Elective II: VLSI Design

Code: CISM 402

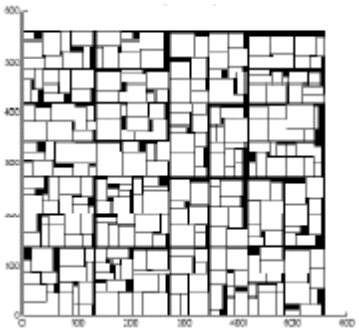
Pritha Banerjee

Courtsey for slides: Debasis Mitra, NIT Durgapur

Floorplanning

- Books:
  - Chapter 5 of Naveed A. Sherwani, *Algorithms for VLSI Physical Design Automation*, Kluwer Academic Publishers
  - Chapter 2 (2.2) of M. Sarafzadeh and C. K. Wong, *An introduction to VLSI Physical Design*, The McGraw Hill Companies, Inc.

Physical Design Flow



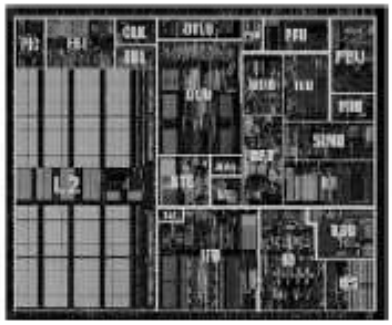
Circuit Partitioning

Floorplanning & Placement

Routing

Layout Compaction

Extraction and Verification



Partitioning is over! What Next?

- Partitioning leads to

Blocks with well-defined **areas and shapes** (rigid/hard blocks).

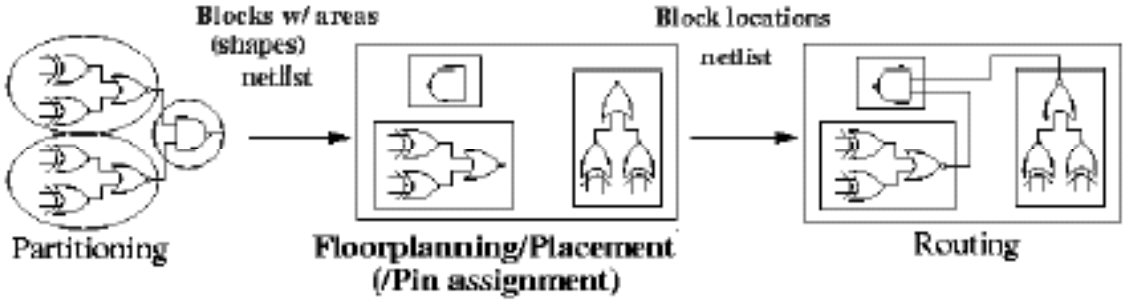
Blocks with approximated areas and no particular shapes (flexible/soft blocks).

A **netlist** specifying connections between the blocks.

- Objectives

Find **locations** for all blocks.

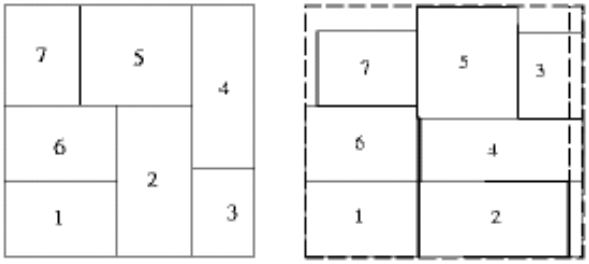
Consider shapes of soft block and pin locations of all the blocks.



Floorplanning

To plan the *positions* and *shapes* of the modules at the beginning of the design cycle to optimize the circuit performance:

- Chip area
- Total wirelength
- Critical path delay
- Routability
- Others (eg, noise, heat dissipation etc.)



An optimal floorplan, in terms of area

A non–optimal floorplan

Floorplanning Vs Placement

- Both determines block positions to optimize the circuit performance
- Floorplanning:
  - Details like shapes of blocks, I/O pin positions, etc. are not yet fixed
  - Blocks with flexible shape are called soft blocks
- Placement:
  - Details like module shapes and I/O pin positions are fixed
  - blocks with no flexibility in shape are called hard blocks

Floorplanning Problem

- Input:
  - *n* Blocks with areas $A_1, \ldots, A_n$
  - 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 $r_i \leq h_i/w_i \leq s_i$
- Objective:
  - To optimize the circuit performance

    A commonly used objective function: a weighted sum of area and wirelength (cost = $\alpha A + \beta L$)

    where A is the total area of the packing, L is the total wirelength, and α and β are constants

Floorplanning

Aspect ratio (h_i/w_i)

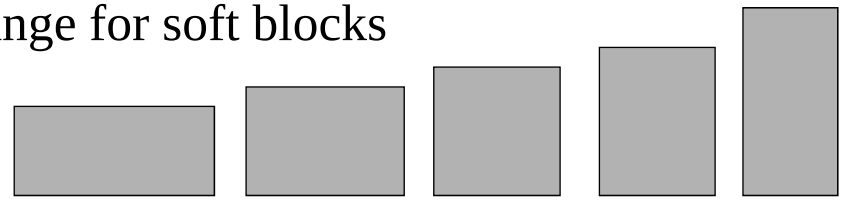
- If there is no bound on the aspect ratios, can we pack everything tightly?
  - Yes

But we don't want to layout blocks as long strips, so we require a bound ($r_i \leq h_i/w_i \leq s_i$)for each i

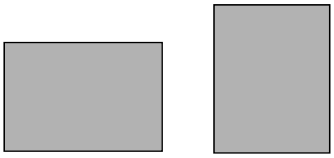
Floorplanning

Aspect ratio (h_i/w_i)

- Several shapes are allowed keeping AR within the range for soft blocks



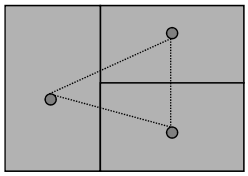
- For hard blocks, the orientations can be changed

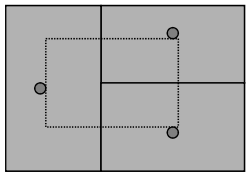


Floorplanning

Wirelength estimation:

- Exact wirelength of each net is not known until routing is done
- In floorplanning, even pin positions are not known yet.
- Some possible wirelength estimations:
  - Center-to-center estimation
  - Half-perimeter estimation

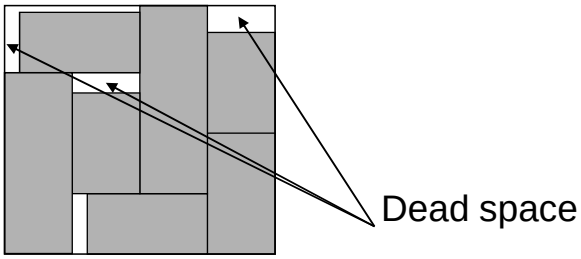




Floorplanning

Dead space:

- The space that is wasted:



- Minimizing area is the same as minimizing dead space
- Dead space percentage is computed as

$$(A - \Sigma_i D_i) / A \times 100\%$$

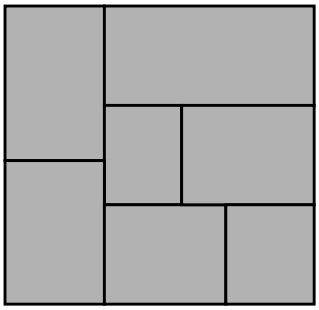
Floorplanning Algorithms

- Floorplanning problem belongs to the class of NP complete problems- thus heuristics
- Floorplanning requires:
  - Floorplanning topology generation: determine the relative location of modules/ blocks
  - Floorplan sizing : determining the exact shape (w, h) of each module
- such that area of floorplan is minimal and total wirelength is minimal
- Need to devise efficient representation of floorplan

Slicing and Non-Slicing Floorplan

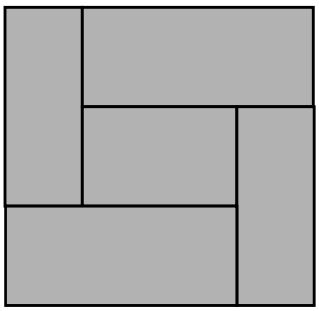
Slicing Floorplan:

One that can be obtained by repetitively subdividing (slicing) rectangles horizontally or vertically



Non-Slicing Floorplan:

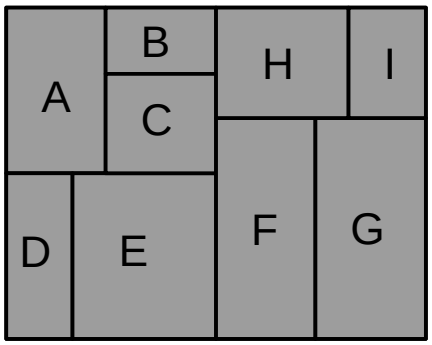
One that may not be obtained by repetitively subdividing alone



Otten (LSSS-82) pointed out that slicing floorplans are much easier to handle

Data structure :Slicing Floorplan

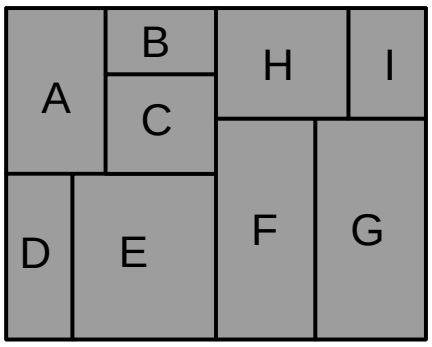
Example:



Data structure: Slicing Floorplan

Slicing Tree:

- A binary tree in which each leaf represents a partition (module) and each internal node represents a cut
- Slicing tree is of order 2

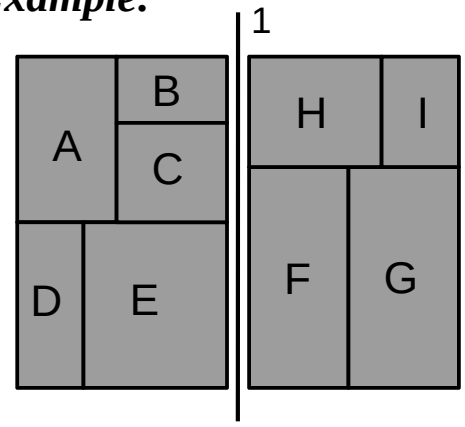


Slicing and Non-Slicing Floorplan

Slicing Tree:

A binary tree in which each leaf represents a partition and each internal node represents a cut

Example:

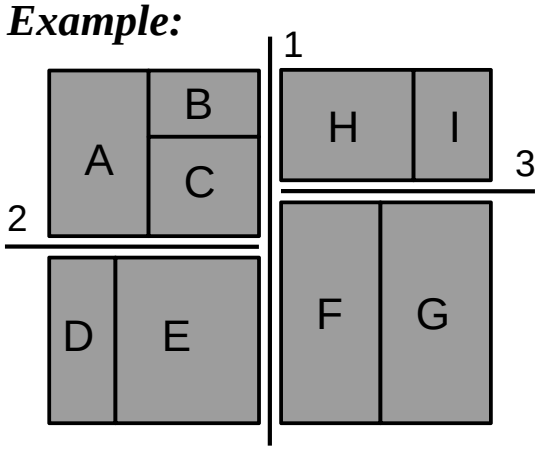


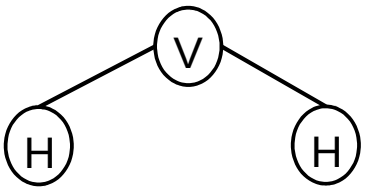
Slicing and Non-Slicing Floorplan

Slicing Tree:

A binary tree in which each leaf represents a partition and each internal node represents a cut

Example:



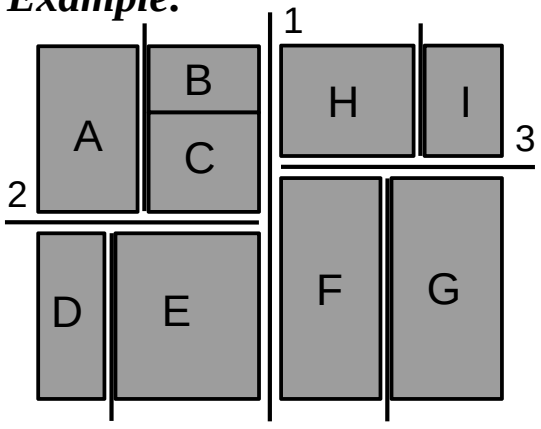


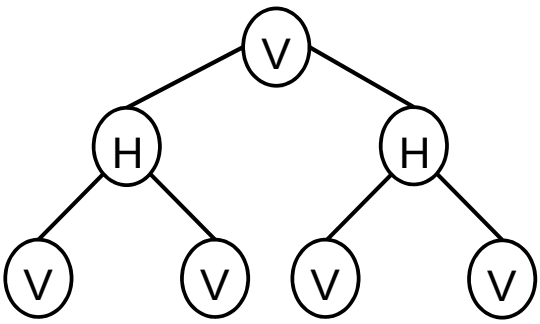
Slicing and Non-Slicing Floorplan

Slicing Tree:

A binary tree in which each leaf represents a partition and each internal node represents a cut

Example:



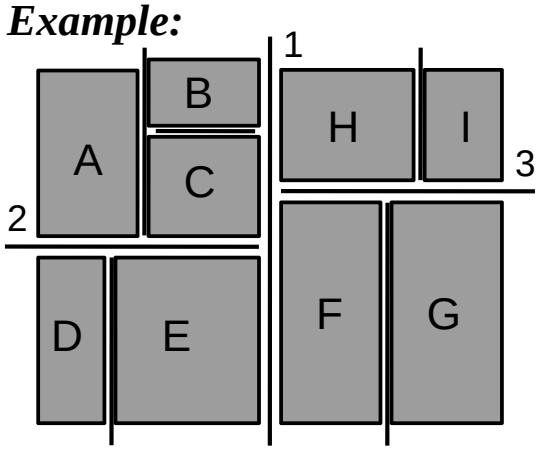


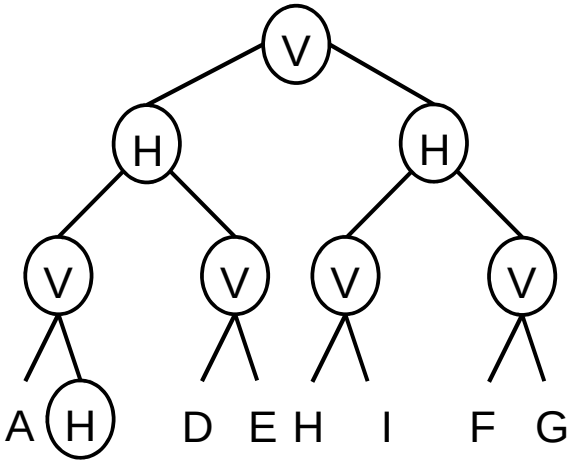
Slicing and Non-Slicing Floorplan

Slicing Tree:

A binary tree in which each leaf represents a partition and each internal node represents a cut

Example:



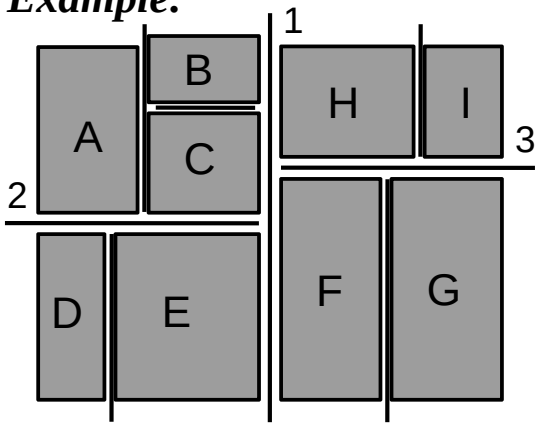


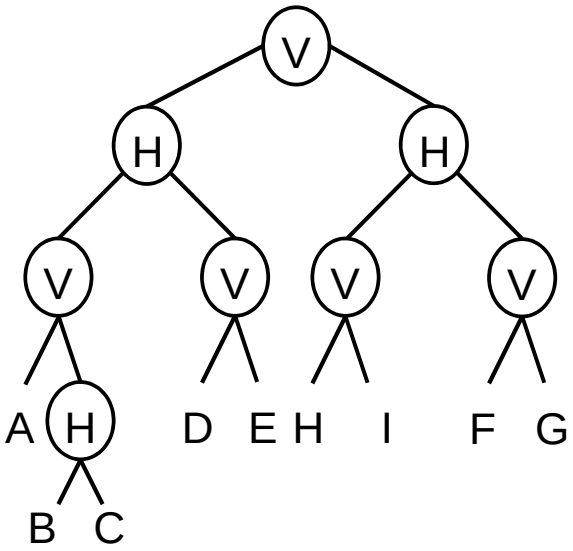
Slicing and Non-Slicing Floorplan

Slicing Tree:

A binary tree in which each leaf represents a partition and each internal node represents a cut

Example:

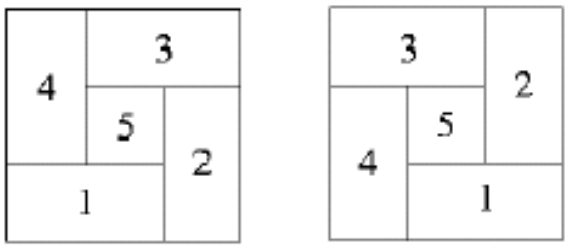




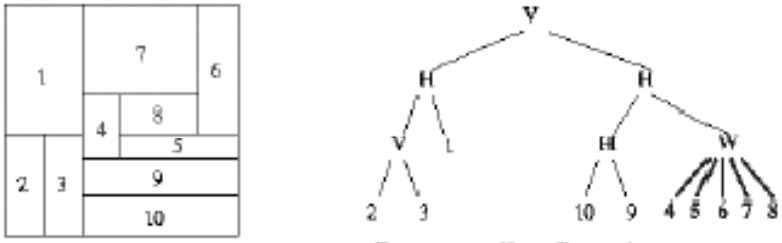
Polish Expression: postorder trversal: ABC-|DE|-HI|FG|-|

Floorplan order: Non-Slicing

- **Wheel:** The smallest non-slicing floorplans (Wang and Wong, TCAD, Aug. 92).
- **Order of a floorplan:** a slicing floorplan is of order 2.
- **Floorplan tree:** A tree representing the hierarchy of partitioning.



The two possible wheels.



A floorplan of order 5

Corresponding floorplan tree

1

Floorplanning approach

- Rectangular dual graph approach to floorplanning
- Hierarchical approach to floorplanning
  - Bottom-up
  - Top-down
- Simulated annealing based approach to floorplanning

Rectangular dual graph approach

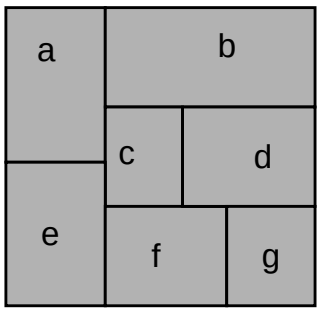
Block Adjacency Graph:

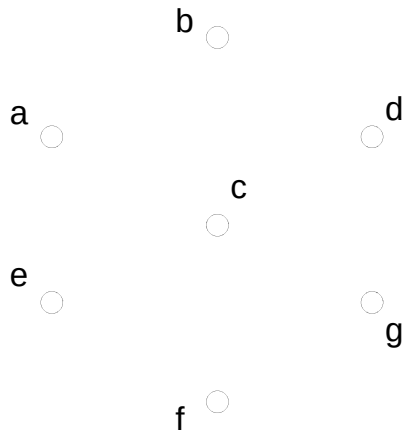
- A floorplan is modeled by an undirected graph:
  - Vertex: block
  - Edge: (v_i, v_j) is an edge if the block corresponding to v_i is adjacent to the block corresponding to v_j

Graph Representation of Floorplan

Block Adjacency Graph:

Example:

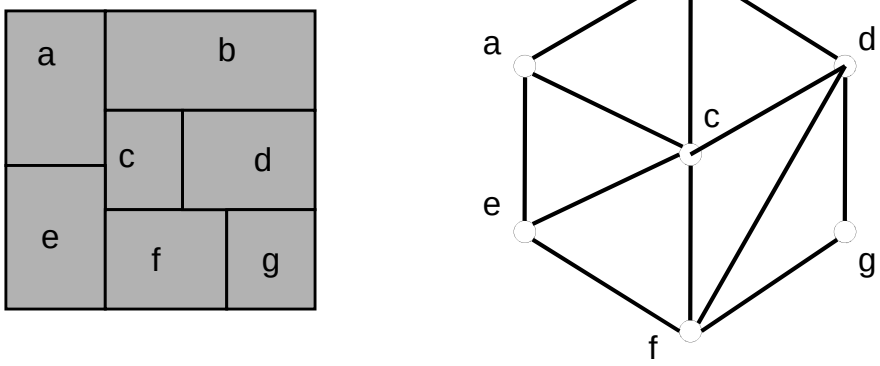




Graph Representation of Floorplan

Block Adjacency Graph:

Example:



Every T junction represents a triangulated face of adjacency graph

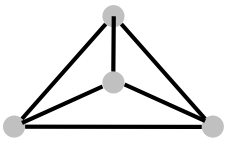
rectangular floorplan with T junction : planar triangulated graph

Graph Representation of Floorplan

Block Adjacency Graph:

Properties:

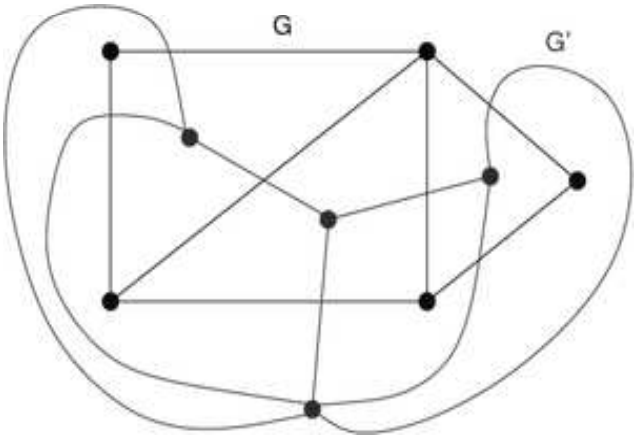
- The graph is planar and rectangular dualizable
- Every face (closed contour within which there is nothing) is a triangle
- There is no complex triangle (cycle (a,b,c) of length 3 is not a face)
  - No valid floorplan exist for this graph



- All interior vertices have degree ≥ 4
  - An interior block must have atleast 4 adjacencies
- There are atmost 4 vertices of degree 2
  - Degree 2 vertices represent corner blocks

Geometric dual

- **Geometric dual** (or simply dual) of a planar graph *G* is a graph which has a vertex for each face of *G*, and an edge for each edge in *G* joining two neighboring faces, for a certain embedding of *G*
- The term "dual" is used because this property is symmetric
  - i.e., if *H* is a dual of *G*, then *G* is a dual of *H* (if *G* is connected)



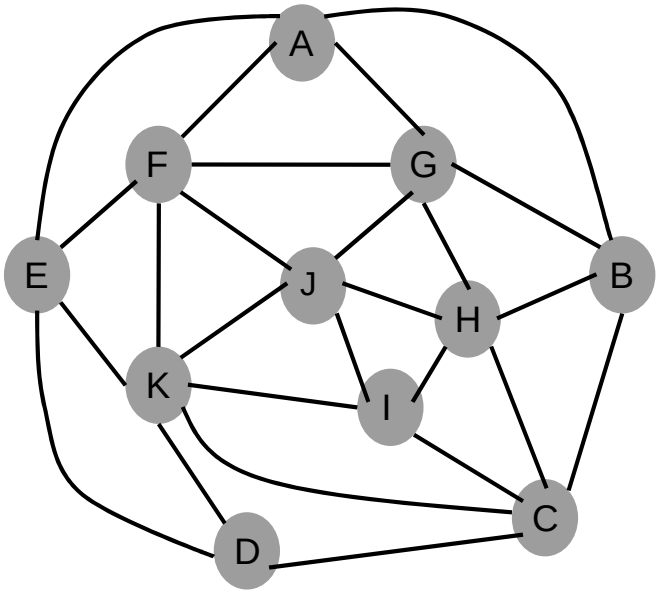
Floorplanning by Rectangular Dualization

Idea:

- Generates topology, i.e., determines dissection pattern based on neighborhood constraints
  - Sizing has to be taken care of separately
- Input: Neighborhood graph obtained from partitioning phase
- Steps:
  - Make sure that
    - The graph is planar
    - Every interior faces are triangles
    - There is no complex triangle
  - Construct the **dual**
  - Redraw it in a rectilinear fashion

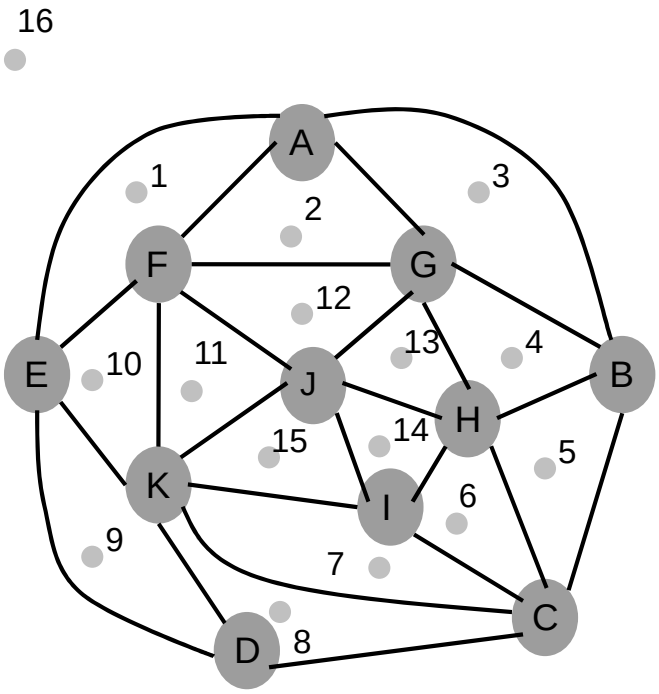
Floorplanning by Rectangular Dualization

Example:



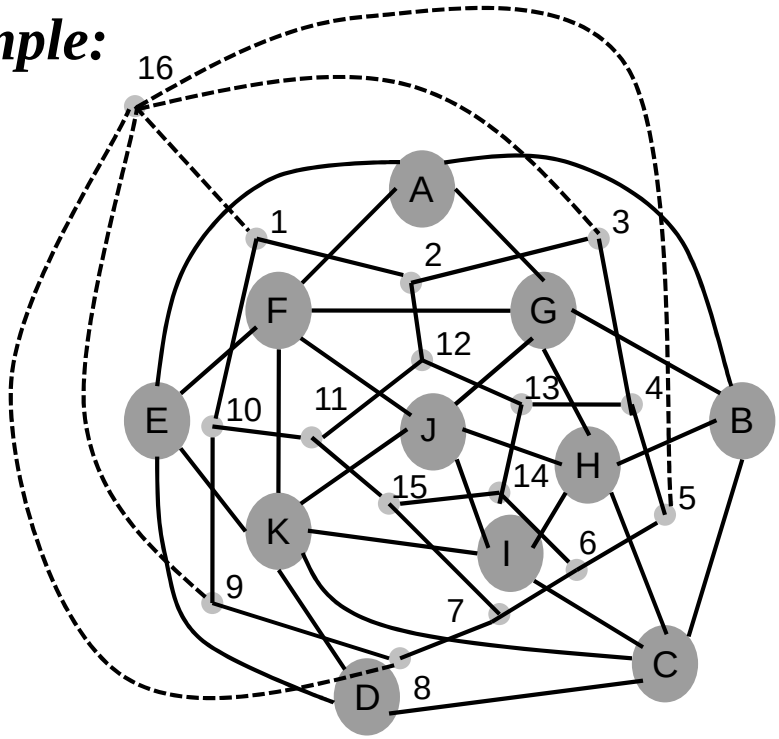
Floorplanning by Rectangular Dualization

Example:



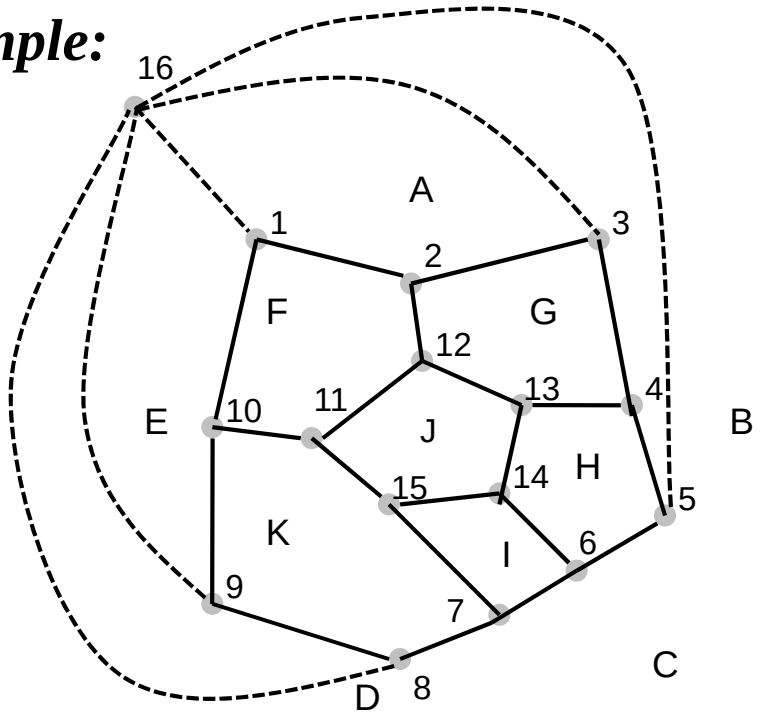
Floorplanning by Rectangular Dualization

Example:



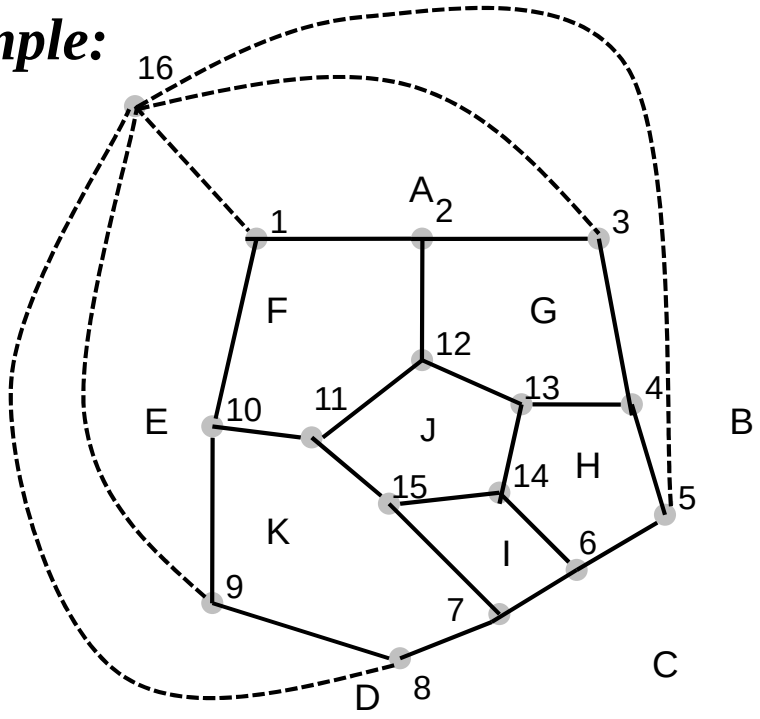
Floorplanning by Rectangular Dualization

Example:



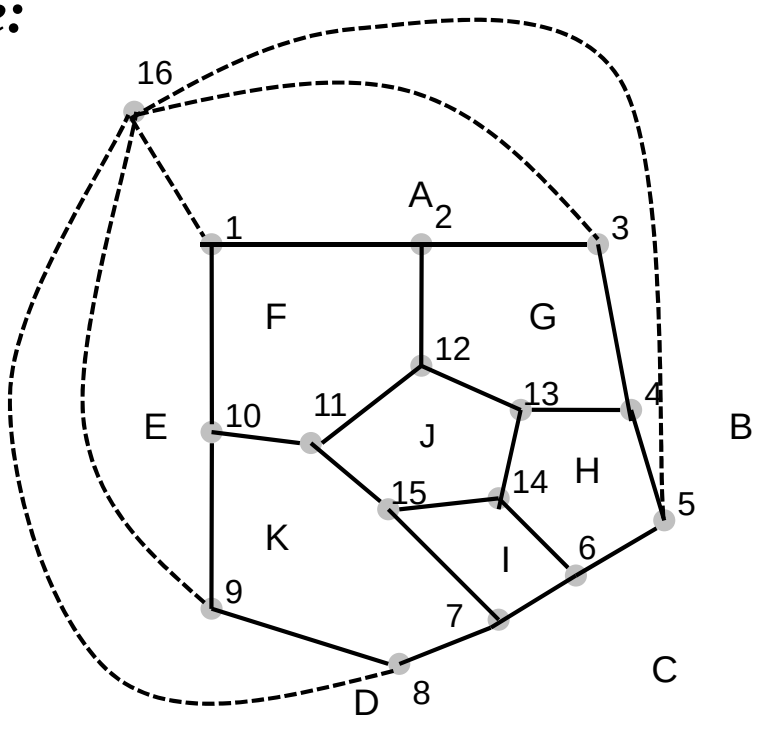
Floorplanning by Rectangular Dualization

Example:



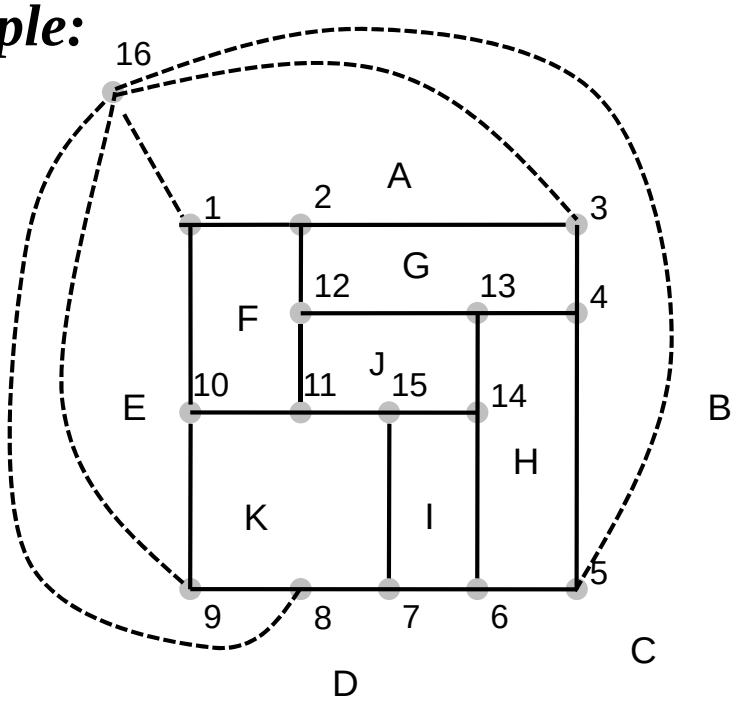
Floorplanning by Rectangular Dualization

Example:



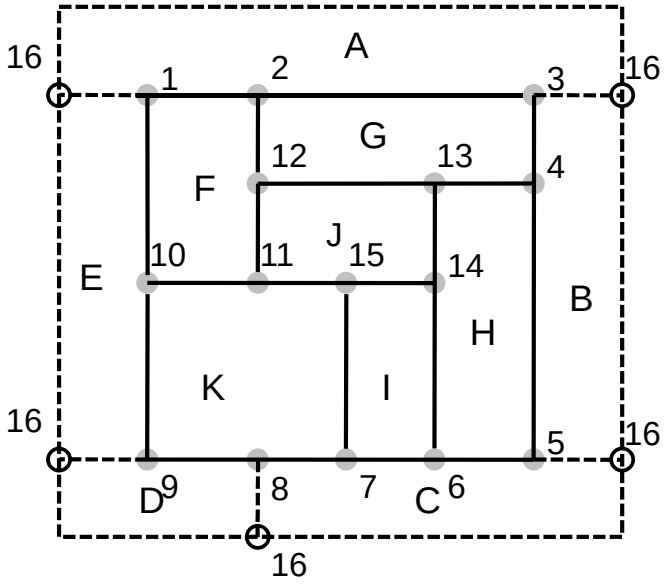
Floorplanning by Rectangular Dualization

Example:



Floorplanning by Rectangular Dualization

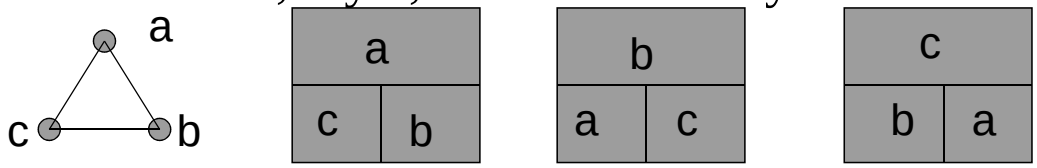
Example:



Problems: netlist is hypergraph, obtaining planar triangulated graph for the netlist, block areas not considered

Hierarchical approach

- Divide and conquer paradigm, a small number of rectangles are considered
- Enumerate all possible floorplans for a small number of modules, say 3, w.r.t connectivity



- Bottom-up approach :
  - cluster modules depending on connectivity, when we have few super blocks, enumerate all possibilities and take the best one
- Top-down approach: recursively partition netlist and enumerate at each level of hierarchy

Graph Representation of Floorplan

Polar Graph representation:

- A floorplan is modeled by a pair of directed acyclic graphs:
  - Horizontal polar graph
  - Vertical polar graph
- For horizontal (vertical) polar graph,
  - Vertex: Vertical (horizontal) channel
  - Edge: 2 channels are on 2 sides of a block
  - Edge weight: Width (height) of the block

Hierarchical approach: bottom-up

Simulated annealing based approach

- Combinatorial optimization problems (like partitioning) can be thought as a State Space Search Problem.
- A State is just a configuration of the combinatorial objects involved.
- The State Space is the set of all possible states (configurations).
- A Neighbourhood Structure is also defined (which states can one go in one step).
- There is a cost corresponding to each state.
- Search for the min (or max) cost state.

State Space Search Problem

- 3 2 1 cost = 1 (only one element (2) is in proper place)
- 3 1 2 cost = 2 (none is in proper place)
- 1 3 2 cost = 1 (only one element (1) is in proper place)
- 1 2 3 cost = 0 (all the tree elements are in proper place)
- 2 1 3 cost = 1 (only one element (3) is in proper place)
- 2 3 1 cost = 2 (none is in proper place)

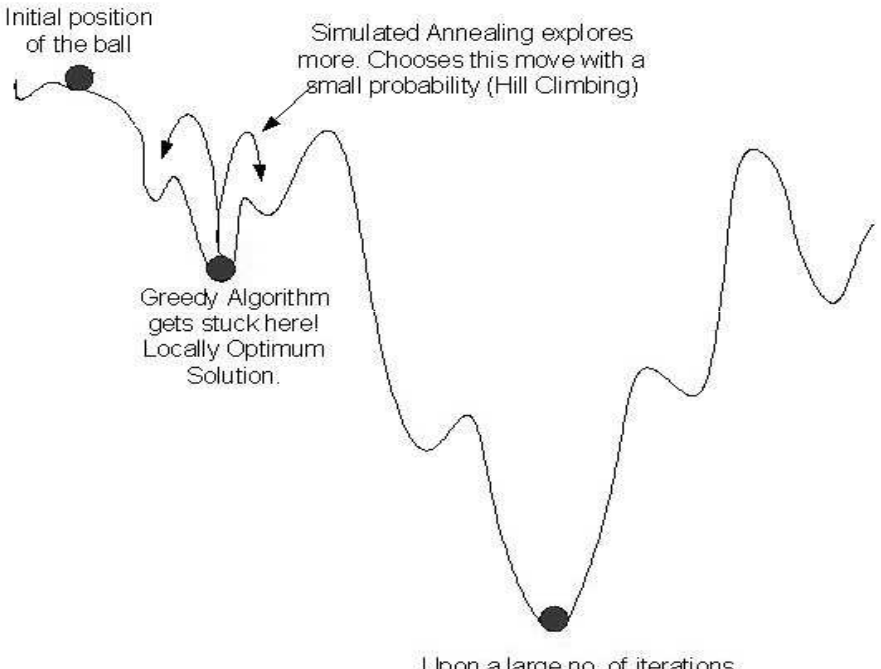
Greedy Algorithm

- A very simple technique for State Space Search Problem.
- Start from any state.
- Always move to a neighbor with the min cost (assume minimization problem).
- Stop when all neighbors have a higher cost than the current state.

Simulated Annealing

- Very general search technique.
- Try to avoid being trapped in local minimum by making probabilistic moves.
- Popularize as a heuristic for optimization by:
  - Kirkpatrick, Gelatt and Vecchi, “Optimization by Simulated Annealing”, Science, 220(4598):498-516, May 1983.

Simulated Annealing



Movement of ball on terrain

Simulated Annealing

Motivation

- Annealing in metals
- Heat the solid state metal to a high temperature
- Cool it down very slowly according to a specific schedule
- *If the heating temperature is sufficiently high to ensure random state and the cooling process is slow enough to ensure thermal equilibrium, then the atoms will place themselves in a pattern that corresponds to the global energy minimum of a perfect crystal*
- Attaining a min cost state in simulated annealing is analogous to attaining a good crystal structure in annealing

Simulated Annealing

The procedure:

Let t be the initial temperature.

Repeat

  Repeat

  - Pick a neighbor of the current state randomly.
  - Let c = cost of current state.
    Let c' = cost of the neighbour picked.
  - If c' < c, then move to the neighbour (downhill move).
  - If c' > c, then move to the neighbour with probablility $e^{-(c'-c)/t}$ (uphill move).

  Until equilibrium is reached.

Reduce t according to cooling schedule.

Until Freezing point is reached.

Simulated Annealing

Things to be decided while using SA:

- The state space
- The neighborhood structure
- The cost function
- The initial state
- The initial temperature
- The cooling schedule (how to change t)
- The freezing point

Simulated Annealing for floorplanning

Given: set of modules and finite number of shapes(w,h)

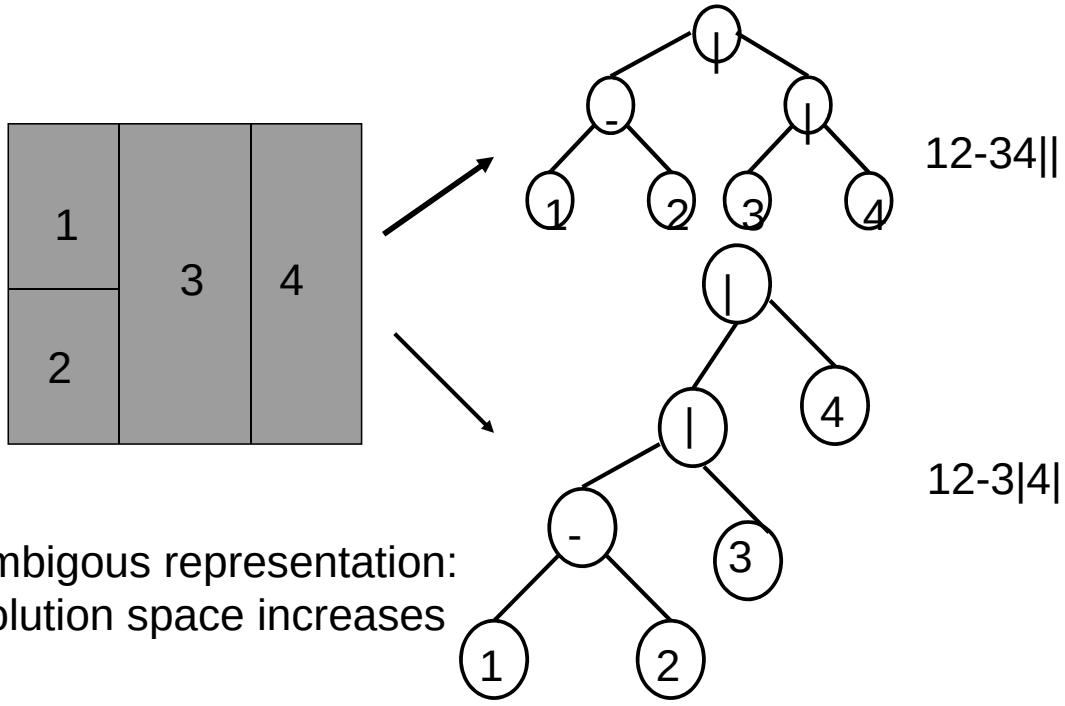
Solution Space: a valid slicing floorplan having non overlapping modules

Obtaining new solution: rules to generate a new solotion from the current solution

Cost function: minimize area and total wire length

Simulated Annealing for floorplanning

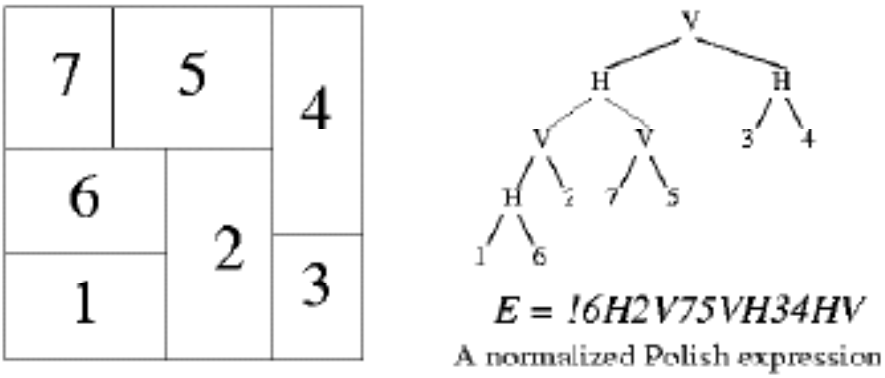
Start with an initial floorplan: Initial slicing tree



Ambigous representation:
Solution space increases

Simulated Annealing for floorplanning

- A Polish expression $E = e_1\ e_2\ \ldots\ e_{2n-1}$ is called normalized iff E has no consecutive operators of the same type (H or V).
- Given a normalized Polish expression, we can construct a unique rectangular slicing structure.



E = !6H2V75VH34HV

A normalized Polish expression

Simulated Annealing for floorplanning

•Obtaining new floorplan configuration /solution:

•3 types of moves:

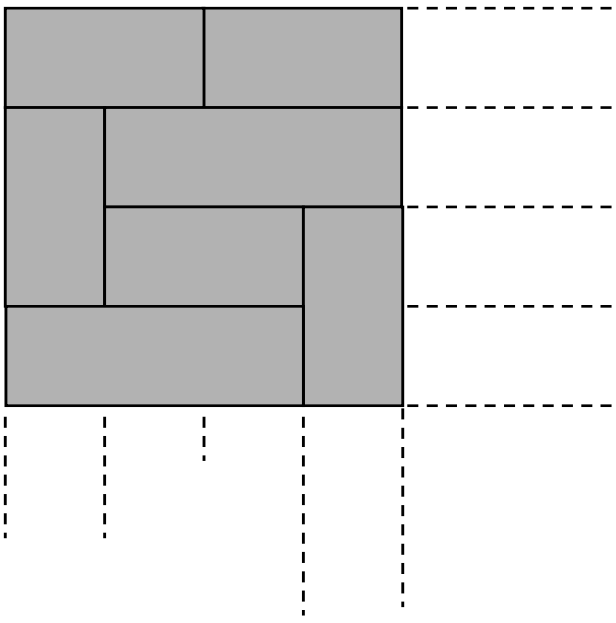
M1 (**Operand Swap**): Swap two adjacent operands
 operands.
M2 (**Chain Invert**): Complement some chain (*V* = *H*, *H* = V).
M3 (**Operator/Operand Swap**): Swap two adjacent operand and operator.

•New configuration is also normalized polish expression

Graph Representation of Floorplan

Polar Graph representation:

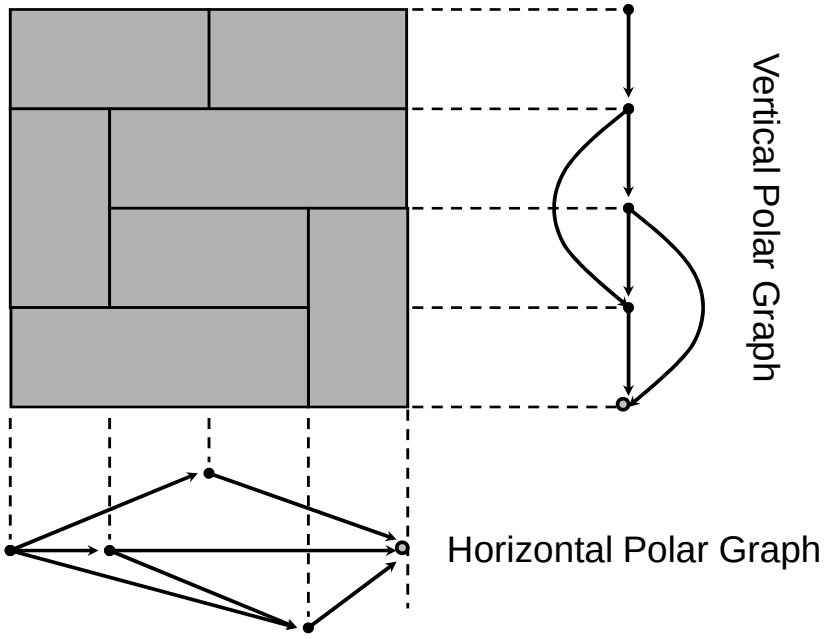
Example:



Graph Representation of Floorplan

Polar Graph representation:

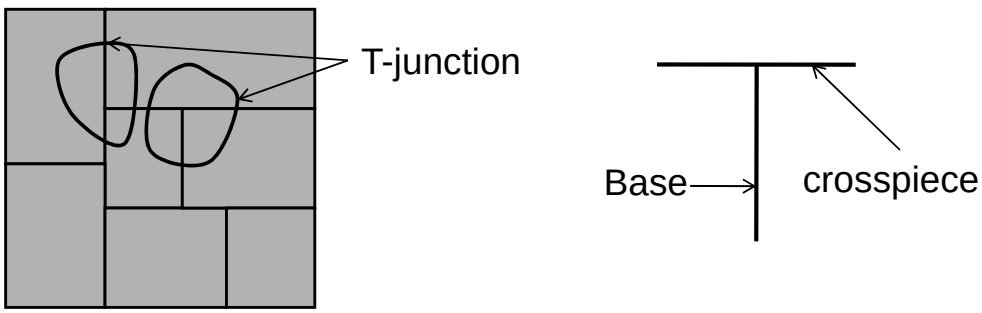
Example:



Graph Representation of Floorplan

Channel digraph representation:

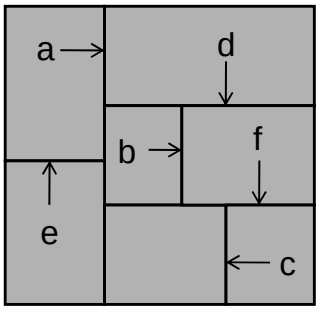
- A floorplan is modeled by a directed graph:
  - Vertex: channel or cut
  - Edge: (v_i, v_j) is an edge if the cut corresponding to v_i is the base an the cut corresponding to v_j is the crosspiece of a T-junction

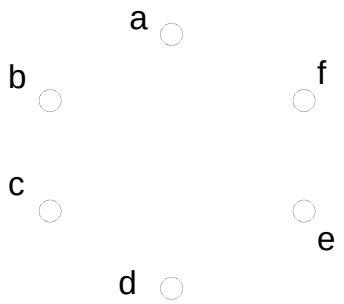


Graph Representation of Floorplan

Channel digraph representation:

Example:

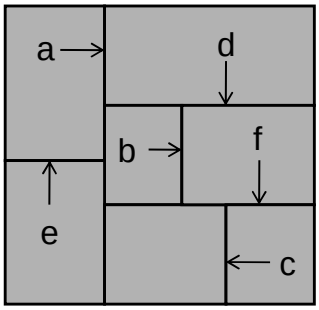


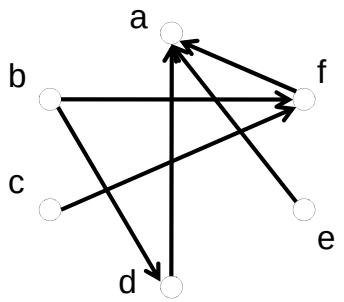


Graph Representation of Floorplan

Channel digraph representation:

Example:

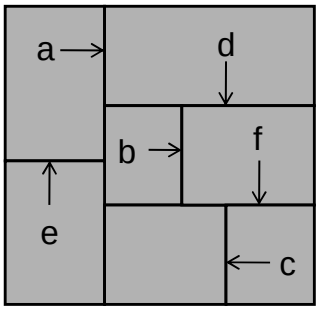


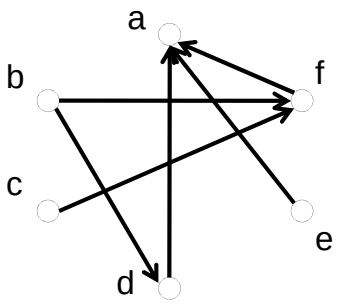


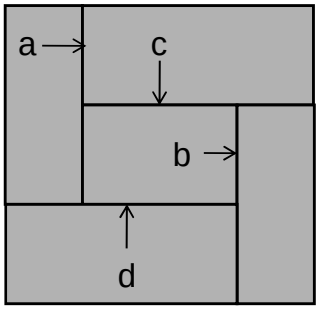
Graph Representation of Floorplan

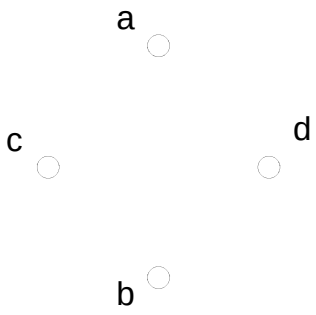
Channel digraph representation:

Example:





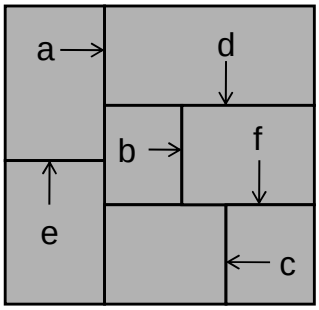


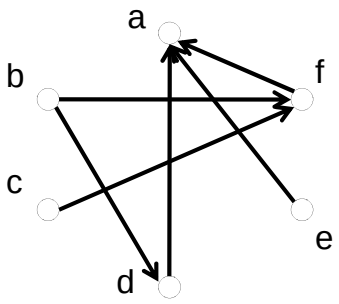


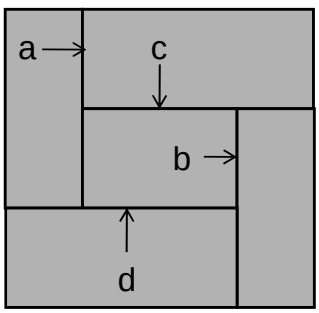
Graph Representation of Floorplan

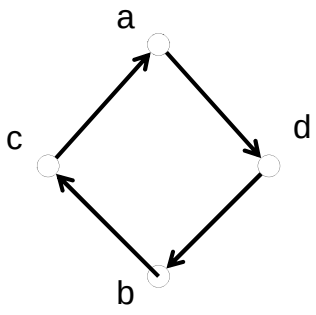
Channel digraph representation:

Example:









Graph Representation of Floorplan

Channel digraph representation:

Properties:

- A vertex in channel digraph corresponding to a through cut (a cut whose each end touches the boundary) has outdegree = 0
  - A through cut cannot be a base
- A floorplan is slicible iff its channel digraph is acyclic
- 4-cycle theorem: A floorplan is non-slicible iff there exist at least one cycle of length 4 in the corresponding channel digraph
  - The channel digraph of the simplest non-slicing floorplan (shown in the previous foil) is a 4-cycle