

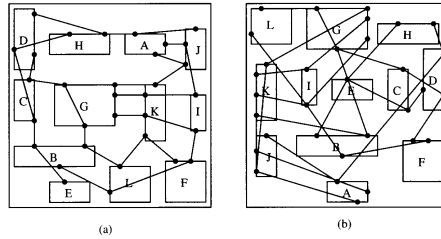
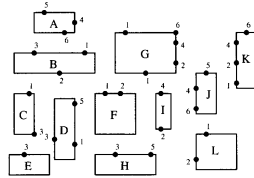
Placement

Special thanks to: Jason Cong (UCLA) and C.K Koh(Purdue)

Placement

- Input to the placement:
 - A set of blocks with well-defined shapes
 - Pin locations
 - A netlist
- Objectives:
 - Minimize area
 - Reduce net length for critical nets

Consequences of Placement

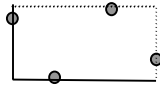


Placement Problem Formulation

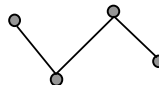
- No two rectangles overlap
- Placement is routable
- The total area of the rectangle bounding cells and routing regions is minimized
 - Very difficult to estimate
- The total wire length is minimized

Routing Estimate in Placement

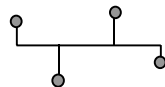
Bounding Box



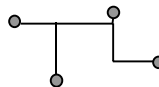
Spanning



Spine



Steiner



Classification of Placement Algorithms

- Partitioning based algorithms
- Clustering based algorithms
- Simulation based algorithms
 - Simulated annealing
 - Simulated evolution (genetic algorithm)
 - Force-directed placement
- Analytical approaches
 - Quadratic programming
 - Resistive network optimization
- Performance-driven placement

Partitioning or Min-Cut Placement [Breuer, DAC77]

- Make use of partitioning techniques
- Partition circuit alternately in the horizontal and vertical directions
- Use areas of sub-circuits to determine cutline on the chip, assign each sub-circuit on one side of cutline
- Stop when each sub-circuit has only one single gate
- Objective function:
 - Number of nets crossing the cutline
 - Weighted sum of wire length and cut number

Partitioning Algorithm [Kernighan-Lin, Bell'70]

- Iterative improvement on an initial partition
- Starts with an arbitrary partition
- Find the i -th pair of unlocked vertices residing in different partitions whose exchange results in largest decrease or smallest increase in cut-cost
 - Mark the pair locked, and record the gain g_i
- Find k such that $\sum_{i=1..k} g_i$ is maximized
- If the overall gain is positive, interchange the first k pairs
- Repeat the process until no positive overall gain

Modeling Hypergraphs with Graphs

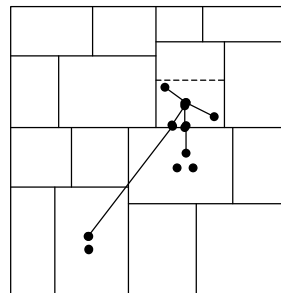
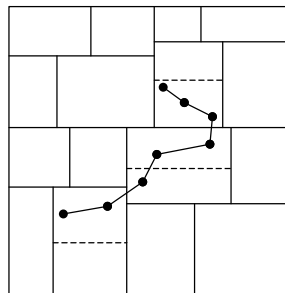
- Net as hypergraph, $H = (N, L)$
 - N : a set of terminals
 - L : a set of hyper-edges, L_i connects a subset N_i of vertices, with $|N_i| > 1$
- Approximation of hypergraphs with complete graphs
 - Weight assignment of each edge

$$\frac{4}{n^2 - \text{mod}(n,2)}$$

- Number of edges cut by bi-partitioning a complete graph of n nodes is at most

$$\frac{n^2 - \text{mod}(n,2)}{4}$$

Terminal Propagation [Dunlop-Kernighan, TCAD'85]



Clustering-Based Placement

- Bottom-up method
- Selecting unplaced components and adding them to a partial placement
- Selection is based on how strongly the unplaced components are connected to the placed components
- Placement is based on how connection cost is reduced

Simulated Annealing

[Kirkpatrick-Gelatt-Vecchi, Science'83]

- Simulation of the annealing process used to temper metals
- Avoids getting trapped in local minimums
- Starts with an initial placement
- Improvements made to initial placement by exchanging blocks
- Moves that decrease cost (C) are always accepted
- Moves that increase cost are accepted with a probability $e^{(-\Delta C/T)}$ depending on temperature T

TimberWolf [Sechen, KAP'88]

- Most widely used placement package for standard cell design
- Move operations:
 - Displacement: randomly place a randomly chosen cell
 - Interchange: exchange two randomly chosen cells
- Temperature-dependent range limiter to restrict the distance over which a cell can move
 - The span decreases logarithmically with the temperature

$$L_{wv}(T) = L_{wv}(T_1) \frac{\log T}{\log T_1}, L_{wh}(T) = L_{wh}(T_1) \frac{\log T}{\log T_1}$$

TimberWolf (Cont'd)

- Cost function is a weighted sum of three components
 - Total wire lengths (or wire spans)
 - Total overlap
 - Actual row length
- Temperature schedule
 - At each temperature, a fixed number of moves per cell is allowed
 - Starts at a very high temperature to accept almost all moves
 - Cooling is represented by

$$T_{i+1} = \mathbf{a}(T)T_i$$

Force-Directed Algorithms

- Number of connection between two modules is related to a force attracting them towards each other

$$F_{ij} = -c_{ij}d_{ij}$$

- c_{ij} is a weighted sum of the nets between the two modules
- d_{ij} is the distance between centers of modules
- Repulsive force between modules to prevent overlapping
- An optimal placement is one that minimizes the sum of the force vectors acting on the modules

Force-Directed Construction

- Module M_j occupy (x_j, y_j)
- Set x-component of the forces acting on M_0 to zero

$$\sum_j F_{0j}^x = \sum_j -c_{0j}d_{0j}^x = 0$$

- Set y-component of the forces acting on M_0 to zero

$$\sum_j F_{0j}^y = \sum_j -c_{0j}d_{0j}^y = 0$$

- If there are no modules with predetermined locations, then a trivial solution is obtained by placing the center of all modules at an arbitrary point

Force-Directed Interchange

- Find a module M with the maximum total force acting on it
- Compute the ideal location (x, y) for M
- Move M to (x, y)
- What about M' that occupies (x, y) originally?
 - Move M' to the original location of M
 - Allow overlap of M and M', hopefully, the violation will be removed later on
- Do not move M too far, consider only its nearest horizontal, vertical, and diagonal neighbors in the direction of desired location

Force-Directed Relaxation

- Similar to force-directed interchange in calculation of force vector
- Move most unstable M to desired location (x, y)
- Module M' that originally occupies (x, y) is moved next
- Stop when a module is moved into an empty slot
- Compute the gain and accept the series of moves only if the placement improves
- Otherwise, reject the series and all components are returned to their previous positions

Force-Directed Pair-wise Interchange

- For every pair of modules calculate the reduction in total force when they are exchanged
- Swap the two modules with the largest reduction
- Lock the two swapped modules and do not consider them in the same iteration

Analytical Approach: Resistive Network

- Cost function is in terms of wire length

$$\Phi(x, y) = \frac{1}{2} \sum_{i,j=1}^n c_{ij} [(x_i - x_j)^2 + (y_i - y_j)^2]$$

- Connectivity matrix $C = [c_{ij}]$
- Indefinite admittance matrix $B = D - C$, where D is a diagonal matrix $d_{ii} = \sum_{j=1..n} c_{ij}$

$$\Phi(x, y) = x^T Bx + y^T B y$$

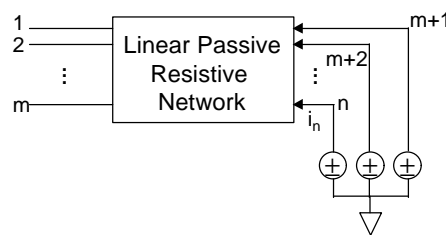
- Only the one-dimensional problem needs to be considered because of the symmetry between x and y

Resistive Network Optimization

- Interpret the objective function $x^T B x$ as the power dissipation of an n -node linear resistive network
- Vector x corresponds to the voltage vector
- $x = [x_1 \ x_2]^T$, x_1 is of dimension m and is to be determined, x_2 is due to the fixed I/O pads
- Placement problem is equivalent to that of choosing voltage vector for which power is a minimum

$$B_{11}x_1 + B_{12}x_2 = 0$$

$$B_{21}x_1 + B_{22}x_2 = i_2$$



Solving a Linear Algebraic Problem

- Solve for $Ax_1 = b$
 $A \equiv B_{11}, b \equiv -B_{12}x_2$
- Successive Over-Relaxation method (generalized Gauss-Seidel method): preserve sparsity, convergence guaranteed as A is real, symmetric, positive definite, and diagonally dominant

$$A = \Lambda(L + I + U)$$

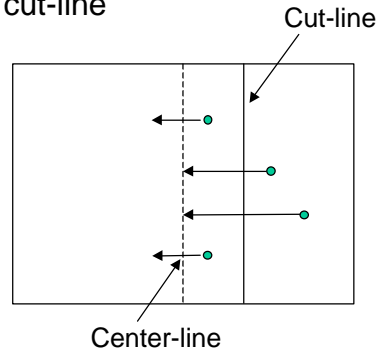
$$x_1(k+1) = Mx_1(k) + a$$

$$M = (I + wL)^{-1}[(1-w)I - wU]$$

$$a = \Lambda^{-1}b$$

Placement and Partitioning

- Linear placement can achieve partitioning
- Add module areas from left to right until roughly half of the total area, that defines cut-line
- Make modules to the right of cut-line fixed, modules to the left of cut-line movable
- Project fixed modules to center-line
- Perform global placement in the left-plane (of center-line)



Placement and Partitioning (Cont'd)

- Make all modules in the left-plane fixed, project to the center line
- Make modules to the right of cut-line movable
- Perform global placement in the right-plane
- Proceed with horizontal cuts on each half
- Continue until each block contains one and only module
- PROUD-2: two-way partitioning in one step
- PROUD-4: four-way partitionings in one step
 - Run-time is 50% longer
 - Wirelength smaller by two to five percent

Mathematical Interpretation

- Equivalent to Block Gauss-Seidel (BGS) method
- Assume a horizontal cut, partition y_1 into y_{1a} , y_{1b}

$$A_{11}y_{1a} + A_{12}y_{1b} = b_{y1}$$

$$A_{21}y_{1a} + A_{22}y_{1b} = b_{y2}$$

$$y_{1a} = A_{11}^{-1} [b_{y1} - A_{12}y_{1b}^*]$$

$$y_{1b} = A_{22}^{-1} [b_{y2} - A_{21}y_{1a}^*]$$

- y_{1a}^* and y_{1b}^* are perturbed solution from y_{1a} and y_{1b} because of the partitioning process

GORDIAN: Quadratic Programming

- Cost function is in terms of wire length

$$\Phi(x, y) = x^T Bx + y^T By$$

- Split modules into movable and fixed, consider only movable modules

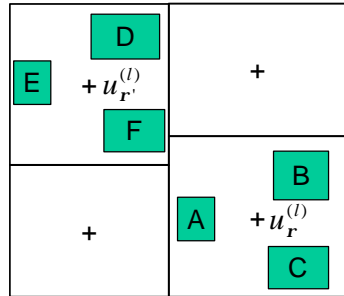
$$\Phi(x, y) = x^T B'x + d_x^T x$$

- Top-down partitioning and placement, use center of region to generate linear constraints for the global placement problem

- At l -th level of optimization, divide the placement area in $q \leq 2^l$ regions

$$A^{(l)}x = u^{(l)}$$

Linear Constraints



$$A^{(l)} = \begin{matrix} \vdots \\ \mathbf{r} \\ \mathbf{r}' \\ \vdots \end{matrix} \begin{bmatrix} A & B & C & D & E & F & G \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ * & * & * & 0 & 0 & 0 & \dots \\ 0 & 0 & 0 & * & * & * & \dots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}$$

$$a_{rm}^{(l)} = \begin{cases} F_u / \sum_{u \in M_r} F_u & \text{if } m \in M_r \\ 0 & \text{otherwise} \end{cases}$$

$$\text{LQP: } \min_x \{ \Phi(x) = x^T B' x + d_x^T x \mid A^{(l)} x = u^{(l)} \}$$

Unconstrained Quadratic Programming

- Linear equality constraint restrict the freedom of movement of modules to a $(m-q)$ -dimensional subspace
- In each region, one (dependent) module has to be moved such that the center-of-gravity constraint is satisfied, the rest (independent) are free to move anywhere

$$x = \begin{bmatrix} x_{d < q >} \\ x_{i < m - q >} \end{bmatrix} \quad A_{< q \times m >}^{(l)} = \begin{bmatrix} D_{< q \times q >} & E_{< q \times m - q >} \end{bmatrix}$$

- D is chosen to be a diagonal matrix, taking biggest entry of each row of A

Unconstrained Quadratic Programming (Cont'd)

$$x_d = -D^{-1}Ex_i + D^{-1}u$$

$$x = Zx_i + x_0$$

$$Z = \begin{bmatrix} -D^{-1}E \\ I \end{bmatrix} \text{ and } x_0 = \begin{bmatrix} D^{-1}u \\ 0 \end{bmatrix}$$

$$\text{UQP: } \min_x \{ \Psi(x) = x_i^T Z^T B' Z x_i + (B' x_0 + d_x)^T Z x_i \}$$

$$\text{Solve for } 2Z^T B' Z x_i^* = -((B' x_0 + d_x)^T Z)^T$$

- $Z^T B' Z$ can be dense, and direct solvers or iterative methods which need the matrix are impractical
- Conjugate-Gradient method is well-suited

Partitioning Schemes

- Use global solution to partition a region evenly
- Trade-off balance partitioning for cut size, also use min cut size to determine cut direction
- Improve partitioning by module interchange
- Repartitioning after each global optimization
 - Large overlap of modules belonging to different son regions indicates a bad partitioning
 - Modules in region ρ migrates to ρ' and from ρ' to ρ
 - Repartition based on new global placement usually results in a better module to region assignment