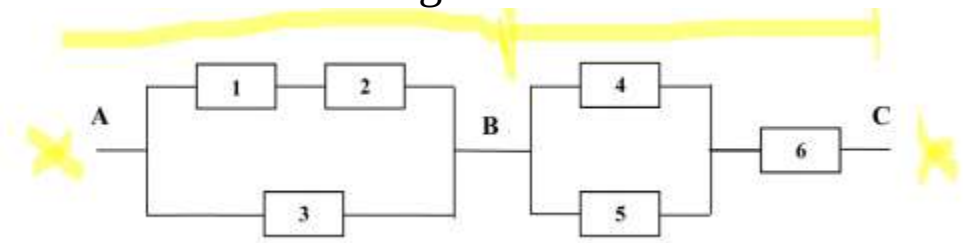
Lecture 12

Chapter # 4 (Cont'd)
RELIABILITY AND AVAILABILITY MODELING

Task

Consider the following network of six routers.



Each router can fail with probability p. Router failures are mutually independent. Showing all steps, derive expressions for the probability that the node:

a) A can successfully send packets to node B
b) B can successfully send packets to node C
c) A can successfully send packets to node C

SYSTEM AVAILABILITY

Probability that the system will be up and running and able to deliver useful services to users at any given time.

So availability & reliability are oviously link with each other as system failure may crash the system.

However availaibility does not just depend on the number of system crashes but also on the time needed to repair the system that has caused the failure. Therefore,

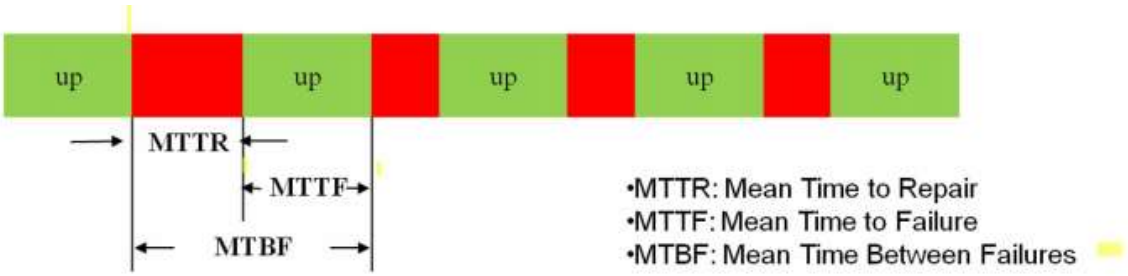
Lets suppose there's a system A that fails once a year & system B fails once a month then system A is clearly more reliable than system B.

However assume that system A takes 3 days to restart after a

failure whereas system B takes only 10 mins to restart. So here the availability of system B is much better than that of system A. Here the down time of system B is 2 hr & the down time of system A is 72 hrs.
So ovio in terms of availability we will prefer system B.

Example:
• For the insulin pump system, the most important dependability properties are:
  • availability (it must work when required),
  • reliability (it must deliver the correct dose of insulin), and
  • safety (it must never deliver a dangerous dose of insulin).
  • Security is not an issue as the pump will not maintain confidential information.

we have discussed that the availability is defined as the probability that services randered by a system will be available to its users at a particular time instant or during a specific time interval.
While reliability assess the continuity of operation. Availability is a measure of readyness* of services when required by the users.
For example; A system having high reliability may have low reliability for example the call center.
But a system having high availability may have low reliability. Here example could be a mobile phone network while most mobile networks has very high up times.
up time is being refferd to availability.
Here the drop calls 10 to be relaitively common.
So the dropped calls are being referred to Reliability.



In this particular graph there are three imp parameters of availability i.e. MTTR, MTTF & MTBF.
Initially the system is up & available
so whenever a fault is diagnosed or the system is crashed the system will not be available for its user.
The time when the system is not available to its user is called the MTTR.
After the system is being repaired & it is again operational so that starting point till the next failure is called the MTTF.
& the sum of these two time is termed as MTBF.

Availability (A) during an interval is calculated as the fraction of time a system is up.
Therefore,

$$A = \frac{Up\ Time}{Up\ Time + Down\ Time}$$

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

We may define unavailability (U) as:

$$U = 1 - A$$

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

Example Problem

A computer has an MTTF = 34 hr and MTTR = 2.5 hr.
a) Determine the availability?
b) If the MTTR is reduced to 1.5 hr, what MTTF can be tolerated without decreasing the availability of computer?
Solve!!!

Answers:
a) 0.9315
b) 20.4 hrs

FAULT, ERROR AND FAILURE

Systems Reliability & Availability problems are mostly caused by system failures.
Many failures are a consequence of ironeous system behaviour that derives from the faults in the system.
When discussing Reliability it is helpful to distinguish between these terms,

• ***Fault***: an incorrect step, process, or data definition which causes the program to perform in an undesirable manner. e.g. absence of a data validation condition.

• ***Error***: A system state that can lead to undesirable system behavior. e.g. assignment of zero value to a variable that has to divide some other variable in the next step.

• ***Failure***: a situation in which the system does not deliver a service according to its specification.

System faults do not always result in system errors and system

errors do not necessarily result in system failures.
because at the same point they are handeled & omitted.

The reasons for this are as follows:
1) Not all code in a program is executed.
Sometime it also happen that the code that includes a fault (for example the failure to initialize a certain variable) may not may never be executed.
2) Errors are transient.
so a state variable may have an incorrect variable may be caused by the execution of faulty code however before this is access & causes a system failure some other systems input may be processed that resets this state to a valid value.
3) The system may include fault detection and protection mechanisms.
These mechanisms ensure that the ironeous behaviour is discovered & corrected before the systems services are effected.

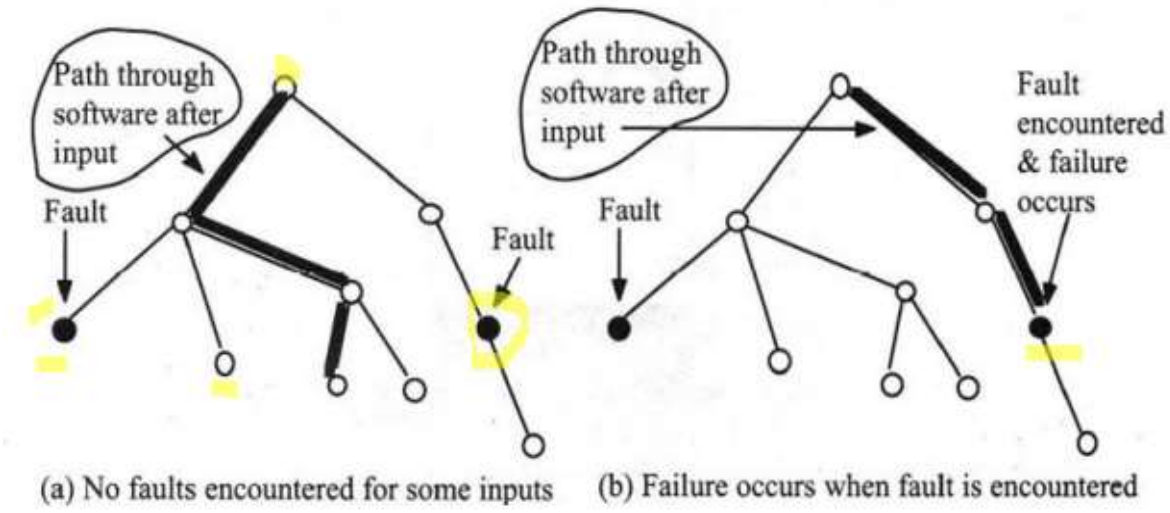


Fig 1: Execution of software modules

In the fisrt fig a, there are various paths of the execution & lets suppose we are starting with this particular point so we reach the

end using various paths. There are total 5 ending points in fig a. But the faults are in 2 particular locations so it may happen that for some sort of input no faults are ever encountered. By following this path the execution of the program is safely performed without any system failure.
But in fig b, I may encounter failure when some fault is occur. Lets suppose If i execute my software by this particular path i.e. after applying certain input it takes then definitely it will encounter the fault & may result in system failure.

Software Reliability vs Hardware Reliability

1) Software has no aging property (no parts to wear out).
All the software failures can be traced to design or implementation problem. So in this sense the factor of time is not explicitly involved.

2) There are different sources of improving reliability.
Improved software reliability results from an intensive program of fault dicsovery & removal. However the increased hardware reliability results from the use of better parts improved design practices as well as execisng fault tolerance approaches with the builtin redundancy.

3) Copies of software systems are identical.
There's a standard redundancy method that is exploited in hardware design is not work/worth* in software systems.

RELIABILITY METRICS

1) Probability of Failure on Demand (POFOD):

• The likelihood that the system will fail when a service request is made.
• Most appropriate for systems where services demanded at relatively long time intervals and there are serious consequences if service is not delivered.
• It might be used to specify protection systems such as the reliability of a pressure relief system in a chemical plant or an emergency shutdown system in a power plant.
• A POFOD of 0.001 means that one out of a thousand service requests may result in failure.

2) Rate of Occurrence of Failures (ROCOF)

• This metric should be used where regular demands are made on system services and where it is important that these services are correctly delivered.
• A ROCOF of 2/100 means that two failures are likely to occur in each 100 operational time units.
• It might be used in the specification of a bank teller system that processes customer transactions or in a hotel reservation system.
• Sometimes called the failure intensity.

3) Mean time to failure (MTTF)

It is one of the metrices to represent the Reliability & Availability of the system. & its significance is that it is used in the calculation of uptime & the availability of the system.
It represents the
• Average time between observed system failures.
• Should be used in systems with long transactions.
• MTTF should be longer than the average transaction length.

• Examples of systems using this metric are word processor systems and CAD systems.
• An MTTF of 500 means one failure can be expected every 500 time units.

4) Availability (AVAIL)

• This metric should be used in non-stop systems where users expect the system to deliver a continuous service.
Availability is there to represent the uptime of any system or it's important for the system to be available & provide services to its users when it's required.
• Examples of such systems are telephone switching systems and railway signaling systems.
• Availability of 0.998 means that the system is likely to be available for 998 of every 1,000 time units. It is defined as:

Availability = [MTTF/(MTTF + MTTR)] x 100%

RELIABILITY VALIDATION

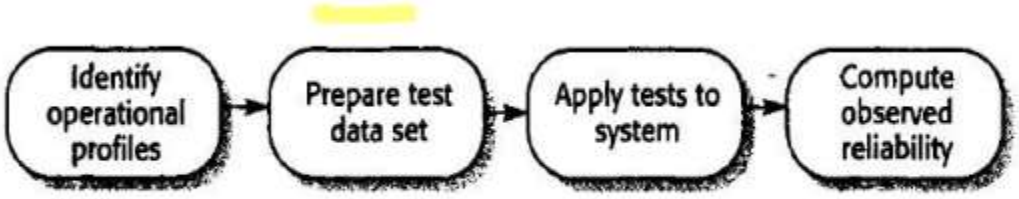


Fig 2: The reliability measurement process

Fig 2: The reliability measurement process

This process involves four distinct stages:
1. (Identification of operational profile); Studying existing systems of same type to establish an operational profile.
So, an operational profile actually identifies classes of system

inputs & the probability that these inputs will occur in normal use*
easier to specify for a new software system that replaces an existing manual or automated system. For example An operational profile can be specified for telecommunication switching system because such companies knows the call patterns to handle.

2. Construct a set of test data that reflect the operational profile.
So normally for prepration of test dataset, we can use the test data generator as well.

3. Apply Test the system using these data and then count the number and type of failures that occur.
Here the times of these failures are also logged.

4. After observing a statistically significant number of failures, compute the appropriate reliability metric value. This approach sometimes called *statistical testing*.
Here in final step we need to compute the observed Reliability. So its aim is to assess the system Reliability & differentiate with detect testing where aim is to discover the system faults.

These are the four steps tha we can use in the calculation of software reliability we can identify the operational profiles by lets suppose comparing various similar systems, we can prepare the test dataset we can test that system & after the application of test data we can actually observe the number of failures in our system. & then depending upon number of failures we can compute the appropriate reliability metric value.

Task

Three identical computers are networked together in parallel configuration. Their failure rate is given by $\lambda = 0.2$ failures/year.

Calculate:

i) MTTF of each computer

Recall the discussion of exponential distribution.
Since we have already identified the fact that the reliability of system is being modeled using exponential distribution.
& whenever we talk about the average values or mean values we think of the expected value or the relation which is given by $1/\lambda$.
keep in mind here we are also talking about the MTTF & we are also provided with the failure rate.
For the calculation of the mean value or the average value we can again use the relation $1/\lambda$.

ii) Reliability at the end of five years

Given; Computer are used in parallel coniguration.
& since the reliability value of individual components are not given so we have to replace the value of reliability with its formula. value of t is 5 years.

Answers:

i) 5 years
ii) 0.747