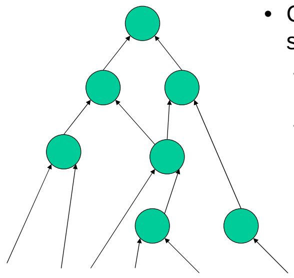
ESE535: Electronic Design Automation

Day 15: March 18, 2009
Static Timing Analysis
and Multi-Level Speedup

Today

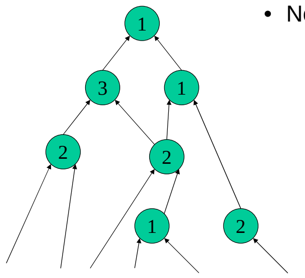
- Topological Worst Case
  - not adequate (too conservative)
- Sensitization Conditions
- Timed Calculus
- Delay-justified paths
  - Timed-PODEM
- Speedup

Topological Worst-Case Delay



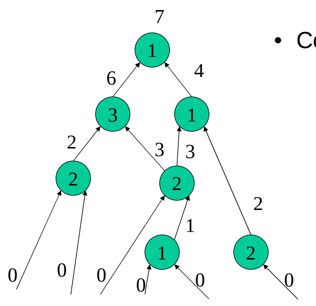
- Compute ASAP schedule
  - Take max of arrival times
  - Apply node Delay

Topological Worst-Case Delay



- Node Delays

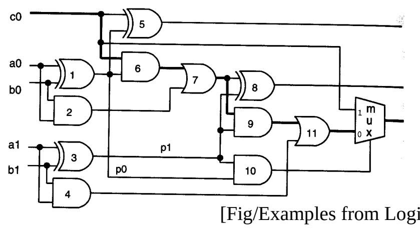
Topological Worst-Case Delay



- Compute Delays

Conservative

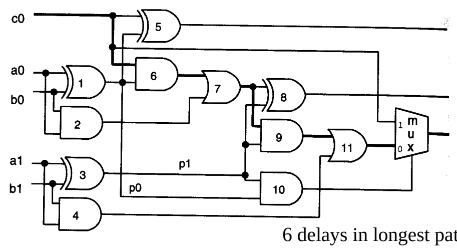
- Topological Worst-Case Delay can be conservative



[Fig/Examples from Logic Synthesis by Devadas, Gosh, Keutzer 1994]

Example

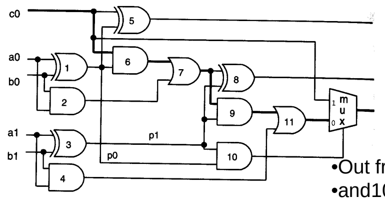
- Assume each gate 1:



6 delays in longest path
(5 if assume c0 latest arriving)

Example

- Is this path possible?



•Out from mux 0 input
•and10 = 0
•p0=0 or p1=0
•p1=0 1→6→7 not matter
•p0=0 c0 not matter
•This path not feasible

False Paths

- Once consider logic for nodes
  - There are logical constraints on data values
- There are paths which cannot logically occur
  - Call them ***false paths***

What can we do?

- Need to assess what paths are real
- Brute force
  - for every pair of inputs
  - compute delay in outputs from in1→in2 input transition
  - take worst case
- Expensive:
  - 2^{2n} delay traces

Alternately

- Look at single vector and determine what controls delay of circuit
  - I.e. look at values on path and determine path *sensitized* to change with input

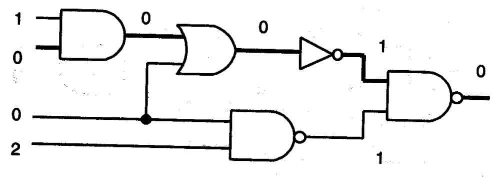
Controlled Inputs

- Controlled input to a gate:
  - input whose value will determine gate output
  - e.g.
    - 0 on a AND gate
    - 1 on a OR gate

Static Sensitization

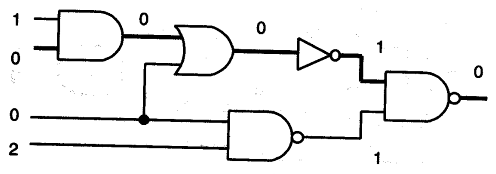
- A path is statically sensitized
  - if all the side (non-path) inputs are non-controlling
  - I.e. this path value flips with the input

Statically Sensitized Path

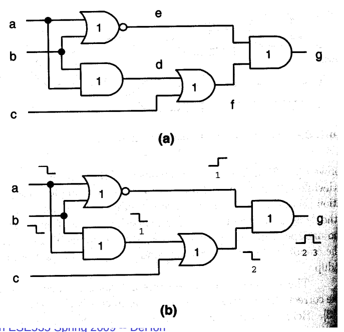


Sufficiency

- Static Sensitization is **sufficient** for a path to be a **true** path in circuit



…but not necessary

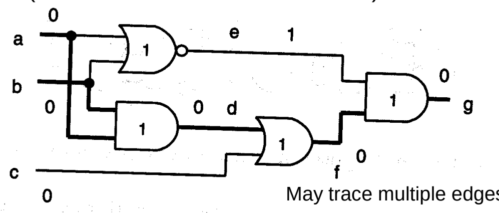


Paths of length 3 not statically sensitizable.

But there is a true path of delay 3.

Static Co-sensitization

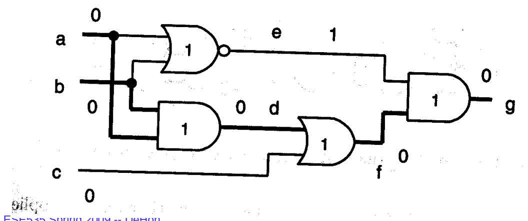
- Each output with a controlled value
  - has a controlling value as input on path
  - (and vice-versa for non-controlled)



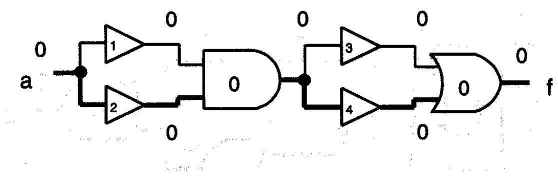
May trace multiple edges

Necessary

- Static Co-sensitization is a **necessary** condition for a path to be true



…but not sufficient



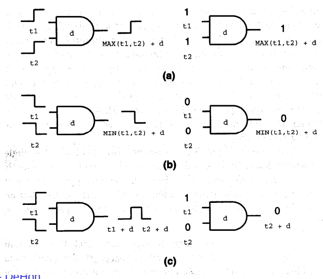
Cosensitize path of length 6.
Real delay is 5.

Combining

- Combine these ideas into a timed-calculus for computing delays for an input vector

Computing Delays

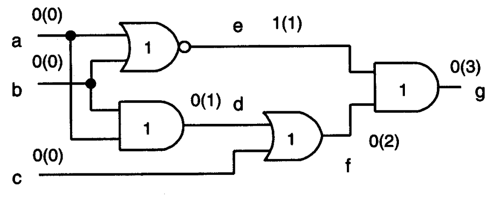
AND
Timing
Calculus



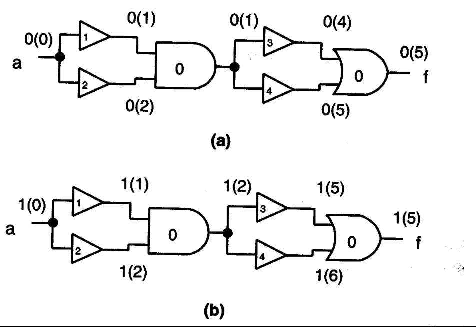
Rules

- If gate output is at a controlling value, pick the minimum input and add gate delay
- If gate output is at a non-controlling value, pick the maximum input and add gate delay

Example (1)



Example (2)



Now...

- We know how to get the delay of a single input condition
- Could:
  - find critical path
  - search for an input vector to sensitize
  - if fail, find next path
  - …until find longest true path
- May be $O(2^n)$

25

Better Approach

- Ask if can justify a delay greater than T
- Search for satisfying vector
  - …or demonstration that none exists
- Binary search to find tightest delay

26

Delay Computation

- Modification of a testing routine
  - used to justify an output value for a circuit
  - similar to SAT
- PODEM
  - backtracking search to find a suitable input vector associated with some target output
  - Simply a branching search with implication pruning
    - Heuristic for smart variable ordering

27

Search

- Takes two lists
  - outputs to set; inputs already set
- Propagate values and implications
- If all outputs satisfied → succeed
- Pick next PI to set and set value
  - Search with this value set
  - If inconsistent
    - If PI not implied
      - Invert value of PI
      - Search with this value set
      - If inconsistent → fail
      - Else succeed
    - Else fail
  - Else succeed

28

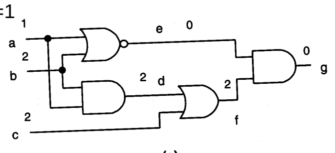
Picking next variable to set

- Follow back gates w/ unknown values
  - sometimes output dictate input must be
    - (AND needing 1 output; with one input already assigned 1)
  - sometimes have to guess what to follow
    - (OR with 1 output and no inputs set)
    - Uses heuristics to decide what to follow

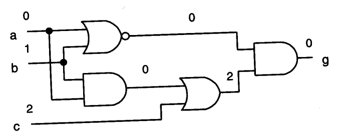
29

Example

Try justify g=1



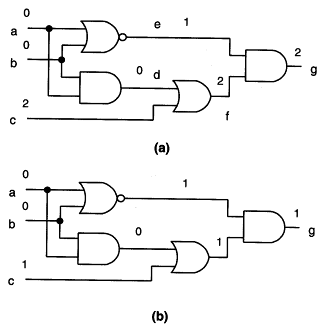
(a)



(b)

30

Example

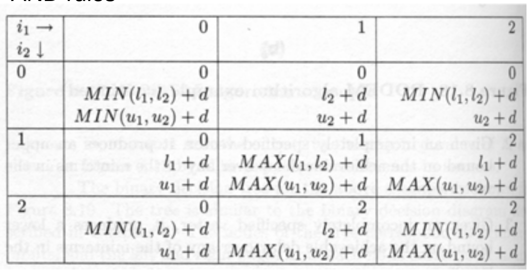


For Timed Justification

- Also want to compute delay
  - on incompletely specified values
- Compute bounds on timing
  - upper bound, lower bound
  - Again, use our timed calculus
    - expanded to unknowns

Delay Calculation

AND rules



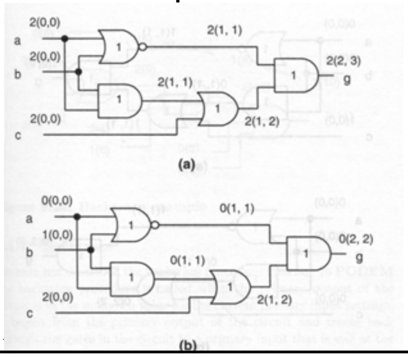
$i_1 \rightarrow$ $i_2 \downarrow$	0	1	2
0	0 $MIN(l_1,l_2)+d$ $MIN(u_1,u_2)+d$	0 l_2+d u_2+d	0 $MIN(l_1,l_2)+d$ u_2+d
1	0 l_1+d u_1+d	1 $MAX(l_1,l_2)+d$ $MAX(u_1,u_2)+d$	2 l_1+d $MAX(u_1,u_2)+d$
2	0 $MIN(l_1,l_2)+d$ u_1+d	2 l_2+d $MAX(u_1,u_2)+d$	2 $MIN(l_1,l_2)+d$ $MAX(u_1,u_2)+d$

Timed PODEM

- **Input:** value to justify and delay T
- **Goal:** find input vector which produces value and exceeds delay T
- Algorithm
  - similar
  - implications check timing as well as logic

Example

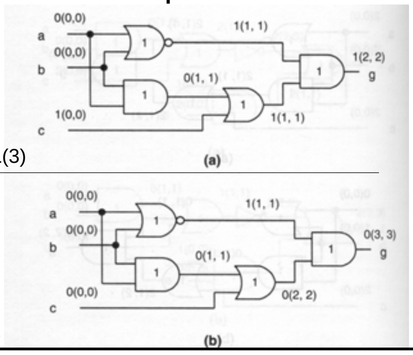
Justify 1(3)



Example

Fail to justify 1(3)

Justify 0(3)



Search

- Less than 2^n
  - pruning due to implications
  - here saw a must be 0
    - no need to search 1xx subtree

Questions

- On static timing analysis?

Speed Up

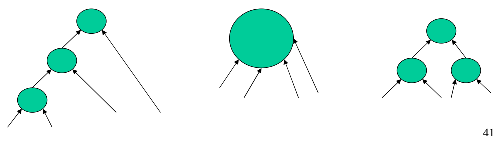
(sketch flavor)

Speed Up

- Start with area optimized network
- Know target arrival times
  - Know delay from static analysis
- Want to reduce delay of node

Basic Idea

- Improve speed by:
  - Collapsing node(s)
  - Refactoring collapsed subgraph to reduce height

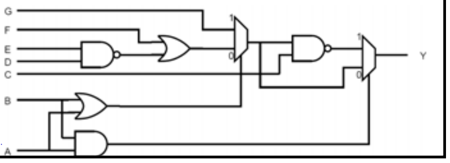


Speed Up

- While (delay decreasing, timing not met)
  - Compute delay (slack)
    - Static timing analysis
  - Generate network close to critical path
    - w/in some delay ε, to some distance d
  - Weight nodes in network
    - Less weight = more potential to improve, prefer to cut
  - Compute **mincut** of *nodes* on weighted network
  - For each node in cutset
    - Partial collapse
  - For each node in cutset
    - Timing redecompose

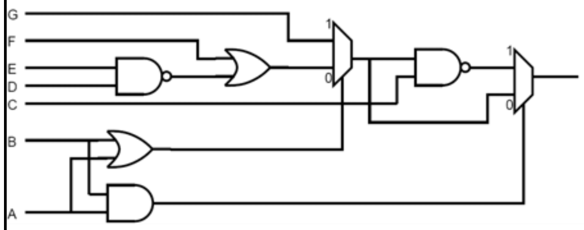
MinCut of Nodes

- Cut nodes not edges
  - Typically will need to transform to dual graph
    - All edges become nodes, nodes become edges
  - Then use maxflow/mincut

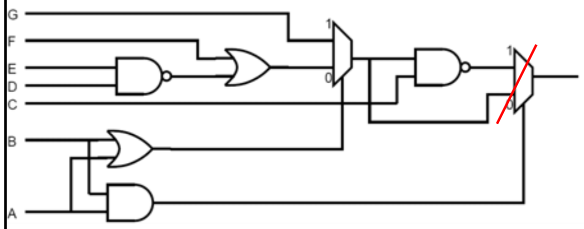


MinCut of Nodes

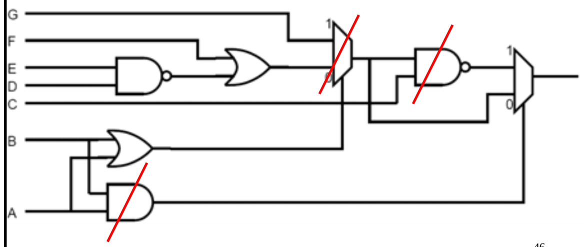
- What are possible cuts?



MinCut of Nodes



MinCut of Nodes

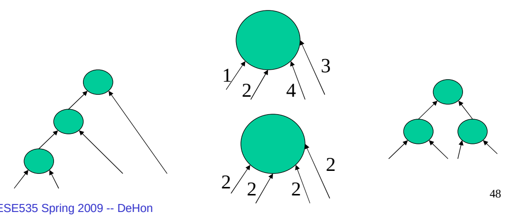


Weighted Cut

- $W=W_t+\alpha W_a$ → α tuning parameter
- Want to minimize area expansion
  - Things in collapsed network may be duplicated
  - *E.g.* W_a=literals in duplicated logic
- Want to maximize likely benefit
  - Prefer nodes with varying input times to the "near critical path" network
    - Quantify: large variance in arrival times
  - Prefer nodes with critical path on longer paths

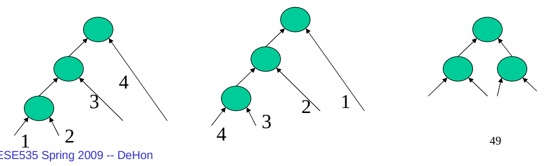
Weighing Benefit

- Want to maximize likely benefit
  - Prefer nodes with varying input times to the "near critical path" network



Weighing Benefit

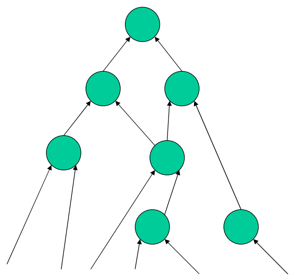
- Want to maximize likely benefit
  - Prefer nodes with critical path on longer paths



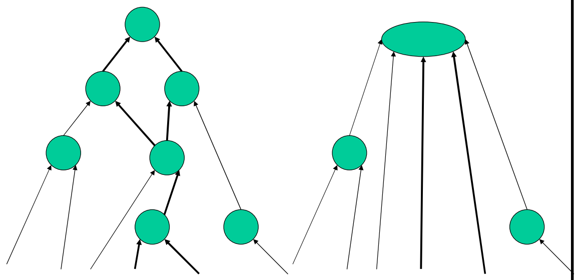
Timing Decomposition

- Extract area saving kernels that do not include critical inputs to node
  - f=abcd+abce+abef (last time)
  - Kernels={cd+ce+ef,e+d,c+f}
  - F=abe(c+f)+abcd, ab(cd+ce+ef), abc(e+d)+abef
  - What if:
    - Critical input is f? d? a? {a,d}?
- When decompose (*e.g.* into nand2's) similarly balance with critical inputs closest to output

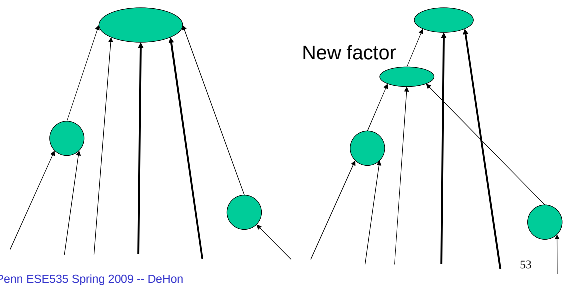
Example



Example



Example



Speed Up (review)

- While (delay decreasing, timing not met)
  - Compute delay (slack)
    - Static timing analysis
  - Generate network close to critical path
    - w/in some delay ε, to some distance d
  - Weight nodes in network
    - Less weight = more potential to improve, prefer to cut
  - Compute **mincut** of *nodes* on weighted network
  - For each node in cutset
    - Partial collapse
  - For each node in cutset
    - timing redecompose

Admin

- Reading
  - Monday (handout today)
- Next week
  - Lecture Monday
  - No lecture Wednesday → work on assignment
  - No office hours → ask questions about problems 1 and 2 before Monday

Big Ideas

- Topological Worst-case delays are conservative
  - Once consider logical constraints
  - may have false paths
- Necessary and sufficient conditions on true paths
- Search for paths by delay
  - or demonstrate non existence
- Search with implications
- Iterative improvement