1. **Give a real world example of a problem that can occur in a distributed system which does not exhibit process synchronization.**

//TODO

1. **What is clock drift in the context of a distributed system?**

Clock drift is the difference in time values because the clocks gradually to get out of sync when a system has many computers and all clocks will run at slightly different rates.

1. **What is Universal Coordinated Time (UTC)?**

UCT is the primary time standard by used to regulate clocks and time (can be considered a true time).

1. **How might one** **use Universal Coordinated Time (UTC) to achieve process synchronization in a distributed system?**

We could use Universal Coordinated Time (UTC) to achieve process synchronization because if one machine has a UTC receiver, the goal becomes keeping all the other machines synchronized to it, and If no machine has a UTC receiver, each machine keeps

track of its own time, and the goal is to keep all the machines together as well as possible.

Use a clock synchronization algorithm whereby one machine is given a UTC receiver and

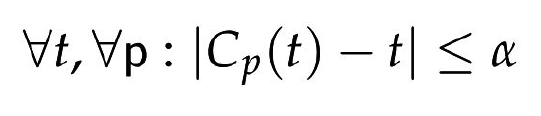
the goal is to keep all the other machines synchronized to it. This could be achieved using

the Network Time Protocol which was briefly described above.

1. **If a single machine in a distributed system has a UTC receiver, how can the problem of achieving process synchronization be solved?**

In this context one could employ the Network Time Protocol. Here the machine with the UTC receiver acts as a time server. The other machines or clients contact the time server to request the current time and update their local clocks appropriately. However this solution may not provide perfect process synchronization because when contacting the server, message delays will have outdated the reported time.

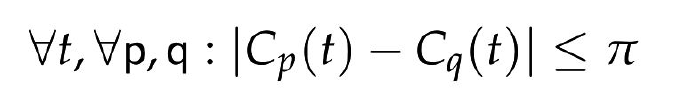
If one machine has a UTC receiver, the goal becomes keeping all the other machines synchronized to it. When the UTC time is t, denote by the value of the clock on machine p at t.A set of clocks has accuracy within a bound if the following equation is satisfied:



The above equation refers to the deviation of clock times from UTC time.

1. **If no machine in a distributed system has a UTC receiver, how can the problem of achieving process synchronization be solved?**

If no machine has a UTC receiver, each machine keeps track of its own time, and the goal is to keep all the machines together as well as possible. A set of clocks has precision within a bound if the following equation is satisfied:



The above equation refers to the deviation between clock times.

1. **What is clock drift rate in the context of a distributed system?**

Clocks are subject to clock drift where clocks on different machines will gradually show different times. The clock drift rate refers to the difference per unit of time from a perfect reference clock.

1. **How can process synchronization in a distributed system be achieved using the network time protocol?**

A common solution to process synchronization is to let clients contact a time server, which can accurately provide the current time.

1. **How can process synchronization in a distributed system be achieved using the Berkeley algorithm?**

The function of the Berkeley algorithm is let the time server actively polls every machine from time to time to ask what time it is there. And based on the answers, it computes an average time. Finally tells all machines to advance/slow their clocks to this average time.

1. **Explain how a logical clock may be used to achieve process synchronization in a distributed system.**

A logical clock models the ordering of events because we need only agree on the order in which events occur and it is not necessary all processes agree on exactly what time it is.

1. **The Lamport logical clock is based on the concept of a happens-before relation. Describe this relation.**

Happens-before relation means all processes agree that first event a occurs, then afterward, event b occurs.

Happens-before can be observed in two situations:

• If a and b are events in the same process, and a occurs before b, then a → b is true.

• If a is the event of a message being sent by one process, and b is the event of the message being received by another process, then a → b is also true.

1. **Lamport proposed an algorithm for assigning timestamps to events in a distributed system such that they satisfy the happens-before relation. Describe this algorithm.**

The algorithm is called Lamport’s algorithm:

Each process Pi maintains a local counter Ci . These counters are updated according to the following 3 steps:

1. Before executing an event Pi increments Ci: Ci ← Ci + 1.

2. When process Pi sends a message m to process Pj , it sets m’s timestamp ts(m) equal to Ci after having executed step 1.

3. Upon the receipt of message m, Pj adjusts its counter as Cj ← max {Cj, ts(m)} after which it executes step 1 and delivers the message to the application.

1. **Why may it be necessary to ensure the mutual exclusive access to resources by processes in a distributed system? Use a real world example to illustrate your answer.**

Because In many cases different processes will want to simultaneously access the same resource. And concurrent accesses may corrupt the resource or make it inconsistent.

Therefore, we need to grant mutual exclusive access by processes.

An example://TODO

1. **Describe how mutual exclusive access to resources in a distributed system can be achieved using a** **centralized coordinator.**

Steps of centralized coordinator:

When a process P1 wants to access a shared resource, it sends a request to the coordinator.

If no other process is currently accessing that resource, the coordinator sends a reply granting permission.

P2 requests permission to access the resource.

The coordinator knows that P1 is already accessing the resource, so it cannot grant permission.

It queues the request from P2.

When P1 is finished with the resource, it sends a message to the coordinator releasing its access.

Coordinator takes the first item off the queue of deferred requests and sends that process a grant message.

1. **Describe how mutual exclusive access to resources in a distributed system can be achieved using a token-ring algorithm.**

Token-ring algorithm construct an overlay network in the form of a ring in which each process is assigned a position in the ring. All that matters is that each process knows who is next in line after itself.

Steps:

When the ring is initialized, process P0 is given a token which circulates around the ring.

The token is required to access the shared resource.

When a process acquires the token, it checks if it needs to access the shared resource.

After it has finished, it passes the token along the ring.

1. **What is the purpose of an election algorithm in a distributed system?**

Election algorithms attempt to locate the process with the highest identifier and designate it as coordinator. And the goal of an election algorithm is to ensure that all processes agree on who the new coordinator is.

1. **With the aid of an example, describe how the bully election algorithm works.**

Steps:

Consider N processes {P0, ... ,PN−1} and let id(Pk) = k.

When a process notices that the coordinator is no longer responding to requests, it initiates an election.

A process Pk holds an election as follows:

1. Pk sends an ELECTION message to all processes with higher identifiers: Pk+1, Pk+2, ..., PN−1.

2. If no one responds, Pk wins the election and becomes coordinator.

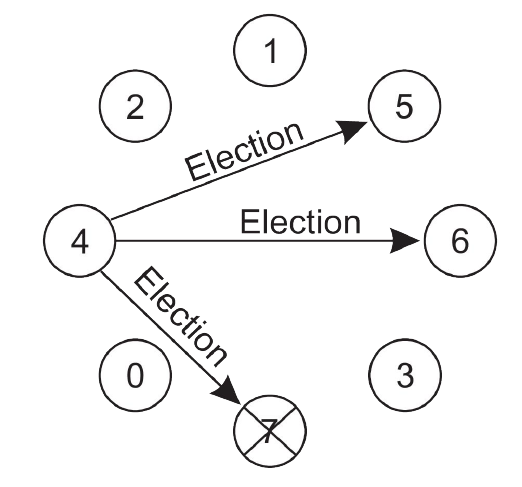
3. If a process responds, it takes over (holds an election) and Pk’s job is done.

Eventually, all processes give up but one, and that one is the new coordinator.

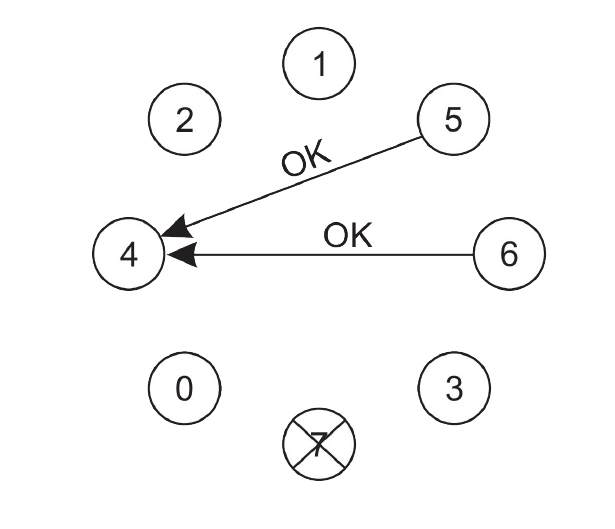
It announces its victory by sending all processes a message telling them it is the new coordinator.

Example:

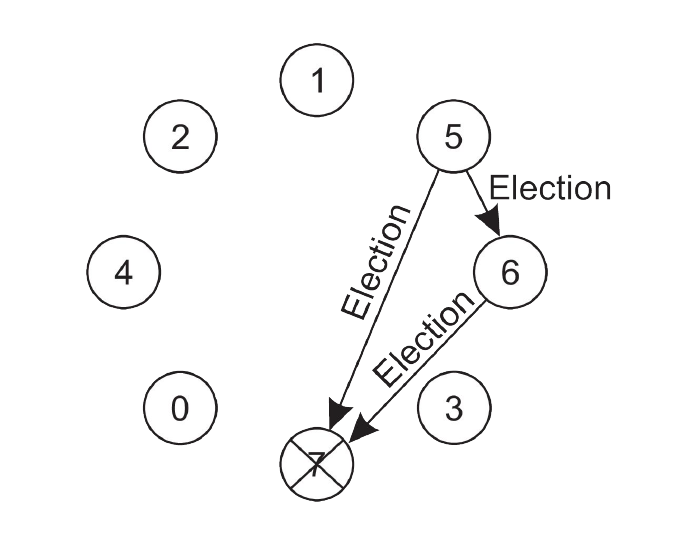
1. Previously process P7 was the coordinator, but it has just crashed. Process P4 is the first one to notice this, so it sends ELECTION messages to all the processes higher:



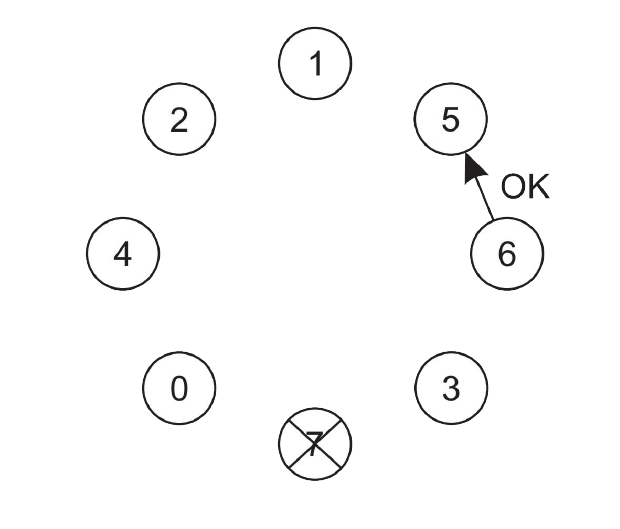
2. Processes P5 and P6 both respond with OK. Upon getting the first of these responses, P4 knows that its job is done:



1. Both P5 and P6 hold elections, each one sending messages only to those processes with identifiers higher than itself.



1. P6 tells P5 that it will take over. At this point P6 knows that P7 is dead and that it (P6) is the winner.



1. P6 announces the takeover by sending a COORDINATOR message to all running processes.

