TTTech

Part 3:

Fault-tolerance and Modeling

Goals of fault-tolerance modeling

- **Design phase**

  Designing and implementing the computer system to achieve the dependability required.

  – avoid construction of costly prototypes
  – modeling is used to evaluate design alternatives
  – analysis of critical components
  – for prediction of dependability parameters
    (safety, reliability, availability, maintainability, ...)

- **Validation phase:**

  Gaining confidence that a certain dependability goal (requirement) has been attained.

  – modeling is used to evaluate the computer system
  – certification

Modeling techniques

- **Deterministic modeling**
  - the maximum number of faults that can be tolerated without system failure is considered
  - the evaluation criteria is ***n*-resiliency**, i.e. a system is said to be *n*-resilent if it can tolerate up to *n* component failures
- **Probabilistic (quantitative) modeling**
  - component failure and repair rates are described as stochastic processes
  - consideration of failure rates
  - statistical models

Probabilistic modeling

There are three different forms of information which can be used to model a system's dependability.

- **Historical information** (statistics)**:**
  Information about the behavior of identical or similar components in the past is assessed.
- **Experimental information** (statistics)**:**
  Information is gained by exercising the system or single components (for software this includes typically the test and debugging)
- **Structural information:**
  Overall dependability of a system is deduced from the structure and the dependability figures of its parts

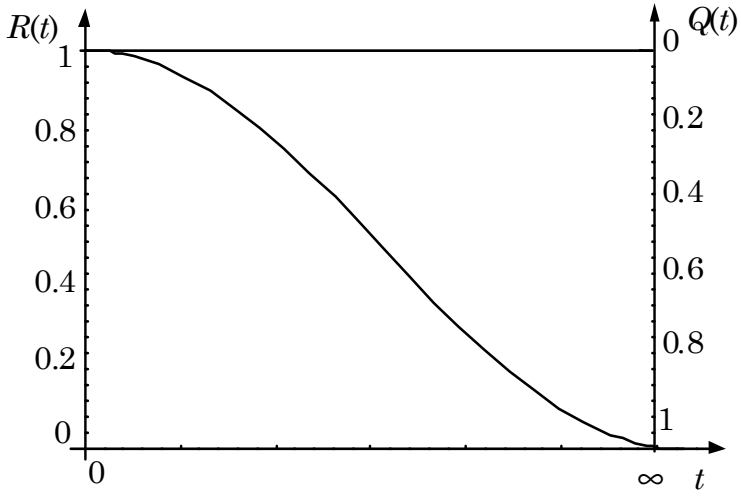
Probabilistic functions

- **Reliability** $R(t)$

  is the probability that the system will conform to its specification throughout a period of duration t.

- **Failure Probability** $Q(t)$

  is the probability that the system will **not** conform to its specification throughout a period of duration t.



$$R(0) = 1 \quad R(\infty) = 0$$

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

Probability density function

Def.: The failure density $f(t)$ at time t is defined by the number of failures during Δt.

$$f(t) = \frac{dQ(t)}{dt} = -\frac{dR(t)}{dt}$$

Failure rate

Def.: The failure rate $\lambda(t)$ at time t is defined by the number of failures during Δt in relation to the number of correct components at time t.

$$\lambda(t) = \frac{f(t)}{R(t)}$$
$$= -\frac{dR(t)}{dt}\frac{1}{R(t)}$$

Constant failure rate

Used to model the normal-life period of the bathtub curve

- **failure rate**

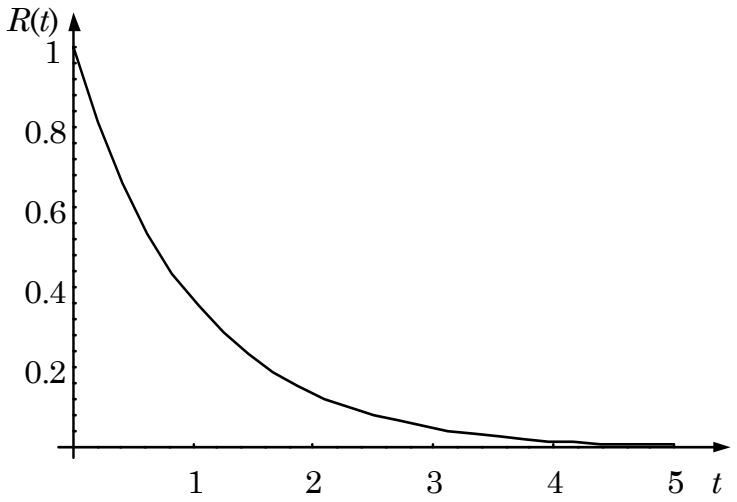
$$\lambda(t) = \lambda$$

- **probability density function**

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

- **reliability**

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



Reliability for constant failure rate

Failure rates

Devices	FIT	Device	FIT
TTL-SSI, -MSI	5	Transistor (bip.)	3
CMOS-SSI	3	Transistor (FET)	3
RAM (Š 1 MBit)	20	Power Transistor	40
RAM (> 1 MBit)	45	Diode	3
PLD	130	Opto-Coupler	5
EPROM (Š 1 MBit)	10	LED/LCD-Display	15
EPROM (> 1 MBit)	20	Resistors	1
µC, µP	20	Lamp 12V	500
DSP	40	Lamp 24V	1000
CMOS-GA (Š 100 kGates)	35	Condensator	3
CMOS-GA (> 100 kGates)	70	Switch (per contact)	7
OP-Amp	3	Relais	70
Analog custom design	45	Solder joint	0.5

FIT = failures / 10^9 [h]

Weibull distributed failure rate

Used to model infant mortality and wear out period of components.

α < 1: failure rate is decreasing with time
α = 1: constant failure rate
α > 1: failure rate is increasing with time

- **failure rate**

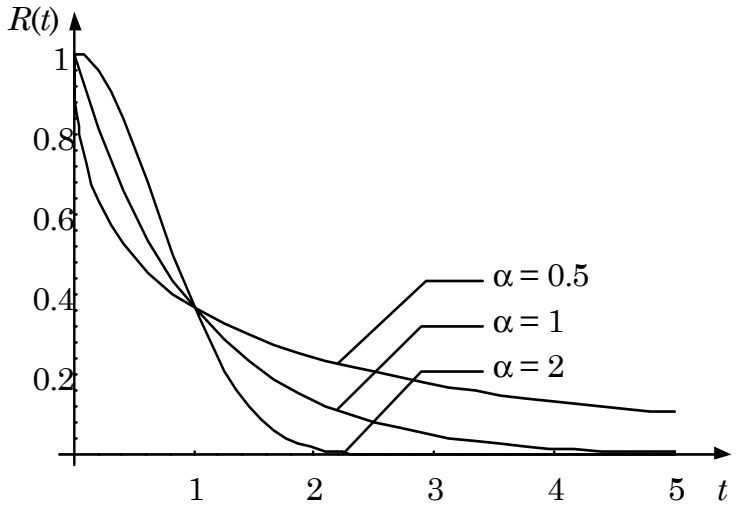
$$\lambda(t) = \alpha\lambda(\lambda t)^{\alpha-1}$$

- **probability density function**

$$f(t) = \alpha\lambda(\lambda t)^{\alpha-1} e^{-(\lambda t)^{\alpha}}$$

- **reliability**

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



Reliability for weibull distributed failure rate

Lognormal distributed failure rate

For semiconductors the lognormal distribution fits more data collections than any other and is assumed to be the proper distribution for semiconductor life.

- **failure rate**

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

- **probability density function**

$$f(t) = \frac{1}{\sigma t\sqrt{2\pi}}\, e^{-\frac{1}{2}\left(\frac{\ln t - \mu}{\sigma}\right)^2}$$

- **reliability**

$$R(t) = 1 - \frac{1}{\sigma\sqrt{2\pi}} \int_0^t \frac{1}{x}\, e^{-\frac{1}{2}\left(\frac{\ln t - \mu}{\sigma}\right)^2} dx$$

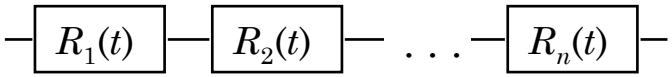
Probabilistic structural based modeling

- **Assumptions:**
  - identifiable (independent) components
  - each component is associated with a given failure rate
  - model construction is based on the structure of the interconnections between components
- **Models:**
  - Simple block diagrams
  - Arbitrary block diagrams
  - Markov models
  - Generalized Stochastic Petri Nets (GSPN)

Simple block diagrams

- assumption of independent components
- combination of series or parallel connected components

Series connection

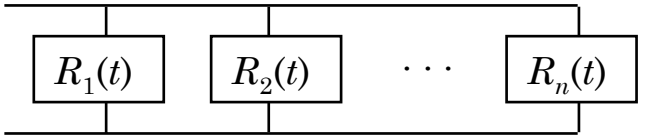


$$R_{series}(t) = \prod_{i=1}^{n} R_i(t)$$

$$Q_{series}(t) = 1 - R_{series}(t) = 1 - \prod_{i=1}^{n} R_i(t)$$

$$= 1 - \prod_{i=1}^{n} (1 - Q_i(t))$$

Parallel connection



$$Q_{parallel}(t) = \prod_{i=1}^{n} Q_i(t)$$

$$R_{parallel}(t) = 1 - Q_{parallel}(t) = 1 - \prod_{i=1}^{n} Q_i(t)$$

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

Constant failure rate

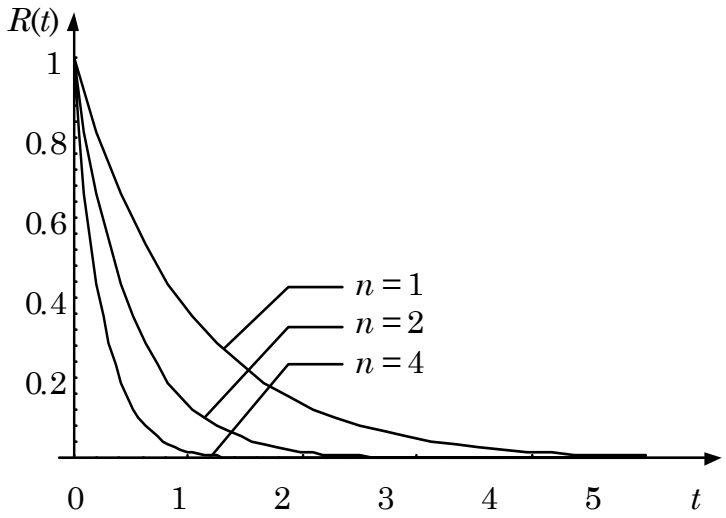
$$\lambda(t) = \lambda$$

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

Series connection

$$R_{series}(t) = \prod_{i=1}^{n} R_i(t) = \prod_{i=1}^{n} e^{-\lambda_i t}$$

$$= e^{-t\sum_{i=1}^{n}\lambda_i}$$



Reliability of 1,2 and 4 series connected components with constant failure rate ($\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4$)

- the resulting failure rate for the system is still constant

Parallel connection

$$R_{parallel}(t) = 1 - \prod_{i}^{n} (1 - R_i(t))$$

$$= 1 - \prod_{i}^{n} \left(1 - e^{-\lambda_i t}\right)$$

for 3 parallel components this gives:

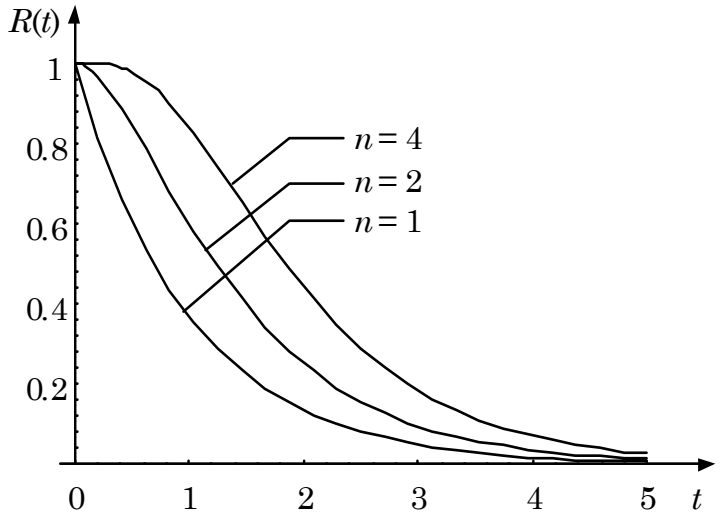
$$R_{parallel}(t) = 1 - \left(\left(1 - e^{-\lambda_1 t}\right)\left(1 - e^{-\lambda_2 t}\right)\left(1 - e^{-\lambda_3 t}\right)\right)$$

$$= e^{-\lambda_1 t} + e^{-\lambda_2 t} + e^{-\lambda_3 t} + e^{-(\lambda_1+\lambda_2+\lambda_3)t} -$$

$$e^{-(\lambda_1+\lambda_2)t} - e^{-(\lambda_1+\lambda_3)t} - e^{-(\lambda_2+\lambda_3)t}$$

under the assumption $\lambda_1 = \lambda_2 = \lambda_3$ it follows

$$R_{parallel}(t) = 3\left(e^{-\lambda t} - e^{-2\lambda t}\right) + e^{-3\lambda t}$$

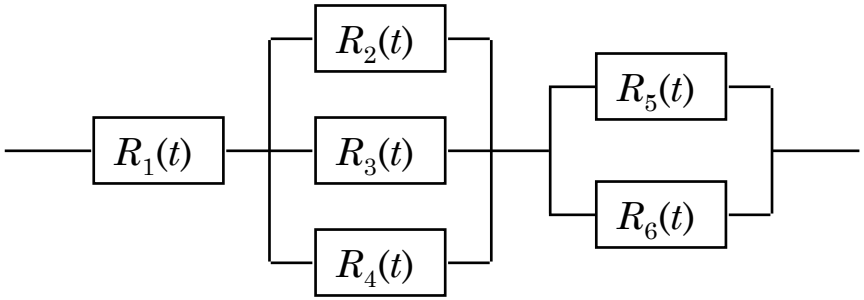


Reliability of 1,2 and 4 parallel connected components with constant failure rate ($\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4$)

the resulting failure rate is no longer constant

Simple block diagrams

- can be used to model arbitrary combinations of series and parallel connected components
- easy mathematics for constant failure rates

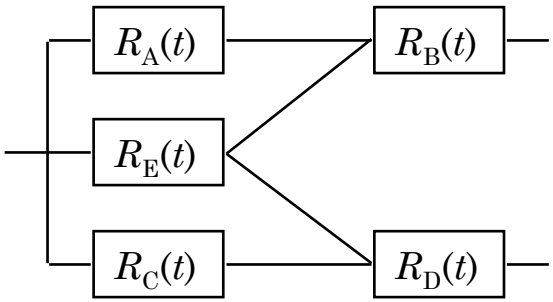


Problems

- assumption of independent failures
- maintenance cannot be modeled
- restricted to series/parallel connection
- only for active redundancy and fail-silence

Arbitrary block diagrams

- no restriction to series/parallel connections



$$R_{block}(t) = R_{AB} + R_{BE} + R_{DE} + R_{CD} - R_{ABE} - R_{ABCD} - R_{BDE} - R_{CDE} + R_{ABCDE}$$

$$R_{ABC} = R_{series}(A, B, C)$$

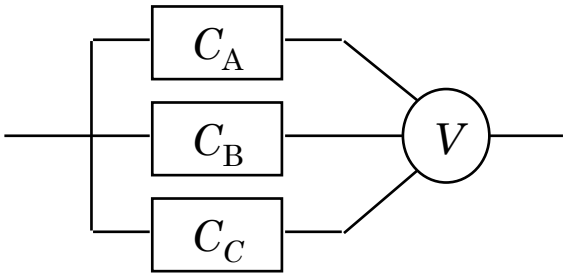
Inclusion/exclusion principle

1:	A	B				+
2:		B			E	+
3:				D	E	+
4:			C	D		+
12:	A	B			E	–
13:	A	B		D	E	–
14:	A	B	C	D		–
23:		B		D	E	–
24:		B	C	D	E	–
34:			C	D	E	–
123:	A	B		D	E	+
124:	A	B	C	D	E	+
134:	A	B	C	D	E	+
234:		B	C	D	E	+
1234:	A	B	C	D	E	–

Active redundancy and voting

- for TMR 2 out of 3 components have to function correctly

$$R_{TMR}(t) = R(C_A, C_B, C_C, t) + R(C_A, C_B|\overline{C}_C, t) + R(C_A, C_C|\overline{C}_B, t) + R(C_B, C_C|\overline{C}_A, t)$$



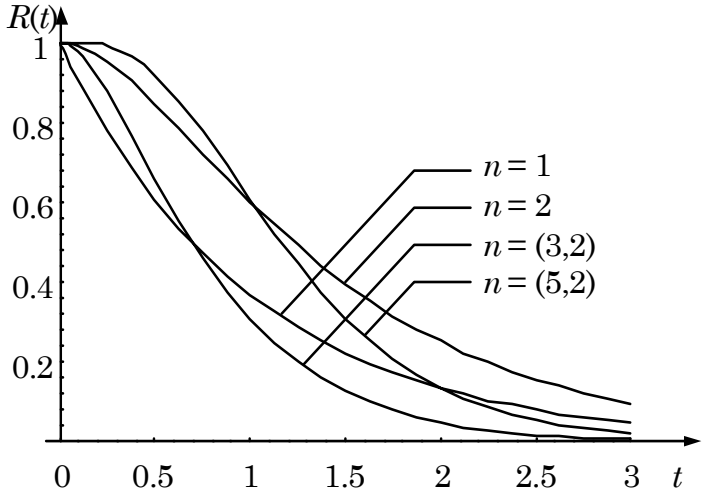
- under the assumption of identical failure rates

$$R_{TMR}(t) = R(t)^3 + 3R(t)^2 Q(t)$$

- for general voting systems where *c* out of *n* components have to function correctly

$$R_{NMR}(t) = \sum_{k=c}^{n} \binom{n}{k} \left(e^{-\lambda t}\right)^k \left(1 - e^{-\lambda t}\right)^{n-k}$$

Parallel fail silent components vs. majority voting



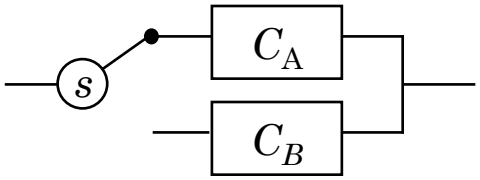
$n = 1$	single component
$n = 2$	two parallel components
$n = (3,2)$	voting, 2 out of 3
$n = (5,2)$	voting, 2out of 5

Neglected issues:

- coverage of fail silence assumption
- reliability of voter

Passive redundancy

- probability that A is performing correctly plus conditional probability that B is performing correctly and A has failed



$$R(t) = R(C_A) + R(C_B|\overline{C}_A)$$

- under the assumption of constant failure rates $\lambda_A = \lambda_B$

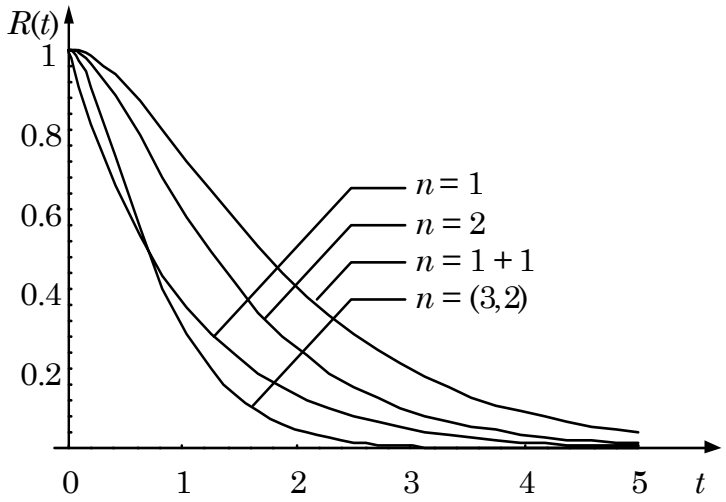
$$R(t) = e^{-\lambda t} + \sum_{x=0}^{t} R_B(t-x+\Delta x)\frac{[R_A(x) - R_A(x+\Delta x)]\Delta x}{\Delta x}$$

$$\Delta x \to 0: \quad e^{-\lambda t} + \int_{x=0}^{t} R_B(t-x) f(x)\, dx$$

$$= e^{-\lambda t} + \int_{x=0}^{t} e^{-\lambda(t-x)} \lambda e^{-\lambda x} dx$$

$$= e^{-\lambda t}(1+\lambda t)$$

Passive vs. active redundancy



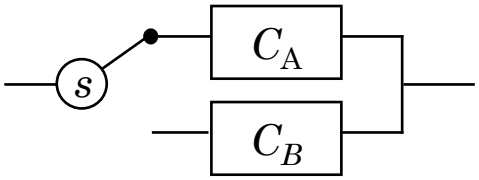
$n = 1$	single component
$n = 2$	two parallel components
$n = (3,2)$	voting, 2 out of 3
$n = 1 + 1$	one passive backup

Neglected issues:

- coverage of fail silence assumption
- reliability of switch

Passive redundancy with an unreliable switch

- assumption that the switch functions correctly with probability $R_s(t)$
- the system reliability is the probability that A is performing correctly plus the conditional probability that B is performing correctly and A has failed **and** the switch still functions correctly

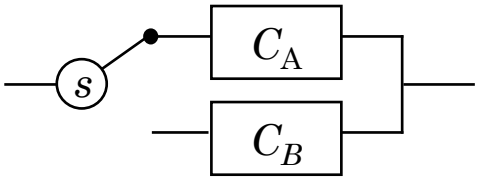


$$R(t) = e^{-\lambda t} + \sum_{x=0}^{t} R_B(t-x+\Delta x)\, R_s(t)\, \left[R_A(x) - R_A(x-\Delta x)\right]$$

$$= e^{-\lambda t} + \int_{x=0}^{t} e^{-\lambda (t-x)}\; e^{-\lambda_s t}\; \lambda e^{-\lambda x} dx$$

Passive redundancy with limited error detection coverage

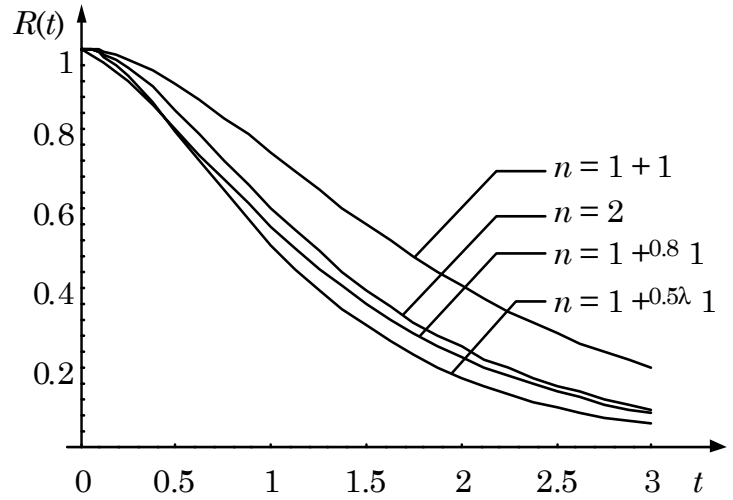
- assumption that errors of component A are not always detected, the error detection coverage is given by c
- the system reliability is the probability that A is performing correctly plus the conditional probability that B is performing correctly and A has failed **and** A's error has been detected



$$R(t) = e^{-\lambda t} + \sum_{x=0}^{t} c\, R_B(t - x + \Delta x)\left[R_A(x) - R_A(x - \Delta x)\right]$$

$$= e^{-\lambda t} + \int_{x=0}^{t} c\, e^{-\lambda(t-x)} \lambda e^{-\lambda x} dx$$

Perfect vs. imperfect passive redundancy



$n = 1 + 1$	one passive backup
$n = 2$	two parallel components
$n = 1 +^{0.8} 1$	error detection coverage 80%
$n = 1 +^{0.5\lambda} 1$	reliability of switch is 0.5λ

- under practical conditions it is impossible to build an *ideal* passive replicated system
- an unreliable switch with $\lambda_s = 0.5\lambda$ or a switch with error detection coverage of 80% reduces the system reliability below that of active redundant components

Single parametric measures

- **Mean time to failure:**

$$MTTF = \int_0^\infty t\, f(t)\, dt$$

- **Mean time to repair:**

$$MTTR = \int_0^\infty t\, f_r(t)\, dt$$

- **Mission reliability:**

$R_m = R(t_m)$ t_m ... mission duration

- **(Steady state) availability:**

$$A = \frac{MTTF}{MTTF + MTTR}$$

Mean time to failure

- **Constant failure rate:**

$$MTTF = \int_0^\infty t\, f(t)\, dt = \int_0^\infty t\, \lambda e^{-\lambda t}\, dt = \frac{1}{\lambda}$$

- **Weibull distributed failure rate:**

$$MTTF = \int_0^\infty t\, \alpha\lambda(\lambda t)^{\alpha-1} e^{-(\lambda t)^\alpha}\, dt = \frac{\Gamma(1+\alpha^{-1})}{\lambda}$$

- **Serial connected components:**

$$MTTF_{series} = \frac{1}{\lambda_1 + \lambda_2 + \cdots + \lambda_n}$$

- **Passive redundancy:**

$$MTTF_{passive} = \frac{1}{\lambda_1} + \frac{1}{\lambda_2} + \cdots + \frac{1}{\lambda_n}$$

- **Parallel connected components:**

$$MTTF_{parallel} = \frac{1}{\lambda}\left(1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}\right)$$

Repair rate

- repair rate $\mu(t)$ analogous to failure rate
- most commonly constant repair rates $\mu(t) = \mu$

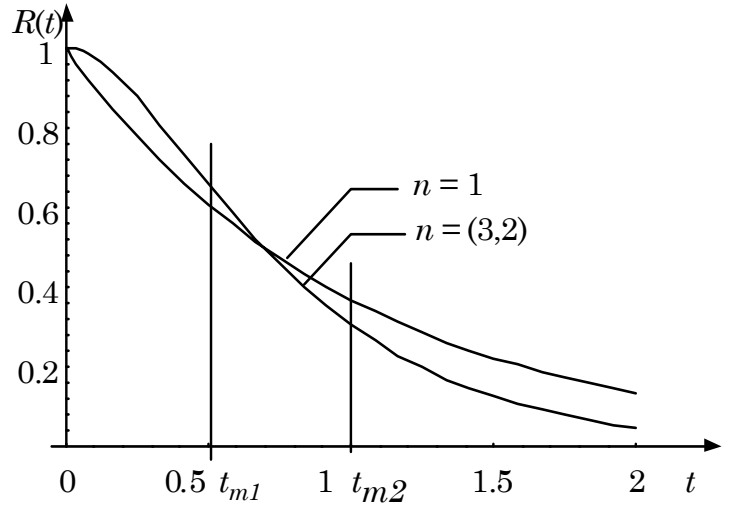
Mean time to repair

- analogous to mean time to failure

$$MTTR = \frac{1}{\mu}$$

Mission reliability

- assumption of a mission time t_m
- during mission there is no possibility of maintenance or repair
- typical examples are space flights or air planes
- suitability of architectures depends on mission time



Availability

- the percentage of time for which the system will conform to its specification
- also called steady state or instantaneous availability

$t \to \infty$:

$$A = \frac{MTTF}{MTBF} = \frac{MTTF}{MTTF + MTTR}$$

mean time between failures (*MTBF*)

- without maintenance and repair

$MTTR = \infty$: $A = 0$

- Mission availability

$t \to t_m$:

$$A_m = \frac{1}{t_m} \int_{t=0}^{t_m} R(t)\,dt$$

Markov models

- suitable for modeling of:
  - arbitrary structures
    (active, passive and voting redundancy)
  - systems with complex dependencies
    (assumption of independent failures is no longer necessary)
  - coverage effects
- Markov property:
  The system behavior at any time instant *t* is independent of history
  (except for the last state).
- restriction to constant failure rates

The two phases for Markov modeling

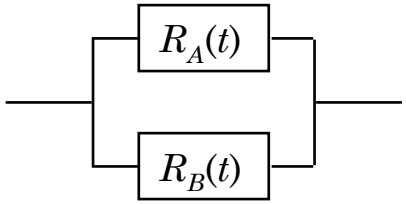
- **Model design:**
  - identification of relevant system states
  - identification of transitions between states
  - construction of Markov graph with transition rates
- **Model evaluation:**

  Differential equation
  Solution of equation gives $R(t)$
  - explicit (by hand)
  - Laplace transformation
  - numeric solution (tool based)

  Integration of differential equation gives *MTTF*
  - system of linear equations

Example model for active redundant system

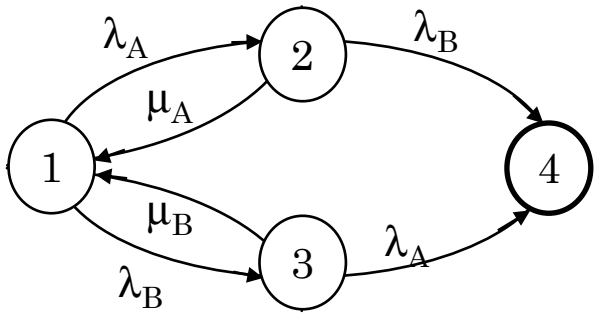
Two parallel connected components A and B with maintenance. Failure rates are λ_A and λ_B, repair rates are μ_A and μ_B.



- **Identification of system states:**

1:	A correct	B correct	$P_1(t)$
2:	A failed	B correct	$P_2(t)$
3:	A correct	B failed	$P_3(t)$
4:	A failed	B failed	$P_4(t)$

- **Construction of Markov Graph**



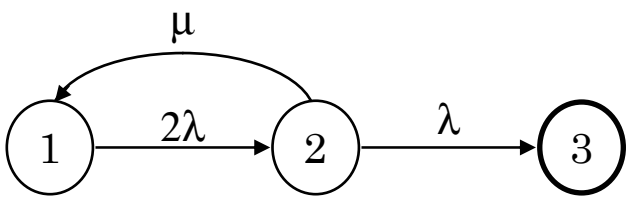
Active redundancy with identical components

- failure rates: $\lambda_A = \lambda_B = \lambda$
- repair rates: $\mu_A = \mu_B = \mu$

- **Identification of system states:**

  1: A correct B correct $P_1(t)$
  2: one failed one correct $P_2(t)$
  3: A failed B failed $P_3(t)$

- **Construction of Markov Graph**



- **Differential equations:**

$$\frac{dP_1(t)}{dt} = -2\lambda P_1(t) + \mu P_2(t)$$
$$\frac{dP_2(t)}{dt} = 2\lambda P_1(t) - (\mu+\lambda) P_2(t)$$
$$\frac{dP_3(t)}{dt} = \lambda P_2(t)$$

MTTF evaluation from Markov model

- In a Markov model the MTTF is given by the period during which the system exhibits states that correspond to correct behavior.
- for the active redundant example system:

$$MTTF = \int_{t=0}^{\infty} (P_1(t) + P_2(t))dt = T_1 + T_2$$

$$T_1 = \int_{t=0}^{\infty} P_1(t)dt \qquad T_2 = \int_{t=0}^{\infty} P_2(t)dt$$

- state probabilities for $t = 0$ and $t = \infty$

$$P_1(0) = 1 \quad P_1(\infty) = 0$$
$$P_2(0) = 0 \quad P_2(\infty) = 0$$
$$P_3(0) = 0 \quad P_3(\infty) = 1$$

MTTF evaluation from Markov model (cont.)

- integration of differential equation

$$\begin{aligned}
\frac{d\,P_1(t)}{dt} &= -2\lambda P_1(t) + \mu P_2(t) \\
\frac{d\,P_2(t)}{dt} &= 2\lambda P_1(t) - (\mu+\lambda) P_2(t) \\
\frac{d\,P_3(t)}{dt} &= \lambda P_2(t)
\end{aligned}
\Longrightarrow
\begin{aligned}
0-1 &= -2\lambda T_1 + \mu T_2 \\
0-0 &= 2\lambda T_1 - (\mu+\lambda) T_2 \\
1-0 &= \lambda T_2
\end{aligned}$$

- solution of linear equation system

$$T_2 = \frac{1}{\lambda}$$

$$T_1 = \frac{\mu+\lambda}{2\lambda} T_2 = \frac{\mu+\lambda}{2\lambda^2} = \frac{1}{2\lambda} + \frac{\mu}{2\lambda^2}$$

$$MTTF = T_2 + T_2 = \frac{3}{2\lambda} + \frac{\mu}{2\lambda^2}$$

Effect of maintenance

- repair and failure rate: $\lambda = \frac{1}{1000}\,[\mathrm{h}]$ $\mu = \frac{1}{10}\,[\mathrm{h}]$

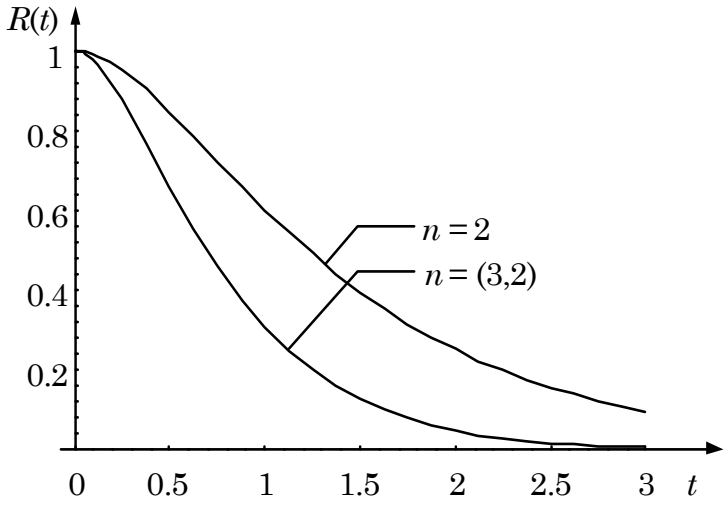
	without maintenance			with maintenance		
	$R(t)$	$MTTF$	h	$R(t)$	$MTTF$	h
2 components in series	$e^{-2\lambda t}$	$\frac{1}{2\lambda}$	500	$e^{-2\lambda t}$	$\frac{1}{2\lambda}$	500
single component	$e^{-\lambda t}$	$\frac{1}{\lambda}$	1000	$e^{-\lambda t}$	$\frac{1}{\lambda}$	1000
2 components in parallel	$2e^{-\lambda t} - e^{-2\lambda t}$	$\frac{3}{2\lambda}$	1500	—	$\frac{3}{2\lambda} + \frac{\mu}{2\lambda^2}$	51500
one passive backup	$e^{-\lambda t}(1+\lambda t)$	$\frac{2}{\lambda}$	2000	—	$\frac{2}{\lambda} + \frac{\mu}{\lambda^2}$	102000

- for 2 active redundant components the MTTF is improved by a factor 34
- for 2 passive redundant components the MTTF is improved by a factor 51

Effect of failure semantics and assumption coverage

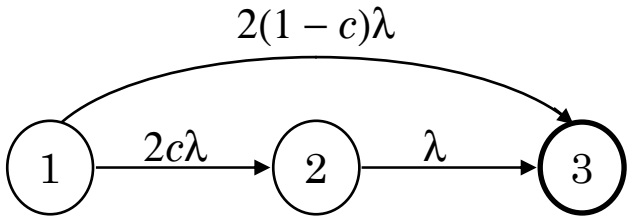
- comparing a system with two active replicated components to a TMR systems shows that under *ideal* conditions active replication has a higher reliability

- **But:** active replication is based on the assumption that components are fail silent
  – assumption coverage ???
- TMR voting is based on the assumption of fail consistent components
  – assumption coverage ≈ 1
  (if properly constructed)



Effect of failure semantics and assumption coverage
(cont.)

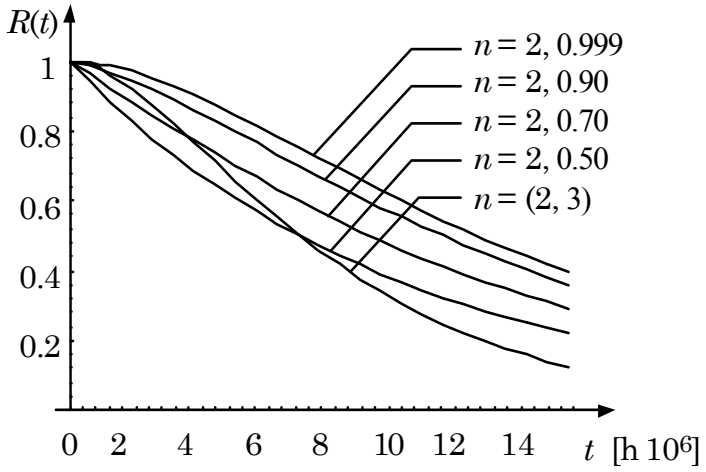
- modeling of the TMR was reasonable since assumption coverage of fail consistent behavior ≈ 1
- modeling of the active redundant system was idealistic since assumption coverage of fail silent behavior < 1
- **Markov model:**
  λ .. failure rate for active redundant parallel connected components
  c .. assumption coverage for fail silent behavior



Effect of failure semantics and assumption coverage (cont.)

- failure rate of a single component: λ = 100 FIT

System	*Description*	*MTTF*
n = 2, 0.999	two parallel components, coverage of fail silent assumption 99.9%	$14.99\ 10^6$
n = 2, 0.90	two parallel components, coverage of fail silent assumption 90%	$14.00\ 10^6$
n = 2, 0.70	two parallel components, coverage of fail silent assumption 70%	$12.00\ 10^6$
n = 2, 0.50	two parallel components, coverage of fail silent assumption 50%	$10.00\ 10^6$
n = (2, 3)	TMR system, coverage of fail consistent assumption 100%	$8.33\ 10^6$

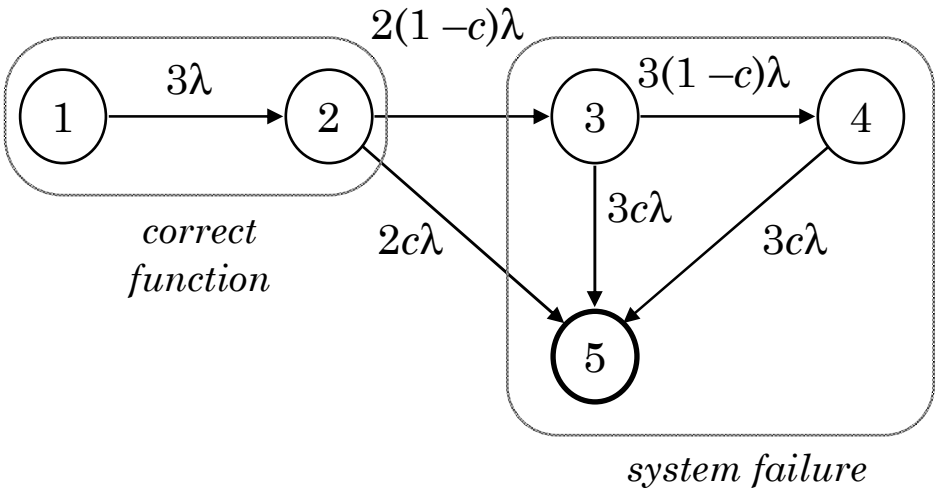


Effect of failure semantics and assumption coverage (cont.)

- comparing parallel components to a TMR systems shows that the reliability of the parallel system is superior for reasonable assumption coverages
- **Safety:**
  from the viewpoint of safety both systems needs to be reevaluated
- **Parallel system:** $R(t) = S(t)$
  for the parallel components the system reliability is equal to the system safety since the system may potentially cause a hazard if it does not function correctly
- **TMR system:** $R(t) < S(t)$
  for TMR systems the reliability is not equal to the safety since the system can be in a safe state although it is not functioning correctly, e.g. all three components disagree

Safety of a TMR system

- to model the safety of a TMR system it needs to be differentiated between incorrect function and the unsafe system state
- **Markov model:**
  λ .. failure rate for single component
  c .. probability of coincident failures of two components

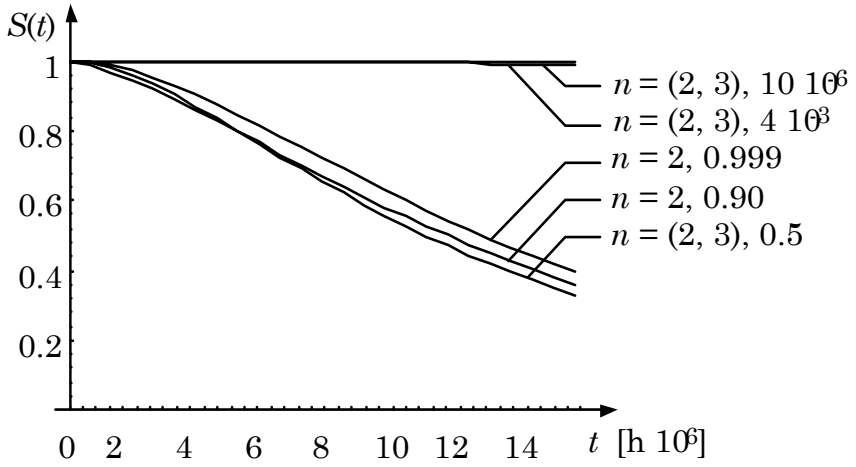


1 .. 3 correct components
2 .. 2 correct, 1 failed comp.
3 .. 1 correct, 2 failed comp.
4 .. 3 failed components
5 .. unsafe state, ≥ 2 coincident component failures

Effect of assumption coverage on safety

- failure rate of a single component: λ = 100 FIT

System	*Description*	$MTTF_S$
n = (2, 3), 10 10^{-6}	TMR system, probability of two coincident failures 10 10^{-6}	333.34 10^9
n = (2, 3), 4 10^{-3}	TMR system, probability of two coincident failures 4 10^{-3}	861.71 10^6
n = (2, 3), 0.5	TMR system, probability of two coincident failures 0.5	13.33 10^6
n = 2, 0.999	two parallel comp., coverage of fail silent assumption 99.9%	14.99 10^6
n = 2, 0.90	two parallel comp., coverage of fail silent assumption 90%	14.00 10^6

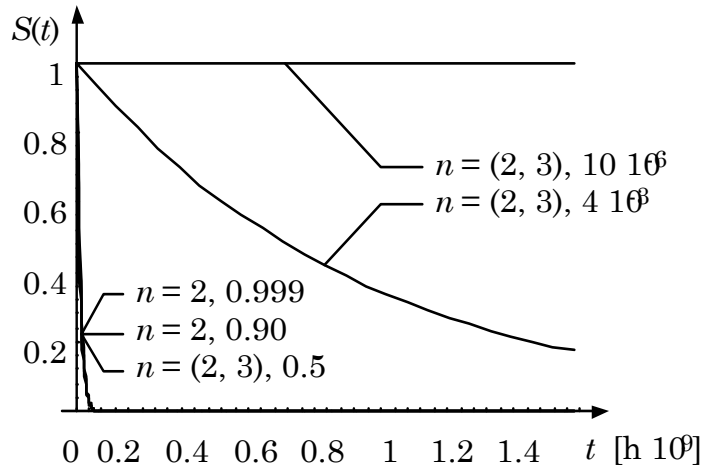


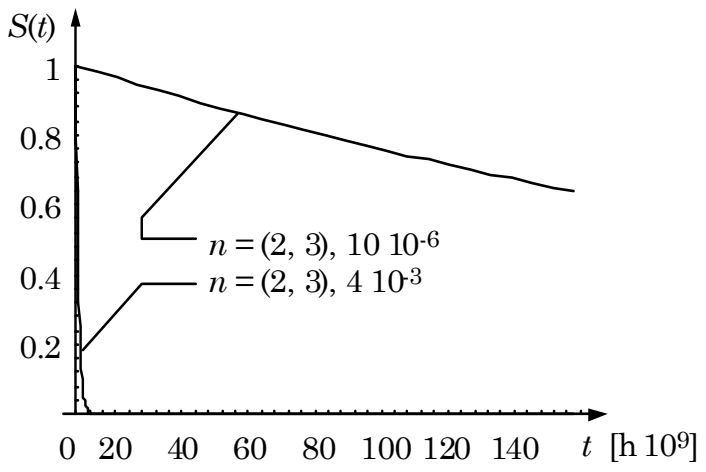
coincidence probability of
two even distributed numbers

16 bit	10 10^{-6}
8 bit	4 10^{-3}
1 bit	0.5

Effect of assumption coverage on safety

- 10 10^{-6} - probability that two 16 bit numbers coincide (independence assumption)
- 4 10^{-3} - probability that two 8 bit numbers coincide ("")
- 0.5 - probability that two 1 bit numbers coincide ("")



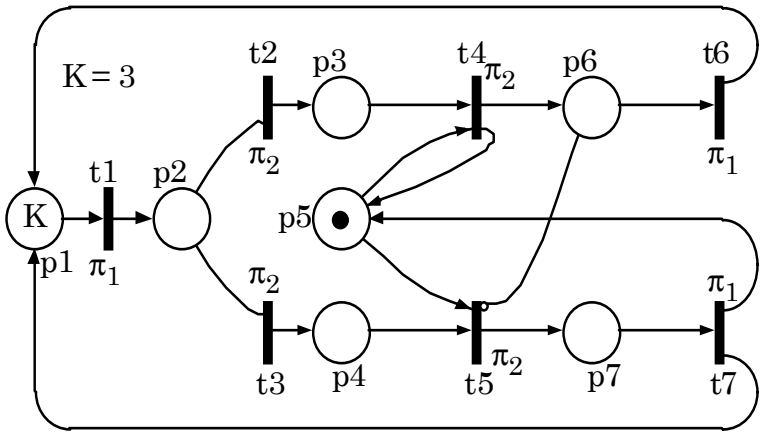


Generalized Stochastic Petri Nets (GSPN)

- because of the very limited mechanisms available, Markov models become easily very complex
- Petri Nets provide much richer mechanisms, they can be used to model and analyze arbitrary systems, algorithms and processes
- basic Petri Nets — which were restricted to discrete behavior only — can be extended to “Generalized Stochastic Petri Nets” by allowing transition delays to be either deterministically equal to zero or exponentially distributed random variables, or to be random variables with different distributions
- it was shown that stochastic Petri Nets are *isomorphic* to continous Markov chains, i.e. for each stochastic Petri Net there exists a functional equivalent Markov chain (and vice versa)

Petri Net Example

Single-writer/multiple-reader access to a shared resource with single access.



p_i ... places

t_i ... transitions

π_i ... transition priorities

- the 3 tokens in place p_1 represents customers that may request the resource
- firing t_1 starts the protocol
- t_2 indicates “read” and t_3 “write” access, respectively
- the single token in p_5 represents the resource

GSPN modeling

To model and analyze a system by means of GSPN the following steps has to be carried out:

- **model construction:**
  usually by means of structured techniques, bottom-up or top-down
- **model validation:**
  structural analysis, possibly formal proves of some behavioral properties
- **definition of performance indices:**
  definition of markings and transition firings (deterministically or stochastic)
- **conversion to Markov chain:**
  generation of reachability set and reachability graph to obtain the Markov chain
- **solution of the Markov chain**

Tool support for all steps exists. Conversion to a Markov chain and solution can be automated completely.

Model simulation vs. analytical solutions

- generalized stochastic petri nets are well suited for simulation
- transition rates are not restricted to be deterministic or exponentially distributed
- complex models are computationally expensive (simulation step width and simulation duration)
- too large simulation step width can result in meaningless results (variance of result is too big)

Open issues of probabilistic structural based models

- large gap between system and model
- model construction is time consuming, error prone and unintuitive
- small changes in the architecture result in considerable changes in the model
- model validation for ultra-high dependability

Probabilistic structural modeling and software

Probabilistic structural based models are not well suited for software. They are rather well suited to analyze hardware architectures and design alternatives.

- for software there are no well defined individual components
- complexity of software structures is very high
- for software the assumption of independent failures is too strong
  - one CPU for many processes
  - one address range for many functions
- real-time aspects are not captured
- parallelism and synchronization is not considered (except for GSPN's)

Reliability growth models

- no assumption on identifiable components and system structure
- based on the idea of an iterative improvement process:

  testing $\rightarrow$ correction $\rightarrow$ re-testing
- major goals of reliability growth models:
  - disciplined and managed process for reliability improvement
  - extrapolating the current reliability status to future results
  - assessing the magnitude of the test, correction and re-test effort
- allows modeling of wearout *and* design faults
- can be used for hardware and software as well

Software reliability growth models

- typically continuous time reliability growth
  - the software is tested
  - the times between successive failures are recorded
  - failures are fixed
- observed execution time data t_1, t_2, t_3, ... t_{i-1} are realizations of the random variables T_1, T_2, T_3, ... T_{i-1}
- based on these data the unobserved T_i, T_{i+1}, ... should be predicted (e.g. T_i = *MTTF*)

But:

- accuracy of models is very variable
- no single model can be trusted to behave well in all contexts

The prediction system

Software reliability growth models are prediction systems which are comprised of:

- The **probabilistic model**
  which specifies the distribution of any subset T_j's conditional on a unknown parameter α.
- A **statistical inference procedure**
  for α involving use of available data (realizations of T_j's)
- A **prediction procedure**
  combining the above two points to allow to make probability statements about future T_j's

The basic idea behind the *Musa* model

- The software starts with N_0 errors, n errors are removed during debugging

$$N = N_0 - n$$

- the failure rate is defined by (failure rate = failure correction rate)

$$f(t) = iKN \quad f(t) = \frac{dn}{dt}$$

i ... average instruction execution rate
K ... error exposure ratio

- the number of errors after time t is given by the differential equation

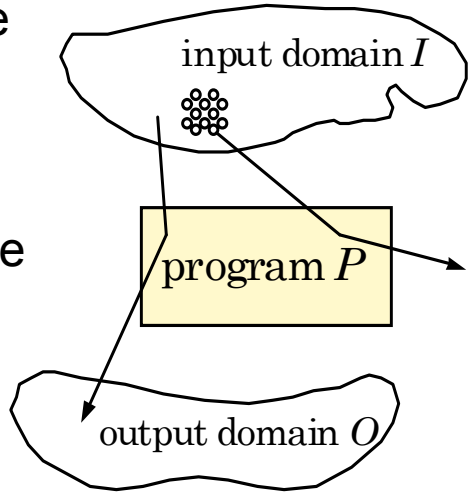
$$\frac{dn}{dt} + iKn = iKN_0$$

$$n(t) = N_0\left(1 - e^{-iKt}\right)$$

$$MTTF = \frac{1}{f(t)} = \frac{1}{iKN_0} e^{iKt}$$

Problems with the *Musa* model

- does not consider “error size”
- **Def.:** The size of an error is the probability that an element selected from *I* results in a failure
- error size usually decreases over time (diminishing returns of heroic debugging)
- assumption of independent inputs is too restrictive (program input is also determined by history)
- assumption of identical failure rates for errors
- assumption that no new errors are introduced (invalid for iterative software development process)
- accuracy is very variable



Error seeding

- an experimental approach to evaluate the software development processes and testing techniques
- the program *P* is seeded with *m* errors (one at a time), and for each error all the test cases are run until the error is detected or the set of test cases is exhausted
- evaluation of the correctness probability:

$$R = \frac{m_{detected}}{m}$$

- evaluation of testing efficiency
- reliability of test cases

Comparison of probabilistic modeling techniques

Method	Advantages	Restrictions and deficiencies
simple block diagrams	simple and easy to understand model, easy to calculate for constant failure rates	restricted to series and parallel connection, assumption of independent failures, maintenance can-not be modelled, only for active redundant systems, not well suited for software
arbitrary block diagrams	can be used to model arbitrary structures	same restrictions as with simple block diagrams, except series and parallel connection, not well suited for software
markov chains	can model arbitrary structures, no restriction to independent failures, complex dependencies can be expressed, modeling of coverage and maintenance, good tool support	compared to GSPN higher model complexity, restriction to constant failure rates, not well suited for software

Comparison of probabilistic modeling techniques (cont.)

Method	Advantages	Restrictions and deficiencies
generalized stochastic petri nets	much richer mechanisms for modeling, allows combination of discrete and stochastic behavior, good tool support, can be used to model algorithmic issues of software	it is difficult to verify that the model agrees with reality (as for any complex model)
reliability growth models	suited for prediction of software reliability, does not make assumptions on the system structure, based on relatively easy obtainable experimental data	accuracy of models is very variable, no general applicable model, user must analyze different models to select suitable one
error seeding	very easy procedure, takes few assumptions on the system	computational complexity (seeded errors by number of test cases), error size needs to be controlled

Limits of validation for ultra-high dependability

- 10^{-9} catastrophic failure conditions per hour for civil transport airplanes
- experimental system evaluation is impossible for critical applications
- modeling is therefore the only possibility to validate ultra-high dependability
- **Limits for reliability growth models:**

  If we want to have an assurance of high dependability, using information obtained from the failure process, then we need to observe the system for a very long time.

- **Limits of testing:**

  If we see a period of 10^9 hours failure free operation a MTTF of 10^9 hours can be expected without bringing any apriori believe to the problem.

  If a MTTF of 10^6 is required and only 10^3 hours of test are carried out, Bayesian analysis shows that essentially we need to *start* with a 50:50 believe that the system will attain a MTTF of 10^6.

Limits of modeling for ultra-high dependability (cont.)

- **Limits of other sources of evidence:**

  Step-wise evolution, simple design, over-engineering can be used only to a limited extent to obtain confidence because there is no continuos system model and there are no identifiable stress factors.

- **Limits of past experience:**

  For software there is no clear understanding of how perceived differences in the design or design methodology affect dependability.

- **Limits of structural modelling:**

  There are obvious limitations with respect to design faults, and software in particular since the assumption of failure independence does not hold.

Limits of modeling for ultra-high dependability (cont.)

- **Limits of formal methods and proofs:**

  "We believe that proofs may eventually give 'practically complete' assurance about software developed for small but well-understood application problems, but the set of these problems is now empty and there is no way of foreseeing whether it will grow to be of some significance."

  (*Littlewood and Strigini, 1993*)

Conclusion

- modeling is used for the design and validation phase
- deterministic and probabilistic modeling
- probabilistic structural based modeling
  - simple block diagrams
  - arbitrary block diagrams
  - Markov models
  - Generalized stochastic Petri Nets (GSPN)
- for evaluation of design alternatives
  - reliability and safety need to be considered individually
  - the interdependence of assumption coverage and system complexity needs special consideration

Conclusion (cont.)

- single parametric measures
  (mean time to failure, mean time to repair, mission reliability, availability)
- limits of modeling for ultra-high dependability
  - currently there is no methodology available to gain confidence that complex systems guarantee ultra-high dependability
  - it is impossible to collect *enough* experience with one system
  - it is impossible to extrapolate from known systems and known methodology