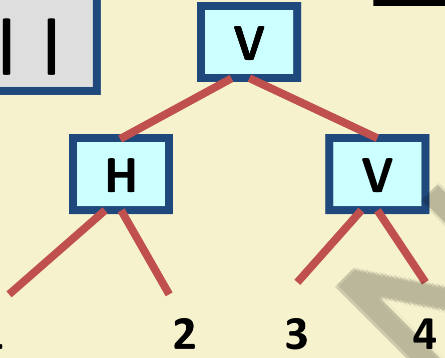
Some Notations

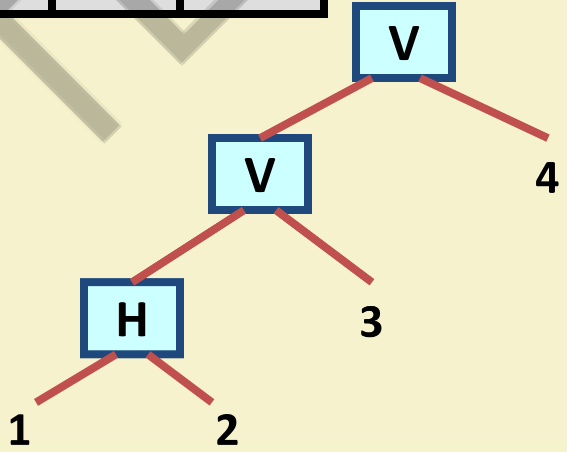
- Dissection operators defined:
  - ijH means rectangle j is on top of rectangle i.
  - ijV means rectangle j is on left of rectangle i.
- Normalized Polish expression:
  - A Polish expression corresponding to a floorplan is called *normalized* if it has no consecutive H’s or V’s.
  - Used for the purpose of removing redundant solutions from the solution space.
    - Several Polish expressions may exist for the same slicing floorplan.
    - Same floorplan can have more than one slicing tree.

Example: Two tree representations of a floorplan



1 2 – 3 4 | |





1 2 – 3 | 4 |

Normalized

Parameters of the Algorithm

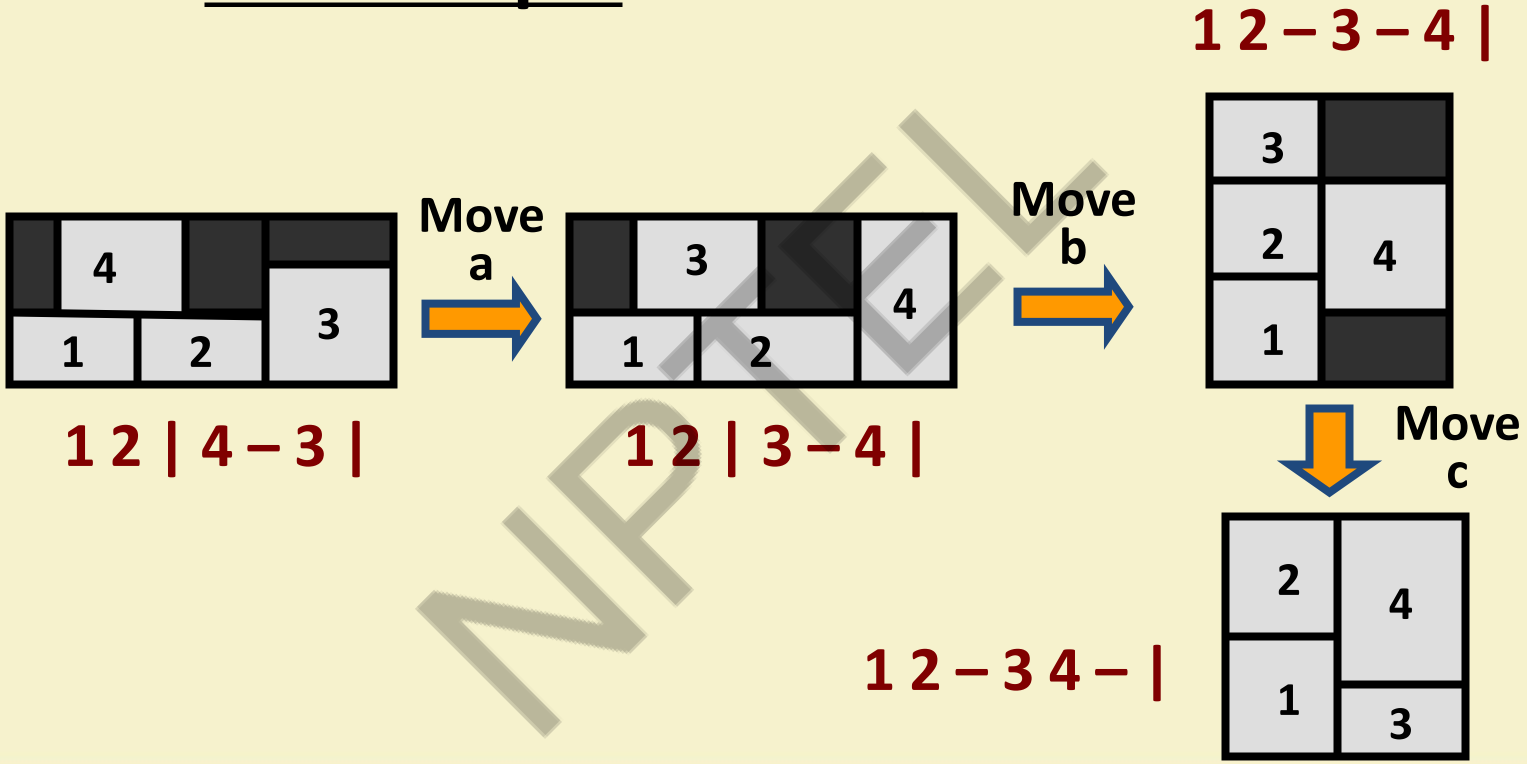
- Solution Perturbations (Move)
  a) Swap two adjacent operands
  b) Complement a series of operators between two operands (called a *chain*). ➔ **V′ = H and H′ = V**
  c) Swap two adjacent operand and operator.
- We accept a move only it is results in a normalized expression.
  - Only move "c" may result in a non-normalized solution.
  - We need to check only when move "c" is applied.

- Typical cost function:

$$\text{Cost} = w_1 * A + w_2 * L$$

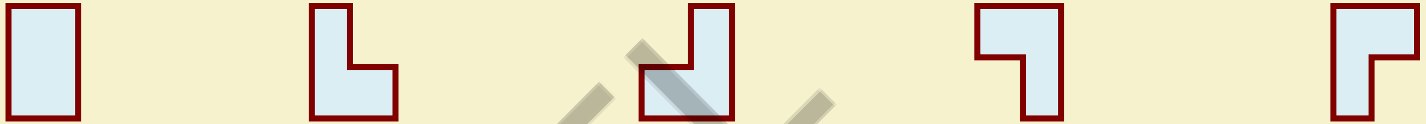
where A is the area of the smallest rectangle enveloping the given basic rectangles, L is the overall interconnection length, w_1 & w_2 are user-specified parameters.

An Example

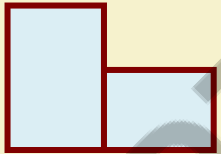


Rectangular and L-shaped Blocks

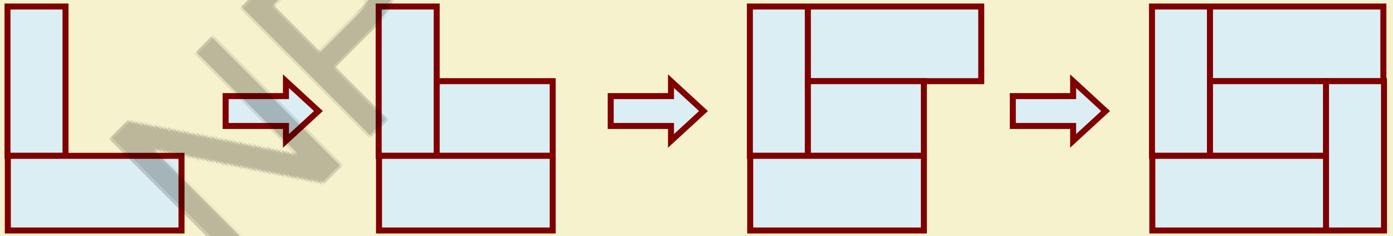
- Possible shapes:



- Note that L-shaped blocks can be produced by merging rectangular blocks.



- Can generate non-slicing floorplans.



END OF LECTURE 09

IIT KHARAGPUR

NPTEL ONLINE
CERTIFICATION COURSES

NPTEL

Lecture 10: PIN ASSIGNMENT

PROF. INDRANIL SENGUPTA
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

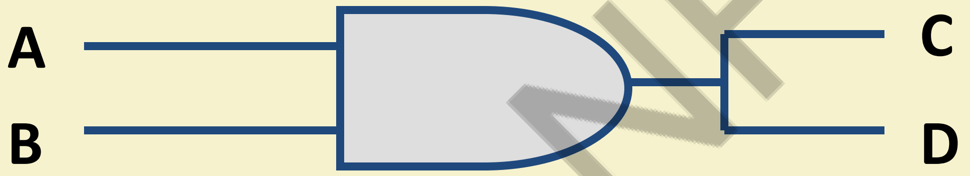
Introduction

- The purpose is to define the signal that each pin will receive.
- It can be done
  - During floorplanning
  - During placement
  - After placement is fixed
- For un-designed blocks, a good assignment of pins improves placement.
- If the blocks are already designed, still some pins can be exchanged.

Pin Assignment

- Input:
  - A placement of blocks.
  - Number of pins on each block, possibly an ordering.
  - A netlist.
- Requirements:
  - To determine the pin locations on the blocks.
- Objectives:
  - To minimize net-length.

- Functionally equivalent pins:
  - Exchanging the signals does not affect the circuit.
- Equipotential pins:
  - Both are internally connected and represent the same net.

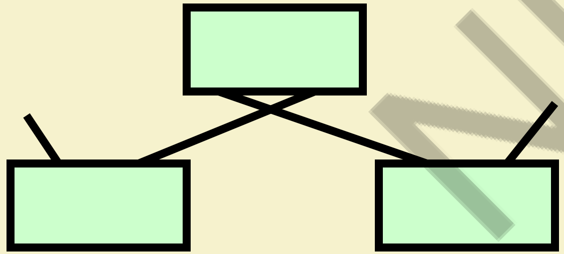


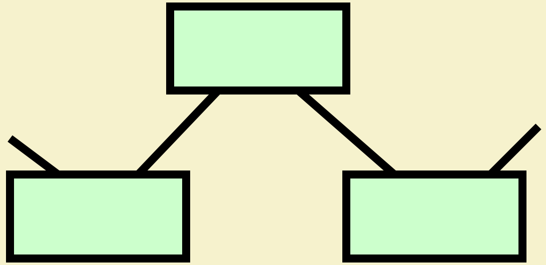
A,B :: functionally equivalent

C,D :: equipotential

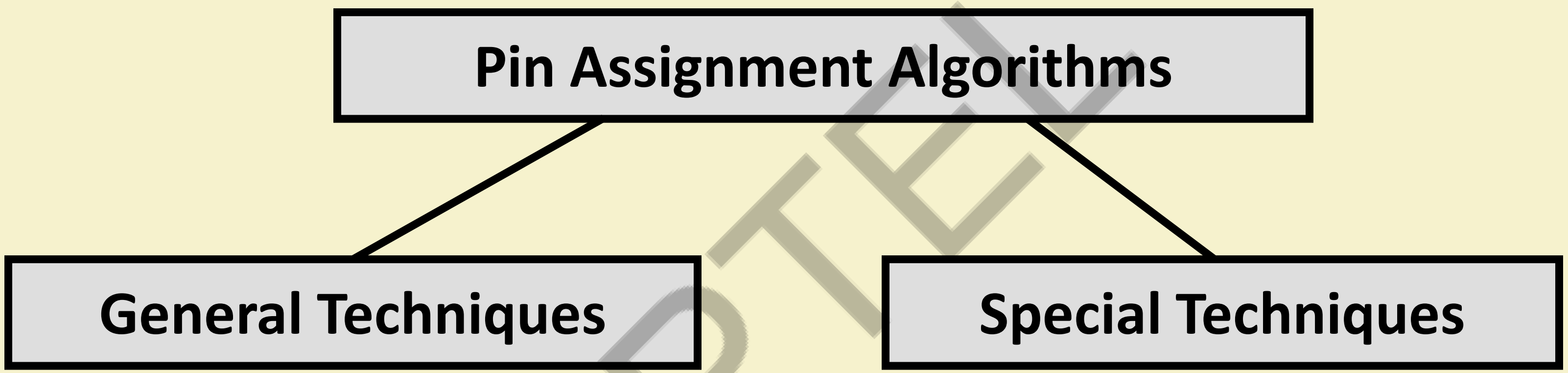
Problem Formulation

- Purpose is to optimize the assignment of nets within a functionally equivalent (or equipotential) pin groups.
- <u>Objective</u>:
  - To reduce congestion or reduce the number of crossovers.





Classification of Algorithms



Concentric Circle Mapping

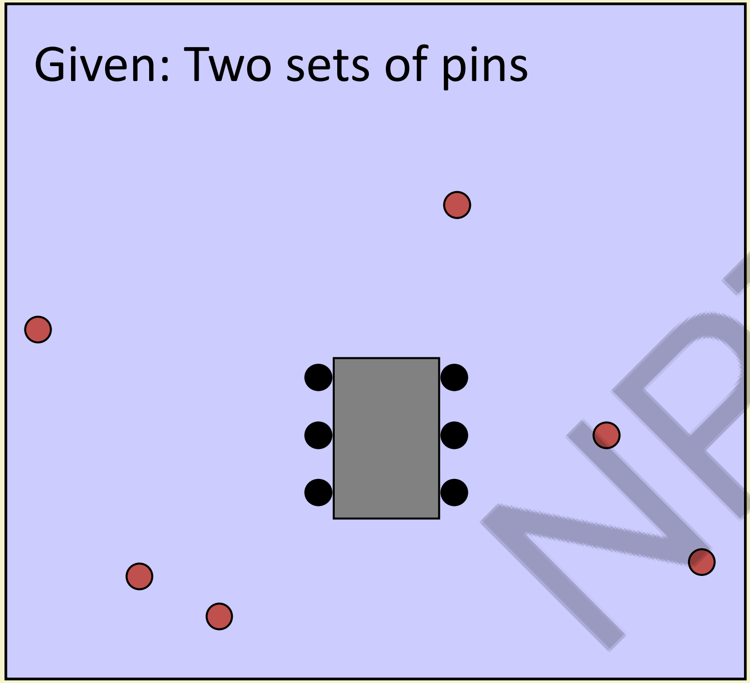
Topological Method

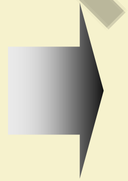
Nine Zone Method

Channel Pin Assignment

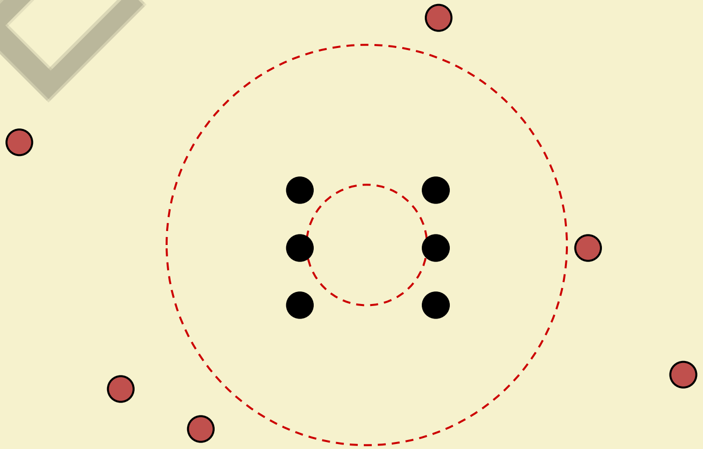
Concentric Circle Mapping

Given: Two sets of pins

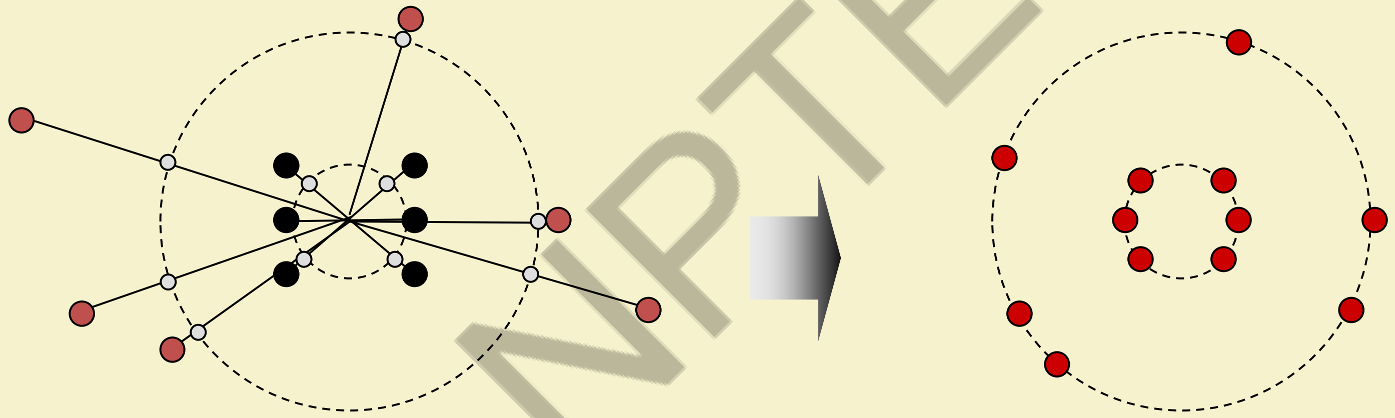




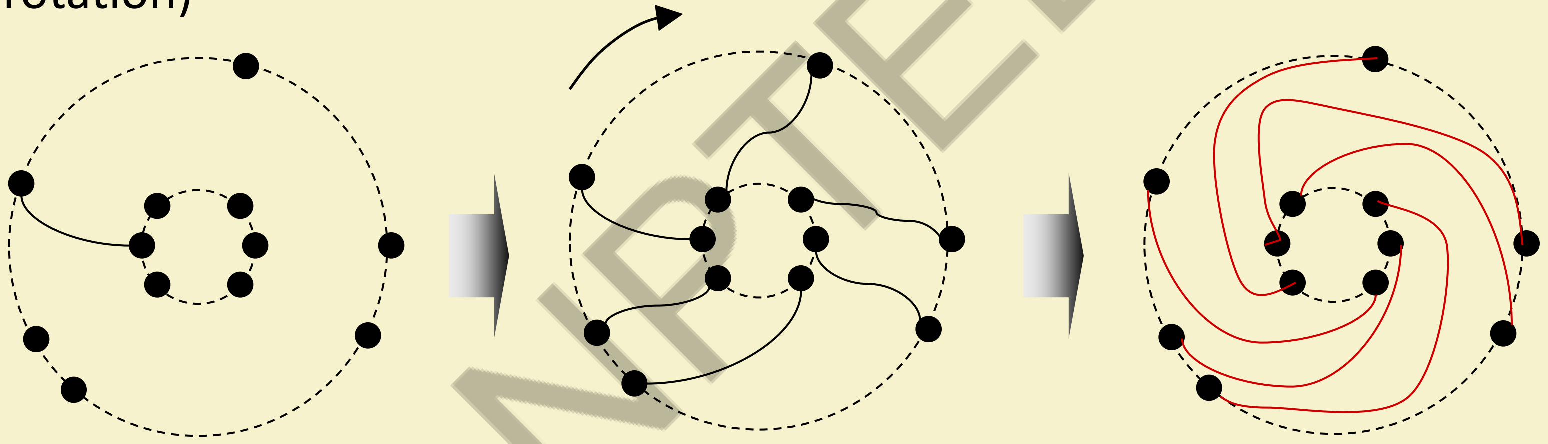
(1) Determine the circles



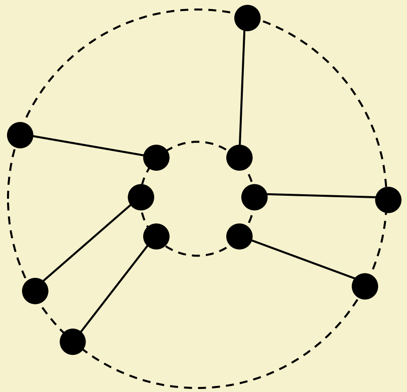
(2) Determine the points

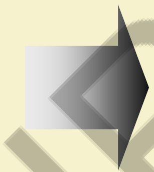


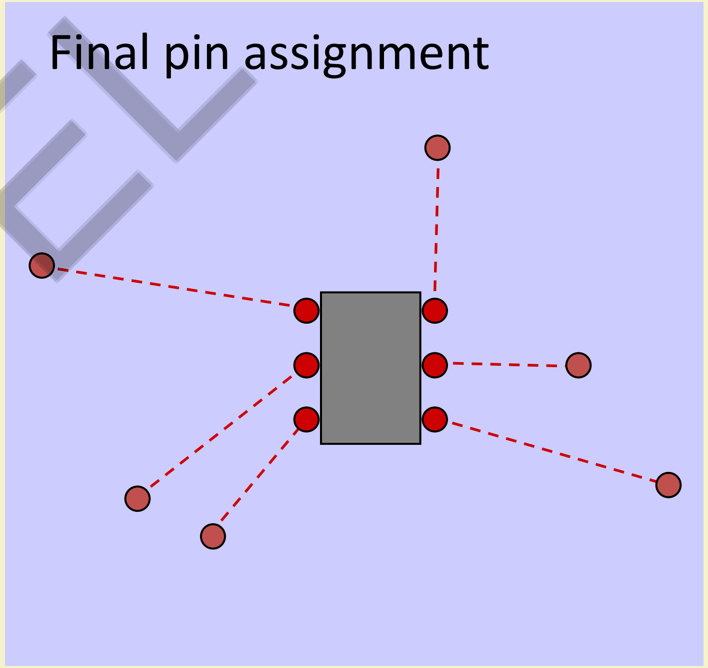
(3) Determine initial mapping and (4) optimize the mapping (complete rotation)



(4) Best mapping

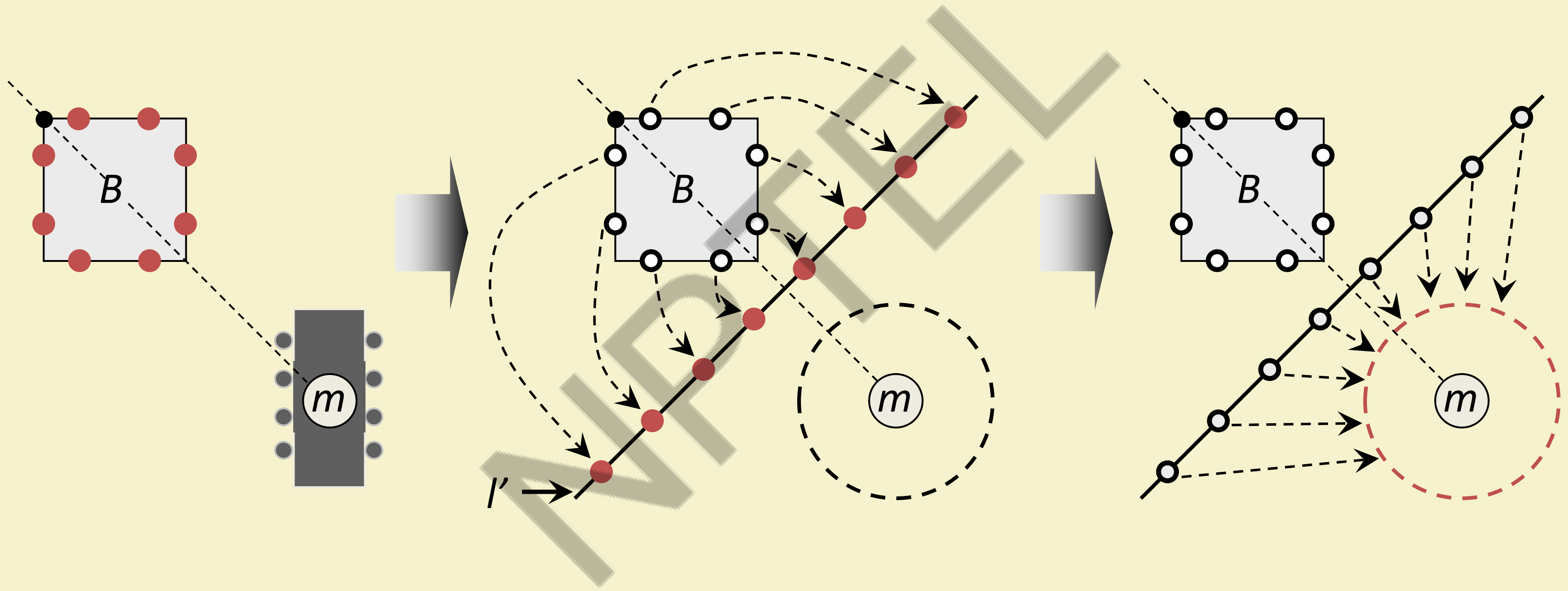






Final pin assignment

Pin Assignment to an External Block B



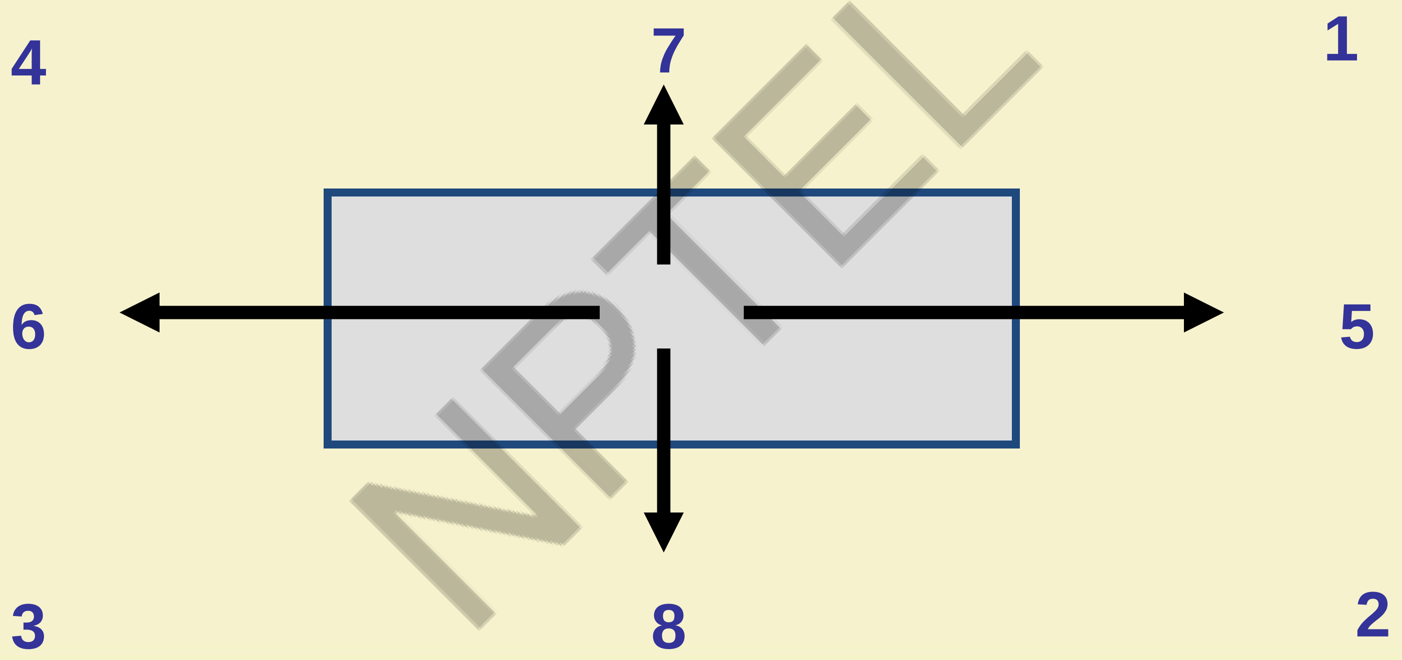
Topological Pin Assignment

- Similar to concentric circle mapping.
- Easier to complete pin assignment.
  - When there is interference from other components and barriers.
  - For nets connected to more than two pins.
- If a net has been assigned to more than two pins, then the pin closest to the center of the primary component is chosen.
- Pins of primary component are mapped onto a circle as before.
- Beginning at the bottom of the circle, and moving clockwise, the pins are assigned to nets.

Nine Zone Method

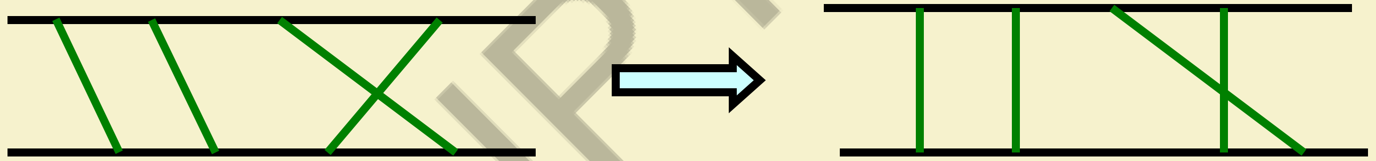
- Based on zones in a Cartesian co-ordinate system.
- The center of the co-ordinate system is located inside a group of interchangeable pins on a component.
  - This component is called *pin class*.
- A net rectangle is defined by each of the nets connected to the pin class.
  - There are nine zones in which this rectangle can be positioned.

The Nine Pin Zones



Channel Pin Assignment

- A significant portion of the chip area is used for channel routing.
  - After the placement phase, the position of terminals on the boundaries of a block are not fixed.
  - They may be moved before routing begins.



- Yang & Wong proposed a dynamic programming formulation to the channel pin assignment problem.

Integrated Approach

- Better understanding of the different stages in physical design automation over the years.
  - Attempts are being made to merge some steps of the design cycle.
  - For example, floorplanning and placement are considered together.
  - Sometimes, placement and routing stages can also be combined together.

END OF LECTURE 10

IIT KHARAGPUR

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

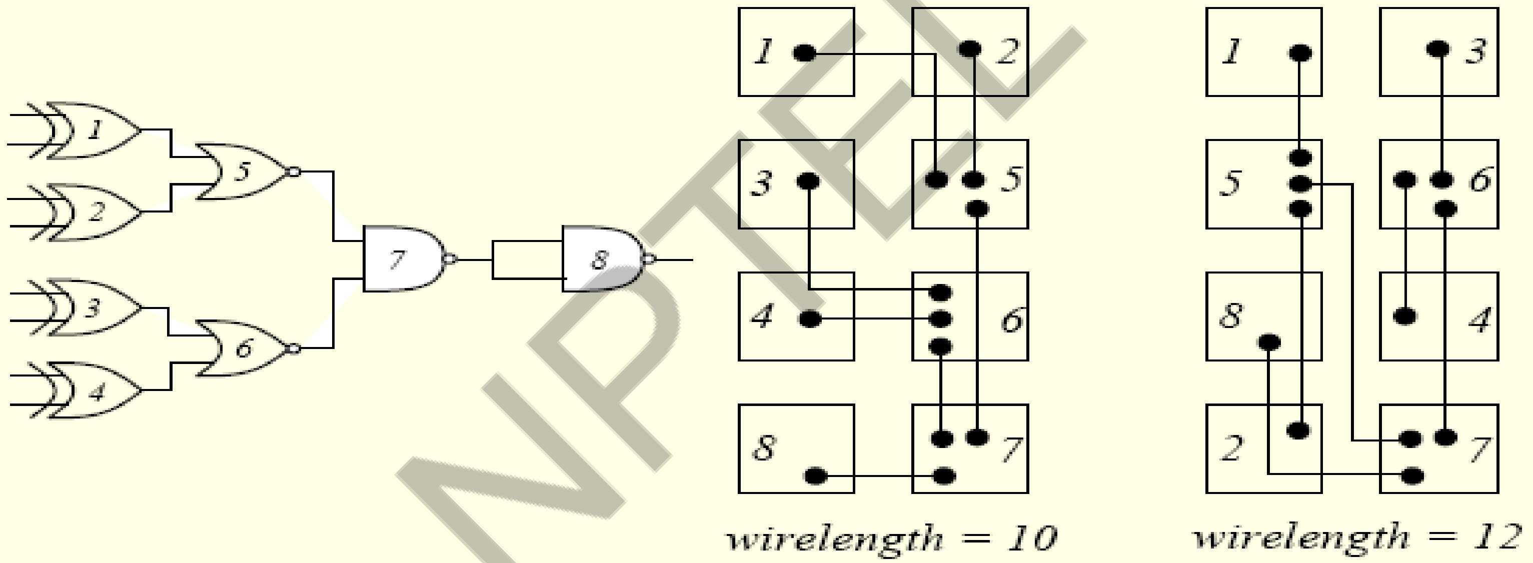
Lecture 11: PLACEMENT (PART 1)

PROF. INDRANIL SENGUPTA
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

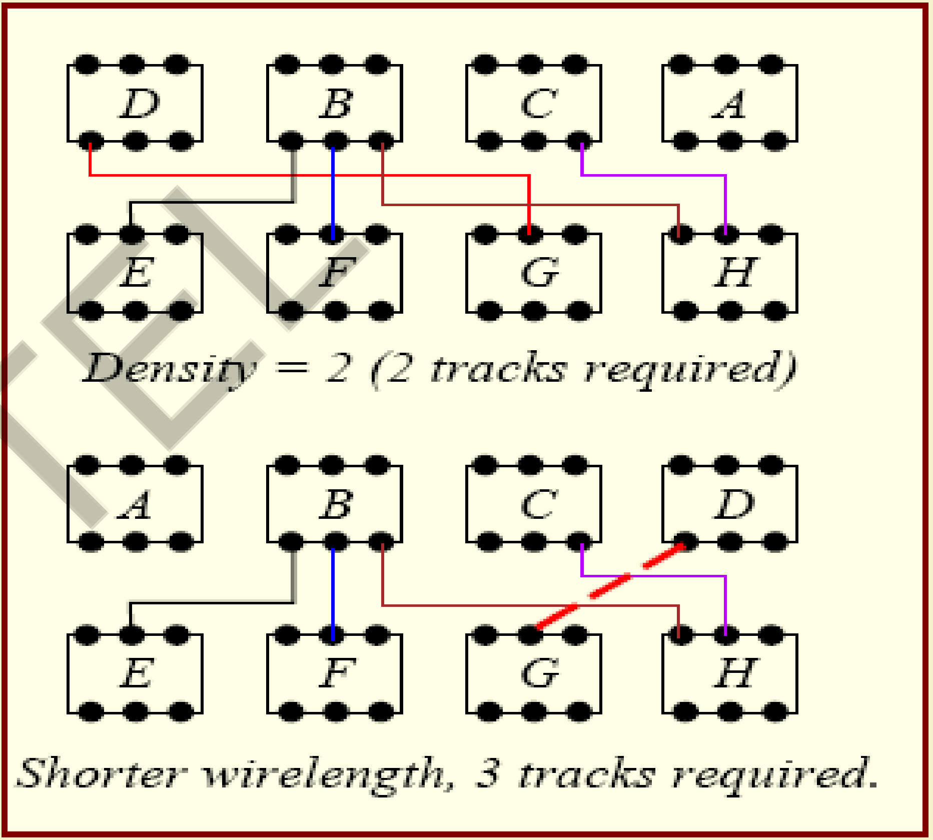
Introduction

- A very important step in physical design cycle.
  - A poor placement requires larger area.
  - Also results in performance degradation.
- It is the process of arranging a set of modules on the layout surface.
  - Each module has fixed shape and fixed terminal locations.
  - A subset of modules may have pre-assigned positions (e.g., I/O pads).

Different Wire Length



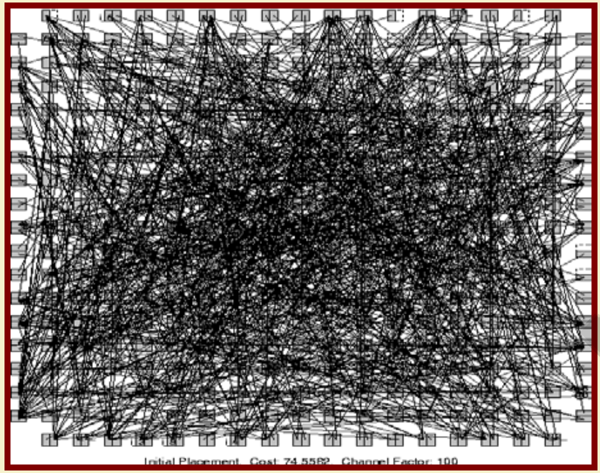
Different Routability/Chip Area


D
B
C
A
E
F
G
H
Density = 2 (2 tracks required)
A
B
C
D
E
F
G
H
Shorter wirelength, 3 tracks required.

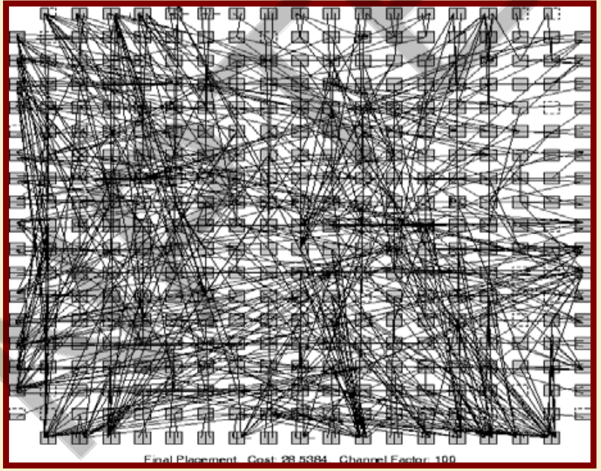
Placement can Make a Difference

- Placement of MCNC enchmark circuit e64 (contains 230 4-LUT) on a FPGA.

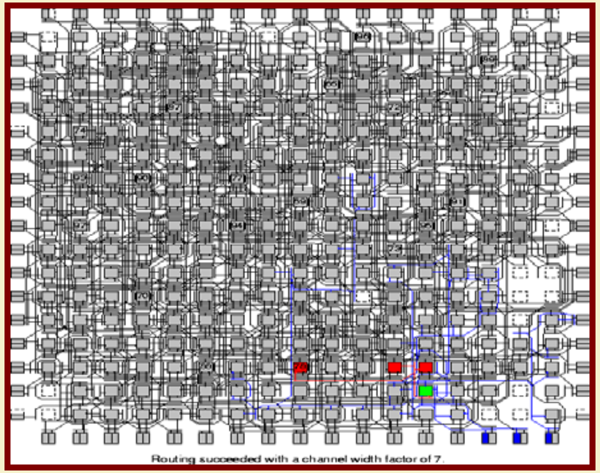
Random Initial Placement



Final Placement



After Detailed Routing



The Placement Problem

- Inputs:
  - A set of modules with (a) well-defined shapes, and (b) fixed locations of pins.
  - A netlist.
- Requirements:
  - Find locations for each module so that no two modules overlap.
  - The placement is routable.
- Objectives:
  - Minimize layout area.
  - Reduce the length of critical nets.
  - Completion of routing.

Placement Problem at Different Levels

1. <u>System-level placement</u>
   - Place all the PCBs together such that
     - Area occupied is minimum
     - Heat dissipation is within limits.
2. <u>Board-level placement</u>
   - All the chips have to be placed on a PCB.
     - Area is fixed
     - All modules of rectangular shape
   - Objective is to: (a) Minimize the number of routing layers, (b) Meet system performance requirements.

3. Chip-level placement

- Normally, floorplanning / placement carried out along with pin assignment.
- Limited number of routing layers (2 to 4).
  - Bad placements may be unroutable.
  - Can be detected only later (during routing).
  - Costly delays in design cycle.
- Minimization of area.

Problem Formulation

- Notations:

$B_1, B_2, \ldots, B_n$	:	modules/blocks to be placed
w_i, h_i	:	width and height of B_i, $1 \leq i \leq n$
$N=\{N_1,N_2,\ldots,N_m\}$	:	set of nets (i.e. the netlist)
$Q=\{Q_1,Q_2,\ldots,Q_k\}$	:	rectangular empty spaces for routing
L_i	:	estimated length of net N_i, $1 \leq i \leq m$

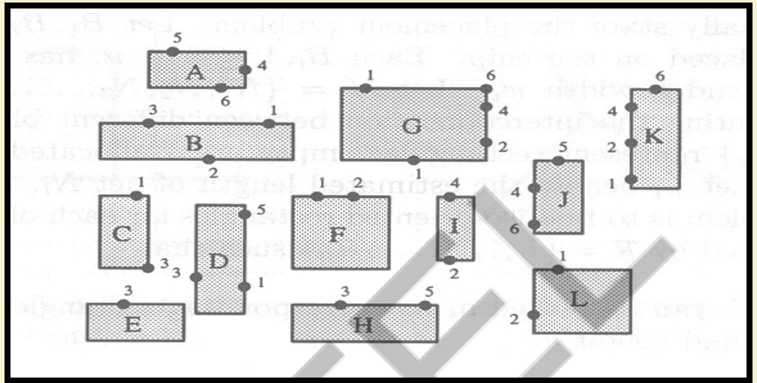
- <u>The problem:</u>

  Find rectangular regions $R=\{R_1,R_2,\ldots R_n\}$ for each of the blocks such that

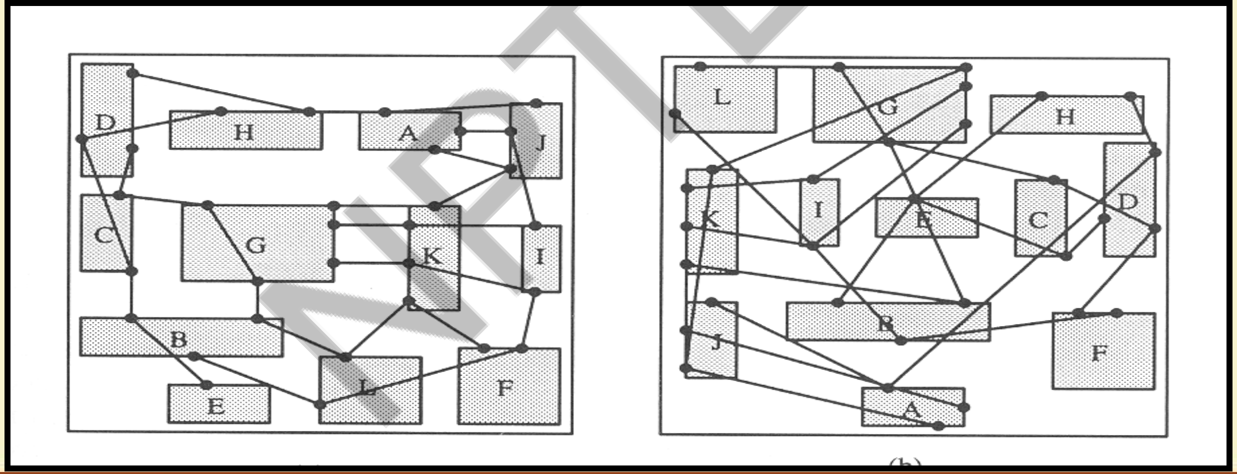
  - Block B_i can be placed in region R_i.
  - No two rectangles overlap, $R_i \cap R_j = \Phi$.
  - Placement is routable (Q is sufficient to route all nets).
  - Total area of rectangle bounding R and Q is minimized.
  - Total wire length ΣL_i is minimized.
  - For high performance circuits, $\max \{L_i \mid i=1,2,\ldots,m\}$ is minimized.

- General problem is NP-complete.
- Algorithms used are heuristic in nature.

Given set of blocks



Good Placement



Bad Placement

Interconnection Topologies

- The actual wiring paths are not known during placement.
  - For making an estimation, a placement algorithm needs to model the topology of the interconnection nets.
    - An interconnection graph structure is used.
    - Vertices are terminals, and edges are interconnections.
- Estimation of wire length is important.

Estimation of Wirelength

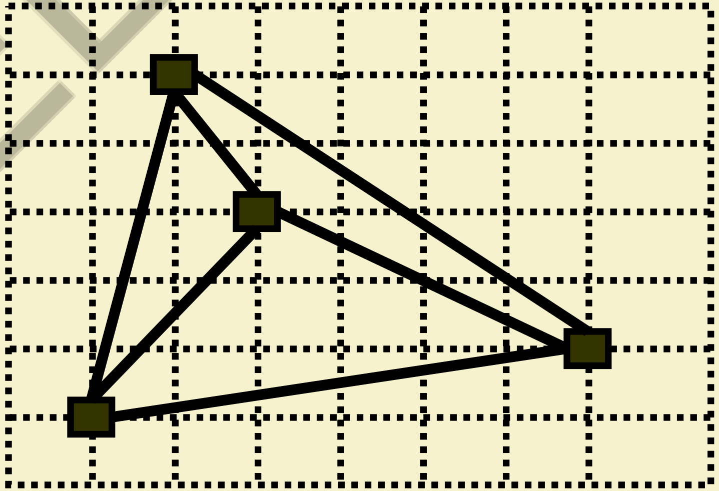
- The speed and quality of estimation has a drastic effect on the performance of placement algorithms.
  - For 2-terminal nets, we can use Manhattan distance as an estimate.
  - If the end co-ordinates are (x_1,y_1) and (x_2,y_2), then the wire length

$$L = |x_1 - x_2| + |y_1 - y_2|$$

- How to estimate length of multi-terminal nets?

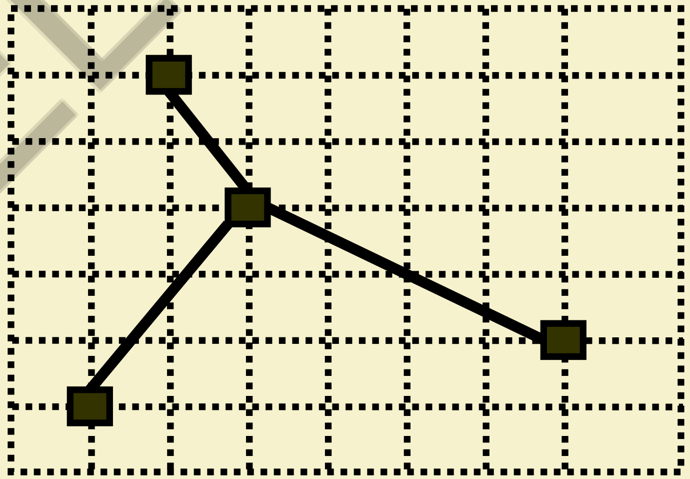
Modeling of Multi-terminal Nets

1. <u>Complete Graph</u>
   - ${}^{n}C_2$ = n(n-1)/2 edges for a n-pin net.
   - A tree has (n-1) edges which is 2/n times the number of edges of the complete graph.
   - Length is estimated as 2/n times the sum of the edge weights.



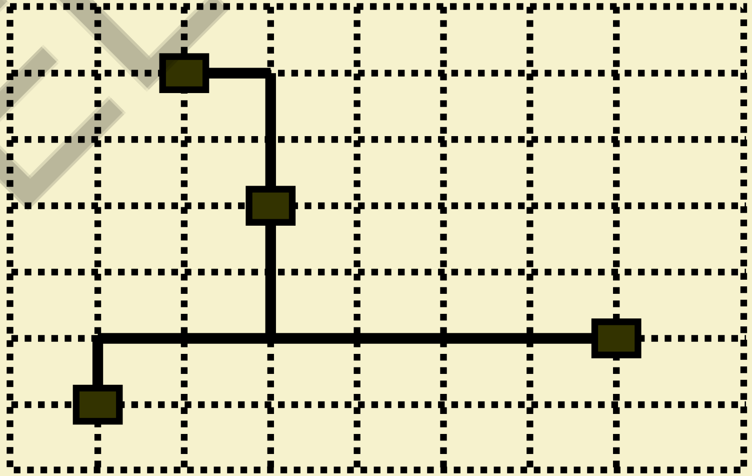
2. <u>Minimum Spanning Tree</u>

- Commonly used structure.
- Branching allowed only at pin locations.
- Easy to compute.



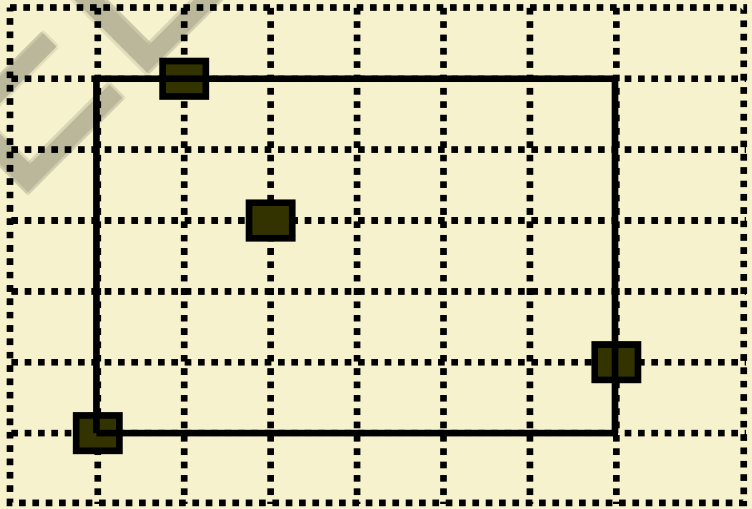
3. <u>Rectangular Steiner Tree</u>

- A Steiner tree is the shortest route for connecting a set of pins.
- A wire can branch from any point along its length.
- Problem of finding Steiner tree is NP-complete.



4. Semi Perimeter

- Efficient and most widely used.
- Finds the smallest bounding rectangle that encloses all the pins to be connected.
- Estimated wire length is half the perimeter of this rectangle.
- Always underestimates the wire length for congested nets.



END OF LECTURE 11

IIT KHARAGPUR

NPTEL ONLINE CERTIFICATION COURSES

NPTEL

Lecture 12: PLACEMENT (PART 2)

PROF. INDRANIL SENGUPTA
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Design Style Specific Issues

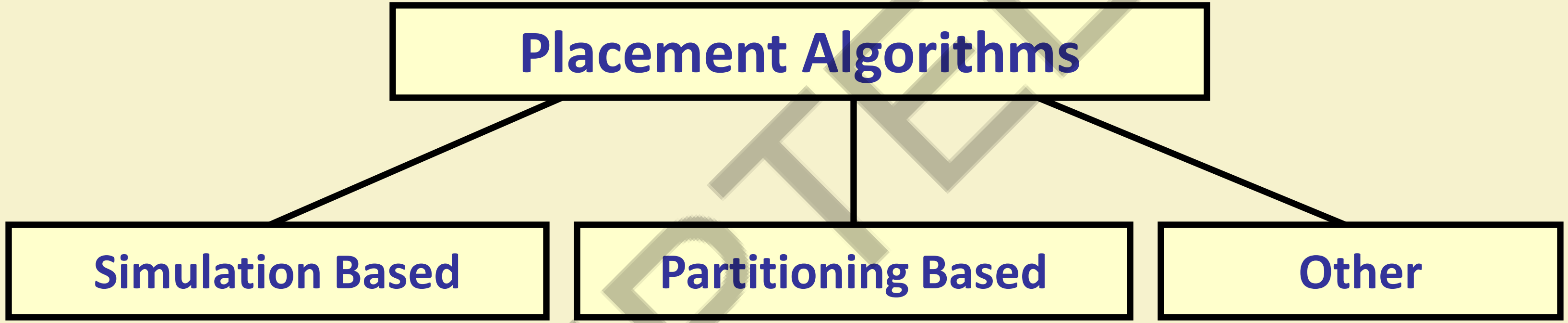
- The main issues in placement can differ depending on the design style used.
  - For instance, in standard cell based design style, the floorplanning and placement problems are the same.
- We discuss the main issues relating to the ASIC design styles:
  - Full custom, standard cell, and gate array.

- <u>Full Custom</u>
  - Placing a number of blocks of various shapes and sizes within a rectangular region.
  - Irregularity of block shapes may lead to unused areas.
  - Both floorplanning and placement steps are required.
  - May require iterations, where the layout may be modified at each step.

- <u>Standard Cell</u>
  - The problem of floorplanning and placement are the same in this design style.
  - Minimization of the layout area means:
    - Minimize sum of channel heights.
    - Minimize width of the widest row.
    - All rows should have equal width.
  - Over-the-cell routing leads to almost *channel-less* standard cell designs.

- <u>Gate Arrays</u>
  - The problem of partitioning, floorplanning and placement are the same in this design style.
  - For FPGAs, the partitioned sub-circuit may be a complex netlist.
    - Map the netlist to one or more basic blocks or LUTs (placement).

Classification of Placement Algorithms



Simulated Annealing	Breuer's Algorithm	Cluster Growth
Simulated Evolution	Terminal Propagation	Force Directed
Force Directed		

Simulated Annealing

- Simulation of the annealing process in metals or glass.
  - Avoids getting trapped in local minima.
  - Starts with an initial placement.
  - Incremental improvements by exchanging blocks, displacing a block, etc.
  - Moves which decrease cost are always accepted.
  - Moves which increase cost are accepted with a probability that decreases with the number of iterations.
- Timberwolf is one of the most successful placement algorithms based on simulated annealing.

Simulated Annealing Algorithm

```
Algorithm  SA_Placement
begin
    T = initial_temperature;
    P = initial_placement;
    while  ( T > final_temperature)  do
        while  (no_of_trials_at_each_temp not yet completed) do
            new_P = PERTURB (P);
            $\Delta C$ = COST (new_P) – COST (P);
            if  ($\Delta C < 0$)  then
                P = new_P;
            else if  (random(0,1) > exp($\Delta C/T$))  then
                    P = new_P;
        T = SCHEDULE (T);        /** Decrease temperature **/
end
```

TimberWolf

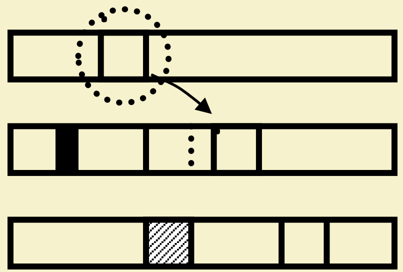
- One of the most successful placement algorithms.
  - Developed by Sechen and Sangiovanni-Vincentelli.
- Parameters used:
  - Initial_temperature = 4,000,000
  - Final_temperature = 0.1
  - SCHEDULE(T) = α(T) x T
    - α(T) specifies the cooling rate which depends on the current temperature.
    - α(T) is 0.8 when the cooling process just starts.
    - α(T) is 0.95 in the medium range of temperature.
    - α(T) is 0.8 again when temperature is low.

The PERTURB Function

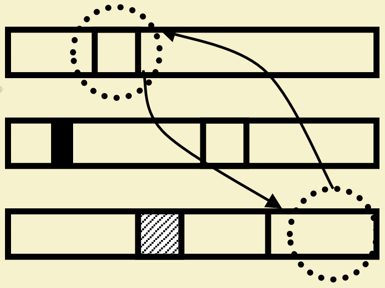
- New configuration is generated by making a weighted random selection from one of the following moves:
  - **M1**. The displacement of a block to a new location.
  - **M2**. The interchange of locations between two blocks.
  - **M3**. An orientation change for a block.
    - – Mirror image of the block's x-coordinate.
    - – Used only when a new configuration generated using alternative M1 is rejected.

Illustration of the Moves

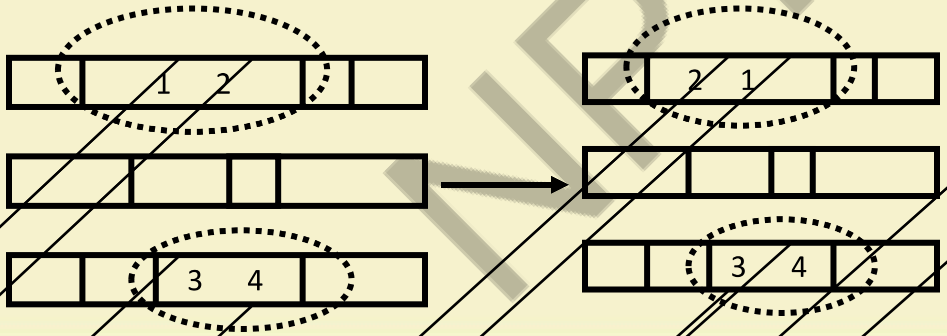
M1

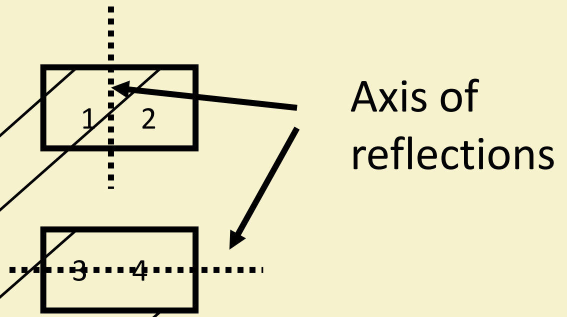


M2



M3





Move Selection

- Timberwolf first tries to select a move between M1 and M2.

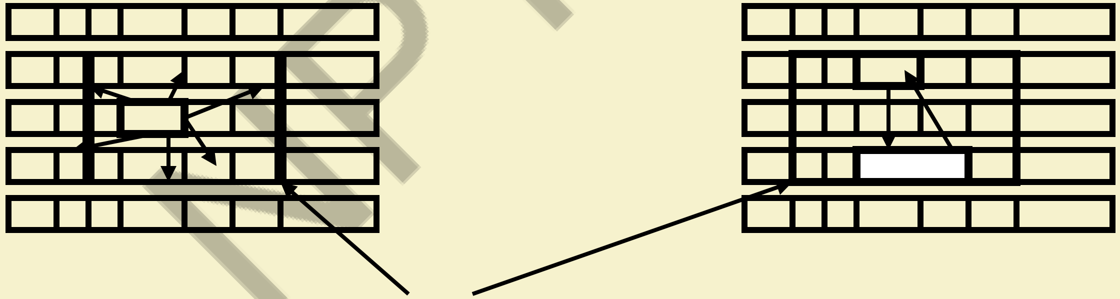
  Prob(M1) = 4/5

  Prob(M2) = 1/5
- If a move of type M1 is chosen (for certain module) and it is rejected, then a move of type M3 (for the same module) will be chosen with probability 1/10.
- Restriction on:
  - How far a module can be displaced
  - What pairs of modules can be interchanged

Move Restriction

Range Limiter:

- At the beginning, R is very large, big enough to contain the whole chip.
- Window size shrinks slowly as the temperature decreases. In fact, height and width of R $\propto \log(T)$.
- Stage 2 begins when window size are so small that no inter-row modules interchanges are possible.



The COST Function

- The cost of a solution is computed as:

  COST = cost1 + cost2 + cost3

  where cost1 : weighted sum of estimated length of all nets

  cost2 : penalty cost for overlapping

  cost3 : penalty cost for uneven length among standard cell rows.

  – Overlap is not allowed in placement.
  – Computationally complex to remove all overlaps.
  – More efficient to allow overlaps during intermediate placements.
    - Cost function (cost2) penalizes the overlapping.

Summary

- Timberwolf is one of the very successful placement tools.
- Gives good placement for standard cell based designs.

END OF LECTURE 12

IIT KHARAGPUR

NPTEL ONLINE
CERTIFICATION COURSES

NPTEL

Lecture 13: PLACEMENT (PART 3)

PROF. INDRANIL SENGUPTA
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Force Directed Placement

- Explores the similarity between placement problem and classical mechanics problem of a system of bodies attached to springs.
- The blocks connected to each other by nets are supposed to exert attractive forces on each other.
  - Magnitude of this force is directly proportional to the distance between the blocks.
    - Analogous to Hooke's law in mechanics.
  - Final configuration is one in which the system achieves equilibrium.

- A cell i connected to several cells j experiences a total force

$$F_i = \Sigma_j (w_{ij} * d_{ij})$$

where w_{ij} is the weight of connection between i and j

d_{ij} is the distance between i and j.

- If the cell i is free to move, it would do so in the direction of force F_i until the resultant force on it is zero.
- When all cells move to their *zero-force target locations*, the total wire length is minimized.

- For cell i, if (x_i^0, y_i^0) represents the zero-force target location, by equating the x- and y-components of the force to zero, we get

$$\sum_j w_{ij} \cdot (x_j^\circ - x_i^\circ) = 0$$

$$\sum_j w_{ij} \cdot (y_j^\circ - y_i^\circ) = 0$$

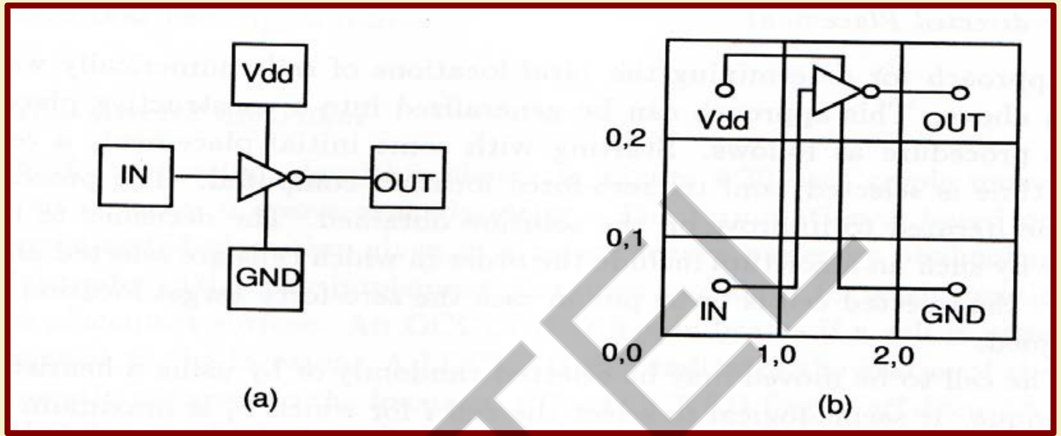
- Solving for x_i^0 and y_i^0, we get

$$x_i^\circ = \frac{\sum_j w_{ij} \cdot x_j}{\sum_j w_{ij}}$$

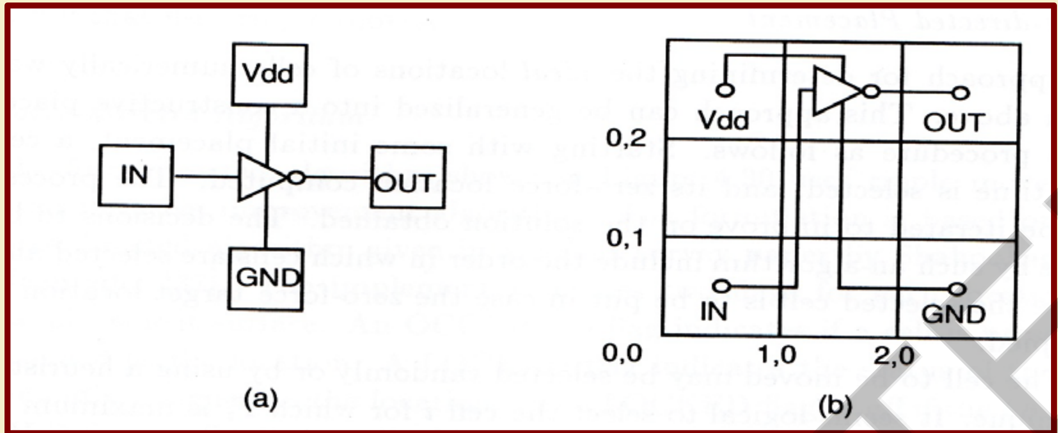
$$y_i^\circ = \frac{\sum_j w_{ij} \cdot y_j}{\sum_j w_{ij}}$$

- Care should be taken to avoid assigning more than one cell to the same location.

Example

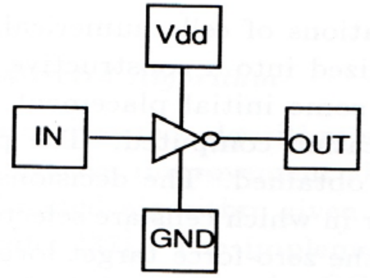


- A circuit with one gate and four I/O pads.
- The four pads are to be placed on the four corners of a 3x3 grid.
- The weights of the wires connected to the gate are: w_{vdd}=8, w_{out}=10, w_{in}=3, and w_{gnd}=3.
- Find the zero-force target location of the gate inside the grid.

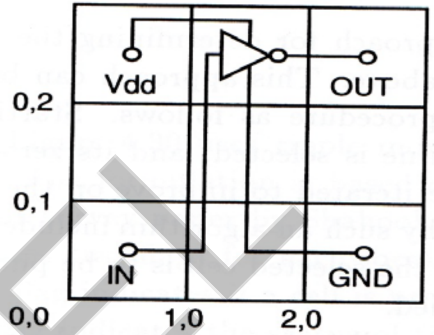


$$x_i^o = \frac{\sum_j w_{ij} \cdot x_j}{\sum_j w_{ij}} = \frac{w_{vdd} \cdot x_{vdd} + w_{out} \cdot x_{out} + w_{in} \cdot x_{in} + w_{gnd} \cdot x_{gnd}}{w_{vdd} + w_{out} + w_{in} + w_{gnd}}$$

$$= \frac{8 \times 0 + 10 \times 2 + 3 \times 0 + 3 \times 2}{8 + 10 + 3 + 3} = \frac{26}{24} = 1.083$$



(a)

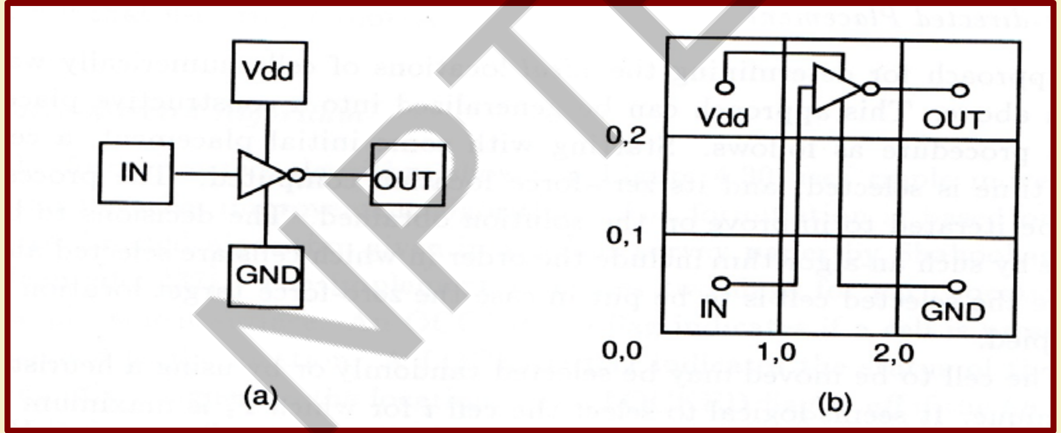


(b)

$$y_i^\circ = \frac{\sum_j w_{ij} \cdot y_j}{\sum_j w_{ij}} = \frac{w_{vdd} \cdot y_{vdd} + w_{out} \cdot y_{out} + w_{in} \cdot y_{in} + w_{gnd} \cdot y_{gnd}}{w_{vdd} + w_{out} + w_{in} + w_{gnd}}$$

$$= \frac{8 \times 2 + 10 \times 2 + 3 \times 0 + 3 \times 0}{8 + 10 + 3 + 3} = \frac{36}{24} = 1.50$$

- The zero-force location for the gate is (1.083, 1.50) that can be approximated to the grid location (1,2).



Force Directed Approach for Constructive Placement

- The basic approach can be generalized for constructive placement.
  - Starting with some initial placement, one module is selected at a time, and its zero-force location F_i computed.
  - The process can be iterated to improve upon the solution obtained.
  - The order of the cells can be random or driven by some heuristic.
    - Select the cell for which F_i is maximum.

- If the zero-force location is occupied by another cell q, then several options to place the cell p under consideration exist.
  1. Move p to a location close to q.
  2. Evaluate the change in cost if p is swapped with q. If the cost decreases, only then is the swap made.
  3. ***Ripple move***: The cell p is placed in the computed location, and a new zero-force location is computed for the displaced cell q. The procedure is continued until all the cells are placed.
  4. ***Chain move***: The cell p is placed in the computed location, and the cell q is moved to an adjacent location. If the adjacent location is occupied by a cell r, then r is moved to its adjacent location, and so on, until a free location is finally found.

END OF LECTURE 13

IIT KHARAGPUR

NPTEL ONLINE
CERTIFICATION COURSES

NPTEL

Lecture 14: PLACEMENT (PART 4)

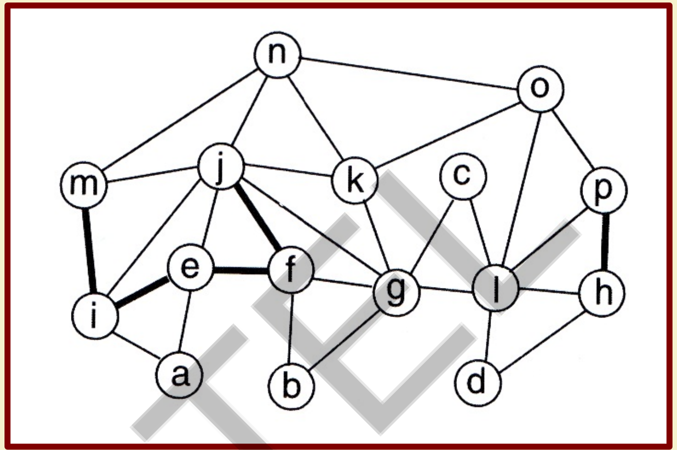
PROF. INDRANIL SENGUPTA
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Breuer's Algorithm

- Partitioning technique used to generate placement.
- The given circuit is repeatedly partitioned into two sub-circuits.
  - At each level of partitioning, the available layout area is partitioned into horizontal and vertical subsections alternately.
  - Each of the sub-circuits is assigned to a subsection.
  - Process continues till each sub-circuit consists of a single gate, and has a unique place on the layout area.

- Several cut-oriented sequences have been proposed.
  - Cutsize is minimized during partitioning.
- We shall illustrate two alternate cut sequences proposed by Breuer:
  1. Quadrature mincut placement
  2. Recursive bipartitioning mincut placement

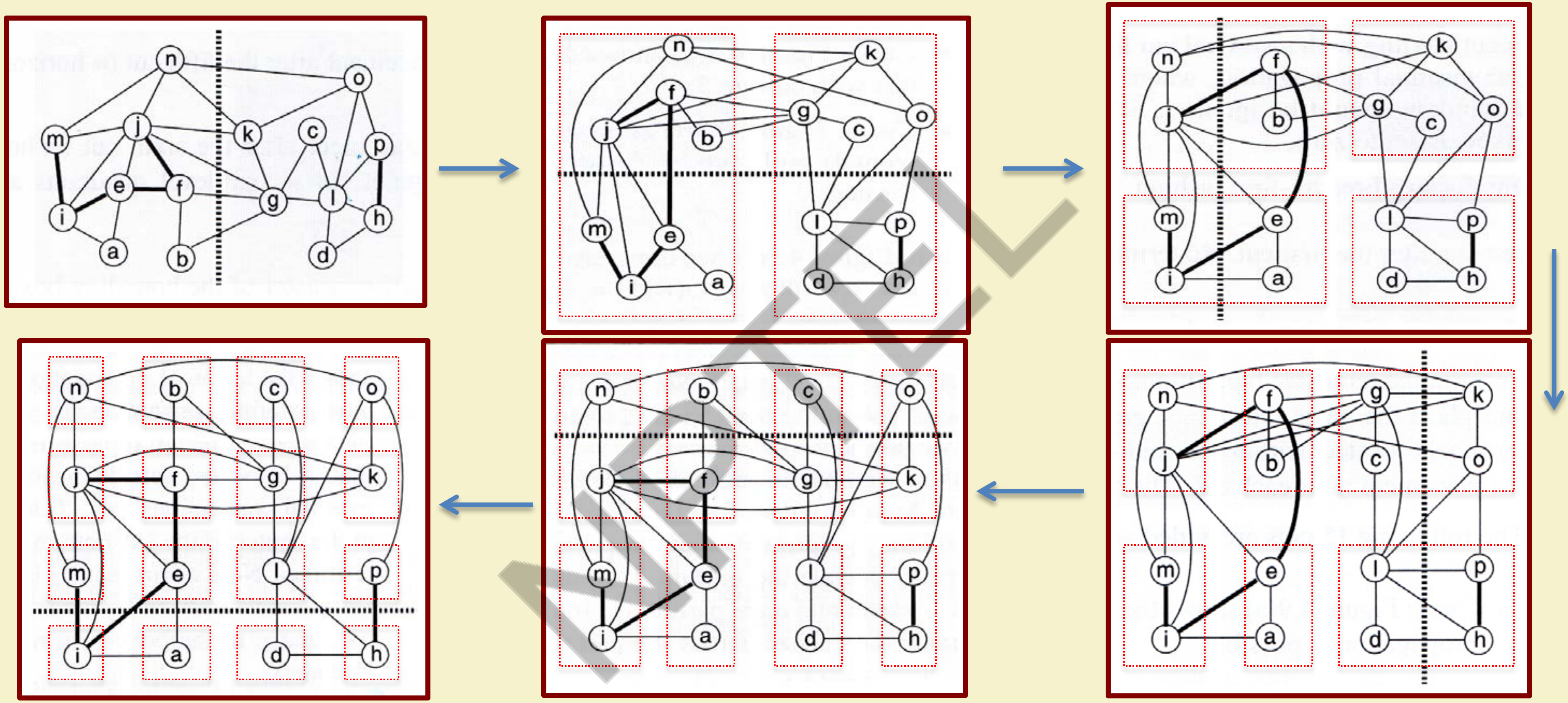
An Example Block Level Netlist



- The thick edges have a weight of 1, and the thin edges have a weight of 0.5.

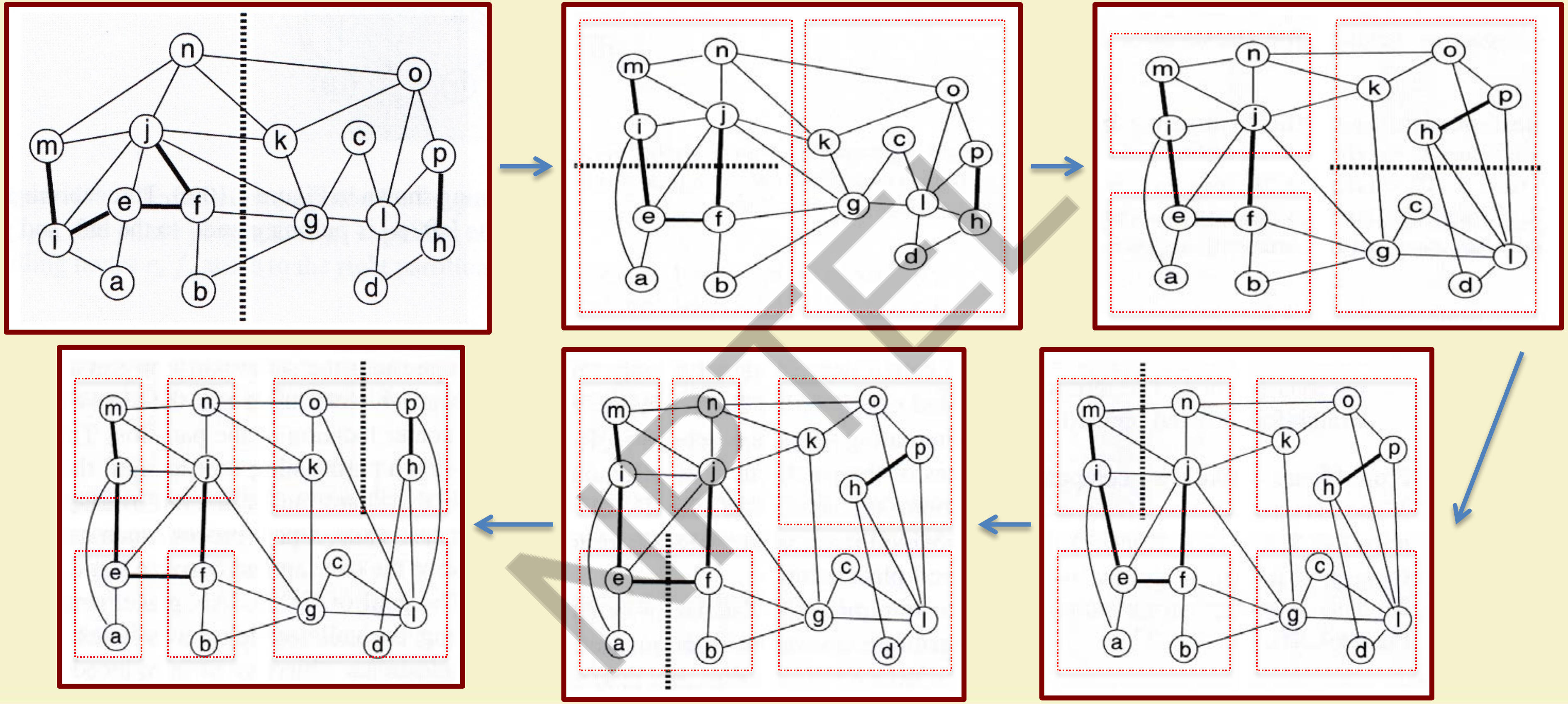
Quadrature Mincut Placement

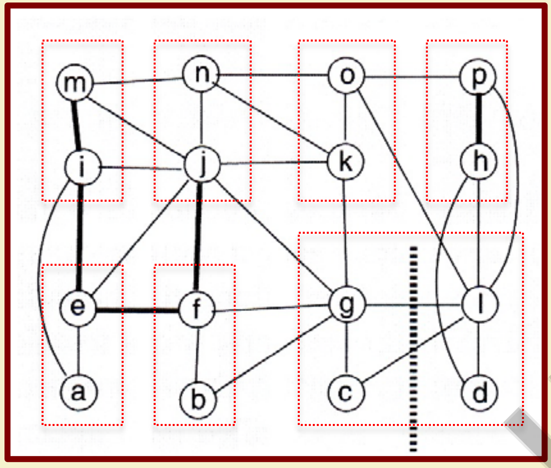
- The layout is divided into 4 units with two cutlines, one vertical and one horizontal, both passing through the center.
- The above division procedure is then recursively applied to each quarter of the layout cut until the entire layout is divided into slots of desired size.

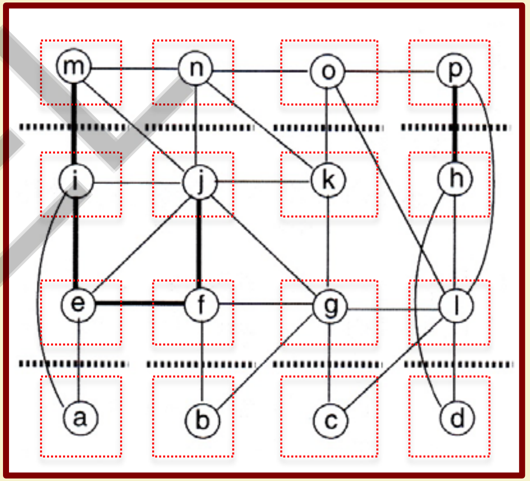


Recursive Bipartitioning Mincut Placement

- The layout is repeatedly divided recursively using horizontal and vertical cutlines as illustrated.


m
n
o
j
k
c
p
e
f
g
l
h
i
a
b
d

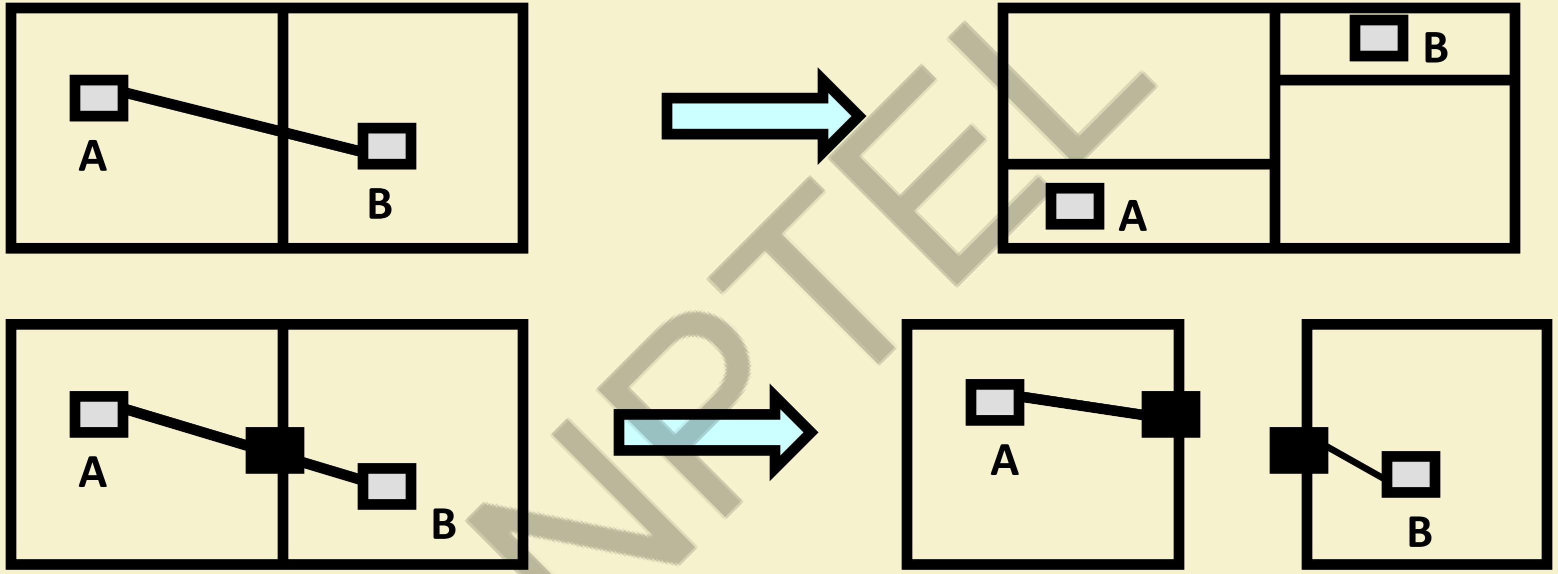

m
n
o
p
i
j
k
h
e
f
g
l
a
b
c
d


m
n
o
p
i
j
k
h
e
f
g
l
a
b
c
d

Terminal Propagation Algorithm

- Partitioning algorithms merely reduce net cut.
- Direct use of partitioning algorithms would increase net length.
  - Also increases congestion in the channels.
- To prevent this, terminal propagation is used.
  - When a net connecting two terminals is cut, a dummy terminal is propagated to the nearest pin on the boundary.
  - When this dummy terminal is generated, the partitioning algorithm will not assign the two terminals in each partition into different partitions, as this would not result in a minimum cut.

Illustration :: Terminal Propagation



■ :: Dummy terminal ▢ :: Terminal

Cluster Growth

- In this constructive placement algorithm, bottom-up approach is used.
- Blocks are placed sequentially in a partially completed layout.
  - The first block (seed) is usually placed by the user.
  - Other blocks are selected and placed one by one.
- Selection of blocks is usually based on connectivity with placed blocks.

Contd.

- Layouts produced are not usually good.
  - Does not take into account the interconnections and other circuit features.
- Useful for generating initial placements.
  - For iterative placement algorithms.

```
Algorithm  Cluster_Growth
begin
      B = set of blocks to be placed;
      Select a seed block S from B;
      Place S in the layout;
      B = B – S;
      while  (B $\neq \phi$) do
          begin
              Select a block X from B;
              Place X in the layout;
              B = B – X;
          end;
end
```

Performance Driven Placement

- The delay at chip level plays an important role in determining the performance of the chip.
  - Depends on interconnecting wires.
- As the blocks in a circuit becomes smaller and smaller:
  - The size of the chip decreases.
  - Interconnection delay becomes a major issue in high-performance circuits.
- Placement algorithms for high-performance chips:
  - Allow routing of nets within timing constraints.

- Two major categories of algorithms:
  1. Net-based approach
     - Try to route the nets to meet the timing constraints on the individual nets instead of considering paths.
     - The timing requirement for each net has to be decided by the algorithm.
     - Usually a pre-timing analysis generates the bounds on the net-lengths which must be satisfied during placement.
  2. Path-based approach
     - Critical paths in the circuit are considered.
     - Try to place the blocks in a manner that the path length is within the timing constraint.

END OF LECTURE 14