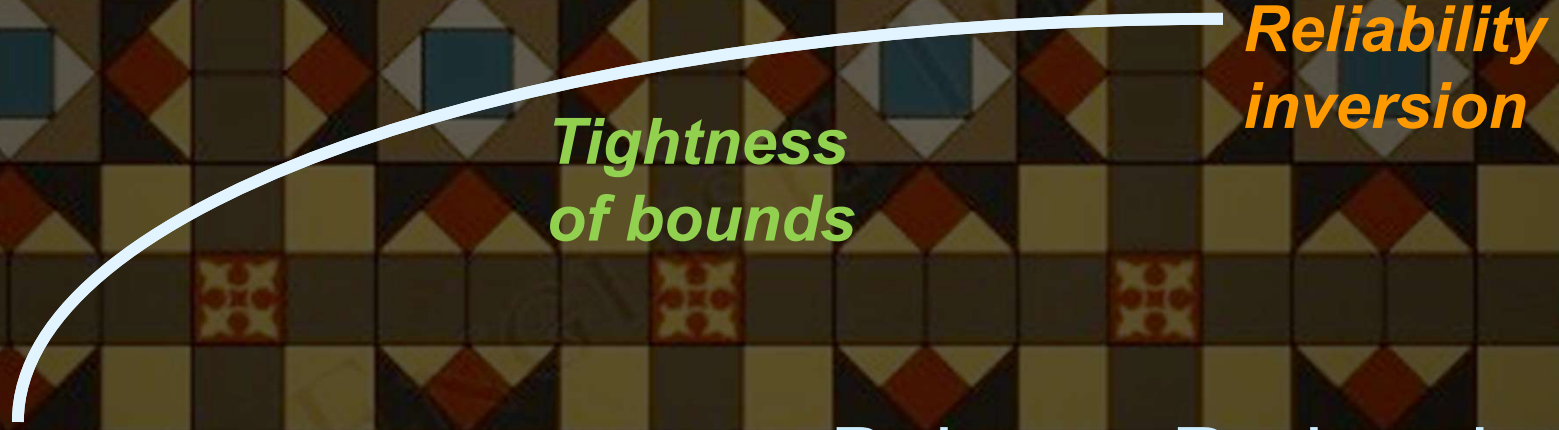
Reliability and Modelability Advantages of Distributed Switching for Reconfigurable 2D Processor Arrays



Behrooz Parhami
University of California
Santa Barbara

About This Presentation

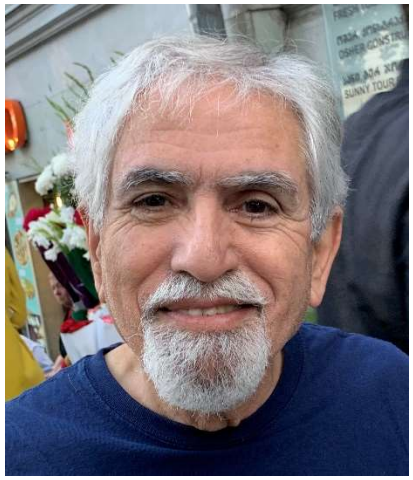
This slide show was first developed in November 2020 for a talk entitled "Reliability and Modelability Advantages of Distributed Switching for Reconfigurable 2D Processor Arrays," presented at the 11th Annual IEEE Information Technology, Electronics and Mobile Communication (Virtual) Conference.

Edition	Released	Revised	Revised	Revised
First	Nov. 2020			

Reliability and Modelability Advantages of Distributed Switching for Reconfigurable 2D Processor Arrays

Processor arrays have been used, either as the main computation engine or as special-purpose adjuncts, for a variety of applications requiring very high performance. As the size of such an array increases, the possibility of processor malfunctions, leading to loss of computational capabilities, can no longer be ignored. While many switching architectures and reconfiguration algorithms have been proposed for building processor arrays, modeling of their reliability has been inadequately addressed. In this paper, I study differences between 2D processor arrays with centralized and distributed switching, pointing to advantages of the latter in terms of reliability, regularity, modularity, and VLSI realizability. As important side results, I formulate the notions of reliability inversion (a less reliable system prevailing over a more reliable one due to modeling uncertainties) and modelability (the property of a system that makes it possible to derive tight reliability bounds, thus making reliability inversion less likely).

Speaker's Brief Technical Bio



Behrooz Parhami (PhD, UCLA 1973) is Professor of Electrical and Computer Engineering, and former Associate Dean for Academic Personnel, College of Engineering, at University of California, Santa Barbara, where he teaches and does research in the field of computer architecture: more specifically, in computer arithmetic, parallel processing, and dependable computing.

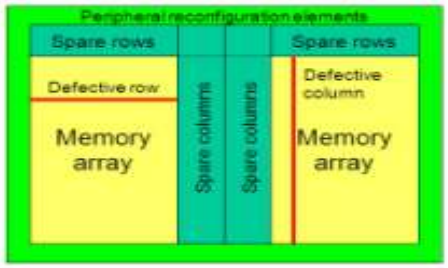
A Life Fellow of IEEE, a Fellow of IET and British Computer Society, and recipient of several other awards (including a most-cited paper award from *J. Parallel & Distributed Computing*), he has written six textbooks and more than 300 peer-reviewed technical papers. Professionally, he serves on journal editorial boards (including for 3 different *IEEE Transactions*) and conference program committees, and he is also active in technical consulting.

Reconfiguration for Yield and Reliability

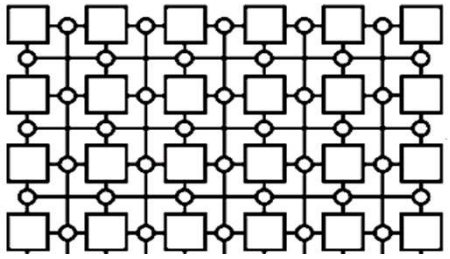
The IC yield problem: Chips with millions of components are sure to contain defective parts; Yield will go to zero without special provisions to circumvent defective elements

Operational fault tolerance: With highly complex SoCs, system MTTF will be low, unless faults can be tolerated

Memory arrays: Spare rows & columns and ECCs allow the memory chip to be used despite defective or faulty cells



Processor arrays: If in an 4×6 array, up to 2 faulty elements can be bypassed, we will have a 22-out-of-24 system

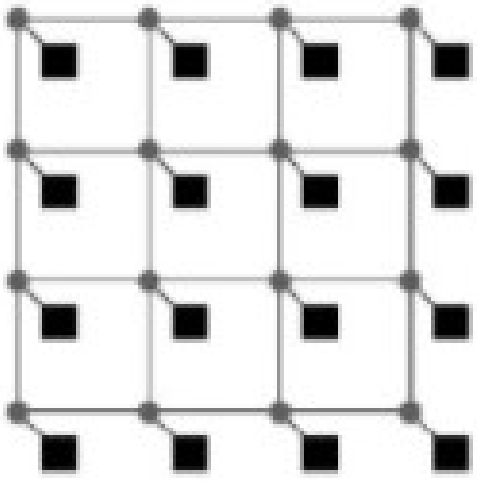


Processor Arrays: Motivation and Uses

Multiple processors, cores, or PEs per chip:
Achieving higher performance through parallel processing
Dissipating less power by distributing load to simpler nodes

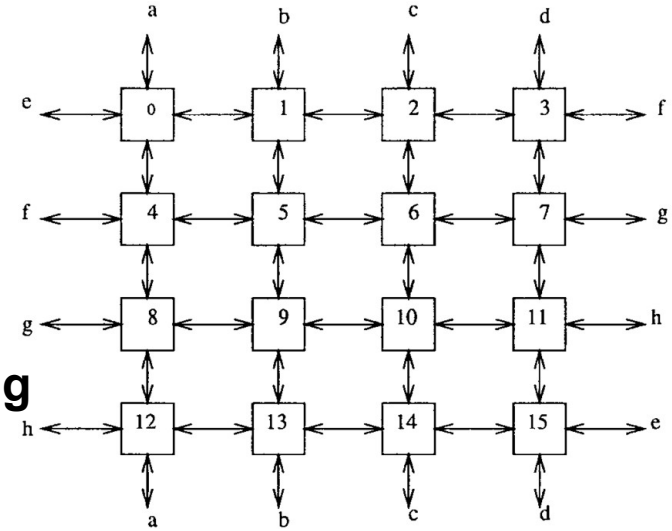
Example: AI / ML accelerators, such as Google's TPU

Example: Packet processing units within Internet routers



Array of Independent Processors

Array of Communicating Processors

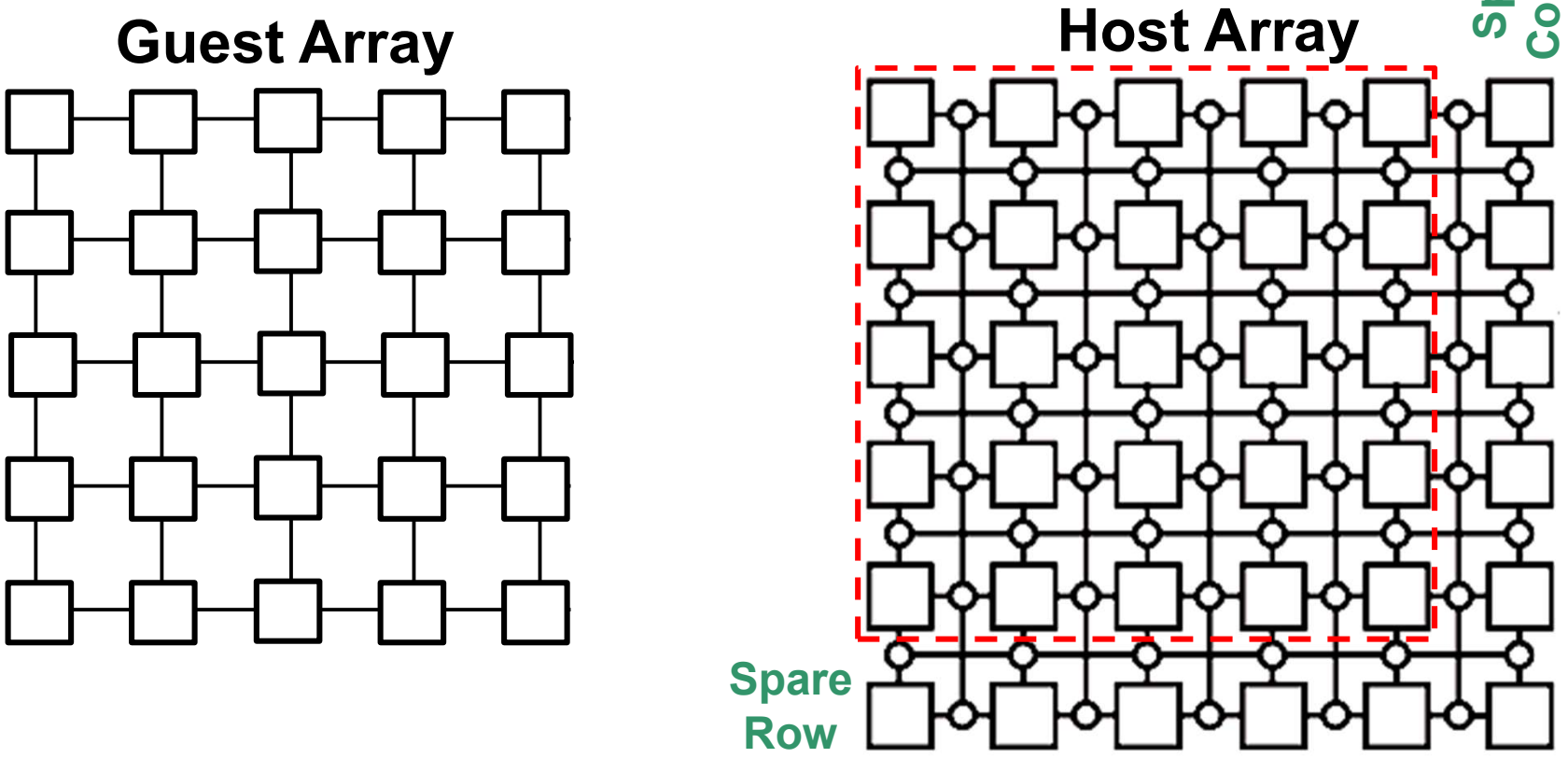


Processor Array: Larger than Needed

Array built with redundant resources

Spare row(s) and column(s)

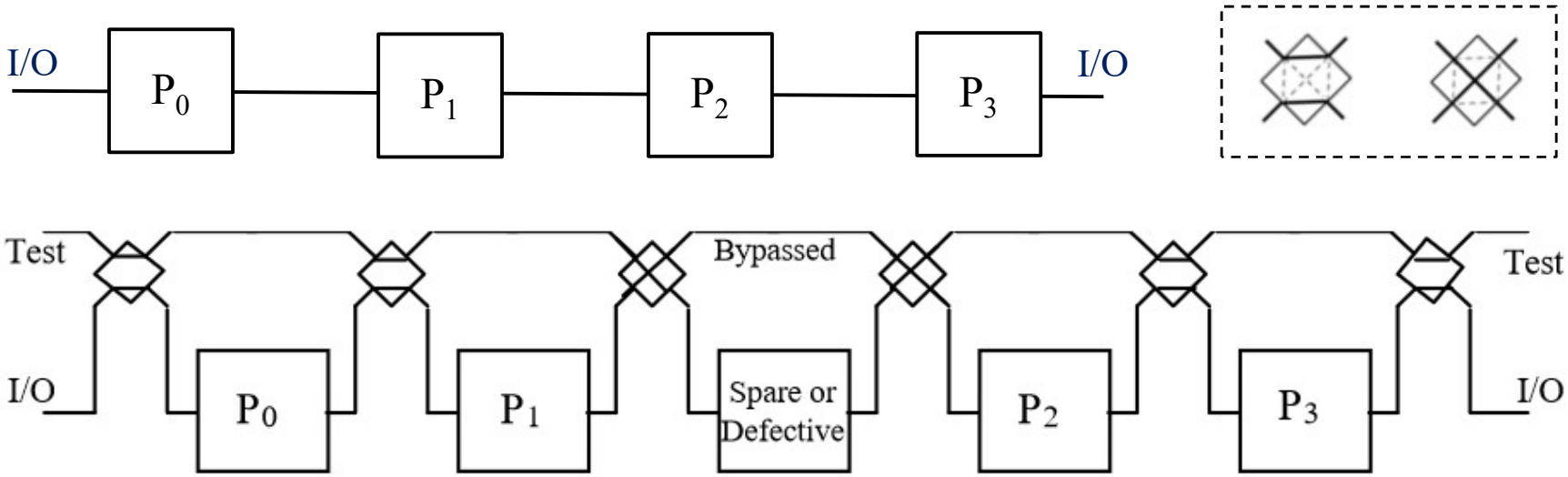
Embedded reconfiguration switches



One-Dimensional Processor Arrays

Linear (1D) array built with redundant resources
Spare processor (s)
Embedded reconfiguration switches

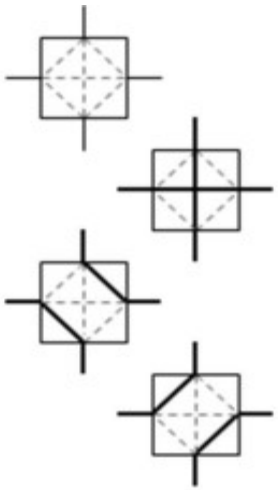
A track of 2-state switches enables processor bypassing
The switches are part of the system's hard core

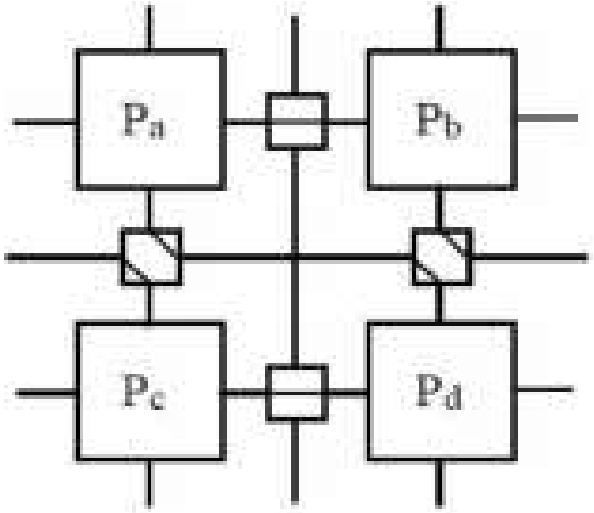


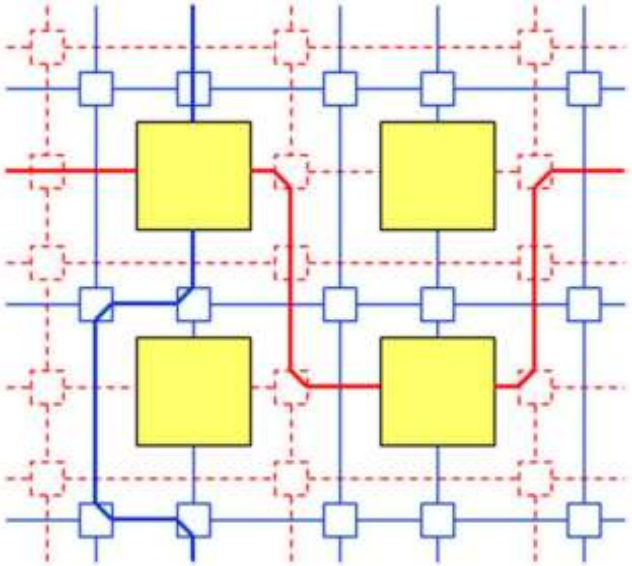
One-Track and Two-Track Switching

Many switch architectures have been proposed
One or two tracks of 3-state switches enable re-routing or processor bypassing within/between rows and columns

- One-track switches: Limited bypassing and robustness
- Two-track switches: More flexibility, plus fault tolerance







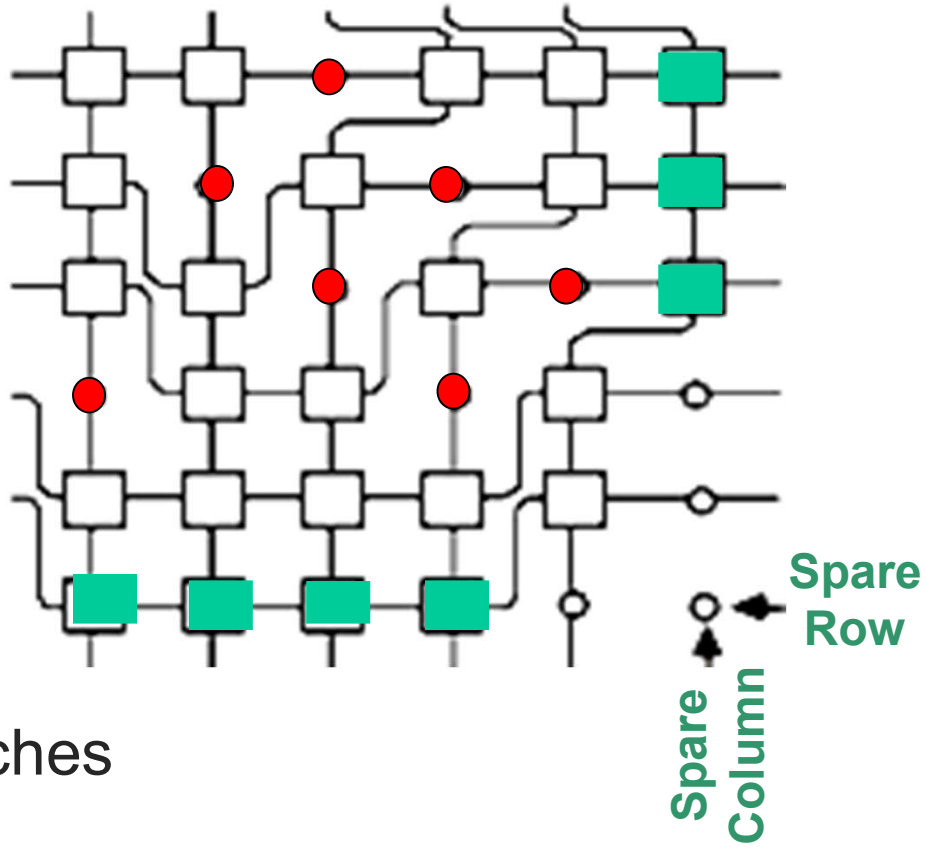
Circumvention of Defects or Faults

Salvaging a healthy array from an injured one
Example: Seven unavailable nodes replaced by spares

- Rows shifted downward
- Columns shifted rightward

In the worst case, only 2 bad nodes can be circumvented and even one bad switch dooms the system

Model as 34-out-of-36 system of nodes, placed in series with a system composed of 60 switches



Modeling with Centralized Switching

Processor failure rate

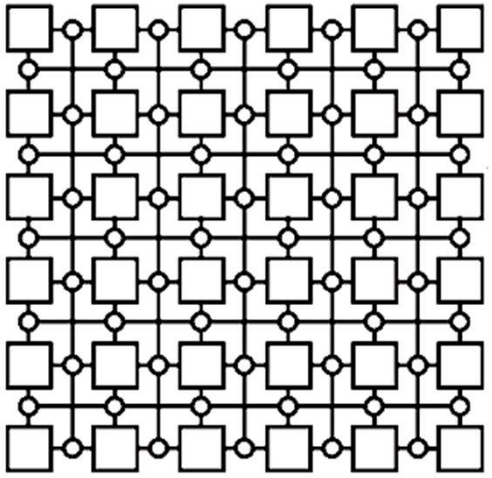
Switch failure rate

$$\text{Module/Processor reliability} = r = e^{-\lambda t} \quad (1)$$

$$\text{Overall switching reliability} = e^{-(60\sigma)t} \quad (2)$$

$$\text{System reliability} = e^{-(60\sigma)t} R_{34\text{-out-of-}36}(r) \quad (3)$$

$$\begin{aligned} R_{34\text{-out-of-}36}(r) &= r^{36} + 36r^{35}(1-r) + (36\times 35/2)r^{34}(1-r)^2 \\ &= r^{34}[r^2 + 36r(1-r) + 630(1-r)^2] \\ &= r^{34}[595r^2 - 1224r + 630] \\ &= r^{34}[1 + (1-r)(629 - 595r)] \end{aligned} \quad (4)$$



Very loose reliability bound due to the extremeness of worst-case assumptions

Unreliability: Centralized Switching

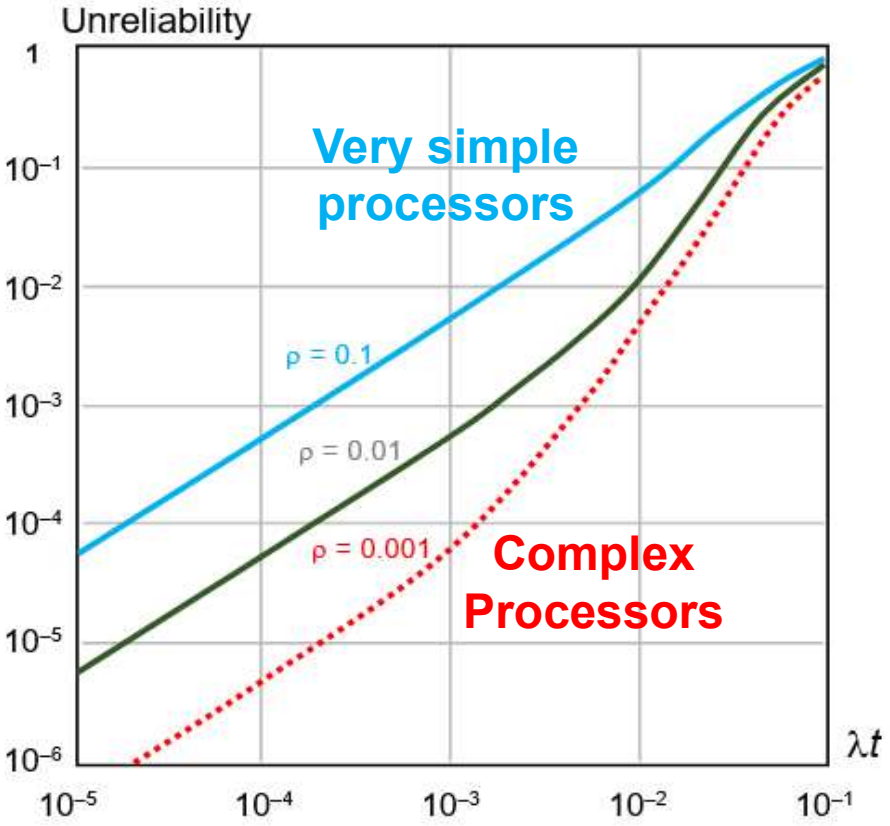
Note the log-log scale

Because unreliability is plotted, lower values are better

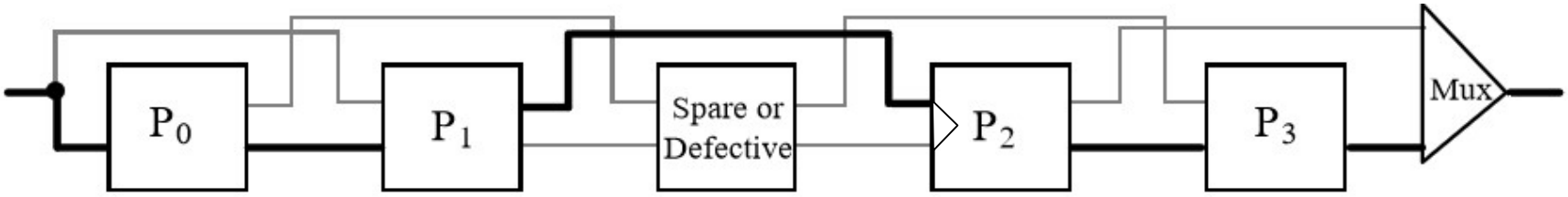
The value of ρ depends on processor complexity

The more complex the processors, the higher our reliability gain over a non-redundant 5×5 array and the better the quality of the lower bound

λ = Processor failure rate
σ = Switch failure rate ($\sigma << \lambda$)
$\rho = \sigma / \lambda$ ($<< 1$)



Distributed Switching: 1D and 2D

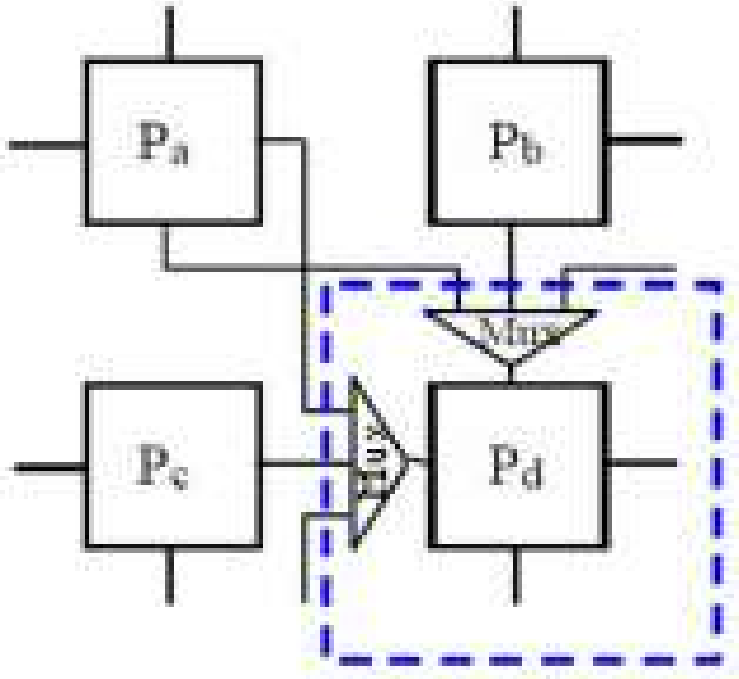


Healthy processors choose their neighbors

Switch defect / fault no longer critical

No need for highly pessimistic assumptions

Bound quality improves



Modeling with Distributed Switching

$$\text{Module reliability} = r' = e^{-(\lambda+\alpha\sigma)t} \quad (5)$$

$$\text{System reliability} = R_{34\text{-out-of-}36}(r') \quad (6)$$

Tighter reliability bound
due to less-extreme worst-case assumptions

TABLE I. LOW SENSITIVITY OF RELIABILITY TO VARIATIONS IN α

$\lambda t \rightarrow$	0.0001	0.0010	0.0100	0.1000
$R(\alpha = 1)$	1.000,000	0.999,993	0.994,343	0.314,752
$R(\alpha = 2)$	1.000,000	0.999,993	0.994,189	0.308,424
$R(\alpha = 3)$	1.000,000	0.993,992	0.994,031	0.302,192

Unreliability: Distributed Switching

Note the log-log scale

Because unreliability is plotted, lower values are better

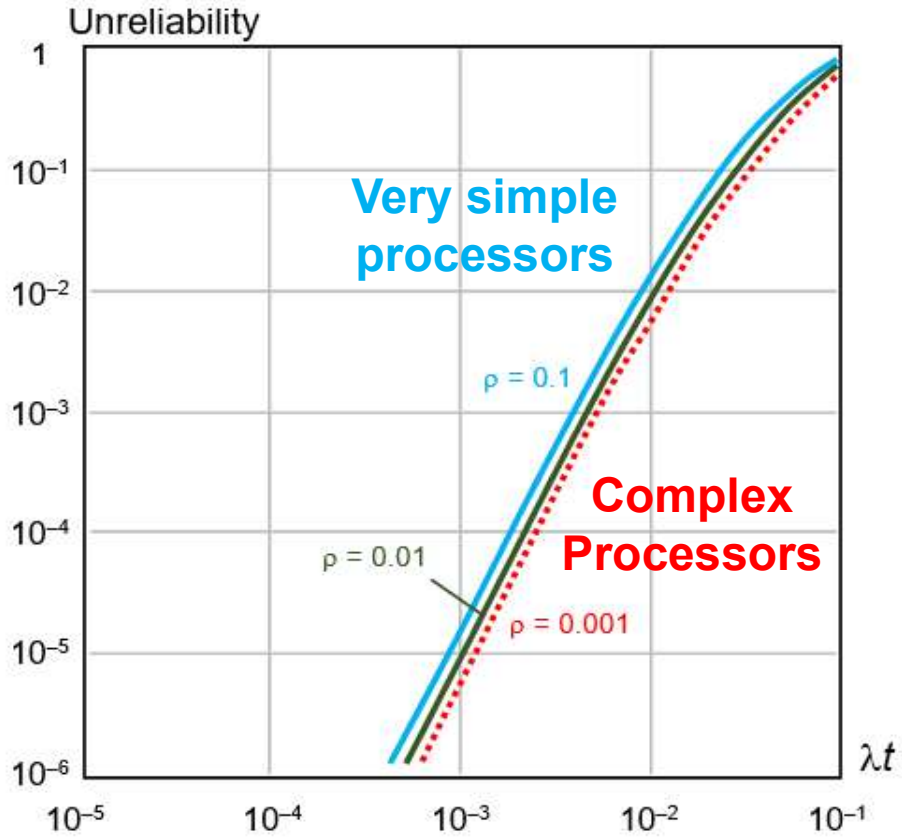
The value of ρ depends on processor complexity

Complex processors lead to greater reliability gain over non-redundant 5×5 array, but the difference is less pronounced here

λ = Processor failure rate

σ = Switch failure rate ($\sigma << \lambda$)

$\rho = \sigma / \lambda$ ($<< 1$)



Comparative Unreliabilities

Parameter values

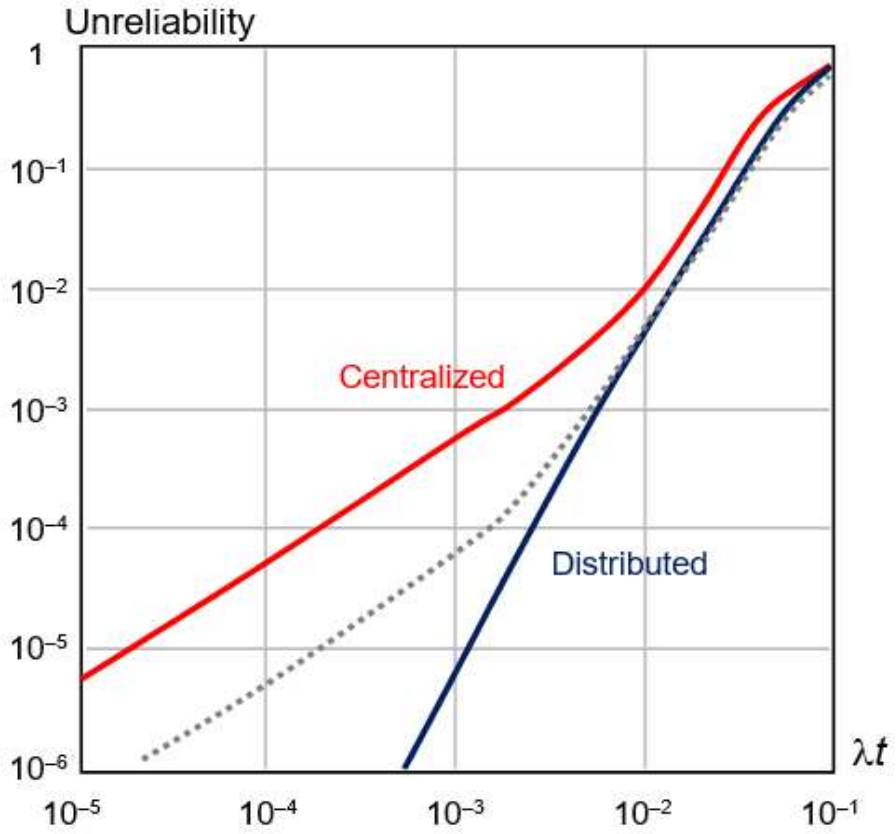
$\rho = 0.01$
$\alpha = 2$ (distribution overhead)

Low sensitivity to α

Advantage of distributed switching greatest for λt values of practical interest

Dashed curve assumes a less pessimistic centralized switching model, but does not guarantee lower bound

λ = Processor failure rate
σ = Switch failure rate ($\sigma << \lambda$)
$\rho = \sigma / \lambda$ ($<< 1$)



Reliability Inversion

Actual reliability of a system is unknowable

Experimental evaluation needs 1000s of units and many years

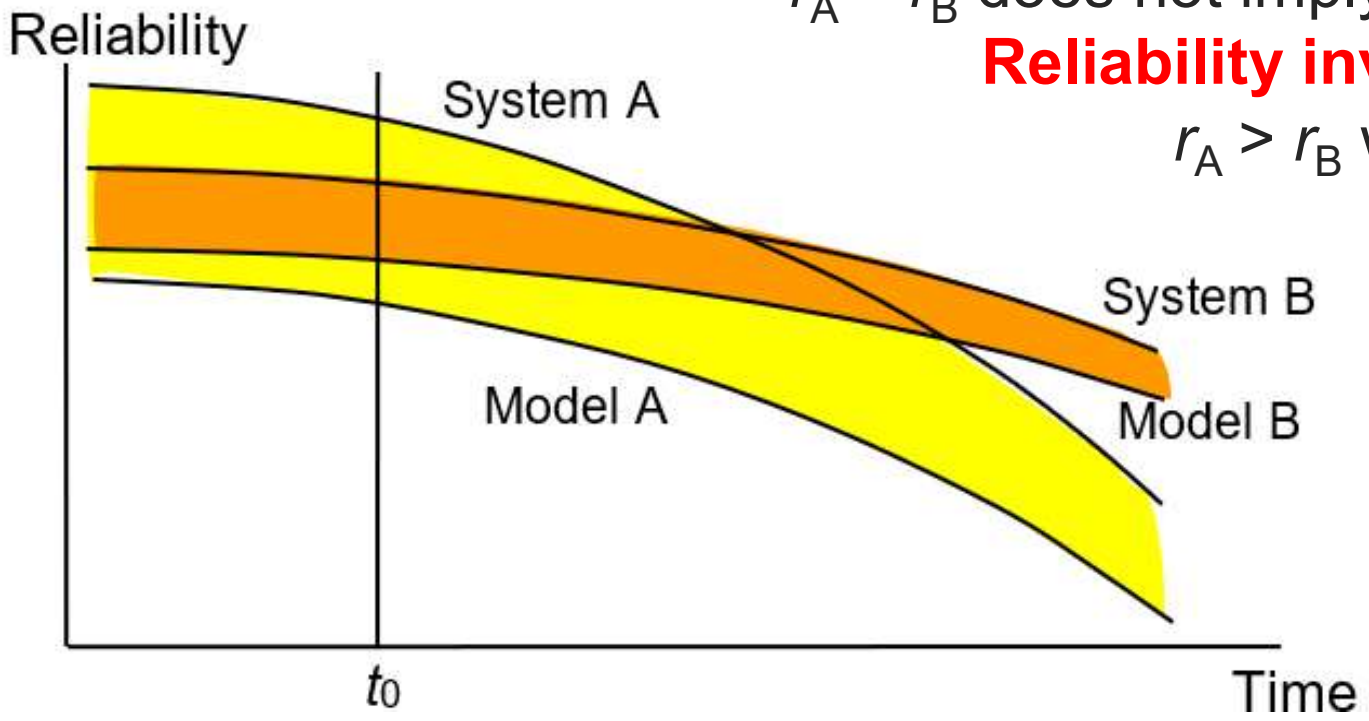
We thus substitute lower bounds for actual values

Lower bounds: $r_A < R_A$, $r_B < R_B$

$r_A > r_B$ does not imply $R_A > R_B$

Reliability inversion:

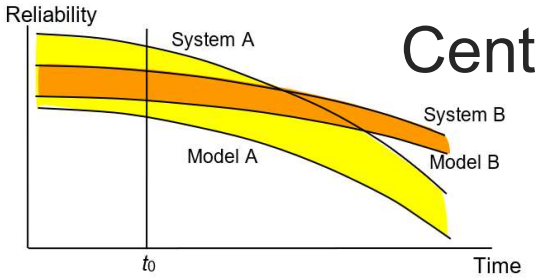
$r_A > r_B$ with $R_A < R_B$



Conclusions and Future Work

Distributed vs. Centralized reconfiguration switching

Good example for the importance of **modelability**



Centralized switching can in fact be better

Hence, possible **reliability inversion**

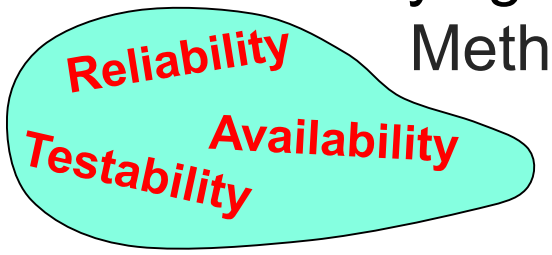
Details: *IEEE Computer*, 6/2020

Our results were shown for a 5 × 5 processor array

Larger arrays improve the advantage of distribution

Extensions: Place of modelability among other -ilities

Quantifying modelability, like reliability, testability, …



Methodologies of designing for modelability

Effect of switching-architecture variations

Application to other system types

Questions or Comments?

parhami@ece.ucsb.edu
http://www.ece.ucsb.edu/~parhami/