# TIME IN DISTRIBUTED SYSTEMS

Network Distributed Computing

Dr. Christina Thorpe

# EXERCISE

- Everyone record the time on their watch at the same moment.
- What are the results?
- What is the range of times?
- What is your conclusion?
- What factors have contributed to the distribution of values?
- How can we synchronise all the time nodes in the room?

# NEED FOR PRECISION TIME

- Stock market buy and sell orders
- Secure document timestamps (with cryptographic certification)
- Distributed network gaming and training
- Aviation traffic control and position reporting
- Multimedia synchronization for real-time teleconferencing
- Event synchronization and ordering
- Network monitoring, measurement and control

# COMPUTER CLOCKS

- Each node has its own private physical clock !
- Physical clocks are HW devices that count oscillations of a quartz
- After a specified number of oscillations, the clock increments a register, thereby adding one clock-tick to a counter the represents the passing of time: Hi(t).
- Resolution: Period between clock updates
- The OS maintains SW Clock by scaling and adding an offset to it: $Ci(t) = \alpha Hi(t) + \beta$.
- Ci(t) approximates the physical time t at process pi. Ci(t) may be implemented by a 64-bit word, representing nanoseconds that have elapsed at time t.
- Successive events can be distinguished if the clock resolutions is smaller that the time interval between the two events.

# DRIFT AND SKEW

- Computer clocks, like any other clocks tend not to be in perfect agreement !!
- Clock skew (offset): the difference between the times on two clocks |Ci(t) – Cj(t)|
- Clock drift : they count time at different rates
  - Ordinary quartz clocks drift by ~ 1sec in 11-12 days. (10^-6 secs/sec).
  - High precision quartz clocks drift rate is ~ 10^-7 or 10^-8 secs/sec
  - Differences in material, Temperature variation.

# SYNCHRONISATION

## External synchronization

• synchronization of process' clocks $C_i$ with an authoritative external source S.

• Let $\delta > 0$ be the synchronization bound and S be the source of UTC.

• Then $|S(t) - C_i(t)| < \delta$ *for* $i=1,2,\ldots,N$ and for all real times t.

• We say that clocks $C_i$ are ***accurate*** within the bound of $\delta$

## Internal synchronization

• synchronization of process' clocks $C_i$ with each other.

•Let $\delta > 0$ be the synchronization bound and $C_i$ and $C_j$ are clocks at processes $p_i$ and $p_j$, respectively.

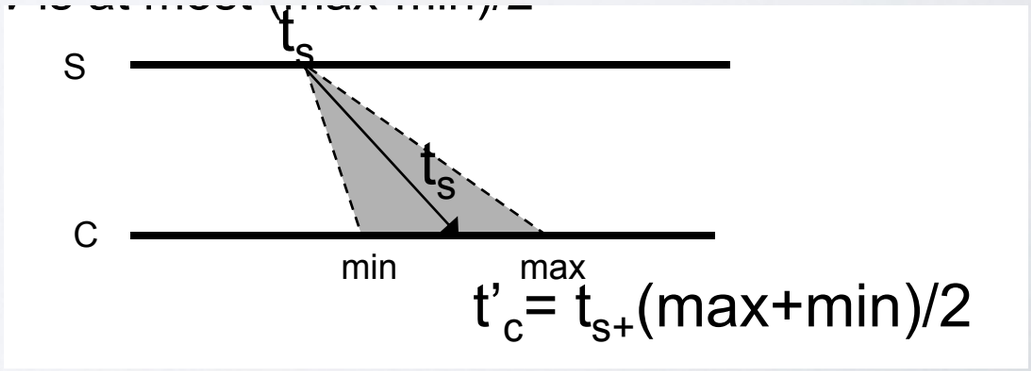
•Then $|C_i(t) - C_j(t)| < \delta$ *for* $i,j=1,2,\ldots,N$ and for all real times t.

•We say that clocks $C_i$, $C_j$ ***agree*** within the bound of $\delta$

Note that clocks that are internally synchronized are not necessarily externally synchronized. i.e., even though they agree with each other, the drift collectively from the external source of time.

# SYNCHRONISATION IN A SYNCHRONOUS SYSTEM

- In a synchronous system we have:
  - known upper (max) and lower (min) bound for communication delay,
  - known maximum clock drift,
  - known maximum time taken for each computational step.
- We synchronize by:
  - time server sends its local time t to a client,
  - Ideally, client sets clock to t_s +Ttran (Unknown!)
  - the client sets its local clock to t_s +(max + min)/2.
  - Skew is at most (max - min)/2

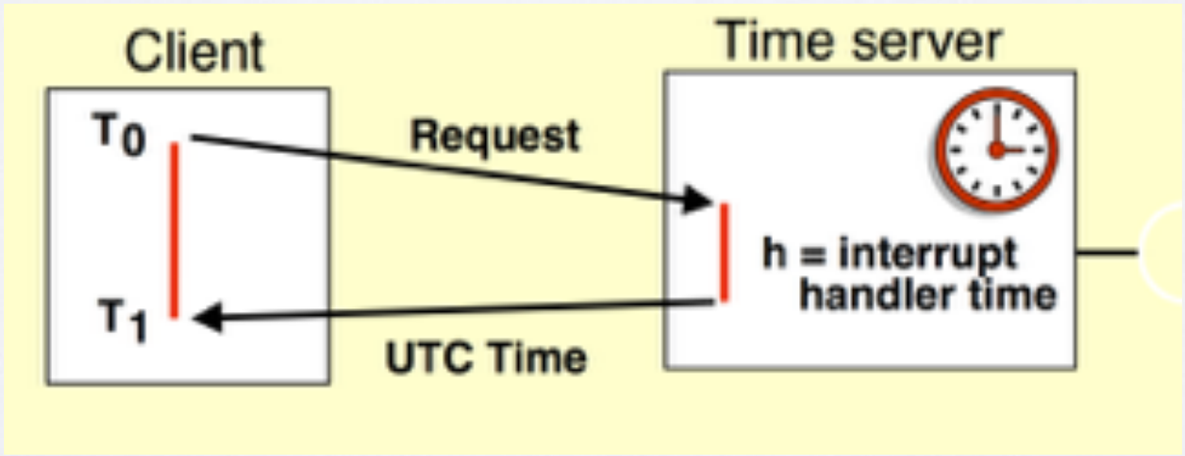


# SYNCHRONIZATION IN AN ASYNCHRONOUS SYSTEM

- Christians algorithm.
- The Berkeley algorithm.
- Network time protocol (NTP)

# CRISTIAN'S ALGORITHM

- A node is a time server TS (presumably with access to UTC). How can the other nodes be sync'ed?
- Periodically, at least every $\delta/2\rho$ seconds, each machine sends a message to the TS asking for the current time and the TS responds.



# CHRISTIANS ALGORITHM

- Client p sends request (mr) to time server S,
- S inserts its time t immediately before reply (mt) is returned,
- p measures how long it takes (Tround=T1 - T0) from mr is send to mt is received
- p sets its local clock to t+T /2.

# ACCURACY OF CHRISTIAN'S

- Assume min = minimal message delay
- t is in the interval [T0+min,T1-min]
- Uncertainty on t = Tround-2min
- Estimated Accuracy= Tround/2 - min

# PROPERTIES OF CHRISTIAN'S

- Monotonicity:
  - Jumps in time backward not permitted
  - Jumps forward may be confusing
  - Receiver adjusts clock rate $\alpha$: $C_i(t) = \alpha H_i(t) + \beta$.
- Improve precision
  - by taking several measurements and taking the smallest round trip
  - or use an average after throwing out the large values

# THE BERKLEY ALGORITHM

- Choose master co-ordinator which periodically polls slaves
  - Master estimates slaves' local time based on round-trip
  - Calculates average time of all, ignoring readings with exceptionally large propagation delay or clocks out of synch
  - Sends message to each slave indicating clock adjustment
  - Synchronisation feasible to within 20-25 msec for 15 computers, with drift rate of 2 x 10^-5 and max round trip propagation time of 10 msec.

**Accuracy**

- depends on the round-trip time

**Fault-tolerant average:**

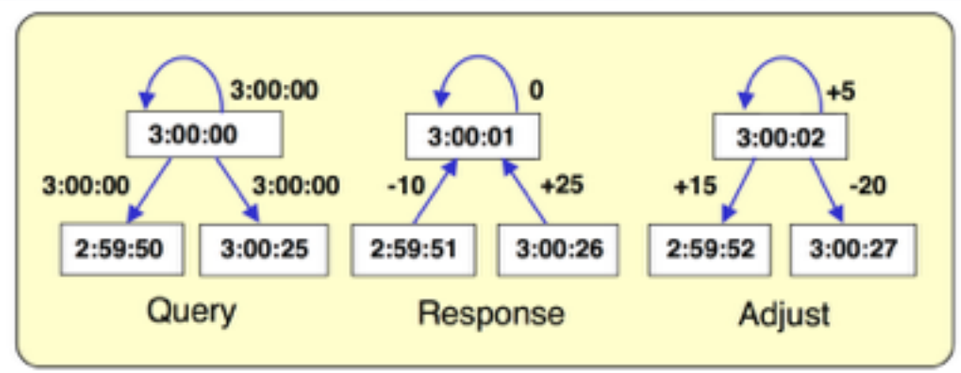
- eliminates readings of faulty clocks - probabilistically
- average over the subset of clocks that differ by up to a specified amount

**If master fails?**

- Elect a new one

# THE BERKELEY ALGORITHM

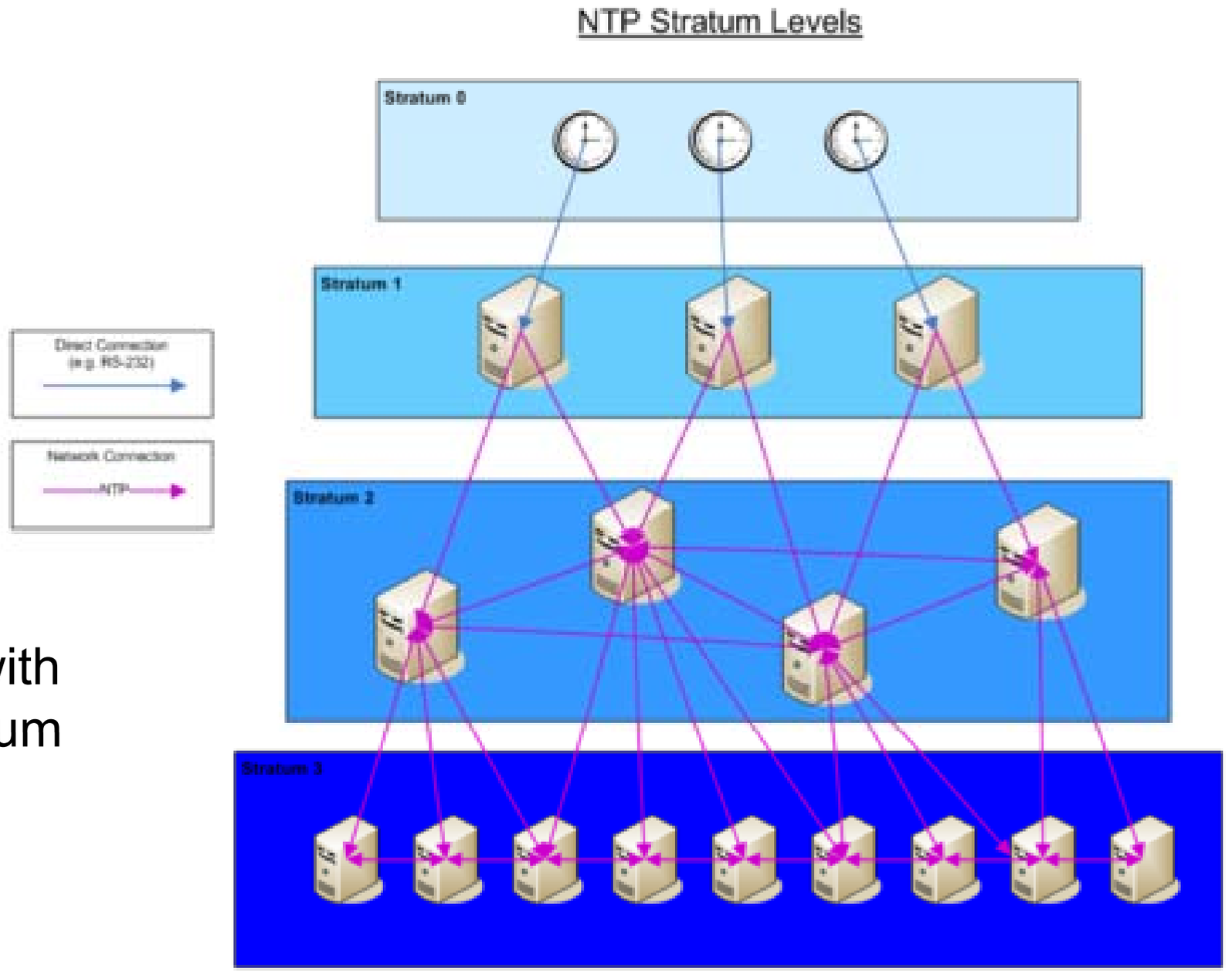
- The time daemon asks all the other machines for their clock values
- The machines answer their offset
- The time daemon tells each how to adjust its clocks



# NETWORK TIME PROTOCOL

- Synchronization of clients relative to UTC on an internet-wide scale
- Reliable, even in the presence of extensive loss of connectivity
- Allow frequent synchronization (relative to clock drift)
- Tolerant against disturbance
  - <1ms within LAN
  - 1-10 ms internet scale

# NTP STRATUM

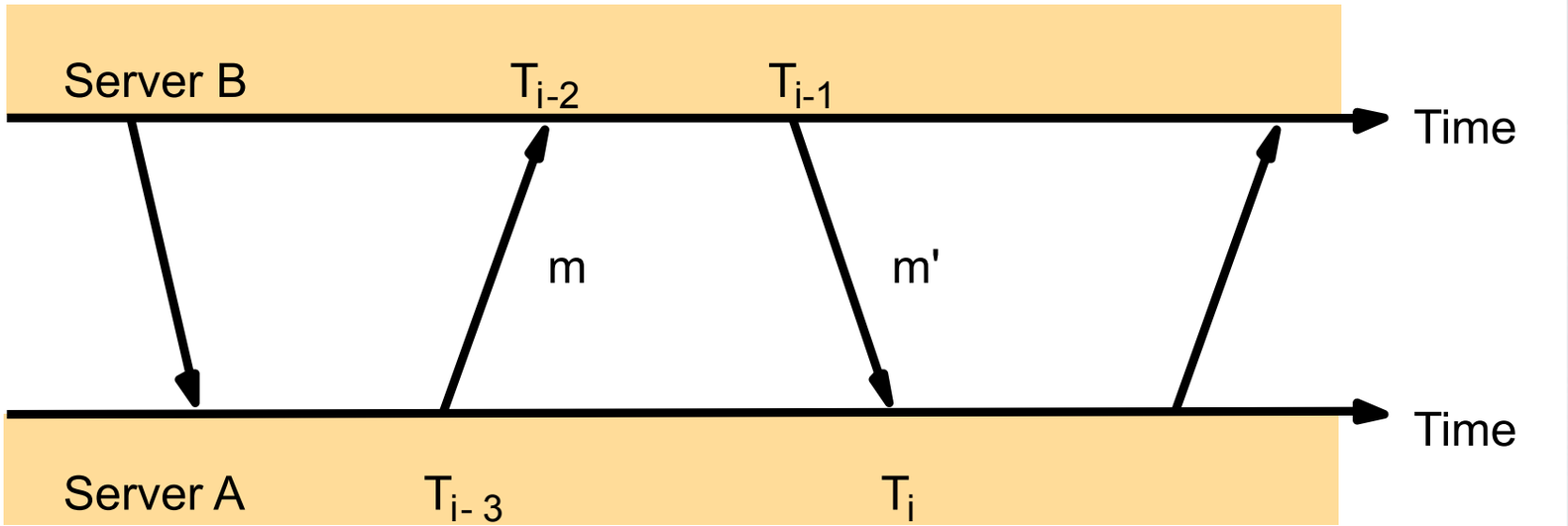



•Never synchronizewith servers at lover stratum

# NTP MODES

- Multicast (for quick LANs, low accuracy)
  - server periodically sends its actual time to its leaves in the LAN
- Procedure-call (medium accuracy)
  - server responds to requests with its actual timestamp
  - like Cristian's algorithm
- Symmetric mode (high accuracy)
  - used to synchronize between pairs of time servers
- In all cases, the UDP is used

# MESSAGES EXCHANGED BETWEEN A PAIR OF NTP PEERS




$o_i=(T_{i-2}-T_{i-3}+T_{i-1}-T_i)/2$ $d_i=T_{i-2}-T_{i-3}+T_i-T_{i-1}$

- Exchanges local timestamps to estimate offset oi and delay di
- NTP server filters pairs <oi,di>, saves 8 latest
- Use the oi, with smallest di.(the smaller delay the better accuracy)

# THEORETICAL FOUNDATION

Inherent characteristic of a distributed system:

– Absence of a global clock:

– Absence of 100% accurately synchronized clocks

– Impact: Due to the absence of global clock, it is difficult to reason about the temporal order of events in distributed system, e.g. scheduling events is more difficult.

# LOGICAL CLOCKS IN A DS

What is important is usually not when things happened but in what order they happened so the integer counter works well in a centralized system.

- However, in a DS, each system has its own logical clock, and you can run into problems if one "clock" gets ahead of others. (like with physical clocks)

- RELIABLE WAY OF ORDERING EVENTS IS REQUIRED

- We need a rule to synchronize the logical clocks.