ACTIVITY 1 OBJECTIVES: To illustrate the **Cristian's synchronization algorithm**.

A system employs the Cristian algorithm to synchronize the clocks of its computers. At some time, one of the client nodes sends the request message (for synchronization) to the server and when the server receives it, server's local clock marks 11:01:04,830 (i.e. eleven o'clock, 1 minute, 4 seconds and eight hundred thirty milliseconds). The client sent the request message at time 11:01:04,750 and received its response at time 11:01:04,880.

a) The client, to which value sets its local clock?

b) Will the client's local clock have to suspend its progress for a while? Why?

c) If in addition we know that the request message needed to be transmitted 50ms and its response 80ms, could we claim that the synchronization made by Cristian algorithm was accurate? Why?

ACTIVITY 2 OBJECTIVES: To illustrate the **Berkeley's synchronization algorithm**.

In a distributed system with 4 nodes, we employ Berkeley algorithm for synchronizing their clocks. One of the nodes (node A) acts as the server of the algorithm, whereas the others (nodes B, C and D) act as clients. Let's suppose that all nodes have clocks that indicate the number of ticks occurred from the same temporary base; and that at one point they have the following value for their clocks: C_A=10000, C_B=10005, C_C=10005 and C_D=10010 (being C_n the clock of node *n*). At that point the *server* (node A) initiates the Berkeley algorithm.

If we assume that it takes 10 ticks each time a message is sent, from its sending to its reception, which is the final value of the clocks of each node?

ACTIVITY 3 OBJECTIVES: To illustrate the **Berkeley's synchronization algorithm**.

Discuss which effect will have on Berkeley's algorithm its usage on a local network with 10 computers and where it occurs the following cases:

a) One of the computers has a faulty clock that is systematically delayed one minute per each hour passed. The rest of nodes have "normal" clocks that do not suffer any relevant forward moves or delays during their behaviour.

b) None of the computers have a correct clock (i.e. accurate clock) but their defects that they might have are compensated between them.

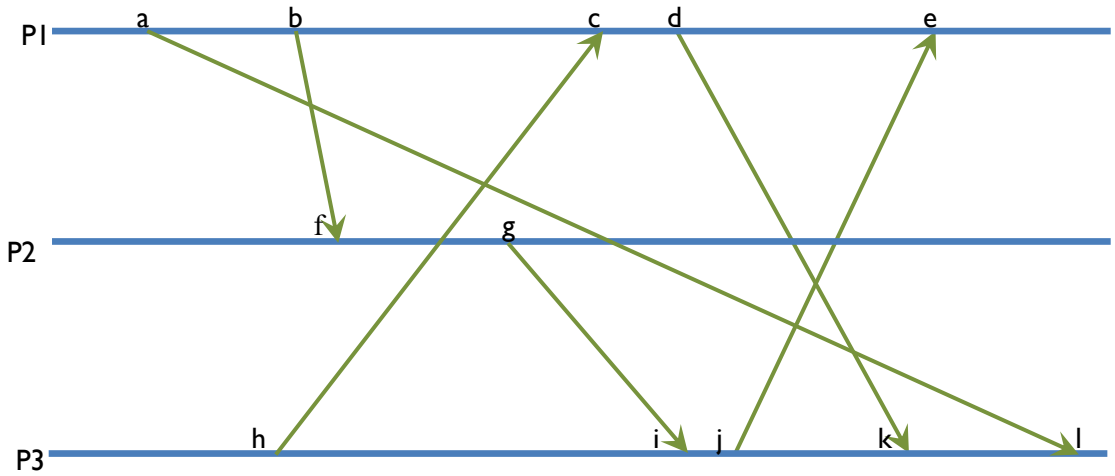
ACTIVITY 4 OBJECTIVES: To describe the synchronization algorithms for physical clocks.

Given the following sentences, update them as needed to make them all **TRUE**.

1. In the Cristian algorithm, if a client C asks the server its time at instant 20.000 (according to C clock), receives the server answer at instant 20.010 (according to C clock) and calculates that the new value for its clock must be 20.024, then you can deduce that the answer of the server has been 14.
2. In the Berkeley algorithm, the node that acts as coordinator never needs to adjust its clock. Only the clocks of the other nodes that participate in the algorithm are adjusted.
3. The Berkeley algorithm assumes that sending a message from node A to node B takes the double of time than the answer from B to A.
4. In the Cristian algorithm, the average between the client clock and the server clock is calculated to set the client time.
5. In the Berkeley algorithm, each client can synchronize its clock at any time, independently of when the other nodes synchronize their clocks.
6. Suppose that six nodes use the Berkeley algorithm to synchronize their clocks. The node N_1 starts the algorithm when the values of the clocks of the six nodes are respectively $N_1 = 2002$, $N_2 = 2008$, $N_3 = 2005$, $N_4 = 2010$, $N_5 = 2000$ and $N_6 = 2005$ and it receives the responses from the other nodes to the same time. Given these assumptions, node N_1 determines that it must advance its own clock in 30 units of time.
7. In the Berkeley algorithm, if we assume a total of N nodes, it is required to send a total 3*N messages.
8. The Cristian's algorithm requires knowing a priori the average delay of the messages exchanged between the different nodes.
9. In the Cristian's algorithm, if a client C asks the server for its time at the instant 10.000 and the server answers 10.006 and this response arrives to C at instant 10.008, then node C must stop its clock for 2 units of time.

ACTIVITY 5 OBJECTIVES. To undertand the usage of logical clocks. To illustrate **Lamport's algorithm**.

FORMULATION: Given the next execution in a distributed system formed by 3 nodes…



a) Indicate for each of these events of sending and receiving that are shown in the figure, which is the value that Lamport algorithm assigns for these events (i.e. the value of the Lamport's logical clock for these events). Assume that the execution begins at the left of the figure: all nodes start from time 0.

a:		b:		c:	
d:		e:		f:	
g:		h:		i:	
j:		k:		l:	

b) Indicate which events are concurrent with event "i".

c) Indicate which events are concurrent with event "e".

d) Can you give a total order of the events? If so, which one?

ACTIVITY 6 OBJECTIVES: To distinguish between **logical and vector clocks**. To identify the advantages of having a logical ordering of events in a distributed application.

Given the execution shown in previous activity:

a) Indicate which vector clock will be assigned each of the events shown in that execution, assuming that initially each process has a local clock with value [0,0,0].

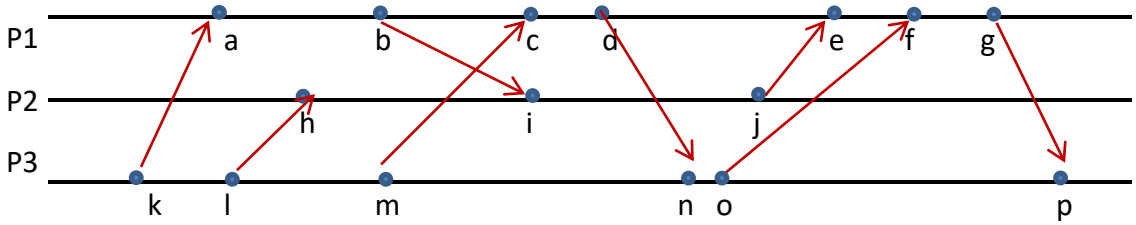
a:		b:		c:	
d:		e:		f:	
g:		h:		i:	
j:		k:		l:	

b) Using the vector clocks of the previous question, give a complete list of pair of events that are concurrent between them. For example, an element of that list will be the pair of events "a" and "h" and their respective vector clocks are [1,0,0] and [0,0,1].

c) Imagine that after the trace shown in that previous execution, process P2 sends a message to process P1 and there is no other event in this system... which vector clocks will be assigned to the sending event and to the receiving event of that message?

ACTIVITY 7 OBJECTIVES: To distinguish between **logical and vector clocks**.

Given the following schedule showing the execution of three processes in a distributed system, each in a different node:



a) Indicate the Lamport's Logical Clock value for each of these events

b) Indicate the relationships "happens - before" and what events are concurrent with each other

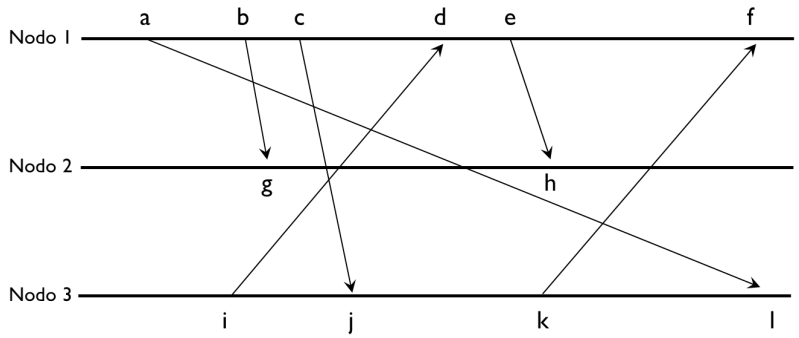
c) Looking only at the values of Lamport's logical clocks, can we determine which events have happened before others? And what events are concurrent?

d) Indicate the Vector Clock value for each of these events

e) Looking only at the values of these clocks, can we determine which events have happened before others? And what events are concurrent?

ACTIVITY 8 OBJECTIVES: To distinguish between **logical and vector clocks**.

Given the following schedule:



Indicate whether the following sentences are true (T) or false (F) and justify your answer:

1. It holds that **c** $\rightarrow$ **l** and that **e** || **l** JUSTIFICATION:	
2. The Lamport logical clock in **l** is equal to 6. JUSTIFICATION:	
3. The vector clock in **d** is equal to V(d)=[4,0,1] and the vector clock in **j** is equal to V(j)= [3,1,2] JUSTIFICATION:	

ACTIVITY 9.- OBJECTIVES: To distinguish between **logical and vector clocks**.

Indicate whether the following sentences are true (T) or false (F). In case it is false, update the sentence to make it true.

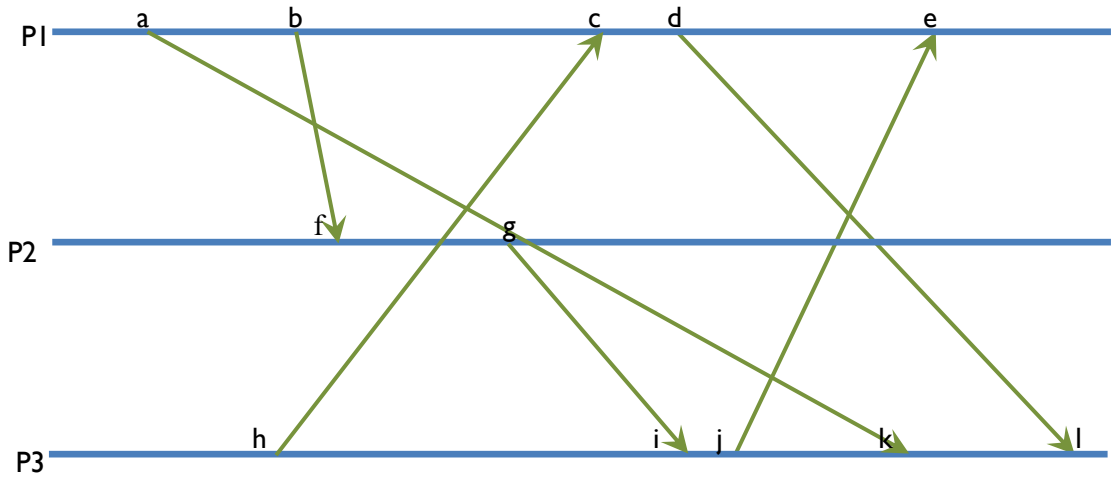
1. We say that event "a" precedes event "b" (a -> b) if all nodes agree that they first see event "a" and then see event "b".
2. The Lamport's logical clocks guarantee that within the same node the values L(x) are always strictly increasing.
3. Two events "a" and "b" of different nodes cannot have logical values associated with L(a) and L(b) such that L(a) = L(b).
4. The vector clock associates a vector value to each event (e.g. for event "x" we speak of V(x)), so that if for any two events "a" and "b", if it holds that V(a)< V(b), then a -> b.
5. In vector clocks, at each node we need a vector of integer values, with one element for each possible event.
6. In vector clocks, for any pair of distinct vectors V(a) V(b), it holds that V(a)<V(b) or V(b)<V(a).

ACTIVITY 10 OBJECTIVES: To understand how to determine the global state that presents a distributed architecture at a specific time . To illustrate **Chandy and Lamport algorithm.**

FORMULATION: Discuss whether Chandy and Lamport algorithm can be used in the system presented in activity 5.

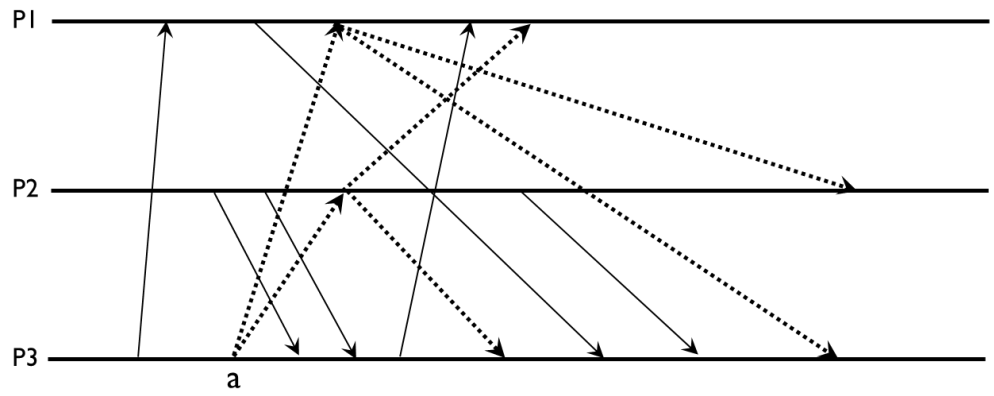
ACTIVITY 11 OBJECTIVES: To illustrate **Chandy and Lamport algorithm**.

Given the following trace of a system formed by three processes:



Assume that process P2 starts the execution of Chandy & Lamport algorithm immediately after generating event "g" and that the MARK message that broadcasts is delivered to P1 and P3 just before their events "c" and "j", respectively, thus generating broadcastings of new MARK messages before events "d" and "j", respectively. Explain which trace will follow this Chandy & Lamport algorithm and what messages in transit (i.e. messages sent and not yet delivered) are stored in each channel.

ACTIVITY 12 OBJECTIVES: To illustrate **Chandy and Lamport algorithm**.



Let us suppose that P3 starts the Chandy-Lamport algorithm at event **a**, being the broken lines the messages sent as a consequence of the execution of this algorithm; and the continuous lines are the normal messages. When the algorithm finishes, please indicate how many messages are registered in these channels:

- In channel (P3,P1) :
- In channel (P2, P3) :
- In channel (P1, P3) :

ACTIVITY 13 OBJECTIVES: To illustrate **Chandy and Lamport algorithm**.

Given the following sentences, modify them as needed to make all them TRUE.

1. A snapshot is considered to be inconsistent if it includes messages sent and already not received.
2. The Chandy-Lamport's algorithm requires total connectivity: between each pair of nodes `a,b` there must be unidirectional channels `(a,b)` and `(b,a)`.
3. The initiator node is the only one who sends MARK messages.
4. Whenever a process receives a MARK message, it must store its local state.
5. The Chandy-Lamport algorithm establishes a total order among the events of a distributed system.

ACTIVITY 14 OBJECTIVES: To describe the **leader election** and **mutual exclusion** algorithms in distributed systems.

FORMULATION: Given the following sentences, justify whether they are True or False:

1. The Bully algorithm fails if two or more nodes start the algorithm simultaneously (i.e. when there is more than one initiator). JUSTIFICATION:	
2. The leader election algorithm for rings fails if two or more nodes start the algorithm simultaneously (i.e. when there is more than one initiator). JUSTIFICATION:	
3. The centralized mutual exclusion algorithm seen at classroom restricts the scalability and fault tolerance because the coordinator is a bottleneck and a single fail point. JUSTIFICATION:	
4. The distributed mutual exclusion algorithm seen at classroom employs logical clocks to order some entry requests to the critical section. JUSTIFICATION:	

5. The Berkeley algorithm is one of the most efficient mutual exclusion algorithms. JUSTIFICATION*:*	
6. The distributed mutual exclusion algorithm does not need any action in its output protocol. JUSTIFICATION:	
7. The Cristian algorithm is a very efficient leader election algorithm. JUSTIFICATION*:*	

ACTIVITY 15 OBJECTIVES: To describe the **leader election** and **mutual exclusion** algorithms in distributed systems.

Let be a system with 8 nodes (N_1, N_2, ... N_8) where at a given moment only the odd nodes are active (and this information is not yet known by the nodes).
COMPLETE THE FOLLOWING SENTENCES:

1. In leader election algorithm for rings, if any node receives an ELECTION message with the content (5, {5, 3}), this message indicates:
2. If the Bully algorithm is used for leader election, if the N_7 node sends an ELECTION message, this indicates that:
3. For the leader election, if node N_1 initiates the Bully algorithm, then it will send an ELECTION message to nodes:
4. If node N_5 initiates the leader election algorithm for rings, at some moment it will receive a message of type ELECTION with the following content:

ACTIVITY 16 OBJECTIVES: To describe the classical solutions to synchronization problems in a distributed environment.

FORMULATION: Given the following sentences, justify whether they are True or False:

1. The Chandy-Lamport algorithm requires the channels not to lose nor disorder messages. JUSTIFICATION:	
2. It is usually problematic to make a computer clock go backwards. JUSTIFICATION:	
3. The Berkeley algorithm does not assume a node with a precise clock. JUSTIFICATION*:*	
4. The Berkeley algorithm synchronized clocks of nodes of a distributed system without taking into account the divergence of these clocks and the real official clock. JUSTIFICATION:	
5. The Cristian algorithm is used to manage logical clocks. JUSTIFICATION:	
6. The Cristian algorithm provides the basis to implement any decentralized algorithm. JUSTIFICATION*:*	
7. Chandy & Lamport algorithm requires FIFO communication channels. JUSTIFICATION*:*	
8. Cristian algorithm allows identifying the messages in transit in each of the communication channels of a distributed system. JUSTIFICATION*:*	

ACTIVITY 17 OBJECTIVES: To describe the **consensus** problem.

Given the following sentences, update them as needed to make them all **TRUE**.

1.- It is a problem of consensus when several nodes adopt as a "consensus value" the average of the values estimated by them.
2.- It is a problem of consensus when several nodes adopt as a "consensus value" the maximum of the values estimated by them.
3.- After executing the consensus algorithm, only the nodes that initially proposed "estimate (V)" will provide "decision (V)" as output.
4.- A correct consensus solution must comply with the properties of mutual exclusion, progress and non-preemption.
5.- At the end of the consensus algorithm, two correct processes could lead to different decisions, as each will have initially proposed an "estimate (Vi)" and, therefore, they will result in a different "decision (Vi)".
6.- The property of "uniform integrity" is fulfilled when, if a process decides "v", then "v" was proposed by some process.
7.- The Paxos and Raft algorithms are widely used implementations of consensus algorithms.
8.- An example of a consensus algorithm is that all the nodes broadcast their "estimate(V_i)" and decide as "decision(V)" the value proposed by the node with the highest identifier.

ACTIVITY 18 OBJECTIVES: To describe the **consensus** problem considering failures.

Given the following sentences, update them as needed to make them all **TRUE**.

1.- In a distributed consensus algorithm, in which it is considered that the nodes can fail, the nodes have a "fault detector" to know if the node has failed when calculating its "estimate (V)" value that needs to propose.
2.- They are called "eventually perfect failure detectors" because they are detectors that will sooner or later determine a failure perfectly.
3.- The consensus algorithm works if there is a maximum of N/2 correct nodes.
4.- If we have 7 nodes, and 2 fail, the consensus algorithm will not work, because the coordinator node will be blocked.
5.- In the algorithm, the nodes execute rounds. The node terminates the algorithm when the coordinator node is chosen.
6.- In each round, the ordinary nodes send a "propose" message to the coordinator of the round.
7.- In each round, the ordinary nodes wait to receive a "propose" message from the coordinator.
8.- The coordinator chooses one of the "estimate" values that he receives from all those who have the maximum value of the round.