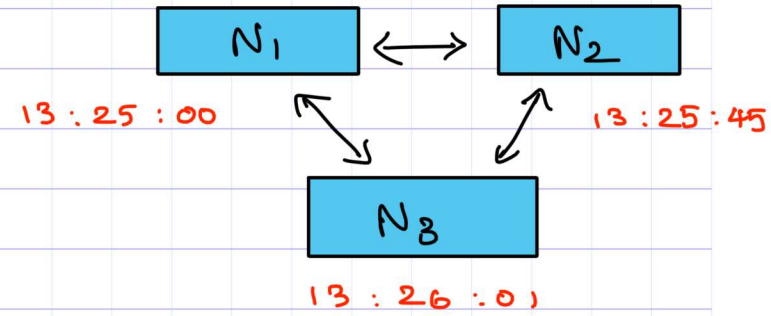
Synchronization (Absence of global clock)

$N_3 \rightarrow N_1, N_2$: 01

$N_1 \rightarrow N_3$: 01

$N_2 \rightarrow N_3$: 46



$N_2 \rightarrow N_1, N_3$: 47

$N_1 \rightarrow N_2$: 03

$N_3 \rightarrow N_2$: 04

Ordering of msgs w/ clock order (of N_1, say) leads to wrong ordering!

- Clock syncing isn't practical.
- Physical clocks aren't possible.
- Geographically distributed nodes ⇒ different timezones
- Cannot have a common clock.

Assumptions:

→ Comm. channel is error free.

→ Comm. channel is FIFO.

Clock syncing with message passing: [rcvd. timestamp + RTT/2] — based on assumptions

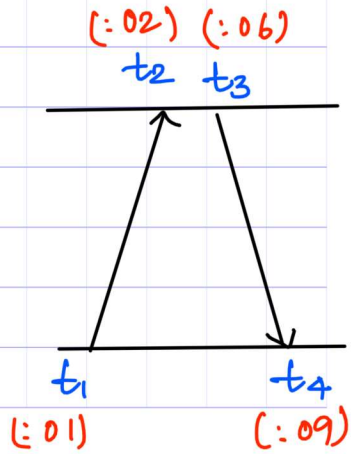
In case of delay & not FIFO:

$(t_4 - t_1) - (t_3 - t_2) = 8 - 4 = 4$

$(t_4 - t_1)$ ↓ RTT

$(t_3 - t_2)$ ↓ processing overhead

4 ↓ true RTT



true RTT/2 = 4/2 = 2

↓

almost accurate.

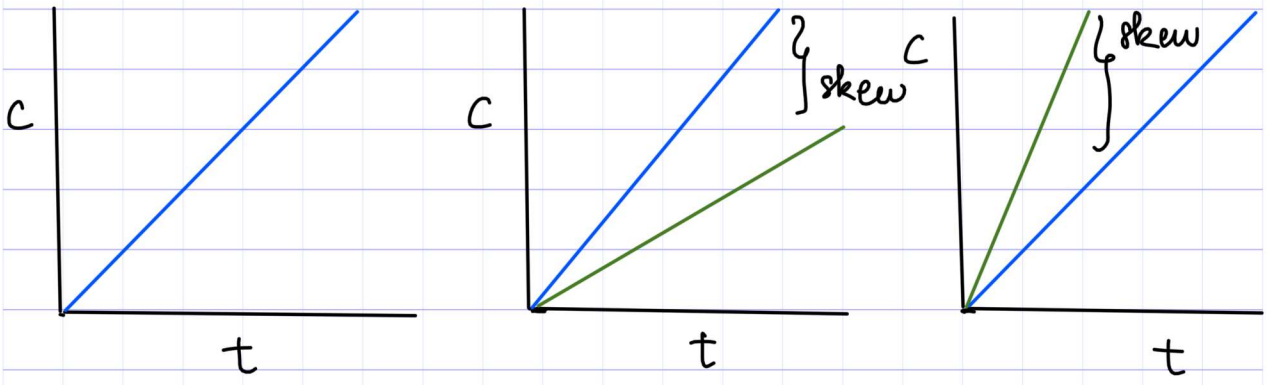
Delay is unpredictable.

∴ Cannot sync 2 physical clocks.

Clock Synchronization

Clock drift : clocks tick @ diff. rates.
quartz clock : 10^{-6} seconds
diff. of 1 sec / 11.6 days

clock skew : diff. between 2 clocks @ one point in time



$\frac{dC}{dt} = 1$

$\frac{dC}{dt} < 1$

make clock run faster till sync

$\frac{dC}{dt} > 1$

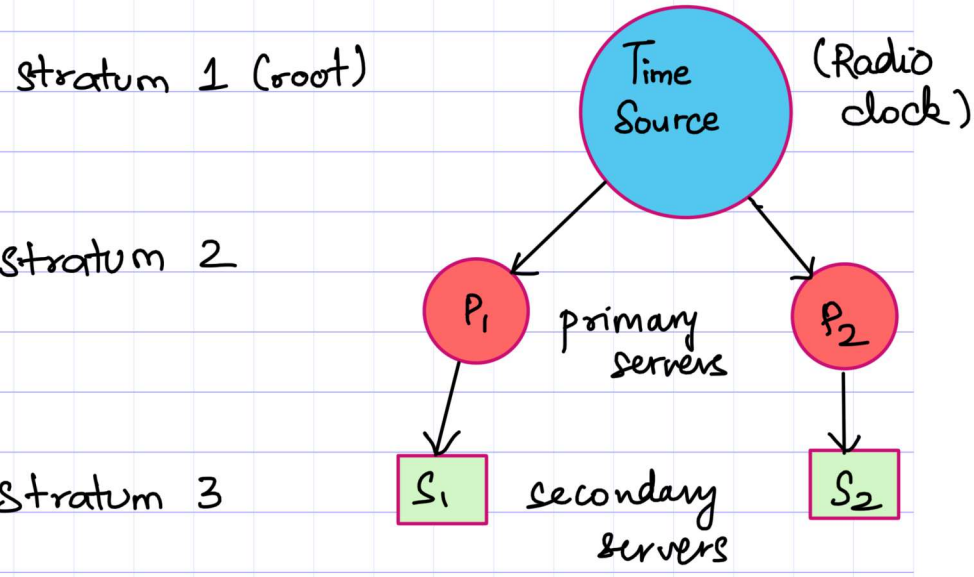
make clock run slower till sync

Mechanisms to sync :

Cristian's Algorithm ($T_{new} = T_{server} + RTT/2$)
Berkeley Algorithm (Fault tolerant avg. timestamping)
Network Time Protocol

Network Time Protocol :

Clients across Internet sync accurately to UTC despite message delays



Modes :

multicast mode - sends time to all other nodes.

procedure call mode - similar to Cristian's algorithm

symmetric mode - for master servers pair of servers exchange messages.

UDP protocol.