Lecture 18

- Administration

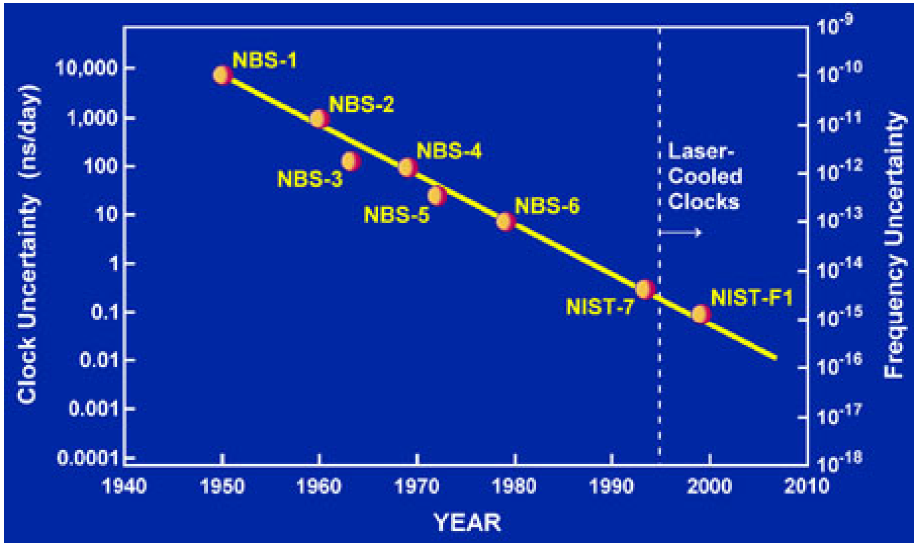
Clocks

- Properties of Physical Clocks
- Externally, Internally synchronizing clocks
- Cristian's Algorithm

Time Domains

- Internet seconds
- Cluster - Data Center milliseconds
- Inside a node microseconds

Atomic Clocks



US Naval Observatory

Coordinated Universal Time

- Crystals
- Atomic time, how to keep it coordinated with astronomical time

Leap Seconds

- Leap Years
- 365.25 days. But because the length of the solar year is actually 365.242216 days (the 400 rule)
- Leap seconds added through the year

Physical Clocks & Synchronization

- In a DS, each process has its own clock.

- Clock Skew versus Drift
  - Clock Skew = Relative Difference in clock *values* of two processes
  - Clock Drift = Relative Difference in clock *frequencies (rates)* of two processes

- *A non-zero clock drift will cause skew to continuously increase.*

- Maximum Drift Rate (MDR) of a clock is defined relative to Coordinated Universal Time (UTC)
  - MDR of a process depends on the environment.

- Max drift rate between two clocks with similar MDR is 2 * MDR

Synchronized Clocks

- Externally Synchronized

- Internally Synchronized

Synchronizing Physical Clocks

- $C_i(t)$: the reading of the software clock at process i when the real time is t.

- External synchronization: For a synchronization bound $D>0$, and for source S of UTC time,

$$|S(t) - C_i(t)| < D,$$

  for $i=1,2,\ldots,N$ and for all real times t in I.
  Clocks C_i are accurate to within the bound D.

- Internal synchronization: For a synchronization bound $D>0$,

$$|C_i(t) - C_j(t)| < D$$

for $i, j=1,2,\ldots,N$ and for all real times t in I.
  Clocks C_i agree within the bound D.

- External synchronization with $D \Rightarrow$ Internal synchronization with $2D$

Properties of clocks

- Monotonically increasing
- Bounded drift (stronger condition on discontinuity)
- Allow discontinuity at re-synchronization points

Central Clock Problems

- Single server
- Clock failure
- Spurious values
- Imposter time server

Cristian's Algorithm

- Uses a *time server* to synchronize clocks
- Time server keeps the reference time
- A client asks the time server for time, the server responds with its current time, and the client uses the value *T* in the response message to set clock
- But round-trip time introduces an error…

Cristian's Clock Algorithm

Setting

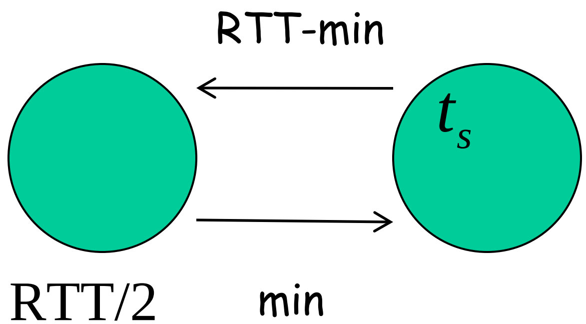
$$c(t) = t_s + \mathrm{RTT}/2$$

Accuracy

$$t_s \pm \left(\frac{\mathrm{RTT}}{2} - \min \right)$$

Cristian's Clock Algorithm

Earliest time that timestamp put in message

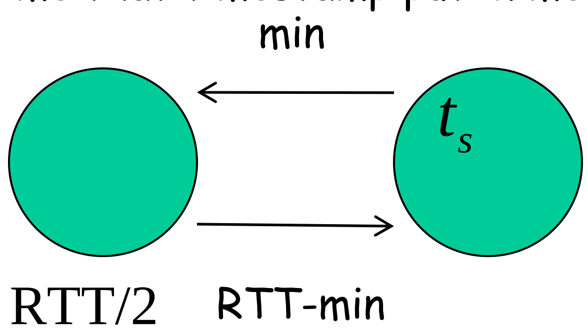


$$c(t) = t_s + \mathrm{RTT}/2$$

In both cases the

$$t_s \pm \left(\frac{\mathrm{RTT}}{2} - \min \right)$$

Latest time that timestamp put in message



$$c(t) = t_s + \mathrm{RTT}/2$$

$$t_s \pm \left(\frac{\mathrm{RTT}}{2} - \min \right)$$

Cristian's Algorithm

- Can increase clock value, but should *never* decrease it – Why?
- Can adjust speed of clock too (take multiple readings) – either up or down is ok.
- For unusually long RTTs, repeat the time request
- For non-uniform RTTs, use *weighted average*

avg-clock-error_0 = local-clock-error

$\text{avg-clock-error}_n = (W_n * \text{local-clock-error}) + (1 - W_n) * \text{local-clock-error}_{n-1}$