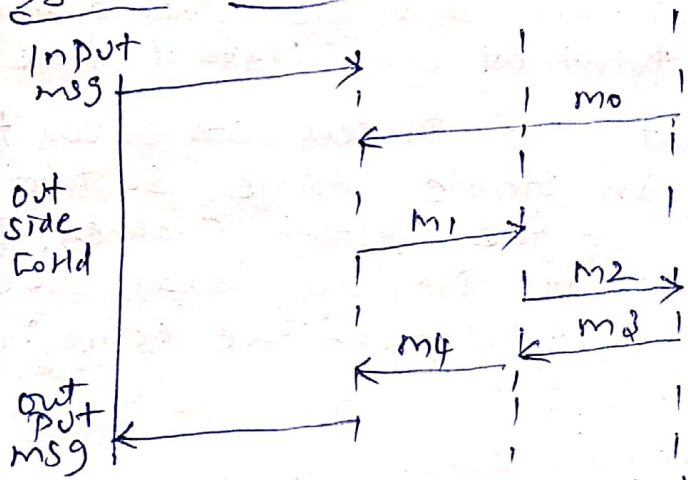
Unit IV

1) Explain about the following (i) System model (ii) A local check point (iii) Consistent system states?

System model:

A set of fixed processes ie $P_1, P_2, \cdots P_n$ that communicate via messages is called a distributed system model. These processes execute a distributed application and communicate with the outside world by sending input messages and receiving output messages.

DS with 3 processes:



In distributed system models, the assumptions like. reliablity of inter processes communications are made by roll back recovery Protocols.

Few protocols assume that the communication sub system is delivered in first out order where as other protocols assume that the communication subsystem can reorder, loose or duplicate the messages.

If any two assumptions, is selected, then it effects the complexity of failure recovery and checkpointing.

If the system recovers properly with consistent internal state with noticeable system behaviour before its failure then this condition is said to be genetic correctness condition for roll back recovery. The protocols of this mechanism maintains the information about (i) internals external interactions.

A Local Check Point:

A local check point is saved state in which all the processes in the distributed system save their local states at specific instance time.

It is defined as the process snapshot state.

The scenario of recording the process state is called as local check pointing.

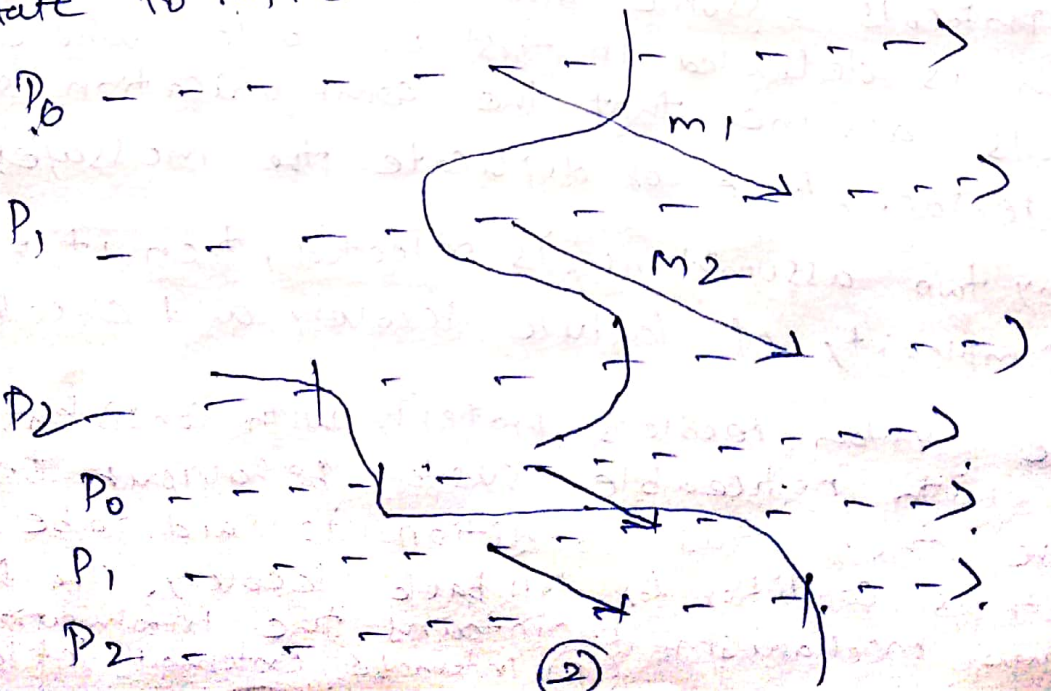
The contents of the check point are based on check pointing mechanism and the application context used.

The process may maintain a single check point or many local check point at any instant time

It is assumed that the process stores all the local check points in stable storage so that they can be recovered when the system crashes. The process may roll back to its previous local check points and hence can restore from the corresponding state.

Consistent State:

A consistent system state is defined as a process state that reflects a message acknowledgement. This effect must cause the corresponding sender state to reflect the message forwarding.



Check Point Based Recovery:

A method which checks the position of each process and communication channel frequently so as to restore them at the time of failure is known as check point based recovery.

The protocols of check point based recovery are less restrictive and easy to implement, because they does not relay on PWD assumptions.

A check point based roll back recovery cannot ensure that after roll back there can be regeneration of the failure execution.

Therefore it is not used for applications that need to interact frequently with outside world.

1. Un Coordinated Check Pointing:

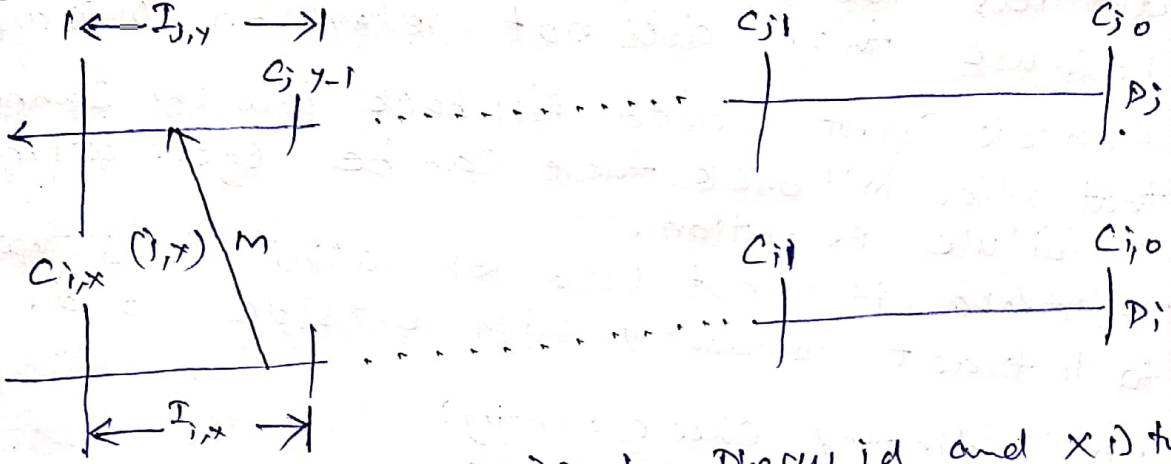
(i) It eliminates synchronization overhead.
(ii) It allows the process to select their check points according to their convenience.
(iii) It provides lower runtime overhead at the time of normal execution.

Uncoordinated check pointing also has the following disadvantages:

(i) There is a possibility of losing huge amount of useful data at the time of data recovery.
(ii) Recovery of data from a failure execution is slow.
(iii) There is no coordination between processes regarding the acceptance of check points. Therefore check points taken by the process can be set as useless check points
(iv) It requires global coordination for computing a recovery line. This coordination has a negative impact over the free selection of the check points.

Every Process at the time of system failure must determine the Global Check Point. While determining the global Check Points all the Processes are required to record the dependencies of their Check points that were taken at the time of failure free operations

An uncoordinated checkpointing make use of direct dependency tracking



P_i & P_j are the process, i is the Process id and x is the checkpoint index, $I_{i,x}$ / $I_{j,y}$ is the checkpoint interval between $C_{i,x}$ and $C_{i,x-1}$ / $C_{i,x}$ and $C_{i,y-1}$

* If a process encounters failures it intiates a rollback by transmitting a dependency request message so that the initiator can easily gather dependency information. maintained by every other process.

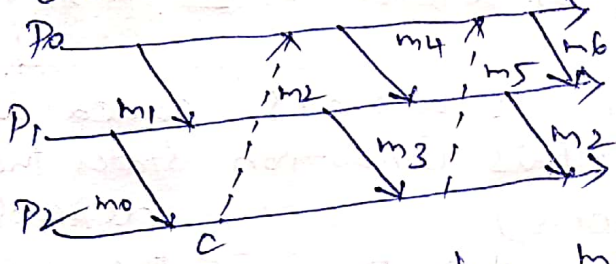
* Any process that receives this request message halts its execution and sends a reply message which contains the dependency information stored on the Stable Storage along with dependency information and again roll back by recovery line request message.

* The process whose current state matches with the request msg resumes the execution and the process whose current state doesnot matches with the request message simply roll back to the previous checkpoint defined by the recovery line.

3) Explain about the deterministic and non deterministic events in the log based rollback recovery.?

* The log based roll back recovery makes excessive use of the fact that a process is executed initially by executing the non deterministic events followed by a sequence of deterministic event time interval.

Here the non deterministic event may be a receipt of message from external (or) internal events of process.



In the above 4 intervals in the execution of a process P_0. The first interval is for creating a process, and other 3 intervals are used to start the receipts of the messages m_0, m_7, m_3.

The process P_0 and the receipt of m_0 uniquely determines the send event of m_2 and hence it can be said that m_2 is a non deterministic event

The simplicity of log based roll back recovery is that, it assumes that the identification of the non deterministic events is easy and their respective determinants can be stored into the stable storage

When a failure occurs, the processes can be recovered using the check points and logged determinants of the non deterministic intervals stored in the stable storage

The No orphan Consistency condition:

Dependency

It defines a set of processes which are influenced by non deterministic event e That is this set includes the processes whose states are depended on event 'e'.

2) Log (e)
It defines a set of processes that logged the determinants of the event 'e' in their volatile storage.

3) Stable e: A process p is said to be orphan when it is not failed by itself.
* A state of process p which is depended on the output of a non deterministic event e and its determinants are not recovered from the volatile memory

$$\forall (e) : \neg \text{stable}(e) \Rightarrow \text{Depend}(e) \subseteq \text{Loge}$$

The above condition is always called as no orphans condition. This condition states that if an ongoing or running process is dependent on non deterministic event e then either e must be logged in the stable storage or the process must have copy of determinant 'e'.
There is an alternative recovery protocol used which guarantees a system which is free from all the orphan processes. This recovery protocol is called as log based roll back recovery protocol.
There are 3 types of log based roll back recovery.
1. pessimistic logging protocol
2. optimistic logging protocol
3. Casual logging protocol.

(4) Explain about Problem definition in Consensus and agreement algorithm?

The fundamental requirement in a distributed system is the agreement between the Processes. This agreement should help in coordinating the Processes to negotiate and exchange the information among them.

Failure model: There are several Processes running in the system, out of which k Processes can be at fault. A faulty Processes is an abnormal Processes that runs in any manner permitted by the failure model. There exists several failure models as send omision and receive omision, fail-stop, Byzantine failures. Each of these models behave in a Different manner.

2. Synchronous / Asynchronous Communication:

If a failure phone process sends a message to some random process say P_1 and fails then it takes a lot of time to figure out the reason of unsuccessful transmission of message in an asynchronous system. While in the synchronous system the failure can be detected easily because the recipient assumes that the expected message might contains some default data and it moves to the next round

3) Network Connectivity: A system supports full logical connectivity. That is each Process in the system can communicate with other Process by directly passing message.

4) Sender Identification:

It should be assumed that the receiver who receives a message can identify the sender. If there are multiple messages expected from a sender in one round then a scheduling algorithm is used to schedule these messages in some default data and it sub rounds.

5) Channel Reliablity:

The channels are always reliable it is the Process that may encounter failure while executing. This is the simplest assumption in the study of agreement algorithms. Even with this simplest assumption there are few agreement Problems that are either solvable or unsolvable.

6) Authenticated Vs non authenticated messages:

In the study of agreement algorithms only unauthenticated messages are dealt. These messages are sent by unreliable sources or faulty Process, which can be tempered or forged. The authenticity of these Process can not be recognized. These messages are also called oral (or) unsigned. messages. To solve the agreement authentication problem the algorithm like digital signature can be used such that the recipient can be recognized any forgery.

7) Agree ment variable: An agreement variable can be a boolean or multi valued variable. It is not necessary for an agreement variable to be an Integer. A boolean variable is used to abstract the values of the algorithms. It has to effect on the other data type results.

(5) Discuss about the Byzantine agreement and other Problems and also Equivalance of Problems and definition?

Byzantine Agreement and other Problems:

This Problem deals with achieving the agreement with other designated Process called Source Process along with its Initial value.

Agreement: This Condition States that Each and every non faulty Processes should agree on the same value.

Validity: This Condition States that if the designated Source Process is non faulty, then the value which is agreed by all the non faulty Processes should be equal to the Initial value of the Source Process.

Termination: This Condition States that every non faulty Process must agree on only one value.

The validity Condition rules is of great Importance. It guarantees that the agreed upon value has a correlation with the source value.

2 other problems in Byzantine agreement:

a) The Interactive Consistency Problem:

This Problem is different from the Byzantine agreement Problem. Here every Process Consists of an Intial value, all the Correct Processes should agree on a group of values such that for each Process has one value.

(i) Agreement: It states that each and every non faulty Process must agree on the same set of values. A [V1, ... Vn].

Validity: It states that of Process say is non faulty with initial value Vi each Process must agree on Vi as the ith element of the array A and if another Process j is found faulty then all the non faulty can agree on any value for array A[j];

3) Termination: It states that all the non faulty Process must consequently decide on the array A.

b) Consensus Problem: This Problem also different from the Byzantine agreement Problem, Here each Process has an initial value and all the correct Processes must agree on the same value.

Agreement It states that each and every Process must agree on the same value.

Validity: It states that if all the Processes consists of same initial value, then the agreed value must be that same value only.

Termination: It states that every non faulty Process must agree on one same value.

Equivalence of the Problems and notations:

The Byzantine agreement Problem, Interactive Consistency Problem and Consensus Problem are equivalent such that the Solution of one Problem can be used as the Solution for the other 2 Problems.

This equivalent Property is used for the reducing one Problem to the other 2 Problems.

The main difference between consensus Problem and agreement problem is that in the Consensus Problem each Process consists of an initial value where as in the agreement Problem only single Process consists of an initial value.

6) Explain about the agreement in synchronous system with failures?

The agreement in synchronous system with failures comes in 2 types.

* Consensus algorithm for crash failure in synchronous system
* Consensus algorithm for Byzantine failures in synchronous system.

① Consensus Algorithm for crash failures in synchronous system:

N processes with upto f (fail stop) Process where $n > f$ for Process P_i $(1 \le i \le n)$

Algorithm:

```
local variable
integer : x ← local value
global Constants:
integer : f;
```

The Process P_i $(1 \le i \le n)$ runs the consensus algorithm upto f crash failures.

```
For round from 1 to f+1
do
if the current value of x is not broad
                              casted.
then
broadcast (x);
y_j ← The value received from the Process.
j in this round;
x ← min ∀j(x, y);
```

Output: x is a consensus value.

Each Process has x_i as an initial value. If f failures need to be tolerated then the algorithm must have f+1 rounds. The Process P_i

forwards the value of its variable x_i to all processes in every round only when the value is not forwarded earlier.

1. Agreement Condition: This condition is satisfied when there exists atleast one round with no process failures in the $f+1$ rounds. In this round, all the processes may not have failed in broadcasting their values and considers only the minimum values broadcast that are received in the round. Hence, the local values at the end are same ie x_i is same for all the non failed processes.

2) Validity Condition: This condition is satisfied when the processes do not forward any fictitious value in the failure model. If the initial value is similar for all i then only value which has been agreed on the agreement condition is forwarded by any of the process.

3. Termination condition: If the two conditions are satisfied then this condition is also satisfied.

Complexity:

The process forwards its value to another process before the failure. In the next rounds the single process having minimum value also manages to broad the value to another process before the occurance of failure.

Lower Bound on the No. of rounds:

A single processes may fail in each round and with $f+1$ rounds, there will be atleast one round with no process failure. which compute the function of the received value to achieve an agreed value.

7) Explain about Phase king algorithm:

Polynomial in synchronous system:

This algorithm operates only in f+1 phases where every phase contains two rounds. It has a special process which plays crucial asymmetrical role as a leader. Hence the algorithm is named as phase king algorithm.

Input variables:

v - boolean
f - Integer
v ← initial value.
f ← maximum number of faulty processes.
f < [n/4];

1. The following f+1 phases are executed by every process. where f < [n/4];
2. For Phase = 1 to f+1
3. do
4. Execute following actions in round 1.
5. Broad cast v to all processes;
6. await value v_j from each process P_j
7. set majority as the value between v_j that occurs > n/2 times.
8. set mult as the total number of times that majority occurs;
9. execute the following actions in round 2.
10. If i = phase
11. then
12. broad cast majority d to all processes;
13. Receive tiebreaks from Pphase.
14. If mult > n/2 tf.
15. then.

12. broadcast majority a to all Processes;
13. Receive the bleaks from P phase.
14. If mult >n/2 f
15. then
16. set V as majority.
17. else.
18. set V as tie breaker.
19. If Phase = f+1.
Do then
21. Display V as output (decision value)

output: The decision value V is the output

Round 1: In each phase of this round every Process delivers the estimation of its consensus to all processes and waits for the values to be broad casted by other Processes. The total number of votes for 0 and 1 are counted at the end of the round. If any of the value is greater than n/2, then its majority variable is set to the consensus value and multi is set to the no. of votes that are received for the majority value.

Round 2: The Phase King for Phase k has the identifier P_k for $k \in \{1 \ldots n\}$. The majority value of Phase king is broad casted is acts as tie breaker for other Processes that have multi values less than n/2 + f. If the Processes receives the breaker from the phase king, it updates the estimation of the decision variable V to the value that is forwarded by the phase king when its multi variable is less than n/2 + f.