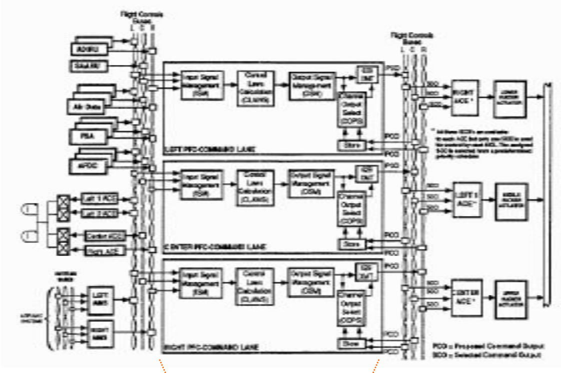
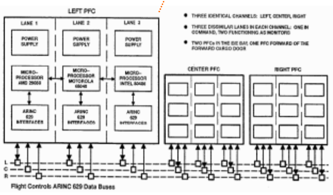
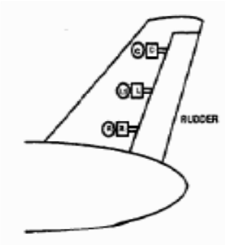
# Embedded Systems 22

# Boeing 777

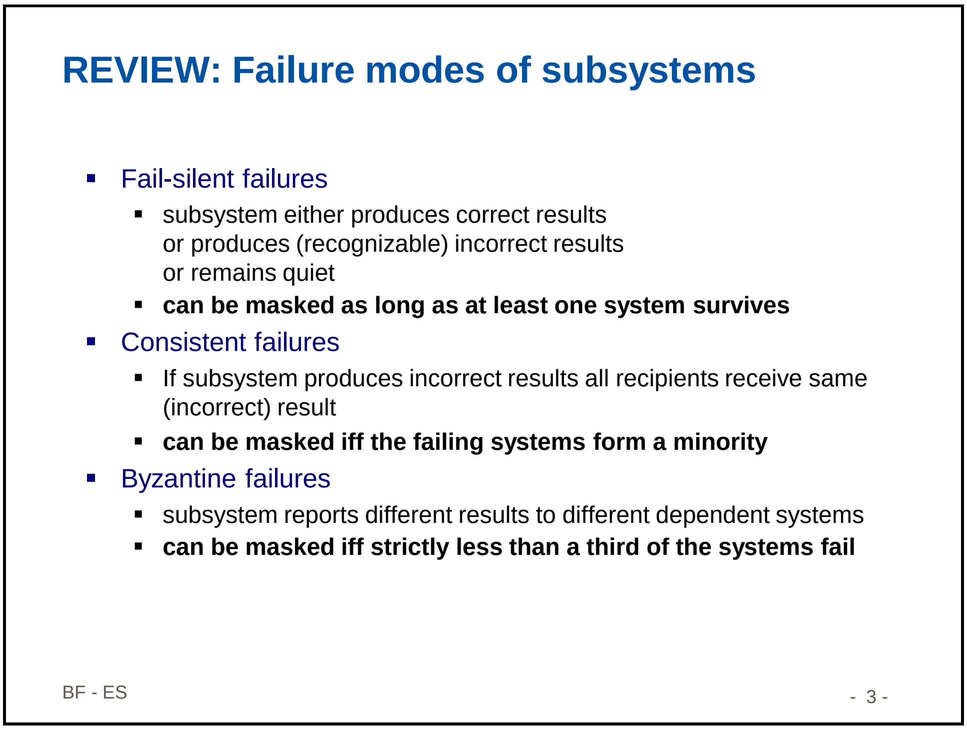



LEFT PFC
LANE 1
LANE 2
LANE 3
POWER SUPPLY
MICRO-PROCESSOR AMD 29050
MICRO-PROCESSOR MOTOROLA 68040
MICRO-PROCESSOR INTEL 80486
ARINC 629 INTERFACES
CENTER PFC
RIGHT PFC
L
C
R
Flight Controls ARINC 629 Data Buses

- THREE IDENTICAL CHANNELS: LEFT, CENTER, RIGHT
- THREE DISSIMILAR LANES IN EACH CHANNEL: ONE IN COMMAND, TWO FUNCTIONING AS MONITORS
- TWO PFCs IN THE EE BAY, ONE PFC FORWARD OF THE FORWARD CARGO DOOR

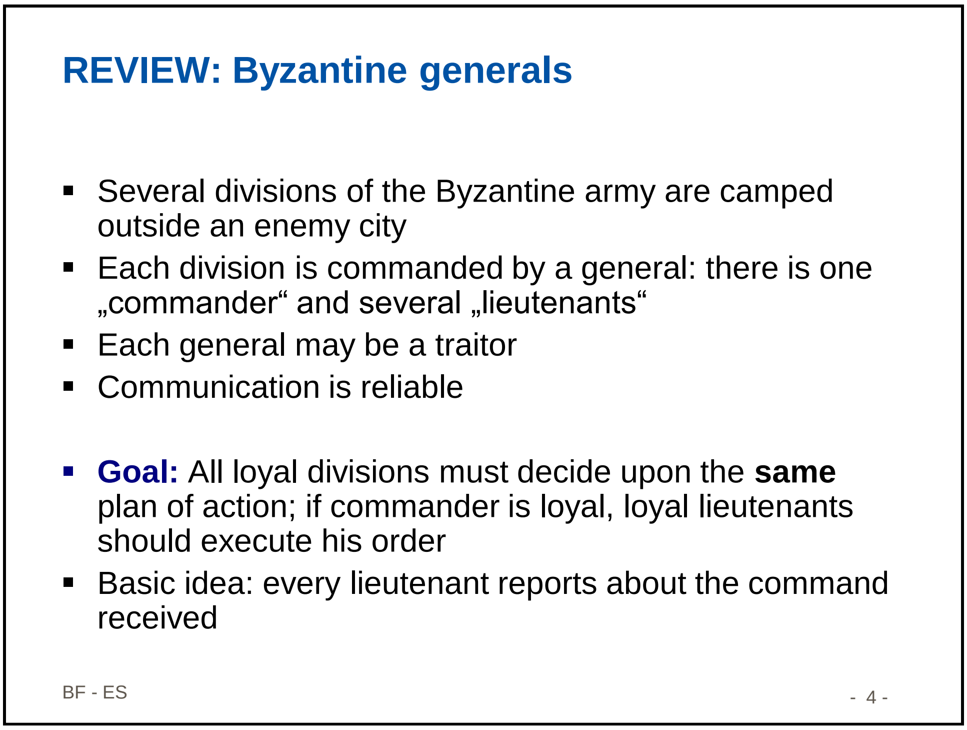


http://www.davi.ws/avionics/TheAvionicsHandbook_Cap_11.pdf



## REVIEW: Failure modes of subsystems

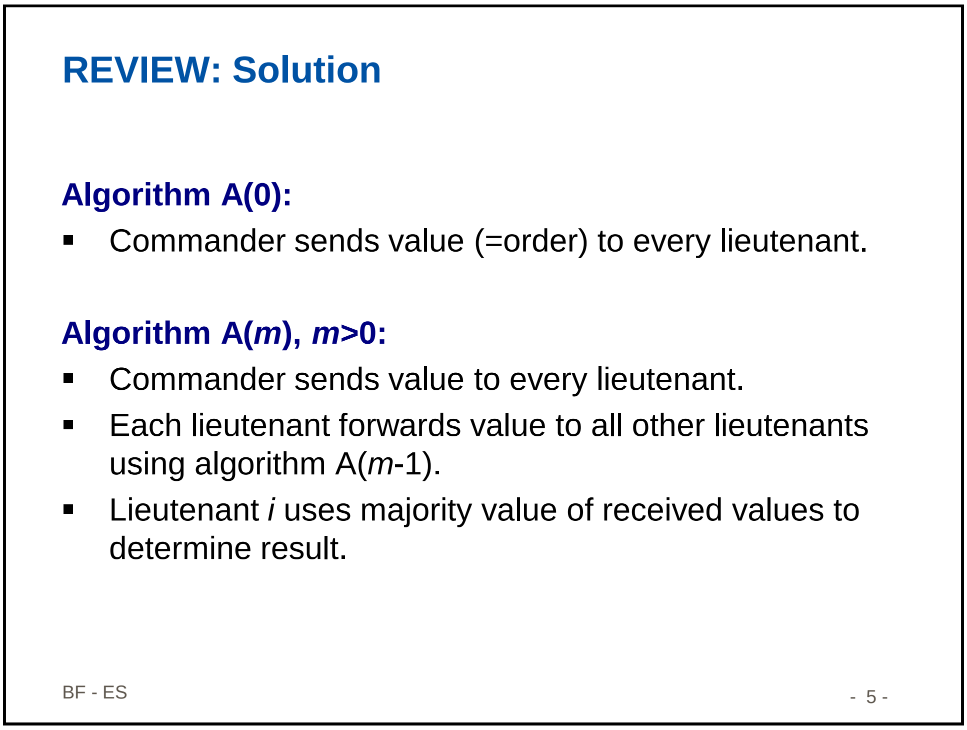
- Fail-silent failures
  - subsystem either produces correct results
    or produces (recognizable) incorrect results
    or remains quiet
  - **can be masked as long as at least one system survives**
- Consistent failures
  - If subsystem produces incorrect results all recipients receive same (incorrect) result
  - **can be masked iff the failing systems form a minority**
- Byzantine failures
  - subsystem reports different results to different dependent systems
  - **can be masked iff strictly less than a third of the systems fail**



## REVIEW: Byzantine generals

- Several divisions of the Byzantine army are camped outside an enemy city
- Each division is commanded by a general: there is one „commander“ and several „lieutenants“
- Each general may be a traitor
- Communication is reliable

- **Goal:** All loyal divisions must decide upon the **same** plan of action; if commander is loyal, loyal lieutenants should execute his order
- Basic idea: every lieutenant reports about the command received



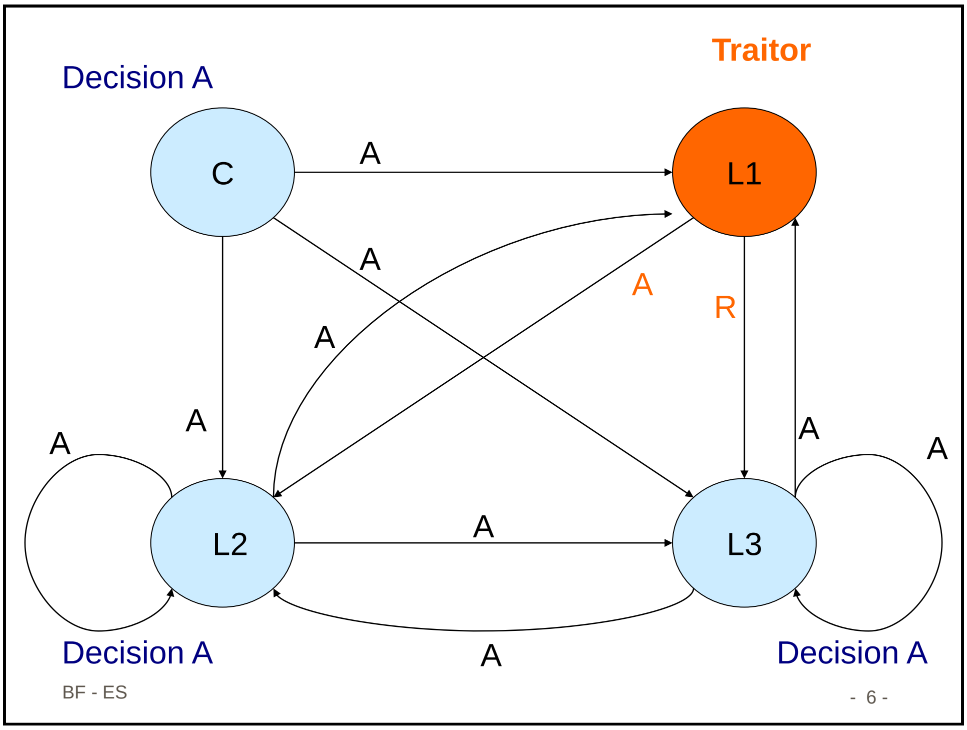
## REVIEW: Solution

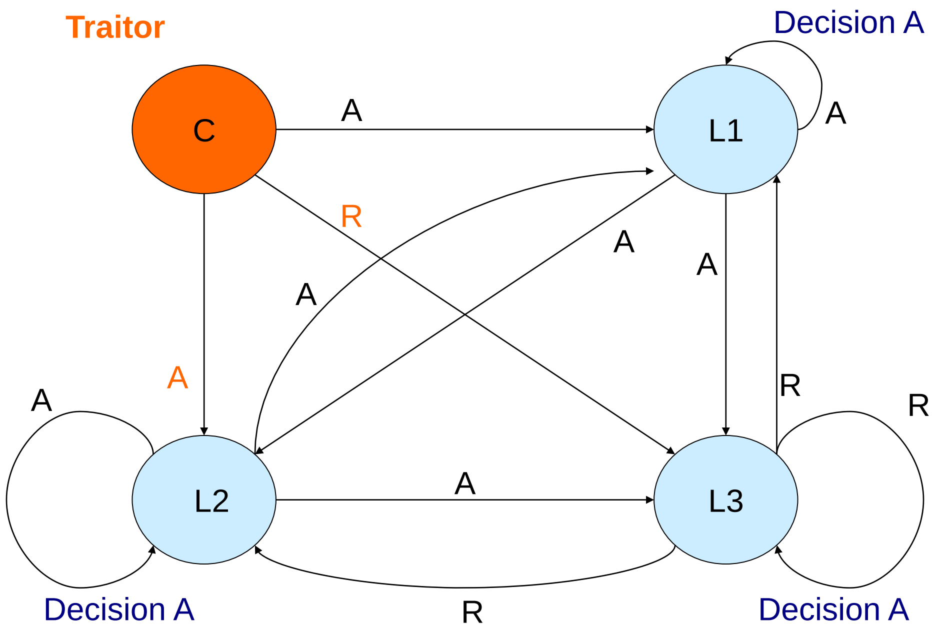
**Algorithm A(0):**

- Commander sends value (=order) to every lieutenant.

**Algorithm A(*m*), *m*>0:**

- Commander sends value to every lieutenant.
- Each lieutenant forwards value to all other lieutenants using algorithm A(*m*-1).
- Lieutenant *i* uses majority value of received values to determine result.





Traitor

Decision A

C

L1

L2

L3

A

R

A

A

A

A

A

A

A

R

R

A

R

Decision A

Decision A

## REVIEW: Lieutenants reach consensus (Case 1 traitor)

Case 1: Commander is loyal, some lieutenant is traitor

⇒ set of forwarded messages differ by at most 1 value

⇒ same majority vote

Case 2: Commander is traitor, lieutenants are loyal

⇒ sets of forwarded messages are identical

⇒ same majority vote at every lieutenant

## Lemma:

- Let there be more than 2*k*+*m* generals and at most *k* traitors. If the commander is loyal, then algorithm A(*m*) guarantees that all loyal lieutenants agree on the commander‘s order.

Induction on m:

- $A(0)$ : Commander is loyal
  $\Rightarrow$ correct result received

- $m-1 \to m$ :
  - Loyal commander sends v to all lieutenants
  - Each loyal lieutenant forwards v using $A(m-1)$ with $> 2k+m-1$ generals
    $\Rightarrow$ By IH, every loyal lieutenant receives v

Since there are at most k traitors, majority is loyal $\Rightarrow$ majority vote is v.

## Theorem

- Let there be more than 3*m* generals and at most *m* traitors. Then algorithm A(*m*) guarantees that the loyal lieutenants reach a consensus. If the commander is loyal, then the consensus is the commander‘s order.

Induction on m

$m=0 \Rightarrow A(0)$ ✓

$m-1 \to m$ :

Case: Commander is loyal $\Rightarrow$ Apply lemma with $k=m$.

Case: Commander is traitor
$\Rightarrow$ At most $m-1$ lieutenants are traitors
There are $> 3m-1$ lieutenants
$3m-1 > 3(m-1)$

By IH, $A(m-1)$ ensures consensus among the loyal lieutenants on forwarded msgs.

$\Rightarrow$ the set of received forwarded msgs is the same for every loyal lieutenant

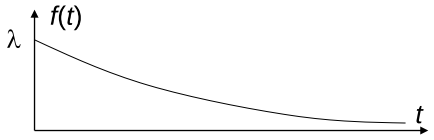
$\Rightarrow$ the majority vote is the same at every loyal lieutenant.

## Reliability: f(t), F(t)

- Let *T*: time until first failure, *T* is a random variable
- Let *f*(*t*) be the density function of *T*

Example: Exponential distribution

$f(t)=\lambda e^{-\lambda t}$

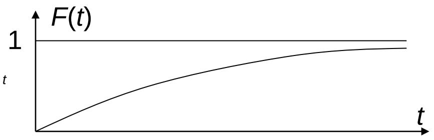


- *F*(*t*) = probability of the system being faulty at time *t*:

$F(t) = \Pr(T \leq t)$ $$F(t) = \int_0^t f(x)\,dx$$

Example: Exponential distribution

$$F(t) = \int_0^t \lambda e^{-\lambda x}\,dx = -[e^{-\lambda x}]_0^t = 1 - e^{-\lambda t}$$



## Reliability: *R*(*t*)

- **Reliability** *R*(*t*) = probability that the time until the first failure is larger than some time *t*:

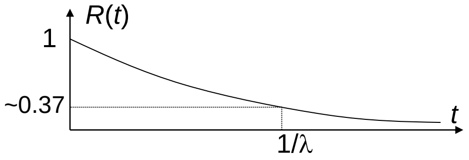
$R(t)=\Pr(T>t),\ t\geq 0$ $\qquad R(t)=\int_t^\infty f(x)dx$

$$F(t)+R(t)=\int_0^t f(x)dx+\int_t^\infty f(x)dx=1$$

$$R(t)=1-F(t)$$

<u>Example</u>: Exponential distribution

$R(t)=e^{-\lambda t};$



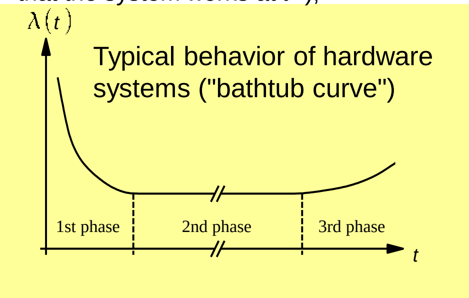
## Failure rate

The failure rate at time *t* is the probability of the system failing between time *t* and time *t+Δt:*

$$\lambda(t)=\lim_{\Delta t\to 0}\frac{\Pr(t<T\leq t+\Delta t\mid T>t)}{\Delta t}=\lim_{\Delta t\to 0}\frac{F(t+\Delta t)-F(t)}{\Delta t R(t)}=\frac{f(t)}{R(t)}$$

Conditional probability ("provided that the system works at *t* ");

P(A|B)=P(AB)/P(B)



Typical behavior of hardware systems ("bathtub curve")

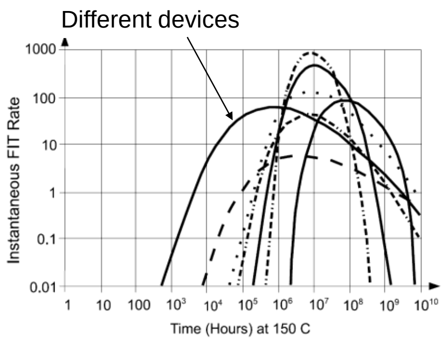
For exponential distribution:

$$\frac{f(t)}{R(t)}=\frac{\lambda e^{-\lambda t}}{e^{-\lambda t}}=\lambda$$

FIT = expected number of failures in $10^9$ hrs.

## Actual failure rates

- Example: failure rates less than 100 FIT for the first 20 years (175,300 hrs) of life at 150°C @ TriQuint (GaAs)
- [www.triquint.com/company/quality/faqs/faq_11.cfm]



Target: Failures rates of systems ≤ 1FIT

Reality: Failures rates of circuits ≤ 100 FIT

☞redundancy is required to make a system more reliable than its components

∃ non-constant failure rates!

## MTTF = *E*{*T* }, the *statistical mean* value of *T*

$$\text{MTTF} = E\{T\} = \int_0^\infty t \cdot f(t)\, dt$$

According to the definition of the statistical mean value

Example: Exponential distribution

$$\text{MTTF}_{\text{exp}} = \int_0^\infty t \cdot \lambda e^{-\lambda t} dt = -\cancel{\left[t \cdot e^{-\lambda t}\right]_0^\infty} + \int_0^\infty e^{-\lambda t} dt$$

$$\int u \cdot v' = u \cdot v - \int u' \cdot v$$

$$\text{MTTF}_{\text{exp}} = -\frac{1}{\lambda}\left[e^{-\lambda t}\right]_0^\infty = -\frac{1}{\lambda}[0-1] = \frac{1}{\lambda}$$

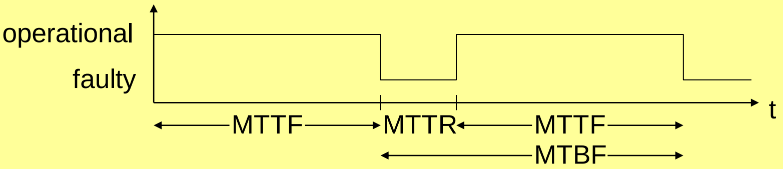
MTTF is the reciprocal value of failure rate.

## MTTF, MTTR and MTBF

MTTR = mean time to repair
(average over repair times using distribution *M*(*d*))
MTBF* = mean time between failures = MTTF + MTTR

$$\text{Availability}\ A = \lim_{t \to \infty} A(t) = \frac{\text{MTTF}}{\text{MTBF}}$$

- Ignoring the statistical nature of faults …



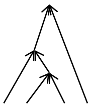
* Mixed up with MTTF, if starting in operational state is implicitly assumed

## Failure mode and effect analysis (FMEA)

- FMEA starts at the components and tries to estimate their reliability. The first step is to create a table containing components, possible faults, probability of faults and consequences on the system behavior.

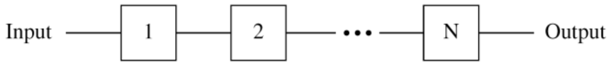
| *Component* | *Failure* | *Consequences* | *Probability* | *Critical?* |
|---|---|---|---|---|
| …<br>Processor<br>… | …<br>metal migration<br>… | …<br>no service<br>… | …<br>$10^{-7}$ /h<br>… | …<br>yes<br>… |

- Using this information, the reliability of the system is computed from the reliability of its parts (corresponding to a bottom-up analysis).

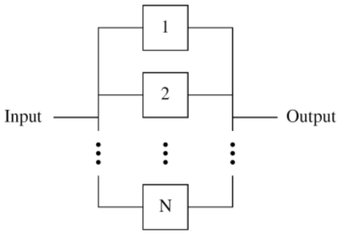


## Reliability block analysis

- **Goal:** compute reliability of a system from the reliability of its components
- **Serial composition**



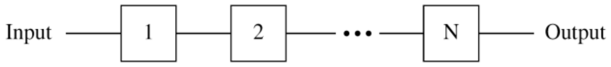
- **Parallel composition**



## Inductive computation of reliability

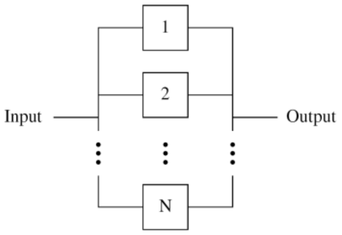
- **Assumption:** failures of the individual components are independent
- **Serial composition**

$$\prod_{i=1}^{N} R_i(t)$$

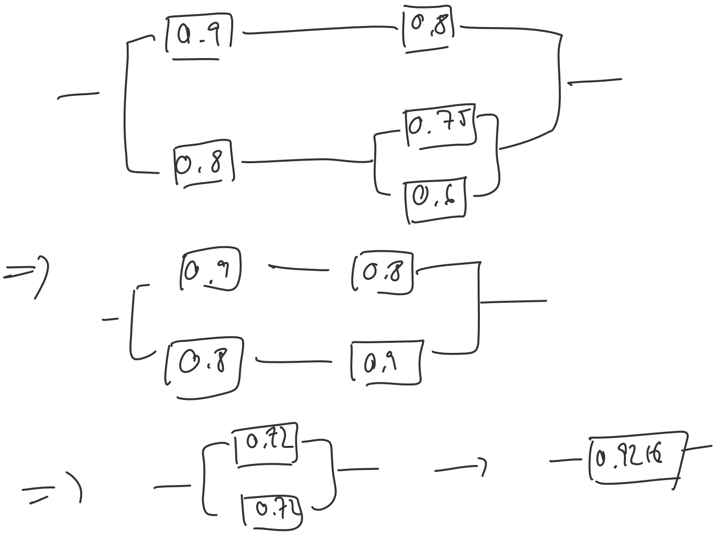


- **Parallel composition**

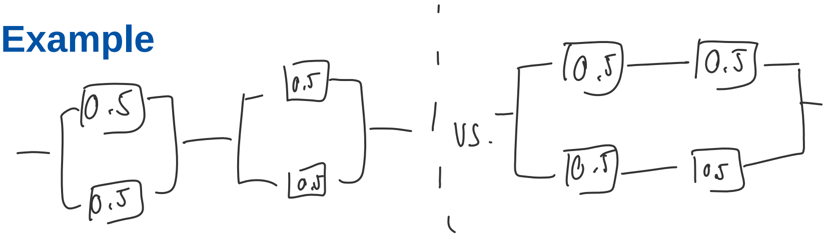
$$1 - \prod^{N} (1 - R_i(t))$$



# Example



# Example



$$R = \left(1-\left(1-\tfrac{1}{2}\right)^2\right)^2 = \left(\tfrac{3}{4}\right)^2 = \frac{9}{16}$$

✓

$$R = 1-\left(1-\left(\tfrac{1}{2}\right)^2\right)^2$$

$$= 1-\left(\tfrac{3}{4}\right)^2 = 1-\frac{9}{16}$$

$$= \frac{7}{16}$$

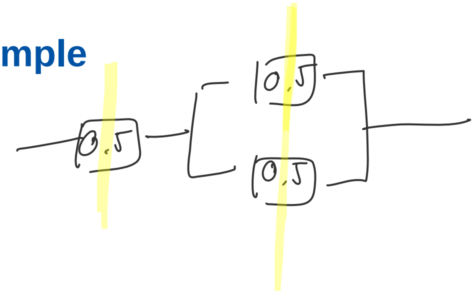
"Redundancy is more efficient at the component level"

# Approximation: Minimal Cuts

- A **minimal cut** is a minimal set of components such that their simultaneous failure causes a system failure
- $$1 - \sum_{j \in MinimalCuts} \prod_{i \in j} [1 - R_i(t)]$$

  is a lower bound for the reliability R(t) of the full system.
- Minimal cuts with a single component are called *single point failures*.

# Example



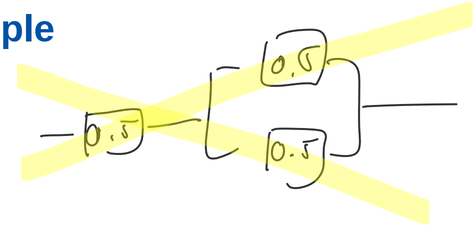
$$R \geq 1 - [(1-0.5) + (1-0,5)^2]$$
$$= 1 - [0,5 + 0,25] = 0,25$$

## Approximation: Minimal Tie Sets

- A **minimal tie set** is a minimal set of components such that their simultaneous functioning guarantees the functioning of the system
- $$\sum_{j \in MinimalTie\;s} \; \prod_{i \in j} R_i(t)$$

  is an upper bound for the reliability R(t) of the full system.

## Example



$0.25 \leq R \leq 0.5$

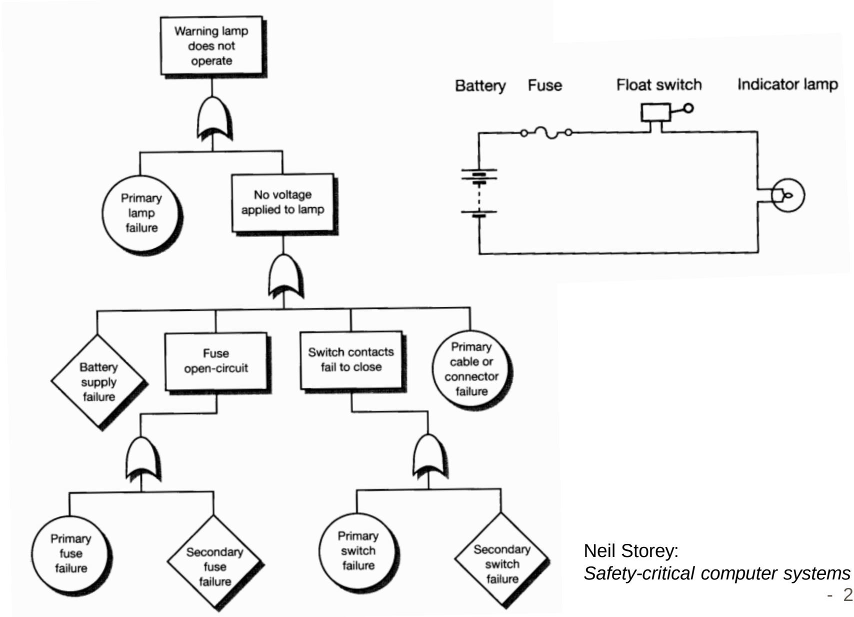
$R \leq 0,5 \cdot 0,5 + 0,5 \cdot 0,5 = 0.5$

$$\left( \text{Exact}: 0.5 \cdot (1 - (1-0,5)^2) = 0,5 \cdot 0.75 = 0,375 \right)$$

## Fault tree Analysis (FTA)

- FTA is a top-down method of analyzing risks. Analysis starts with possible damage, tries to come up with possible scenarios that lead to that damage.
- FTA typically uses a graphical representation of possible damages, including symbols for AND- and OR-gates.
- OR-gates are used if a single event could result in a hazard.
- AND-gates are used when several events or conditions are required for that hazard to exist.

## Example: Brake fluid warning lamp



Neil Storey:
*Safety-critical computer systems*

## Direct Analysis

$$1 - \sum_{\vec{p} \in \{0,1\}^n} (FT(\vec{p}) \cdot \prod_{i=1}^{n} (1 - R_i(t))^{p_i} \cdot R_i(t)^{1-p_i})$$

where

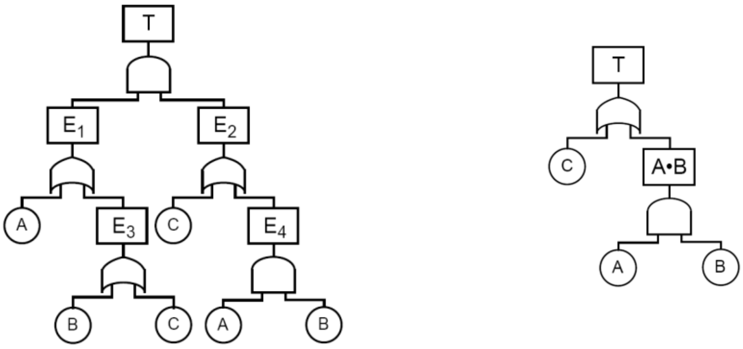
$\vec{p} = (p_1,\ldots, p_n)$ denotes the occurrence of the base events, and

$FT(\vec{p})$ denotes the value of the top event

Problem: combinatorial explosion!

## Equivalence

- Two fault trees are equivalent if the associated logical formulas are equivalent.
- E.g., $(A \vee (B \vee C) \wedge (C \vee (A \wedge B)))$ ≡ $(C \vee (A \wedge B))$



## Minimal cut sets

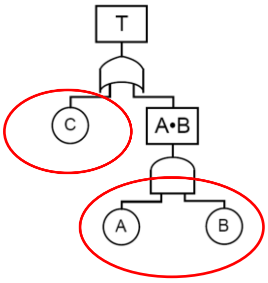
Minimal cut set = “smallest set of basic events which, in conjunction, cause the top level event to occur”.

Logically: Disjunctive Normal Form (DNF) = disjunction of conjunctions of basic events.

**Example:**

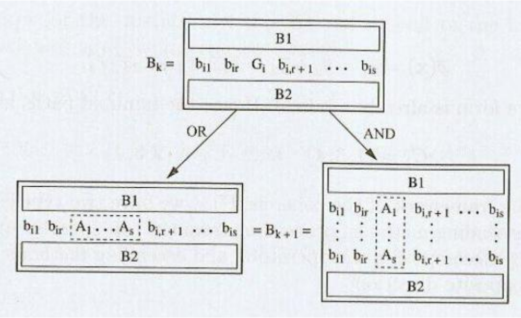
C (single point of failure) and

A ∧ B.



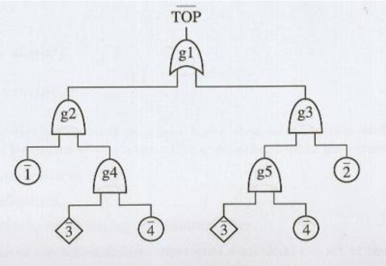
## Mocus Algorithm (1972) „Method of Obtaining Cut Sets“

- Initialize the first element of a matrix with the top event operator
- As long as there is still an operator in the matrix:
  - If it is an AND operator, replace it with its inputs in the column
  - If it is an OR operator, replace it with its inputs in the row.
- Each column corresponds to a cut set; reduce to obtain minimal cut sets.


$B_k =$
B1
$b_{i1}$ $b_{ir}$ $G_i$ $b_{i,r+1}$ … $b_{is}$
B2
OR
AND
B1
$b_{i1}$ $b_{ir}$ $A_1 \ldots A_s$ $b_{i,r+1}$ … $b_{is}$
B2
$= B_{k+1} =$
B1
$b_{i1}$ $b_{ir}$ $A_1$ $b_{i,r+1}$ … $b_{is}$
$b_{i1}$ $b_{ir}$ $A_s$ $b_{i,r+1}$ … $b_{is}$
B2

Nikolaos Limnios: *Fault Trees*

## Example



$$[\underline{\underline{g_1}}] \rightarrow [\underline{\underline{g_2}}\ g_3]$$

$$\rightarrow \begin{bmatrix} 1 & \underline{\underline{g_3}} \\ g_4 & g_3 \end{bmatrix}$$

$$\rightarrow \begin{bmatrix} 1 & 2 \\ 1 & \underline{\underline{g_5}} \\ g_4 & g_3 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 2 \\ 1 & 3 \\ 1 & 4 \\ \underline{\underline{g_4}} & g_3 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 2 \\ 1 & 3 \\ 1 & 4 \\ 3 & \underline{\underline{g_3}} \\ 4 & \underline{\underline{g_3}} \end{bmatrix} \rightarrow \ldots \begin{bmatrix} 12 \\ 13 \\ 14 \\ 32 \\ 34 \\ 32 \\ 42 \\ 43 \\ 44 \end{bmatrix}$$

$\rightarrow$ cut sets: $\{1,3,4\}$, $\{2,1,4\}$

## Limitations of combinatorial models

- Assumption that failure probability is independent of the system state is often wrong.

**Example:** cold-spare redundancy

- Failure during standby is unlikely
- Failure during activation is likely

⇒ state-based models are required

## Markov Chains

A (discrete-time) **Markov chain** consists of

- a finite set of states Q
- an initial distribution i: Q → [0,1]
- a transition probability function t: Q × Q → [0,1]

such that $\sum_{q \in Q} i(q) = 1$ and

$\sum_{q' \in Q} t(q, q') = 1$ for every $q' \in Q$

## State probabilities

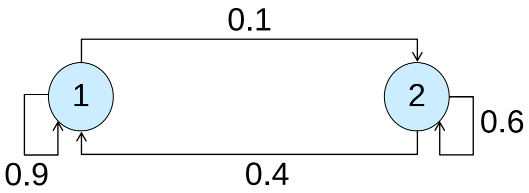
Given the initial distribution i and the transition probabilty function t, we can compute the probability that the system is in a given state after n steps:

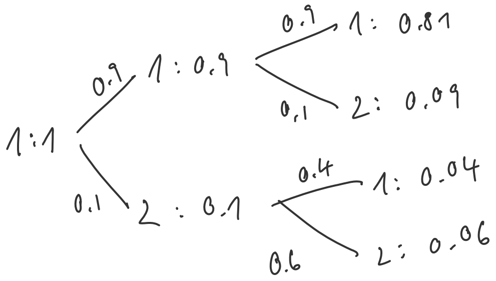
$$s_0(q) = i(q)$$
$$s_{n+1}(q) = \sum_{r \in Q} (s_n(r) t(r, q))$$

## Example

$i(1) = 1$

$i(2) = 0$





1: 0.85
2 : 0.15

## Limit probabilities

- Simple reliability models often have strictly positive transition probabilities:
  $t(q, q') > 0$ for all $(q, q') \in Q \times Q$

- Then, the limit probabilities $s_{lim} = \lim_{n\to\infty} s_n$ exists and can be computed by solving the linear equation system

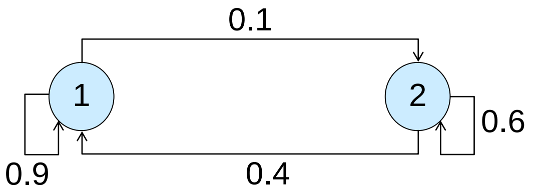
$$\bigwedge_{q\in Q} s_{lim}(q) = \sum_{r\in Q} (s_{lim}(r) t(r, q))$$

$$1 = \sum_{q\in Q} s_{lim}(q)$$

## Example

$i(1) = 1$

$i(2) = 0$



$$S_1 = S_1 \cdot 0.9 + S_2 \cdot 0.4$$
$$S_2 = S_1 \cdot 0.1 + S_2 \cdot 0.6$$
$$S_1 + S_2 = 1$$

$$-0.1 S_1 + 0.4 S_2 = 0$$
$$0.1 S_1 - 0.4 S_2 = 0$$
$$S_1 + S_2 = 1$$

$$0.1 S_1 - 0.4(1 - S_1) = 0$$
$$0.5 S_1 - 0.4 = 0$$
$$S_1 = 0.8$$
$$S_2 = 0.2$$

## Safety cases

- In a “safety case”, an independent authority has to be convinced that certain technical equipment is indeed safe.
- One of the commonly requested properties of technical systems is that no single failing component should potentially cause a catastrophe.

## Dependability requirements

- Allowed failures may be in the order of 1 failure per $10^9$ h.
- ~ 1000 times less than typical failure rates of chips.

☞ For safety-critical systems, the system as a whole must be more dependable than any of its parts.

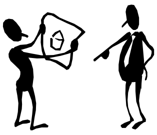
☞ fault-tolerance mechanisms must be used.

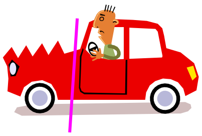
Low acceptable failure rate → systems not 100% testable.

☞ Safety must be shown by a combination of testing and reasoning. Abstraction must be used to make the system explainable using a hierarchical set of behavioral models. Design faults and human failures must be taken into account.

## Kopetz‘s 12 design principles (1-3)

1. Safety considerations may have to be used as the important part of the specification, driving the entire design process.
2. Precise specifications of design hypotheses must be made right at the beginning. These include expected failures and their probability.
3. Fault containment regions (FCRs) must be considered. Faults in one FCR should not affect other FCRs.



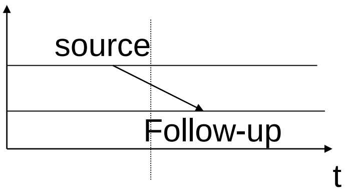


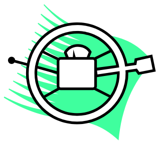
Passenger compart- ment stable

Safety-critical & non-safety critical electronics

## Kopetz‘s 12 design principles (4-6)

4. A consistent notion of time and state must be established. Otherwise, it will be impossible to differentiate between original and follow-up errors.
5. Well-defined interfaces have to hide the internals of components.
6. It must be ensured that components fail independently.



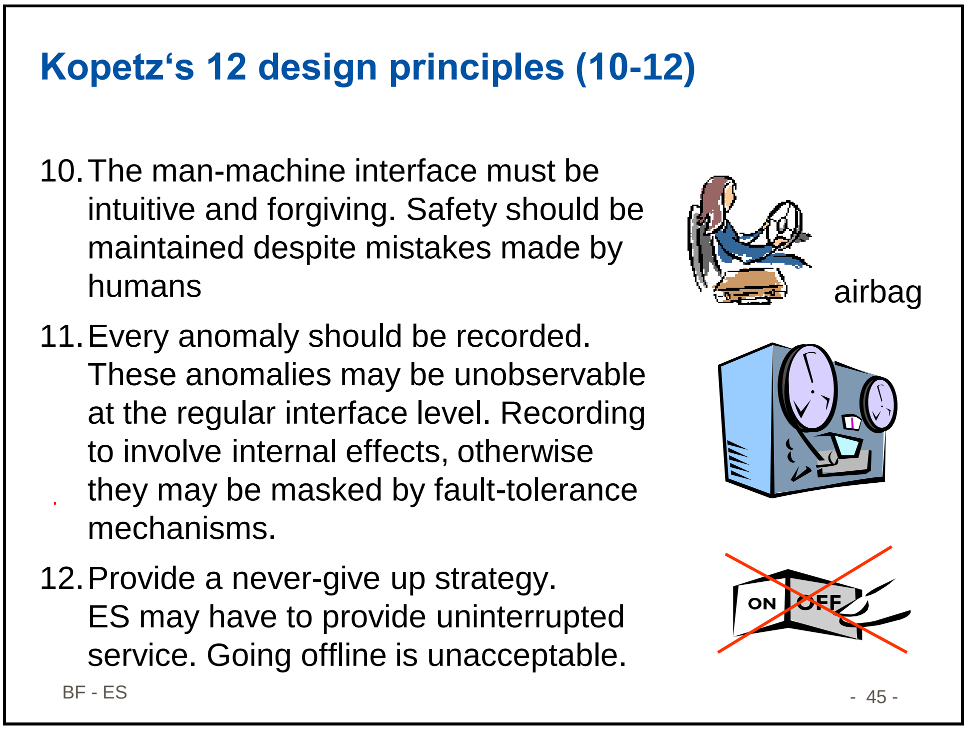


2 independent brake hose systems

## Kopetz‘s 12 design principles (7-9)

7. Components should consider themselves to be correct unless two or more other components pretend the contrary to be true (principle of self-confidence).
8. Fault tolerance mechanisms must be designed such that they do not create any additional difficulty in explaining the behavior of the system. Fault tolerance mechanisms should be decoupled from the regular function.
9. The system must be designed for diagnosis. For example, it has to be possible to identifying existing (but masked) errors.

one of the systems sufficient for braking



## Kopetz‘s 12 design principles (10-12)

10. The man-machine interface must be intuitive and forgiving. Safety should be maintained despite mistakes made by humans

airbag

11. Every anomaly should be recorded. These anomalies may be unobservable at the regular interface level. Recording to involve internal effects, otherwise they may be masked by fault-tolerance mechanisms.

12. Provide a never-give up strategy. ES may have to provide uninterrupted service. Going offline is unacceptable.