Physical clock synchronization [flaviu cristian]

Setup

- master clock that is assumed to keep perfect time (RT)
    - keeps time t

- slave clocks Ci that we want to synchronize to master
    - each keeps local time Ci(t)
    - assume that Ci is "correct" if it drifts at a rate p
        - i.e., $(1-p)\Delta <= Ci(t+\Delta) - Ci(t) <= (1+p)\Delta$

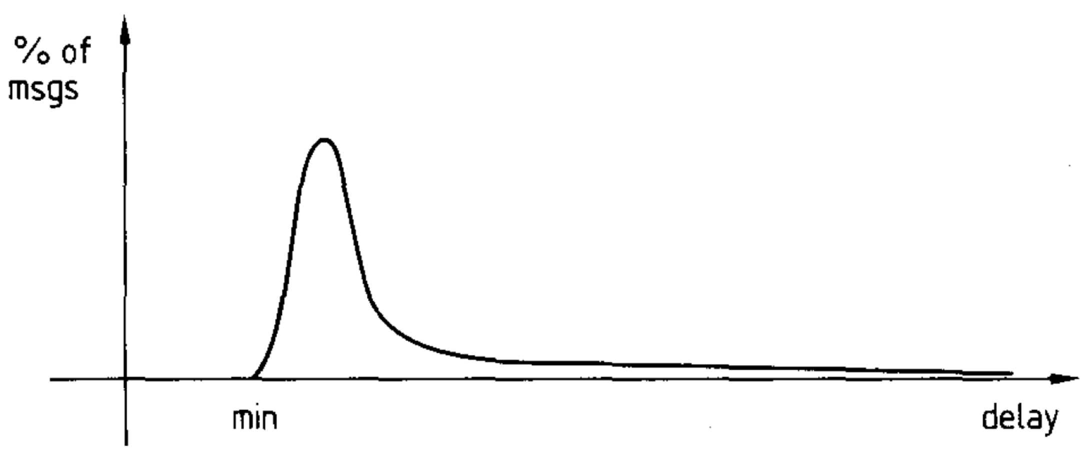
- want two properties from clock synchronization
    - Clock consistency (internal): $| Ci(t) - Cj(t) | < d1$ for all i, j
    - Clock accuracy (external): $| Ci(t) - t | < d2$ for all i

If you have external synchronization, get internal synchronization for free.

Why is clock synchronization hard?

We have to assume an asynchronous network. So, messages have:

- lower bound "min" on propagation delay, dictated by speed of light
    - if unknown, assume min = 0 (hurts estimates the most)
- no real upper bound on propagation delay
    - some algorithms assume a known max – problematic in practice



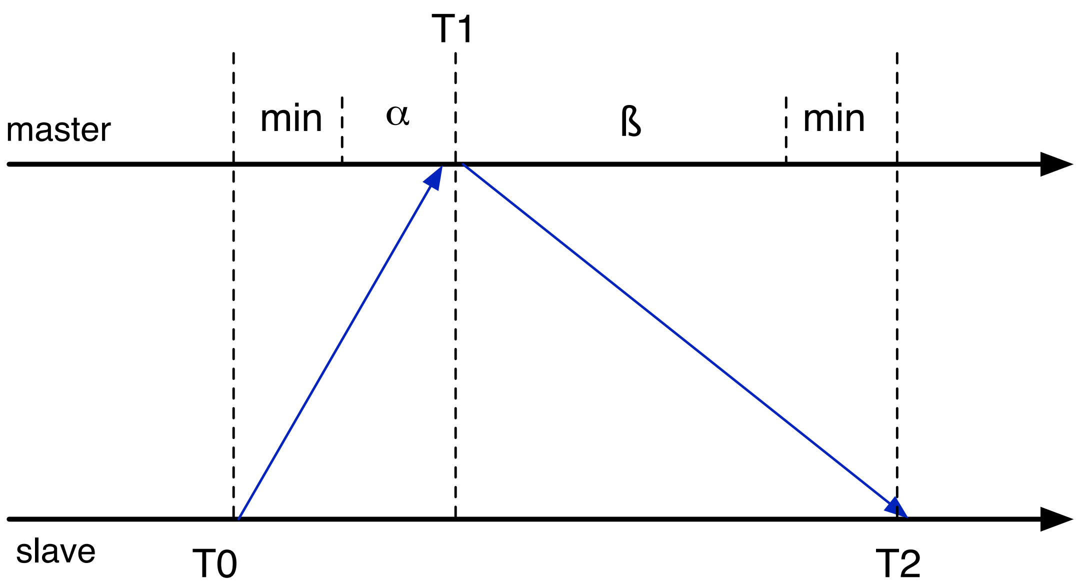
--> start the ping experiment

Simple broadcast-based time synchronization

Clock broadcasts time to all slaves
- broadcast message contains t
- slaves set clock to (t + min) when they receive broadcast

What is the accuracy of the clock?
- depends on where in the distribution the message delay is
- if assume "max" delay, then error could fall anywhere in the range (max – min)
- provable that this is the tightest error bound with probability 100%
  - therefore tightest consistency / accuracy

Interrogation-based time synchronization



Goal:
- figure out what the master's clock says when the slave's clock says T2
  - it depends on alpha and beta, obviously
    - bounded by two cases: alpha = 0, and beta = 0
  - if alpha = 0, then beta = (T2-T0) – 2*min
    - Cmaster(T2) = T1 + min + beta
    - Cmaster(T2) = T1 + (T2-T0) – min
  - If beta = 0 , then:
    - Cmaster(T2) = T1 + min

- least possible error is to pick the midpoint
  - Cmaster(T2) = T1 + ((T2 – T0) / 2)
  - Max error = ((T2 – T0) / 2) - min

That was ignoring clock skew p. If you factor in clock skew, then the equations get a little more complicated:

- Least possible error is to pick:
    - Cmaster(T2) = T + ((T2 – T0)/2)(1 + 2p) – min p
    - Max error = ((T2 – T0)/2)(1 + 2p) - min

Many implications to this:

- max error grows as clock skew climbs
- if you don't know "min"
    - have to set min = 0, and max error is basically proportional to the RTT
- error diminishes as the measurement trial RTT approaches 2*min
    - is a probabilistic tradeoff
        - can require measurements to be close to RTT to "accept" them and achieve rapport – increase number of trials necessary, but get tight error bounds
        - can be sloppy and take any measurement – decreases number of trials, but get worse error bounds

<u>Other realities</u>

- don't want jump discontinuities in time
    - play around with clock rate, rather than clock setting, to make clock drift into sync with master over a configurable time period
- often don't have a single master, but a distributed hierarchy of clocks
    - need a way to average estimates from multiple parents
- Q: does GPS change any of this fundamentally?
    - can get a pretty tight bound on "min"
    - alpha, beta are low
    - get very good synchronization error bounds as a result