

CS137: Electronic Design Automation

Day 14: February 27, 2002
Routing 2
(Pathfinder)



CALTECH CS137 Winter2002 -- DeHon

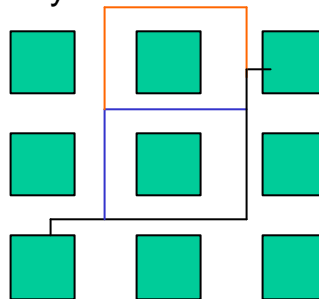
Today

- Routing
 - Pathfinder
 - graph based
 - global routing
 - simultaneous global/detail

CALTECH CS137 Winter2002 -- DeHon

Global Routing

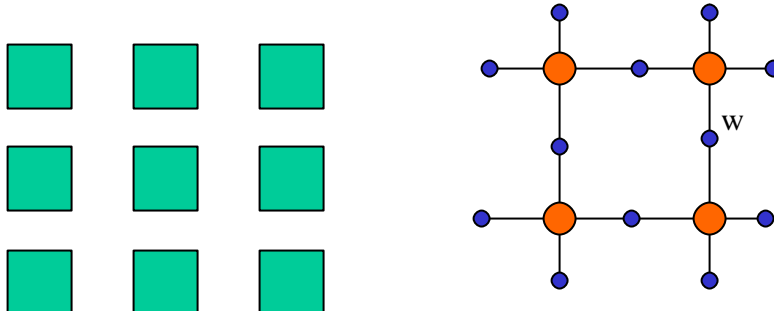
- **Problem:** Find sequence of channels for all routes
 - minimizing channel sizes
 - minimize max channel size
 - meeting channel capacity limits



CALTECH CS137 Winter2002 -- DeHon

Global \rightarrow Graph

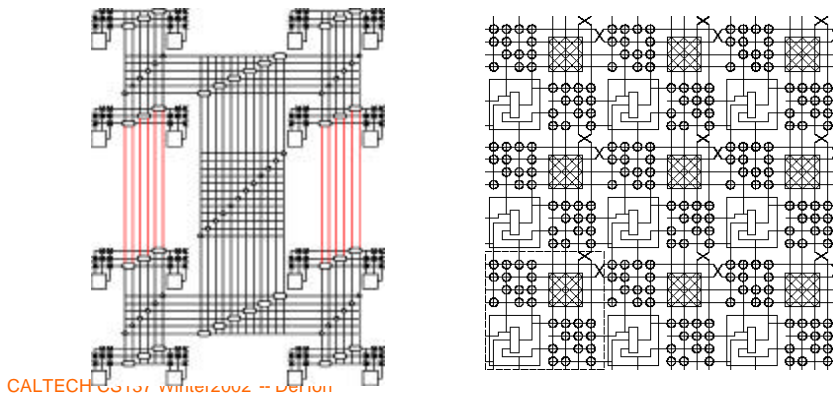
- Graph Problem on routes through regions



CALTECH CS137 Winter2002 -- DeHon

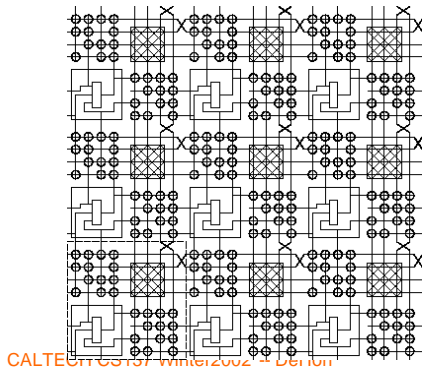
Global/Detail

- With limited switching (e.g. FPGA)
 - can represent routing graph exactly



Routing in Graph

- Find (shortest/available) path between source and sink
 - search problem (e.g. BFS, Alpha-Beta)

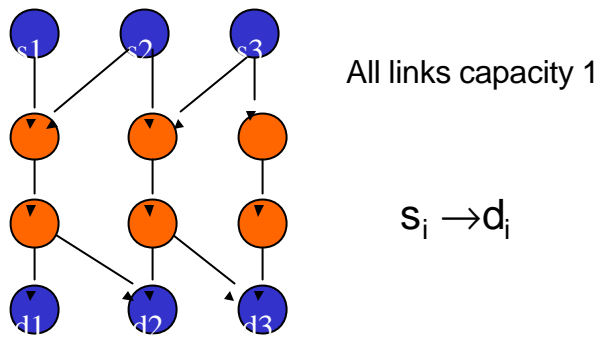


Easy?

- Finding a path is moderately easy
- What's hard?
 - Can I just iterate and pick paths?

CALTECH CS137 Winter2002 -- DeHon

Example



CALTECH CS137 Winter2002 -- DeHon

Challenge

- Satisfy *all* routes simultaneously
- Routes share potential resources
- Greedy/iterative
 - not know who needs will need which resources
 - *i.e.* resource/path choice looks arbitrary
 - ...but earlier decisions limit flexibility for later
 - (somewhat like scheduling)
 - order affect result

CALTECH CS137 Winter2002 -- DeHon

Negotiated Congestion

- Old idea
 - try once
 - see where we run into problems
 - undo problematic/blocking allocation
 - rip-up
 - use that information to redirect/update costs on subsequent trials
 - retry

CALTECH CS137 Winter2002 -- DeHon

Negotiated Congestion

- Here
 - route signals
 - allow overuse
 - identify overuse and encourage signals to avoid
 - reroute signals based on overuse/past congestion

CALTECH CS137 Winter2002 -- DeHon

Basic Algorithm

- Route signals along minimum cost path
- If congestion/overuse
 - assign higher cost to congested resources
- Repeat until done

CALTECH CS137 Winter2002 -- DeHon

Key Idea

- Congested paths/resources become expensive
- When there is freedom
 - future routes, with freedom to avoid congestion will avoid it
- When there is less freedom
 - must take congested routes
- Routes which must use congested resources will, while others will chose uncongested paths

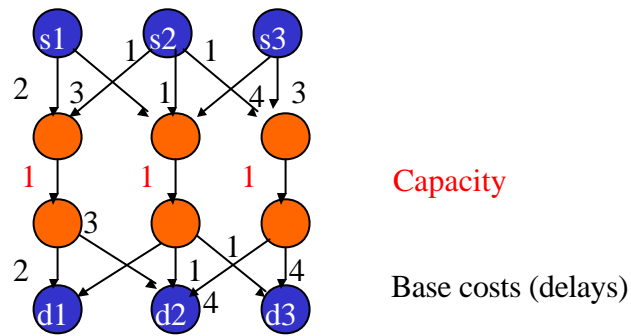
CALTECH CS137 Winter2002 -- DeHon

Cost Function (1)

- Cost = base \times f(#routes using, time)
- Base cost of resource
 - delay of path
 - Encourage minimum resource usage
 - (minimum length path, if possible)
 - minimizing delay = minimizing resources
- Congestion
 - penalizes (over) sharing
 - increase sharing penalty over time

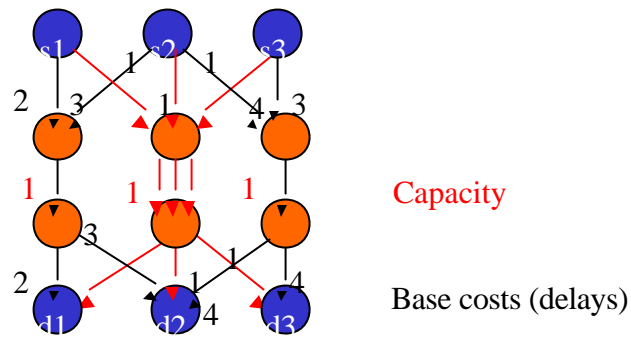
CALTECH CS137 Winter2002 -- DeHon

Example (first order congestion)



CALTECH CS137 Winter2002 -- DeHon

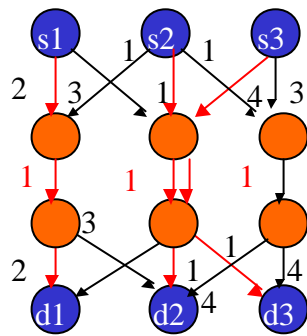
Example (first order congestion)



All, individual routes prefer middle; create congestion.

CALTECH CS137 Winter2002 -- DeHon

Example (first order congestion)



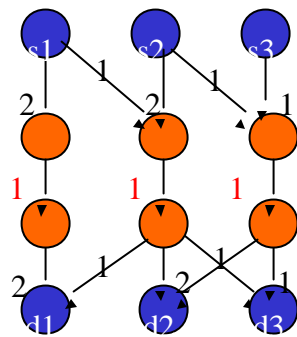
Capacity

Base costs (delays)

Reroute, avoid congestion.

CALTECH CS137 Winter2002 -- DeHon

Example (need for history)



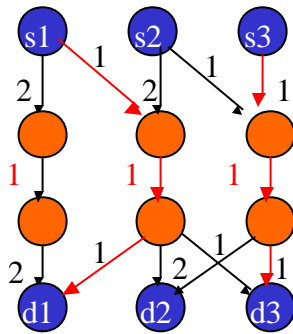
Capacity

Base costs (delays)

Need to redirect uncongested paths; how encourage?

CALTECH CS137 Winter2002 -- DeHon

Example (need for history)



Local congestion alone won't drive in right directions.

May ping-pong back and forth.

(can imagine longer chain like this)

Cannot route $s2 \rightarrow d2$

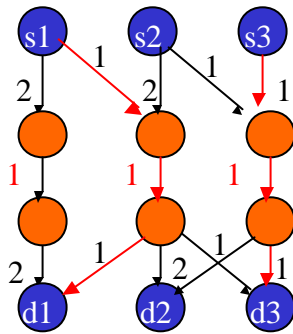
CALTECH CS137 Winter2002 -- DeHon

Cost Function (2)

- $\text{Cost} = (\text{base} + \text{history}) * f(\# \text{resources}, \text{time})$
- History
 - avoid paths with history of congestion

CALTECH CS137 Winter2002 -- DeHon

Example (need for history)



S2→d3 initially ping-pong

Builds up congestion history on path 2 and 3

Eventually makes path 2 and 3 more expensive than path 1; ...resolves conflict...

CALTECH CS137 Winter2002 -- DeHon

What about delay?

- Existing formulation uses delay to reduce resources, but doesn't directly treat
- Want:
 - prioritize critical path elements for shorter delay
 - allow nodes with slack to take longer paths

CALTECH CS137 Winter2002 -- DeHon

Cost Function (Delay)

- Cost=
 - $W(\text{edge}) * \text{delay} + (1 - W(\text{edge})) * \text{congest}$
 - congest as before
 - $(\text{base} + \text{history}) * f(\# \text{signals}, \text{time})$
- $W(\text{edge}) = D(\text{edge}) / D_{\text{max}}$
 - 1 for critical path
 - < 1 for paths with slack
- Use $W(\text{edge})$ to order routes
- Update each round

CALTECH CS137 Winter2002 -- DeHon

Convergence

- Chan+Schlag [FPGA'2000]
 - cases where doesn't converge
 - special case of bipartite graphs
 - converge if incremental
 - or if prefer uncongested to least history cost
- theory (continuous)
 - only reroute overflow
 - converge in $O(|E|)$ reroutes
 - But then have fractional routes...

CALTECH CS137 Winter2002 -- DeHon

Rerouting

- Default: reroute everything
- Can get away rerouting only congested nodes
 - if keep routes in place
 - history force into new tracks
 - causing greedy/uncongested routes to be rerouted

CALTECH CS137 Winter2002 -- DeHon

Rerouting

- Effect (only reroute congest)
 - maybe more iterations
 - (not reroute a signal until congested)
 - less time
 - ? Better convergence
 - ? Hurt quality?
 - (not see strong case for)
 - ...but might hurt delay quality
 - Maybe followup rerouting everything once clear up congestion?

CALTECH CS137 Winter2002 -- DeHon

Run Time?

- Route $|E|$ edges
- Each path search $O(|E_{\text{graph}}|)$ worst case
 - ...generally less
- Iterations?

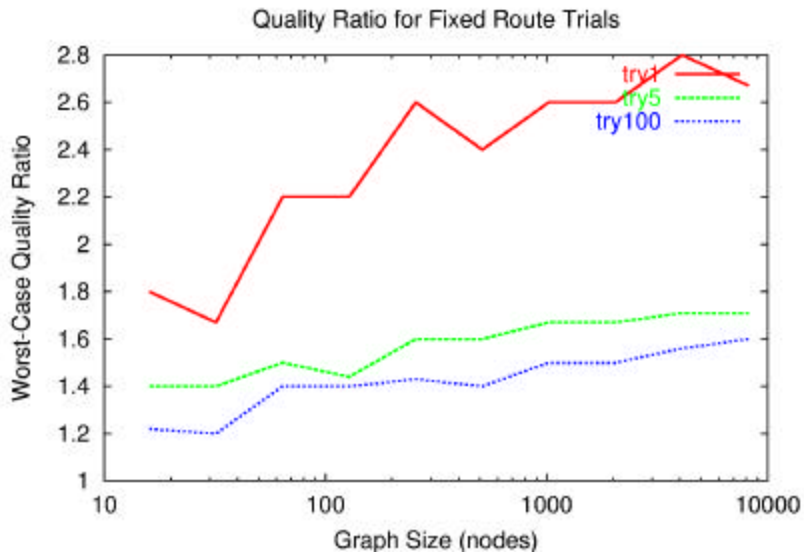
CALTECH CS137 Winter2002 -- DeHon

Quality and Runtime...

- For Synthetic netlists on HSRA
 - Expect to be worst-case problems
- Number of individual route trials limited (measured) as multiple of nets in design
 - (not measuring work per route trial)

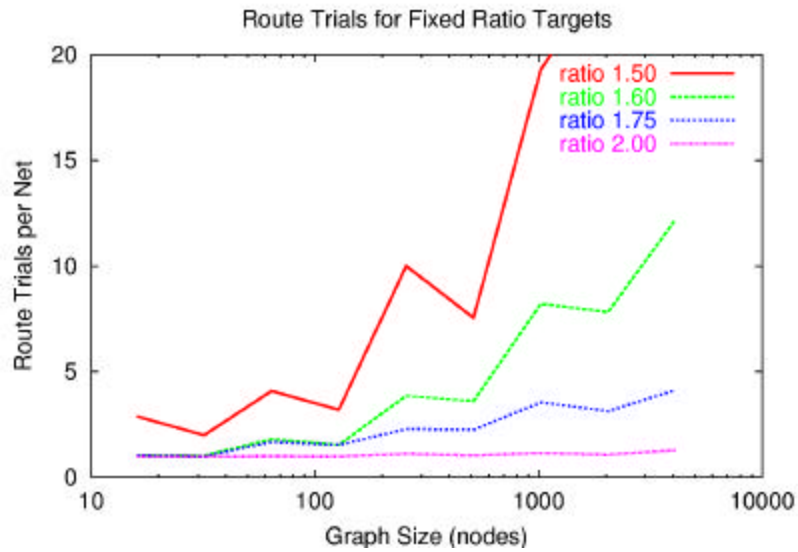
CALTECH CS137 Winter2002 -- DeHon

Quality: fixed runtime



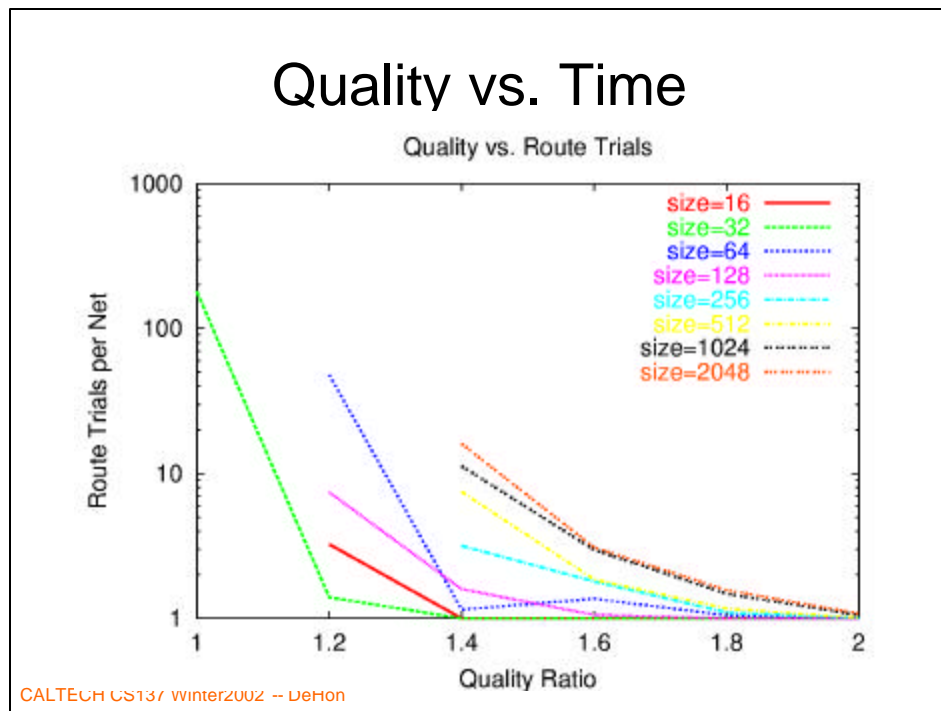
CALTECH CS137 Winter2002 -- DeHon

Quality Target



CALTECH CS137 Winter2002 -- DeHon

Quality vs. Time



Search Ordering

- Default: breadth first search for shortest
 - $O(\text{total-paths})$
 - $O(N^p)$ for HSRA
- Alternately: use A*:
 - estimated costs/path length, prune candidates earlier
 - can be more depth first
 - (search promising paths as long as know can't be worse)

CALTECH CS137 Winter2002 -- DeHon

Searching

- In general:
 - greedy/depth first searching
 - find a path faster
 - may be more expensive
 - (not least delay, congest cost)
 - tradeoff by weighting
 - estimated delay on remaining path vs. cost to this point
 - control greediness of router
 - More greedy is faster at cost of less optimal paths (wider channels)
 - 40% W → 10x time [Tessier/thesis'98]

CALTECH CS137 Winter2002 -- DeHon

Searching

- Use alpha-beta like search
 - Always expanded (deepen) along shortest ...as long as can prove no other path will dominate
 - Uncongested: takes $O(\text{path-length})$ time
 - Worst-case reduces to breadth-first
 - $O(\text{total-paths})$
 - $O(N^P)$ for HSRA

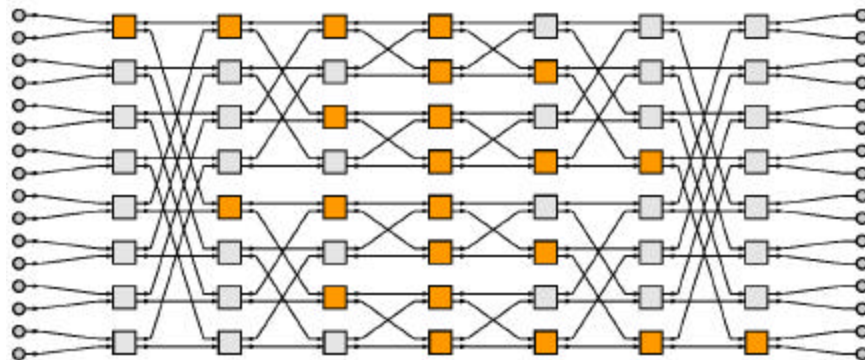
CALTECH CS137 Winter2002 -- DeHon

Domain Negotiation

- For Conventional FPGAs (and many networks)
 - path freedom
 - bushy in middle
 - low on endpoints

CALTECH CS137 Winter2002 -- DeHon

Multistage/Benes

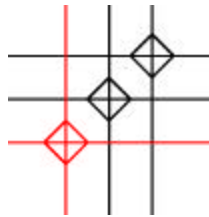


Switches in all paths 0000 to 1111

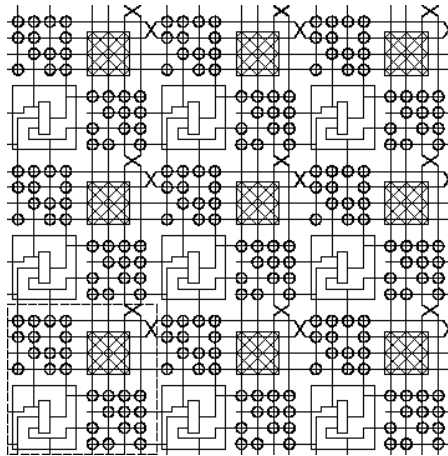
CALTECH CS137 Winter2002 -- DeHon

Conventional FPGA Domains

Called:
subset
disjoint

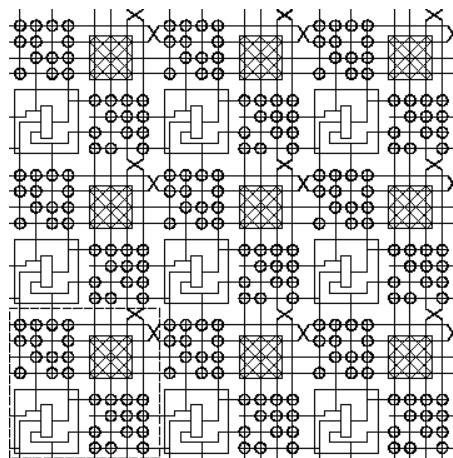


CALTECH CS137 Winter2002 -- DeHon



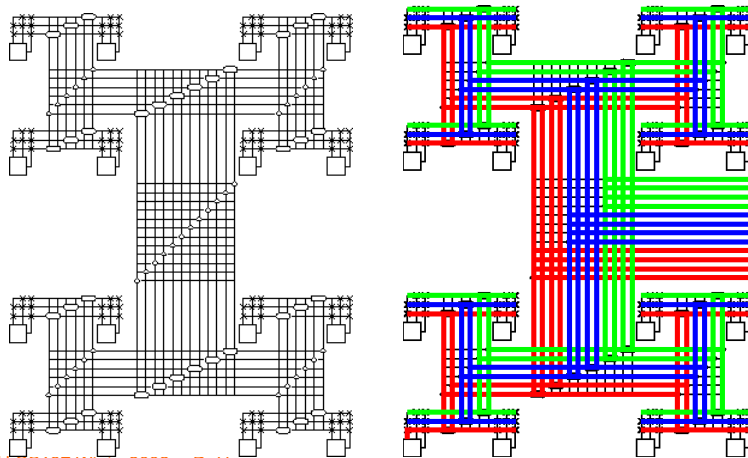
Domain Routing

- No point in searching along an entire path from source
- Just to find it's heavily congested at sink



CALTECH CS137 Winter2002 -- DeHon

HSRA Domains



CALTECH CS137 Winter2002 -- DeHon

Domain Negotiation

- Path bottlenecks exist at **both** endpoints
- Most critical place for congestion
- Most efficient: work search from both ends
 - more limiting in A*/Alpha-Beta search
 - focus on paths with least (no) congestion on endpoints first
 - FPGAs -- picking “domain” first
 - otherwise paths may look equally good up to end (little pruning)

CALTECH CS137 Winter2002 -- DeHon

Summary

- Finding short path easy/well known
- **Complication:** need to route set of signals
 - who gets which path?
 - Arbitrary decisions earlier limit options later
- Idea: iterate/relax using congestion history
 - update path costs based on congestion
 - reroute with new costs
- Accommodate delay and congestion

CALTECH CS137 Winter2002 -- DeHon

Big Ideas

- Exploit freedom
- Technique:
 - Graph algorithms (BFS, DFS)
 - Search techniques (A*, Alpha-Beta)
 - Iterative improvement/relaxation

CALTECH CS137 Winter2002 -- DeHon