ECE 720 – ESL & Physical Design

Lecture 13: Partitioning & Placement

W. Rhett Davis
NC State University
(with lots of slides from A. Richard Newton)

Announcements

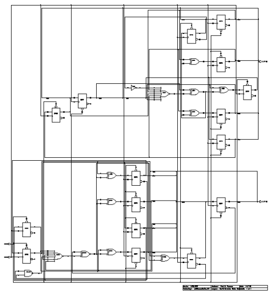
- Homework 5 solution posted
- Project 2 Due in 1 Week
- Homework 6 Due in 2 Weeks

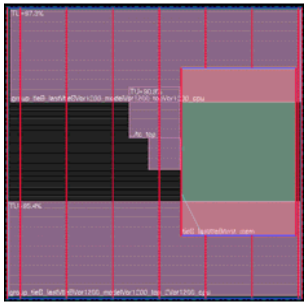
Design Flow

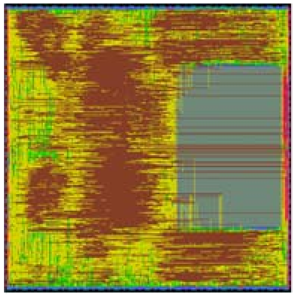
Front-End Design / Logic Design
- Standard Cell Synthesis

Back-End Design / Physical Design
- Floorplanning
- Placement
- Clock Tree Synthesis
- Repeater Insertion
- Routing

Verification
- Static Timing Analysis
- Signal Integrity Analysis
- Power Analysis
- Power Distribution Analysis

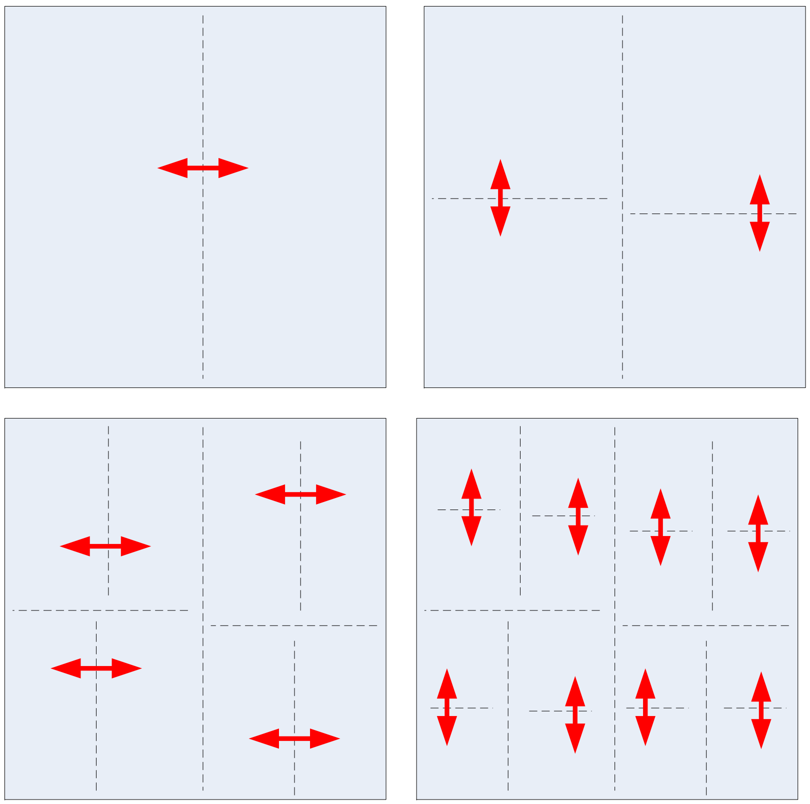






Partitioning is Placement

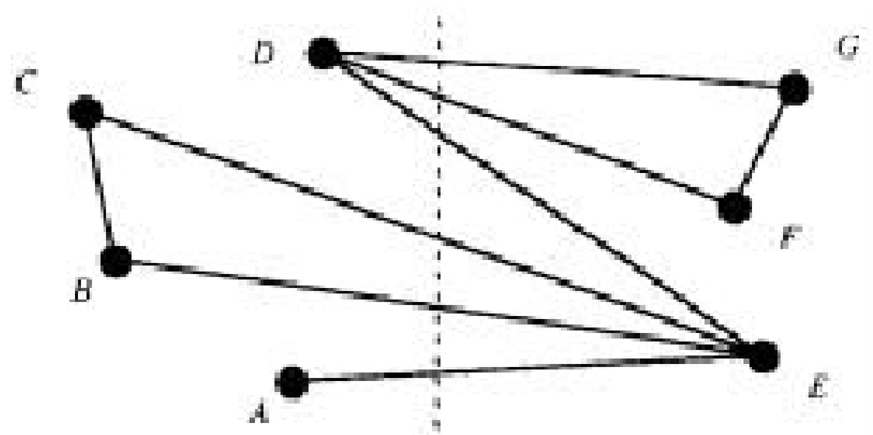
- With a good partitioning algorithm, placement can be accomplished with a successive partitioning of regions



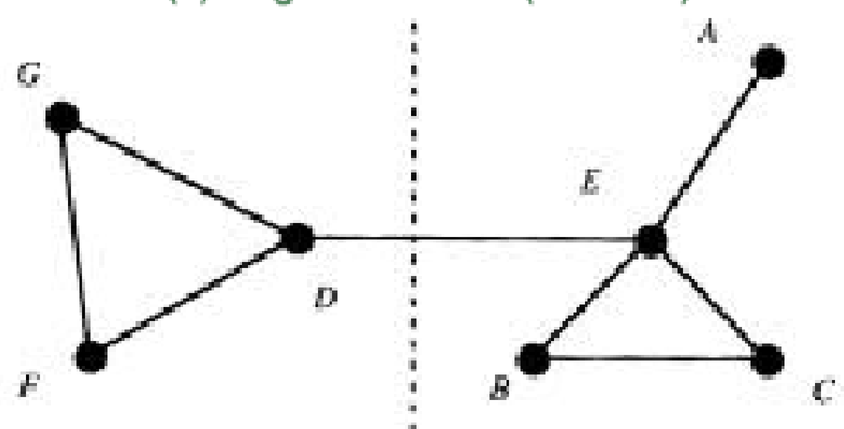
Today's Lecture

- Kernighan & Lin (1970)
- Fiduccia & Mattheyses (1982)
- Karypis – hMETIS (1997)
- Antreich - GORDIAN (1991)

Partitioning for a Minimal Cut-Set



(a) Original Partition (Random)



(b) Improved Partition

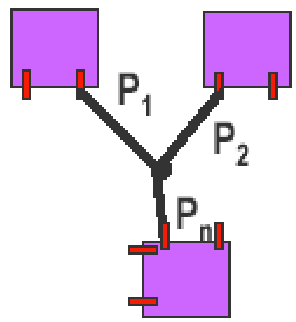
Converting a Hypergraph into a Graph

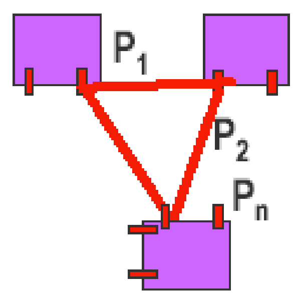
- Early partitioning algorithms worked on graphs
  - edges between exactly two vertices
- Most netlists are hypergraphs
  - hyperedges can exist between two or more vertices
- How to convert?

  - Replace each net S_i with its complete graph.
  - What weight on each edge?
  - For n-pin net, w=2/(n-1) has been used, also w=2/n
  - Best weight is:

    $$w = 4/(n^2\text{-}mod(n,2))$$

    for n=3, w=4/(9-1)=0.5





Partitioning

- Given a graph, *G*, with *n* nodes with sizes (weights) *w*:

  $$0 < wi \le p, i = 1, \cdots, n$$

  with costs on its edges, partition the nodes of *G* into *k*, subsets, *k >0*, no larger than a given maximim size, *p*, so as to minimize the total cost of the edges cut.
- Define : $C = (cij), i, j = 1, \cdots, n$

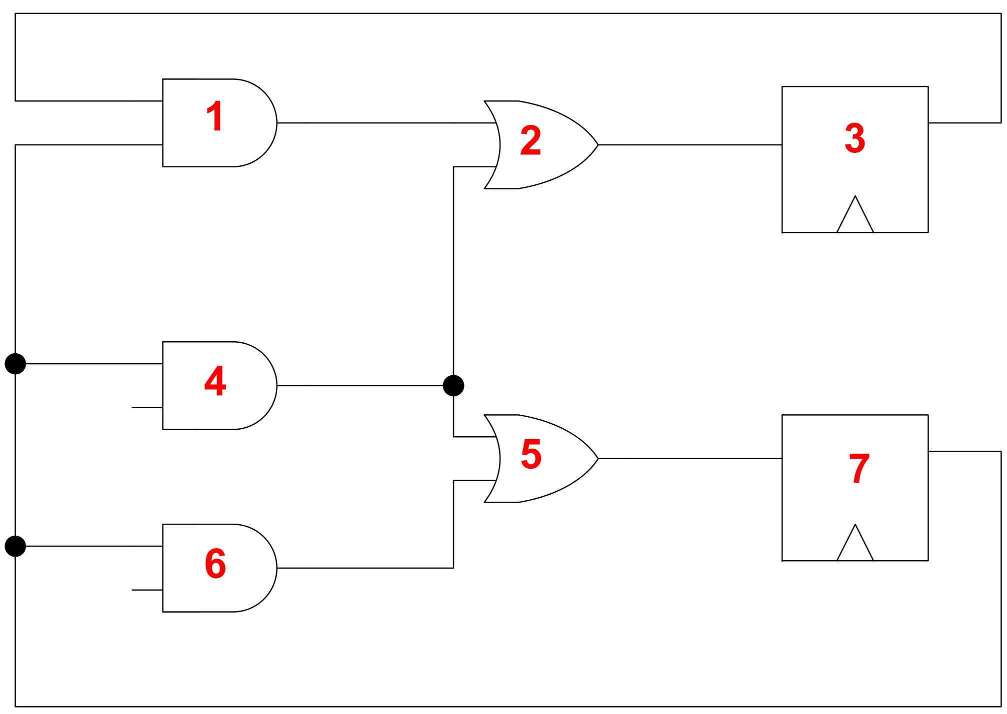
  as a weighted connectivity matrix describing the edges of *G*.
- A *k-way partition* of *G* is a set of non-empty, pairwise-disjoint subsets of *G*, *v1,…,vk*, such that $\bigcup_{i=1}^{k} vi = G$
- A partition is said to be admissible if $|vi| \le p, i = 1, \cdots, k$
- Problem: Find a minimal-cost permissible partition of *G*

permissible if $\forall_i \sum_{a \in v_i} w_a \le P$

2-Way Partitioning (Kernighan & Lin)

- Consider the set *S* of *2n* vertices, all of equal size for now, with an associated cost matrix $C = (c_{ij}), i, j = 1, \cdots, 2n$
- Assume *C* is symmetric and $c_{ii} = 0 \forall i$
- We want to partition *S* into two subsets *A* and *B*, each with *n* points, such that the external cost $T = \sum_{A \times B} c_{ab}$ is minimized
- Start with any arbitrary partition [*A*,*B*] of *S* and try to decrease the initial cost *T* by a series of interchanges of subsets of *A* and *B*
- When no further improvement is possible, the resulting partition [*A'*,*B'*] is a *local minimum* and has a fairly high probability of being a *global minimum* with this scheme

Example



- edge weight = $4/(n^2\text{-mod}(n,2))$ where n=no. of terminals on net
- Annotate this graph with its edge weights

Example

- Start w/ partition A = { 1, 2, 4, 5, 6 }
  B = { 3, 7 }

$$T = \sum_{A \times B} c_{ab} =$$

- Assume weights w_{AND}, w_{OR}=4, w_{FF} = 10
  And partition size P = 22
  Is this partition admissible?

$$\sum_{a \in A} w_a = \qquad \sum_{b \in B} w_b =$$

Example

- Find two cells to swap such that gain is maximized
- Exchange 3 for 6

A = { 1, 2, 3, 4, 5 }
B = { 6, 7 }

  - $T = \sum_{A \times B} c_{ab} =$
  - gain =

- Is this partition admissible?

$\sum_{a \in A} w_a =$

$\sum_{b \in B} w_b =$

Example

- Find two cells to swap such that gain is maximized
- Exchange 5

A = { 1, 2, 3, 4 }
B = { 5, 6, 7 }

  - $T = \sum_{A \times B} c_{ab} =$
  - gain =

- Is this partition admissible?

$\sum_{a \in A} w_a =$

$\sum_{b \in B} w_b =$

Example

- Find two cells to swap such that gain is maximized
- Exchange 4

A = { 1, 2, 3 }
B = { 4, 5, 6, 7 }

  - $T = \sum_{A \times B} c_{ab} =$
  - gain =

- Is this partition admissible?

$\sum_{a \in A} w_a =$

$\sum_{b \in B} w_b =$

2-Way Partitioning (Kernighan & Lin)

- For each $a \in A$:
  - external cost $E_a = \sum_{y \in B} c_{ay}$ (same for E_b)
  - internal cost $I_a = \sum_{x \in A} c_{ax}$ (same for I_b)

  $D_z = E_z - I_z \forall z \in S$
- If $a \in A$ and $b \in B$ are interchanged, then the gain:

  $g = D_a + D_b - 2c_{ab}$
- Proof: If Z is the total cost of connections between A and B, excluding a and b, then:

$$\left.\begin{aligned} T_{a,b} &= Z + E_a + E_b - c_{ab} \\ T_{b,a} &= Z + I_a + I_b + c_{ab} \end{aligned}\right\} gain = T_{a,b} - T_{b,a} = D_a + D_b - 2c_{ab}$$

2-Way Partitioning (Kernighan & Lin)

(1) Compute all *D* values in *S*

(2) Choose a_i, b_i such that $g_i = D_{a_i} + D_{b_j} - 2c_{a_i b_j}$
is maximized

(3) Set a_i and b_i asside and call them a_i' and b_i'

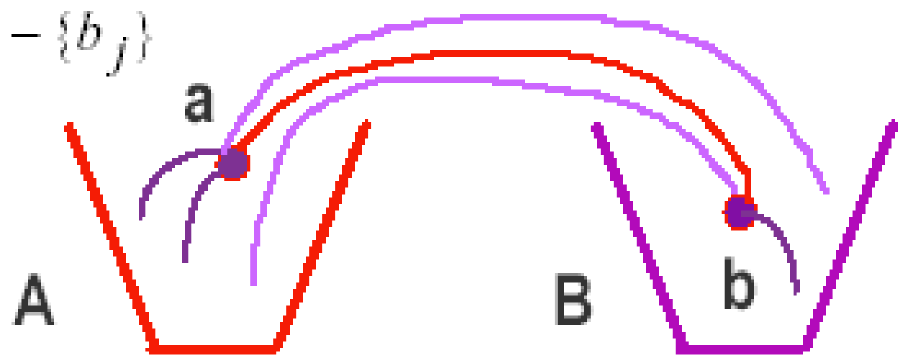
(4) Recalculate the D values for all the elements of
$A - \{a_i\},\ B - \{b_j\}$

$$D_x' = D_x + 2c_{xa_i} - 2c_{xb_j},\ x \in A - \{a_i\}$$

$$D_y' = D_y + 2c_{yb_j} - 2c_{ya_i},\ y \in B - \{b_j\}$$

$$D'_{ai} = -D_{ai}$$

$$D'_{bj} = -D_{bj}$$



Example, Revisited

swap	z	1	2	3	4	5	6	7
	E_z	5/4	1	2	1/4	1	1/4	7/4
	I_z	3/2	2	0	3/2	2	3/2	0
	D_z	-1/4	-1	2	-5/4	-1	-5/4	7/4
6,3	$2c_z a_i$	1/2	0	-	1/2	2	-	1/2
	$2c_z b_j$	2	2	-	0	0	-	0
	D_z	-7/4	-3	-2	-3/4	1	5/4	5/4
5,-	$2c_z a_i$	0	1	0	1	-	2	2
	D_z	-7/4	-2	-2	1/4	-1	-3/4	-3/4
4,-	$2c_z a_i$	1/2	1	0	-	1	1/2	1/2
	D_z	-5/4	-1	-2	-1/4	0	-1/4	-1/4

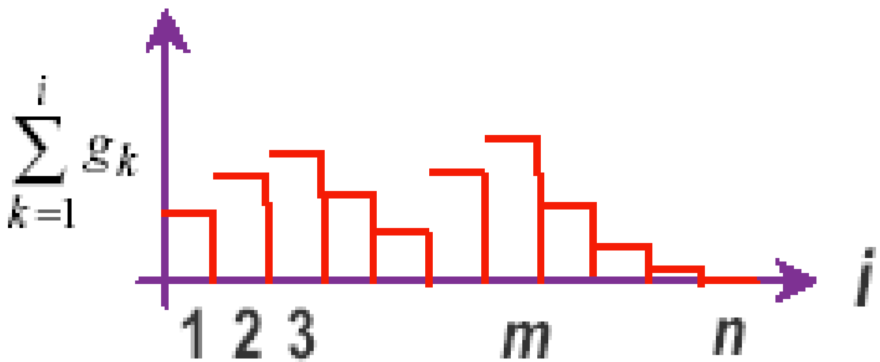
- Initial partition
  - A={1,2,4,5,6}
  - B={3,7}
- Swap a_i=6, b_j=3
  - g=2-5/4-0=3/4
- Swap a_i=5
  - g=1
- Swap a_i=4
  - g=1/4

Number of operations is reduced

2-Way Partitioning (Kernighan & Lin)

- Repeat (2)-(4) on a new pair until all nodes exhausted

$$(a_1', b_1'), (a_2', b_2'), \cdots, (a_n', b_n')$$



- If sum to $m > 0$, some gain, so repeat until sum to m=0

Today's Lecture

- Kernighan & Lin (1970)
- Fiduccia & Mattheyses (1982)
- Karypis – hMETIS (1997)
- Antreich - GORDIAN (1991)

2-Way Partitioning (Fiduccia & Mattheyses)

- Move one cell at a time from one side of the partition to the other in an attempt to minimize the cutset of the final partition
  - base cell -- cell to be moved
  - gain *g(i)* -- no. of nets by which the cutset would decrease if cell i were moved from partition *A* to partition *B* (may be negative)
- To prevent thrashing, once a cell is moved it is locked for an entire pass
- Claim is O(*n*) time

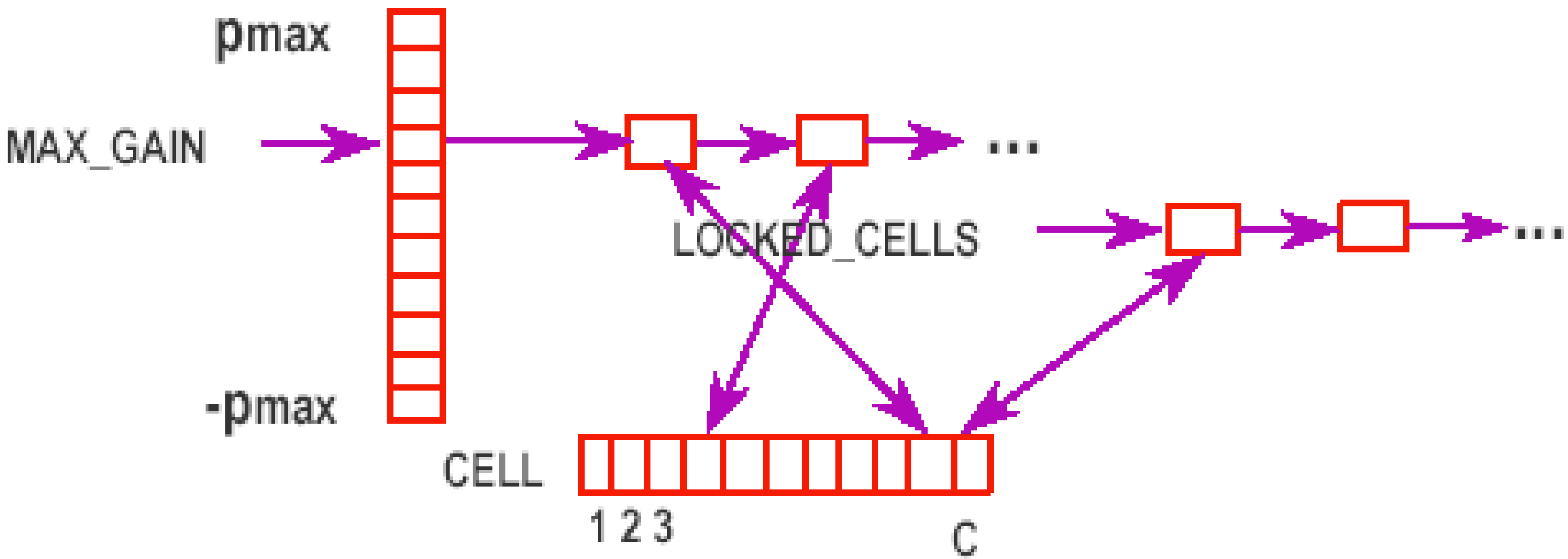
2-Way Partitioning (Fiduccia & Mattheyses)

- Steps:
  (1) Choose a cell
  (2) Move it
  (3) Update the g(i)'s of the neighbors
- O(n) neighbors, each cell recomputed each time neighbor moved, and must recompute for each pin (assume #pins=K.n, Amdahl 470 K~2.7)

$$n_i^2 + n_i^2 + \cdots n_i^2 = O(\#\ pins^{\ 2}) /\ pass\ !$$

2-Way Partitioning (Fiduccia & Mattheyses)

- If *p(i)* = no. of pins on cell *i*: $-p(i) < g_i < p(i)$
- Bin-sort cells on *gi*

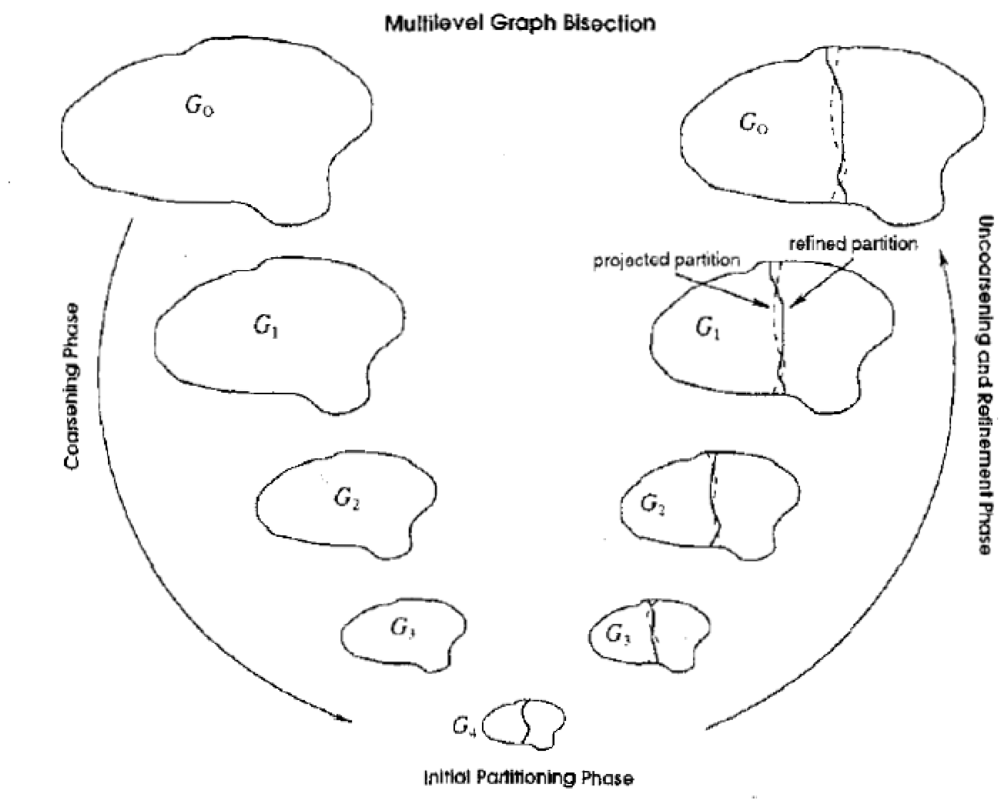


- Time required to maintain each bucket array O(P)/pass

Today's Lecture

- Kernighan & Lin (1970)
- Fiduccia & Mattheyses (1982)
- Karypis – hMETIS (1997)
- Antreich - GORDIAN (1991)

Coarsening/Uncoarsening

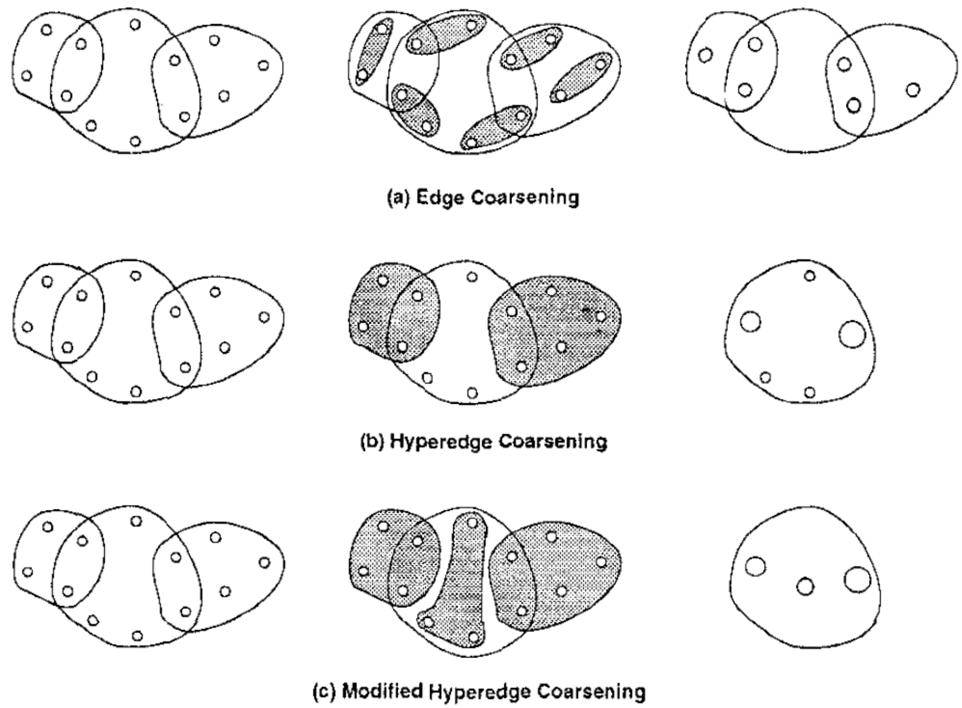


- To speed up partitioning, a large hypergraph is reduced to a smaller one through "coarsening"
- The reduced hypergraph is then successively partitioned (using KLFM) and "uncoarsened"

Source: Karypis, *et al, DAC* 1997

3 Types of Coarsening

- Different types are used during coarsening process



- In the figure, each circle represents a hyperedge (or multi-terminal net), each vertex represents a cell

Source: Karypis, *et al, DAC* 1997

- Claims partitions equal in quality to KLFM, but 10-100 times faster

Today's Lecture

- Kernighan & Lin (1970)
- Fiduccia & Mattheyses (1982)
- Karypis – hMETIS (1997)
- Antreich - GORDIAN (1991)

GORDIAN

J. Kleinhaus, G. Sigl, F. Johannes, K. Antreich, *GORDIAN: VLSI Placement by Quadratic Programming and Slicing Optimization*, IEEE Trans. on CAD, March, 1991, pp. 356 - 365

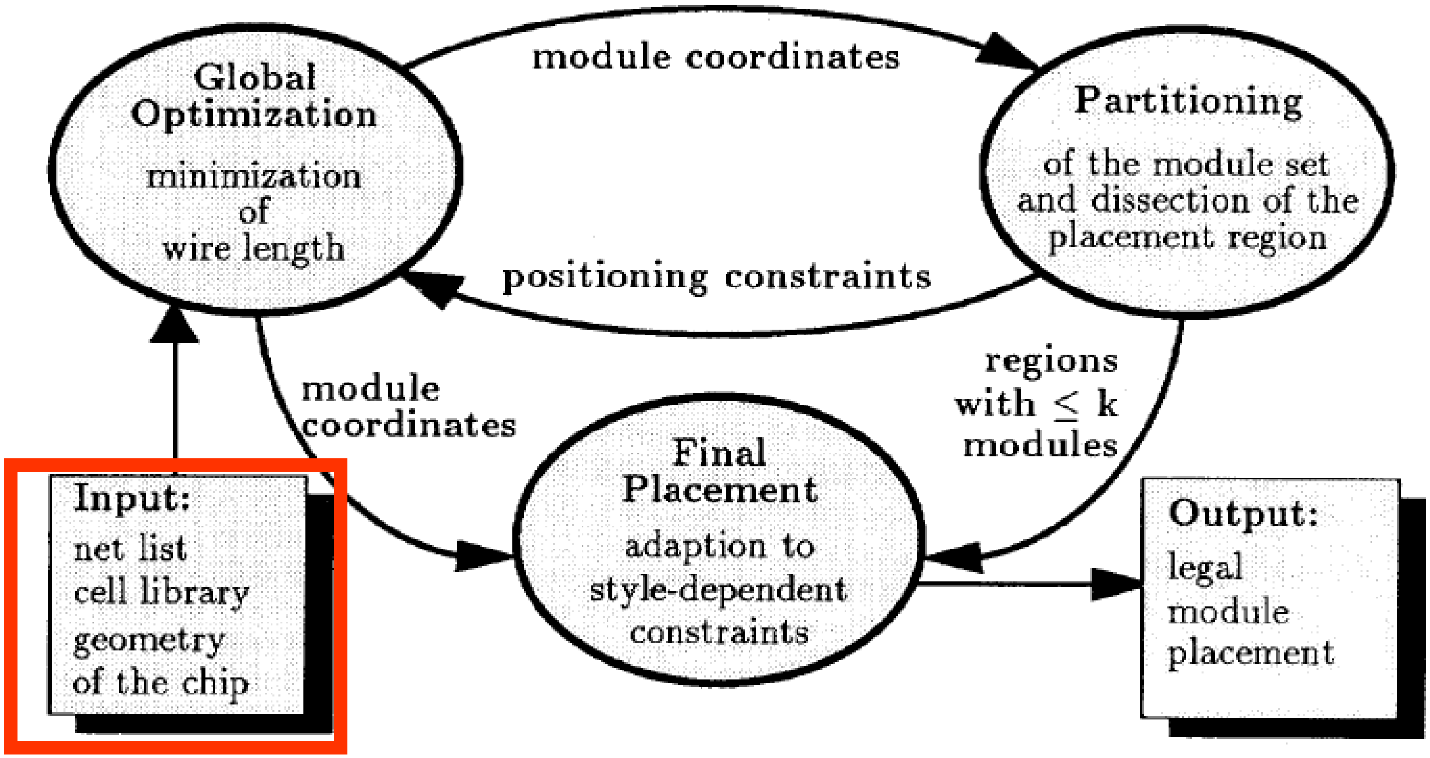


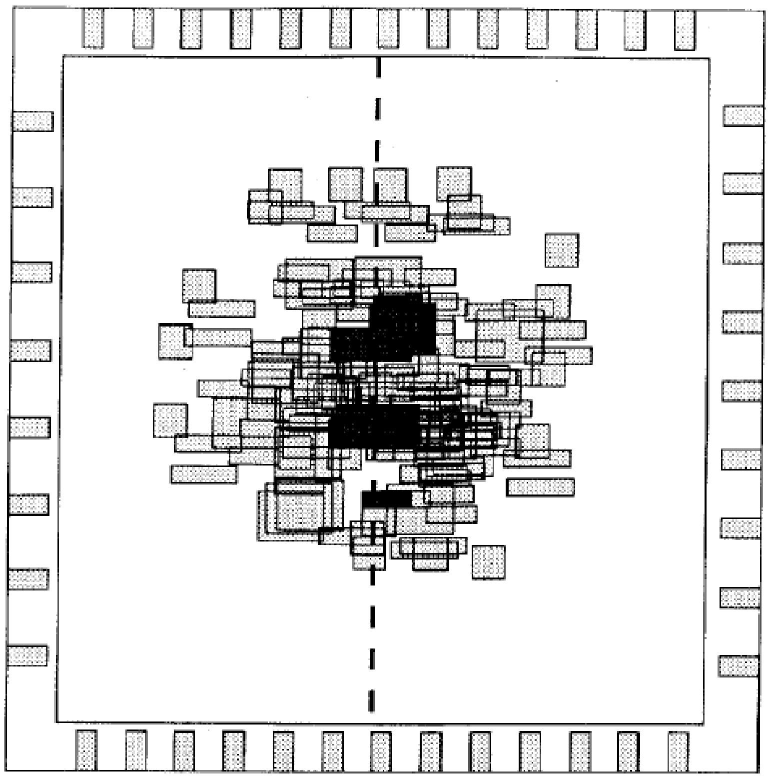
Fig. 1. Data flow in the placement procedure GORDIAN.

Basic Approach of GORDIAN

- A System Matrix (C) is defined that includes the squared "rubber-band" lengths of each net
- A Constraint Matrix (A) is defined that gives the current position of each block
- Objective function ϕ gives total cost of current placement, and is guaranteed to be convex and have a global minimum
- Convex optimization approach can be used to find the optimum
- Constraint Matrix is updated, and we iterate until a good solution is found

$$\text{LQP:} \min_{x \in \Re^m} \left\{ \phi(x) = \tfrac{1}{2} \boldsymbol{x}^T \boldsymbol{C} \boldsymbol{x} + \boldsymbol{d}^T \boldsymbol{x} \,\middle|\, \boldsymbol{A}^{(l)} \boldsymbol{x} = \boldsymbol{u}^{(l)} \right\}.$$

GORDIAN



- In 1991, placement with GORDIAN left a large amount of unused area
- Since then, many of these problems have been solved, and this basic formulation is commonly used
  - Dragon2000 – Wang, Yang, & Sarrafzadeh ICCAD 2000
  - FastPlace – Viswanathan & Chu, IEEE Trans. CAD 2005