The state of the art in fault tree modeling and analysis

Enno Ruijters

November 11, 2014

Outline

Outline

About me

- Enno Ruijters
- PhD Student at University of Twente
- ArRangeer project
    - ProRail / STW
    - Improving railroad maintenance using Dynamic Fault Trees and Stochastic Model Checking

Group members

Enno Ruijters

Dennis Guck

prof. dr. ir. Joost-Pieter Katoen

dr. Mariëlle Stoelinga

Introduction to fault trees

- Developed in 1961 by Nuclear Regulatory Agency
- Question: How reliable is your system?
- Now used by:

Introduction to fault trees

- Developed in 1961 by Nuclear Regulatory Agency
- Question: How reliable is your system?
- Now used by:

Why fault trees?

- Some things really should not fail
- Risk assessment is sometimes mandatory
  - Probability of catastrophic failures?
  - Biggest risk factors?
  - Possible mitigations?

Why fault trees?

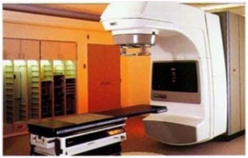
- Some things really should not fail

  Reliability Probability of failing within given time

Why fault trees?

- Some things really should not fail

  Reliability Probability of failing within given time



Why fault trees?

- Some thing should not fail for long

  Availability Proportion of time in functioning state

Why fault trees?

- Some thing should not fail for long

  Availability Proportion of time in functioning state

amazon.com

What do we want to know?

Qualitative:

- Insight into biggest risks
- Relatively fast to perform
- Easy to understand
- Limited information

Quantitative:

- Quantify total risk
- Quantify effect of mitigation
- Time consuming
- Hard to estimate numbers

What do we want to know?

Qualitative:

- **Cut sets**: Sets of components causing failure
  Example: Airplane fails when both engines fail

What do we want to know?

Qualitative:

- **Cut sets**: Sets of components causing failure
  Example: Airplane fails when both engines fail
- **Common cause failures**: Multiple failures with one cause
  Example: Redundant computers running same program

What to we want to know?

Quantitative:

- **Reliability** $\equiv$ Probability of failure within time t
  Example: Probability of containment failure within 25 year nuclear plant lifetime

What to we want to know?

Quantitative:

- **Reliability** $\equiv$ Probability of failure within time t
  Example: Probability of containment failure within 25 year nuclear plant lifetime
- **Availability** $\equiv$ Proportion of time (in $[0, \infty\rangle$ or $[0, t]$) spent not failed
  Example: Amazon EC2 cloud offers SLA of 99.95% uptime

What to we want to know?

Quantitative:

- **Reliability** $\equiv$ Probability of failure within time t
  Example: Probability of containment failure within 25 year nuclear plant lifetime
- **Availability** $\equiv$ Proportion of time (in $[0, \infty\rangle$ or $[0, t]$) spent not failed
  Example: Amazon EC2 cloud offers SLA of 99.95% uptime
- **MTBF** $\equiv$ Expected time between two successive failures (in finite or infinite horizon)
  Example: How frequently will my car break down?

What to we want to know?

Quantitative:

MTTF $\equiv$ Expected time between system becoming functioning and failing
Example: How long will my car run after a service?

What to we want to know?

Quantitative:

MTTF $\equiv$ Expected time between system becoming functioning and failing
Example: How long will my car run after a service?

MTTFF $\equiv$ Expected time before first failure
Example: How long will my new car without failing?

What to we want to know?

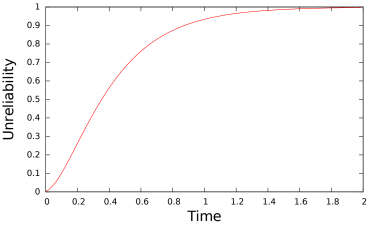
Quantitative:

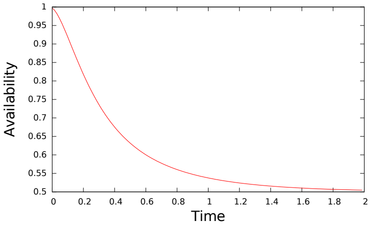
MTTF $\equiv$ Expected time between system becoming functioning and failing
Example: How long will my car run after a service?

MTTFF $\equiv$ Expected time before first failure
Example: How long will my new car without failing?

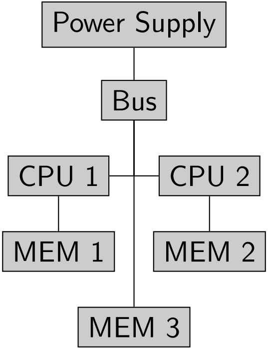
ENF $\equiv$ Expected number of failures
Example: How many switches will fail in the country per year?

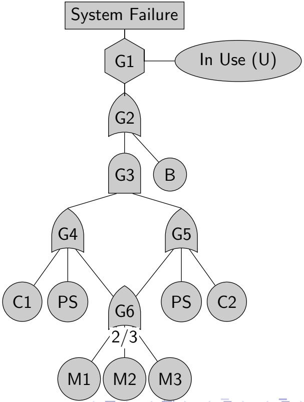
Time-dependent metrics





Fault tree example





- Redundant CPUs
- 1 shared spare memory unit

Fault tree elements

- Basic events (leaves)
- Intermediate Events (gates)
- Top (Level) Event (gate)
- DAG, but often shown as tree with duplicated events

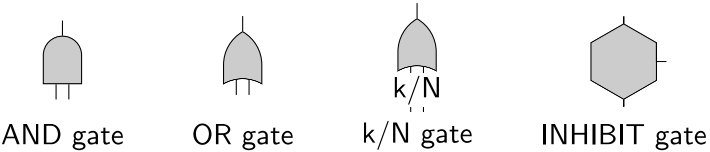
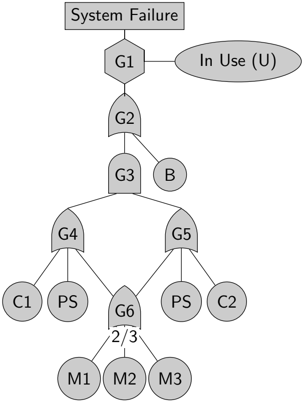


Figure: Images of the gates types in a static fault tree

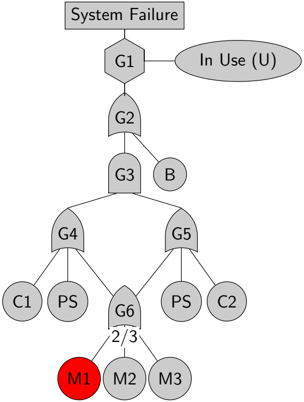
Example of fault tree failure propagation

- No failures



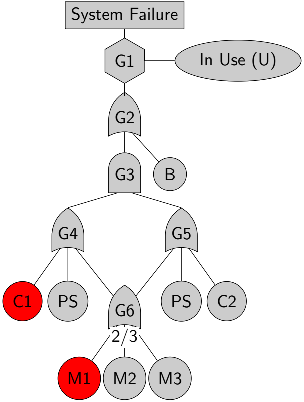
Example of fault tree failure propagation

- Failure of M1



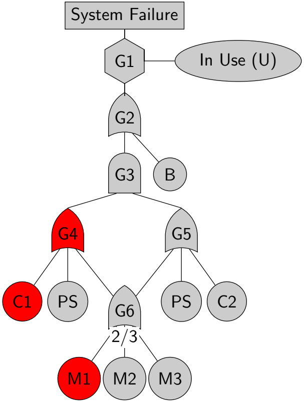
Example of fault tree failure propagation

- Failure of M1
- Failure of C1



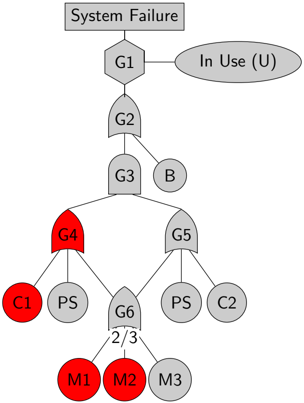
Example of fault tree failure propagation

- Failure of M1
- Failure of C1



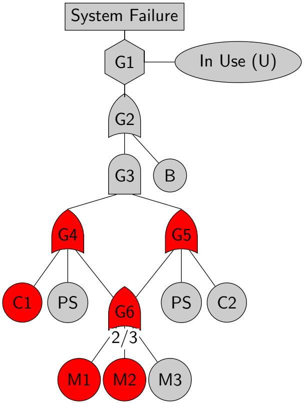
Example of fault tree failure propagation

- Failure of M1
- Failure of C1
- Failure of M2



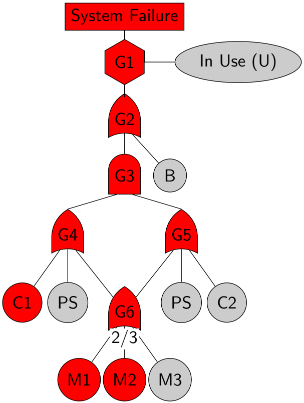
Example of fault tree failure propagation

- Failure of M1
- Failure of C1
- Failure of M2



Example of fault tree failure propagation

- Failure of M1
- Failure of C1
- Failure of M2



Outline

Measures of interest

Qualitative:

- Cut sets
- Path sets
- Common Cause Failures

Quantitative:

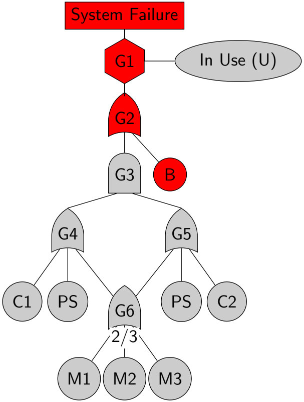
- Reliability
- Availability
- MTBF/MTTF/MTTFF
- Expected number of failures
- importance values

Qual. analysis: Cut sets

- Set of components causing failure
- Usually minimal cut sets
- Small cut sets like candidates for system improvement

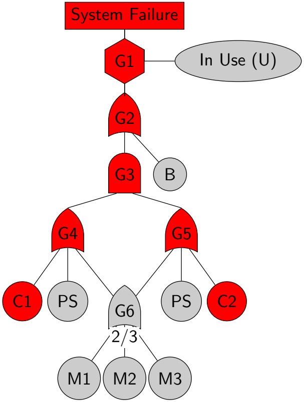
Qual. analysis: Cut sets

- Set of components causing failure
- Usually minimal cut sets
- Small cut sets like candidates for system improvement
- Examples: {U,B}



Qual. analysis: Cut sets

- Set of components causing failure
- Usually minimal cut sets
- Small cut sets like candidates for system improvement
- Examples: {U,B}, {U,C1,C2}, ...

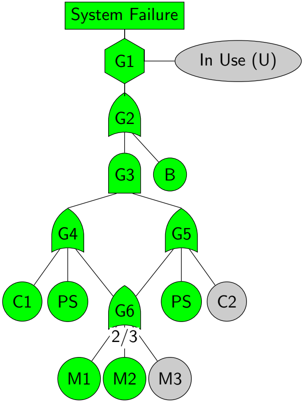


Qual. analysis: Path sets

- Set of components NOT causing failue
- Usually minimal path sets
- No small path sets can indicate low redundancy

Qual. analysis: Path sets

- Set of components NOT causing failue
- Usually minimal path sets
- No small path sets can indicate low redundancy
- Example: {B,PS,C1,M1,M2}



Outline

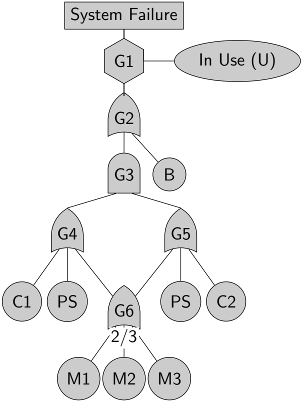
- Cut sets: Boolean manipulation
- Cut sets: Binary Decision Diagrams
- Common cause failures

Cut set analysis: Boolean manipulation

- Use boolean algebra to construct DNF
- Bottom-up: Start with leaves
- Top-down: Start with root

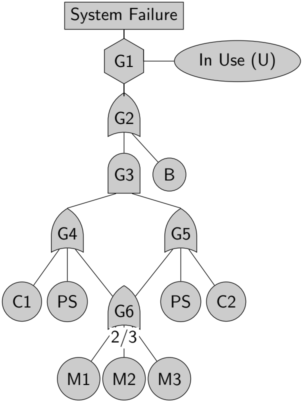
Cut set analysis: Boolean manipulation

- Use boolean algebra to construct DNF
- Bottom-up: Start with leaves
- Top-down: Start with root
- Example (top-down):
- $G1 = U \wedge G2$ and $G2 = B \vee G3$



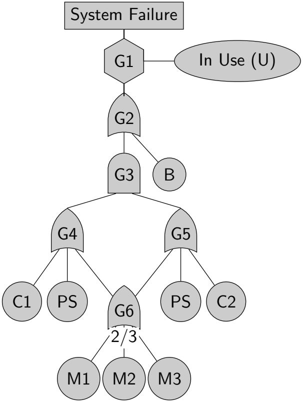
Cut set analysis: Boolean manipulation

- Use boolean algebra to construct DNF
- Bottom-up: Start with leaves
- Top-down: Start with root
- Example (top-down):
- $G1 = U \wedge G2$ and $G2 = B \vee G3$
- $G1 = U \wedge (B \vee G3)$



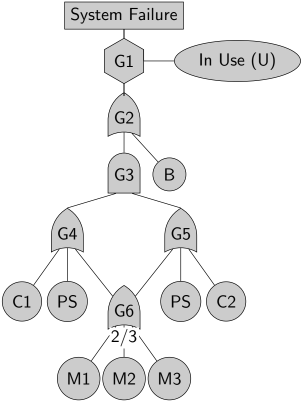
Cut set analysis: Boolean manipulation

- Use boolean algebra to construct DNF
- Bottom-up: Start with leaves
- Top-down: Start with root
- Example (top-down):
- $G1 = U \wedge G2$ and $G2 = B \vee G3$
- $G1 = U \wedge (B \vee G3)$
- $G1 = (U \wedge B) \vee (U \wedge G3)$



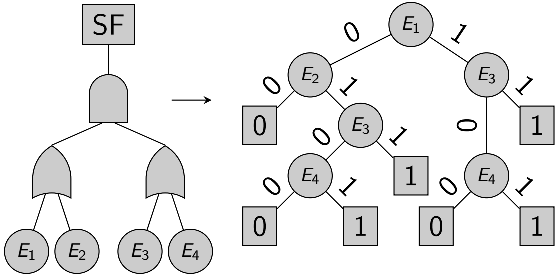
Cut set analysis: Boolean manipulation

- Use boolean algebra to construct DNF
- Bottom-up: Start with leaves
- Top-down: Start with root
- Example (top-down):
- $G1 = U \wedge G2$ and $G2 = B \vee G3$
- $G1 = U \wedge (B \vee G3)$
- $G1 = (U \wedge B) \vee (U \wedge G3)$
- $G1 = (U \wedge B) \vee (U \wedge (G4 \wedge G5))$



Cut set analysis: Binary Decision Diagrams

- DAG representing boolean function
- Leaves are 0 or 1
- All paths from the root have the same variable ordering



Common cause failures

- Simultaneous failures of multiple components
- Examples: fire, earthquake, wear of identical components
- Cannot be derived from FT structure
- Expert insight to determine CCF within cut sets

Outline

- Fault tree types
- Bottom-up method
- Rare-event approximation
- Bayesian networks
- Monte Carlo Simulation

Fault tree types

Time:

- Discrete-time (one-shot)
- Continuous-time without repairs
- Continuous-time with independent repairs

Failure distributions:

- Single probability (discrete-time only)
- Exponential distribution
- Arbitrary distribution

Quant. analysis: Bottom-up method

- When no events are shared:
- $\mathbb{P}[X_{AND}(X_1, X_2, \cdots, X_n) = 1]$
- $= \mathbb{P}[X_1 = 1 \wedge X_2 = 1 \wedge \cdots \wedge X_n = 1]$
- $= \mathbb{P}[X_1 = 1]\mathbb{P}[X_2 = 1] \cdots \mathbb{P}[X_n = 1]$
- Likewise for other gates
- Same for availability

Quant. analysis: Rare event approximation

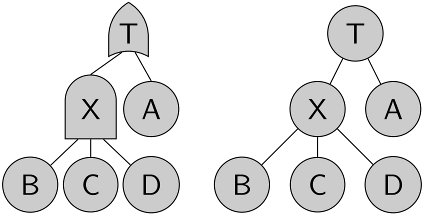
- Assuming failures are infrequent (e.g. 10^{-9})
  - Approximate using $\mathbb{P}(A \vee B) \approx \mathbb{P}(A) + \mathbb{P}(B)$
  - Sum unavailabilities or unreliabilities of cut sets
- Can be made exact using inclusion-exclusion principle:
  - $\mathbb{P}(A \vee B) = \mathbb{P}(A) + \mathbb{P}(B) - \mathbb{P}(A \wedge B)$

Quant. analysis: Bayesian Networks

- General technique used in many probabilistic analyses
- Express fault tree in conditional probabilities

Quant. analysis: Bayesian Networks

- General technique used in many probabilistic analyses
- Express fault tree in conditional probabilities
- Example (A or (B and C and D)):



$$\mathbb{P}(T = 1 | A = 1 \vee X = 1) = 1$$
$$\mathbb{P}(A = 1) = 0.1$$
$$\mathbb{P}(X = 1 | B = C = D = 1) = 1$$
$$\mathbb{P}(B = 1) = 0.3$$
$$\mathbb{P}(C = 1) = 0.4$$
$$\mathbb{P}(D = 1) = 0.1$$

Quant. analysis: Bayesian Networks

Advantages:

- Inference using existing tools
- Allows diagnosis
- FT structure persists into model
- Easy to extend with e.g. probabilistic gates

Disadvantages:

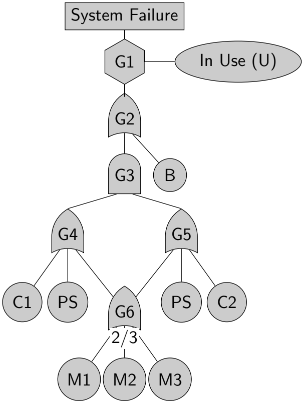
- Conditional probability table exponentially large in nr. of inputs

Quant. analysis: Monte Carlo simulation

- Simulation used in many applications
- Sample failures or failure times, and repair times if needed
- Propagate failures through the tree at every failure or repair
- Track measure of interest through repeated simulations

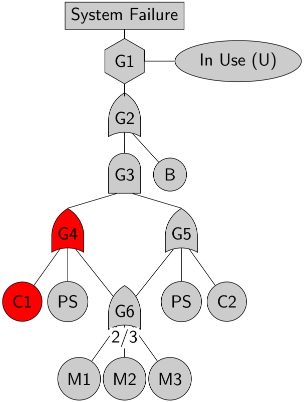
Monte Carlo Simulation example

- All BEs have failure probability 0.2
- Runs: 0
- Failures: 0
- Estimated reliability:



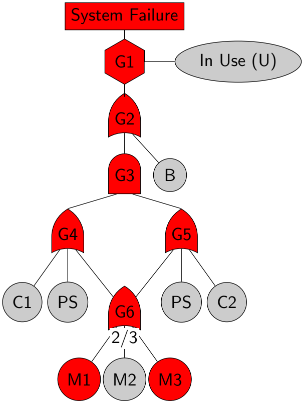
Monte Carlo Simulation example

- All BEs have failure probability 0.2
- Runs: 1
- Failures: 0
- Estimated reliability: 1



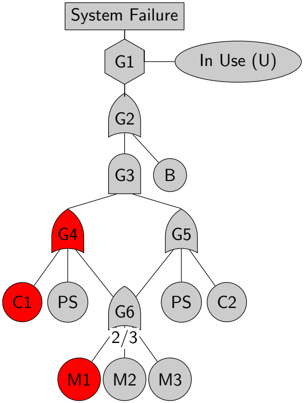
Monte Carlo Simulation example

- All BEs have failure probability 0.2
- Runs: 2
- Failures: 1
- Estimated reliability: 0.5



Monte Carlo Simulation example

- All BEs have failure probability 0.2
- Runs: 3
- Failures: 1
- Estimated reliability: 0.666



Summary

Quantitative analysis techniques:

- Bottom-up method
- Rare-event approximation
- Bayesian networks
- Monte Carlo Simulation

Other techniques:

- Algebraic analysis
- Algebraic approximation

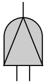
Outline

Shortcomings of fault trees

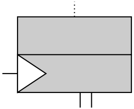
- No information about failure sequences
- Poor modeling of shared spare components
- Dependencies cause large trees
- One solution: Dynamic fault trees (DFTs)

Dynamic fault trees

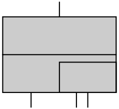
Three new gates:



PAND gate

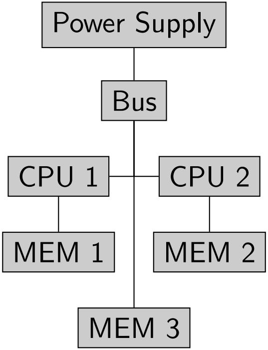


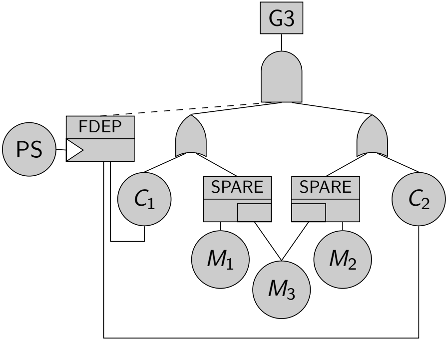
FDEP gate



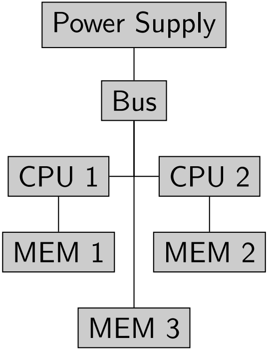
SPARE gate

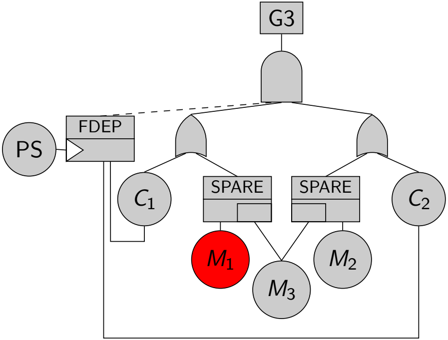
DFT Example


Power Supply
Bus
CPU 1
CPU 2
MEM 1
MEM 2
MEM 3

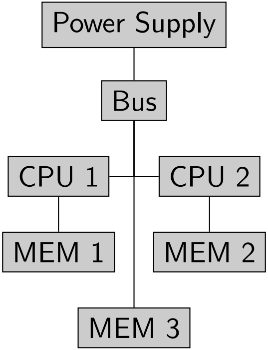

G3
FDEP
PS
SPARE
SPARE
C_1
C_2
M_1
M_2
M_3

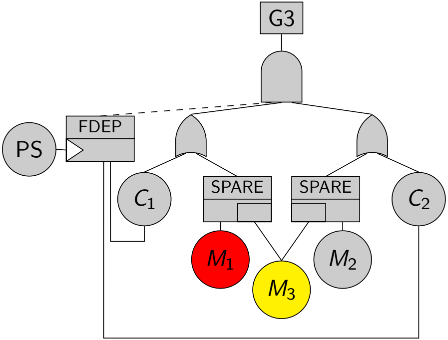
DFT Example


Power Supply
Bus
CPU 1
CPU 2
MEM 1
MEM 2
MEM 3

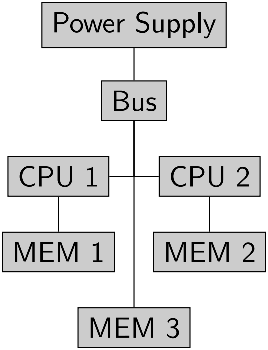

G3
FDEP
PS
SPARE
SPARE
C_1
C_2
M_1
M_2
M_3

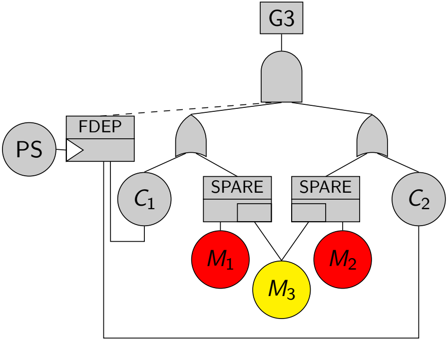
DFT Example


Power Supply
Bus
CPU 1
CPU 2
MEM 1
MEM 2
MEM 3

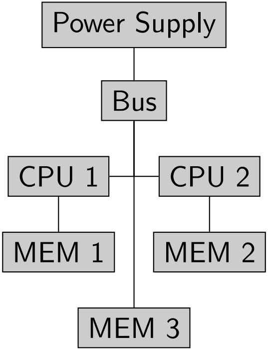

G3
FDEP
PS
C_1
SPARE
SPARE
C_2
M_1
M_2
M_3

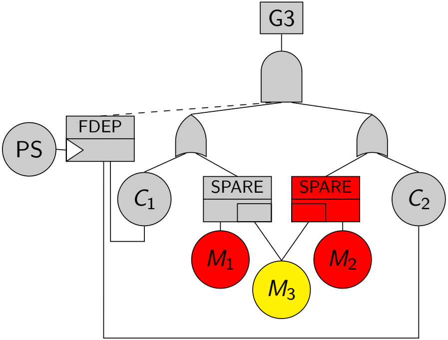
DFT Example


Power Supply
Bus
CPU 1
CPU 2
MEM 1
MEM 2
MEM 3

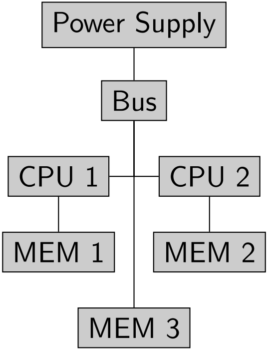

G3
FDEP
PS
SPARE
SPARE
C_1
C_2
M_1
M_2
M_3

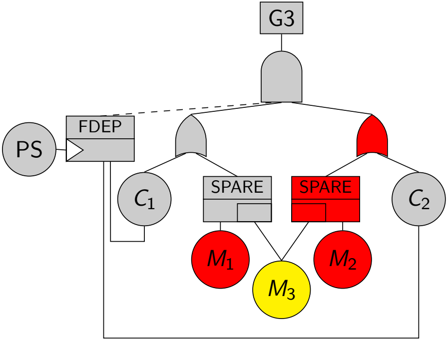
DFT Example


Power Supply
Bus
CPU 1
CPU 2
MEM 1
MEM 2
MEM 3

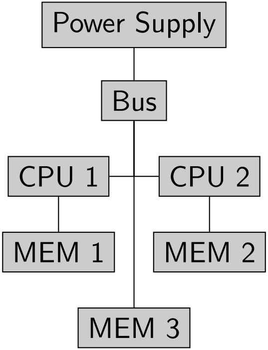

G3
FDEP
PS
C_1
SPARE
SPARE
C_2
M_1
M_3
M_2

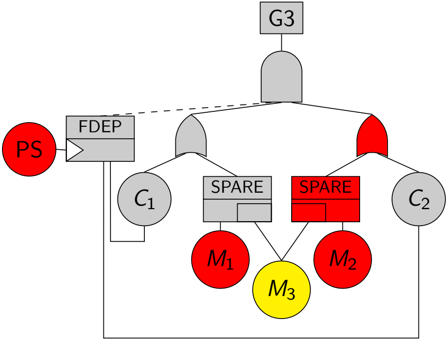
DFT Example


Power Supply
Bus
CPU 1
CPU 2
MEM 1
MEM 2
MEM 3

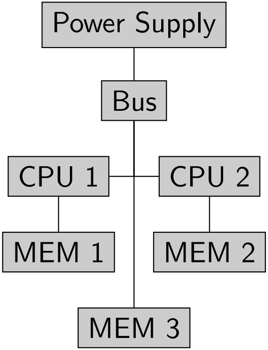

G3
FDEP
PS
C_1
SPARE
SPARE
C_2
M_1
M_3
M_2

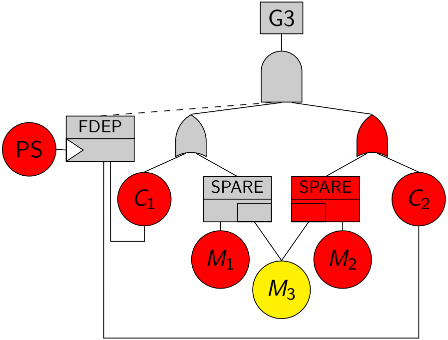
DFT Example


Power Supply
Bus
CPU 1
CPU 2
MEM 1
MEM 2
MEM 3

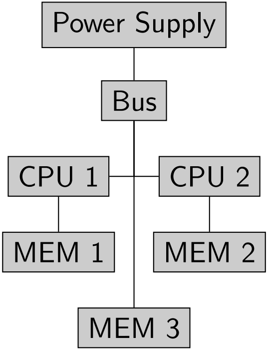

G3
FDEP
PS
C_1
SPARE
SPARE
C_2
M_1
M_3
M_2

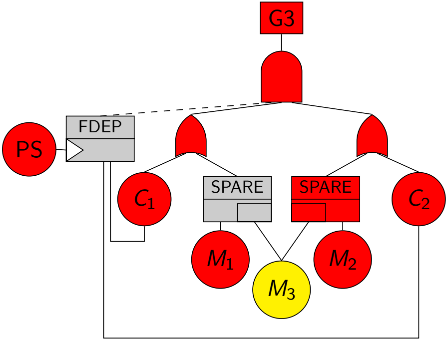
DFT Example


Power Supply
Bus
CPU 1
CPU 2
MEM 1
MEM 2
MEM 3


G3
FDEP
PS
SPARE
SPARE
C_1
C_2
M_1
M_2
M_3

DFT Example


Power Supply
Bus
CPU 1
CPU 2
MEM 1
MEM 2
MEM 3


G3
FDEP
PS
SPARE
SPARE
C_1
C_2
M_1
M_2
M_3

Outline

Measures of interest

Qualitative:

- Cut/path sets
- Cut sequences

Quantitative:

- Reliability
- Availability
- MTBF/MTTF/MTTFF
- Expected number of failures

Outline

- Cut sets
- Cut sequences

Qualitative analysis

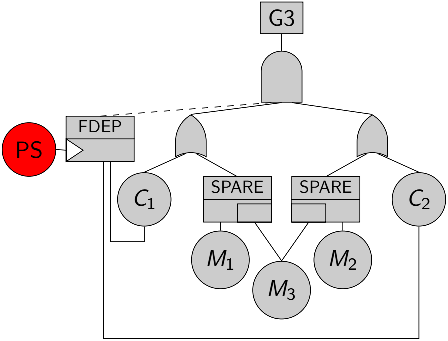
Cut sets for DFTs:

- Failures of cut sets CAN cause system failures, depending on ordering
  - Due to shared spares, failure not always caused by cut sets
- Convert DFT into SFT, by replacing:
  - PAND $\rightarrow$ AND
  - SPARE $\rightarrow$ AND
  - FDEP $\rightarrow$ OR

DFT cut sets Example

Example cut sets:

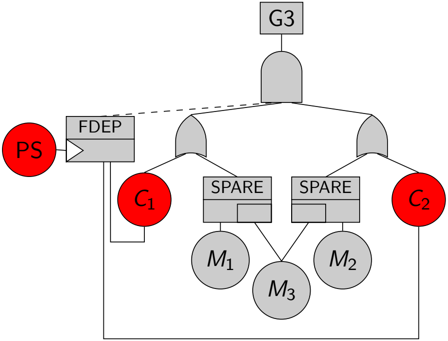
- {PS}



DFT cut sets Example

Example cut sets:

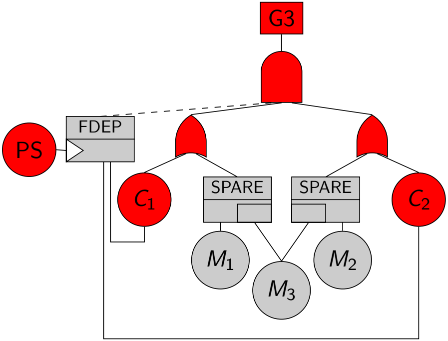
- {PS}



DFT cut sets Example

Example cut sets:

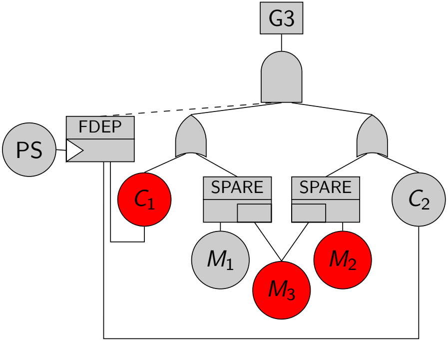
- {PS}



DFT cut sets Example

Example cut sets:

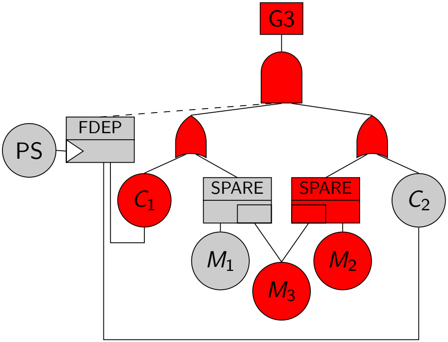
- {PS}
- {C1,M1,M2}



DFT cut sets Example

Example cut sets:

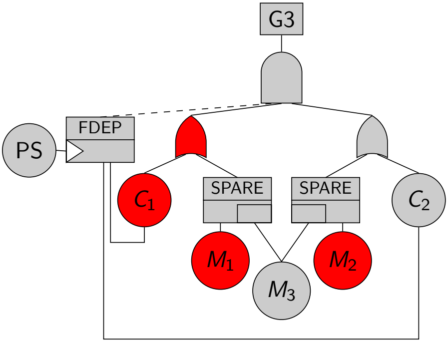
- {PS}
- {C1,M2,M3}



DFT cut sets Example

Example cut sets:

- {PS}
- {C1,M2,M3}
- NOT {C1,M1,M2}



Qualitative analysis

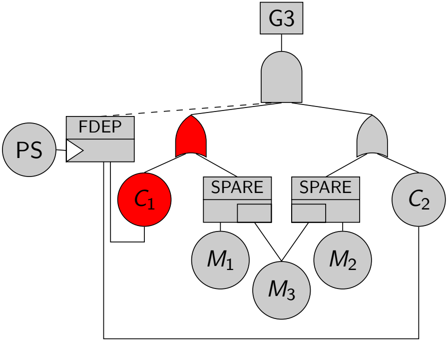
Cut sequences:

- Like cut sets, but include sequence information
- Failure of a cut sequence always causes system failure
- Any system failure is caused by a cut set failure

DFT cut sequences Example

Example cut sequence:

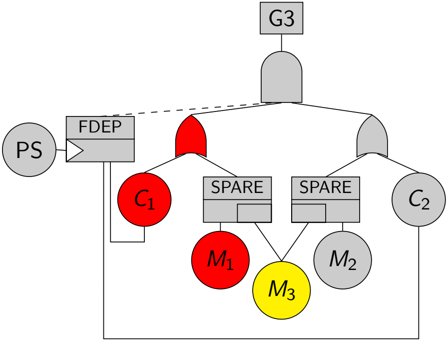
- $\langle C1, M1, M2 \rangle$



DFT cut sequences Example

Example cut sequence:

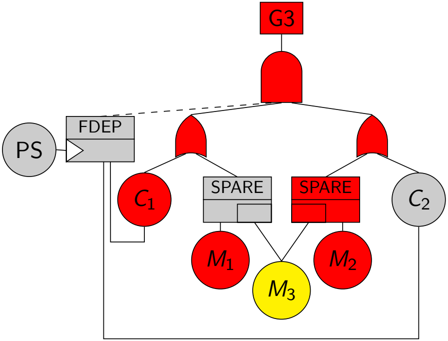
- $\langle \mathrm{C1}, \mathrm{M1}, \mathrm{M2} \rangle$



DFT cut sequences Example

Example cut sequence:

- $\langle C1, M1, M2 \rangle$

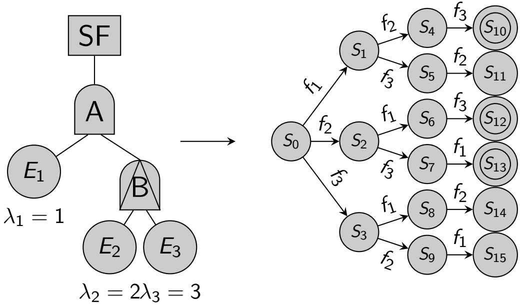


Outline

- Markov analysis
- I/O IMC

Quant. analysis: Markov chain

Analysis by markov chain:



Quant. analysis: Markov chain

Advantages:

- Exact semantics
- No nondeterminacy
- Reuse of existing modelcheckers (PRISM, etc.)

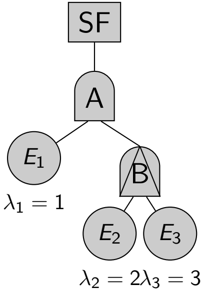
Disadvantages:

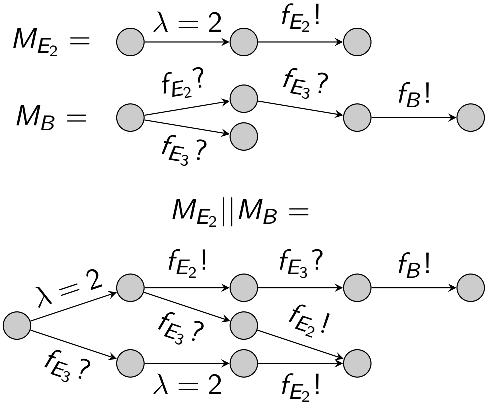
- Semantics are ca. 20 pages long
- Combinatorial explosion

Quant. analysis: Compositional Markov Analysis

- *Input/Output Interactive Markov Chains* exist of gates and basic events
- Input/Output signals allow parallel composition
- Models of FT elements are composed into one large model

Quant. analysis: I/O IMC example





Quant. analysis: Compositional Markov Analysis

Advantages:

- Semantics easier to understand
- Intermediate minimization reduces state-space explosion
- Easy to add new gates or events
- Can model nondeterminacy

Quant. analysis: Compositional Markov Analysis

Advantages:

- Semantics easier to understand
- Intermediate minimization reduces state-space explosion
- Easy to add new gates or events
- Can model nondeterminacy

Disadvantages:

- Still has state-space explosion
- Nondeterminacy

Quant. analysis: Compositional Markov Analysis

Advantages:

- Semantics easier to understand
- Intermediate minimization reduces state-space explosion
- Easy to add new gates or events
- Can model nondeterminacy

Disadvantages:

- Still has state-space explosion
- Nondeterminacy

Note: This is the approach used in DFTCalc and the ArRangeer project

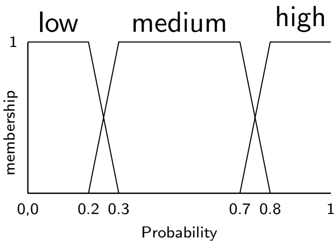
Other quantitative analysis methods

- Petri Nets
- Dynamic Bayesian Networks
- Modularization of static and dynamic subtrees
- Monte Carlo Simulation

Outline

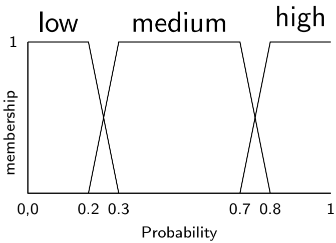
Fuzzy numbers

- Uncertainty and variation in BE probabilities
- Expert judgement not exact
- Possible solution: BE probabilities in fuzzy sets
- Several frameworks for computations on fuzzy numbers
- Can compute same measures as for non-fuzzy FTs.



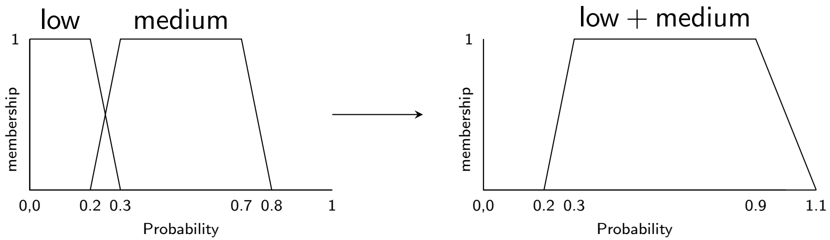
Fuzzy computations

- Combine fuzzy sets using mathematical operations
- Problem: probability distribution unknown
- Various assumptions exist, partly for computational efficiency
- Example: medium + medium = (not low, maybe medium, likely high)



Fuzzy addition

- $\mu_{A+B}(z) = \max_{z=x+y}(\min\{\mu_A(x), \mu_B(y)\})$
- Example: $\mu_{\text{low+medium}}(1) = 0.5$



Fuzzy arithmetic

- Problem: Fuzzy arithmetic does not return original values
- Various methods to may fuzzy sets back onto descriptors
- In practice: expert judgement

Other uncertain FTs

- 'Intuitionistic fuzzy set theory': Membership function uncertain
- Probability distribution for BE failure rates
- Multi-state BE with uncertain states
- Normal distribution approximation

Outline

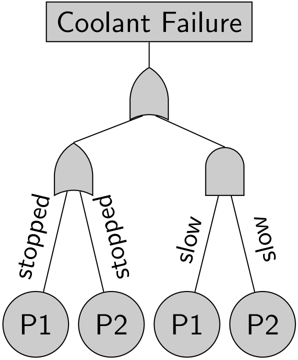
FTs with dependent events

- Normal FTs assume independent BEs
- Not always realistic (‘valve stuck open’ and ‘valve stuck closed’ are not independent)
- Component failures and degradation may propagate

Extended fault trees

- Components have multiple states (between ‘failed’ and ‘perfectly working’)
- Component state can affect other component failure rates
- Gates allow for different combinations of states
- Textual DSL needed for specification
- Only quantitative continuous-time analysis defined

Extended fault trees




```
DEFINE FAILDEP pump1:
         CAUSE = P1.slow;
         EFFECT = RATECHANGES P2:*2;
END
DEFINE FAILDEP pump2:
         CAUSE = P2.slow;
         EFFECT = RATECHANGES P1:*2;
END
```

Boolean Driven Markov Processes

- BEs and gates represented as multiple Markov Processes (MPs)
- States in the MPs can trigger other elements to switch MPs
- Applications: Changing failure rates, multistate components, new gates
- Disadvantage: Harder to quickly oversee
- Only quantitative continuous-time analysis defined

Other dependent event extensions

- Multiple FTs for different failure modes
- Specifying mutually exclusive events
- Replace BEs by Petri nets

Outline

Repairable fault trees

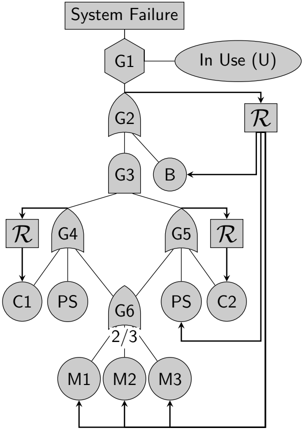
- Simple repair model: Simultaneous independent repairs
- Problem: Limited resources for repairs in real life
- Problem: Hidden failures
- Solution method: Repairable Fault Trees

Repairable fault trees

- Add *Repair Boxes* to tree (now becomes cyclic)
- Repair box has one input: repair starts when input fails
- Repair box specifies multiple components to repair
- Repair policy determines how repairs procees (simultaneous, sequential, combination, etc.)
- Qualitative analysis (cut sets) possible but less useful
- Several quantitative analysis techniques defined

Example RFT

- Repair shared components when system fails
- Repair CPUs when cluster fails



Outline

Fault trees with temporal properties

- Static FTs do not consider timing information
- DFTs are one approach to include them, others exist

FTs with temporal gates

Add new gate types:

- AND-THEN gate: Requires one event 'immediately after' another
  - Formal description with informal predicate
  - Only qualitative analysis defined (extended cut sequence)

FTs with temporal gates

Add new gate types:

- AND-THEN gate: Requires one event ‘immediately after’ another
  - Formal description with informal predicate
  - Only qualitative analysis defined (extended cut sequence)
- POR: fail when first input fails before others
- SAND: fail on simultaneous failure of all inputs
  - PAND + POR + SAND strictly more expressive than AND-THEN gate
  - Only qualitative analysis defined
  - Quantitative analysis seems easy to add

FTs with temporal logic

Several approaches add temporal logics to FTs:

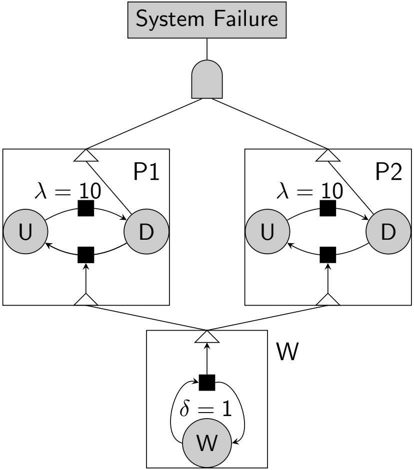
- Cause-consequence gates
  - Allows indeterminate delays
  - Qualitative analysis for failure-preventing cut sets
  - No other analysis possible
- Duration calculus
  - Calculus allows reasoning about delays
  - Not proven decidable
  - No automated analysis available
- Propositional Linear Temporal Logic
  - Adds single-input gates like PREV and SOMETIME-PAST
  - Qualitative analysis defined
  - Quantitative analysis probably also possible

Outline

State-Event Fault Trees

- Practical system failures are sometimes state-dependent
- Especially true of computer software
- SEFT combine state machines with FT gates
- State transitions cause events
- Events and states are combined in gates
- Events can cause state transitions
- Later additions include delays, probabilistic gates
- Quantitative analysis by Petri Nets

State-Event Fault Trees



Outline

Future work

- Inclusion of preventive maintenance
- More complex failure models
- Synthesis of maintenance and repair policies

Conclusion