Lecture 12: Placement

Matthew Guthaus
Professor
UCSC, Computer Science & Engineering
http://vlsida.soe.ucsc.edu
mrg@ucsc.edu

Today's Lecture

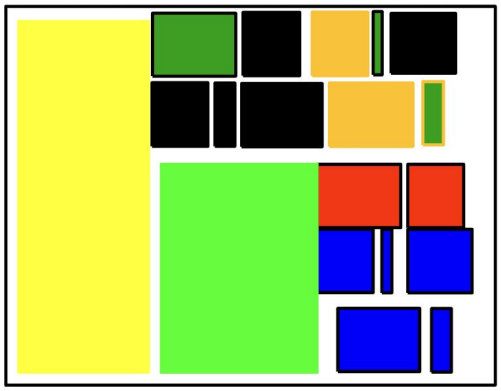
- Good vs Bad Placements
- Partitioning
- A few placement algorithms
  - Recursive Bipartitioning Placement
  - Analytical (Quadratic) Placement
  - Simulated Annealing Placement
- OpenLane Placement

Placement Problem

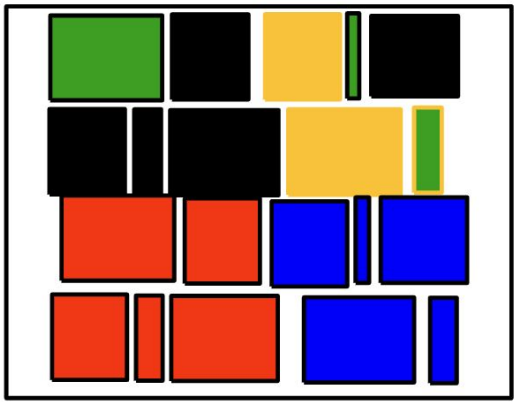
- Input:
  - A set of cells and their complete information (a cell library).
  - Connectivity information between cells (netlist information).
- Output:
  - A set of locations on the chip: one location for each cell.
- Goal:
  - The cells are placed to produce a routable chip that meets timing and other constraints (e.g., low-power, noise, etc.)
- Challenge:
  - The number of cells in a design is very large (> 1 million).
  - The timing constraints are very tight.

Placement Problems

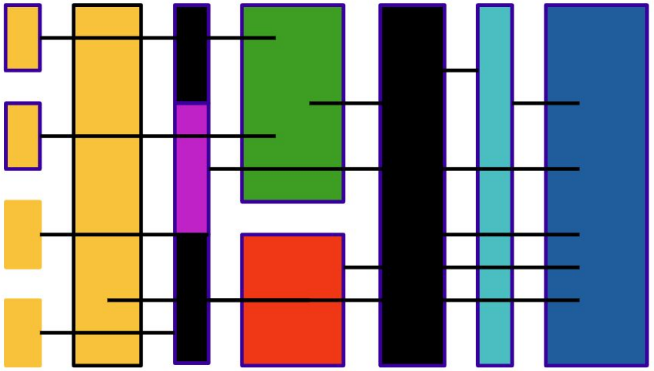
IP - Floorplanning



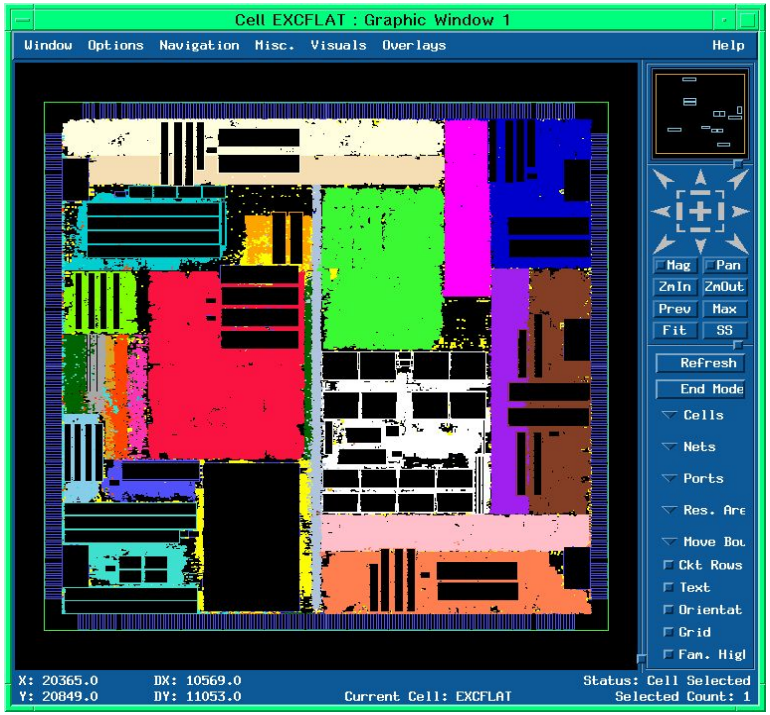
Standard Cell:

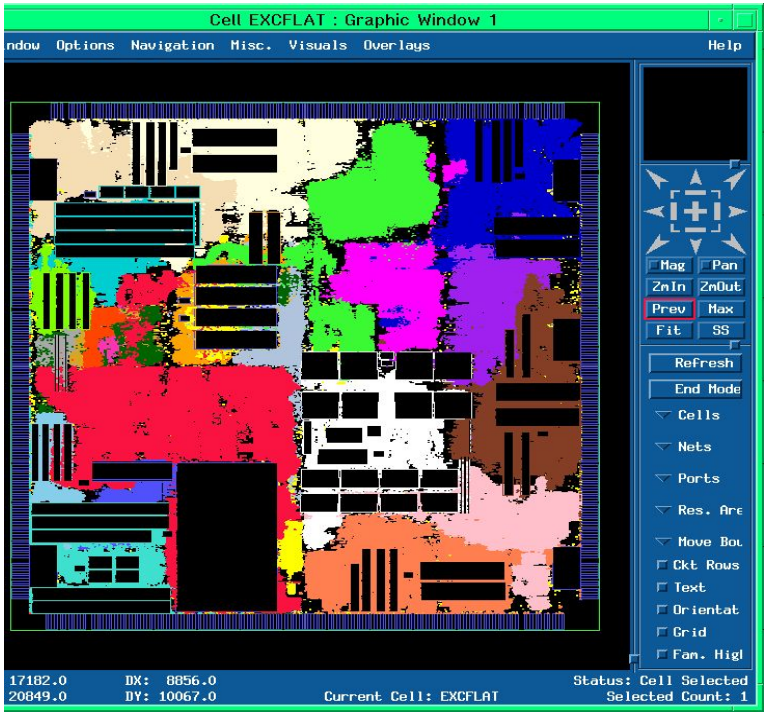


Data Path:



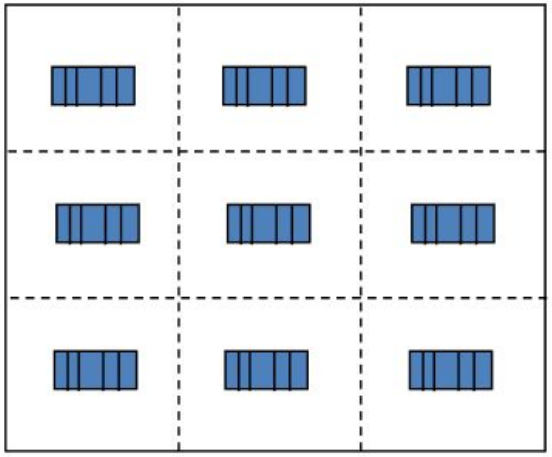
Region Assignment

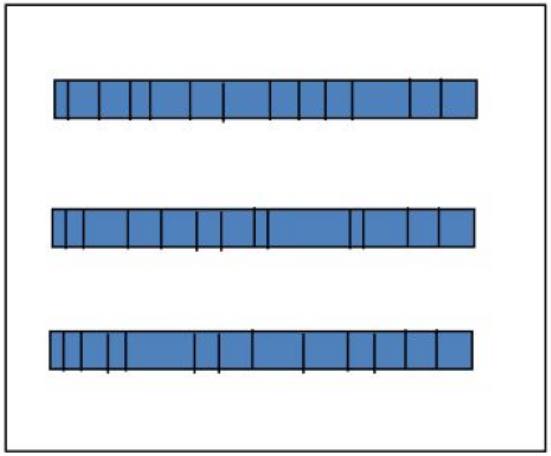




Global and Detailed Placement

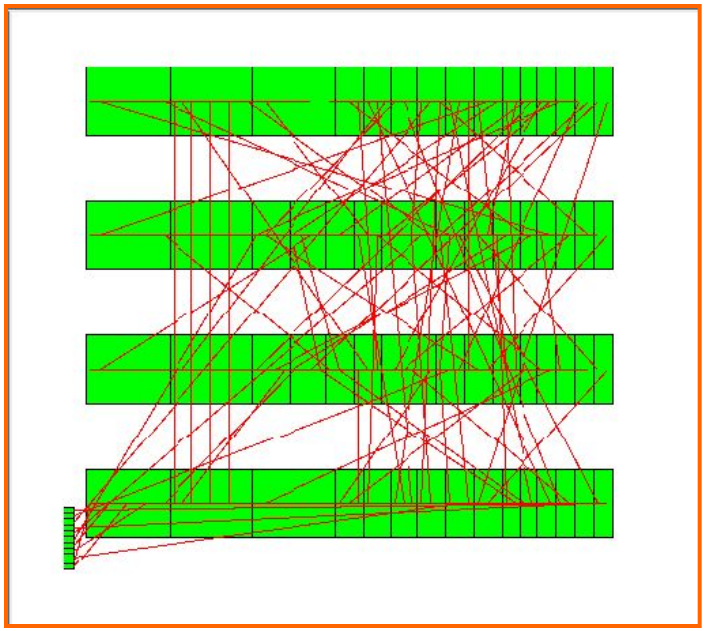
- In global placement, we decide the approximate locations for cells by placing cells in global bins.
  - Overlapping is ok and exact positions are not known
  - **Closely resembles partitioning!**
- In detailed placement, we make some local adjustment to obtain the final non-overlapping placement.
  - Legalization is what determines the final non-overlapping placement



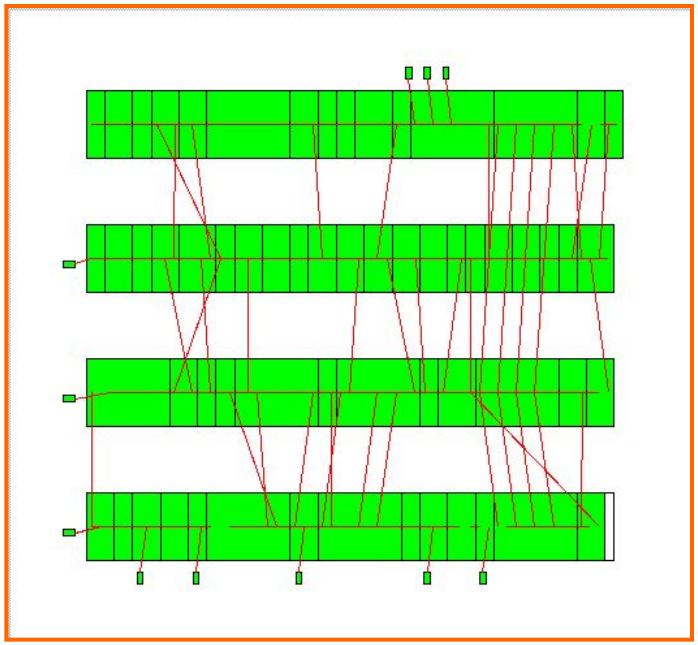


Placement Examples

What makes a good placement?



bad placement



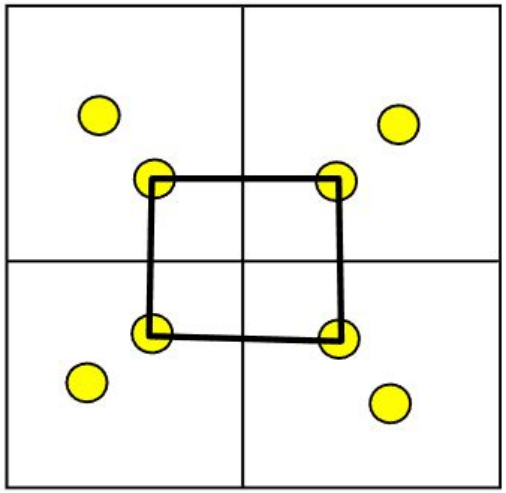
good placement

Metrics: Area & Overlap

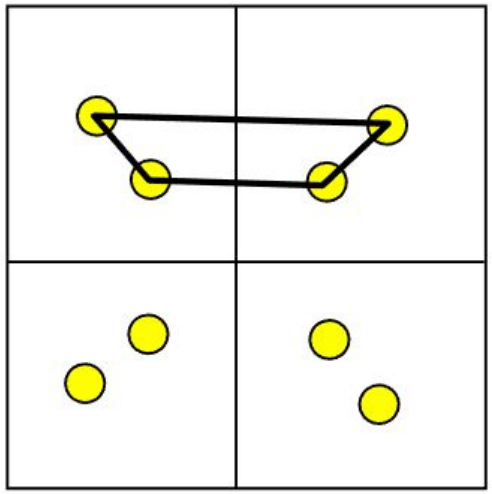
- Area
  - Excessive space between cells will be a waste.
- Overlap
  - Cells must not overlap to have a legal placement. (DRC errors!)

Metrics: Wire Length

- Minimizing wire length may reduce:
  - Delay
  - Total Power
  - Congestion
- But, not really.



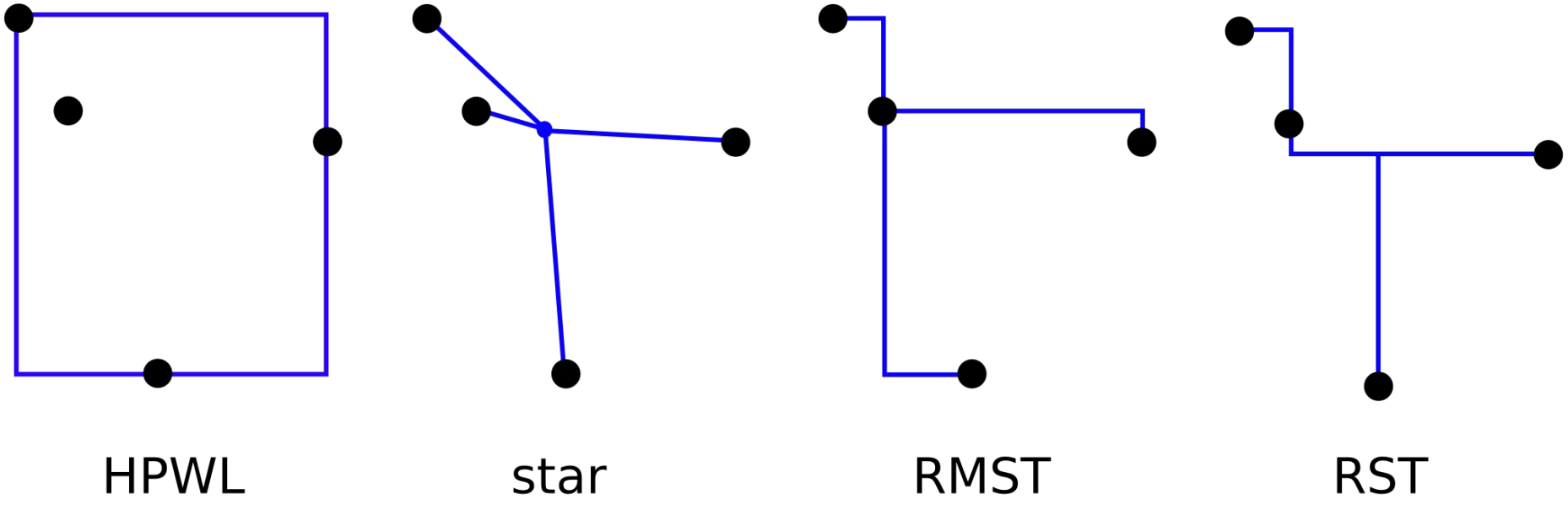
A congestion minimized placement



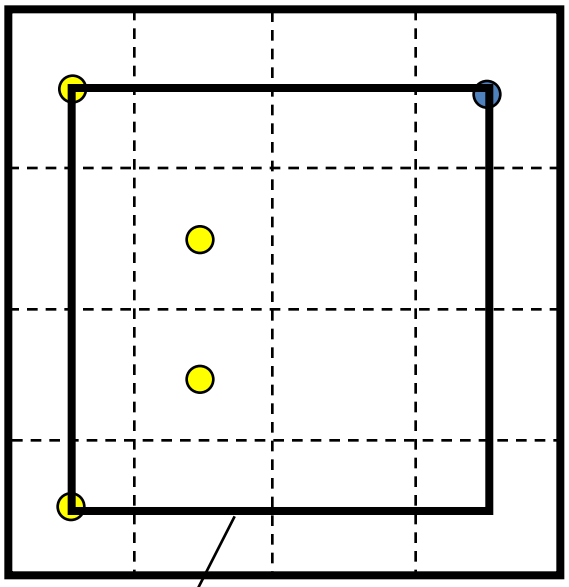
A wirelength minimized placement

Metrics: Wire Length

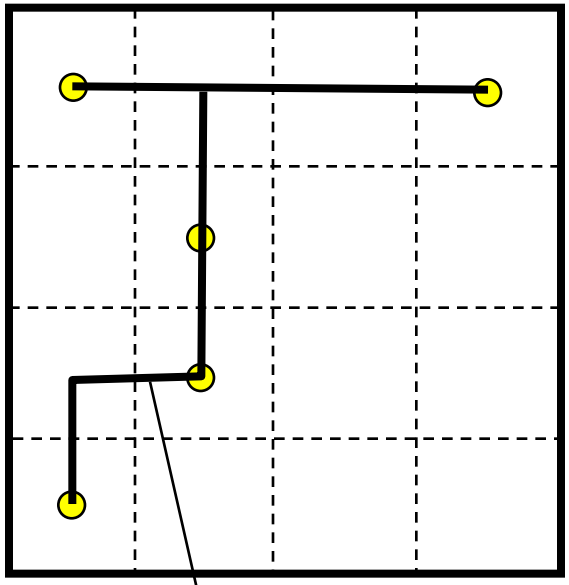
- Routing is complex (more later), so estimate wire lengths
  - HPWL fine for few points
  - More during routing lecture…



Different Routing Models

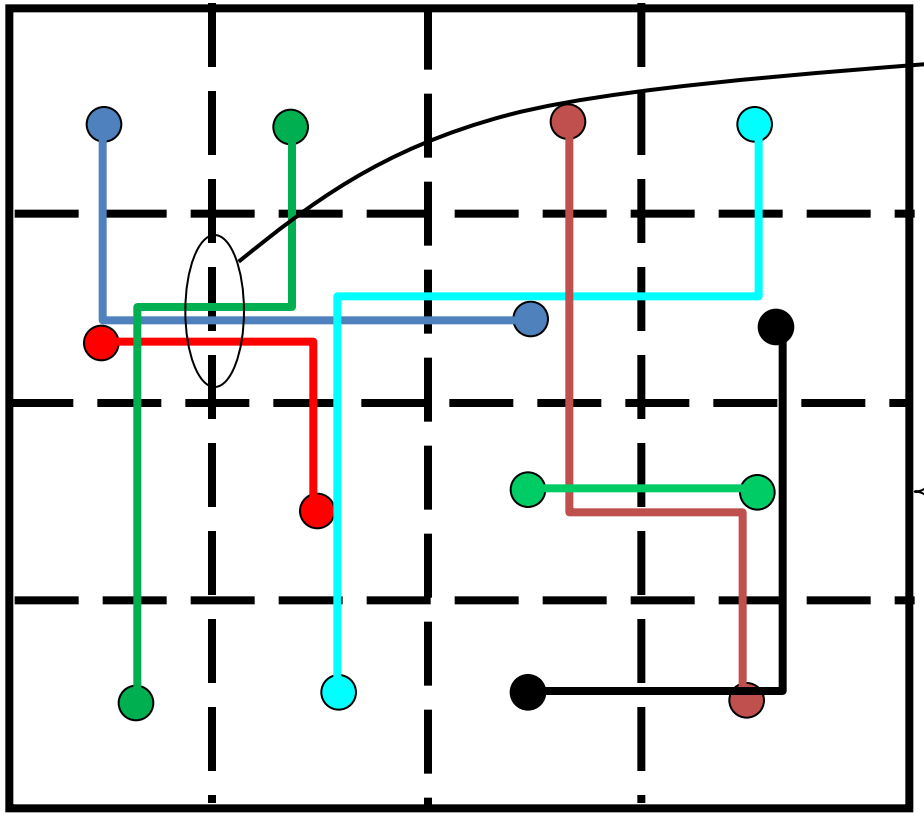


A bounding box routing model



A MST+shortest_path routing model

Metrics: Congestion



Routing demand = 3
Assume routing supply is 1,
overflow = 3 - 1 = 2 on this edge.

Overflow on each edge =

Routing Demand - Routing Supply
(if Routing Demand > Routing Supply)
0 (otherwise)

$$\text{Overflow} = \sum_{\text{all edges}} \text{overflow}$$

Congestion Minimization

- Traditional placement problem is to minimize interconnection length (wirelength)
- A valid placement has to be routable
- Congestion is important because it represents routability (lower congestion implies better routability)
- Congestion is often difficult around certain areas
  - Macro pins
  - Dense pin areas
  - High net to cell ratio

Metrics: Timing

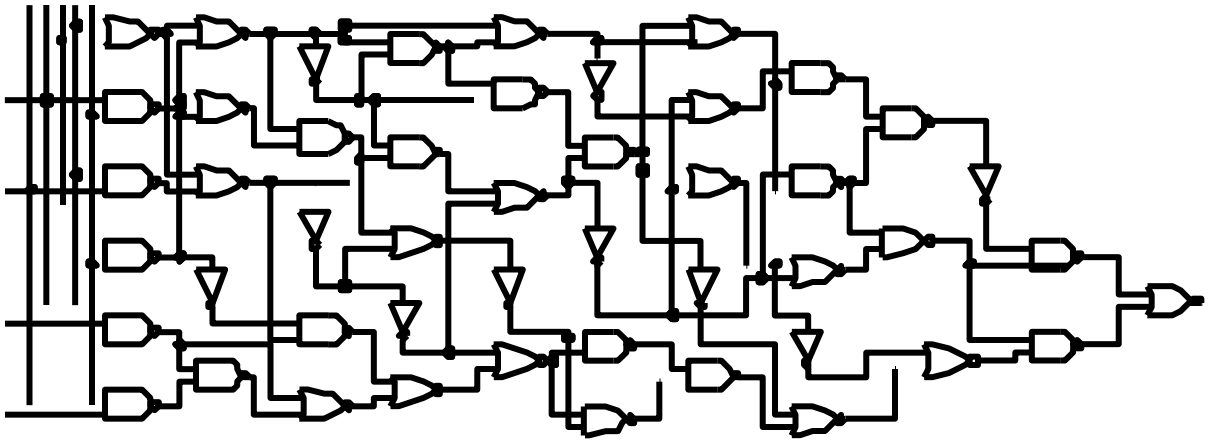
- Perform STA during placement!
  - Incremental placement is useful
  - What about buffering of long nets?
- How to incorporate timing into placement?
  - Chicken and Egg: need placement for timing, but need timing for placement.
  - Usually this is done iteratively with net weights

Closely Related: Partitioning

- Decomposition of a complex system into smaller subsystems
  - Done hierarchically
  - Partitioning done until each subsystem has manageable size
  - Each subsystem can be designed independently
- Interconnections between partitions minimized
  - Less hassle interfacing the subsystems
  - Communication between subsystems usually costly
  - Time-budgeting

Example: Partitioning of a Circuit

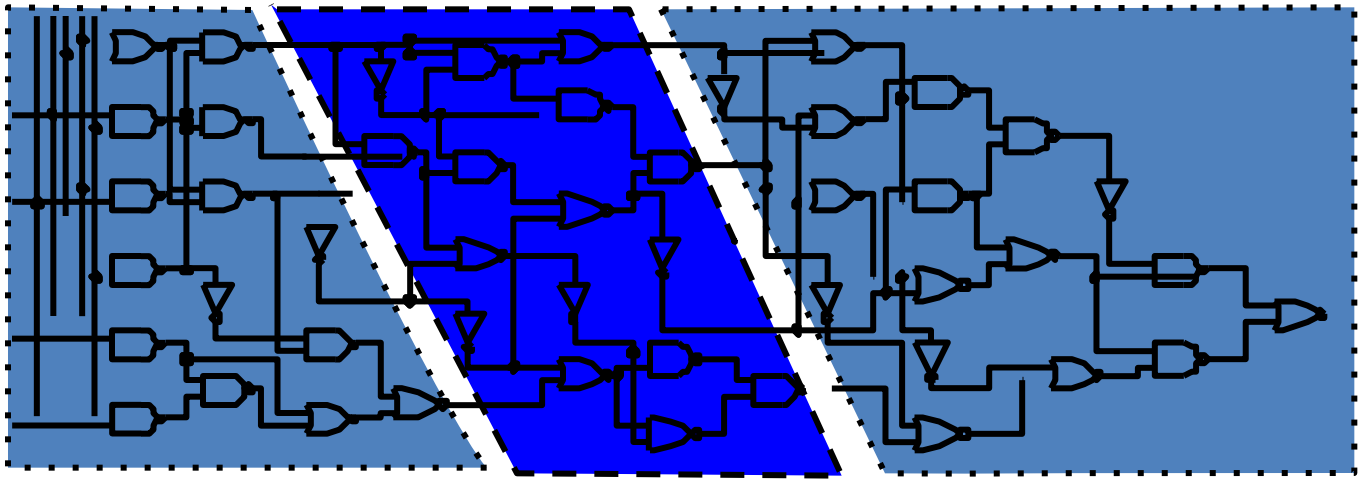
Input size: 48



Cut 1=4
Size 1=15

Cut 2=4
Size 2=16

Size 3=17



Logical Hierarchy vs Partitioning

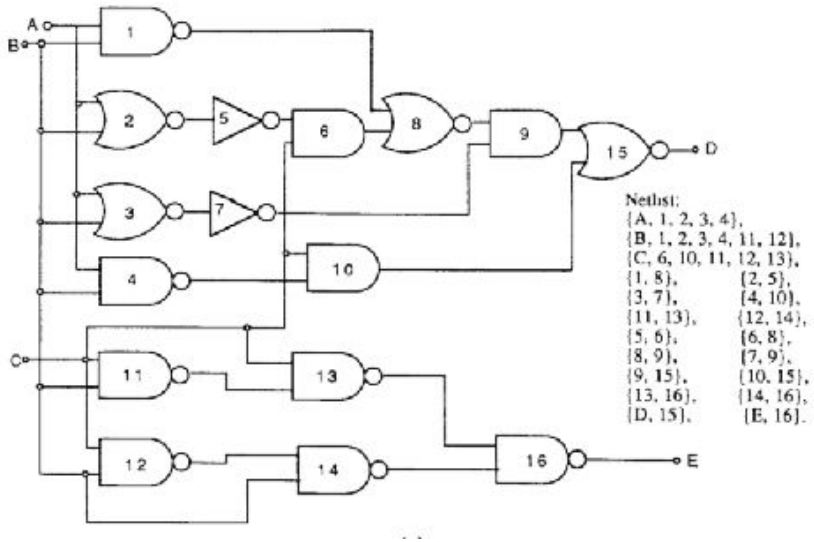
- Sometimes logical hierarchy is not balanced.
  - Partitions based on function, not structure.
  - A few lines of Verilog (e.g. a memory) can be significantly larger which results in unbalanced partitions.
- Logical hierarchy is easier to understand, however.

Partitioning-based Approach

- Try to group closely connected modules together.
- Repetitively divide a circuit into sub-circuits such that the cut value is minimized.
- The placement region is partitioned by cutlines (need not be balanced).
- Each sub-circuit is assigned to one partition of the placement region.

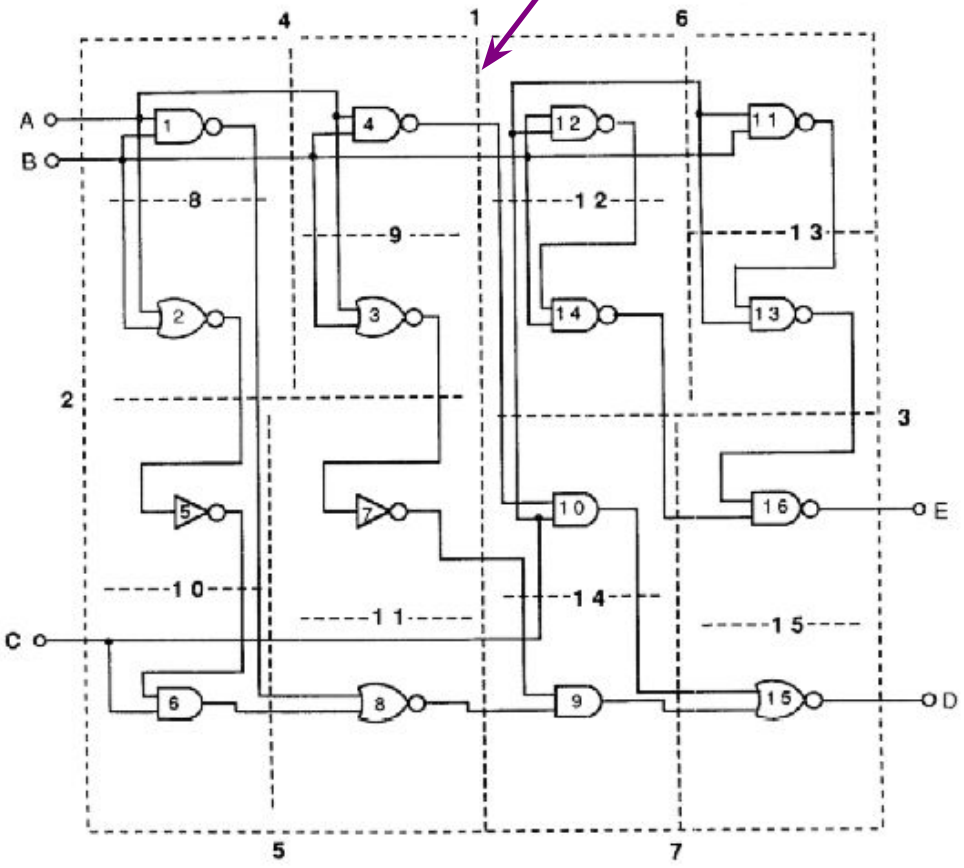
- Note: Also called min-cut placement approach.

An Example



Circuit

Cutline



Placement

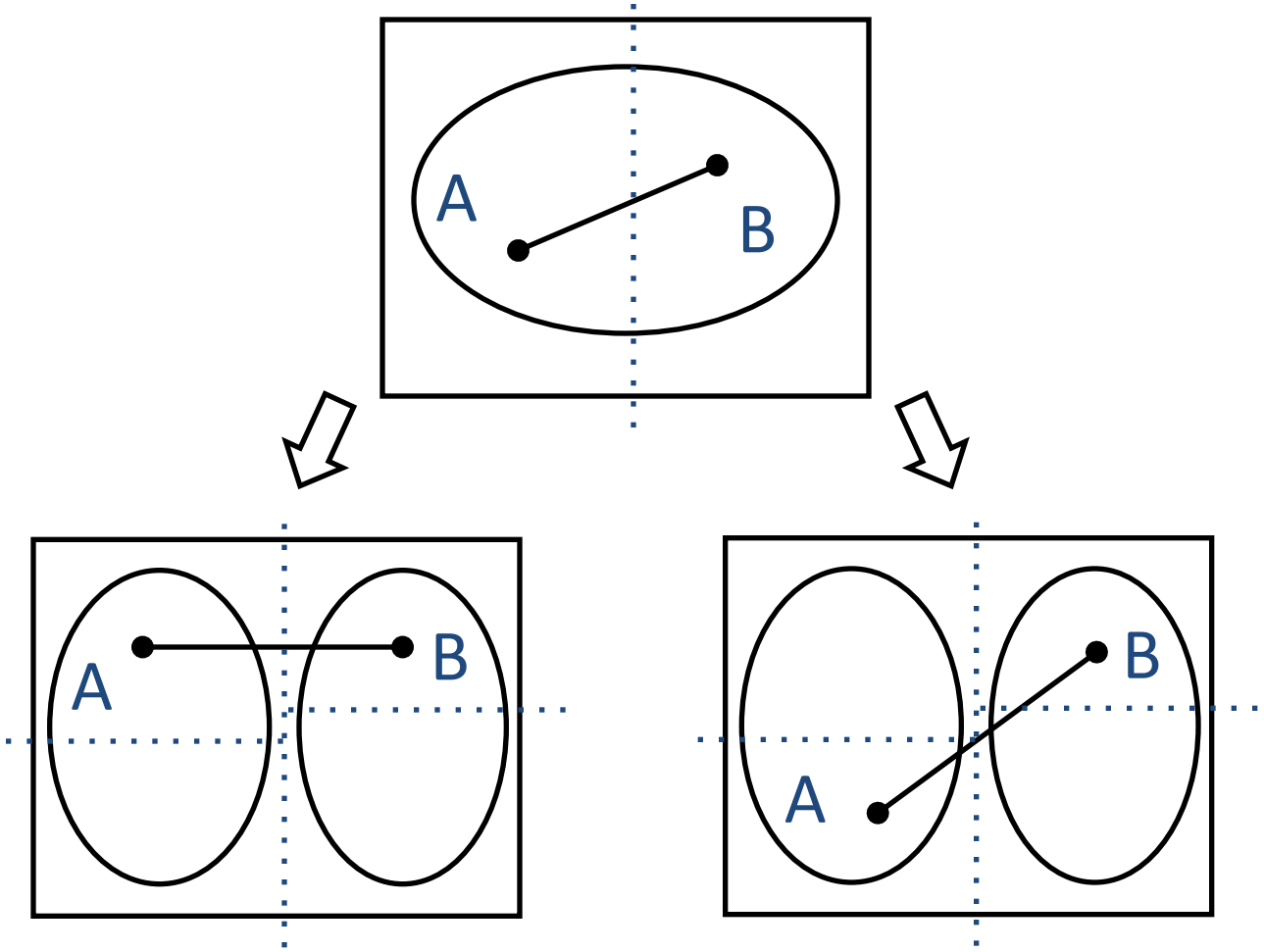
Partitioning Algorithms

Min-cut partitioning is a general CS problem and has very good, fast (polynomial time) solutions

- Fiduccia-Mattheyses (FM) Algorithm
- Kernighan-Lin (KL) Algorithm
- Network Flow
- Spectral Partitioning
- Multilevel Partitioning

Partitioning with other constraints, such as for placement, however, is harder.

Problem of Partitioning Subcircuits



Cost of these 2 partitionings are not the same.

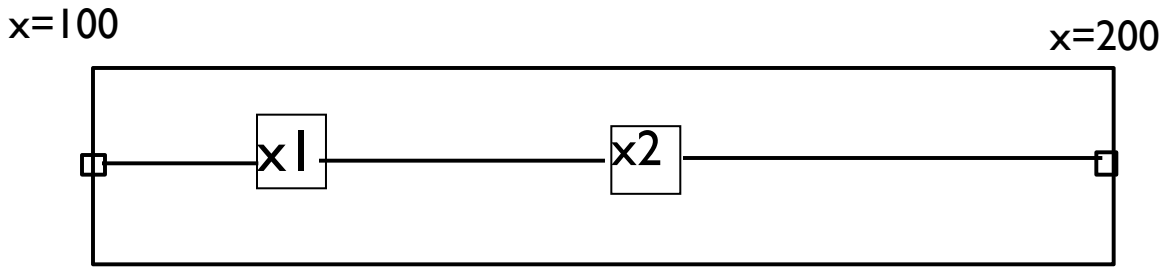
Pros/Cons of Partition-based

- Pros
  - Very fast
- Cons
  - Not stable (different solutions with minor changes) for some algorithms
    - Multiple restart technique
    - How to seed the partition?
  - Timing metric is hard
    - Number of cuts on a given net, not just “is it cut?”

Analytical Placement

- Write down the placement problem as an analytical mathematical problem
- Solve the placement problem directly
- Example:
  - Sum of squared wire length is quadratic in the cell coordinates.
  - So the wirelength minimization problem can be formulated as a quadratic program.
  - It can be proved that the quadratic program is convex, hence polynomial time solvable

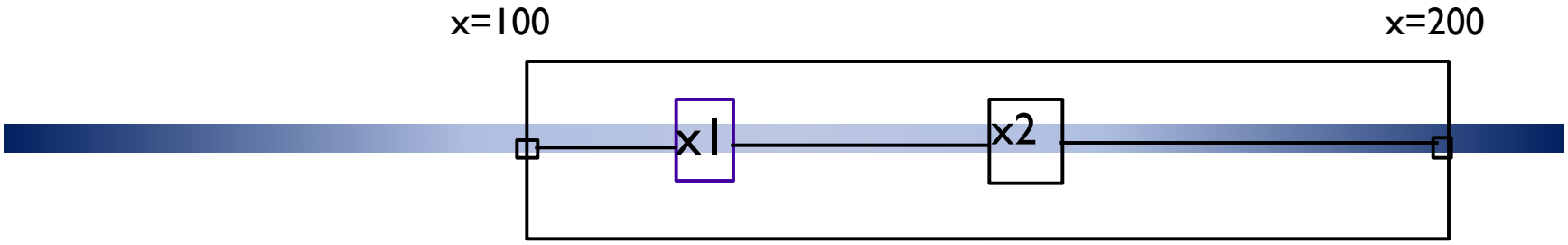
Toy 1-D Example



$$Cost = (x_1 - 100)^2 + (x_1 - x_2)^2 + (x_2 - 200)^2$$

$$\frac{\partial}{\partial x_1} Cost = 2(x_1 - 100) + 2(x_1 - x_2)$$ (Partial w.r.t. x1)

$$\frac{\partial}{\partial x_2} Cost = -2(x_1 - x_2) + 2(x_2 - 200)$$ (Partial w.r.t. x2)



setting the partial derivatives = 0 we solve for the minimum Cost:

$$Ax + B = 0$$

$$\begin{bmatrix} 4 & -2 \\ -2 & 4 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}+\begin{bmatrix} -200 \\ -400 \end{bmatrix} = 0$$

$$\begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}+\begin{bmatrix} -100 \\ -200 \end{bmatrix} = 0$$ (Scaled by ½)

x1=400/3 x2=500/3

Interpretation of matrices A and B:

The diagonal values A[i,i] correspond to the number of connections to xi
The off diagonal values A[i,j] are -1 if object i is connected to object j, 0 otherwise
The values B[i] correspond to the sum of the locations of fixed objects connected to object i

Why formulate the problem this way?

- Because we can
- Because it is trivial to solve
- Because there is only one solution
- Because the solution is a global optimum
- Because the solution conveys “relative order” information
- Because the solution conveys “global position” information

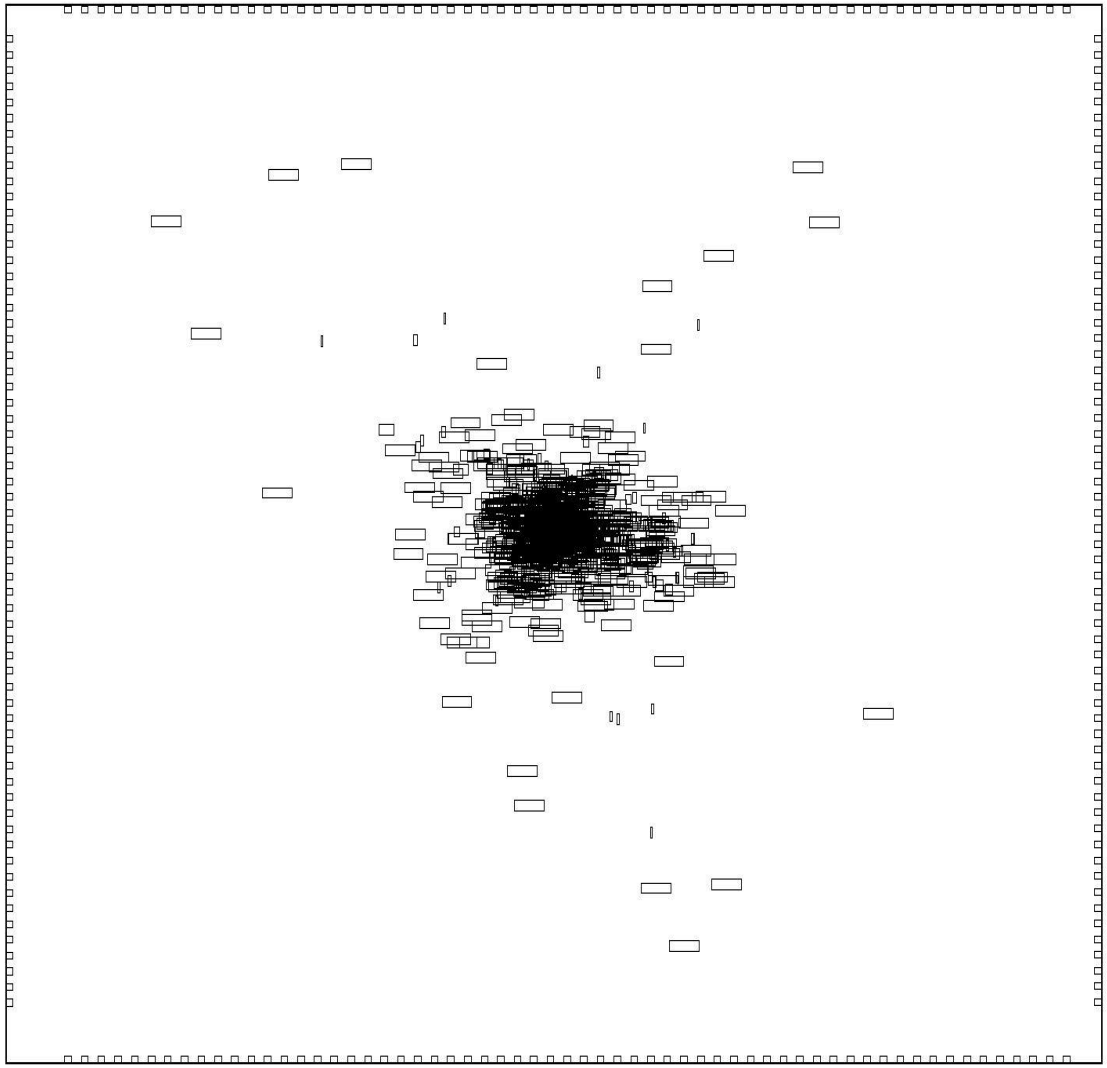
Gordian: A Quadratic Placement Approach

- Global optimization:
  solves a sequence of quadratic programming problems
- Partitioning:
  enforces the non-overlap constraints

***Ref. 1: Gordian: VLSI Placement by Quadratic Programming and slicing Optimization*, by J. M. Kleinhans, G.Sigl, F.M. Johannes, K.J. Antreich IEEE Trans. On CAD, March 1991. pp 356-365**

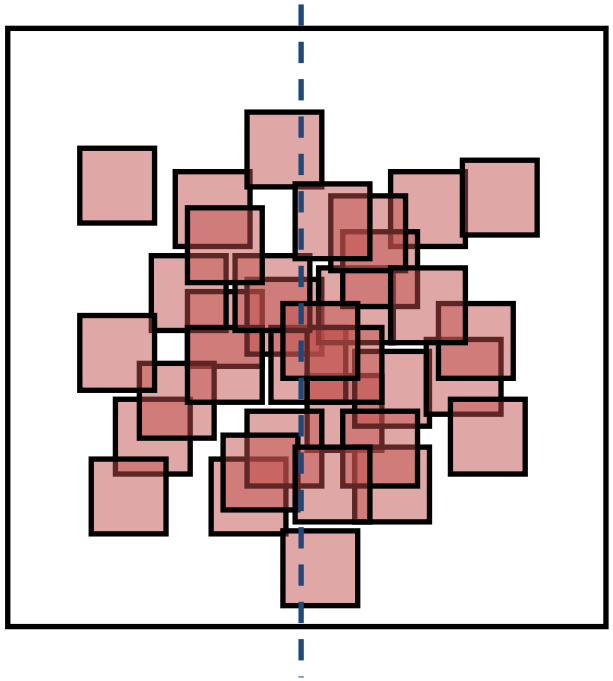
***Ref. 2: Analytical Placement: A Linear or a Quadratic Objective Function? By G. Sigl, K. Doll, F.M. Johannes, DAC'91 pp 427-423**

Solution of the Original QP



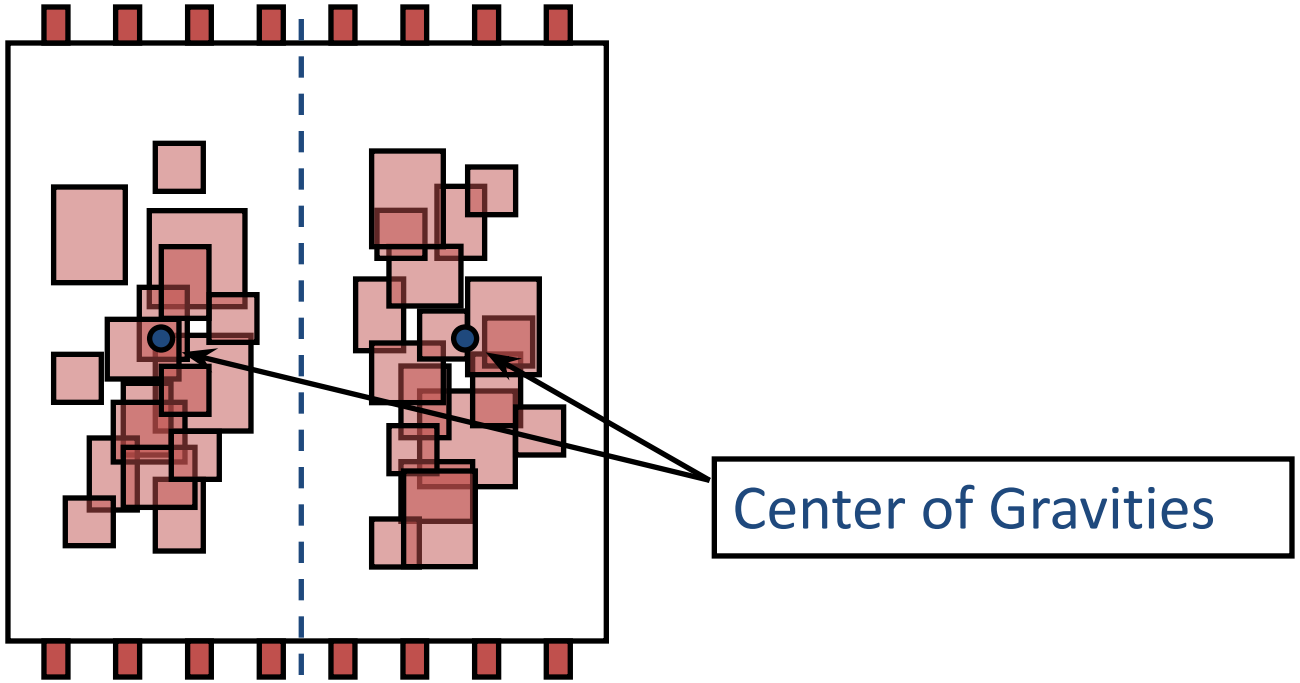
Partitioning

- Find a good cut direction and position.
- Improve the cut value using iterative partitioning.



Applying the Idea Recursively

- Before every level of partitioning, do the Global Optimization again with additional constraints that the center of gravities should be in the center of regions.



- Always solve a single QP (i.e., global).

Pro/Con of Quadratic

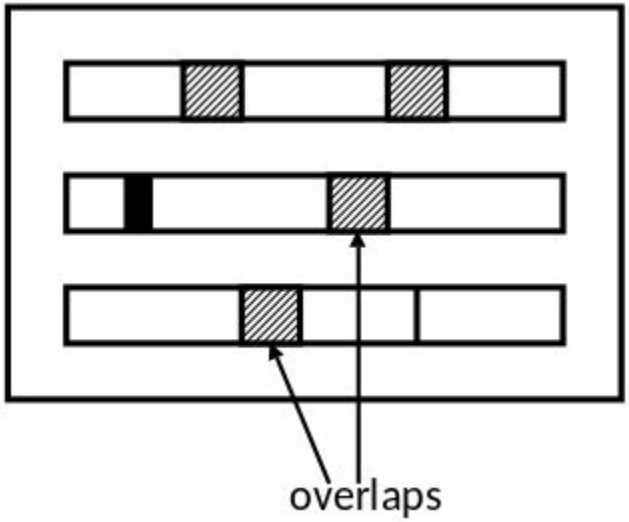
- Pros:
  - mathematically well behaved
  - efficient solution techniques find global optimum
  - great quality

- Cons:
  - solution of Ax + B = 0 is not a legal placement, so generally some additional partitioning techniques are required.
  - solution of Ax + B = 0 is that of the "mapped" problem, i.e., nets are represented as cliques, and the solution minimizes wire length squared, not linear wire length unless additional methods are deployed
  - fixed IOs are required for these techniques to work well
    - but shouldn't the IOs depend on the placement? Chicken and egg.

Simulated Annealing

Solution Space

- All possible arrangements of modules into rows possibly with overlaps
- In practice, can discretize the x locations to a multiple or fraction of the minimum cell width.

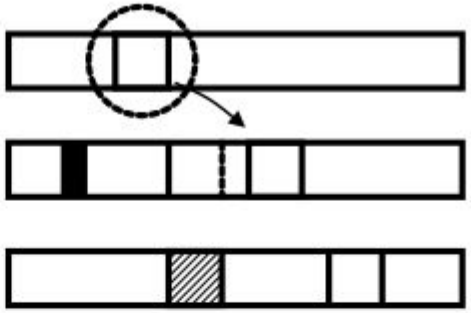


white = cells
black = space
lines = overlap

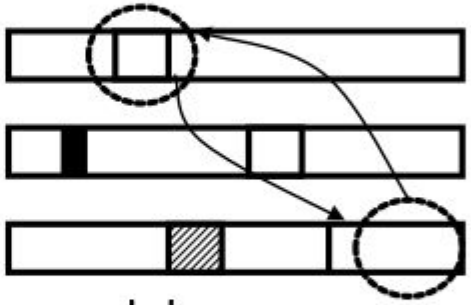
Neighboring Solutions

Three types of moves:

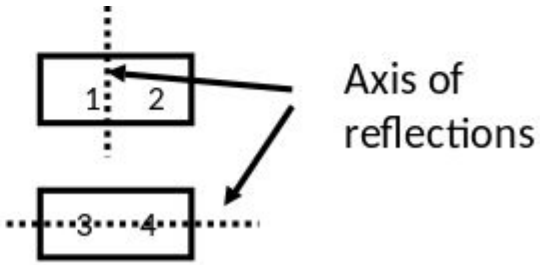
M1: Displace a module to a new location



M2: Interchange two modules



M3: Change the orientation of a module



Move Selection

- Timberwolf first tries to select a move between M1 and M2
  - Prob(M1)=4/5
  - Prob(M2)=1/5

M1: Displacement

M2: Interchange

M3: Reflection

- 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

Annealing Schedule

- $T_k = r(k) \bullet T_{k-1}$ $k = 1, 2, 3, \ldots$
- r(k) increase from 0.8 to max value 0.94 and then decrease to 0.1
- At each temperature, a total number of K•n attempts is made
- n= number of modules
- K= user specified constant

Simulated Annealing Based Placement

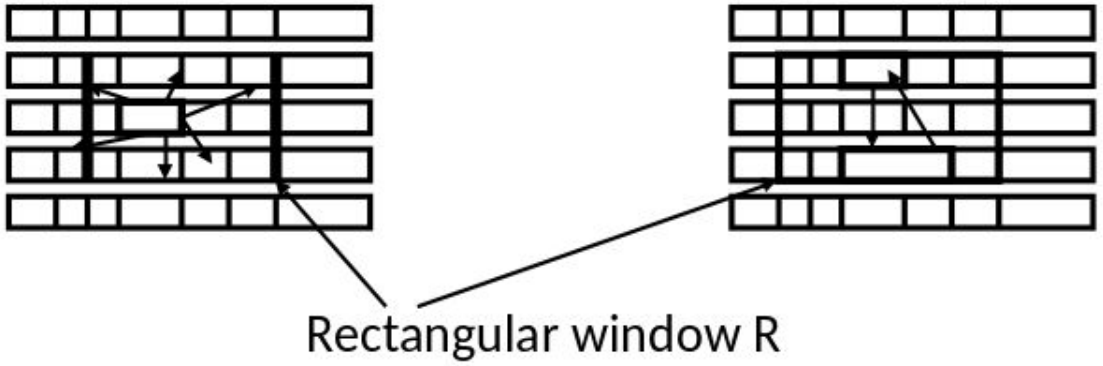
- Timber wolf
- Stage 1
  - Modules are moved between different rows as well as within the same row
  - Module overlaps are allowed
  - when the temperature is reduced below a certain value, stage 2 begins
- Stage 2
  - Remove overlaps
  - Annealing process continues, but only interchanges adjacent modules within the same row

"The Timberwolf Placement and Routing Package", Sechen, Sangiovanni; IEEE Journal of Solid-State Circuits, vol SC-20, No. 2(1985) 510-522

"Timber wolf 3.2: A New Standard Cell Placement and Global Routing Package" Sechen, Sangiovanni, 23rd DAC, 1986, 432-439

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 is so small that no inter-row modules interchanges are possible

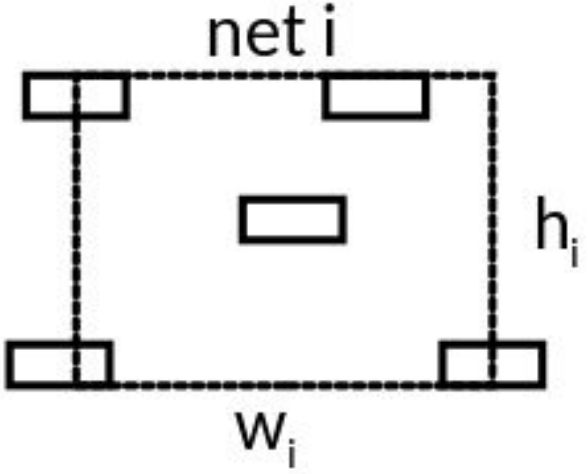


Cost Function

- Critical nets: Increase both αi and βi

Total Cost:

$\Psi = C_1 + C_2 + C_3$



$$C_1 : \sum_i (\alpha_i w_i + \beta_i h_i)$$

α_i, β_i are horizontal and vertical weights, respectively

α_i =1, β_i =1 ⇒1/2 •perimeter of bounding box

Cost Function (Cont'd)

C_2: Penalty function for module overlaps

O(i,j) = amount of overlaps in the X-dimension between modules i and j

$$C_2 = \sum_{i \neq j} (O(i,j) + \alpha)^2$$

α — offset parameter to ensure $C_2 \rightarrow 0$ when $T \rightarrow 0$

C_3: Penalty function that controls the row lengths

Desired row length = d(r)

l(r) = sum of the widths of the modules in row r

$$C_3 = \sum_r \beta \left| l(r) - d(r) \right|$$

Pros/Cons of SA

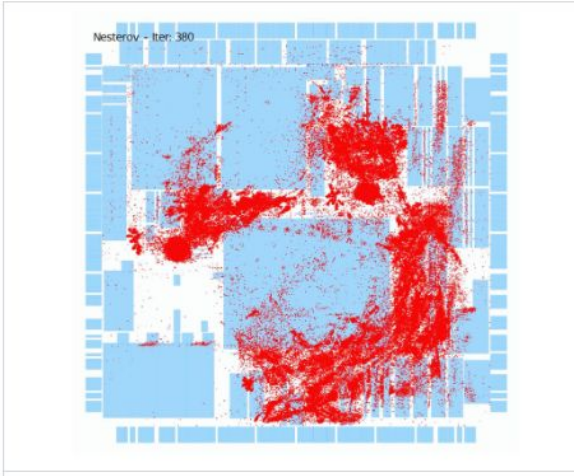
- Pros
  - Any cost function can be handled
  - Can get to optimal solution if run for long enough
- Cons
  - Slow
  - Slow
  - Slow
  - Need to tune the probabilities, moves, etc.

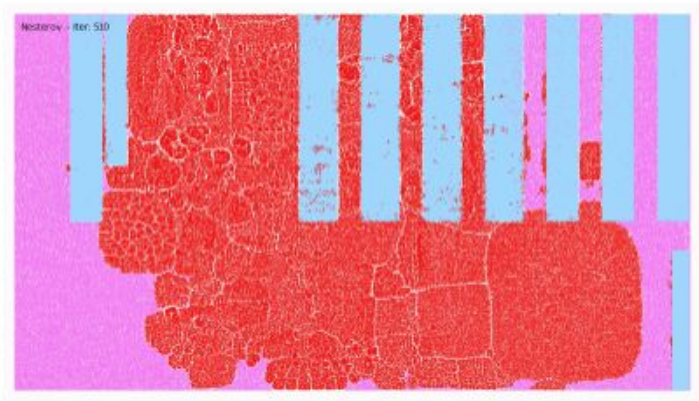
RePlAce (Global Placement)

Analytical Placer

Demo videos:

https://github.com/The-OpenROAD-Project/OpenROAD/blob/master/src/gpl/README.md





Floorplan Options

- FP_SIZING: relative or absolute
  - CORE_AREA: rectangle when absolute mode
  - FP_ASPECT_RATIO: height to width ratio (how square?) for relative
  - FP_CORE_UTIL: area of cells in relative mode
- FP_PIN_ORDER_CFG: File with placement of IOs on sides (N, W, E, S)
- FP_IO_MODE: equidistant or not

Macro Options

PL_MACRO_HALO is an area “halo” around macros (e.g. SRAMs) for routing.

PL_MACRO_CHANNEL is space between macros for routing when macros are automatically placed.

MACRO_PLACEMENT_CFG is an optional config to place macros at absolute locations. RECOMMEND!

(Some) Placement Options

PL_TARGET_DENSITY (range from 1=dense, 0=spread, default is derived from other params)

PL_ROUTABILITY_DRIVEN (on default)

PL_TIME_DRIVEN (on default)

PL_WIRELENGTH_COEF (0.25 default)

PL_RANDOM_GLBL_PLACMENT (off default)

Legalization Distance

PL_MAX_DISPLACEMENT_X (500um default)
PL_MAX_DISPLACEMENT_Y (100um default)

For detailed placement legalization, specifies maximum distance to move cells to remove overlap.

What if you have big macros?
NEED >=500um FOR SRAMs!

PL_RESIZER

Combines gate sizing with placement.

Lots of options for setup vs hold tradeoff.

Suboptimality Visualization

feng shui 5.0 GUI with an image mapped to the optimal placement of PEKO01

Dragon placement

Capo placement

feng shui recursive bisection placement

feng shui structural mode placement

mPL placement

Kraftwerk placement (not legalized)

Recursive Bisection Placement: Feng Shui 5.0 Implementation Details, Agnitori et al, ISPD 2005.

Next Lecture

- See routing