TECHNISCHE UNIVERSITÄT
KAISERSLAUTERN

0101seda010100
software engineering dependability

Safety and Reliability of Embedded Systems

(Sicherheit und Zuverlässigkeit eingebetteter Systeme)

Fault Tree Analysis
Current Research Issues

Content

- Current Research Issues
- Original Contributions from Researchers
  - Prof. John Andrews, Loughborough University, United Kingdom
  - Prof. Joanne Bechta Dugan, University of Virginia, USA
  - Prof. Janusz Górski, Gdansk University of Technology, Poland
  - Dr. Bernhard Kaiser, formerly Fraunhofer IESE, Germany
  - Prof. Antoine Rauzy, Institut de Mathématiques de Luminy, France
  - Prof. Anders Ravn, Aalborg University, Denmark
  - Prof. Wolfgang Reif, Universität Augsburg, Germany
  - Prof. Kishor Trivedi, Duke University, USA

... show their view on fault trees and their research approaches

Current Research Issues

- Formalization and Extension of Fault Trees
  - Formal semantics of FTs, events and gates
  - Checking FTs for completeness and consistency
  - Temporal order and real time
  - Multistate components
- Analysis Techniques
  - Performance, accuracy and usability issues with BDDs
  - Other techniques (Markov, Petri Net...) where BDD is not applicable
- Integration with Other Techniques
  - Automatic FTA generation from SW/HW documents
  - Integration with other safety analysis techniques
  - Integration with formal methods
  - Integration into a development / safety analysis process

Professor John Andrews
Department of Mathematical Sciences
Loughborough University
Loughborough
Leicestershire, LE11 3TU
(01509) 227286.

email: J.D.Andrews@lboro.ac.uk

Fault Tree Analysis - Methodology

- Increasing the FTA efficiency / accuracy
  - Analysis by BDD
    - Procedures to permit all qualitative and quantitative analysis currently performed by conventional FTA methods to be performed utilising the inherent efficiency and accuracy of the BDD
    - Development of efficient BDD construction methods
      - Variable ordering schemes / direct FT gate transformation
  - Non-coherent fault tree analysis
    - Extension of analysis methods for this type of assessment (useful for Event Tree Analysis)
    - Development of appropriate component importance measures to account for the contribution to the system failure of both failed and functioning component states
  - Automatic generation of the fault tree from the system schematic

provided by:
John Andrews, Loughborough University. J.D.Andrews@lboro.ac.uk

Fault Tree Analysis - Applications

- Increasing the range of applications of the FTA method
    - System Reliability Analysis
        - Phased Mission
            - A mission made up of several consecutive phases all of which must be successful for mission success (repairable and non-repairable)
        - Dependency Modelling
            - Integrate FTA with Markov and simulation modelling techniques to handle dependencies (such as the time limited dispatch of commercial aircraft – the aircraft is permitted to take off carrying known faults for a limited period of time)
        - Optimisation
            - Embedding the system analysis within an optimisation process (usually a genetic algorithm) to yield the best rather than adequate level of performance within the limitations placed on the resources available

provided by:
John Andrews, Loughborough University. J.D.Andrews@lboro.ac.uk

Fault Tree Analysis – Other Uses

- Increasing the range of applications of the FTA method
  - Systems Diagnostics and Prognostics
    - System Diagnostics
      - Use fault trees which develop causes of sensor deviations to determine if component faults exist on a system and if so what they are
    - System Prognostics
      - Integrated with the fault diagnostics approach, it is used to determine the likelihood of a mission success when faults occur. When an unacceptable likelihood is predicted the mission or the system can be reconfigured (applications such as UAVs – unmanned air vehicles)

provided by:

John Andrews, Loughborough University. J.D.Andrews@lboro.ac.uk

Joanne Bechta Dugan
Professor of Electrical, Computer &
Systems Engineering
University of Virginia
jbd@Virginia.edu

(434)982-2078
FAX:(434)924-8818

351 McCormick Road
PO Box 400743
Charlottesville, VA 22904-4743

Fault Tree Analysis – Other Uses

- Fault tree analysis (FTA) is a widely accepted methodology for reliability analysis and provides core functionality to PRA (Probabilistic Risk Assessment)
- However, FTA cannot model failure events that depend on the order in which components fail
- Dynamic fault trees (DFT) extend FTA to allow accurate analysis of computer-based systems characterized by
  - complex redundancy management
  - spares (cold, warm, pooled)
  - functional and sequence dependencies
  - hardware and software components
  - imperfect coverage and other common cause failures
  - phased missions

provided by:
Joanne Bechta Dugan, University of Virginia, jbd@Virginia.edu

DFT: Dynamic Fault Tree Analysis

- The DFT methodology supports modularization
  - Overall DFT model is **automatically** divided into modules that can be solved separately
  - Modules are classified as static (containing traditional gates) or dynamic (containing at least one dynamic gate)
  - Separate modules are solved using most appropriate means and results are synthesized **automatically**

provided by:

Joanne Bechta Dugan, University of Virginia, jbd@Virginia.edu

Static Fault Trees

- Combinatorial model (models combinations of events)
  - AND gates
  - OR gates
  - K-of-M gates
- New approach for solution: BDD (Binary Decision Diagrams)
- Advantages
  - Exact analysis without cutsets
  - Can include repeated events
  - Can include coverage modeling
  - Fast solution for very large models
- Disadvantage
  - Static model: cannot include sequence dependencies

provided by:
Joanne Bechta Dugan, University of Virginia, jbd@Virginia.edu

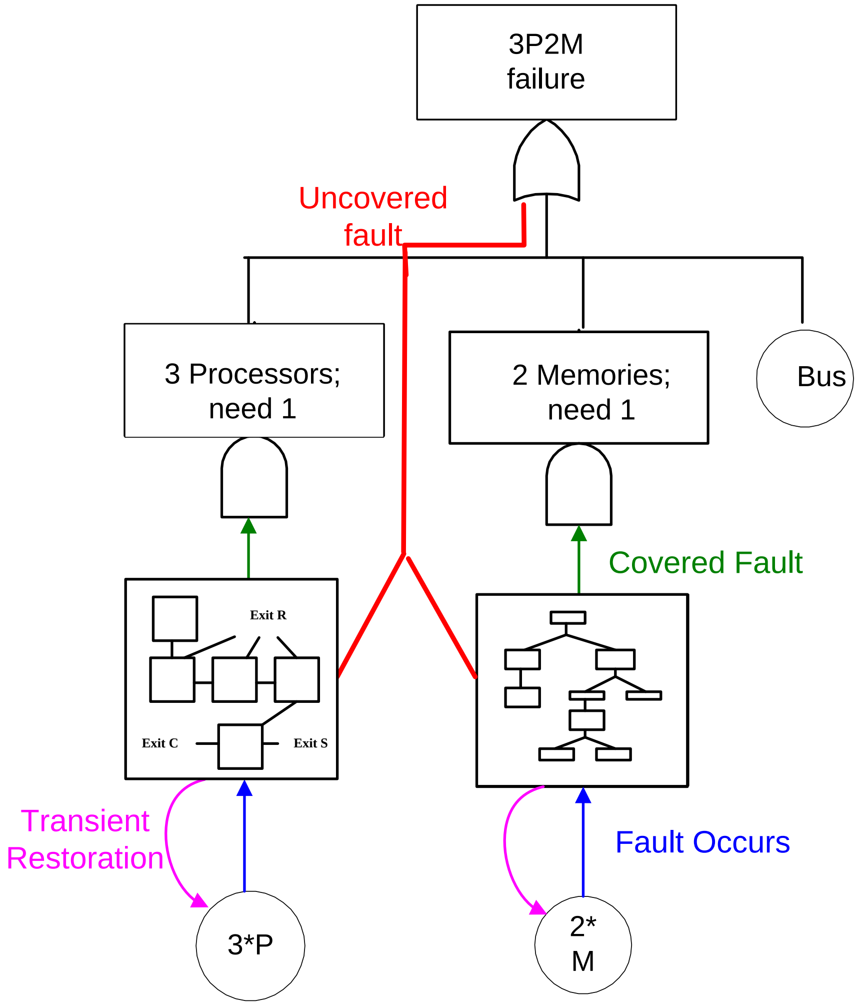
Dynamic Fault Trees

- Include special constructs for modeling sequence dependencies
  - functional dependencies
  - hot, warm and cold spares
  - priority-AND
  - sequence enforcing
- Solution: convert to Markov chain
- Advantages
  - easier to use fault tree than Markov model directly
  - can model dynamic redundancy, shared pools of spares, etc
- Disadvantage
  - state space explosion -- worst case exponential in number of basic events

provided by:
Joanne Bechta Dugan, University of Virginia, jbd@Virginia.edu

Coverage Modeling

- Adaptive (computer-based) systems can exhibit multiple failure modes
- *Covered* (benign) failure can be handled automatically
  - error is detected and located
  - switch in spare or bypass faulty component
  - system can continue operation without manual intervention
- *Uncovered* failure is globally malicious
  - undetected error escapes from embedded system
  - faulted component cannot be disabled
  - malicious behavior confuses recovery procedures
- System dependability measures are very sensitive to coverage
- Good techniques exist for incorporating coverage into static and dynamic fault trees



provided by:

Joanne Bechta Dugan, University of Virginia, jbd@Virginia.edu

Unique Features in our* DFT Methodology

- Modularization allows solution of large systems
- Combines Markov analysis with BDD analysis automatically
- Automatic generation of Markov model or BDD
- Coverage modeling for computer-based systems
- Common cause failure analysis is modeled implicitly (efficiently)
- Sensitivity analysis for static and dynamic models

*Joint work with Kevin Sullivan, Dept. Computer Science, University of Virginia and David Coppit, Dept Computer Science, The College of William & Mary.

provided by:
Joanne Bechta Dugan, University of Virginia, jbd@Virginia.edu

Unique Features in our* DFT Methodology

- Phased mission analysis for both static & dynamic models
- Exact results (no hidden approximations)
- Uncertainty analysis for both static & dynamic trees
- Diagnostic support to help determine cause of failure given symptoms and partial information
- DFT model has been formally specified* (i.e. in Zed) to ensure that subtle interactions between gates are handled properly

*Joint work with Kevin Sullivan, Dept. Computer Science, University of Virginia and David Coppit, Dept Computer Science, The College of William & Mary.

provided by:
Joanne Bechta Dugan, University of Virginia, jbd@Virginia.edu

FTA
Formalization and analysis of timing properties

Prof. Janusz Górski
jango@pg.gda.pl
Department of Software Engineering
Gdansk University of Technology
Gdansk, Poland

Approach: event structures and duration calculus

Initiation of the research

- Górski J: *Towards a common formal semantics base for safety description model*. EUREKA Project SEW 263, Rep. SDM/JG/01, 1990.
- Górski J: *Interfacing fault trees to formal methods.* EUREKA Project SEW 263, Rep. SDM/JG/03, 1990.

Continuation of the research - Publications

- Bloomfield R.E., Chang J. H., Górski J.: *Towards a Common Safety Description Model.* Proc. SAFECOMP'91, (J.F. Lindeberg, Ed.), Pergamon Press, 1991, pp. 1-6
- Górski J., *Extending Safety Analysis techniques with Formal Semantics*, in *Technology and Assessment of Safety-Critical Systems* (F.J. Redmill and T. Anderson, eds.), Springer-Verlag, 1994, pp. 147-163
- Górski, J. and Wardziński, A., *Formalizing Fault Trees,* Safety Critical Systems Symposium, Brighton (UK), February 1995, Springer Verlag, 1994. pp. 311-327
- Górski, J. and A. Wardziński, *Formalizing Fault Trees*, in *Achievement and Assurance of Safety,* (F Redmill and T Anderson, eds.), Springer Verlag, 1995, pp. 311-327
- Górski, J., and A. Wardziński, *Deriving Real-Time Requirements for Software from Safety Analysis*, 8th EUROMICRO Workshop on Real-Time Systems, L'Aquila (Italy), June 12-14, 1996, IEEE Press, 1996, pp. 9-14
- Górski, J. and Wardzinski, A., *Timing Aspects of Safety Analysis,* in *Safer Systems, (*F Redmill and T Anderson Eds.), Springer Verlag, 1997, pp. 231-244

provided by:

Janusz Górski, Gdansk University of Technology, jango@pg.gda.pl

Approach: event structures and duration calculus

Doctoral Dissertation

- Andrzej Wardziński, *Fault Tree Analysis of Safety Realted Computer Systems* (in Polish), Faculty of Electronics, Telecommunications and Informatics, Gdansk University of Technologies, 1997 (Supervisor: Prof. Janusz Gorski)

Master Dissertations

- Jarmuż Piotr, *Safety analysis of real time computer systems* (in Polish), Franco-Polish School of New Information and Communication Technologies , 1996 (Supervisor: prof. J. Górski)
- Grzegorz Gołaszewski, *A tool supporting Fault Tree Analysis of real time requirements* (in Polish), Faculty of Electronics, Telecommunications and Informatics, Gdansk University of Technologies, 2004 (Supervisor: Prof. Janusz Gorski)

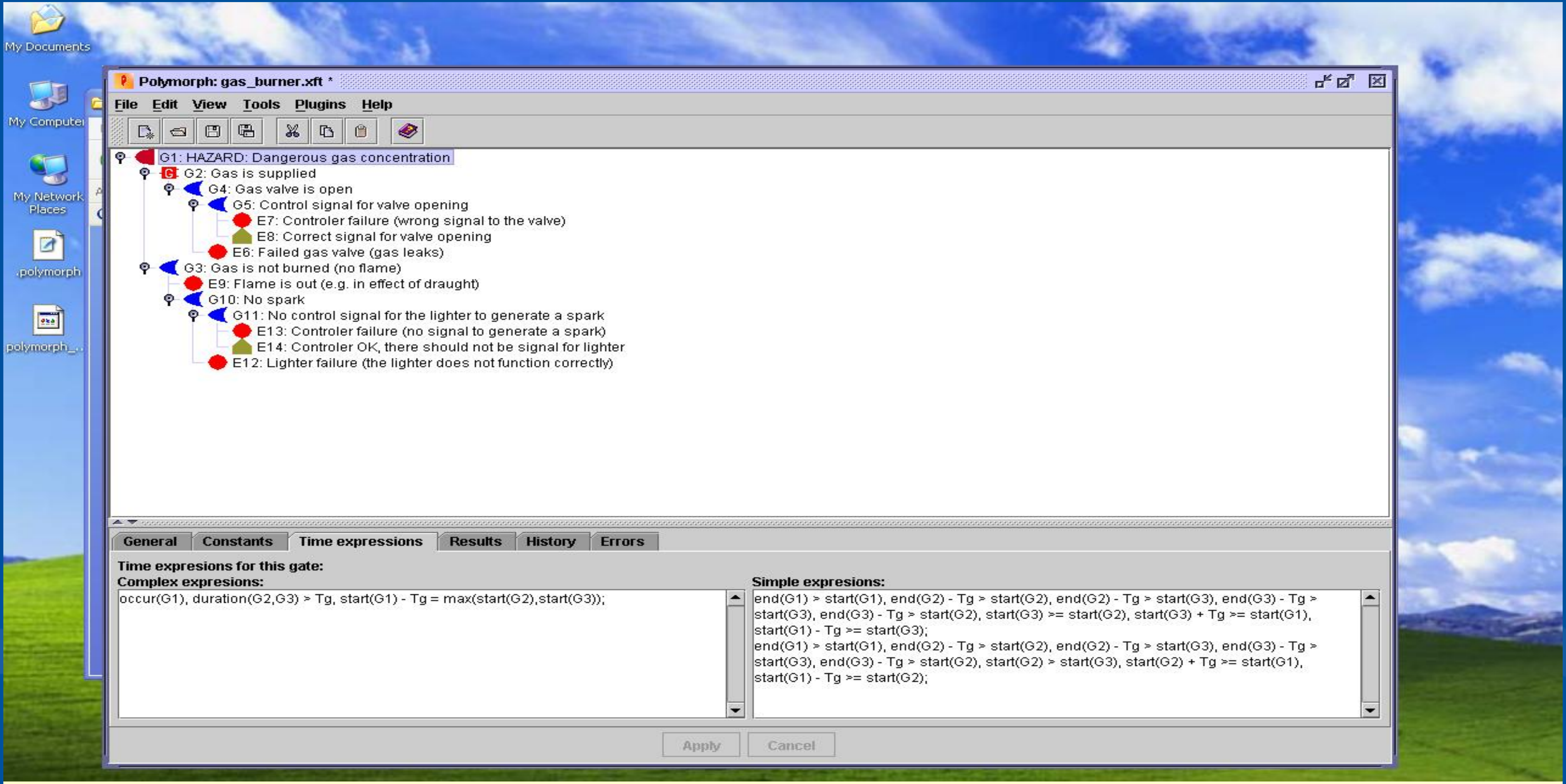
provided by:
Janusz Górski, Gdansk University of Technology, jango@pg.gda.pl

Approach: Petri Net modeling

- Górski J., Magott J. and Wardziński A., *Modelling Fault Trees Using Timed Petri Nets* (G Rabe, ed.), Springer Verlag, 1995, pp. 90-100
- Magott J, Skrobanek P, *A method of analysis of fault trees with time dependencies*, (Koorneef F and van der Meulen M eds) Springer-Verlag, 2000, 382-394
- J Magott, P Skrobanek, *Method of time Petri net analysis for analysis of fault trees with time dependencies*, IEE Proc. . Computers and Digital Techniques vol 149 no 6, 2002

provided by:

Janusz Górski, Gdansk University of Technology, jango@pg.gda.pl



G1: HAZARD: Dangerous gas concentration
- G2: Gas is supplied
  - G4: Gas valve is open
    - G5: Control signal for valve opening
      - E7: Controler failure (wrong signal to the valve)
      - E8: Correct signal for valve opening
    - E6: Failed gas valve (gas leaks)
- G3: Gas is not burned (no flame)
  - E9: Flame is out (e.g. in effect of draught)
  - G10: No spark
    - G11: No control signal for the lighter to generate a spark
      - E13: Controler failure (no signal to generate a spark)
      - E14: Controler OK, there should not be signal for lighter
    - E12: Lighter failure (the lighter does not function correctly)

Time expresions for this gate:

Complex expresions:

```
occur(G1), duration(G2,G3) > Tg, start(G1) - Tg = max(start(G2),start(G3));
```

Simple expresions:

```
end(G1) > start(G1), end(G2) - Tg > start(G2), end(G2) - Tg > start(G3), end(G3) - Tg >
start(G3), end(G3) - Tg > start(G2), start(G3) >= start(G2), start(G3) + Tg >= start(G1),
start(G1) - Tg >= start(G3);
end(G1) > start(G1), end(G2) - Tg > start(G2), end(G2) - Tg > start(G3), end(G3) - Tg >
start(G3), end(G3) - Tg > start(G2), start(G2) > start(G3), start(G2) + Tg >= start(G1),
start(G1) - Tg >= start(G2);
```

The tool POLYMORPH-FTA
supporting time analysis of Fault Trees
Prototype version of the tool will be available by the end of 2004

The Gas Burner case study

provided by:
Janusz Górski, Gdansk University of Technology, jango@pg.gda.pl

Extending FTA for software-controlled systems: Component Fault Trees, State-Event-Fault-Trees

Prof. Dr. Peter Liggesmeyer, Dr. Bernhard Kaiser*
[*Formerly] Fraunhofer Institute for
Experimental Software Engineering
Kaiserslautern, Germany
peter.liggesmeyer@iese.fraunhofer.de
Tel. +49 (631) 205-3449

www.iese.fraunhofer.de
www.essarel.de

Problems Addressed

- Many Embedded Systems go into Safety Critical Areas
  - ➔Safety and Reliability Analysis (e.g. by Fault Tree Analysis) is required
- Need for Compositional Techniques
  - ! Models must be attached to Components defined during Design
  - ! Component models need interfaces that allow integration
  - ☹ Traditional FT Modularisation only for *Independent Subtrees*
  - ☺ New Component Fault Tree Concept allows Input and Output Ports

provided by:

Bernhard Kaiser, formerly Fraunhofer IESE

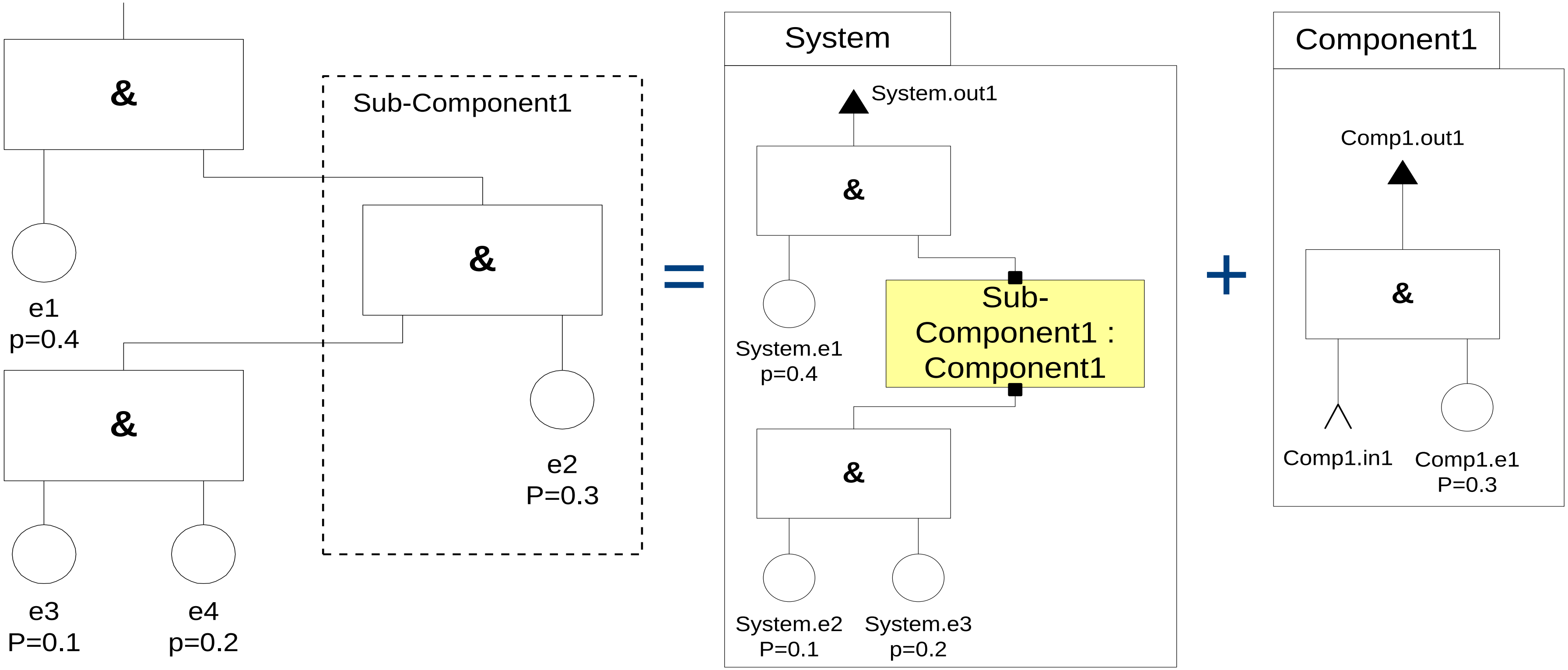
Problems Addressed

- Software-Controlled Systems require Adequate Models
  - ! Capture Software Behaviour (States, Sequences of Actions)
  - ! Model Multi-State Components
  - ! Integrate with Software Design Models
  - ☹ Traditional FTA is a Combinatorial Model (only Boolean Logic)
  - ☺ State-Event-Fault-Trees are a State-Based Model distinguishing states and events and allowing temporal propositions in an intuitive notation

provided by:

Bernhard Kaiser, formerly Fraunhofer IESE

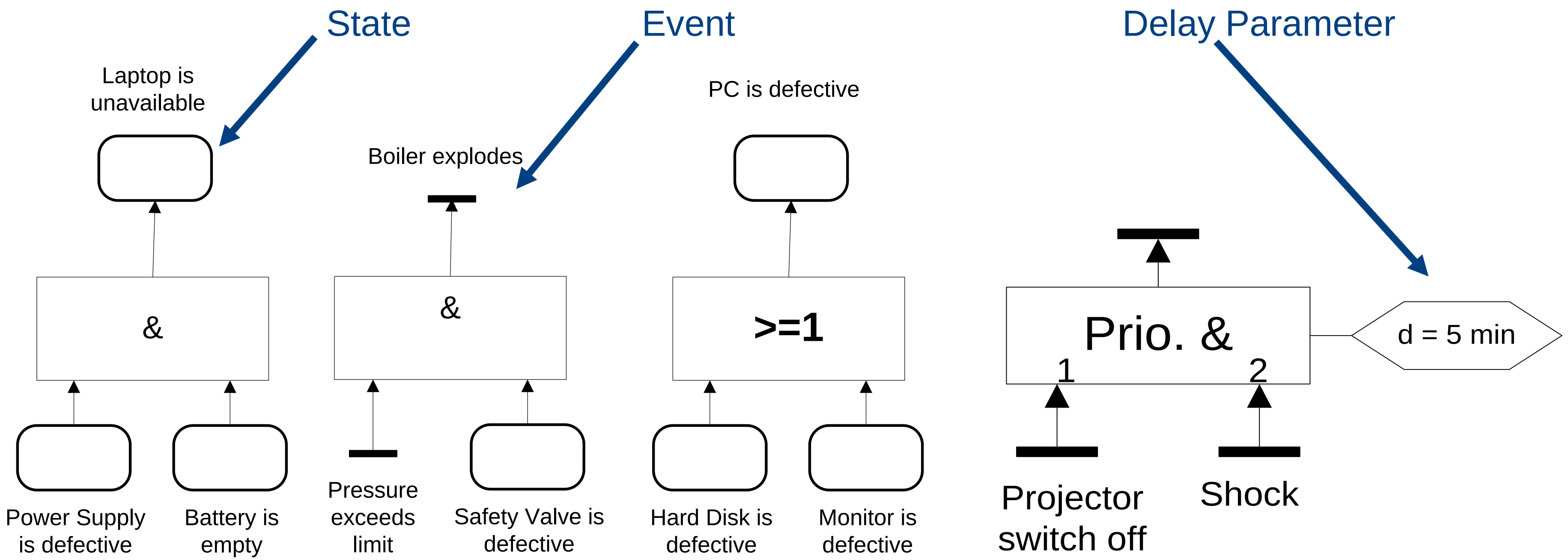
Component Fault Trees



Components are actual technical units, joint by ports.
Components represent Boolean formulas, not probabilities!

provided by:
Bernhard Kaiser, formerly Fraunhofer IESE

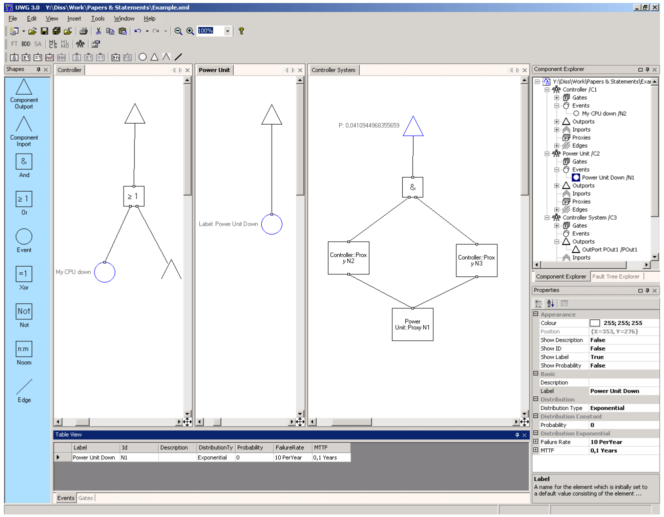
State-Event-Fault-Trees



Informally: Combination of Fault Trees and Statecharts
- State/Event Consistency is checked
- Unambiguous Semantics
- Analysis by Translation into Petri Nets

provided by:
Bernhard Kaiser, formerly Fraunhofer IESE

The UWG3 / ESSaRel Tool Project



Label	Id	Description	DistributionTy	Probability	FailureRate	MTTF
Power Unit Down	N1		Exponential	0	10 PerYear	0,1 Years

Appearance	
Colour	255; 255; 255
Position	{X=353, Y=276}
Show Description	False
Show ID	False
Show Label	True
Show Probability	False
Basic	
Description	
Label	Power Unit Down
Distribution	
Distribution Type	Exponential
Distribution Constant	
Probability	0
Distribution Exponential	
Failure Rate	10 PerYear
MTTF	0,1 Years

Label
A name for the element which is initially set to a default value consisting of the element ...

- Windows based GUI Tools under .NET
  - Intuitive Use (Drag&Drop, Project Explorer...)
  - XML File-Format for Collaboration and Reuse
- UWG3
  - Supports Component Fault Trees
  - Collaboration with Siemens
  - First Version in 2003
  - Successful Evaluation in Industry Projects
- ESSaRel
  - Embedded Systems Safety and Reliability Analyser
  - Available Spring 2005
  - SEFTs, Markov Chains, State Diagrams
  - Analysis by Translation to Petri Nets (DSPNs)
  - Interfaces to Rational Rose RT and TimeNET
- Download at **www.essarel.de**

provided by:
Bernhard Kaiser, formerly Fraunhofer IESE

Prof. Yiannis Papadopoulos
Department of Computer Science
University of Hull
http://www2.dcs.hull.ac.uk/people/cssyp/
y.i.papadopoulos@dcs.hull.ac.uk

Semi-automatic synthesis of Fault Trees

- Inputs to the synthesis algorithm

  A. Topological model of the system that identifies components and material, energy and data transactions among those components

  B. Logical expressions that, for each component, determine how deviations of component outputs are caused by internal malfunctions or deviations of component inputs

  In this type of failure logic, input and output deviations are described qualitatively representing conditions such as the omission or commission of parameters and deviations from correct value (i.e. hi-low) or expected timing behaviour (i.e. early-late)

- Fault Tree Synthesis Algorithm: Combines a backward traversal of the model and evaluation of failure expressions encountered in the course of the traversal

provided by:
Yiannis Papadopoulos – Department of Computer Science - University of Hull

Semi-automatic synthesis of Fault Trees

- Output of the synthesis algorithm

  A network of interconnected fault trees which show how component failures combine and propagate through the model to cause hazardous failures at system outputs

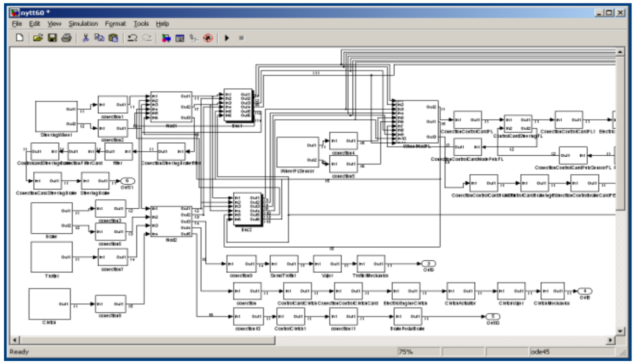
  Fault trees currently incorporate only classical AND & OR gates. However, the aim is to extend this logic with NOT and temporal gates (i.e. “Priority AND” or “AND THEN” gates)

- Synthesis tool operates on Matlab Simulink and Simulation X models. It has its own fault tree analysis capabilities, but also interfaces with FT+ a commercial fault tree analysis tool (by Isograph Ltd). The tool is experimental but usable by third parties
- Case studies have been reported on complex prototypes in conjunction with DaimlerChrysler, Volvo Cars and Germanisher Lloyd

provided by:
Yiannis Papadopoulos – Department of Computer Science - University of Hull

Semi-automatic synthesis of Fault Trees

TECHNISCHE UNIVERSITÄT KAISERSLAUTERN



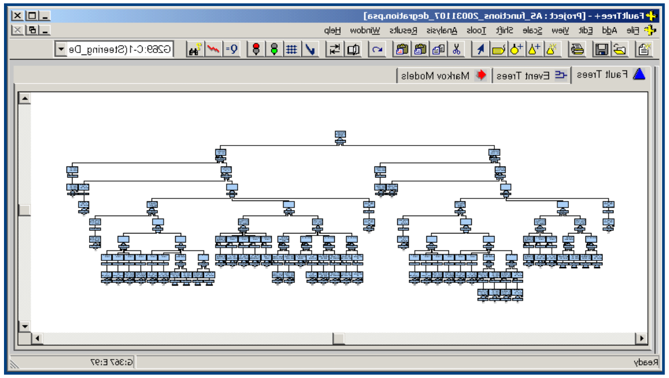
Fault Tree Plus

Simulink model and fault tree of a steer-by-wire prototype by Volvo

OTHER RELATED WORK BY THE SAME GROUP

1. Automatic Analysis: Synthesis of FMEAs
2. Automated Monitoring: Diagnosis & correction of failures using a combination of state-charts and fault trees as an executable monitoring model
3. Optimisation of system designs (combining genetic algorithms and fault tree synthesis)

- **Optimal allocation of redundancies**
- **Optimal allocation of reliability requirements on components of evolving architectures**



Web-browser

Further info @ www2.dcs.hull.ac.uk/people/cssyp

provided by:
Yiannis Papadopoulos – Department of Computer Science - University of Hull

Can We Trust Fault Trees / Event Trees Analyses?

Antoine Rauzy

Institut de Mathématiques de Luminy
163, avenue de Luminy, Case 907
13288 Marseille CEDEX 9
FRANCE

Work tel+fax +33 4 91 26 96 34
Home tel+fax +33 4 91 73 26 15

http://iml.univ-mrs.fr/~arauzy/
arauzy@iml.univ-mrs.fr

Current situation in the (nuclear, avionic, …) industry

- Models (fault trees/event trees) are
  - Huge (several thousands gates and basic events)
  - Highly tool-dependant

  therefore …
  - Difficult to master (size)
  - Impossible to redesign (economical reasons)

- Assessment tools are
  - Very efficient and user friendly

  but …
  - Based on the same technology (minimal cutsets) that relies on approximations (rare events, cut-offs, *ad-hoc* treatment of success branches)

provided by:
Antoine Rauzy, Institut de Mathématiques de Luminy, arauzy@iml.univ-mrs.fr

Problems and Perspectives

- Known problems
  - Most part of models is useless (in general, more that 90% of the basic events never show up in cutsets)
  - Accuracy of results is unpredictable (approximations)
  - Cross verifications of results with different tools is hard (tool-dependency) and useless (same underlying technology)
- Hot research topics
  - Design of a sound mathematical framework (non coherent models, importance factors, …)
  - Improvement the BDD technology to make it able to deal with (all, most of the) large models of the industry
  - Automated model refactoring

provided by:
Antoine Rauzy, Institut de Mathématiques de Luminy, arauzy@iml.univ-mrs.fr

Anders P. Ravn
Department of Computer Science
Aalborg University
Fr. Bajersvej 7E
9220 Aalborg East
Denmark

Office: B2-210Email : apr@cs.aau.dk
PHONE : +45 96 35 88 87
(Direct)FAX : +45 98 15 98 89

Fault Trees in Safety Analysis

- A fault tree is a formula in a logic used for analysing safety
- The formula is constructed by backwards reachability analysis from a primary fault - a top event
- The formula is a "counterexample" to the system being safe
- The safety requirement is formally the negation of the formula

provided by:
Anders P.Ravn, Aalborg University, apr@cs.aau.dk, www.cs.auc.dk/~apr/

Fault Tree Semantics

- Safety analysis is concerned with dynamical systems (state changing with time)
  - Special case: programs (transition systems) in general hybrid systems
- The model for the logic must thus include both state and time,
  and the formula must be able to specify both temporal and state dependent properties
- Intermediate nodes must be names of formulas, elementary nodes must denote properties of the dynamical system, combined using the logical connectives

provided by:
Anders P.Ravn, Aalborg University, apr@cs.aau.dk, www.cs.auc.dk/~apr/

Fault Tree Semantics

- The safety requirement corresponding to a top event E is **NOT <> E** (not somewhere E) or equivalently **[] NOT E** (invariantly NOT E). E is itself a composite formula
  - Note in particular, that in modal logics <>(A AND B) is much stronger than <>A AND <>B
- That is the rationale for looking for "cut sets" where one assumes simultaneous occurrence
- If one wants complications, one can use (semi)Markov processes as model

provided by:
Anders P.Ravn, Aalborg University, apr@cs.aau.dk, www.cs.auc.dk/~apr/

Fault Trees as Software Requirements

- The dynamical system model of the safety analysis is represented by an observed state in embedded software
- The algorithms in embedded software manipulate this representation of the state and produces inputs and outputs
- The invariants should be maintained by the algorithms
- Could e.g. be model checked

provided by:
Anders P.Ravn, Aalborg University, apr@cs.aau.dk, www.cs.auc.dk/~apr/

Wolfgang Reif, Frank Ortmeier, Gerhard Schellhorn
University of Augsburg
http://www.informatik.uni-augsburg.de/lehrstuehle/swt/se/

Formal FTA

- Formal semantics in interval temporal logic (ITL)
  - Each gate is represented by an ITL-formula

- Features
  - Events may have durations
  - Distinction between synchronous causes and asynchronous causes
  - Causes must happen before consequences (better than Hansen semantics)

- Formally proven: minimal cut set theorem

provided by:
Wolfgang Reif, University of Augsburg, reif@Informatik.Uni-Augsburg.DE

Minimal cut set theorem

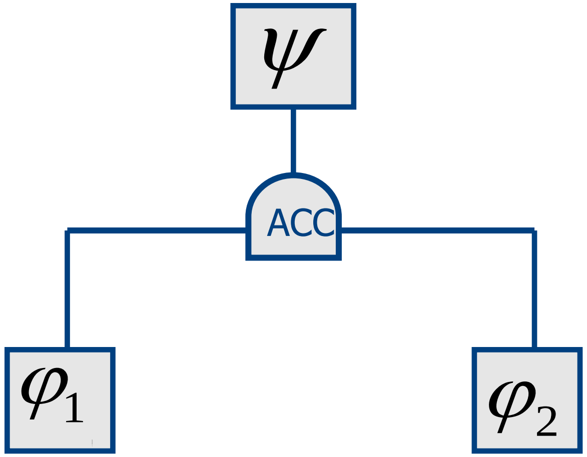
- Minimal cut sets of ITL semantics preserve intuitive understanding
- Theorem: if all gates have been verified, than prevention of one element of every minimal cut set prevents the hazard (i.e. no branches have been forgotten in the fault tree)

provided by:

Wolfgang Reif, University of Augsburg, reif@Informatik.Uni-Augsburg.DE

Example

- Asynchronous cause-consequence gate
  - Informal: Both φ_1 and φ_2 must happen before Ψ, but not necessarily simultaneously
- Verification condition
  - Informal: „There exists no trace in the system, such that Ψ occurs without previous appearance of φ_1 and φ_2"



ITL-operators

$$\neg\left(\neg\Diamond\varphi_1 \wedge \neg\Diamond\varphi_2 ; \psi\right)$$

provided by:
Wolfgang Reif, University of Augsburg, reif@Informatik.Uni-Augsburg.DE

Summary: FTA Semantics

- 7 different types of gates for describing precise fault trees, including
  - AND-, OR-gates with Boolean semantics
  - AND-, OR-gates with cause-consequence relationship
  - Distinction Synchronous/Asynchronous
  - INHIBIT-gates

provided by:

Wolfgang Reif, University of Augsburg, reif@Informatik.Uni-Augsburg.DE

Tool Support

- For infinite state models
  - interactive theorem prover KIV (supports fault trees, state charts, ITL)
- For finite state models
  - Automatic model checking with SMV in CTL (only possible for events without duration)
- Integrated in formal safety analysis approach
  - ForMoSA approach (includes formal verification, analysis of failure modes, formal FTA, formal FMEA and quantitative risk optimizations)

provided by:

Wolfgang Reif, University of Augsburg, reif@Informatik.Uni-Augsburg.DE

Kishor S. Trivedi
ECE Dept
Duke University
Durham NC USA
Phone: (919)401-0299 ext 306
e-mail: kst@ee.duke.edu
URL: www.ee.duke.edu/~kst

FTA Contributions

- Better understanding of FTA
  - Compare modeling powers of various dependability models [mal94]
  - Survey of SDP method in FTA [rai95]
- Extension of FTA
  - Use FTA for phased-mission systems [ma99, zang99]
  - Use FTA for multistate systems [vee94, zang03]
  - Reliability analysis using FT for repairable systems [bal95]
- New algorithms for FTA
  - SDP based [luo98]
  - BDD based [zang99, zst99, zang03]
- Combining FTA with other modeling techniques
  - Hierarchical modeling [bbook, rbook]
  - Relation between FT and Petri nets [mal95]

provided by:
Kishor Trivedi, Duke University, Durham NC, kst@ee.duke.edu, www.ee.duke.edu/~kst

FTA in SHARPE

- SHARPE software package [sha]
- Symbolic Hierarchical Automated Reliability and Performance Evaluator
- Supported model types using FTA technique
  - Reliability Block Diagrams
  - Fault trees with repeated events
  - Reliability Graphs
  - Phased-mission Systems
  - Multi-state Fault Trees
- FTA techniques in SHARPE
  - Factoring, SDP, BDD
- Measures obtained using FTA in SHARPE
  - Reliability/availability, MTTF, importance measures, min-cuts, min-paths, product form CDFs

provided by:
Kishor Trivedi, Duke University, Durham NC, kst@ee.duke.edu, www.ee.duke.edu/~kst

References

- [mal94] Power-Hierarchy of Dependability Model Types, Manish Malhotra and K. Trivedi, IEEE Transactions on Reliability, Vol. 43, No. 2, pp. 493-502, Sept. 1994.
- [rai95] A Survey on Efficient Computation of Reliability Using Disjoint Products Approach, Suresh Rai, Malathi Veeraraghavan and K. Trivedi, Networks, Vol. 25, No. 3 (1995), pp. 147-163
- [ma99] An Algorithm for Reliability Analysis of Phased-Mission Systems, Yue Ma and K. Trivedi. Reliability Engineering and System Safety, Vol 66, No. 2, pp. 157-170, 1999
- [vee94] A Combinatorial Algorithm for Performance and Reliability Analysis using Multistate Models, M. Veeraraghavan and K. Trivedi, IEEE Transactions on Computers, Vol. 43, No. 2, pp. 229-234, February 1994
- [luo98] An Improved Algorithm for Coherent System Reliability, Tong Luo and K. Trivedi, IEEE Trans. on Reliability, Vol. 47, No. 1, pp. 73-78, March 1998
- [zang99] A BDD-based Algorithm for Reliability Analysis of Phased-Mission Systems, Xinyu Zang, Hairong Sun and K. Trivedi, IEEE Transactions on Reliability, Vol. 48, No. 1, pp. 50-60, March 1999
- [zst99] A BDD Approach to Dependability Analysis of Distributed Computer Systems with Imperfect Coverage, X. Zang, H. Sun and K. Trivedi, in Dependable Network Computing,
D. R. Avresky (ed.), pp. 167-190, Kluwer Academic Publishers, The Netherlands, 1999.

provided by:

Kishor Trivedi, Duke University, Durham NC, kst@ee.duke.edu, www.ee.duke.edu/~kst

References

- [bal95] Componentwise Decomposition for an Efficient Reliability Computation of Systems with Repairable Components, M. Balakrishnan and K. Trivedi, Proc. Twenty-fifth International Symposium on Fault-Tolerant Computing, Pasadena, CA, pp. 259-268, July 1995
- [zang03] A BDD-based algorithm for analysis of multistate systems with multistate components, Zang, X.; Wang, D.; Sun, H.and Trivedi, K.S.; Computers, IEEE Transactions on ,Volume: 52 , Issue: 12, Dec. 2003, Pages:1608 – 1618
- [sha] SHARPE: http://www.ee.duke.edu/~kst/software_packages.html
- [bbook] Probability and Statistics with Reliability, Queuing, and Computer Science Applications, K. Trivedi, John Wiley and Sons, New York, 2001
- [rbook] Performance and Reliability Analysis of Computer Systems: An Example-Based Approach Using the SHARPE Software Package, Robin A. Sahner, Kishor S. Trivedi and Antonio Puliafito, Kluwer Academic Publishers, 1996.
- [mal95] Dependability Modeling Using Petri-Nets, M. Malhotra and K.Trivedi, IEEE Transactions on Reliability, Vol. 44, No. 3, pp. 428-440, Sept., 1995

provided by:

Kishor Trivedi, Duke University, Durham NC, kst@ee.duke.edu, www.ee.duke.edu/~kst