Synchronization of Physical Clocks

Alessio Bechini Dept. of Information Engineering, Univ. of Pisa

a.bechini@ing.unipi.it

Outline

Getting
Computer Clocks
Synchronized

- Generalities
- Cristian's algorithm
- Berkeley algorithm
- Network Time Protocol

Generalities

Time References

Every node has an internal clock, whose value has to be kept as close as possible to a reference time.

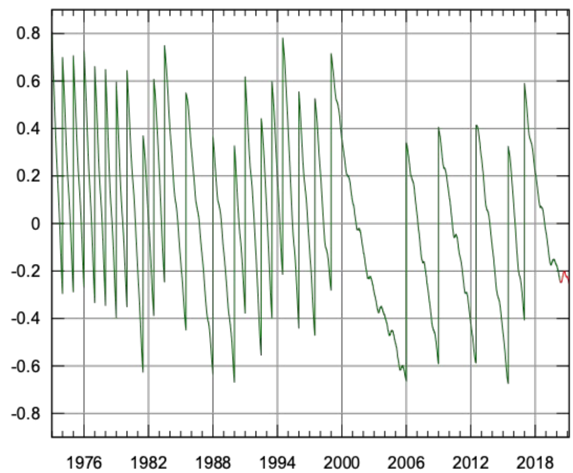
- **UTC** - coordinated universal time, primary reference for the scientific community, and based on International Atomic Time
- **UT1** - successor of GMT, “solar-based” reference time
- **Unix (POSIX) time** - nr. of seconds elapsed since 00:00:00 (UTC), Thursday, 1 January 1970, minus in-between *leap seconds*

Leap Seconds?

Earth's rotation slowdown and irregularities make UT1 deviate from UTC.

Thus, corrections are needed.

Chart aside: UT1-UTC vs UTC



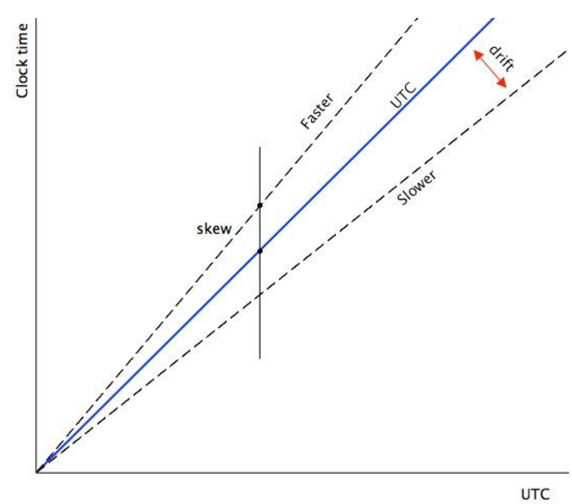
From Wikipedia

Non-Aligned Clocks

Skew - difference in instantaneous reads of two clocks

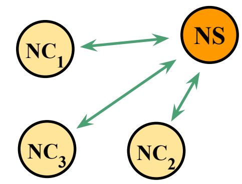
Drift - difference in the rate of clocks.

Keeping two clocks synchronized means imposing an upper bound D on any of their instantaneous reads.



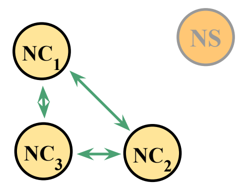
External vs Internal Synchronization

External: w.r.t. a trusted source of reference time $S(t)$, i.e.



$$| S(t) - C_i(t) | < D \quad \forall\ i$$

Internal: agreement within a group of nodes, i.e.



$$| C_i(t) - C_j(t) | < D \quad \forall\ i, j$$

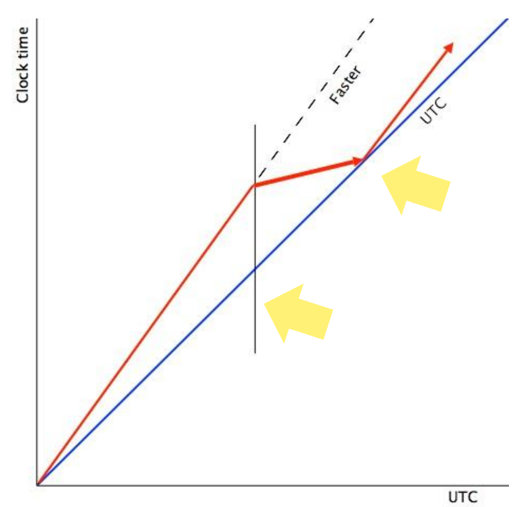
If a system is externally synchronized with an accuracy D, then its internal clocks agree within a bound of $2D$

Resetting a Clock

Important: clock *monotonicity* must be assured!

No way to set back a clock: instead, its pace should be slowed down up to the point the correct value is reached.

Typical example: the "make" compilation utility.



Cristian's Algorithm

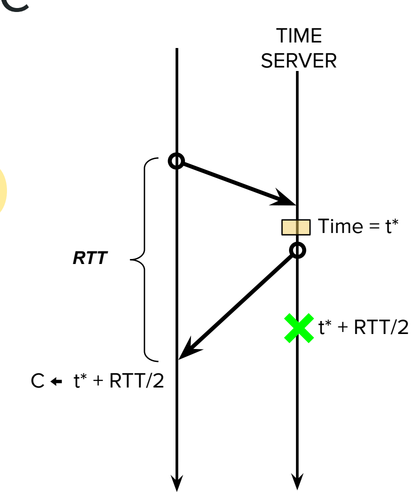
Idea: Exploit Round Trip Time

RTT can be measured by the client.

If RTT << 1, assuming req/reply latencies to be the same does not yield an excessive error, thus:

$$C \simeq t^* + RTT/2$$

In practice, several tries can be done, so to possibly get lower RTTs.



Internal Synchronization:
The Berkeley Algorithm, and Beyond

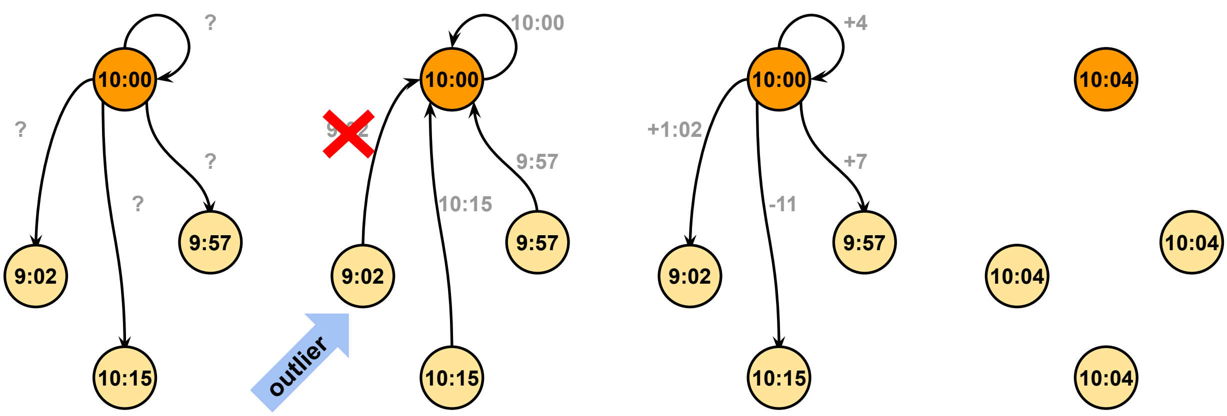
A Centralized Solution

Developed for groups of UNIX computers.

One process/node acts as master (*time daemon*); Successive steps:

- The time daemon asks all the others for their clock values.
- Each node answers back its actual time value;
  the master annotate each RTT as well.
- The time daemon computes a *proper* average.
- The time daemon sends each node the clock correction value
  (why not broadcasting the time?)

Berkeley Algorithm: an Illustration



Decentralized Averaging Algorithm

Each node has a daemon, with approximated UTC.

- Periodically, each node broadcasts its own time.
- On each node, the new time value is obtained by averaging the local time and the received values.

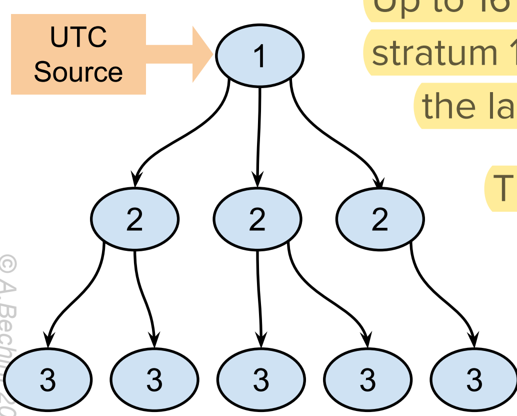
NTP - Network Time Protocol

NTP - Overview

To enable clients across Internet to get synchronized with UTC.

- Scalable to large networks
- Statistical techniques to filter data
- Authentication against interference
- Hierarchy of time servers, spread across the Internet
- Messages sent over UDP

Time Servers' Hierarchy - Strata



Up to 16 strata:

stratum 1 is connected to an UTC source;

the last one is "not synchronized".

Time servers synchronize in 3 modes:

Multicast mode

Procedure Call mode

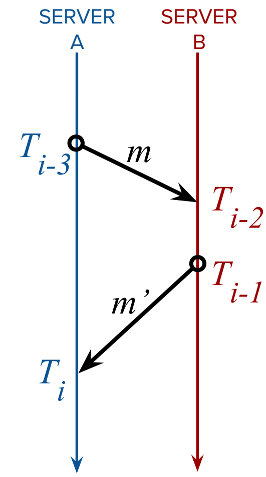
Symmetric mode

NTP Modes

- Multicast: 1+ servers multicast the time to the other nodes; suitable for LANs
- Procedure Call: more accurate because of latency compensation, using an approach based on Cristian's algorithm
- Symmetric: the most accurate and expensive, used between servers in the upper (most precise) strata; Pairs of servers exchange messages carrying the timestamp for the involved events

Symmetric Mode

From the exchanged messages, a process must be able to get an estimate o_i of the actual offset o to fix its clock



$$\begin{cases} T_{i-2} = T_{i-3} + t_m + o \\ T_i = T_{i-1} + t_{m'} - o \end{cases} \qquad \begin{cases} o = T_{i-2} - T_{i-3} - t_m \\ o = T_{i-1} - T_i + t_{m'} \end{cases}$$

$$RTT = t_m + t_{m'} = \Delta T_A - \Delta T_B = (T_i - T_{i-3}) - (T_{i-1} - T_{i-2})$$

$$o = ½\,(T_{i-2} - T_{i-3} + T_{i-1} - T_i - t_m + t_{m'}) = o_i + ½\,(t_{m'} - t_m)$$

$$o_i - ½\,RTT \leq o \leq o_i + ½\,RTT$$

Symmetric Mode: How to Fix Time

Estimated offset: $o_i = ½\,(T_{i-2} - T_{i-3} + T_{i-1} - T_i)$

Accuracy (upper bound): RTT

- To fix the time, several pairs $\langle o_i, RTT \rangle$ are collected, and the most accurate value is used.