**Lanny Xu** 1.    A client makes 3 successive queries to a timeserver with a WWV receiver. It records the round-trip times and times returned by each query in the table below.

Query   Round-trip (ms)    Time (hr:min:sec)  
---------------------------------------------  
  
1            32               2:32:11.970       
  
2            28               2:32:11.999  
  
3            34               2:32:12.072

1. Which of these times should it use to set its clock? (Pick the one that minimizes the error.) To what time should it set it? Estimate the accuracy of the setting with respect to the server's clock.

The second one, because the round-trip times is the minimum 0.028s. The actual time should between be T+ 0.028/2 = 2:32:12.013. The accuracy is 28/2 = ±14ms.

1. If it is known that the time for the request to reach the server is at least 3 ms and the time for the reply to reach the client is also at least 3ms, do your answers (time, accuracy) change? If so, how?

Time does not change (because the transfer time is the same T+3+round-trip time-3), but the accuracy changes to (28-3-3)/2 = ±11ms.

1. If it is known that the time for the request to reach the server is at least 5 ms and the time for the reply to reach the client is at least 3ms, do your answers (time, accuracy) change? If so, how?

Time changes, (T+5+round-trip time-3)/2 = 2:32:12.012. And accuracy changes to (28-5-3)/2 = ±10ms.

2.    Consider the behavior of two machines in a distributed system. Both have clocks that are supposed to tick 1000 times per millisecond. One of them actually does, but the other ticks only 960 times per millisecond.

1. If UTC updates come in once a minute, what is the maximum clock skew that will occur?

The second clock ticks 960,000 times per second, giving an error of r = 40 milliseconds per second. In a minute this error has grown to 2400 milliseconds.

1. How often should we re-synchronize the clocks if we allow a skew of at most 200 milliseconds?

Re-synchronize every 200/40 = 5.

3.    Assign Lamport timestamps to the events A through P below.

    A        B   C   D  E  
P1---------------------------  
     \      /   /   /   \   
      \F   /  G/   /     \H   
P2--------/-------/----------  
   I\    / J\    /     \K  
     \  /    \  /       \  
P3---------------------------  
     L  M    N  O        P

Explanation of above drawing: events A and F are the send and receive events of the same message, respectively. So are G and C, M and B, J and N, etc.

    A1      B4   C5  D6  E7  
P1---------------------------  
                    
      F2   G4        H8   
P2---------------------------  
  I1    J3        K5  
                    
P3---------------------------  
    L2  M3   N4  O5        P6

4.    Repeat the above with vector timestamps.

A      B   C  D  E  
P1---------------------------  
                    
      F   G        H   
P2---------------------------  
  I    J        K  
                    
P3---------------------------  
     L  M    N  O        P 

**A****[1, 0, 0] B[2, 1, 2] C[3, 4, 2] D[4, 4, 4] E[5, 4, 4]**

**I[0, 1, 0] F[1, 2, 0] J[1, 3, 0] G[1, 4, 0] K[1, 5, 0] H[5, 6, 4]**

**L[0, 1, 1] M[0, 1, 2] N[1, 3, 3] O[1, 3, 4] P[1, 5, 5]**

5.    Suppose process A chimed when its **vector** clock was (3, 2, 5), process B chimed when its clock was (6, 5, 5), and process C chimed when its clock was (4, 4, 7). Can you answer these questions: Who chimed first? Who chimed last?

It’s clear to see that every component in A is less than both B and C, so process A chimed first.

But B and C is a tricky question because even though 6>4 and 5> 4, 5< 7. So this cause the impossibilities to pick which one chimed last.  
  
6.    Three processes P, Q, and R connected through bi-directional channels P<-->Q, Q<-->R, and R <--> P constantly rotate TWO messages L and K. L moves clockwise: from P to Q to R to P etc., while K moves counterclockwise. At any one time, there is only one copy of L and one copy of K in the system. Each process's state consists of a pair of counters (l,k) where l is the number of times the process has received L, and k is the number of times the process has received K. All counters are initially (0,0). At a certain point, suppose P has the message L, R has the message K and the states of P, Q, and R are respectively (6,4), (5,3) and (5,4). Immediately after sending L, P initiates the snapshot algorithm -- note that the snapshot marker m will follow message L. Explain the operation of the algorithm in this case, giving the snapshot reported by it. Do not forget the message(s) in transit. Assume lossless FIFO communication.

P sends L to Q turns Q to (6, 3), and record the state P(6, 4).

P sends marker M1 to Q, and M2 to R.

Q sends L to R which turns R to (6, 4), and record the state Q(6, 3).

M1 has nothing so record PQ is ∅, Q sends marker M3 to R and M4 to P.

M4 has nothing so record QP is ∅.

R sends L to P and since P is (6, 4), it makes RP to L.

R sends K to Q, which turns Q from (6, 3) to (6, 4) and record state R(6, 4).

R: M3 has nothing so record QR is ∅, and R sends marker M5 to P and M6 to Q.

Q sends to P and since P is (6, 4), this makes QP switch from ∅ to K. M5/6 has nothing so record RP/RQ to ∅.

**P (6, 4) RP= L QP = K**

**R(6, 4) QR =** **∅, RP = ∅ Q(6, 4) PQ =** **∅, RQ = ∅**