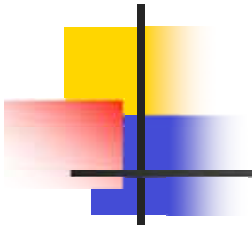
IFIP WG 10.4 53rd Meeting

Natal, Brasil

Research Report

Kishor S. Trivedi
Dept. of Electrical & Computer Engineering
Duke University
Durham, NC 27708
kst@ee.duke.edu
www.ee.duke.edu/~kst

February 25, 2008



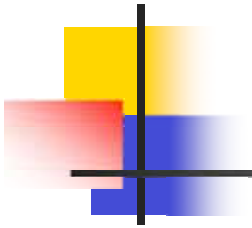
Two Recent Projects

- Reliability Analysis of Boeing 787 Current Return Network (CRN) for FAA Certification
- User-perceived reliability of SIP protocol on High Availability IBM WebSphere/BladeCenter

Boeing 787 CRN

- Work done with Dazhi Wang, Tilak Sharma, A. Ramesh and others at Boeing
- Modeled as a reliability graph or relgraph
- Also known as the s-t connectedness problem
- Or as the Network reliability problem
- A simple, series-parallel version is known as the reliability block diagram (RBD)

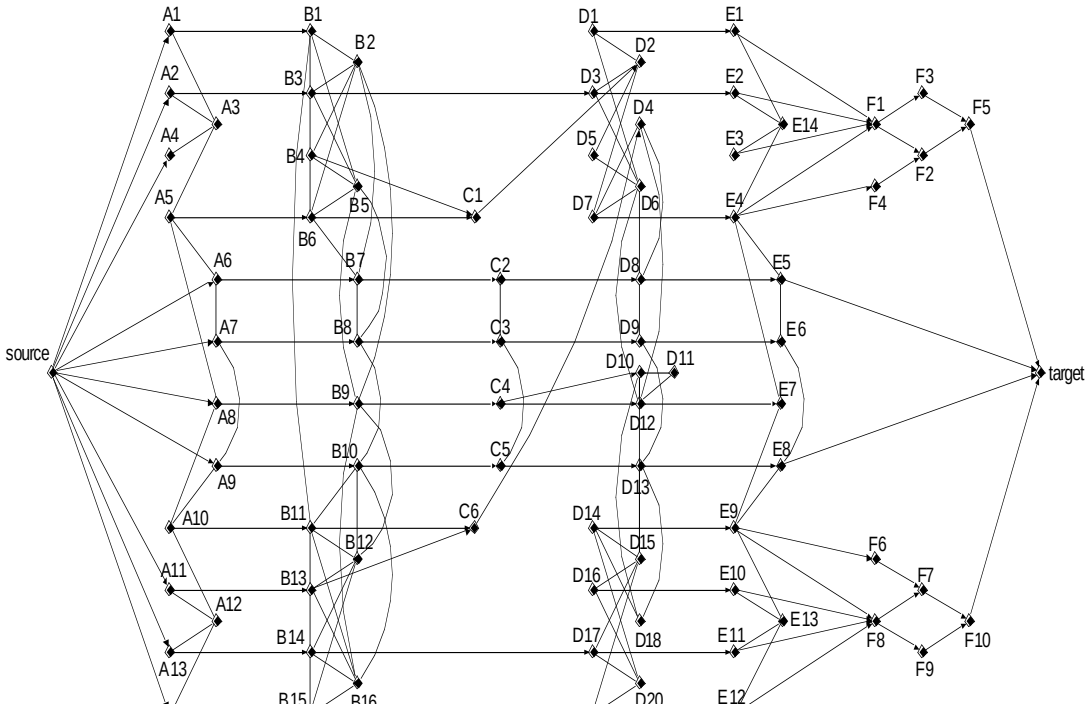
It is a combinatorial on non-state-space model type which are thought of not being plagued by the largeness problem



Reliability Graph

- Consists of a set of nodes and edges
- Edges represent components that can fail
- Two distinguished nodes:
  - Source and target nodes
- System fails when no path from source to target
- This model type is less commonly found in software packages compared with fault tree

Current Return Network Modeled as a Reliability Graph



DUKE

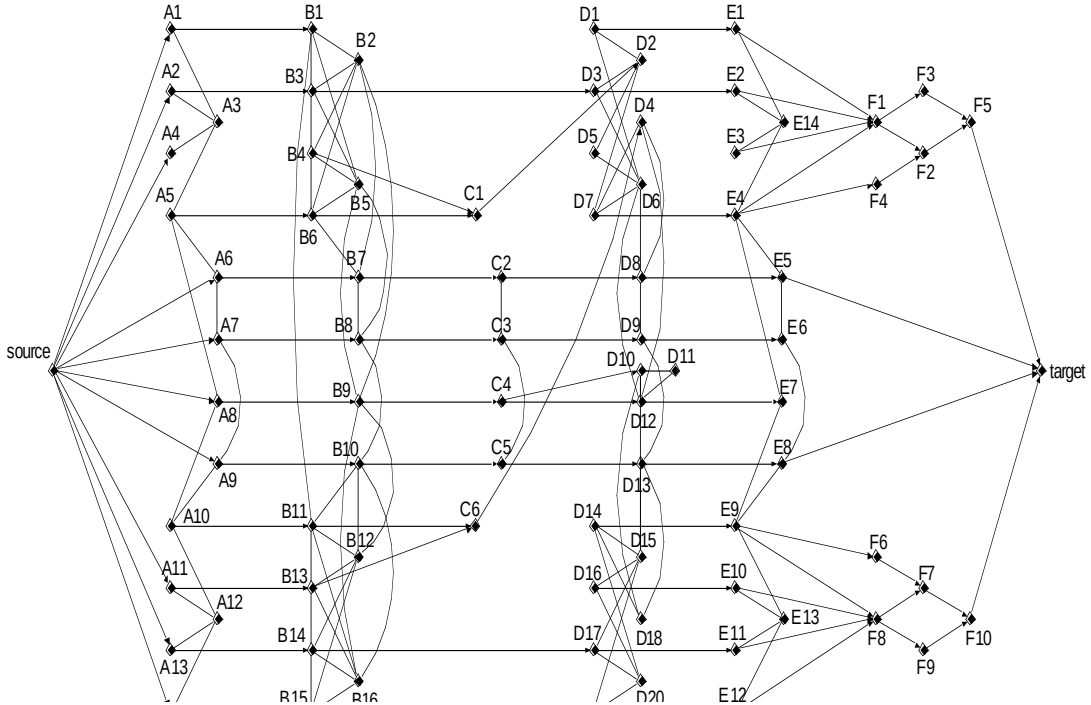
Relgraph solution methods

- Factoring or conditioning
  - Not easy to decide which link to factor on
  - Repeated factoring needed
- Find all minpaths followed by sdp (sum of disjoint products
- Bdd (binary decision diagrams)-based method

- Last two have been implemented in SHARPE
- Initial run by SHARPE could not solve the problem!

Too many minpaths

- Combinatorial models may also face largeness problem



node	#paths
$E_7 \rightarrow$target	40
$D_{12} \rightarrow$target	143140
$C_4 \rightarrow$target	308055
$B_9 \rightarrow$target	21054950355
$A_8 \rightarrow$target	461604232201
source $\rightarrow$ target	$4248274506778 \approx 4 \times 10^{12}$

Number of paths from source to target

- Compute reliability bounds instead of exact reliability

Our Approach

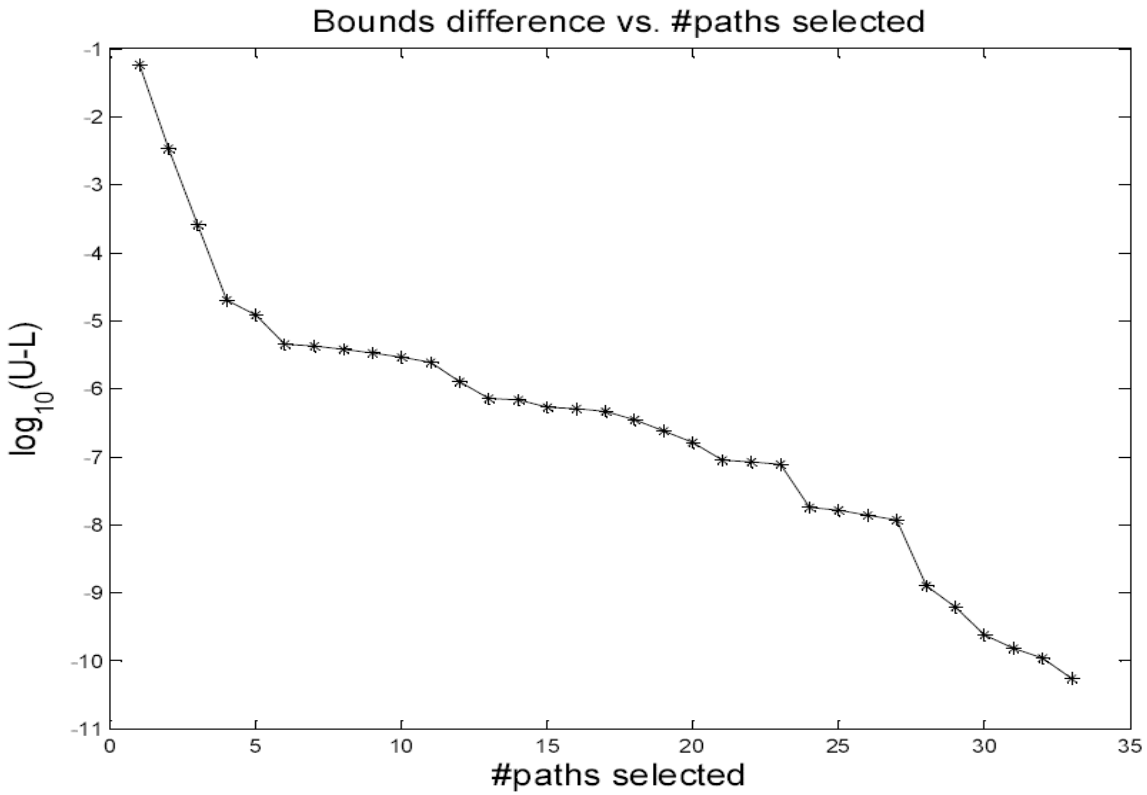
- Developed a new efficient algorithm for (un)reliability bounds computation and incorporated in SHARPE

runtime	20 seconds	120 seconds	900 seconds
upper bound	1.1460365721e-008	1.0814324701e-008	1.0255197263e-008
lower bound	1.0199959877e-008	1.0199959877e-008	1.0199959877e-008

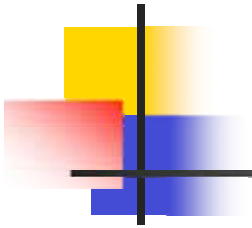
- Boeing has decided to file a patent on the algorithm

- Satisfying FAA that SHARPE development used DO-178 B software standard was the hardest part

Numerical Results



Bounds Difference vs. Paths/Cutsets Selected



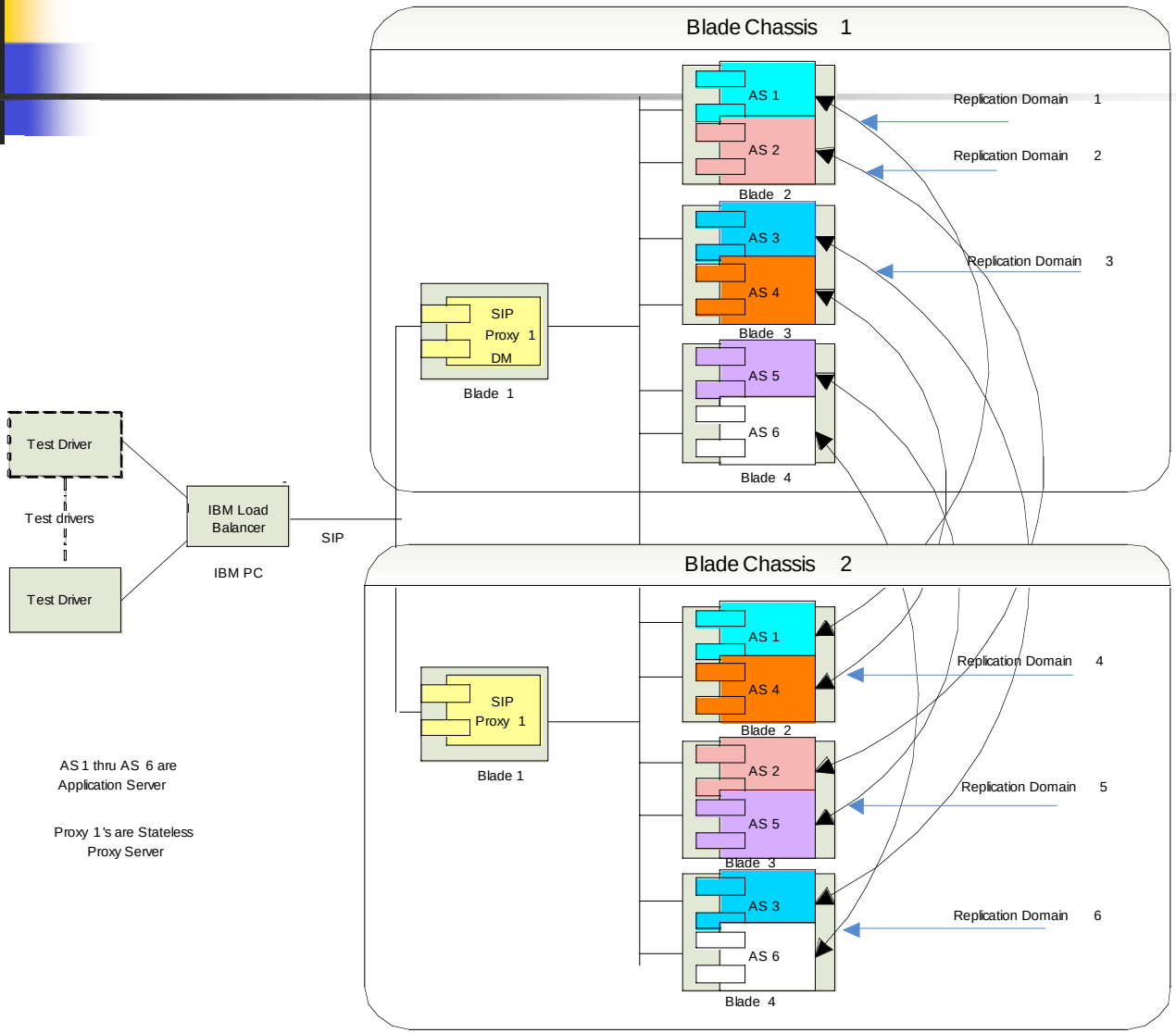
Modeling SIP Application Server Dependability

Kishor Trivedi

Contributors: Dazhi Wang, Jason Hunt, Andy Rindos, and many others at IBM and at TELCO customer

Dept. of Electrical & Computer Engineering, Duke University

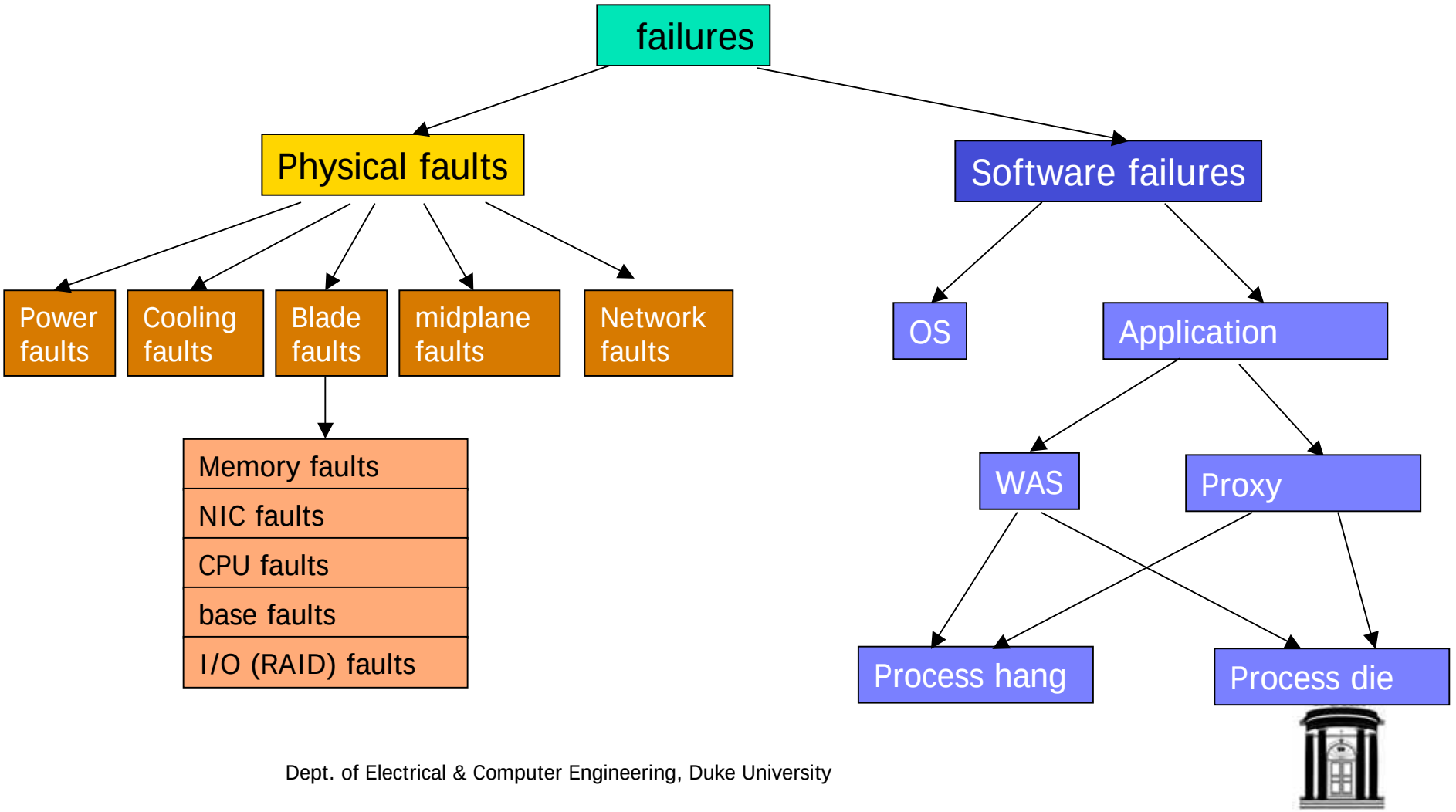
Hardware/Software Configuration

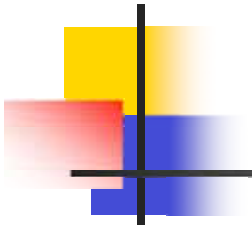


Replication domain	Nodes
1	A, D
2	A, E
3	B, F
4	B, D
5	C, E
6	C, F

DUKE

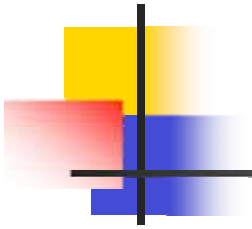
Failures Incorporated in Models





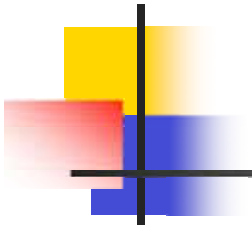
Our Contributions (1)

- Developed a very comprehensive availability model
  - "Discovered the Software failure/recovery architecture
  - Hardware and software failures
  - Hardware and Software failure-detection delays
  - Software Detection/Failover/Restart/Reboot delay
  - Escalated levels of recovery
    - Automated and manual restart, failover, reboot, repair
  - Imperfect coverage (detection, failover, restart, reboot)



Our Contributions (2)

- Developed a new (first?) method for calculating DPM (defects per million calls) (IBM is filing for a patent on this algorithm)
  - Taking into account interactions between call flow and failure/recovery & Retry of messages
- Many of the parameters collected from experiments
- Detailed sensitivity analysis to find bottlenecks and give feedback to designers
- This model made the sale of this system to the Telco customer



Parameterization

- Hardware/Software Configuration parameters
- Hardware component MTTFs
- Hardware/Software Detection/Failover/Restart/Reboot times
- Repair time
  - Hot swap, multiple components at once, field service travel time
- Software component MTTFs (experiments have started for this)
  - OS, WAS, SIP/Proxy
- Coverage (Success) probabilities
  - Detection, restart, failover, reboot, repair
- Validation (?)

Thank You!