CS294-7: Reconfigurable Computing

Day 12: Thursday, February 27

Interconnect

- Admin
- Interconnect Requirements

Admin

- CAD Tools for Project 1
  - `~cs254/doc/xilinx.help`
  - `/usr/pub/xilinx.help`
  - `/share/k/xilinx/online/online`
  - `<http://www.xilinx.com>`

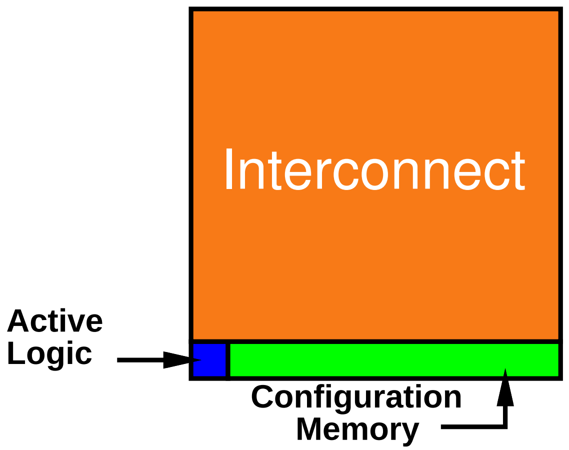
Quote

It's the interconnect, stupid.

– Tom Knight, Jr.

Review (day4): Area Breakdown (from Model)

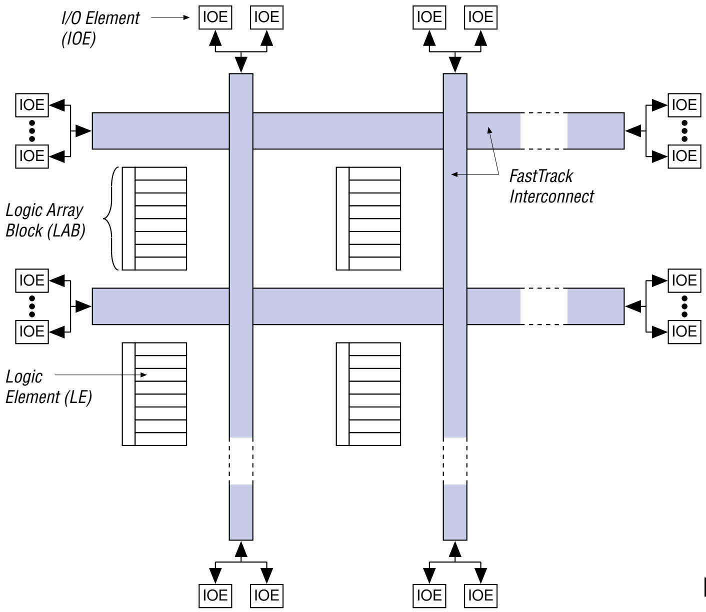
$$A_{bit\_elm} = A_{fixed} + \underbrace{N_{SW}(N_p, w, p) \cdot A_{SW}}_{\textbf{interconnect}} + \underbrace{\left(\frac{c}{w}\right) \cdot n_{ibits} \cdot A_{mem\_cell}}_{\textbf{instruction memory}} + \underbrace{d \cdot A_{mem\_cell}}_{\textbf{retiming memory}}$$



Function	Area (λ^2)
LUT MUX + ff	20K (generous, closer to 10K)
Programming Memory	80K (240K typical unencoded)
Interconnect	700K (for $N_p = 2048$)

Review (day 4): Delay Breakdown

Design	Path	Total Delay	LUT Delay	Inter. %
Altera 8K	LUT-local-LUT	**2.5 ns**	**2 ns**	**20%**
	LUT-row-local-LUT	**7.5 ns**	**2 ns**	**73%**
	LUT-row-column-local-LUT	**10.5 ns**	**2 ns**	**81%**
DPGA	LUT-LUT (in subarray)	**3.5 ns**	**1.5 ns**	**60%**
	LUT-xbar-LUT	**7 ns**	**1.5 ns**	**80%**

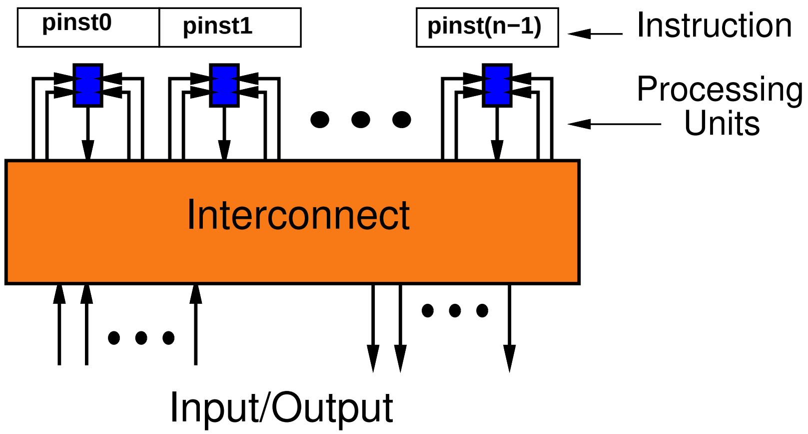


Interconnect

Problem:

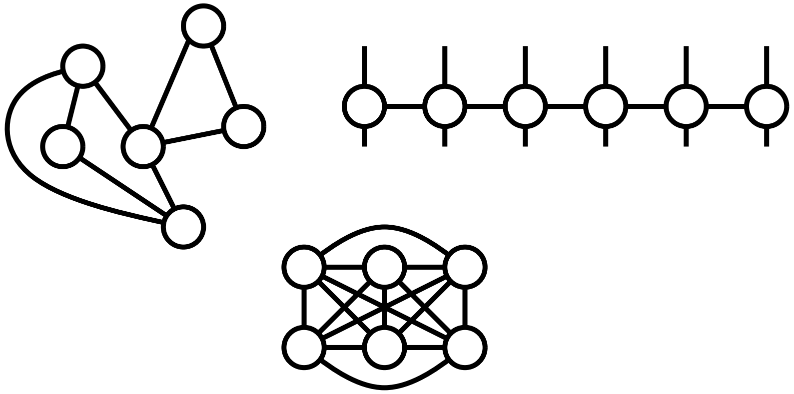
- Thousands of bit operators producing results.
- Each taking as inputs the results of other bit operators
- Interconnect is **late bound** – don't know characteristics at fab time.

Initial Assumptions: Have to connect them all *simultaneously*.



Definition: Bisection Bandwidth

- Partition the design into two, equal size halves
- Allocate "gates" to partitions to minimize the number of wires (nets) travelling between the two halves
- **Bisection Bandwidth** is the number of wires in this crossing



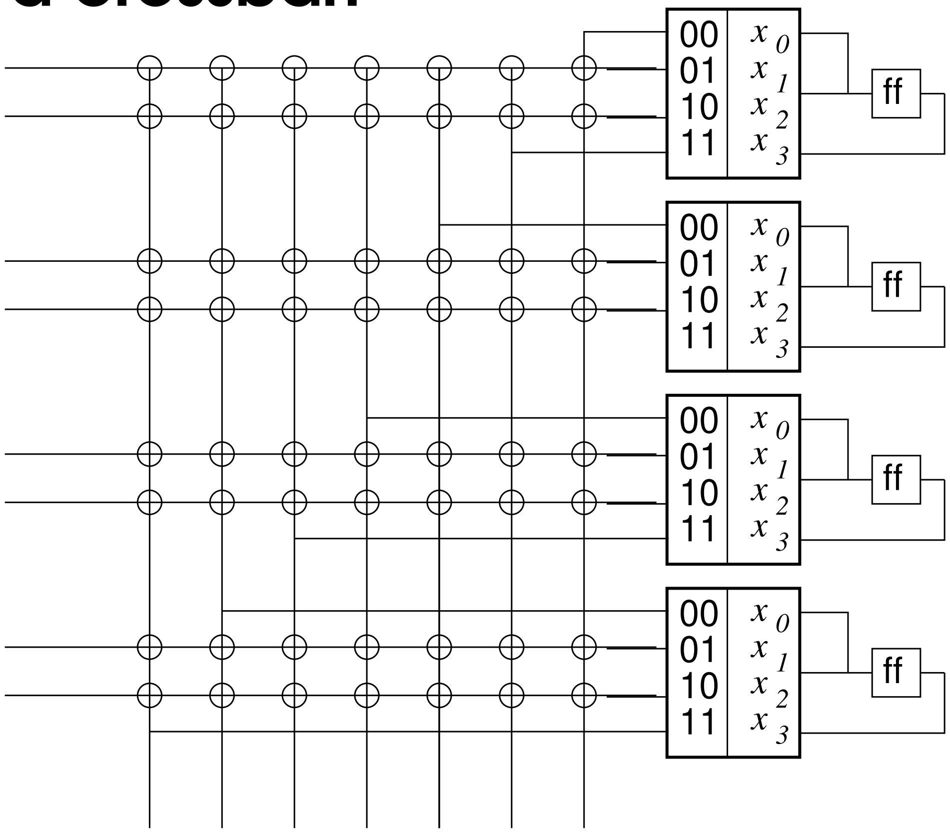
Interconnect Design Issues

1. **Flexibility** – route anything (within reason?)
   - bisection bandwidth
   - simultaneous routes
2. **Area**
   - bisection bandwidth
   - switches
3. **Delay (and Power)**
   - switches in path
   - switch parasitics (stubs on path)
   - wire length
4. **Routability**
   - difficulty of finding a route

Attempt 1

Any operator may want to take as input the output of any other operator.

$\Rightarrow$ Let's build a crossbar!



Crossbar Analysis

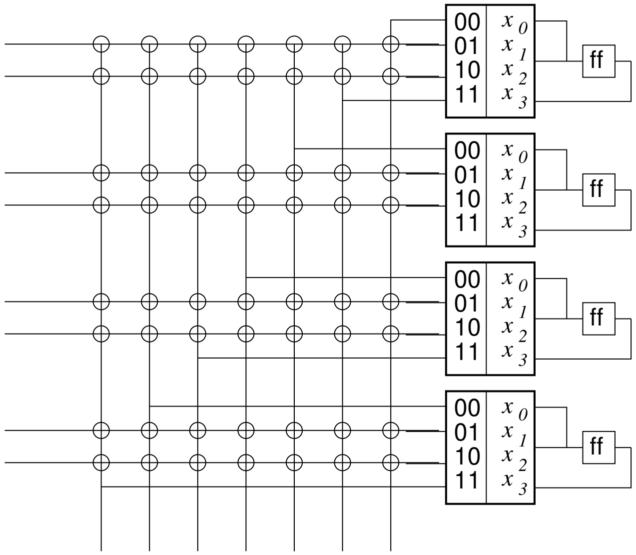
Delay

- **Parasitic loads** $(k+1)n$
- **Switches/path** 1
- **Wire Length** $O(kn)$
- **Delay** $O(kn)$

Area

- **Bisection BW** n
- **Switches** kn^2
- **Area** $O(n^2)$

Routes everything – guaranteed.



Structure

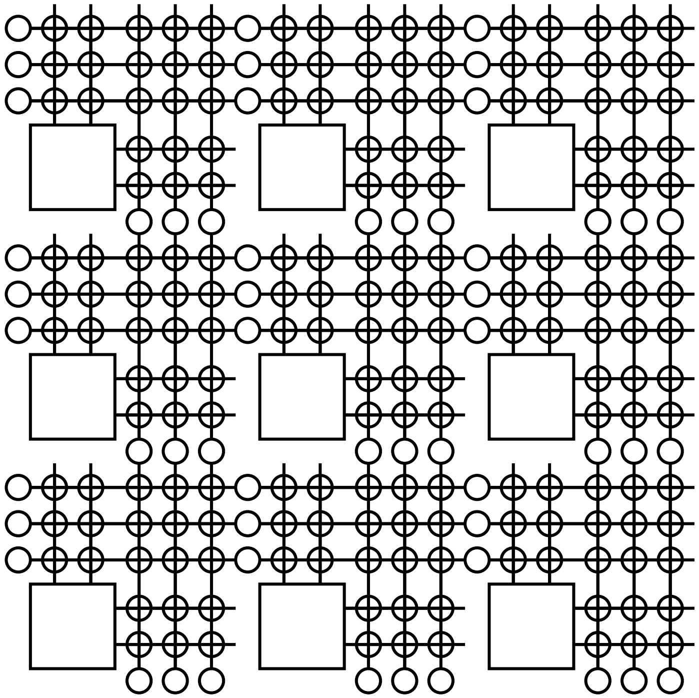
Typical trick: Understand and exploit expected structure

Locallity $\rightarrow$ put connected components "close" together.

Attempt 2

- Wires are expensive.
- Local interconnections cheaper than long ones.

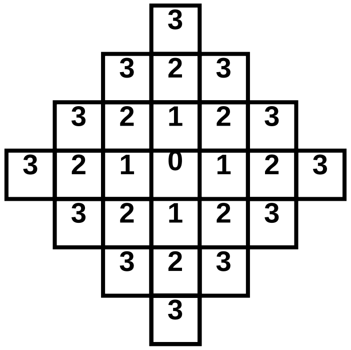
→ How about a mesh?!?

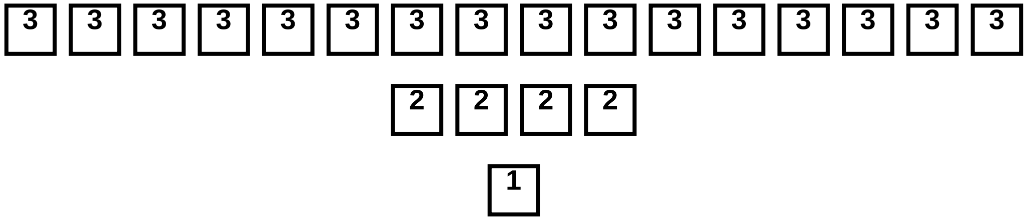


Mesh Analysis

Try placing everything "close"

Manhattan Distance	Places	Transitive Fanin
1	4	4
2	8	16
3	12	64
⋮	⋮	⋮
n	$4n$	4^n





Mesh Analysis

Delay

Switches/path	l
Wire Length	$O(l)$
Best Delay	$O(w)$
Worst Delay	$O(\sqrt{n}w)$

Area

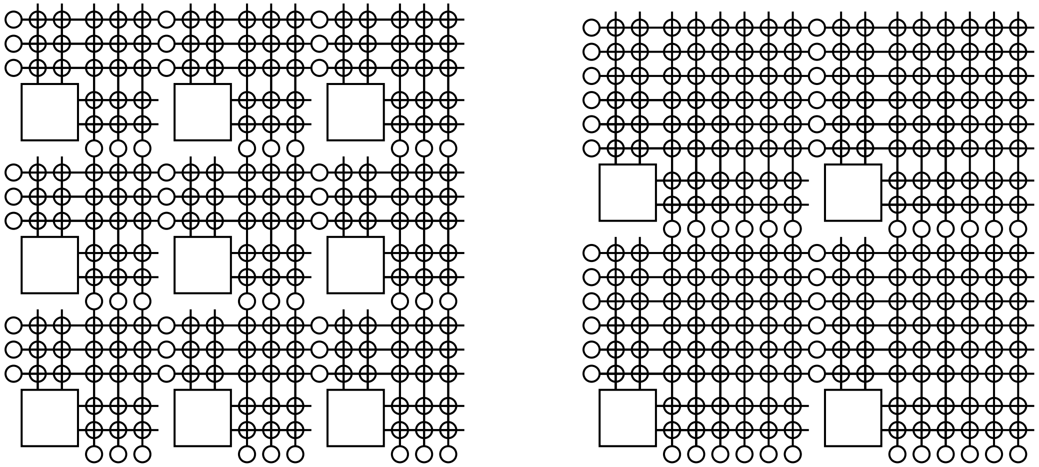
Bisection BW	$w\sqrt{n}$
Switches	$O(nw^2)$
Area	$O(nw^2)$

What's w?

What can it route?

Typical wire delay?

(watching constants – switches slower than wires)



Rent's Rule

Empirical relationship –
capture interconnect requirements.

Given: a collection of n gates, the number of I/O signals is given by:

$$IO = Cn^p$$

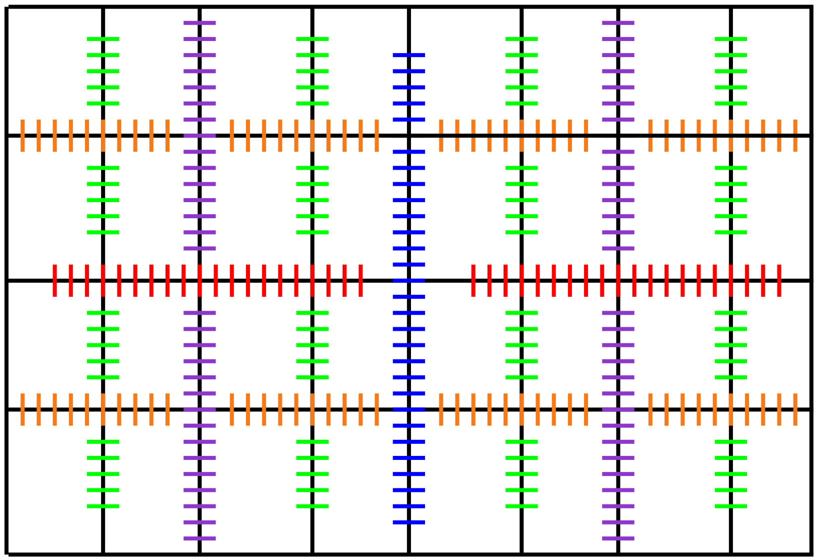
$(p \leq 1.0)$

Captures locallity notion:

- **some signals are generated and consumed locally**
- **fewer long distance lines**

Rent's Rule

The Rent equation tells us the bisection bandwidth ...recursively through the chip.



C=5

p=0.6

1 – 5
2 – 10
4 – 15
8 – 20
16 – 30

Also, average wire length and channel capacity (w) are direct consequents of Rent's Rule.

Rent's Rule

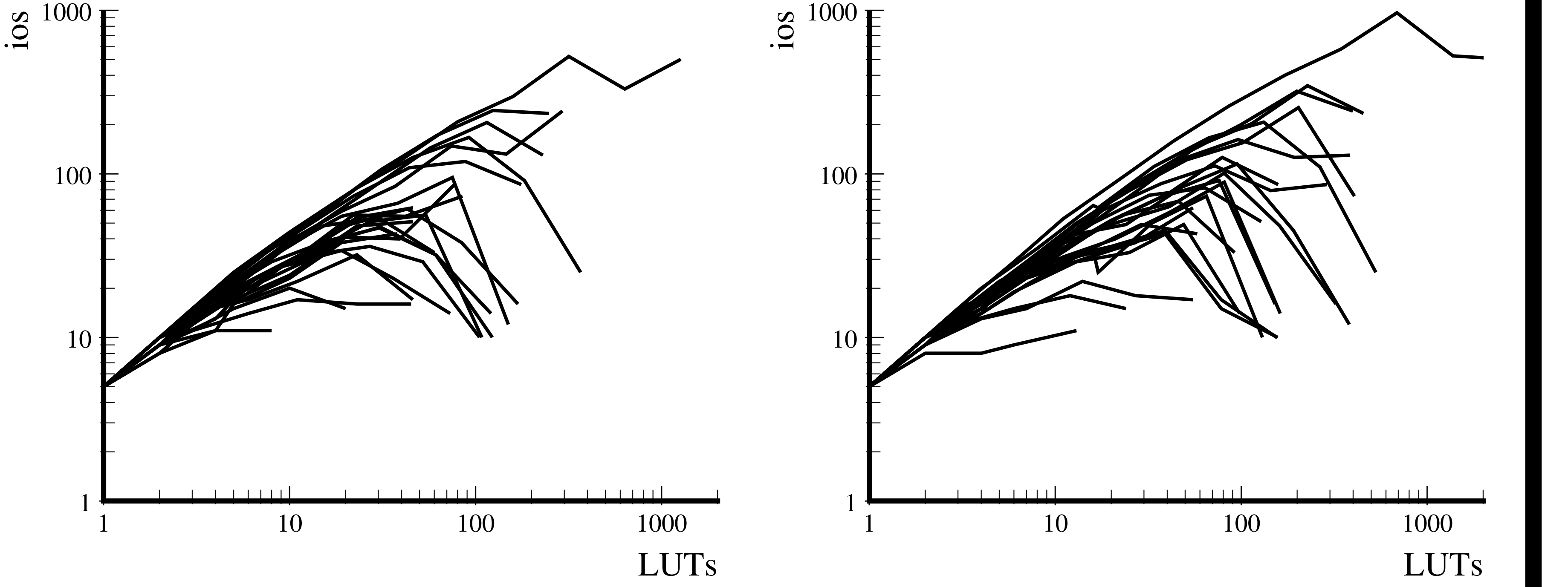
Typically consider $0.5 \leq p \leq 0.75$
unpartitioned/unstructured $0.9 \leq p \leq 1.0$

Bakoglu:

Type	C	p
SRAM	6	0.12
Microprocessor	0.8	0.45
Gate Array	1.9	0.5
"High-speed" computer	1.4	0.63

Donath – exponential relationship arises from hierarchical composition

Rent's Rule – Random Logic



design	area C	area p	delay C	delay p
des	8.1	0.66	6.4	0.8
rot	6.7	0.68	5.7	0.78
f51m	6.4	0.37	6.2	0.67

(simple partitioner – room for better partitions)

Rent's Rule

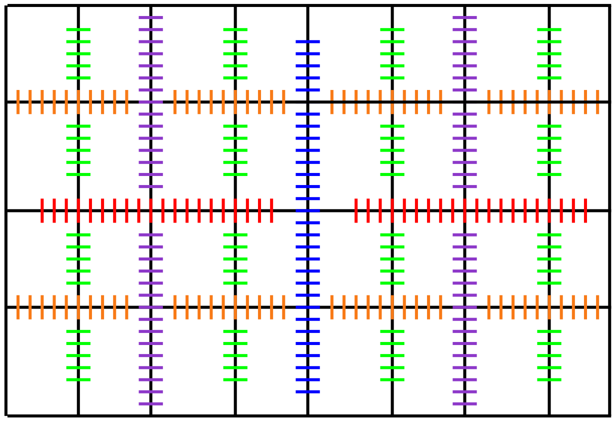
Cellular	$IO \propto \sqrt{n}$	$p = 0.5$
Systolic (DNA,FIR)	$IO = C$	$p = 0$ **?**
Systolic (Matrix Mpy)	$IO \propto \sqrt{n}$	$p = 0.5$
bit serial	Isshiki claim	$p = 0.3 - 0.4$

Rent's Rule – Caveats

- Modern "systems" on a chip – contain subcomponents of varying Rent complexity
- Less I/O at certain "natural boundaries"
- At some point get system close
  (Rent's Rule apply to my workstation, PC, pda?)

Wiring Requirements

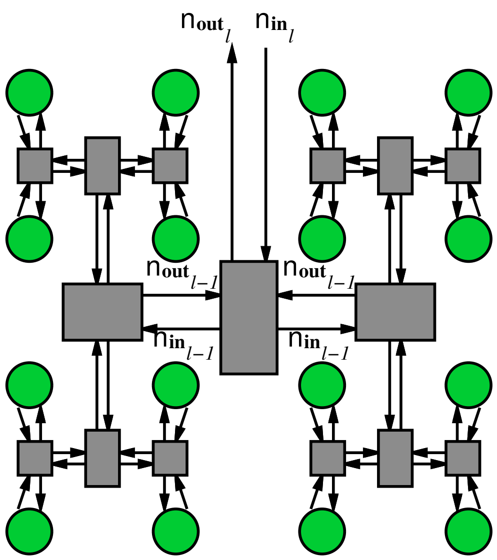
- $IO \propto n^p$
- **Bisection BW** $\propto n^p$
- **Side length at least** $\propto n^p$
- **Area at least** $\propto (n^p)^2 = n^{2p}$

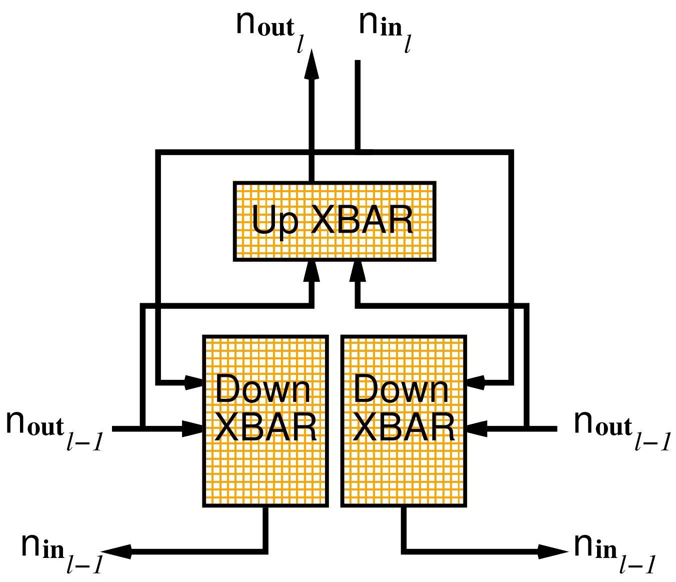


- ☐ **If wires are limited resource, Area is** $O(n^{2p})$
- ☐ **If wires are limited resource, Area per LUT is** $O(n^{2p-1})$

  $(p > 0.5)$

Switches





$$N_{SW-total_l} = \left(n \times \underbrace{\left(\overbrace{((n-1)\cdot n_{out_{l-1}} + n_{in_l})}^{\text{inputs to down xbar}} \times n_{in_{l-1}} \right)}_{\text{each down xbar}} \right) + \underbrace{(n_{out_l} \times (n \cdot n_{out_{l-1}}))}_{\text{up xbar}}$$

n – tree arity ($n = 2$ in cases shown)

Switches

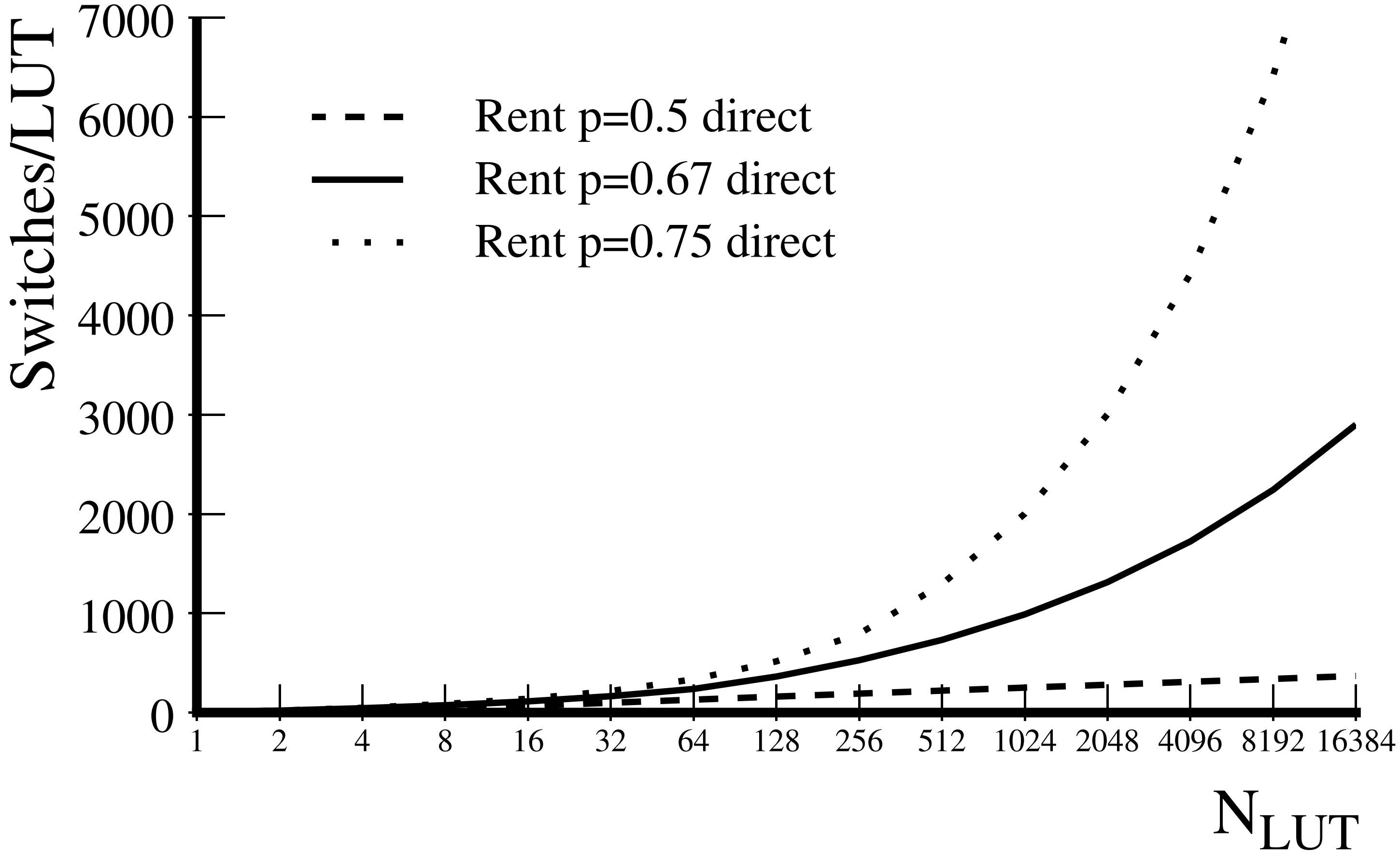
(two dense pages of math from previous slide to this result – see thesis)

Per LUT:

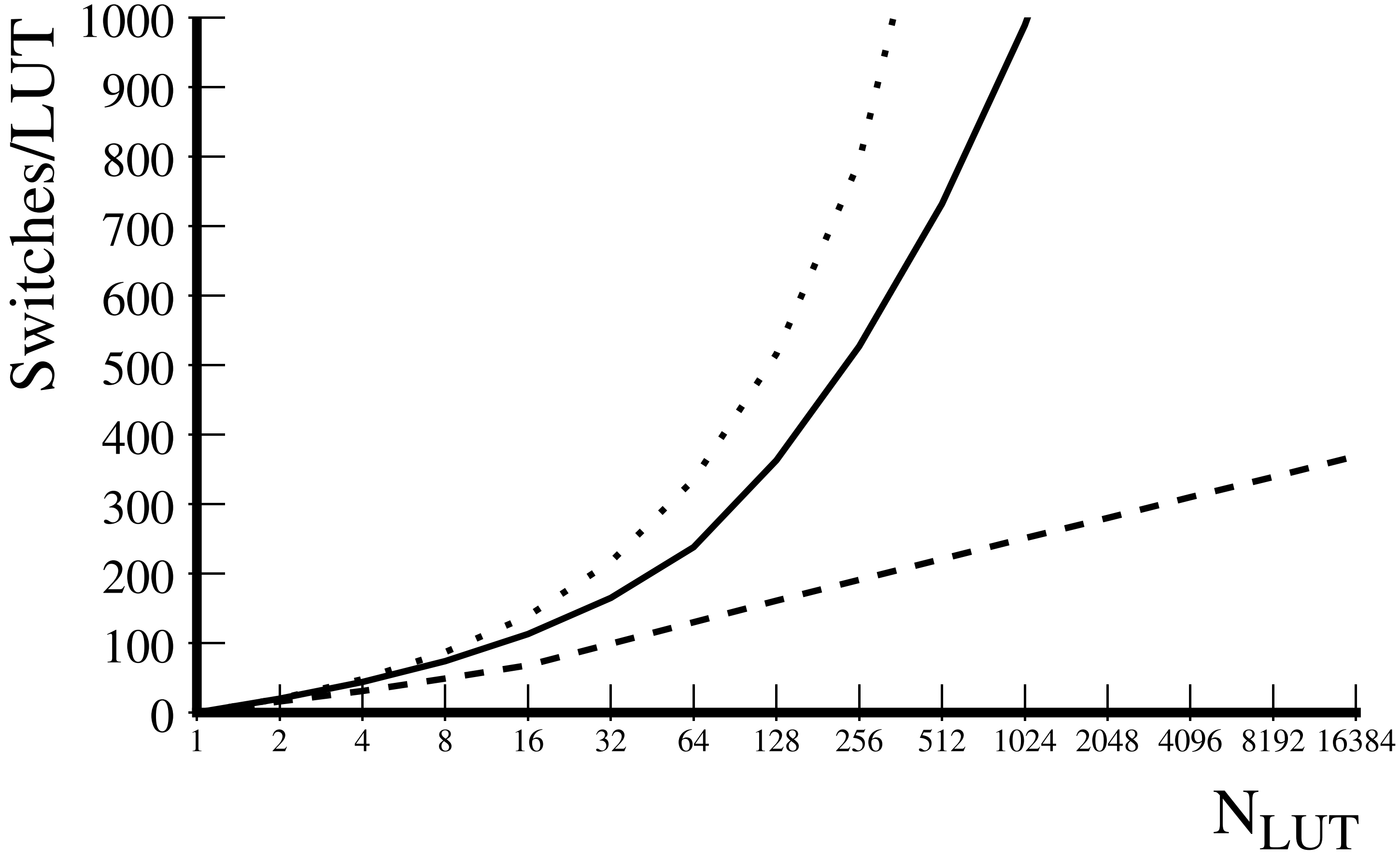
$$n_{SW} \leq \begin{cases} K^2 \cdot \left(n - 1 + 2n^p\right) \cdot \left(\frac{n^{(1-2p)}}{n^{(1-2p)}-1}\right) & p < 0.5 \\ \log_n\left(N_{LUT}\right) \cdot K^2 \cdot \left(n - 1 + 2\sqrt{n}\right) & p = 0.5 \\ \left(N_{LUT}^{(2p-1)}\right) \cdot K^2 \cdot \left(\frac{1}{n^{(2p-1)}-1}\right) \cdot \left(n - 1 + 2n^p\right) & p > 0.5 \end{cases}$$

- **If switches are limited resource,**
  Area per LUT is $O(N_{LUT}^{2p-1})$ (for $p > 0.5$).

Switches and Wires



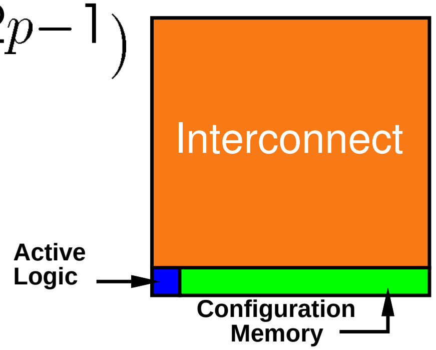
Switches and Wires



Switches and Wires

$p < 0.5 \Rightarrow$ **Asymptotic Area per LUT is** $O(1)$

$p > 0.5 \Rightarrow$ **Asymptotic Area per LUT is** $O(n^{2p-1})$



For $p > 0.5$ **interconnect dominates and will continue to dominate.**

Note, an architecture with bisection BW growth (switching growth) as shown is **necessary** to route graphs with a given p, but not **sufficient**.

We can construct architectures with BW and switching growth as shown to route any graph with $p_{graph} \leq p_{net}$**.**

What Rent didn't tell us

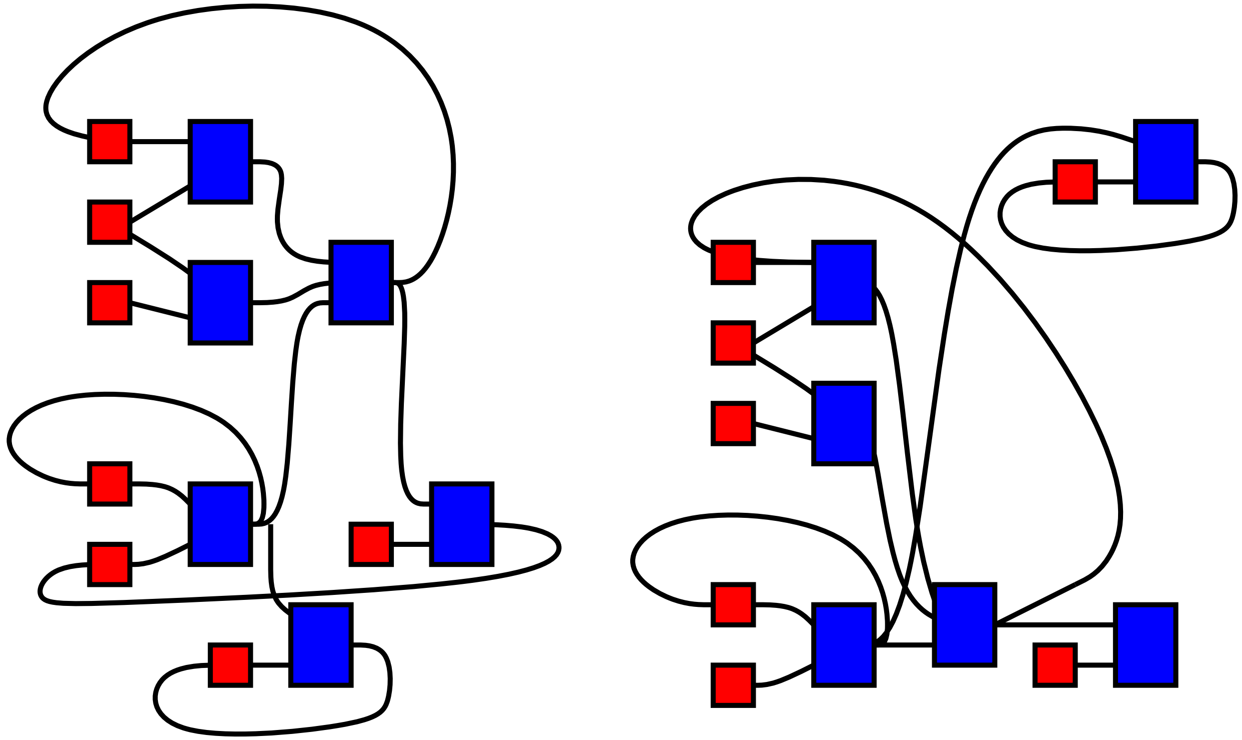
Rent's Rule tells us how many wires/switches we need find a route.

Purely geometrical $\rightarrow$ no explicit consideration of critical path.

i.e. a partition may leave the critical path weaving between halves

(This is part of the reason we find "high-speed" logic has $p \approx 0.75$, while logic in general is closer to $p \approx 0.5$)

Critical Path and Bisection



- Minimum cut may cross critical path multiple times.
- Minimizing long wires in critical paths will increase cut size.

Efficiency

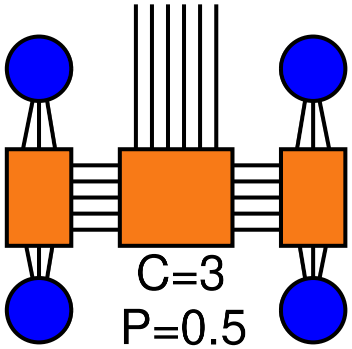
What happens when $p_{net} \neq p_{design}$?

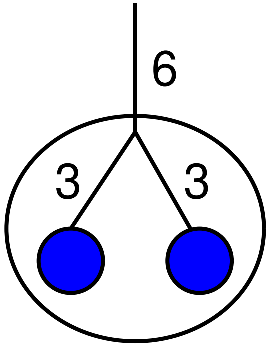
- $p_{net} > p_{design}$
- $p_{design} > p_{net}$

Efficiency – Model

$p_{design} > p_{net}$ – underuse LUTs by C so can route:

$$K N_{LUT}^{p_{design}} \leq K(C \cdot N_{LUT})^{p_{net}}$$



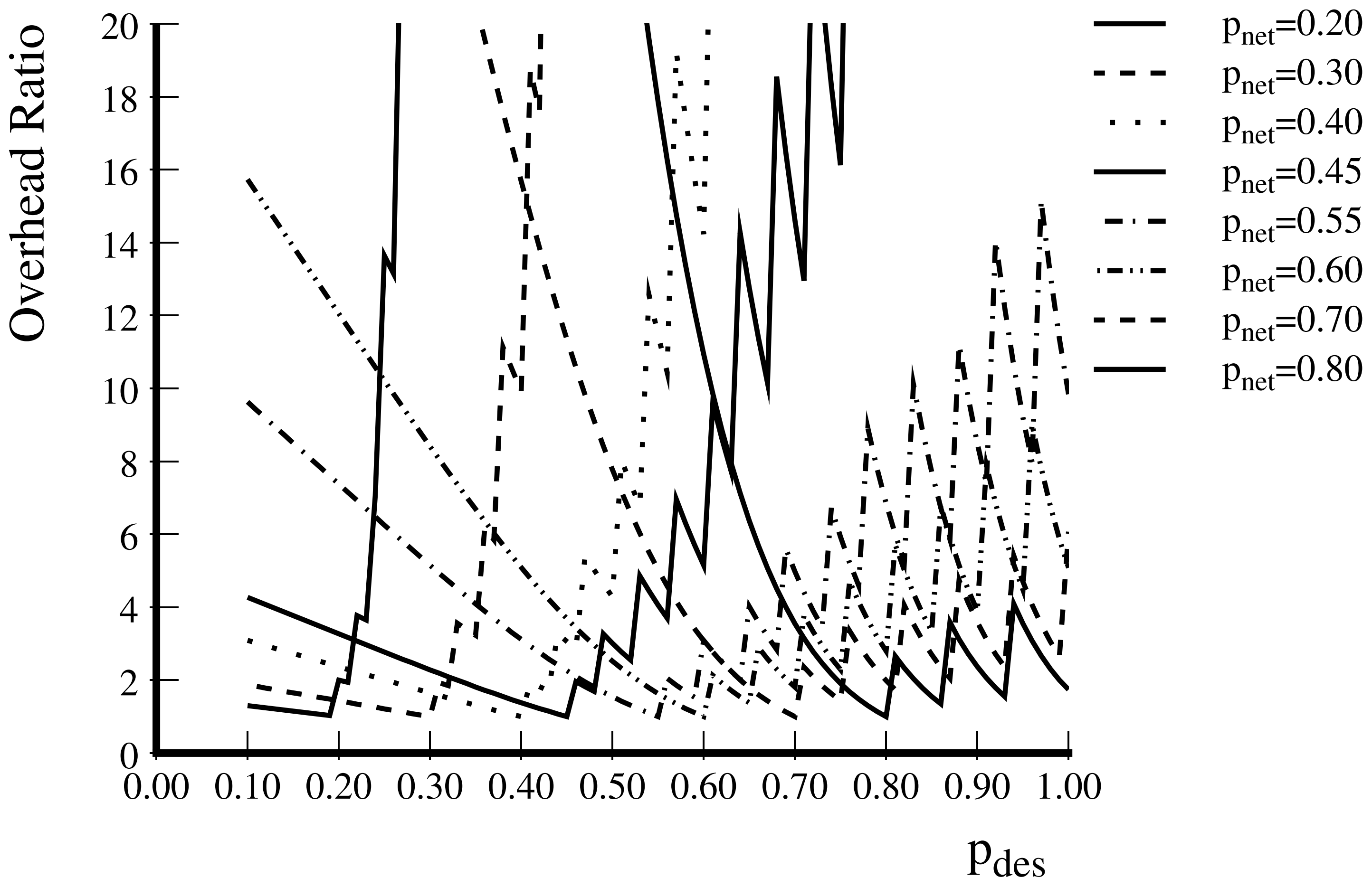


Scale up:

(*i.e.*, shift design up heirarchy until i/os fit)

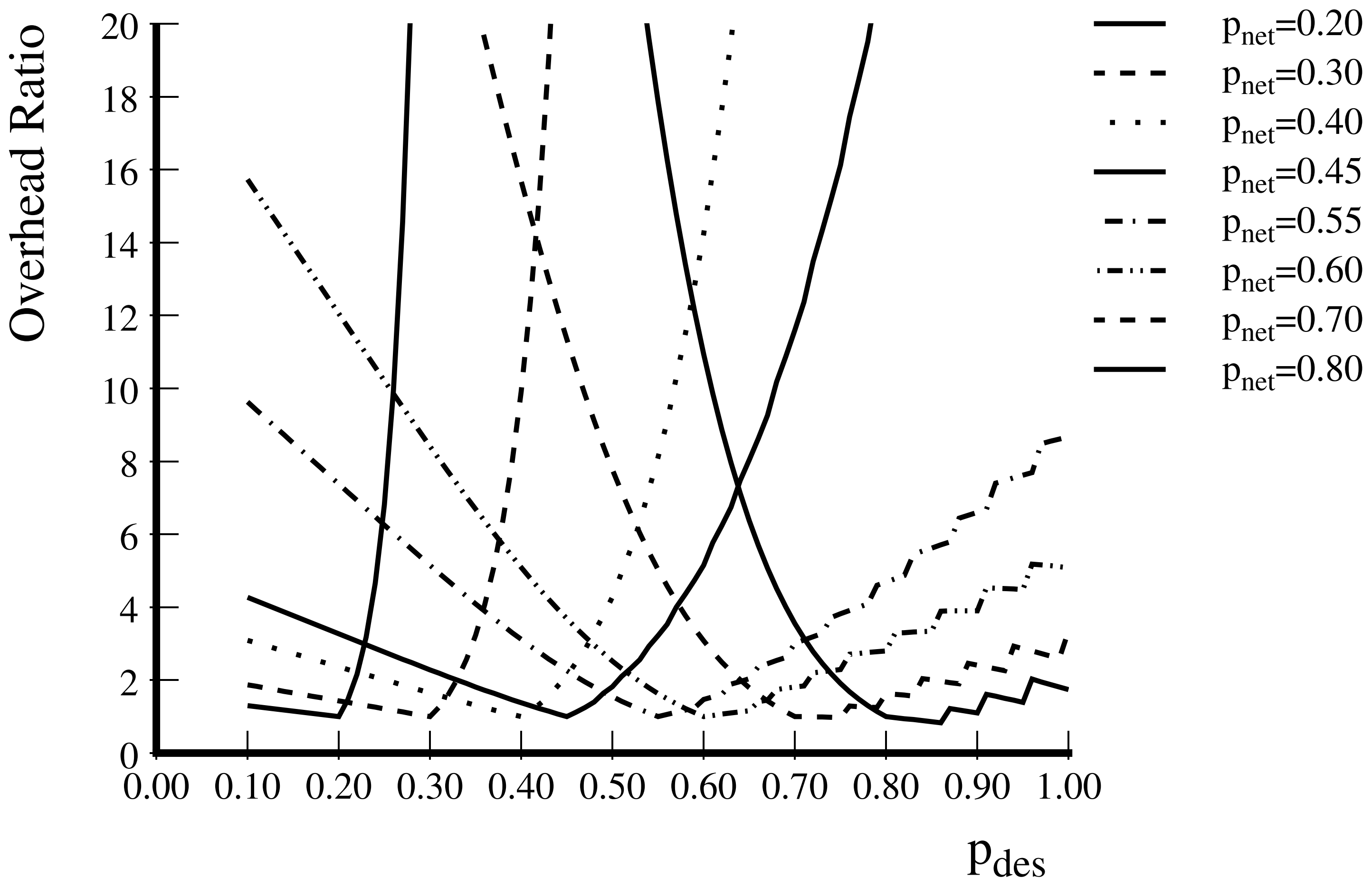
$$\begin{aligned} N_{LUT}^{p_{design}} &= (C \cdot N_{LUT})^{p_{net}} \\ N_{LUT}^{p_{design}/p_{net}} &= C \cdot N_{LUT} \\ C &= N_{LUT}^{(p_{design}/p_{net}-1)} \end{aligned}$$

Efficiency – Discrete



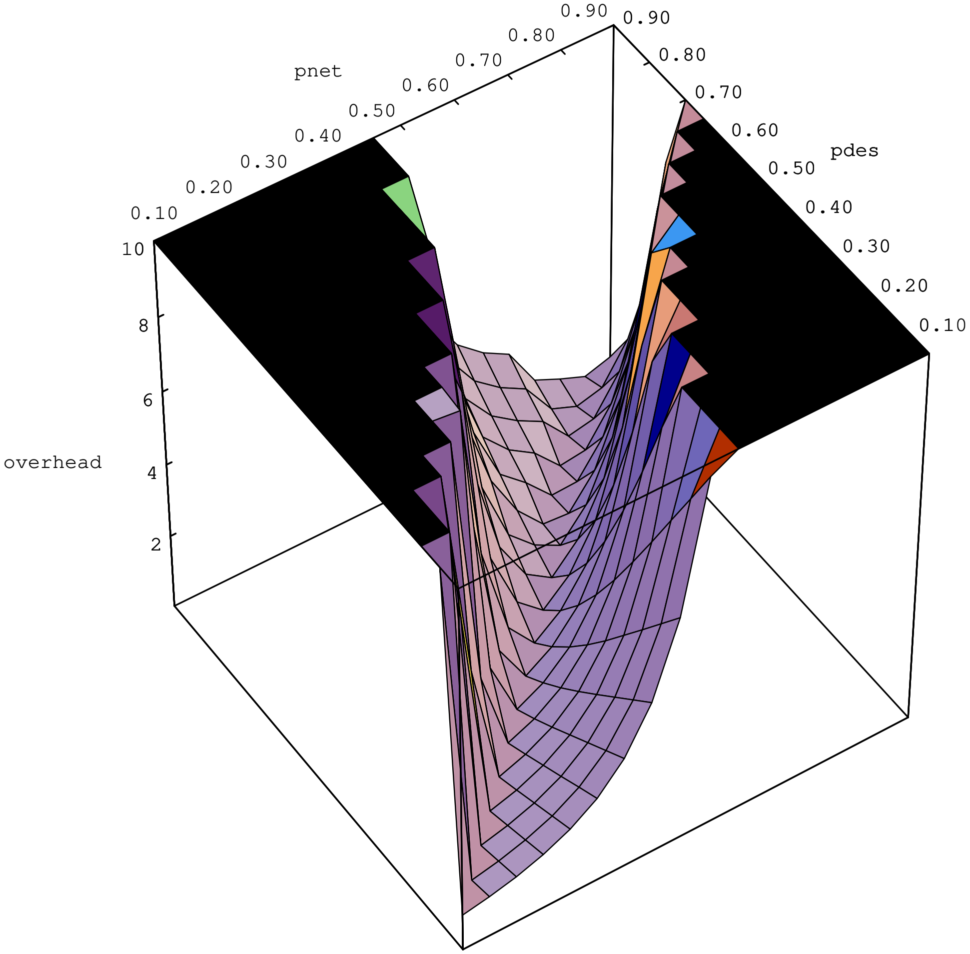
$k = 4$, $n = 2$, $N_{LUT} = 4096$

Efficiency – Continuous



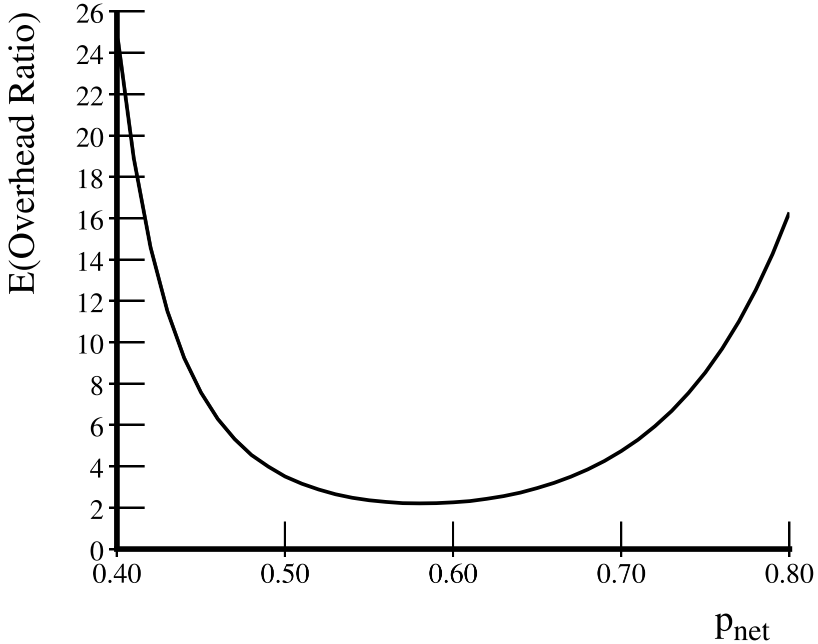
$k = 4$, $n = 2$, $N_{LUT} = 4096$

Efficiency – Continuous



$k = 4$, $n = 2$, $N_{LUT} = 4096$

Expected Overhead

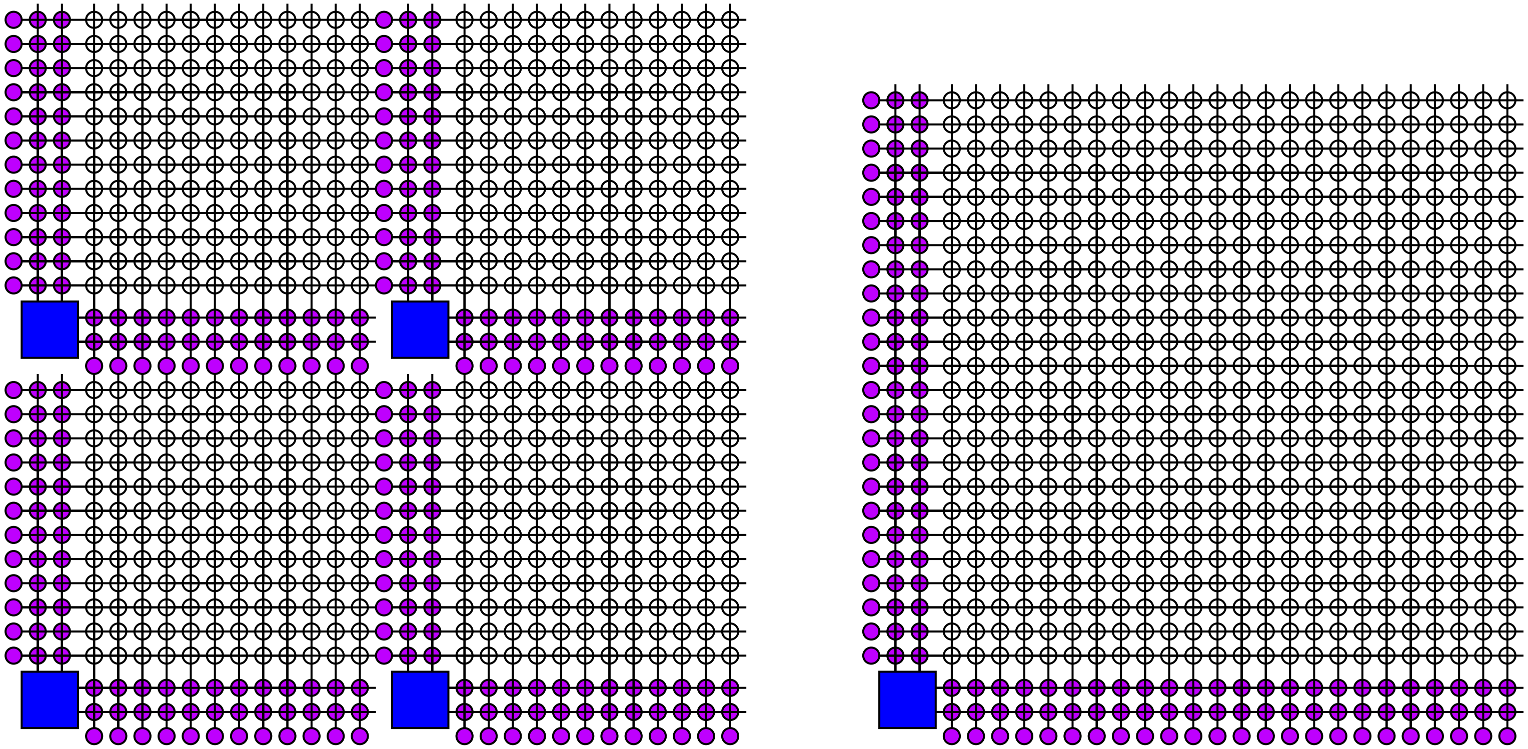


$k = 4$, $n = 2$, $N_{LUT} = 4096$

Assume PDF for p_{design} uniformly distributed:
$0.4 \leq p_{design} \leq 0.8$.

Sea of Gates

Once interconnect dominates **and** one expects a range of p's for subcomponents $\Rightarrow$ focus on using interconnect



If most of the interconnect resources are usable as needed (not dedicated to a LUT), it doesn't hurt to have gates which you ignore when p is high.

Review

- Can't afford full interconnect
- Can't make all interconnect local
- Rent's Rule captures typical locallity/structure in nets
- Rent parameter varies among computing structures
- Interconnect resources grow as $O(n^{2p})$
- Too much interconnect can be just as inefficient as too little
- (demanding full routability of all designs is not area wise)
- Time for a sea-of-gates philosophy for FPGAs?