CS184a: Computer Architecture (Structures and Organization)

Day12: November 1, 2000
Interconnect Requirements and Richness

Last Time

- Dominance of Interconnect
- Simple things
  - and why they don’t work
- Characterizing Interconnect Requirements
  - start
Today

• Followups from Monday (3)
• Interconnect Design Space
• Characterizing Interconnect Requirements
• Interconnect Implications
• How rich should interconnect be
  – specifics of understanding interconnect
  – methodology for attacking these kinds of questions

Tree Cut

• Bisection bandwidth
  – binary: 1
  – general: \( \log(n) \)
• Rent IO Cut
  – IO\( \sim K/2 \times N \)
  – \( P=1 \)
• Difference:
  – include input
Resource Bounded Scheduling

- Last time: pointed out can get lower bound on time (upper bound on performance)

- Scheduling in general NP-hard
  - (find optimum)
  - can approximate in O(E) time

Lower Bound: Critical Path

- ASAP schedule ignoring resource constraints
  - (look at length of remaining critical path)

- Certainly cannot finish any faster than that
Lower Bound: Resource Capacity

- Sum up all capacity required per resource
- Divide by total resource (for type)
- Lower bound on remaining schedule time
  – (best can do is pack all use densely)

Example

Critical Path
Resource Bound (2 resources)
Resource Bound (4 resources)
Example 2

RB = $8/2 = 4$
LB = 5
best delay = 6

Example 3

LB = 3
RB = $13/2 = 7$
best delay = 7
Good Model?

Log-log plot ==> straight lines represent geometric growth

Rent’s Rule

- Long standing empirical relationship
  - IO = C*N^P
  - 0 ≤ P ≤ 1.0
  - compare (F,α)-bifurcator
    - α = 2^P

- Captures notion of locality
  - some signals generated and consumed locally
  - reconvergent fanout
Rent and Locality

- Rent and IO capture locality
  - local consumption
  - local fanout

Resuming...
Rent’s Rule

• Typically consider
  – $0.5 \leq P \leq 0.75$
• “High-Speed” Logic $P=0.67$
• Memory ($P \sim 0.1-0.2$)
• Example (i10)
  – max $C=7$, $P=0.68$
  – avg $C=5$, $P=0.72$

What tell us about design?

• Recursive bandwidth requirements in network
What tell us about design?

• Recursive bandwidth requirements in network
  – lower bound on resource requirements

• N.B. **necessary** but not **sufficient** condition on network design
  – *i.e.* design must also be able to *use* the wires

• Interconnect lengths
  – Intuition
    • if \( p > 0.5 \), everything cannot be nearest neighbor
    • as \( p \) grows, so wire distances
What tell us about design?

- Interconnect lengths
  - \( \text{IO} = (n^2)^p \) cross distance \( n \)
  - \( \frac{d\text{IO}}{dn} \) end at exactly distance \( n \)
  - \( E(I) = \int_0^n \sqrt{N} \)
    - of \( n\cdot\left(\frac{d\text{IO}}{dn}\right)n^2 \)
    - assume iid sources
  - \( E(I) = O(N^{(p-0.5)}) \)
    - \( p > 0.5 \)

N.B. 2D VLSI world has “natural” Rent of \( P=0.5 \)
(area vs. perimeter)
Rent’s Rule Caveats

• Modern “systems” on a chip -- likely to contain subcomponents of varying Rent complexity
• Less I/O at certain “natural” boundaries
• System close
  – (Rent’s Rule apply to workstation, PC, PDA?)

Area/Wire Length

• Bad news
  – Area ~ O(N^{2p})
    • faster than N
  – Avg. Wire Length ~ O(N^{(p-0.5)})
    • grows with N
• Can designers/CAD control p (locality) once appreciate its effects?
• I.e. maybe this cost changes design style/criteria so we mitigate effects?
What Rent didn’t tell us

- Bisection bandwidth purely geometrical
- No constraint for delay
  - *I.e.* a partition may leave critical path weaving between halves

Critical Path and Bisection

Minimum cut may cross critical path multiple times. Minimizing long wires in critical path => increase cut size.
Rent Weakness

• Not account for path topology

• ? Can we define a “Temporal” Rent which takes into consideration?
  – Promising research topic

Administrative Interlude

• …won’t catchup today + lots more stuff
• No Class Wed 11/8
• Can we meet Friday 11/10?

• Homework 3+4 graded
• P/F
  – (reluctantly) …if you must
  – must attempt all (>90%) problems to get passing grade
Interconnect Richness

Now What?

• There is structure (locality)
• Rent characterizes locality

• How rich should interconnect be?
  – Allow full utilization?
  – Model requirements and area impact
Step 1: Build Architecture Model

- Assume geometric growth
- Pick parameters: Build architecture can tune
  - F, C
  - \( \alpha, p \)

Tree of Meshes

- Tree
- Restricted internal bandwidth
- Can match to model
Parameterize C

\[(2 1)^* \Rightarrow \alpha = \sqrt{2}\]

\[(2 2 1)^* \Rightarrow \alpha = (2^2)^{1/3} = 2^{2/3}\]

Parameterize Growth

\[(2 2 1)^* \Rightarrow \alpha = 2^{(3/4)}\]
Wednesday class stopped here

Step 2: Area Model

- Need to know effect of architecture parameters on area (costs)
  - focus on dominant components
    - wires
    - switches
    - logic blocks(?)
Area Parameters

- $A_{\text{logic}} = 40K\lambda^2$
- $A_{\text{sw}} = 2.5K\lambda^2$
- Wire Pitch = $8\lambda$

Switchbox Population

- Full population is excessive (next week?)
- Hypothesis: linear population adequate
  - still to be (dis)proven
“Cartoon” VLSI Area Model

(Example artificially small for clarity)

Larger “Cartoon”

1024 LUT Network

P=0.67

LUT Area 3%
Effects of P (α) on Area

P=0.5  P=0.67  P=0.75

1024 LUT Area Comparison

Effects of P on Capacity
Step 3: Characterize Application Requirements

- Identify representative applications.
  - Today: IWLS93 logic benchmarks
- How much structure there?
- How much variation among applications?

Application Requirements

Max: C=7, P=0.68    Avg: C=5, P=0.72
Benchmark Wide

Benchmark Parameters
Complication

- Interconnect requirements vary among applications
- Interconnect richness has large effect on area
- What is effect of architecture/application mismatch?
  – Interconnect too rich?
  – Interconnect too poor?
Step 4: Assess Resource Impact

- Map designs to parameterized architecture
- Identify architectural resource required

**Compare**: mapping to k-LUTs; LUT count vs. k.

Mapping to Fixed Wire Schedule

- Easy if need less wires than Net
- If need more wires than net, must depopulate to meet interconnect limitations.
Mapping to Fixed-WS

- Better results if “reassociate” rather than keeping original subtrees.

Observation

- Don’t really want a “bisection” of LUTs
  - subtree filled to capacity by either of
    - LUTs
    - root bandwidth
  - May be profitable to cut at some place other than midpoint
    - not require “balance” condition
  - “Bisection” should account for both LUT and wiring limitations
Challenge

• Not know where to cut design into
  – not knowing when wires will limit subtree capacity

Brute Force Solution

• Explore all cuts
  – start with all LUTs in group
  – consider “all” balances
  – try cut
  – recurse
Brute Force

- Too expensive
- Exponential work

…viable if solving same subproblems

Simplification

- Single linear ordering
- Partitions = pick split point on ordering
- Reduce to finding cost of [start,end] ranges (subtrees) within linear ordering
- Only $n^2$ such subproblems
- Can solve with dynamic programming
Dynamic Programming

- Start with base set of size 1
- Compute all splits of size n, from solutions to all problems of size n-1 or smaller
- Done when compute where to split 0,N-1

Dynamic Programming

- Just one possible “heuristic” solution to this problem
  - not optimal
  - dependent on ordering
  - sacrifices ability to reorder on splits to avoid exponential problem size

- Opportunity to find a better solution here...
Ordering LUTs

- Another problem
  - lay out gates in 1D line
  - minimize sum of squared wire length
    - tend to cluster connected gates together
  - Is solvable mathematically for optimal
    - Eigenvector of connectivity matrix

- Use this 1D ordering for our linear ordering
Step 5: Apply Area Model

- Assess impact of resource results
Picking Network Design Point

<table>
<thead>
<tr>
<th>Minimize Objective</th>
<th>params</th>
<th>Sigma rel area</th>
<th>LUT Util.</th>
</tr>
</thead>
<tbody>
<tr>
<td>area relative area</td>
<td>6</td>
<td>1.23</td>
<td>0.87</td>
</tr>
<tr>
<td>area with full util</td>
<td>10</td>
<td>2.98</td>
<td>1.00</td>
</tr>
</tbody>
</table>

Don’t optimize for 100% compute util. (100% yield) also don’t optimize for highest peak.
What about a single design?

LUT Utilization predict Area?

C=6.0 Lsize Area Comparison

Single design
Methodology

• Architecture model (parameterized)
• Cost model
• Important task characteristics
• Mapping Algorithm
  – Map to determine resources
• Apply cost model
• Digest results
  – find optimum (multiple?)
  – understand conflicts (avoidable?)

Big Ideas

[MSB Ideas]

• Rent’s rule characterize locality
• => Area growth $O(N^{2p})$
• $p>0.5$ => interconnect growing faster than compute elements
  – expect interconnect to dominate other resources
Big Ideas
[MSB Ideas]

• Interconnect area dominates logic area
• Interconnect requirements vary
  – among designs
  – within a single design
• To minimize area
  – focus on using dominant resource
    (interconnect)
  – may underuse non-dominant resources (LUTs)

Big Ideas
[MSB Ideas]

• Two different resources here
  – compute, interconnect
• Balance of resources required varies among designs (even within designs)
• Cannot expect full utilization of every resource
• Most area-efficient designs may waste some compute resources (cheaper resource)