Partitioning Ⓟ

Goal: Partition *cells* into *blocks* $\Longrightarrow$ Graph partitioning.
Given a graph $G(V,E)$, with $|V|$ nodes and $|E|$ edges.
Each node has area $s(v)$ and each edge has cost/weight $w(e)$.
Divide graph G into k disjoint subgraphs. (NP-hard)
Optimization goals:

- Minimum number of connections between partitions
- Design constraints (size, external connections)
- Balanced partition

◇ **Kernighan-Lin Algorithm**.
Given: Graph with $2n$ nodes, same weight.
Goal: Partition into A and B; min. cost, $|A|=|B|=n$.
*Cost $D(v)$ of moving a node v

$$D(v)=|E_c(v)|-|E_{nc}(v)|$$

$$E_c(v)/E_{nc}(v)=\text{edges of } v \text{ that are (not) cut}$$

*Gain of swapping a pair of nodes a and b

$$\Delta g = D(a)+D(b)-2\cdot c(a,b)$$

$$D(a),D(b)=\text{costs of nodes } a,b$$

$$c(a,b)=\text{connection weight between } a \text{ and } b$$

*Maximum positive gain G_m of a pass

$$G_m=\sum_{i=1}^{m}\Delta g_i$$

Algorithm: $O(n^3)$ per pass

repeat `// a single pass`
  repeat
    calculate $D(v)$ for all nodes
    find max. Δg_i and swap
  until each node swapped over
  select max. positive gain $G_m=\sum_{i=1}^{m}\Delta g_i$
  perform m moves
until no improvement ($G_m<0$)

Extensions:

- unequal partition size
- unequal node weights
- k-way partitioning (recursive)

◇ **Fiduccia-Mattheyses Algorithm**.

- swapping becomes moving nodes
- slight imbalance allowed
- edges become nets (w/ 2+ pins)
- Node = cell; subgraph = block

Given: Hypergraph $G(V,H)$, weighted hyperedges
Goal: Partition int k, min. weight cost
*Gain $\Delta g(c)$ for cell c

$$\Delta g(c)=FS(c)-TE(c)$$

$$FS(c)=\text{moving force, i.e. cut nets only connected to } c$$

$$TE(c)=\text{retention force, i.e. uncet nets connected to } c$$

*Maximum positive gain G_m of a pass

$$G_m=\sum_{i=1}^{m}\Delta g_i$$

*Ratio factor, i.e. relative balance between two partitions

$$r=\frac{area(A)}{area(A)+area(B)}$$

*Balance criterion

$$area(V)\in[r\cdot area(V)\pm area_{max}(V)]$$

*Base cell: Cell c with max. cell gain $\Delta g(c)$ among all free cells and does not violate balance criterion
Algorithm: $O(n)$ per pass (sorted node list)

repeat `// a single pass`
  compute balance criterion
  repeat
    find max. Δg_i and fix cell c_i
  until all cells are fixed
  select max. positive gain $G_m=\sum_{i=1}^{m}\Delta g_i$
  perform m moves
until no improvement

◇ **Multilevel Partitioning**.
Clustering: ($\Longrightarrow$ Simulated Annealing)

- group tighlty-connected nodes into clusters
- partition clusters
- refine partition by splitting clusters

Floor Planning Ⓟ

$$\left\{\begin{array}{c}\text{Set of}\\ \text{modules}\end{array}\right\}\Rightarrow\begin{array}{l}\text{- Layout}\\ \text{- Network}\\ \text{- Pins}\end{array}$$

Optimization goals:

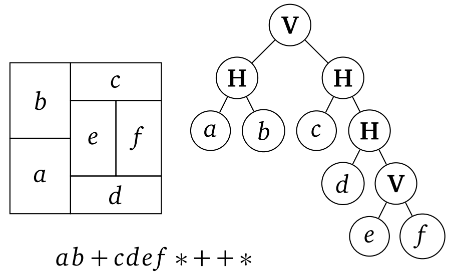
- minimize area of global bounding box
- minimize total wire length
- combination of the above ($\alpha-\beta$)
- signal delays (critical path)

*Rectangular dissection, division of the chip area into a set of blocks (non-overlapping rectangles).
*Slicing floorplan, rectangular dissection

- Obtained by repeatedly dividing each rectangle
- Horizontal or vertical cut line

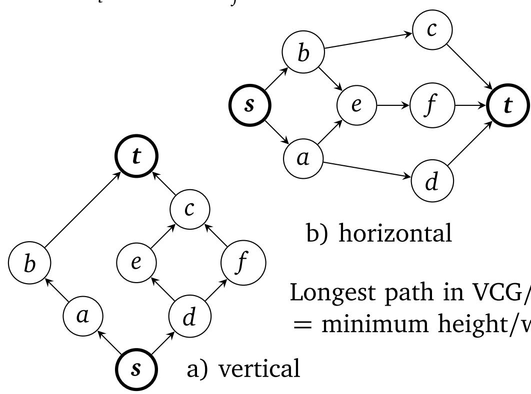
*Slicing tree, binary tree with k leaves, $k-1$ internal nodes

- Each leaf represents a block
- Each internal node represents a h/v cut line
- Polish expression ($V\to *, H\to +, W\to$ wheel)



$$ab+cdef*+++*$$

*Non-slicing floorplan (wheels)
*Vertical constraint graph (VCG)
Edge from node v_i to v_j if block m_i under m_j.
*Horizontal constraint graph (HCG)
Edge from node v_i to v_j
if block m_i is left of m_j.



b) horizontal

Longest path in VCG/HCG
= minimum height/width

a) vertical

*Sequence pair (S_+,S_-)
Two permutations represent geometric relations between every pair of blocks, e.g. $(bacedf, abcdecf)$

$$(\ldots A\ldots B\ldots,\ldots A\ldots B\ldots)\to A \text{ is left of } B$$

$$(\ldots A\ldots B\ldots,\ldots B\ldots A\ldots)\to A \text{ is above of } B$$

$$(\ldots B\ldots A\ldots,\ldots A\ldots B\ldots)\to A \text{ is below of } B$$

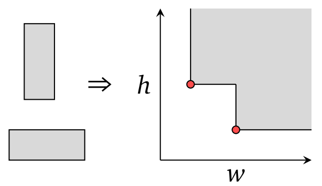
$$(\ldots B\ldots A\ldots,\ldots B\ldots A\ldots)\to A \text{ is right of } B$$

◇ **Floorplan Sizing.**

Goals: Minimize wire length and area

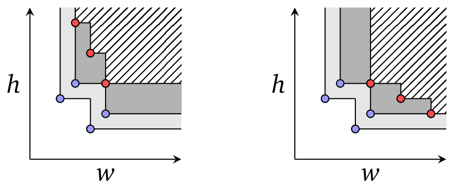
*Shape function,*Corner points

Step 1: Construct the shape functions of the blocks



Step 2: Determine the shape fun. of the top-level floorplan

Choose minium bounding box ($w \times h$)



a) vertical (↑) b) horizontal (→)

Step 3: Find individual block's dimensions and locations

◇ **Cluster Growth.**

- Iteratively add blocks to the cluster until all are assigned
- Only the different orientations of the blocks are taken into account
- Linear ordering to minimize total wire length

Continue..?

Placement Ⓟ