Today

- Placement Problem
- Partitioning → Placement
- Quadrisecion
- Refinement

Placement

- **Problem**: Pick locations for all building blocks
  - minimizing energy, delay, area
  - really:
    - minimize wire length
    - minimize channel density

Bad Placement

- How bad can it be?
  - Area
  - Delay
  - Energy

Bad: Area

- All wires cross bisection
- $O(N^2)$ area
- good: $O(N)$

Bad: Delay

- All critical path wires cross chip
- $\text{Delay} = O(|\text{PATH}| \times 2 \times L_{\text{side}})$
  - [and $L_{\text{side}}$ is $O(N)$]
- good: $O(|\text{PATH}| \times L_{\text{cell}})$
- compare 50ps gates to many nanoseconds to cross chip
Clock Cycle Radius

- Radius of logic can reach in one cycle (45 nm)
  - 1 Cycle Radius = 10
  - Few hundred PEs
- Chip side 600-700 PE
  - 400-500 thousand PEs
  - 100s of cycles to cross

Bad: Energy

- All wires cross chip:
  - $O(L_{\text{side}})$ long → $O(L_{\text{side}})$ capacitance per wire
    - Recall Area $\sim O(N^2)$
    - So $L_{\text{side}} \rightarrow O(N)$
    - $O(N)$ wires → $O(N^2)$ capacitance
  - Good:
    - O(1) long wires → O(N) capacitance

Distance

- Can we place everything close?

Illustration

- Consider a complete tree
  - nand2’s, no fanout
  - N nodes
- Logical circuit depth?
- Circuit Area?
- Side Length?
- Average wire length between nand gates? (lower bound)

“Closeness”

- Try placing “everything” close

Another Example

- Consider a cut size $F(N) > \sqrt{N}$
  - If optimally place all $F(N)$ producers right next to bisection
    - How many cells deep is producer farthest from the bisection?
  - Lower bound on wire length?
Problem Characteristics

• Familiar
  – NP Complete
  – local, greedy not work
  – greedy gets stuck in local minima

Constructive Placement

Basic Idea

• Partition (bisect) to define halves of chip
  – minimize wire crossing
• Recurse to refine
• When get down to single component, done

Adequate?

• Does recursive bisection capture the primary constraints of two-dimensional placement?

Problems

• Greedy, top-down cuts
  – maybe better pay cost early?
• Two-dimensional problem
  – (often) no real cost difference between H and V cuts
• Interaction between subtrees
  – not modeled by recursive bisect

Interaction
Example

Ideal split (not typical)

"Equivalent" split ignoring external constraints
Practically -- makes all H cuts also be V cuts

Interaction

Problem

• Need to keep track of where things are
  – outside of current partition
  – include costs induced by above
• …but don’t necessarily know where things are
  – still solving problem

Improvement: Ordered

• Order operations
• Keep track of existing solution
• Use to constrain or pass costs to next subproblem
• Flow cut
  – use existing in src/sink
  – A nets = src, B nets = sink
• FM: start with fixed, unmovable nets for side-biased inputs
**Improvement: Constrain**

- Partition once
- Constrain movement within existing partitions
- Account for both H and V crossings
- Partition next
  - (simultaneously work parallel problems)
  - easy modification to FM

**Constrain Partition**

Solve AB and CD concurrently.

**Improvement: Quadrisect**

- Solve more of problem at once
- Quadrisect:
  - partition into 4 bins simultaneously
  - keep track of costs all around

**Quadrisect**

- Modify FM to work on multiple buckets
- k-way has:
  - k(k-1) buckets
  - | from | to |
  - quad → 12
- reformulate gains
- update still O(1)

**Quadrisect**

- Cases (15):
  - (1 partition) → 4
  - (2 part) → 6 = (4 choose 2)
  - (3 part) → 4 = (4 choose 3)
  - (4 part) → 1

**Recurse**

- Keep outside constraints
  - (cost effects)
- Don’t know detail place
- Model as at center of unrefined region
Option: Terminal Propagation

- Abstract inputs as terminals
- Partition based upon
- Represent cost effects on placement/refinement decisions

Option: Refine

- Keep refined placement
- Use in cost estimates

Problem

- Still have ordering problem
- Earlier subproblems solved with weak constraints from later
  - (cruder placement estimates)
- Solved previous case by flattening
  - ...but in extreme give up divide and conquer

Iterate

- After solve later problems
- Relax solution
- Solve earlier problems again with refined placements (cost estimates)
- Repeat until converge

Iteration/Cycling

- General technique to deal with phase-ordering problem
  - What order do we perform transformations, make decisions?
  - How get accurate information to everyone
- Still basically greedy

Refinement

- Relax using overlapping windows
- Deal with edging effects
  - Huang&Kahng claim 10-15% improve
    - cycle
    - overlap
Possible Refinement

- Allow unbalanced cuts
  - most things still work
  - just distort refinement groups
  - allowing unbalance using FM quadrisection looks a bit tricky
  - gives another 5-10% improvement

Runtime

- Each gain update still $O(1)$
  - (bigger constants)
  - so, FM partition pass still $O(N)$
- $O(1)$ iterations expected
- assume $O(1)$ overlaps exploited
- $O(\log(N))$ levels

  - Total: $O(N \log(N))$
  - very fast compared to typical annealing
    - (annealing next time)

Quality: Area

<table>
<thead>
<tr>
<th>Case</th>
<th>GOREL</th>
<th>DOMINO</th>
<th>ICAD</th>
<th>GOREL</th>
<th>DOMINO</th>
</tr>
</thead>
<tbody>
<tr>
<td>pin1</td>
<td>19635</td>
<td>19651</td>
<td>19208</td>
<td>2.8%</td>
<td>-1.9%</td>
</tr>
<tr>
<td>pin2</td>
<td>45549</td>
<td>44716</td>
<td>44407</td>
<td>1.3%</td>
<td>-1.8%</td>
</tr>
<tr>
<td>loc1</td>
<td>44554</td>
<td>44716</td>
<td>44407</td>
<td>1.3%</td>
<td>-1.8%</td>
</tr>
<tr>
<td>loc2</td>
<td>112500</td>
<td>104672</td>
<td>107908</td>
<td>10.5%</td>
<td>2.7%</td>
</tr>
<tr>
<td>rect1</td>
<td>4578</td>
<td>383</td>
<td>380</td>
<td>0.0%</td>
<td>0.8%</td>
</tr>
<tr>
<td>C1108</td>
<td>1834</td>
<td>1757</td>
<td>1630</td>
<td>1.5%</td>
<td>-3.5%</td>
</tr>
<tr>
<td>C15315</td>
<td>6220</td>
<td>5922</td>
<td>6185</td>
<td>0.6%</td>
<td>-4.4%</td>
</tr>
<tr>
<td>C8288</td>
<td>8794</td>
<td>8259</td>
<td>8312</td>
<td>6.5%</td>
<td>0.3%</td>
</tr>
<tr>
<td>s1453</td>
<td>3344</td>
<td>2508</td>
<td>2555</td>
<td>2.0%</td>
<td>-2.0%</td>
</tr>
<tr>
<td>s1458</td>
<td>5058</td>
<td>5508</td>
<td>2470</td>
<td>7.5%</td>
<td>8.3%</td>
</tr>
<tr>
<td>s378</td>
<td>8600</td>
<td>8152</td>
<td>8208</td>
<td>4.2%</td>
<td>-0.3%</td>
</tr>
<tr>
<td>s7231</td>
<td>14685</td>
<td>14623</td>
<td>13674</td>
<td>6.7%</td>
<td>1.3%</td>
</tr>
<tr>
<td>s15450</td>
<td>37092</td>
<td>35591</td>
<td>33025</td>
<td>9.3%</td>
<td>5.5%</td>
</tr>
<tr>
<td>stuart</td>
<td>4196</td>
<td>3967</td>
<td>4160</td>
<td>-0.9%</td>
<td>-5.8%</td>
</tr>
<tr>
<td>biomed</td>
<td>34777</td>
<td>33712</td>
<td>33787</td>
<td>2.8%</td>
<td>-0.2%</td>
</tr>
<tr>
<td>avg</td>
<td>35498</td>
<td>35254</td>
<td>35867</td>
<td>-0.5%</td>
<td>-3.8%</td>
</tr>
<tr>
<td>best</td>
<td>100500</td>
<td>97925</td>
<td>97108</td>
<td>-1.3%</td>
<td>-4.2%</td>
</tr>
</tbody>
</table>

Quality: Delay

- Weight edges based on criticality
  - Periodic, interleaved timing analysis

<table>
<thead>
<tr>
<th>Case</th>
<th>Measure</th>
<th>Max Intrinsic Path Delay</th>
<th>TW7.0 Timing-QUAD</th>
</tr>
</thead>
<tbody>
<tr>
<td>tract</td>
<td>Delay</td>
<td>MST100</td>
<td>10.6</td>
</tr>
<tr>
<td>struct</td>
<td>Delay</td>
<td>MST100</td>
<td>40.9</td>
</tr>
<tr>
<td>avg</td>
<td>Delay</td>
<td>MST100</td>
<td>37.3</td>
</tr>
</tbody>
</table>

Uses

- Good by self
- Starting point for simulated annealing
  - speed convergence
- With synthesis (both high level and logic)
  - get a quick estimate of physical effects
  - (play role in estimation/refinement at larger level)
- Early/fast placement
  - before willing to spend time looking for best
- For fast placement where time matters
  - FPGAs, online placement?

Summary

- Partition to minimize cut size
- Additional constraints to do well
  - Improving constant factors
- Quadrisection
- Keep track of estimated placement
- Relax/iterate/Refine
Admin

• Reading for Wednesday
• Assignment 4 out
  – Placement (and clustering) for timing

Big Ideas:

• Potential dominance of interconnect
• Divide-and-conquer
• Successive Refinement
• Phase ordering: estimate/relax/iterate