TUTORIAL - 3

Date 19/01
Page 06

U19CS012

1> Explain the following terms:

(a) Availability : system is ready to be used immediately

① It refers to the probability that the system is operating correctly at any given moment and is available to perform its functions on behalf of its users.

② Highly Available System ⇒ one that will most likely be working at given time instant

(b) Reliability : system can run continously without failure.

① Highly Reliable system ⇒ one that will most likely continue to work without interruption during a relatively long period of time.

(c) Safety : When a system (temporarily) fails to operate correctly, nothing catastrophic / dangerous happens.

Eg: many Process Control systems → controlling Nuclear Power Plants / Sending people into space, are required to provide high degree of safety. If system fails temporarily, the effects could be disastrous.

(d) Maintainability : how easily a failed system can be repaired

Highly maintainable system may also show high degree of availability, esp if failures can be detected and repaired automatically.

U19CS012

2> Explain the concepts of failure, error and fault with example.

2.>

Failure ① A system is said to "fail" when it cannot meet its promises.

② In particular, if Distributed system is designed to provide its user with a number of services, the system has failed, when 1 or more of services can be (completely) provided.

Error ① An Error is a part of system state that may lead to failure.

Eg: When transmitting packets across network, it is expected that some packets have been damaged when they arrive at the receiver. ↳(bit value change) [0→1]

Fault ① The cause of Error is called Fault.

② Clearly, finding out what caused an error is important, for eg: a wrong / bad transmission medium may easily cause packets to be damaged. (In this case, it is relatively easy to remove the fault.).

3> Explain different types of faults.

3.> There are three main types of 'fault':

① Transient Fault - appear once, then disappears.

Eg: If the operation, is repeated, the fault goes away. A bird flying through the beam of microwave transmitter may cause lost bits on some network.

② Intermittent Fault - occurs, vanishes, reappears; but follows no real pattern (worst kind)

Eg: A loose contact on a connector will often cause an intermittent fault.
Intermittent faults cause a great deal of aggravation because they are difficult to diagnose.

③ Permanent Fault - once it occurs, only the replacement/repair of faulty component will allow DS to function normally.

Eg: Burnt-out chips, Software bugs & Disk head crashes.

Identify faults in the following:

a.) a program may (Cause 1) fail to initialise a variable which is required to be initially zero; if the program is run in circumstances such that memory is almost always clear before starts, it will malfunction on rare occasions that (Cause 2) the memory where the variable is stored happens to be non-zero beforehand.
- ~~Transient Fault~~ ✓① occurs, → vanishes → reappears
∴ Intermittent fault ✓② follows no pattern

b.) A bird flying through the beam of microwave transmitter may cause lost bits on some network.
- Transient Fault

c.) Burnt-out chips - Permanent Fault

4) Explain Failure masking by redundancy.

4) Strategy: Hide the occurance of failure from other processes using Redundancy.

Failure Masking By Redundancy

Information Redundancy

- add extra bits for error detection/recovery

Eg: Hamming codes

Time Redundancy

- Perform operation and if needs be, perform it again. Think about how transaction work (BEGIN/END/COMMIT/ABORT)

Physical Redundancy

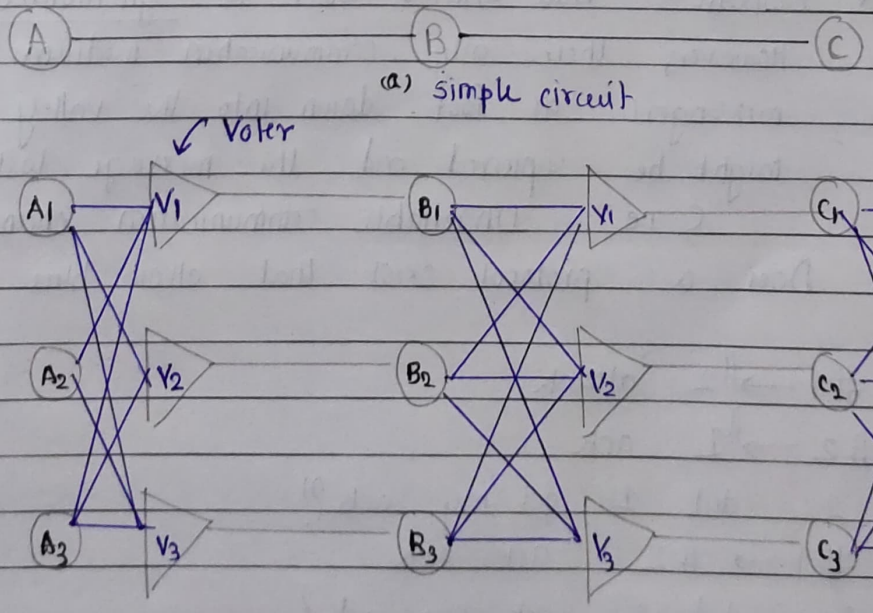
- add extra (duplicate) hardware and/or software to the system.

Physical redundancy (hardware).

Consider, for eg

signals pass through devices A, B and C in sequence.

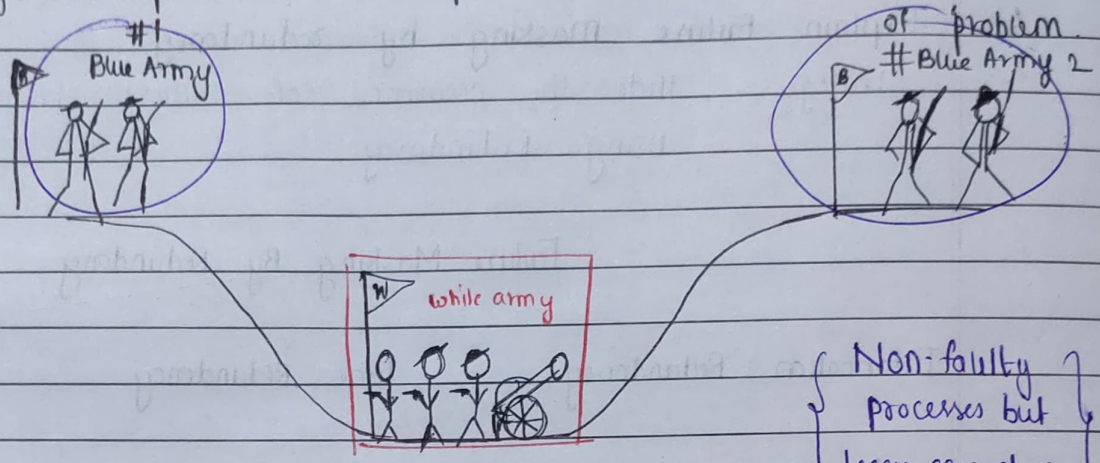
If one of them is faulty, the final result will probably be incorrect.



(a) simple circuit

(b) Triple Modular Redundancy) [TMR]

5.> Identify the problem in the picture and Explain your understanding of problem.



{ Non-faulty processes but lossy connection }

Problem in above Picture - Two-army Problem

① Imagine that a white army is encamped in a valley as shown above.
② on both the surrounding hill sides are blue armies.
③ White army is larger than either of blue armies alone; but together blue armies are larger than white army ((2+2) > 3)
④ If either blue army attack by itself, it will be defeated, but if the two blue armies attack simultaneously, they will be victorious.

* Problem - Blue armies want to synchronize their attack. However, their only communication medium is to send messengers on foot down into the valley, where they might be captured and the message lost.
(ie Unreliable communication channel)

Does a protocol exist that allow blue armies to win?!

#1 → #2 attack
#2 → #1 ack
2: did 1 get my ack?!
#1 → #2 ack ack
1: did 2 get my acks?

U19CS012

6.> Understand the steps in BYZANTINE GENERAL PROBLEM. Illustrate the working of the algorithm for the case $n = 5$, $m = 1$.

(n ↓ no. of process, m → faulty process.)

6.> Algorithm for Byzantine General Problem [Recursive by Lamport]

Step 1 > Every general sends a (reliable) message to every other general announcing his troop strength.

Step 2 > Results of announcements of step 1 are collected together in the form of the vectors.

Step 3 > Every general passing vectors to other generals

Step 4 > each general examines the i^{th} element of each of the newly received vectors. If no majority, corresponding element will be considered to be UNKNOWN

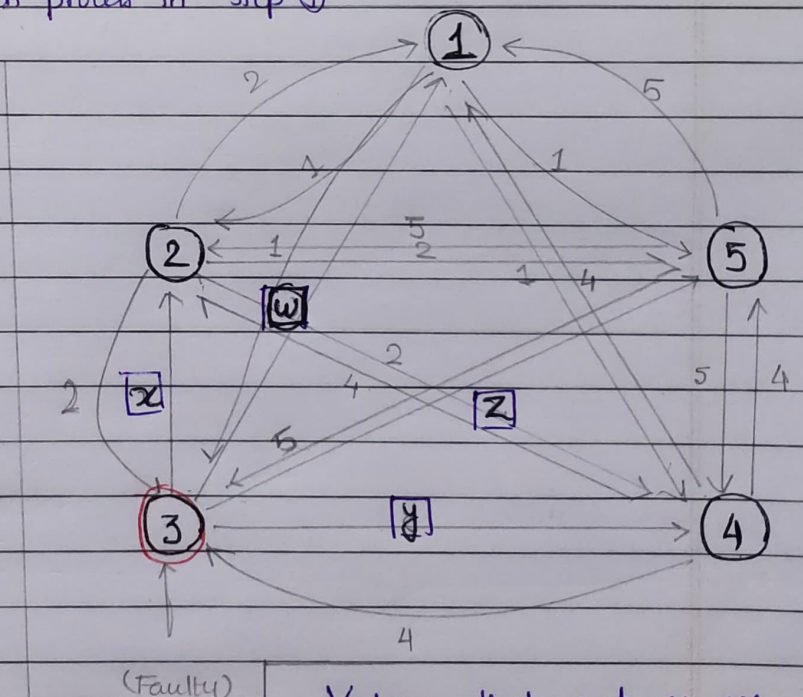
Vectors received by each process in step ①

1 got
2 (1, 2, x, 4, 5)
3 (a, b, c, d, e)
4 (1, 2, c_1, 4, 5)
5 (1, 2, c_2, 4, 5)

2 got
1 (1, 2, a, 4, 5)
3 (b_1, b_2, b_3, b_4, b_5)
4 (1, 2, c_4, 4, 5)
5 (1, 2, c_2, 4, 5)

3 got
1 (1, 2, a, 4, 5)
2 (1, 2, b, 4, 5)
4 (1, 2, c, 4, 5)
5 (1, 2, d, 4, 5)

4 got
1 (1, 2, a, 4, 5)
2 (1, 2, b, 4, 5)
3 (d_1, d_2, d_3, d_4, d_5)
5 (1, 2, c_3, 4, 5)



5 got
1 (1, 2, a, 4, 5)
2 (1, 2, b_1, 4, 5)
3 (f_1, f_2, f_3, f_4, f_5)
4 (1, 2, c_2, 4, 5)

Vectors that each process assembles based on previous step

1 got (1, 2, x, 4, 5)
2 got (1, 2, y, 4, 5)
4 got (1, 2, z, 4, 5)
5 got (1, 2, w, 4, 5)