Assignment (Lec. 11 & 12): Distributed Operating System Principles (COP5615)

Name: Vikas Chaubey**, UFID:** 3511 5826**, Email:** vikas.chaubey@ufl.edu

1) Mutual Centralized algorithm: In order to understand centralized algorithm, we need to first address the problem of mutual exclusion**.**

What is Mutual Exclusion?

Mutual exclusion is a concurrency control property and it is used to avoid race conditions in distributed systems. Preventing race condition means that a process can not interfere with another process while this process is doing some critical execution. Only one process should be allowed to perform that critical operation at one point of time. In case of distributed system solving mutual exclusion, problem is difficult because since all different processes reside on different machines hence, the shared memory or physical clocks are different, hence it is hard for one process to estimate the status of another process. Hence solving the mutual exclusion problem in distributed systems is done via messages.

How Mutual Centralized algorithm works?

In case of distributed systems one process is elected as a coordinator which coordinates all processes in the network. When one process wants to enter a critical region and do some execution then the process asks the coordinator first for permission to enter that region. If any other process is already in the critical region then the permission is not granted. When the executing process completes execution then execution permission is granted to already waiting processes. That's how synchronization between processes is maintained. This algorithm also requires that there is no condition of deadlock in the network i.e. Two or more site should not wait endlessly for messages to arrive, there should be no starvation condition i.e. every process should get an opportunity to do execution in finite time without waiting endlessly and in case of failure, the failure should be recognized in order to continue processing without any issue. This algorithm guarantees mutual exclusion by letting one process at a time to execute in critical region.

2) Distributed Algorithm: Distributed algorithm are the algorithms which is executed concurrently, in this case the different parts of the algorithm are run on different processors. Each piece of the algorithm has limited information related to or necessary to perform its own task or execution. Since memory is not shared and information is divided in such manner one part of the algorithm does not have information about other parts of the executing algorithm. The main challenge while designing the distributed algorithm is that it should be able to synchronize the different parts of the execution which are happening in different regions. Using distributed algorithms various problems could be resolved such as mutual exclusion, leader election among multiple nodes in distributed systems, consensus, distributed search etc. There are various types of distributed algorithms some important ones are consensus algorithms which try to solve a problem agreement among multiple process on a common decision. Leader election is another type of distributed algorithm in which a single process is elected as leader or organizer for managing and synchronizing various separate processes which are running on different machines.

3) Election Algorithm: Election algorithm is a type of distributed algorithm which is used in distributed systems. In distributed systems various processes run on different machines, hence in that case in order to organize the synchronized execution of processes in the system when they lie on different machines and don’t share memory or information with each other, an organizer process is used to coordinate different processes in the network. Election algorithm is used to elect a coordinator or leader in the system to which multiple processes sends messages in order to know the status of another processes. The processes ask the coordinator for permission in order to access a certain region for execution and coordinator manages different processes synchronously so that race condition could be avoided, and system works synchronously.

How Election Algorithms Work?
Election algorithm works on the basis of priority assigned to different processes running in the system. This is done by assigning a unique priority number to each process. The process with highest priority number is selected as a leader. In case of failure new process with highest priority number among active processes will be selected as new leader. There are mainly two types of election algorithm:

1) **The bully Algorithm:** This algorithm is used in a system where each process can send messages to every other process in the distributed system. In this algorithm a process sends a message to coordinator process and waits for a response for certain duration of time, if it does not receive a message then the process will send message to all high priority number processes and awaits response. If the response is not received, then the process becomes coordinator and it sends the message to all lower priority processes to declare that it is elected as a coordinator.

2) **The Ring Algorithm:** This algorithm is used in a ring system which has unidirectional links among the processes. In this system a process can message a process which is on its right. A list is used which has priority number of all active processes in the system. The processes message the further processes unidirectionally and update the active list in order to elect a leader.

4) Consistency and Replication: The main reason to choose distributed systems is the benefits like scalability, locality and availability. Since processes are distributed on multiple machines in distributed system reliability and performance of the system is a big challenge hence replication of data is used in order to increase the reliability and performance of the system. Replication is a technique which is used to cope with data corruption and replica crashes, to keep all replicas in exactly the same state and operation. It is necessary to maintain the state of the replicated data among different processes, hence consistency is required. This is a main scalability challenge as maintaining a system with high consistency is difficult in case of distributed systems. The problem of scalability can only be resolved if the consistency requirements are defined in advance.

a) **Causal consistency**: it is a type of weak consistency. It defines that the causally related operations appear in the same order on all processes. Though processes might disagree about the order of causally independent operations.

b) **Sequential Consistency:** In sequential consistency operation appear to be executed in some order and that order is followed by each process in the same system.

c) **Eventual Consistency:** This consistency is used to maintain high availability in distributed system. It ensures that if data item accessed by the processes is not updated then processes accessing the data item should return the last updated value.

References:

1) https://www.geeksforgeeks.org/mutual-exclusion-in-distributed-system/
2) https://medium.com/cosmosgaminghub/consistency
3) https://www.allthingsdistributed.com/2008/12/eventually_consistent.html
4) https://www.cs.columbia.edu/~du/ds/assets/lectures/lecture12.pdf
5) https://whatis.techtarget.com/definition/sequential-consistency
6) http://user.it.uu.se/~carle/DS2/Notes/03_ElectionAlgorithms.html