Distributed Systems

Distributed Time

Prof. Dr. Oliver Hahm

Frankfurt University of Applied Sciences
Faculty 2: Computer Science and Engineering
oliver.hahm@fb2.fra-uas.de
https://teaching.dahahm.de

02.06.2023

Motivation

Problems of unsynchronized computer clocks

- Timestamps of files (make)
- Time-triggered execution of tasks ($\rightarrow$ cron)
- TDMA based medium access control

Goal

- Establishing a systemwide **time** in distributed systems
- **Synchronicity** with real (external) time
- Synchronization of computer **clocks**

Applications

- Correct functioning of **timebased** local and distributed **applications**
- Correct **ordering of events** in distributed systems
- **Performance measurement** in distributed systems
- Distributed **real-time systems** including the synchronicity with the real global time

Agenda

- Definition of Time
- Computer Clocks
- Synchronization Protocols
- Logical Clocks

Agenda

- Definition of Time
- Computer Clocks
- Synchronization Protocols
- Logical Clocks

What is Time?

Could you define what time is?

Time in the Course of History

Astronomical Time

- Based on the **uniform movement of celestial bodies** and their observation

Time in the Course of History

Astronomical Time

- Based on the **uniform movement of celestial bodies** and their observation
- **Apparent time**
  - Average duration of the rotation of the earth
  - Mean solar day: Zenith to Zenith (until 1956)
  - 1 second $= \frac{1}{24 \cdot 60 \cdot 60}$ solar day
  - Little stability (Deceleration of the rotation of the earth, fluctuation because of mass displacement)

Time in the Course of History

Astronomical Time

- Based on the **uniform movement of celestial bodies** and their observation
- **Apparent time**
  - Average duration of the rotation of the earth
  - Mean solar day: Zenith to Zenith (until 1956)
  - 1 second $= \frac{1}{24 \cdot 60 \cdot 60}$ solar day
  - Little stability (Deceleration of the rotation of the earth, fluctuation because of mass displacement)
- **Sidereal time** ($\rightarrow$ *stardate* in *Star Trek*)
  - Average duration of the period of the earth around the sun
  - 1 second $= \frac{1}{31.556.925{,}9747}$ part of the tropical year 1900 (since 1957)

Physical Time

Based on (periodical) physical processes

Typical examples:

- Candle clock (burning of wax)
- Pendulum clock (accuracy at best: 10^{-7})
- Quartz clock (accuracy at best: 10^{-9}, typical: $10^{-5} \ldots 10^{-6}$)

Physical Time

Based on (periodical) physical processes

Typical examples:

- Candle clock (burning of wax)
- Pendulum clock (accuracy at best: 10^{-7})
- Quartz clock (accuracy at best: 10^{-9}, typical: $10^{-5} \ldots 10^{-6}$)

Atomic Clock

- Definition in the International System of Units (SI) (since 1967): *"1 second is the duration of 9192631770 periods of radiation produced by transition of an electron between two hyperfine levels of a Cesium-133 atom."*
- Cesium-133 clock, accuracy at best: 10^{-14}, typical: 10^{-13}
  $\rightarrow < 1\mu s$ per year
- Cesium fountain, accuracy: $< 10^{-15}$

Physikalisch-Technische Bundesanstalt (PTB)

- In Braunschweig
- Operation of multiple atomic clocks (CS1-CS4, CSF1)
- Responsibility for the **legal time** in Germany (since 1978)
- Operation of distribution services

https://www.meinberg.de/images/xatomuhr.jpg.pagespeed.ic.3l8wJGqj54.jpg

Which time is it?

Now we have an accurate definition for the period called second, but...

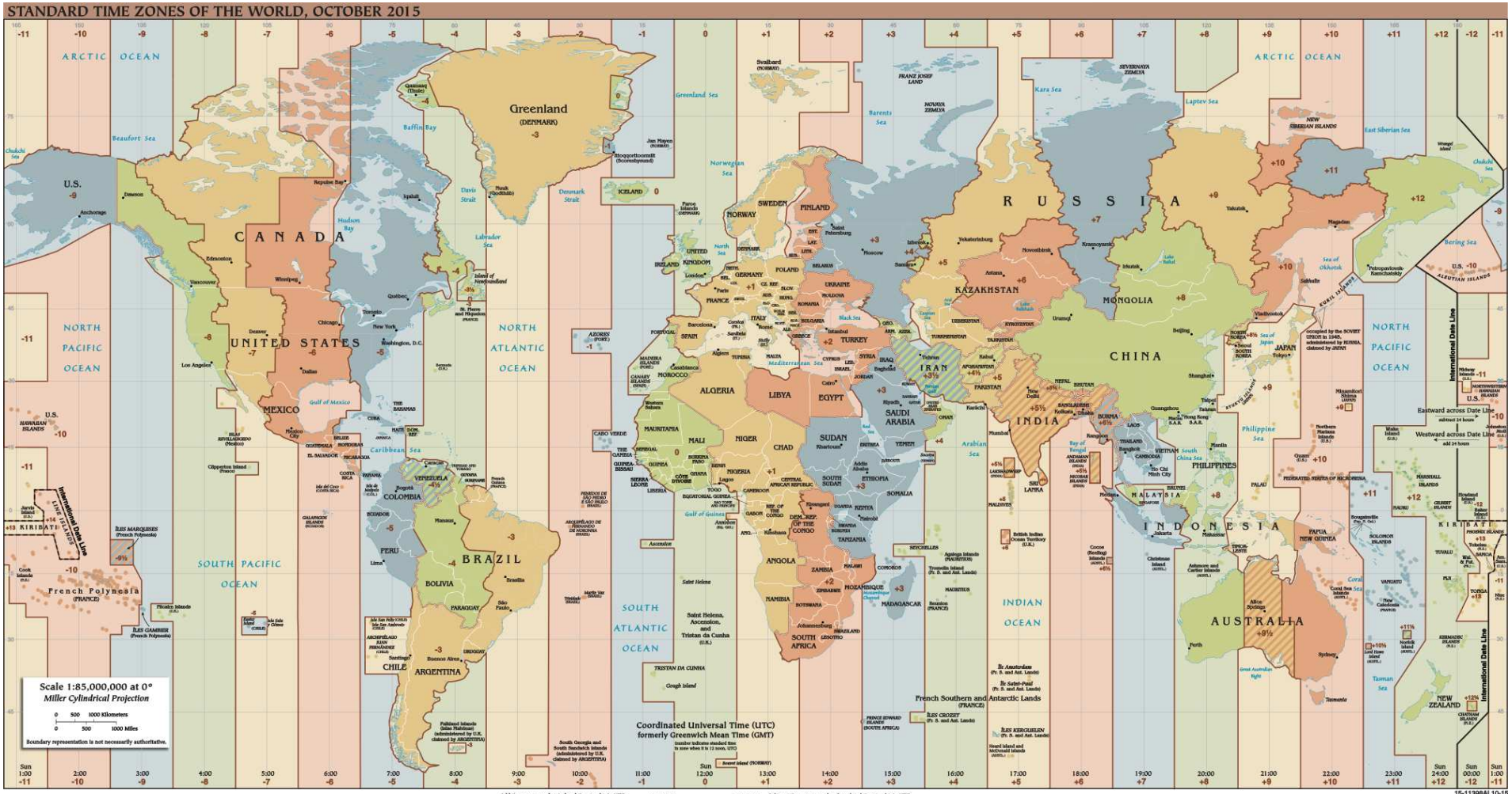
How do we know which time it is?

Time Systems

GMT: Greenwich Mean Time

- Local time (real and average), common until ca. 1880
- Problem for the railway schedules
- Greenwich Mean Time legal standard time in England since 1880
- Since 01.06.1891: German and Austro-Hungarian railway administration introduce the time at the 15th longitude as mitteleuropäische Eisenbahn-Zeit (M. E. Z.).
- *Deutsches Reich*: legal time since 01.04.1893 is defined as *"the average apparent time at the 15th longitude eastern of Greenwich"*
- Meridian Conference at Washington 1884 defines Greenwich as prime meridian and introduces time zones → apparent time
- Since 01.01.1925: Begin of the day at midnight (for astronoms until then at midday)

Time Zones



https://commons.wikimedia.org/wiki/File:Standard_Time_Zones_of_the_World_(October_2015).svg

UT, TAI, and UTC

UT: Universal Time

- World time is derived from sidereal time (since 1957) at prime meridian
- UT1: Correction of polar motion

UT, TAI, and UTC

UT: Universal Time

- World time is derived from sidereal time (since 1957) at prime meridian
- UT1: Correction of polar motion

TAI: Temps Atomique International

- Mean atomic time since 01.01.1958
- Operation of ca. 250 atomic clocks worldwide
- Coordinated on a global scale by the Bureau International de l'Heure (BIH)

UT, TAI, and UTC

UT: Universal Time

- World time is derived from sidereal time (since 1957) at prime meridian
- UT1: Correction of polar motion

TAI: Temps Atomique International

- Mean atomic time since 01.01.1958
- Operation of ca. 250 atomic clocks worldwide
- Coordinated on a global scale by the Bureau International de l'Heure (BIH)

UTC: Coordinated Universal Time

- Current standard for time (since 1972)
- Based on TAI, but adjusted to UT1 by leap seconds on a difference of more than 900 ms
- **Deviation**: 1 second in $300,000$ years

Historical Time Distribution Services

- *One O'Clock Gun*

https://upload.wikimedia.org/wikipedia/commons/d/d9/One_OClock_Gun.JPG

  - **Problem:**
    Transmit time of sound

- Time ball

https://www.flickr.com/photos/flissphil/2530153597/

  - **Problem:**
    Line of sight required

Modern Time Distribution Services

Long-wave radio transmitter

- e.g., in Germany: DCF77 (77.5 kHz, Frankfurt/Mainflingen)
- Based on atomic clock CS-2 at PTB
- Second intervals
- Modulated full BCD time code (58 bit) every minute
- Accuracy
  - $2 \cdot 10^{-13}$ averaged over 100 days
  - 1-10 ms per second (atmospheric noise)

GOES Satellite System

- Geostationary Operational Environment Satellite
- Accuracy ca. 0.5 ms
- Service of the NOAA from 1974 to 2004

GPS-based Time Distribution

- Global Positioning System, original primary military
- 24 satellites, orbital period 12 h, at least 4 are always apparent
- Cesium clocks on board
- Synchronization against clocks of other satellites via ground station with an accuracy of ± 5 ns
- Position determination through different **signal runtimes**
- Artificial inaccuracies in times of crisis
- Differential GPS makes additionally use of ground stations with a known location (geodesy)

GPS based clock

- GPS signal as reference of a PLL circuit
- High-precision second pulse (pps = pulse per second)
- Typical accuracy: ca. $1\mu s$

Other Satellite-based Time Distribution Services

Galileo satellite system (operated by EU/ESA)

- European system, compatible with GPS (GPS III)
- Up to 30 satellites
  - Each of it with two atomic clocks
  - Sending timestamps and location data
  - Global coverage
- Services
  - Free service for locating, navigation, and clock synchronization (accuracy ca. 4 m horizontal, 8 m vertical)
  - Commercial service (accuracy 1 m, movement 0.2 m/sec) (surveying, network synchronization, fleet management)
  - Safety-of-Life service, (safety critical applications aviation, shipping, and railways)
  - Service "'of public interest"', (Signal with very high accuracy, quality, reliability, and integrity for sovereign applications)

Other Systems: GLONASS (Russia) and Beidou (China)

Terms

Reference time

- Approximation of the real physical time

Deviation and accuracy

- Absolute or relative difference to a reference time

Precision and resolution

- Smallest period between two successive displayable points in time

Stability

- Frequency variation of a clock (Often given as ppm)

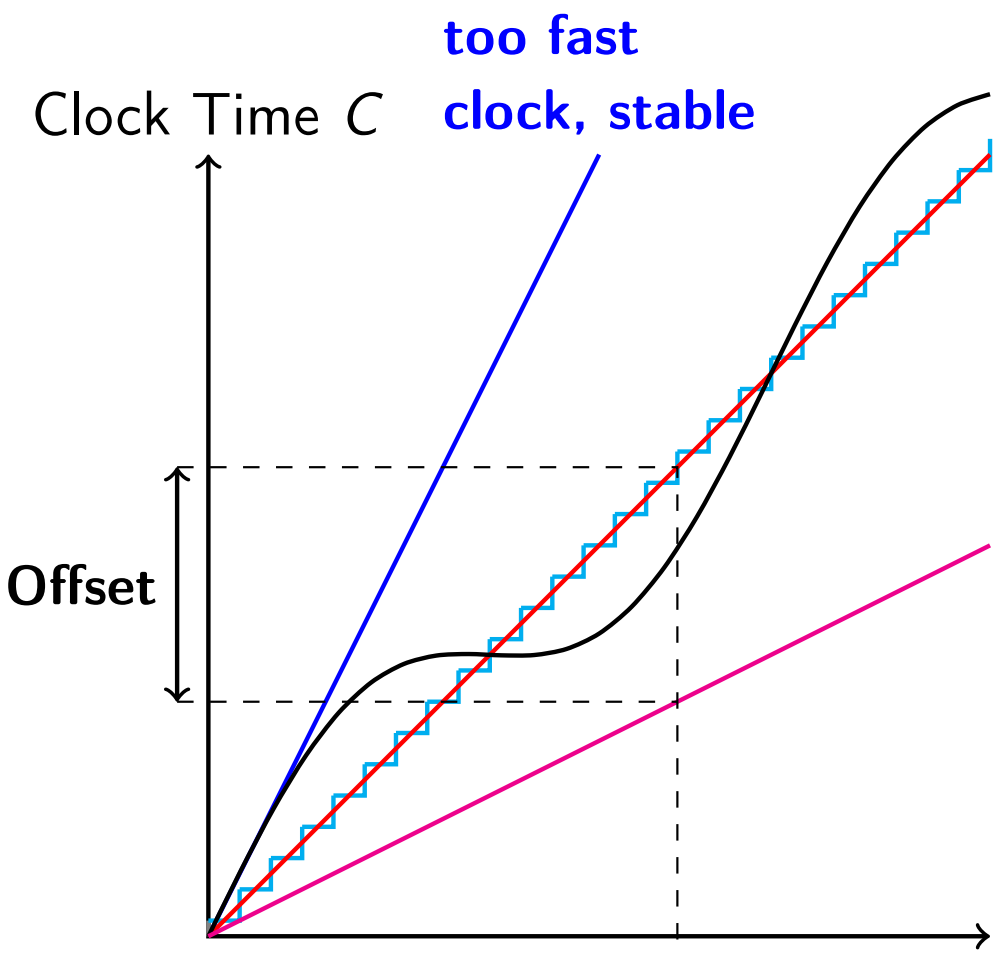
Offset

- Time difference between two clocks

Drift

- Frequency difference between two clocks

Illustration



Clock with high accuracy, but low stability

ideal clock: $\frac{dC}{dt} = 1$

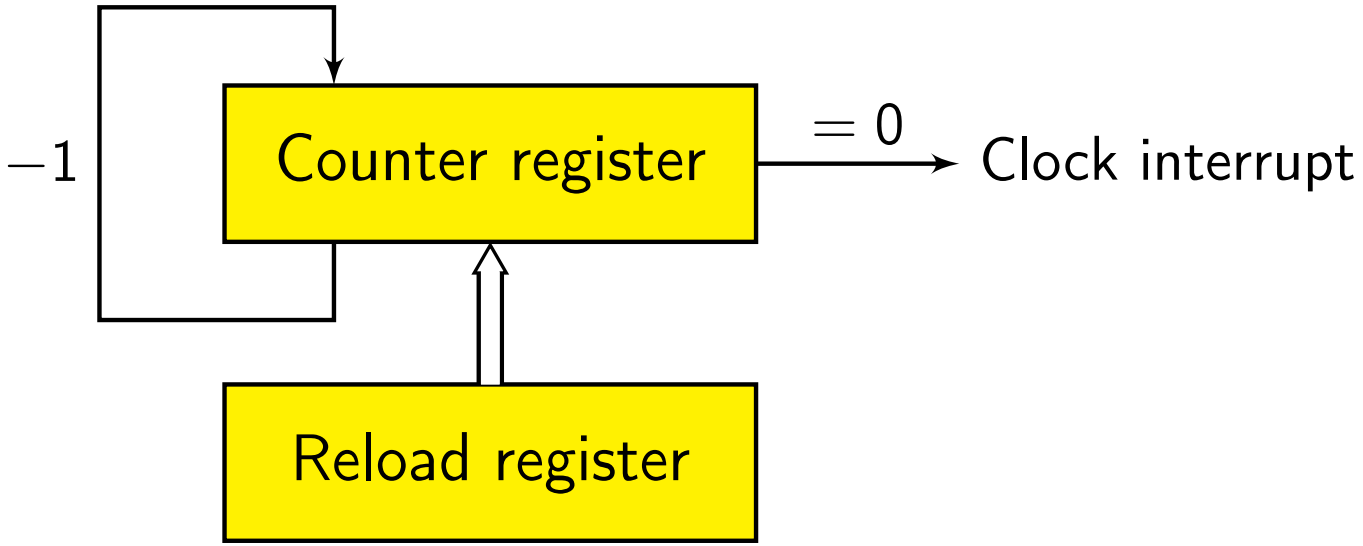
Clock with high accuracy and stability, but low precision

too slow clock, stable

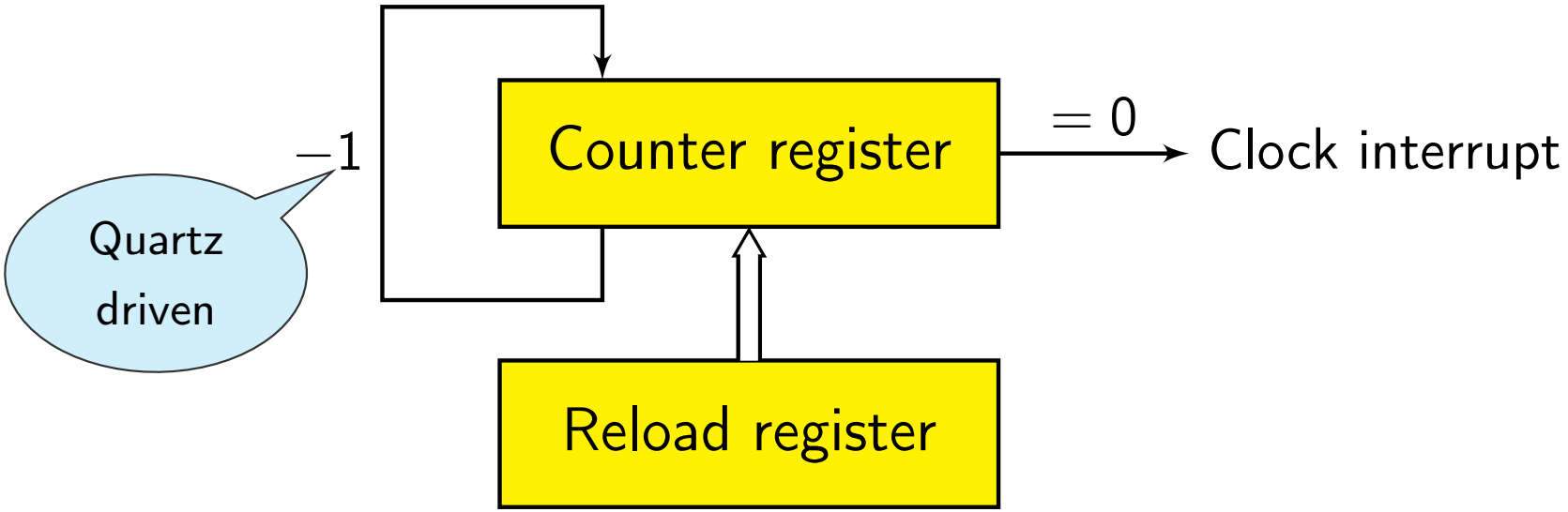
Agenda

- Definition of Time
- Computer Clocks
- Synchronization Protocols
- Logical Clocks

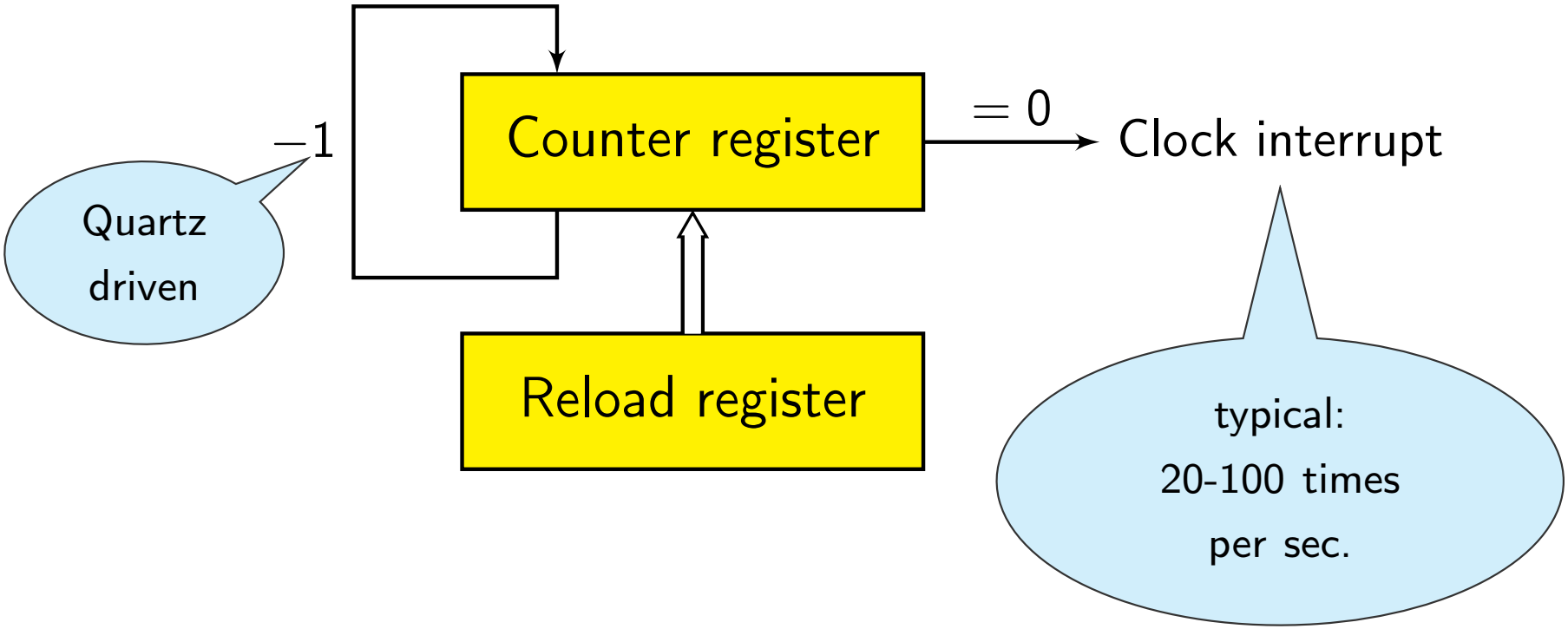
Hardware of a Computer Clock



Hardware of a Computer Clock



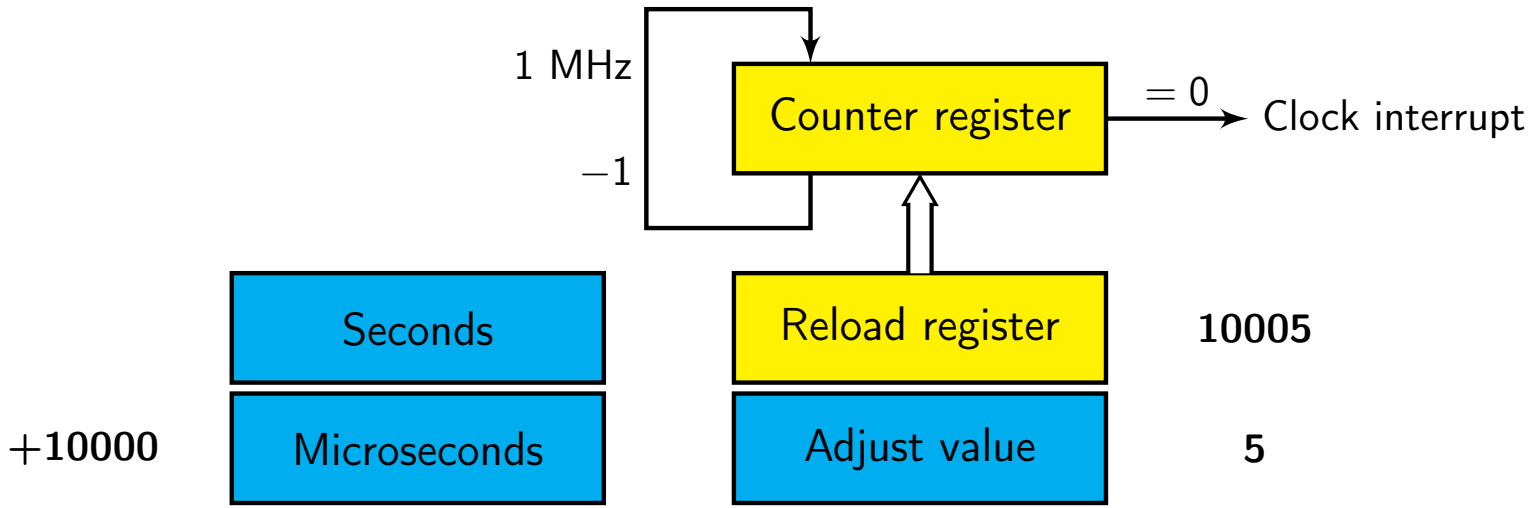
Hardware of a Computer Clock



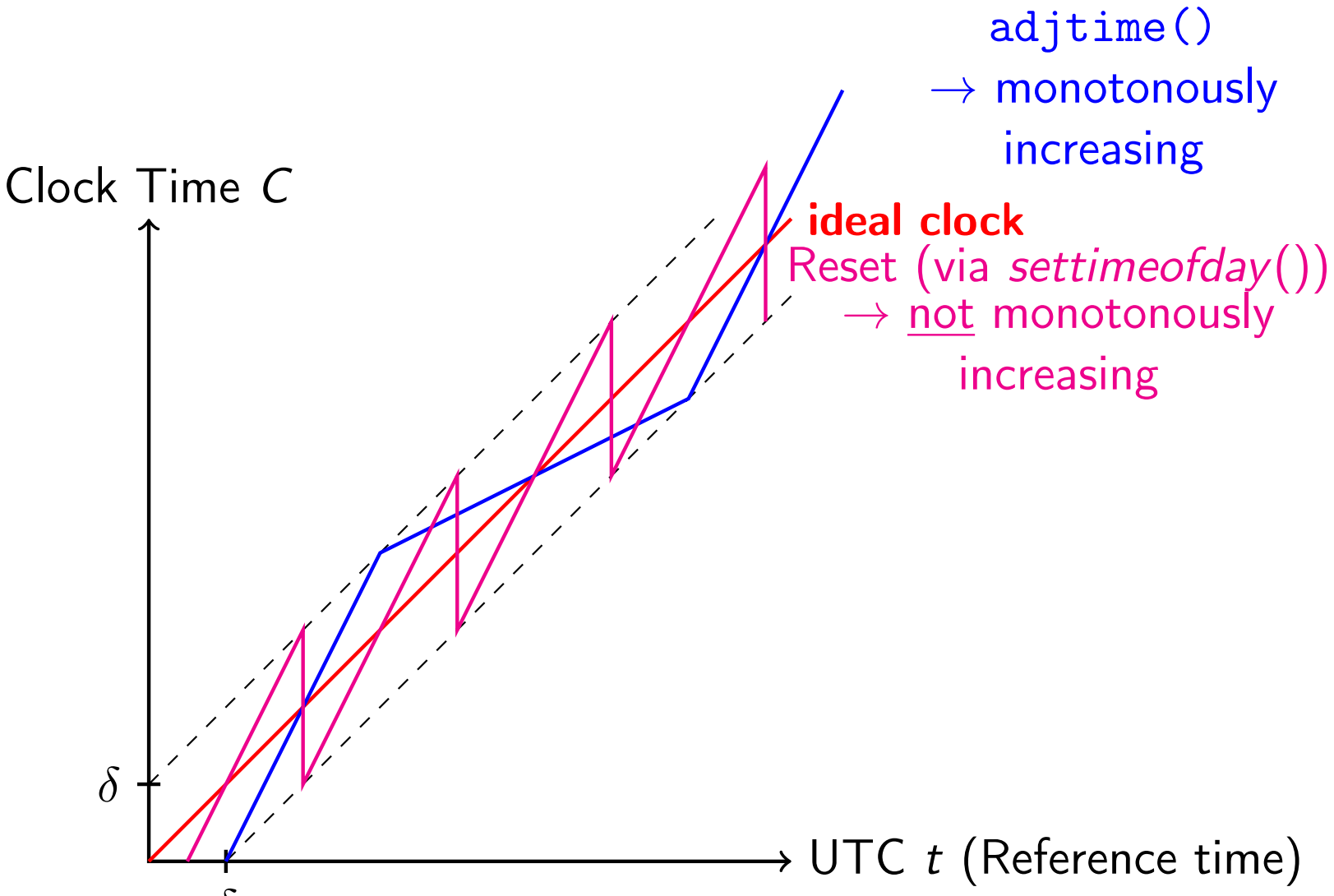
Operating System Clocks

Example: UNIX

- two 32 bit (or 64 bit) integer variables
  - Number of seconds since 1.1.1970 ($\rightarrow$ *Year 2038 problem*)
  - Number of μs (or ns) in the current second
- Typical 100 interrupts/s
- On interrupt the variables are increased by the nominal number of μs
- Adjust value for compensation of the drift of the quartz
- System services: `settimeofday`, `adjtime`



Correction Principle



Reference Time Sources

DCF77 clock for relaxed system time requirements

- Accuracy typical: $\pm$ 2 ms

GPS clock for high system time requirements

- Accuracy typical: $\pm$ 250 ns

Atomic clock

- Rubidium/Cesium sources
- Special approval necessary
- Rack mounting
- In part only for military usage

Computer interface

- Generation of pulse-per-second (pps) signals as interrupts
- Encoded timecode signals,
  e.g., IRIG standard (Inter Range Instrumentation Group)

Accurate Local OS Clocks

Use of a external reference time source
Linux kernel with Nano-Kernel-Patch

- Increase of the system clock resolution to $1ns$ (instead of μs)
- Default in newer Linux kernels
- Use of pps signals as reference time source via interrupts
- Correction of system clock according to reference time of the hardware clock
- Variance of interrupt latency effects the accuracy
- Multiple external time sources for a single computer are possible to further increase the accuracy
- Accuracy: typical < 1 μs

Example: David L. Mills' Uhren (Uni Delaware)

http://doc.ntp.org/4.1.2/refclock.htm

- Spectracom 8170 WWVB Receiver
- Spectracom 8183 GPS Receiver
- Hewlett Packard 105A Quartz Frequency Standard
- Hewlett Packard 5061A Cesium Beam Frequency Standard
- NTP primary time server **rackety** and **pogo** (elsewhere)

Commercial Time Server

Time Server

- Dedicated LAN network node for time synchronization
- Internal or external reference time source
- Support for standard protocols (NTP, SNTP, PTP/IEEE 1588)

Producs of many variants

- Meinberg (D)
- IPCAS (D)
- Galleon (UK)
- ELPROMA (NL)
- Time Tools (UK)

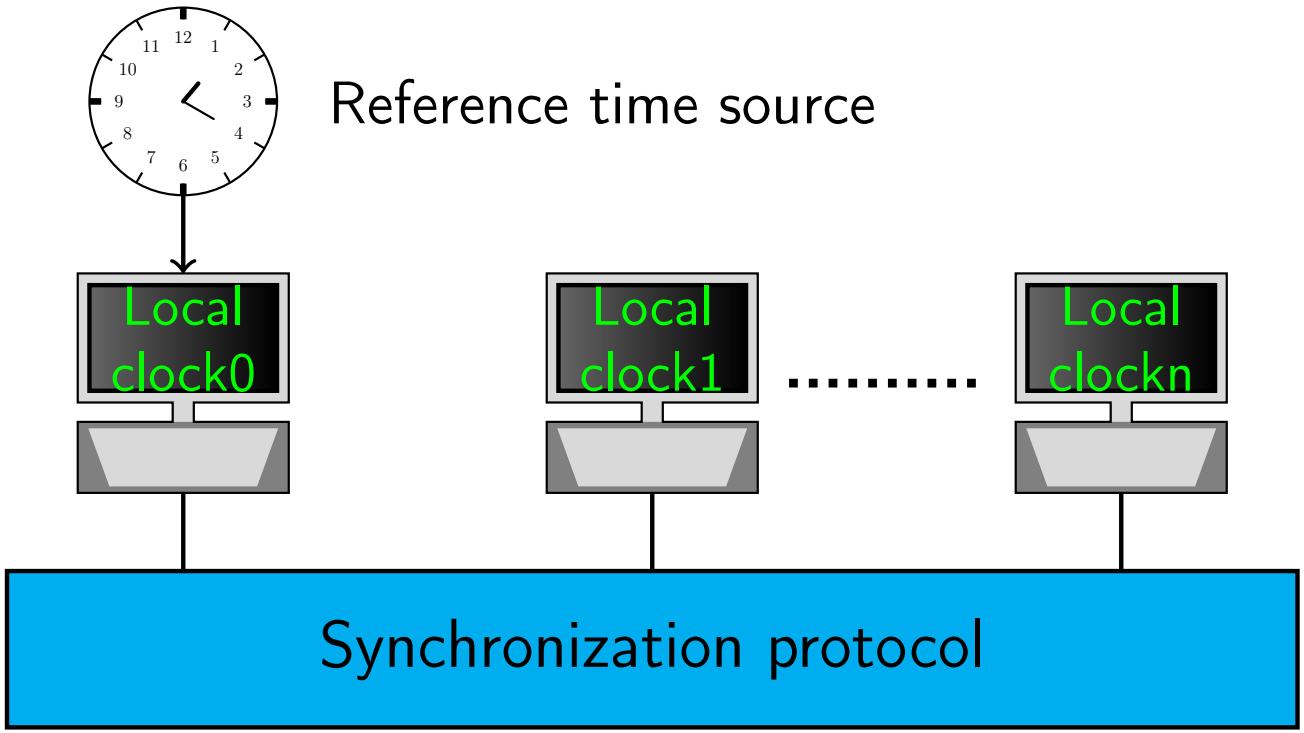
Agenda

- Definition of Time
- Computer Clocks
- Synchronization Protocols
- Logical Clocks

Synchronization Protocols

Construction of a distributed time basis for computer systems

- UTC-based external reference time source
- Local clocks in computer systems
- **Synchronization protocol**



Problems

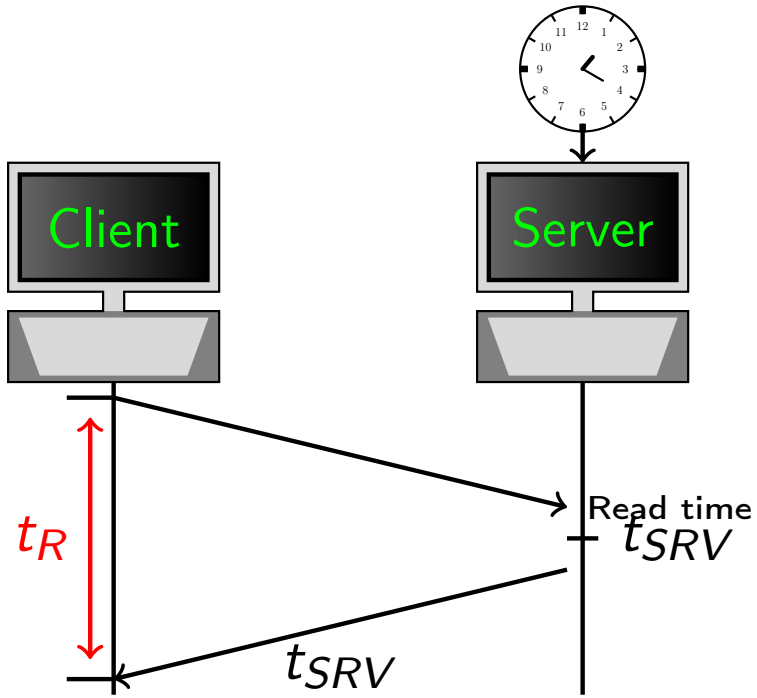
- Message delay in a network is not deterministic
- Processing time for protocol messages is not deterministic

$\Rightarrow$ **No exact synchronization possible**

Cristian's Algorithm (1989)

- Passive time server (as reference time source)
- Periodical polling of time by clients
- Measure average round trip time (including processing time on server)
- **Drawbacks:**
  - "Going backward" of a clock is possible
  - Jitter in message runtimes

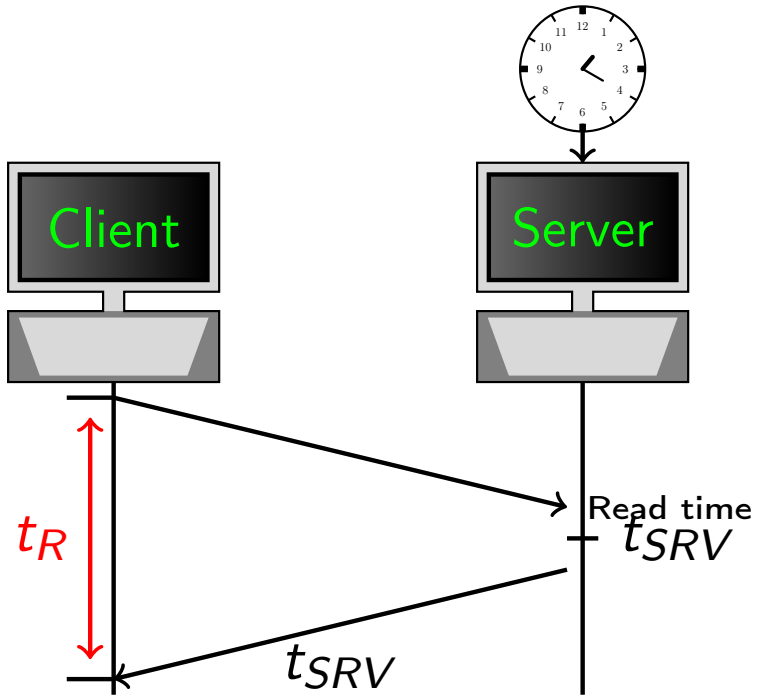
Set: $t_{local} = t_{SRV} + \frac{t_R}{2}$



Cristian's Algorithm (1989)

- Passive time server (as reference time source)
- Periodical polling of time by clients
- Measure average round trip time (including processing time on server)
- **Drawbacks:**
  - "Going backward" of a clock is possible
  - Jitter in message runtimes

Set: $t_{local} = t_{SRV} + \frac{t_R}{2}$



Round trip time (RTT) – measured with local clock of the client

Time Synchronization Protocol (TSP)

- Berkeley UNIX `timed`
- Based on ICMP/IP
- Establishes average network time on all sites
- Client/server algorithm
  - Active server: Polls current time from all nodes and calculates average
  - Distributes offset to each client
- Uses `settimeofday()` and `adjtime()` on the nodes
- **Severe drawbacks**
  - "Going backward" of a clock is possible
  - No compensation of variation of message runtimes
  - No error estimation
  - Bad scalability
- Variant: Server with external reference time source distributes current time instead of calculated average

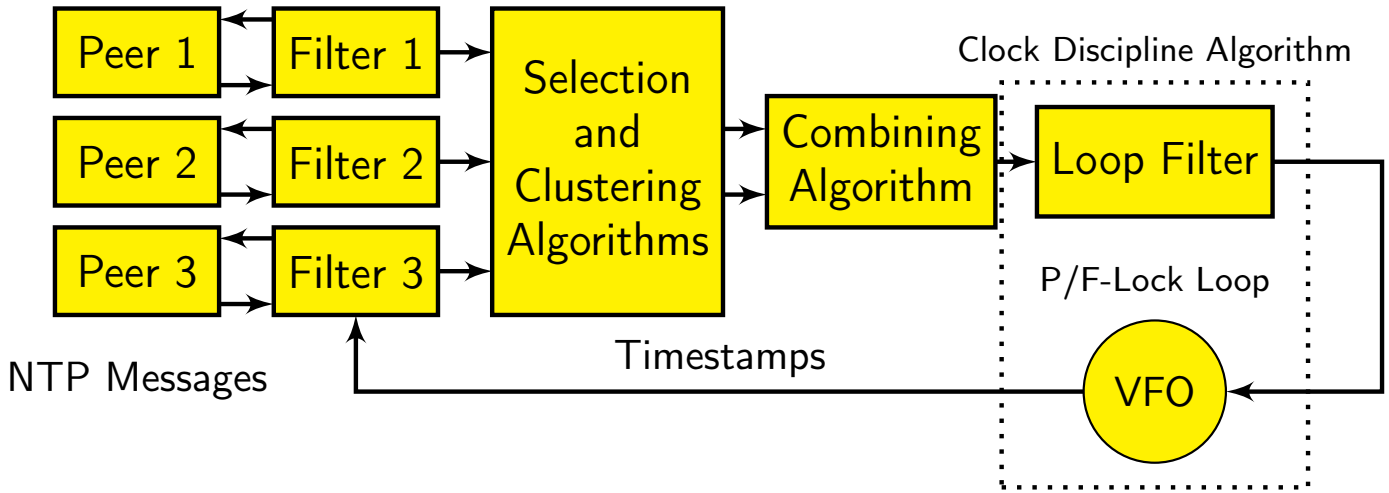
Network Time Protocol (NTP)

- Development primarily driven by D. Mills (Univ. of Delaware)
- **Reference implementation:** `http://www.ntp.org`
- **Goals**:
  - High accuracy
  - Handle varying message runtimes
  - Handle computer failures by using multiple time servers (peers)
  - Sorting out of obviously useless time sources ($\rightarrow$ false ticker)
  - Limited authentication and encryption
  - High scalability
- Accuracy:
  - In LAN < 1 ms, Internet $<$ *ca.*10 ms

Today's Internet standard

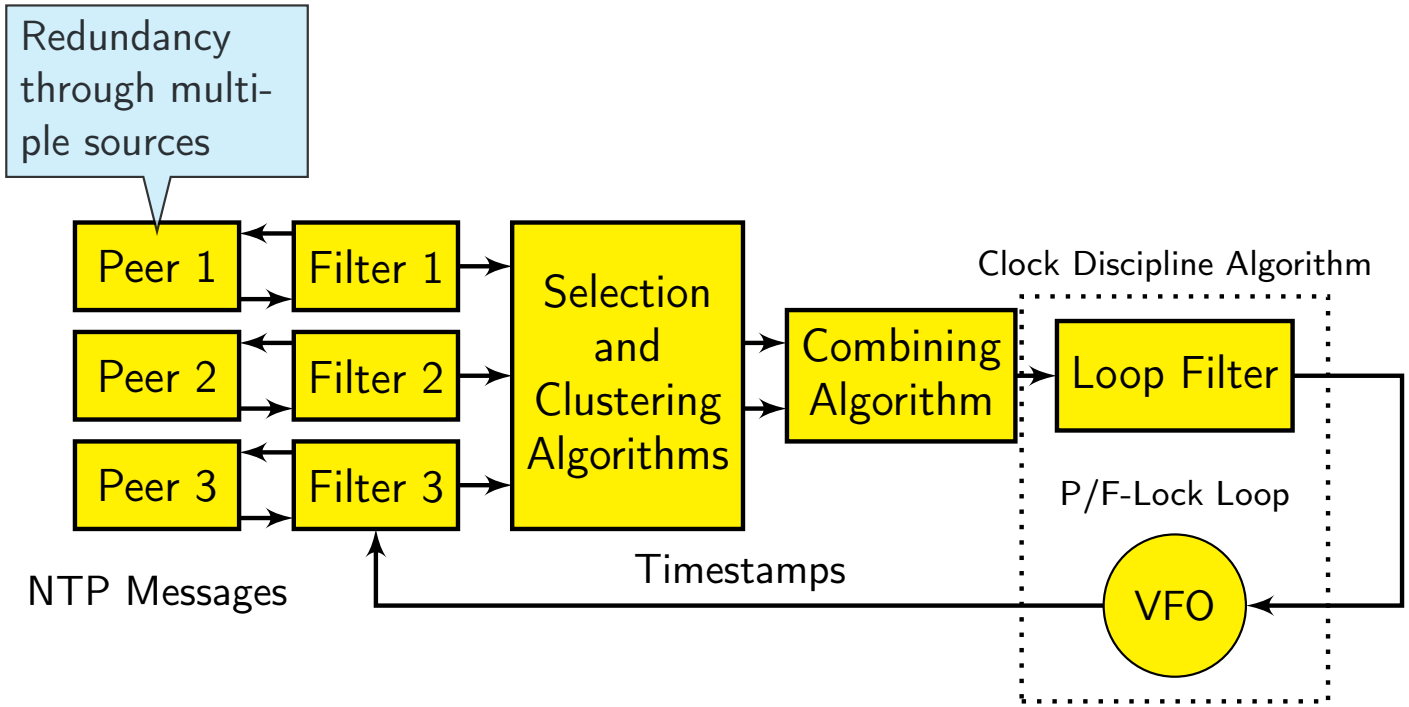
- RFC 1305, 1992, early Versions RFC 1129, RFC 958 (1985)
- $> 1,000,000$ hosts, router ...
- Uses UDP, port 123
- UNIX `ntpd`, `xntpd` (Clients available for most other systems)

NTP Functioning



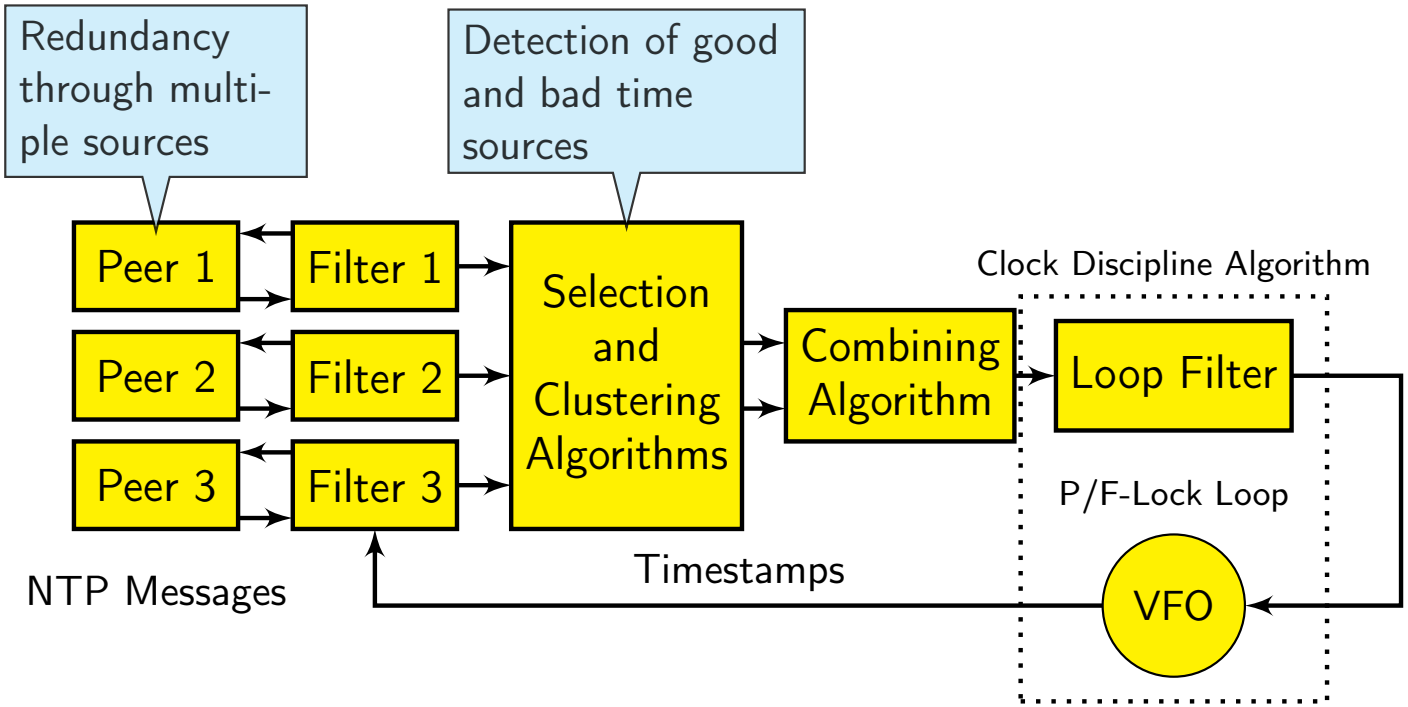
(Figure by Mills)

NTP Functioning



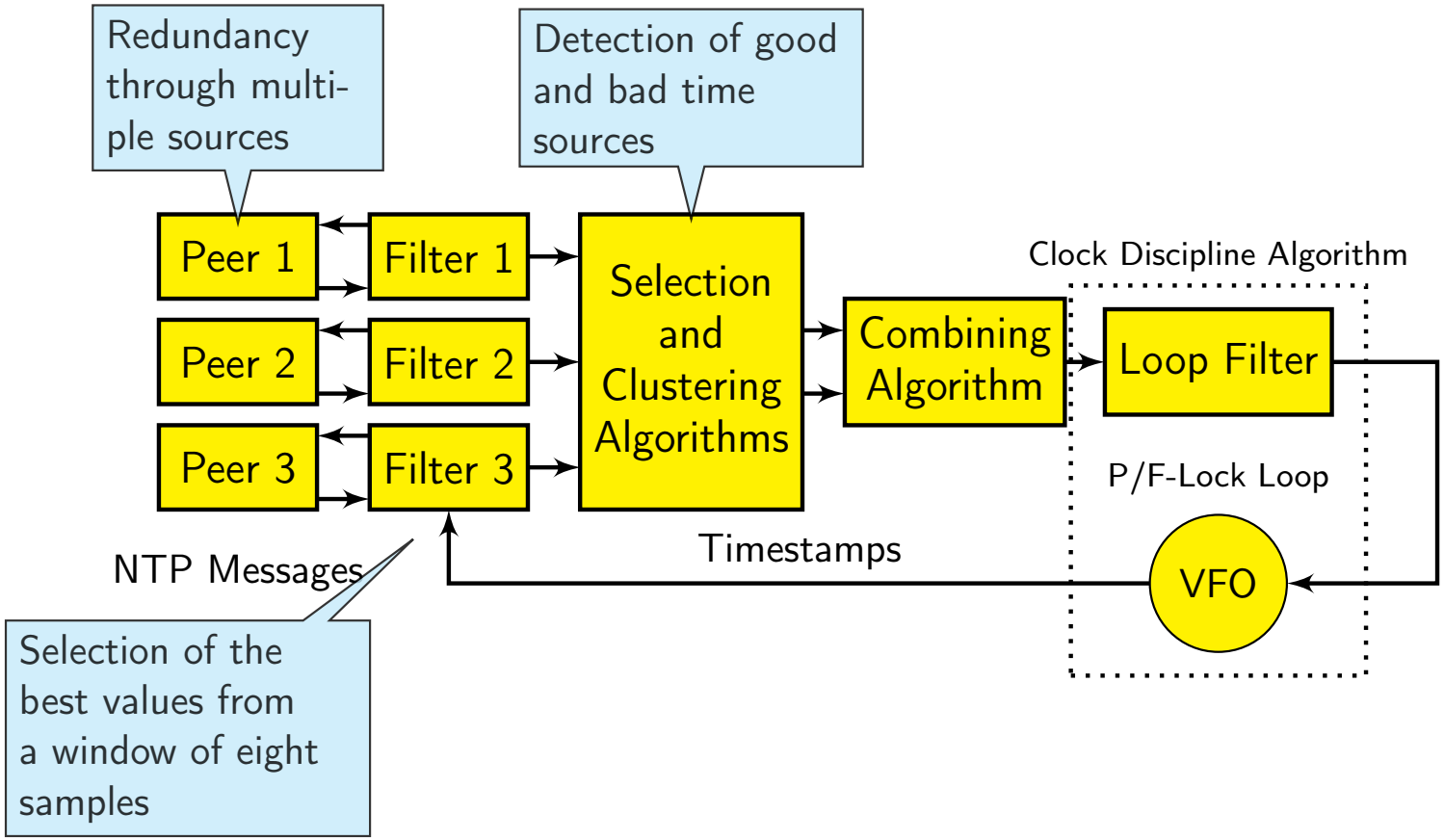
(Figure by Mills)

NTP Functioning



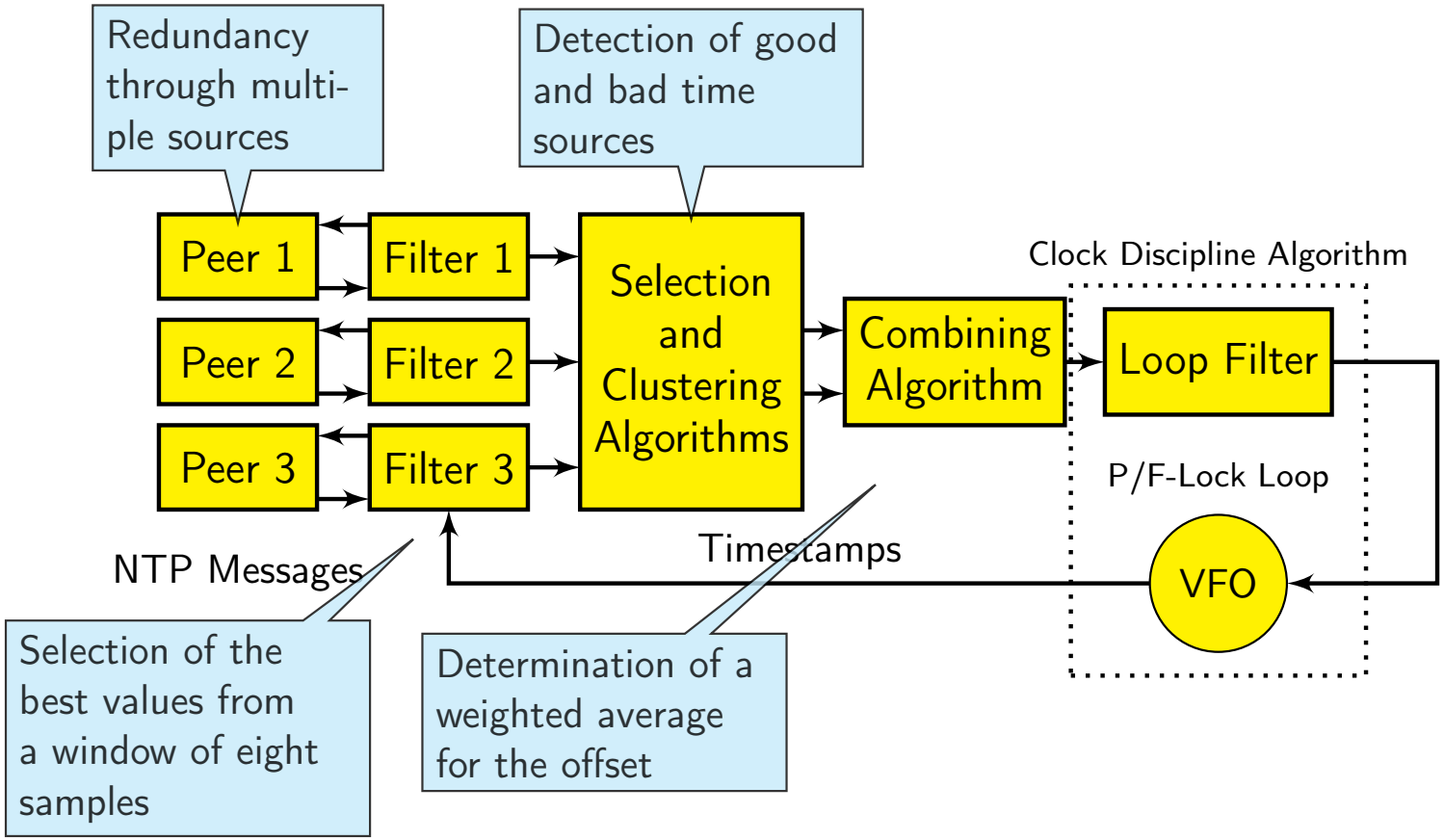
(Figure by Mills)

NTP Functioning



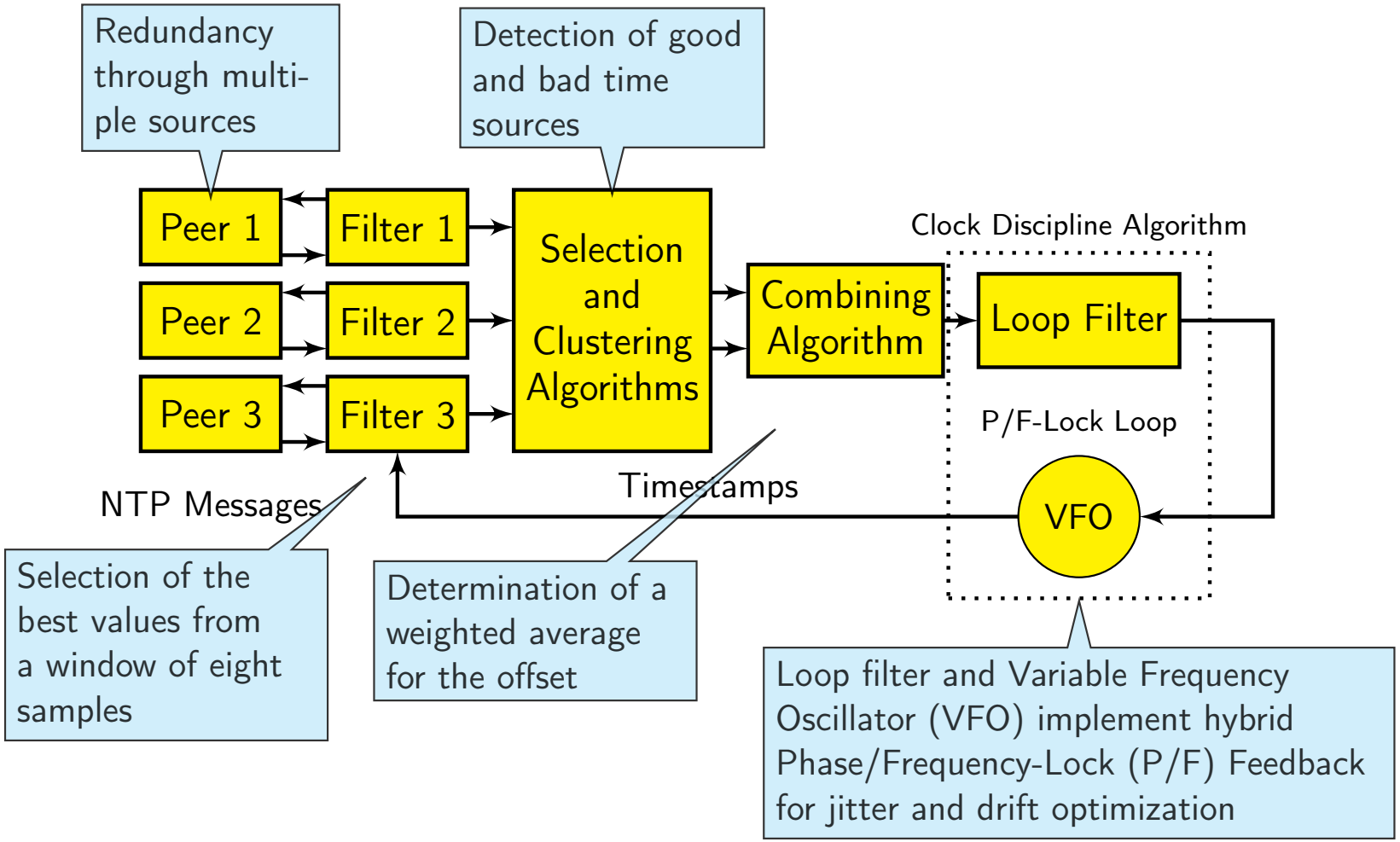
(Figure by Mills)

NTP Functioning



(Figure by Mills)

NTP Functioning



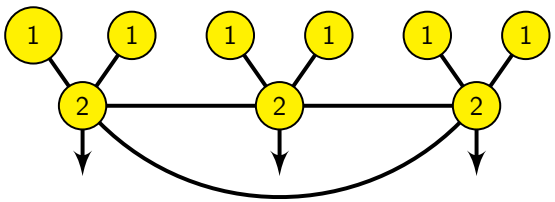
(Figure by Mills)

NTP Hierarchies

Server set and provide time, clients request time

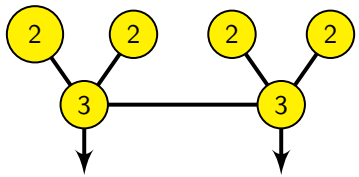
Hierarchy formation of the servers by **stratum** level

- Nodes with external reference time sources form stratum 1 server (Accuracy: < 1 μs possible)
- Stratum n - server synchronize with stratum $n - 1$ server etc.
- Internet usage (2022)
  - Respectively ca. 300 active stratum 1 and stratum 2 servers
    `http://support.ntp.org/bin/view/Servers/StratumOneTimeServers`
  - Practical: 4 layer hierarchy, load balancing through regional NTP pool server
- Typical structures:

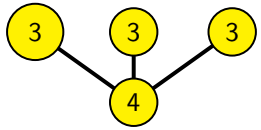


time server
(fault-tolerant)

Company



Department time server
(fault-tolerant)



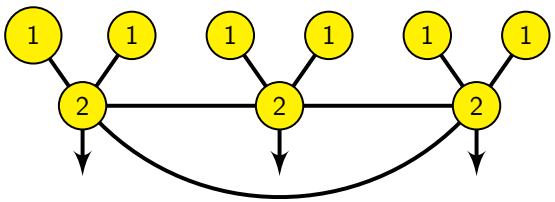
Workstation

NTP Hierarchies

Server set and provide time, clients request time

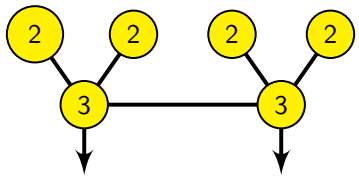
Hierarchy formation of the servers by **stratum** level

- Nodes with external reference time sources form stratum 1 server (Accuracy: < 1 μs possible)
- Stratum n - server synchronize with stratum $n - 1$ server etc.
- Internet usage (2022)
  - Respectively ca. 300 active stratum 1 and stratum 2 servers
    `http://support.ntp.org/bin/view/Servers/StratumOneTimeServers`
  - Practical: 4 layer hierarchy, load balancing through regional NTP pool server
- Typical structures:

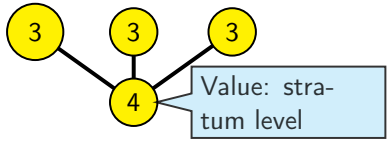


time server
(fault-tolerant)

Company



Department time server
(fault-tolerant)

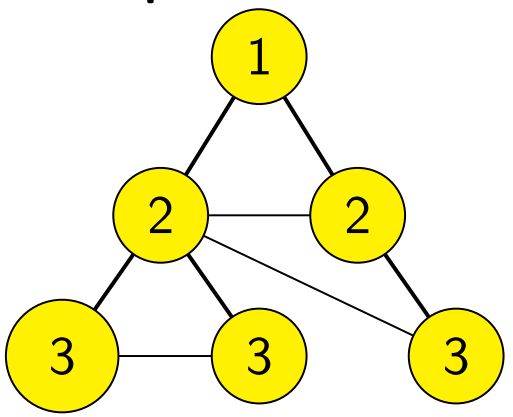


Workstation

NTP Properties

- Dynamically determined logical connection structure with redundant backup connections
  - Spanning trees with minimal weight based on the server level and overall synchronization delay of each server towards primary servers
- Message exchange between servers between 64s and 1024s (17 min) - depending on the connection quality
- 64 bit timestamps
  - 32 bit for seconds since 01.01.1900 00:00:00
  - 32 bit for parts of second
- Use of `settimeofday()` and `adjtime()` to adjust bigger resp. smaller ($< 0.128sec$) corrections
- Avoid jumps

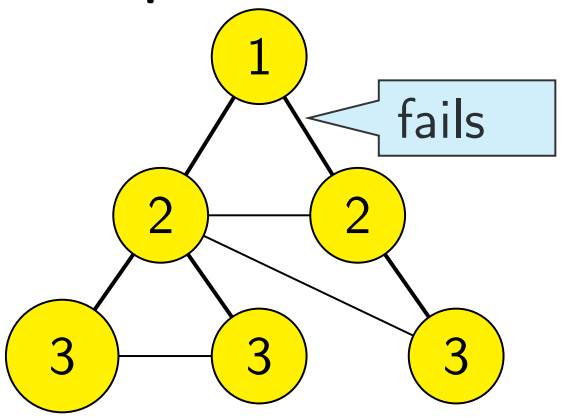
Example connection topology



NTP Properties

- Dynamically determined logical connection structure with redundant backup connections
  - Spanning trees with minimal weight based on the server level and overall synchronization delay of each server towards primary servers
- Message exchange between servers between 64s and 1024s (17 min) - depending on the connection quality
- 64 bit timestamps
  - 32 bit for seconds since 01.01.1900 00:00:00
  - 32 bit for parts of second
- Use of `settimeofday()` and `adjtime()` to adjust bigger resp. smaller ($< 0.128sec$) corrections
- Avoid jumps

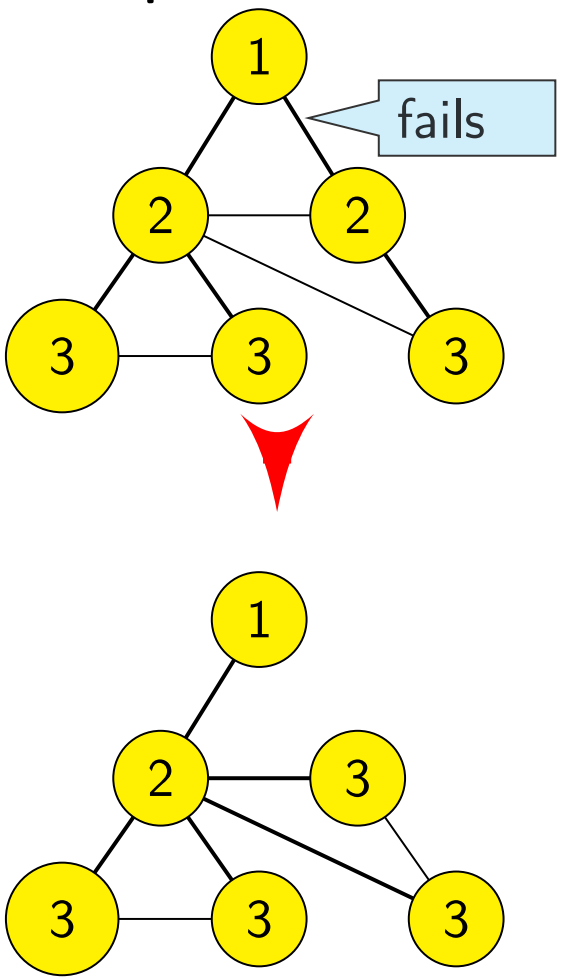
Example connection topology



NTP Properties

- Dynamically determined logical connection structure with redundant backup connections
  - Spanning trees with minimal weight based on the server level and overall synchronization delay of each server towards primary servers
- Message exchange between servers between 64s and 1024s (17 min) - depending on the connection quality
- 64 bit timestamps
  - 32 bit for seconds since 01.01.1900 00:00:00
  - 32 bit for parts of second
- Use of `settimeofday()` and `adjtime()` to adjust bigger resp. smaller ($< 0.128sec$) corrections
- Avoid jumps

Example connection topology

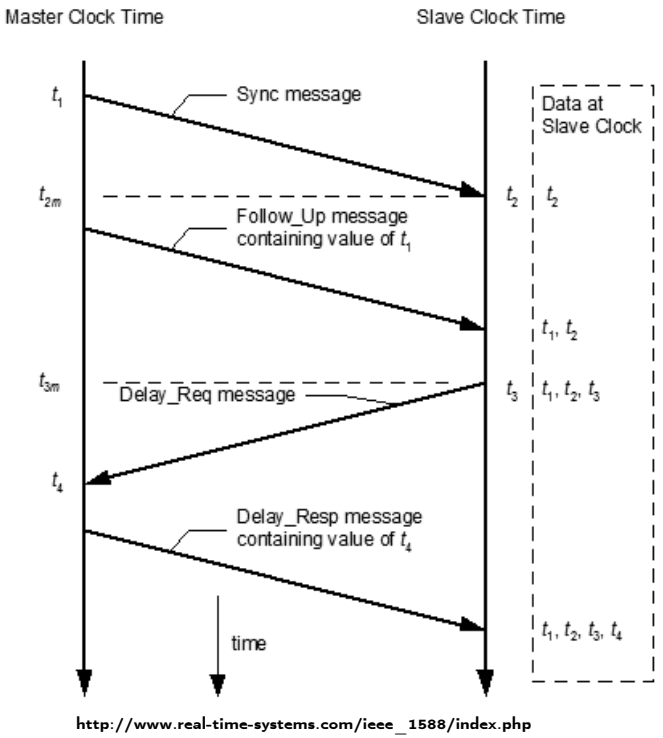


Precision Time Protocol (PTP, IEEE 1588)

- Mostly for measurement and control applications
- Achieves higher precision compared to NTP for networks with limited spatial expansion
- Client/server procedure
- Automatical selection of the best clock as grandmaster clock
- Primarily used for Ethernet networks
- Timestamping unit can be implemented as part of the network controller (in hardware)

$\Rightarrow$ Accuracy in the range of nanoseconds, in software in the range of μs

- Ptpd as open implementation
- Improved version IEEE 1588-2008



$$t_2 - t_1 = offset + d$$
$$t_4 - t_3 = -offset + d$$
$$offset = \frac{(t_2 - t_1 - t_4 + t_3)}{2}$$

Related Standards and Alternatives

- Simple Network Time Protocol (SNTP)
  - Simplified version of NTP
  - Compatible message format
  - No frequency compensation $\Rightarrow$ less accuracy but also less computational resources required
- Secure versions of NTP
  - Network Time Security (NTS)
  - NTPsec
  - tlsdate

Agenda

- Definition of Time
- Computer Clocks
- Synchronization Protocols
- Logical Clocks

Logical Timestamps

Real time is not always required

Examples:

- Order of events (before - after)
- Timestamp-based concurrency control in databases

Happened-before Relation

- **Notation**: $a \rightarrow b$ (a happened-before b)
- Events within the same process are ordered linearly
- Message sending:
  - a be the event of sending a message m
  - b be the reception of a message m at a different process
  - $\Rightarrow a \rightarrow b$
- Relation is transitive:
  - $a \rightarrow b,\ b \rightarrow c \Rightarrow a \rightarrow c$
- Concurrency:
  - If neither $a \rightarrow b$ nor $b \rightarrow a$ is true, a and b are called concurrent

Clock condition

Definition

$a \rightarrow b \Rightarrow C(a) < C(b)$

$C(a)$ refers to the (logical) time when event a is happening

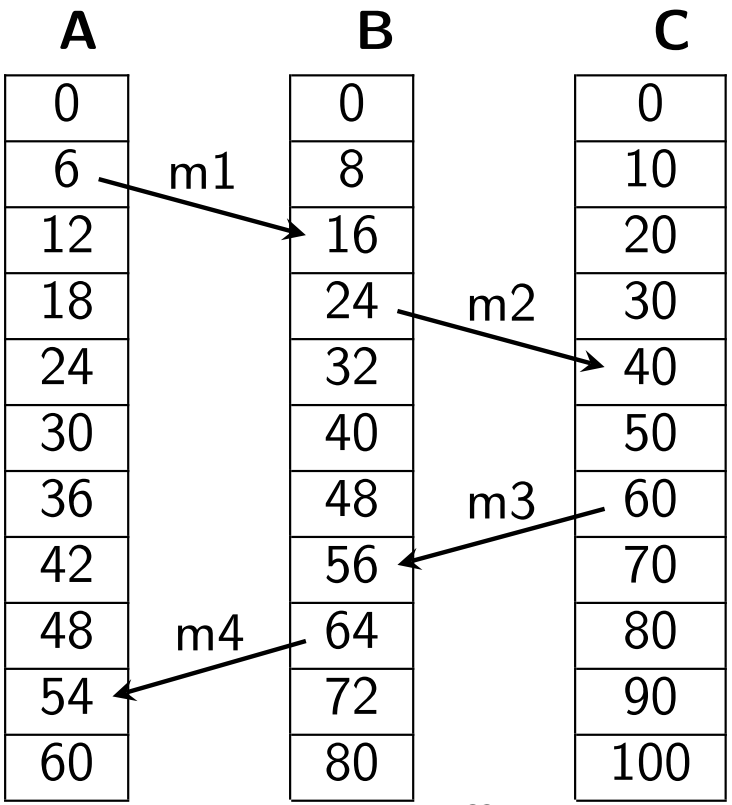
Lamport Clocks

Lamport algorithm for logical clocks (1978)

Assumptions:

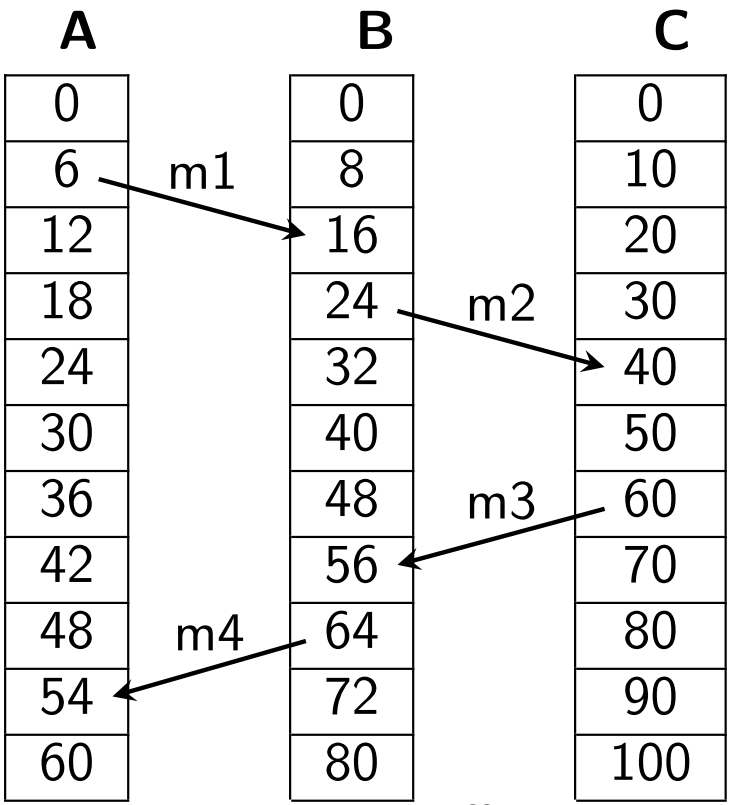
- Processes communicate via messages (and only via messages)
- Each process P has a logical clock C_P
- Each event e of process P receives a logical timestamp ($\rightarrow$ *Lamport Timestamps*): $C_P(e)$
- Two successive events e_i and e_{i+1} of a process have never the same timestamp: $C_P(e_i) < C_P(e_i + 1)$

Example for a Lamport Clock

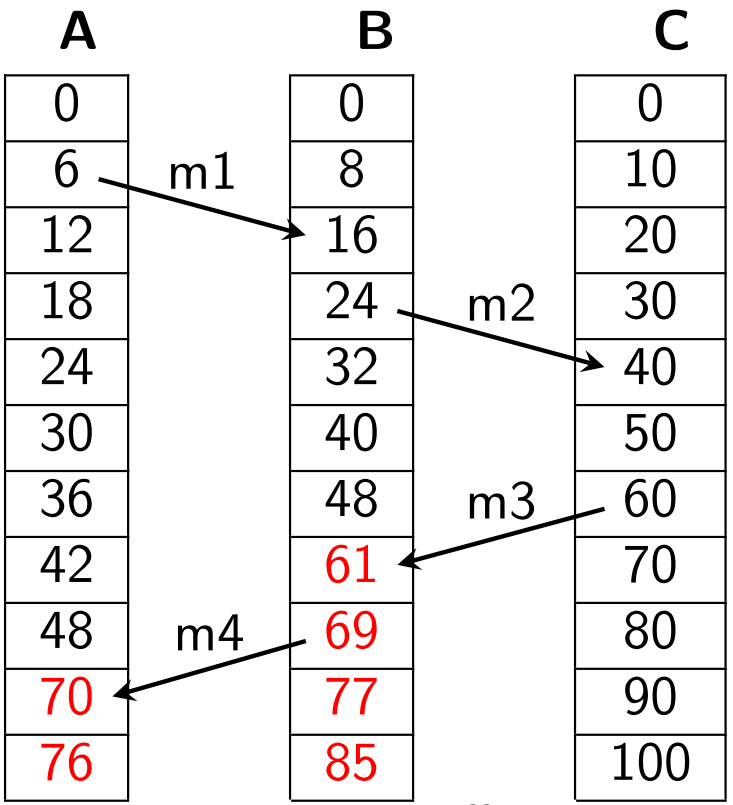


Clocks with differing speed without correction

Example for a Lamport Clock



Clocks with differing speed without correction



Clocks with differing speed with correction

Lamport Clock Algorithm

- Consideration of **causality** in message communication
- Sending event s of a message m in process A:
  - Timestamp $C_A(s)$
  - Send message m together with the current timestamp of the sending process $t = C_A(s)$
- Reception event e for message m at process B:
  - Be $C_B(old)$ the timestamp of the last event in B
  - Set $C_B(e) := max\{C_B(old), t\} + 1$
- If two events in different processes should have the same timestamp, ordering happens wrt to the process order
- Algorithm fulfills the clock condition

Lamport Clock Algorithm

- Consideration of **causality** in message communication
- Sending event s of a message m in process A:
  - Timestamp $C_A(s)$
  - Send message m together with the current timestamp of the sending process $t = C_A(s)$
- Reception event e for message m at process B:
  - Be $C_B(old)$ the timestamp of the last event in B
  - Set $C_B(e) := max\{C_B(old), t\} + 1$
- If two events in different processes should have the same timestamp, ordering happens wrt to the process order
- Algorithm fulfills the clock condition
- Reversal does **not** hold true:
  $C(a) < C(b) \Rightarrow a \rightarrow b$ is wrong!

$\rightarrow$ Lamport clocks do not solve the causality problem

Vector Clocks

by Mattern (Uni Kaiserslautern, 1989)

Vector clocks solve (among others) the causality problem

Algorithm:

- Message based communication
- Each process P_i has a local clock VC_i as vector of timestamps
- Local event in P_i:
  - $VC_i[i] := VC_i[i] + 1$, otherwise unchanged
- Sending event in P_i:
  - $VC_i[i] := VC_i[i] + 1$ (increase own event counter)
  - Send message containing the local vector time $vt = VC_i$
- Reception event in P_k:
  - $VC_k[j] := max\{VC_k[j], vt[j]\}$ for all j
  - $VC_k[k] := VC_k[k] + 1$ (increase own event counter)

Comparison of Timestamps for Vector Clocks

- $S \leq T :\Rightarrow S[i] \leq T[i]$ for all i
- $S < T :\Rightarrow S \leq T$ and $S \neq T$
- $S||T :\Rightarrow \neg(S < T)$ and $\neg(T < S)$

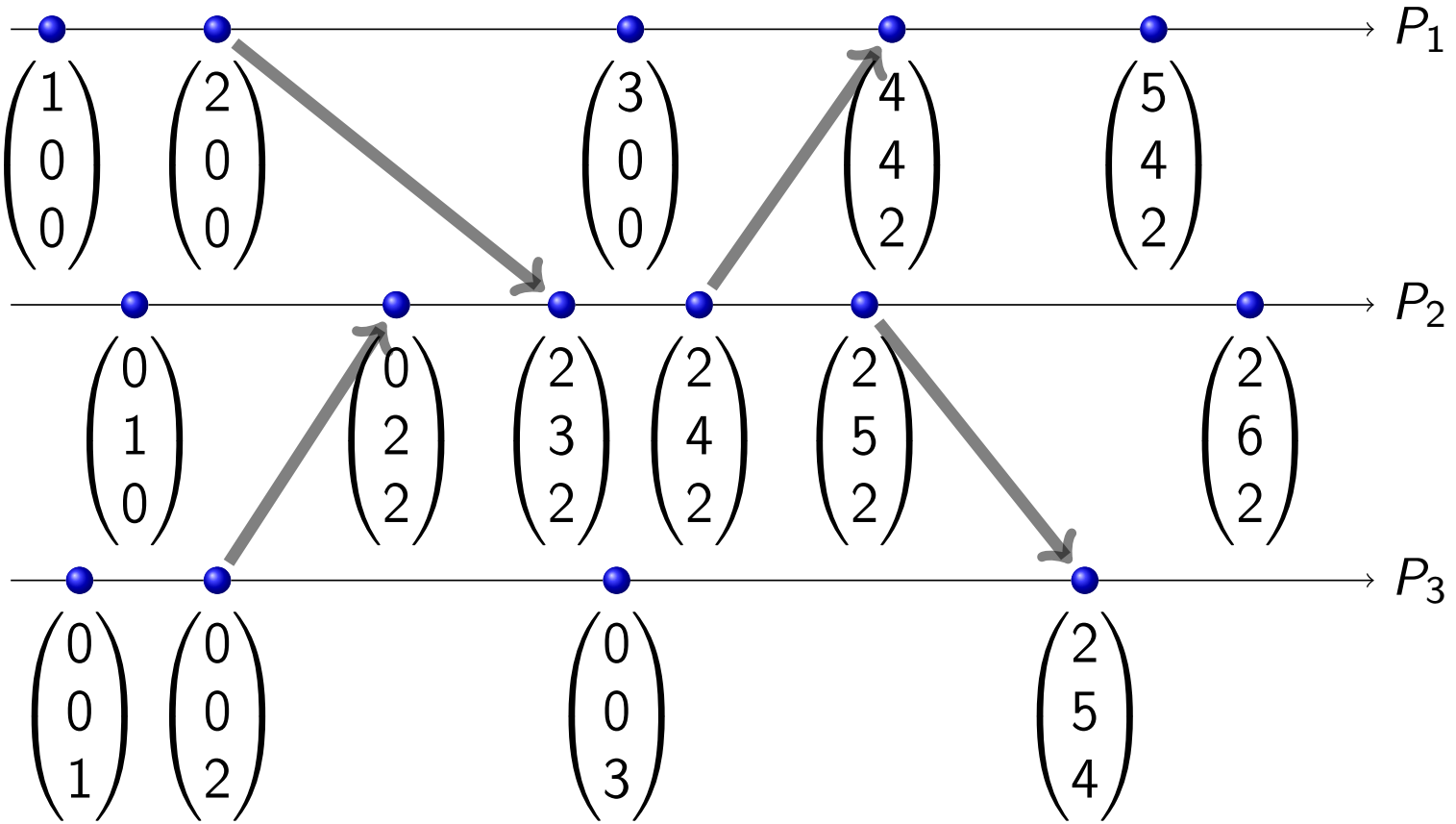
Concurrency

- Events a and b are concurrent $\Leftrightarrow VC(a)||VC(b)$

Causality

- $a \rightarrow b \Leftrightarrow VC(a) < VC(b)$

Vector Clocks Example



- Causal dependent events, e.g., $(0,0,1) \rightarrow (5,4,2)$, $(1,0,0) \rightarrow (2,6,2)$
- Concurrent events, e.g., $(0,0,3) || (5,4,2)$

Important takeaway messages of this chapter

- Times can be compared to a reference time wrt to stability, precision, and resolution. Drift and offset can be measured.
- To synchronize local clocks in a distributed system to a reference time source a clock synchronization protocol is required.
- Lamport and vector clocks can be used for logical timestamps