# CSIE52400 Distributed Systems, Spring 2011
# Final Exam

ID: ___________________ Dept:__________________ Name: ________________

1. (25%) **Basic concepts**.

(a) What are replication, concurrency, and failure transparency in distributed systems? Give examples to show your point. Achieving full transparency is impossible. Why? Give an example that illustrates why it is not a good idea to always aim at complete transparency.

(b) What is RMI? What are the differences between RMI and RPC? Are there any cases where we must use one approach instead of the other?

(c) What do we mean by openness in distributed systems? How can we achieve openness? Give an example of a distributed system that can be considered as open? Does this system satisfy the definition of openness you just described?

(d) What does it mean for two events (say A and B) to be concurrent? What is the relationship of the Lamport's logical clock of A and B? What about Vector clock of A and B?

.

(e) What is the relationship between blocking/non-blocking and synchronous/asynchronous primitives? Is it possible to have a blocking asynchronous operation? What about nonblocking synchronous operation?

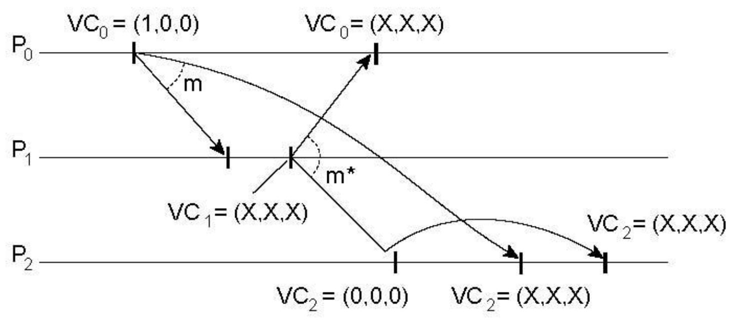
2. **(30%)** Answer the following TRUE/FALSE questions.

| No. | Problem Description | TRUE/FALSE |
|---|---|---|
| (a) | In a distributed system, we can always determine the causal relationship between two events. | |
| (b) | For any two events x and y, if $x \rightarrow y$, then there must exist a path in the space-time diagram from y to x. | |
| (c) | For any two events x and y, $\neg(x \rightarrow y) \Rightarrow x \parallel y$. | |
| (d) | For any two events x and y, $\neg( x \parallel y) \Rightarrow$ x and y are causally related. | |
| (e) | For the purpose of synchronization in a distributed system, we can simply synchronize the physical clock of each process during initialization, and then use the clock time for synchronization. | |
| (f) | The global state of a distributed system with *n* processes is the union of the local states of the *n* processes. | |
| (g) | In a consistent global state, proper causality must be maintained. Therefore a message *m* is sent in some process, it must also be received in another process in the same global state. | |
| (h) | In a space-time diagram, a consistent cut corresponds to exactly one consistent global state. | |
| (i) | In a synchronous communication, the sender must wait until the receiver process has received the message. | |
| (j) | In general, synchronous communication allows higher degree of parallelism while asynchronous communication leads to easier programming. | |
| (k) | A distributed system with higher degree of transparency is always better than the case with lower degree of transparency. | |
| (l) | A cluster computing system is a type of distributed system normally with heterogeneous nodes running on high speed networks. | |
| (m) | The degree of transparency of a network operating system is usually much lower than that of a multiprocessor operating system. | |
| (n) | A message-passing system can act as a shared memory system using emulation. | |
| (o) | It is usually easier to program a message-passing system than a shared memory system. | |

3. **(25%) Logical Clocks and Vector Clocks**.

(a) Suppose there are three processes A,B and C. All clock runs at the same rate but initially A's clock reads 5, B's clock reads 0 and C's clock reads 10. At time 5 by A's clock, A sends a message to B, this message takes 3 units of time to reach B. B then waits one unit of time and then sends a message onto C which takes 4 units of time to reach C. Assuming that the system implements Lamport's logical clocks, draw a picture illustrating the logical timestamp for each message and explain how it is obtained.

(b) Fill in the vector clocks in the following figure in a causally order multicast environment and explain why the delivery of message m* at process P2 is delayed.



(c) Let a, b be two events and L(a), L(b) be their logical timestamps and VC(a), VC(b) be their vector timestamps. Determine the correctness of the following statements about event ordering and logical time. Explain your answer.

1. $a \rightarrow b \Rightarrow L(a) < L(b)$
2. $L(a) < L(b) \Rightarrow a \rightarrow b$
3. $a \rightarrow b \Rightarrow VC(a) < VC(b)$
4. $VC(a) < VC(b) \Rightarrow a \rightarrow b$

(Empty page for answering the exam questions.)

4. (20%) Answer the following questions about the algorithms and techniques introduced in the class.

(a) In the Chandy-Lamport snapshot algorithm, what is the purpose of sending the markers? When do we record a channel as empty and why is it considered empty? The state of a channel is recorded according to the Marker Receiving Rule. Why do we record the state of a channel as the set of messages received after recording the process state and before the receiving of the marker?

(b) In Java threads, what are the differences between inheriting from the `Thread` class and implementing the `Runnable` interface? In the reader/writer programming assignment, how do we allow many readers to read the shared array concurrently? How do we synchronize the reader/writer threads such that writer threads have privilege over reader threads (i.e. when there are both readers and writers waiting for the array, writers are allowed to access before readers.)

(c) In the RMI assignment, what would happen if two or more processes are sending messages to the same mailbox at the same time?  In other words, what would happen when a remote method is invoked by two or more clients at the same time?  How do you synchronize the updates(i.e. send, read, check, delete) to the same mailbox by different clients?  Do we need a multithreaded RMI server?