Fault Tolerance

- Real-time computing systems must be fault-tolerant: they must be able to continue operating despite the failure of a limited subset of their hardware or software.
- They must also allow graceful degradation: as the size of the faulty set increases, the system must not suddenly collapse but continue executing part of its workload.

- Faults $\rightarrow$ errors $\rightarrow$ failures
  - A fault is a physical defect, imperfection or flaw that occurs within some hardware or software component. A fault can be caused by specification mistakes, implementation mistakes, component defects or external disturbance (environmental effects).
  - An error is the manifestation of a fault.
  - If the error results in the system performing its function(s) incorrectly, then a system failure occurs.

Dealing with Faults

- Fault avoidance aims at preventing the occurrence of faults at the first place: design reviews, component screening, testing.
- Fault Tolerance is the ability of a system to continue to perform its tasks after the occurrence of faults
  - Fault Masking: preventing faults from introducing errors
  - Reconfiguration: Fault detection, location, containment and recovery

- forward-error recovery: the error is masked without any computations having to be re-done.
- backward-error recovery: periodically take checkpoints to save a correct computation state. When error is detected, roll back to a previous checkpoint, restore the correct state and resume execution.
Reliability and availability

- The reliability at time t, \( R(t) \), is the conditional probability that the system performs correctly during the period \([0,t]\), given that the system was performing correctly at time 0.

- The unreliability, \( F(t) \), is equal to \( 1 - R(t) \). Often referred to as the probability of failure.

- The availability at time t, \( A(t) \) is the probability that a system is operating correctly and is available to perform its functions at time t. Unlike reliability, the availability is defined at an instant of time.

- The system may incur failures but can be repaired promptly, leading to high availability.
- A system may have very low reliability, but very high availability!

Types of faults

- A permanent fault remains in existence indefinitely if no corrective action is taken.
- A transient fault disappears within a short period of time.
- An intermittent fault may appear and disappear repeatedly.

Types of Redundancy

- Hardware Redundancy: Based on physical replication of hardware.
- Software Redundancy: The system is provided with different software versions of tasks, preferably written independently by different teams.
- Time Redundancy: Based on multiple executions on the same hardware in different times.
- Information Redundancy: Based on coding data in such a way that a certain number of bit errors can be detected and/or corrected.
Redundant systems

- **Sparing**: Can have spares (hot or cold spares) and use a spare after a permanent fault is detected in the primary hardware.

- **Duplex systems**: can detect a fault by executing twice (on separate hardware on sequentially on the same hardware) and compare the results.

- **Triple modular redundancy (TMR)**: can mask an error by executing three times and taking a majority vote (may use more than one voter).

- **N modular redundancy (NMR)**: can mask an error by executing $N$ times and taking a majority vote. How many faults can be tolerated?

Fault-tolerant software

- **Consistency checks**: a software acceptance test to detect wrong results.

- **N-version programming**: Prepare $N$ different versions and run them (in parallel or sequentially). The voting at the end will select the output of the majority.

- **Sources of common-mode failures**:
  - Ambiguities in the specification
  - Choice of the programming language
  - Choice of numerical algorithms
  - Common background of the software developers

- **Recovery block approach**:
  - Each job/task has a primary version and one or more alternatives.
  - When primary version is completed, an acceptance test is performed.
  - If the acceptance test fails, an alternative version can be invoked.
Mean time to failure (FTTF)

- Let \( R(t) \) be the reliability of a system and \( F(t) = 1 - R(t) \).

- \( F(t) \) is the probability that the system is not functioning correctly at time \( t \). Hence, \( \frac{dF(t)}{dt} = -\frac{dR(t)}{dt} \) is the probability that the system fails exactly at time \( t \) (failure density function).

- The average time to failure is

\[
MTTF = \int_0^\infty \frac{dF(t)}{dt} dt = -\int_0^\infty \frac{dR(t)}{dt} dt = \left[ -tR(t) \right]_0^\infty + \int_0^\infty R(t) dt = \int_0^\infty R(t) dt
\]

- Example: if \( R(t) = e^{-\lambda t} \), then
  - \( MTTF = \frac{1}{\lambda} \).
  - \( \lambda \) is the failure rate.

Combinatorial calculation of the reliability

- For \( n \) units connected in series, the system is functioning if all the units are functioning, thus the reliability of the system is

\[
R(t) = R_1(t) R_2(t) \ldots R_n(t)
\]

- For \( n \) units connected in parallel, the system is functioning if at least one unit is functioning, thus

\[
1 - R(t) = (1 - R_1(t)) (1 - R_2(t)) \ldots (1 - R_n(t)),
\]

and the system reliability is

\[
R(t) = 1 - (1 - R_1(t)) (1 - R_2(t)) \ldots (1 - R_n(t))
\]

- Example: Reliability of a TMR system is

\[
(3R_{\text{unit}}^2 (1 - R_{\text{unit}}) + R_{\text{unit}}^3)_{\text{TMR}}
\]
Markov processes

- Is a process that can be represented by states and probabilistic state transitions, such that the probability of moving from a state $s_i$ to another state $s_j$, does not depend on the way the process reached state $s_i$.

- Example: a TMR system with unit MTTF $= 1/\lambda$, and mean time to repair equal to $= 1/\mu$.

\[ \begin{align*}
(1-\lambda)^3 & \quad \mu \\
3(1-\lambda)^2 \lambda & \quad (1-\lambda)^2 - \mu \\
2(1-\lambda) \lambda + \lambda^2 & \quad 3(1-\lambda)\lambda^2 + \lambda^3
\end{align*} \]

- Note that the failure state is an absorbing state.
- For discrete time processes, one transition occurs in every time unit.

Discrete Markov processes

- A Markov process with $n$ states can be represented by an $n \times n$ probability matrix $A = [a_{ij}]$, where $a_{ij}$ is the probability of moving from state $i$ to state $j$ in one time unit.

- The sum of the elements in each row of $A$ is equal to 1.

- If $p(t) = [p_i(t)]$ is a vector such that $p_i(t)$ is the probabilities of being in state $i$ at time $t$, then, $p(t+k) = B^t p(t)$, where $B$ is the transpose of $A$.

- Can use the first step analysis to find
  - the average number of transitions before absorption, and
  - the average time of being in a certain state.
Average # of transitions before absorption

- Consider an \( n \) state Markov process in which state \( n \) is an absorption state, and let \( v_i \) be the average number of steps to absorption if we start at state \( i \).
- Hence, for every \( i=1, \ldots, n-1 \) we have
  \[ v_i = a_{i,1} (1+v_1) + \ldots + a_{i,n-1} (1+v_{n-1}) + a_{i,n} \]
- Solve the above \( n-1 \) equations and find the values of \( v_1, \ldots, v_{n-1} \)
- Given an initial probability distribution \( p(0) \), the average time to absorption is
  \[ p_1 v_1 + \ldots + p_{n-1} v_{n-1} + 0 \cdot p_n \]
- Example: The TMR system without repair (\( \mu = 0 \)) and ignoring \( \lambda^2 \) terms
  \[
  v_1 = (1-3\lambda)(v_1+1) + 3\lambda(v_2+1) \\
  v_2 = (1-2\lambda)(v_2+1) + 2\lambda \\
  \]
  Which gives \( v_2 = 1/2\lambda \) and \( v_1 = 5/6\lambda \).

Another example of the first step analysis

- Consider an \( n+2 \) state Markov process in which states \( n+1 \) and \( n+2 \) are absorption states, we want to find out what is the probability that the process will end up in state \( n+2 \) (as opposed to \( n+1 \)).
- Let \( u_i \) be the probability that the process will eventually end up in state \( n+2 \) assuming that the process starts at state \( i \).
- Hence, for every \( i=1, \ldots, n \) we have
  \[ u_i = a_{i,1} u_1 + \ldots + a_{i,n} u_n + a_{i,n+2} \]
- Solve the above \( n \) equations and find the values of \( u_1, \ldots, u_n \)
- Given an initial probability distribution \( p(0) \), the probability of absorption to state \( n+2 \) is
  \[ p_1 u_1 + \ldots + p_n u_n \]
- Example: The TMR system with a voter and voter failure rate \( \lambda_v \)
  \[
  u_1 = (1 - 3\lambda_v - \lambda_v) u_1 + 3\lambda_v u_2 + \lambda_v \\
  u_2 = (1 - 2\lambda_v - \lambda_v) u_2 + \lambda_v \\
  \]
  Which gives \( u_1 = \lambda_v (5\lambda_v + \lambda_v) / (2\lambda_v + \lambda_v) (3\lambda_v + \lambda_v) \).