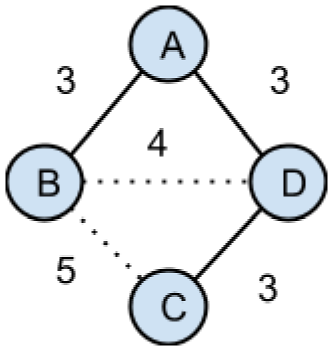
Assignment 3 - ECE 454

Piyush Gadigone, 20366910
Yash Malik, 20379044

1.

Consider the following MST:

In the below figure, the MST is B-A-D-C. Consider making a Steiner tree using vertices B and D. The Steiner tree would be composed of edge B-D since it has the minimum weight. This Steiner tree is not a subtree of MST, hence disproving the claim.



2.

This can be proved using induction.

Base Case:
Consider a tree where V[T] = 1. A tree with one vertex requires no edges. Hence, E[T] = 0.

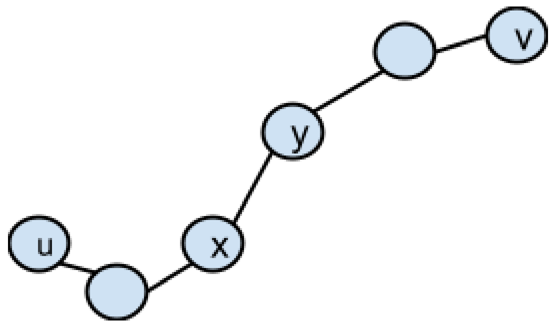
Induction Hypothesis:
Let T(k) be any node with k-1 edges.

Inductive Step:
Now we need to prove that if the statement is true for T(k), then it is also true for T(k+1).
Let T(k+1) be any tree with V[T] = k+1. Take any node v of T(k+1) of degree 1. Consider T(k), subgraph of T(k+1) created by removing v and the edge connecting it to the rest of the graph. The number of vertices for T(k) is k and it has one less edge than T(k+1) by definition. By hypothesis, T(k) has k-1 edges. So T(k+1) must have k edges.

This proves the claim that E[T] = V[T] - 1.

3.

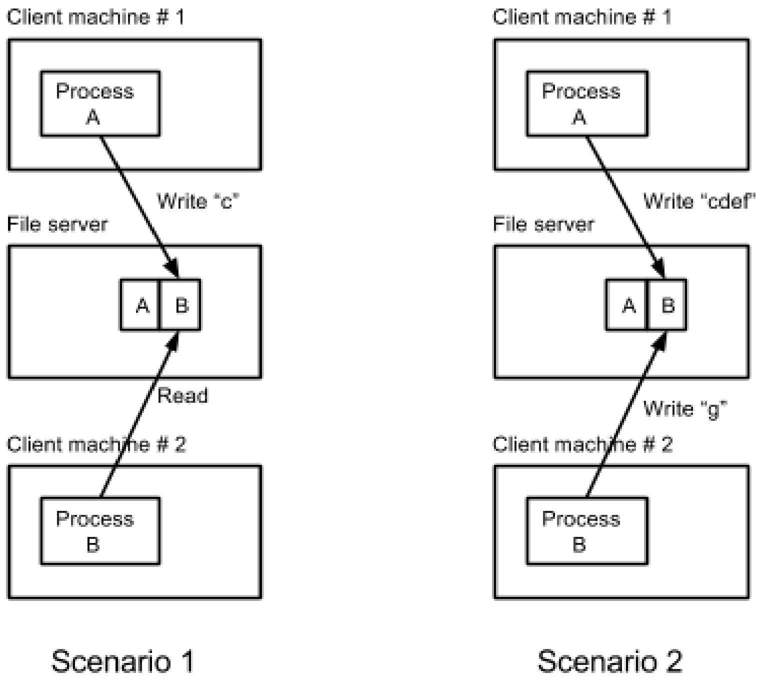
Consider the following graph:



In the graph above, there exists a unique path from u to v. If there exists no edge between x and y, there is no longer a path between u and v and consequently the graph is not an MST. If there are multiple paths between u and v, then the path from u to v is not longer UNIQUE. Thus, there must be exactly one edge crossing the cut connecting u and v.

4. a)

Consider the following two scenario:



Scenario 1

1) Client 1 request to write “c” to the file server.
2) While Client 1 is writing, Client 2 reads, but since Client 1 is not done writing, it reads “ab”.
3) Client 1 completes write “c”

In the above scenario, Client 1 initiated a write before Client 2 initiated a read, thus, Client 2 should have read the value “abc”. Because there is no mechanism in place to ensure this, the property “a read(file) returns the effect of the last write(file)” is violated.

Scenario 2

1) Client 1 request to write “cdef” to FILE on the server.
2) Client 2 request to write “g” to FILE on the server.
3) Client 2 has a smaller write content, it finishes before Client 1. When Client 1 finishes, the file server accepts “cdef” instead of “g”.

In the above scenario, because Client 2 initiates a write (FILE) after Client 1, its changes should be kept. Because there is no mechanism in place to ensure this, the property “if we have two write(file) operations that overlap in time, the last write takes effect.” is violated.

b)

Yes, the UNIX notion of correctness is met in this case. Re-consider Scenario 2 from part a), it was shown that the write content of Client 1 is reflected in the file because it finishes after Client 2. If we change the notion of UNIX correctness to be “... the last write to complete take effect.”, then the happenings in scenario 2 are correct.

c)

The following conditions need to be satisfied to meet the UNIX notion of correctness:
i) a read(file) returns the effect of the last write(file), and,
ii) if we have two write(file) operations that overlap in time, the last write takes effect.”

The Start transaction and End transaction messages can be used by the server to ensure that the UNIX notion of correctness are met.

When a process, A, needs to make changes to a file, it starts off by issuing a Start transaction message. The server now knows to only accept read and write requests from process A, and drops all other Start transaction requests. When process A completes its transaction, it issues an End transaction message, notifying the server that its transaction is complete. At this point, the server is open to other Start transaction messages. This ensures i), where if a write(file) from process A is in progress, read(file) gets ignored, and ii) where the second write request is ignored, until the first one completes.

5)
The maximum skew would be equal to the difference in one minute in the times of the clock that ticks 1000 times/millisecond and the clock that ticks 990 times/millisecond.

$$Skew = (1000\ ticks/ms - 990\ ticks/ms) * 1000\ ms/s * 60\ s = 600000\ ticks$$

$$Skew\ in\ seconds = 600000\ ticks * 0.0001\ ms/ticks = 600\ ms = 0.6\ s$$

The maximum skew would be 0.6 seconds until the UTC update arrives every minute..

6)
Yes, acknowledgements are necessary to achieve totally-ordered multicasting. INCOMPLETE