

# Echtzeitverarbeitung


R. Kaiser, K. Beckmann, R. Kröger

(HTTP: <http://www.cs.hs-rm.de/~kaiser>


E-Mail: [robert.kaiser@hs-rm.de](mailto:robert.kaiser@hs-rm.de))

Sommersemester 2021


## 2. Zeit und Ordnen





### Mein Tageslauf



16


Schreibe unter jeden Wecker die Uhrzeit (wie im Beispiel)! Setze dann die richtigen Nummern zu den Weckern!


**1** Du schläfst noch.
 


**2** Du stehst auf!
 


**3** Du wäschst dich.
 

**4** Du ziehst dich an.
 


**5** Du frühstückst.
 


**6** Du putzt deine Zähne.
 


**7** Du gehst zur Schule.
 





7:05 Uhr














<https://www.unterstufe.ch/hinweise.php?id=25668>

# Inhalt



## 2. Zeit und Ordnen

### 2.1 Einführung

### 2.2 Zeitbegriff und Zeitsysteme

### 2.3 Rechneruhren

### 2.4 Globale Zeitbasis und Synchronisationsprotokolle

### 2.5 Logische Zeitmarken

# Einführung



- Zeit als physikalische Größe
- Ordnen von Ereignissen aufgrund zeitlicher Ordnung
- Verteilte Systeme mit mehreren Uhren
- Probleme mit nicht-synchronisierten Rechneruhren (Bsp.)
  - ▶ Zeitstempel von Dateien (make)
  - ▶ zeitgesteuertes Ausführen von Aufträgen (cron)
- Globaler Zeitbegriff
  - ▶ Etablierung von systemweiter Zeit in verteilten Systemen durch Synchronisation von Rechneruhren
  - ▶ Synchronität mit realer Außenzeit
- Anwendungsprogrammierung
  - ▶ Nutzung von Zeitdiensten
  - ▶ Zeitmessung

# Einführung (2)



## Anwendungen

- korrekte Funktion zeitbezogener lokaler und verteilter Anwendungen
- korrektes Ordnen von Ereignissen in verteilten Systemen, z.B. für Prozessvisualisierung
- effizientere verteilte Algorithmen durch Verringerung des Kommunikationsaufwands (vgl. [Liskov])
- Leistungsmessung in verteilten Systemen
- verteilte Echtzeitsysteme benötigen Zeitbegriff einschließlich Synchronität mit realer Außenzeit

# Zeitbegriff und Zeitsysteme



## Verschiedene Zeitbegriffe im Laufe der Geschichte

### Astronomische Zeit

- basiert auf der gleichförmigen Bewegung von Himmelskörpern und deren Beobachtung
- Sonnenzeit
  - ▶ mittlere Dauer einer Erdumdrehung
  - ▶ Sonnentag: Zenit-Zenit (bis 1956)
  - ▶  $1 \text{ Sek} = \frac{1}{24 \cdot 60 \cdot 60} \text{ Sonnentag}$
  - ▶ wenig stabil (Abbremsung der Erdrotation, Schwankungen durch Massenverlagerungen)
- Sternzeit
  - ▶ mittlere Dauer der Umlaufzeit der Erde um die Sonne
  - ▶  $1 \text{ Sek} = \frac{1}{31.556.925,9747} \text{ Teil des trop. Jahres 1900 (ab 1957)}$

# Physikalische Zeit



basiert auf (periodischen) physikalischen Prozessen

**Beispiele:**

- Kerzenuhr (Verbrennen von Wachs)
- Pendeluhr, Genauigkeit best:  $10^{-7}$
- Quarzuhr, Genauigkeit best:  $10^{-9}$ , typisch:  $10^{-5} \dots 10^{-6}$

**Atomuhr**

- Definition im SI-Einheitensystem (ab 1967):  
*„Die Sekunde ist das 9.192.631.770fache der Periodendauer der dem Übergang zwischen den beiden Hyperfeinstrukturniveaus des Grundzustandes von Atomen des Nuklids  $^{133}\text{Cs}$  entsprechenden Strahlung.“*
- Cäsium-133-Uhr, Genauigkeit best:  $10^{-14}$ , typisch:  $10^{-13}$   
( $< 1\mu\text{s}$  pro Jahr)
- Caesium-Fontäne, Genauigkeit  $< 10^{-15}$

# Physikalisch-Technische Bundesanstalt (PTB)



Hochschule RheinMain

- in Braunschweig
- Betrieb mehrerer Atomuhren (CS1-CS4, CSF1)
- Verantwortung für die gesetzliche Zeit in D (ab 1978)
- Betrieb von Verteildiensten



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

Foto: PTB



# Zeitsysteme



## GMT: Greenwich Mean Time

- Lokale Ortszeitangaben (wahre und mittlere) üblich bis ca. 1880
- Probleme für Eisenbahn-Fahrpläne
- „Greenwich Mean Time“ gesetzliche Standardzeit in England ab 1880
- Ab 1.06.1891: deutsche und österreichisch/ungarische Eisenbahnverwaltungen führen die Zeit des 15. Längengrads als *mitteleuropäische Eisenbahn-Zeit (M. E. Z.)* ein.
- Deutsches Reich: gesetzliche Uhrzeit ab 1.04.1893 ist „die mittlere Sonnenzeit des fünfzehnten Längengrades östlich von Greenwich“
- Meridiankonferenz Washington 1884 definiert Greenwich als Null-Meridian und führt Zeitzonen ein → GMT Sonnenzeit
- ab 1.1.1925 Beginn des Tags um Mitternacht (für Astronomen bis da hin mittags)

# Zeitsysteme (2)



## UT: Universal Time

- Weltzeit abgeleitet aus Sternzeit (ab 1957) am Null-Meridian
- UT1: Berücksichtigung der Polschwankungen

## TAI: Temps Atomique International

- mittlere Atomzeit seit 1.1.1958
- Betrieb von ca. 250 Atomuhren weltweit
- weltweit koordiniert durch Bureau International de l'Heure (BIH)

## UTC: Universal Time Coordinated

- heutiger Zeitstandard (ab 1972)
- basiert auf TAI, aber Anpassungen an UT1 durch „Schaltsekunde“ bei mehr als 900 ms Unterschied
- Abweichung: 1 Sek in 300.000 Jahren

# Zeitverteildienste



## Langwellen-Radiosender

- z.B. in D: DCF77 (77.5 kHz, Frankfurt/Mainflingen)
- basierend auf Atomuhr CS-2 der PTB
- Sekundentakt
- aufmodulierter voller BCD-Zeitcode (58 Bit) in jeder Minute
- Genauigkeit
  - ▶  $2 \cdot 10^{-13}$  gemittelt über 100 Tage
  - ▶ 1-10 msec je Sek. (atmosphärische Störungen)

## GEOS Satellitensystem

- Geostationary Operational Environment Satellite
- Genauigkeit ca. 0.5 msec

# Zeitverteildienste (2)



## GPS-Satellitensystem als Basis

- Global Positioning System, primär militärisch
- 24 Satelliten, Umlaufzeit 12 h, mind. 4 jederzeit „sichtbar“
- Cäsium-Uhren an Bord
- Synchronisation gegenüber Uhren anderer Satelliten durch Bodenstation auf  $\pm 5$  ns genau
- Standortbestimmung durch Unterschiede in Signallaufzeiten ( $5\text{ns} \cong 1.5\text{m}$  mil.;  $1\mu\text{s} \cong 300\text{m}$  zivil)
- künstliche Ungenauigkeiten in Krisenzeiten
- Differentielles GPS nutzt zusätzlich Bodenstationen mit bekannten Standorten (Geodäsie)

## GPS-basierte Uhr

- GPS-Signal als Referenz einer PLL-Schaltung
- hochgenaue Sekundenimpulse (pps pulse-per-second)
- typ. Genauigkeit: ca.  $1\mu\text{s}$  (nach ca.  $\frac{1}{2}$  h Betrieb, z.B. Meinberg)

# Zeitverteildienste (3)



## Galileo-Satellitensystem der EU/ESA (bis 201x)

- europäisches, zu GPS kompatibles System (GPS III)
- bis zu 30 Satelliten
  - ▶ mit je 2 Atomuhren
  - ▶ senden Zeitsignal und Positionsdaten
  - ▶ globale Abdeckung
- Dienste
  - ▶ Unterscheidung in globale, regionale, lokale Ebene
  - ▶ kostenloser Dienst für Ortung, Navigation, Zeitsynchronisation (Genauigkeit ca. 4 m horizontal, 8 m vertikal)
  - ▶ kommerzieller Dienst (Genauigkeit 1 m, Bewegungen 0.2 m/sec) (Vermessungswesen, Netzsynchronisation, Flottenmanagement)
  - ▶ Safety-of-Life-Dienst, (sicherheitskritische Anwendungen in Luft- und Schifffahrt, Bahnverkehr)
  - ▶ Dienst „von öffentlichem Interesse“, (Signal mit sehr hoher Genauigkeit, Qualität, Zuverlässigkeit und Integrität für hoheitliche Anwendungen)



# Zeitverteildienste (4)



## Galileo-Satellitensystem der EU/ESA (Stand)

- Sicht 2004:  
Entwicklung bis 2006, Betrieb ab 2008 geplant
- Sicht 2006:  
Konzessionsvergabe bis 2008, Errichtung 2009/10, Betrieb ab 2010
- Sicht 2008:  
Konsortium zerbrochen, Betrieb ab 2013 im Auftrag der EU, Kosten ca. 3,6 Mrd. €
- Sicht 2011:  
21.10.11: die ersten beiden Satelliten mit Sojus-Rakete in Orbit gebracht.
- Sicht 2012:  
13.10.12: zwei weitere Satelliten, neue Kostenschätzung: 5,0 Mrd. €, Probetrieb ab 2013, Funktionsfähigkeit 2020 (?!?), Bodentestbetrieb Region Berchtesgaden (virtueller Satellitenbetrieb)
- Sicht 2016:  
18 der vorgesehenen 30 Satelliten im Orbit. Letzte Satelliten sollen 2018 in ihre Umlaufbahn geschossen werden. System ist seit 15. Dezember 2016 allgemein zugänglich.



# Begriffe (nach Kopetz)



- Zeit modelliert als Zeitstrahl
  - ▶ Unendliche Menge  $\mathbb{T}$  von Zeitpunkten mit
  - ▶  $\mathbb{T}$  ist geordnet:  $p, q \in \mathbb{T} : p < q, p = q, p > q$
  - ▶  $\mathbb{T}$  ist dicht:  
 $p, q \in \mathbb{T}, p \neq q : \exists r$  zwischen  $p$  und  $q$ , d.h. für  $p < q : p < r < q$
- Zeitdauer als Intervall auf Zeitstrahl
- Ereignis findet zu einem Zeitpunkt statt  
( $\rightarrow$  temporale Ordnung auf Ereignissen)
- Ereignisse finden gleichzeitig statt, wenn sie zu gleichem Zeitpunkt stattfinden
- Menge der Ereignisse nur partiell geordnet
  - ▶ wegen gleichzeitiger Ereignisse
  - ▶ Totale Ordnung konstruierbar mit zusätzlichem Kriterium (z.B. Knotennummer in verteilten Systemen)

# Begriffe (nach Kopetz)



- Kausale Ordnung von Ereignissen
  - ▶ Ursache/Wirkungs-Beziehung
  - ▶ Aus kausaler Ordnung folgt temporale Ordnung der Ereignisse,
  - ▶ Umkehrung gilt nicht
  - ▶ Kausale Ordnung damit strenger als temporale Ordnung
- Einheitliche Empfangsordnung von Ereignissen (*Delivery Order*)
  - ▶ Kommunikationssystem stellt einheitliche Ordnung aller Nachrichten bei allen Empfängern sicher
  - ▶ Empfangsordnung muss nicht mit temporaler Ordnung oder Empfangsordnung übereinstimmen



# Begriffe



## Referenzzeit

- Approximation der wahren physikalischen Zeit
- Zähler der Referenzuhr zeigt immer korrekten Wert an

## Abweichung, Genauigkeit (Accuracy)

- absolute oder relative Differenz zu einer Referenzzeit
- Offset als Zeitdifferenz zwischen zwei Uhren bzw. zur Referenzzeit

## Auflösung/Granularität (Granularity)

- kleinste Zeitdauer zwischen zwei aufeinander folgenden anzeigbaren Zeitpunkten ( $\frac{1}{f}$ , wobei  $f$  Frequenz)

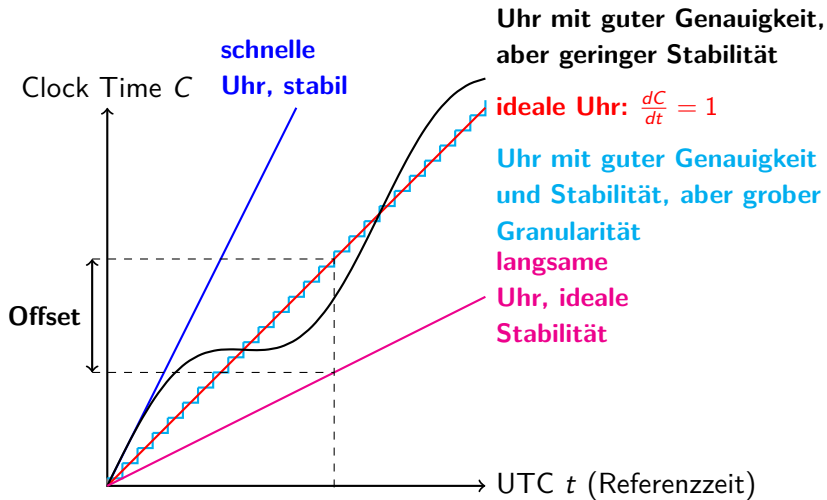
## Stabilität (Stability)

- Frequenzschwankung einer Uhr
- Drift als Frequenzdifferenz zwischen zwei Uhren bzw. zur Referenzzeit

## Präzision einer Menge von Uhren (Precision)

- Max. Offset zwischen irgend zwei Uhren der Menge

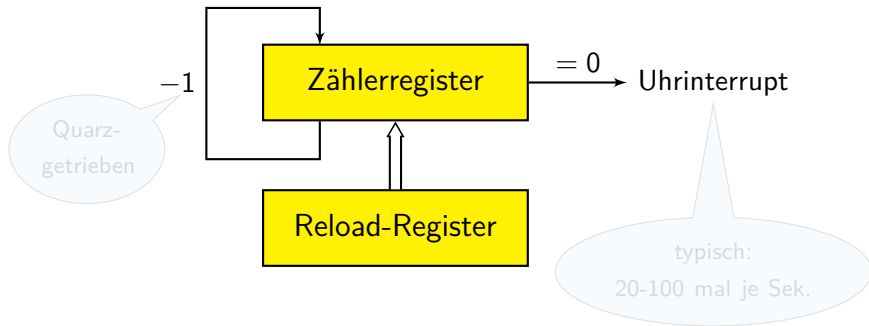
# Veranschaulichung



# Rechneruhren



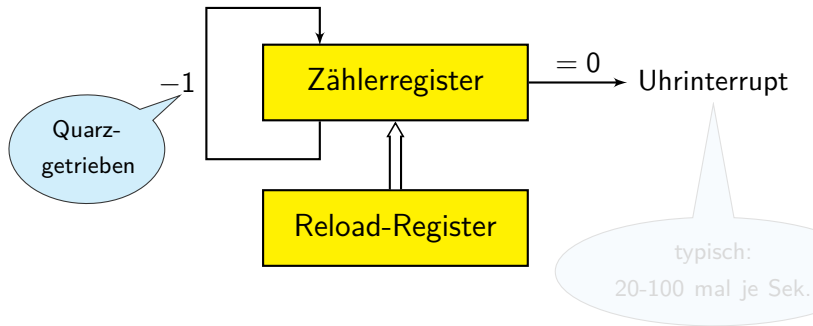
## Hardware einer lokalen Rechensystemuhr



# Rechneruhren



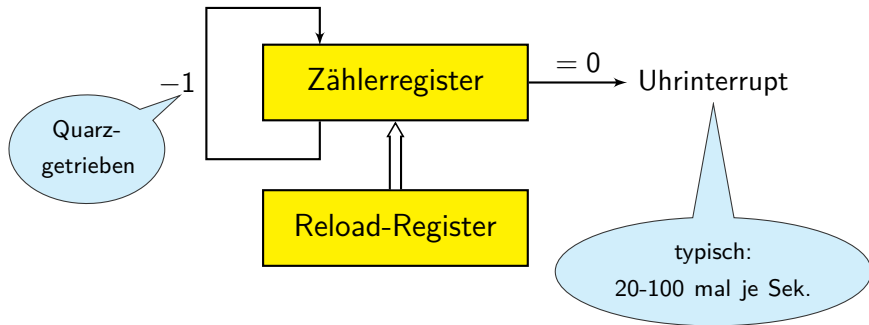
## Hardware einer lokalen Rechensystemuhr



# Rechneruhren



## Hardware einer lokalen Rechensystemuhr

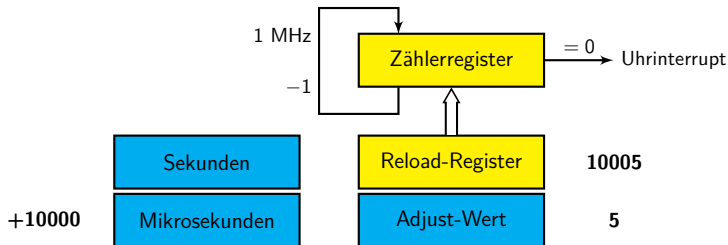


# Betriebssystem-Uhren

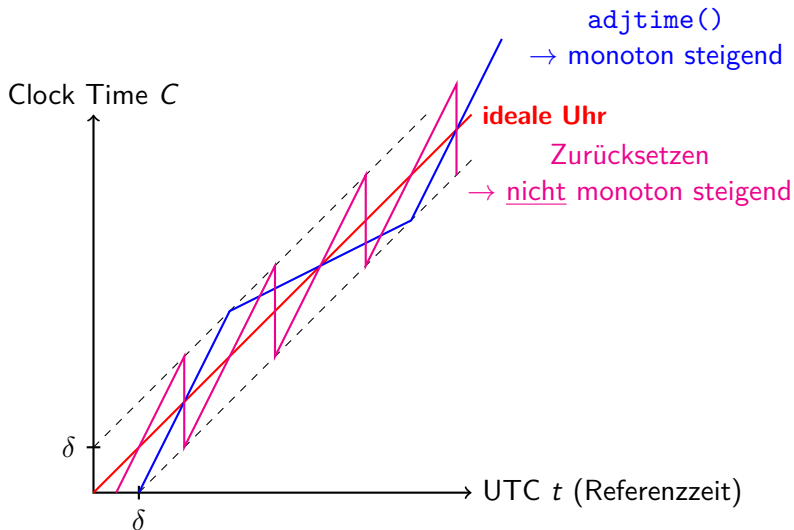


## Beispiel UNIX

- zwei 32-Bit (oder 64-Bit) Integer-Variablen
  - ▶ Anzahl Sekunden seit 1.1.1970
  - ▶ Anzahl  $\mu\text{s}$  (oder ns) in der aktuellen Sekunde
- typ. 100 Interrupts/s
- bei Interrupt werden Variablen um nominelle Anzahl  $\mu\text{s}$  erhöht
- Korrekturwert (Adjust-Wert) für Ausgleich der Drift des Quarzes
- Systemdienste `settimeofday`, `adjtime`



# Prinzip der Korrektur



# Referenzzeitquellen



## DCF77-Uhr für einfache Anforderungen an Systemzeit

- Genauigkeit typisch:  $\pm 2$  msec

## GPS-Uhr bei hohen Anforderungen (z.B. Messsystem)

- Genauigkeit typisch:  $\pm 250$  nsec

## Atom-Uhr

- Rubidium / Caesium-Quellen
- Spezielle Zulassung erforderlich
- Montage in Rack
- z.T. ausschließlich für militärische Zwecke

## Rechnerschnittstelle

- Erzeugung von Pulse-Per-Second (pps)-Signalen als Interrupts
- Kodierte Timecode-Signale,  
z.B. IRIG-Standard (Inter Range Instrumentation Group)



# Genaue lokale Betriebssystem-Uhren



## Verwendung einer externen Referenzzeitquelle Linux-Kern mit „Nano-Kernel-Patch“

- Erhöhung der Auflösung der Systemuhr auf 1 ns (statt  $\mu s$ )
- Standard in neueren Linux-Kernen
- Nutzung der pps-Signale der Referenzzeitquelle als Interrupts
- Korrektur der Systemuhr entsprechend Referenzzeit der Hardware-Uhr
- Varianz der Interrupt-Latenzzeiten beeinflusst Genauigkeit
- mehrere externe Zeitquellen an einem Rechner möglich zur weiteren Erhöhung der Genauigkeit
- Genauigkeit: typisch  $< 1 \mu s$

# Beispiel: David L. Mill's Uhren (Uni Delaware)



Hochschule RheinMain



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

- Spectracom 8170 WWVB Receiver
- Spectracom 8183 GPS Receiver
- 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)

# Kommerzielle Time Server



## Time Server

- Dedizierter LAN-Netzwerkknoten zur Zeitsynchronisation
- Interne oder externe Referenzzeitquelle
- Unterstützung für Standard-Protokolle (s.u.) (NTP, SNTP, PTP/IEEE 1588)

## Produkte in vielen Varianten

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

# Globale Zeitbasis und Synchronisationsprotokolle

## Globale Zeitbasis und Synchronisationsprotokolle

- Abstraktion
- Approximiert durch lokale Zeiten eines Ensembles synchronisierter lokaler Uhren

### Plausibilitätsbedingung für globale Granularität $g$

- Die globale Zeit heißt plausibel für die Granularität  $g$ , wenn
    - ▶ die sie lokal implementierenden Uhren des Ensembles eine beschränkte Präzision  $\Pi$  besitzen (max. Unterschied je zweier Uhren)
    - ▶  $g > \Pi$
- Synchronisationsfehler ist kleiner als ein Tick der gedachten globalen Uhr
- für jedes Ereignis  $e$  unterscheiden sich die globalen Zeitmarken beliebiger lokaler Uhren  $j$  und  $k$  um maximal eine Einheit (Tick):
- $$|t^j(e) - t^k(e)| \leq 1$$
- Dies ist das optimal erreichbare Ergebnis

# Folgerungen



- Wenn sich die globalen Zeitmarken zweier Ereignisse um max. einen Tick (Granularität  $g$ ) unterscheiden, kann die korrekte temporale Ordnung nicht hergestellt werden.
- Ab Unterschied von zwei globalen Ticks ist korrekte temporale Ordnung möglich und durch die Zeitmarken gegeben
- Für die wahre Dauer  $d_{true}$  eines Zeitintervalls bei beobachteter Dauer  $d_{obs}$  gilt

$$(d_{obs} - 2g) < d_{true} < (d_{obs} + 2g)$$

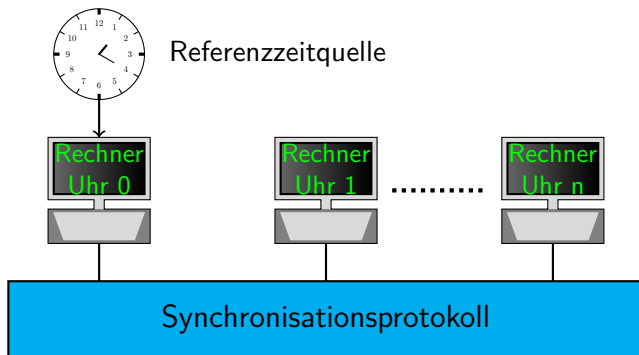
- Für eine konsistente globale temporale Ordnung zweier globaler Ereignisse durch zwei lokale Knoten ist ein Abstand der Ereignisse von  $3g$  notwendig

# Synchronisationsprotokolle



## Konstruktion einer verteilten Zeitbasis für Rechensysteme

- UTC-basierte externe Referenzzeitquelle
- lokale Uhren in den Rechensystemen
- Synchronisationsprotokoll



# Probleme



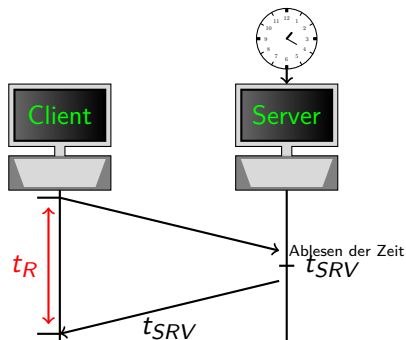
- Nachrichtenverzögerung im Netzwerk nicht deterministisch
  - Bearbeitung der Protokollnachrichten zeitlich nicht deterministisch
- ⇒ **keine exakte Synchronisation möglich**

# Algorithmus von Christian (1989)



- passiver Zeitserver (als Referenzzeitquelle)
- periodisches Abfragen der Zeit durch Klienten
- mittlere Roundtriplaufzeit (incl. Verarbeitungszeit auf dem Server) messen und berücksichtigen
- Schwächen:
  - ▶ „Rückwärtsgehen“ einer Uhr ist möglich
  - ▶ Schwankungen in Nachrichtenlaufzeiten

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





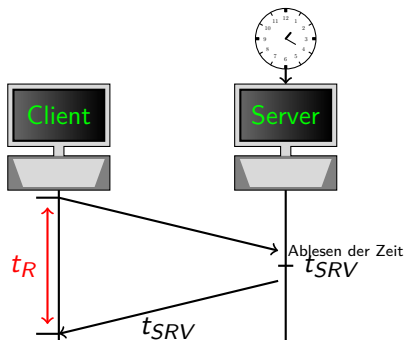
# Algorithmus von Christian (1989)



- passiver Zeitserver (als Referenzzeitquelle)
- periodisches Abfragen der Zeit durch Klienten
- mittlere Roundtriplaufzeit (incl. Verarbeitungszeit auf dem Server) messen und berücksichtigen
- Schwächen:
  - ▶ „Rückwärtsgehen“ einer Uhr ist möglich
  - ▶ Schwankungen in Nachrichtenlaufzeiten

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

Roundtrip-Zeit  
gemessen mit  
lokaler Uhr



# Time Synchronisation Protocol (TSP)



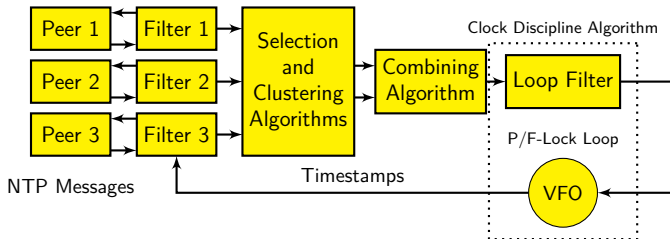
- Berkeley UNIX `timed`
- basiert auf ICMP/IP
- etabliert „mittlere Netzwerkzeit“ in allen Stellen
- Master/Slave-Algorithmus
  - ▶ aktiver Master: fragt aktuelle Zeiten aller Knoten ab, berechnet Mittel
  - ▶ verteilt Differenz (Offset) an jeden Client
- nutzt `settimeofday()` und `adjtime()` in den Knoten
- deutliche Schwächen
  - ▶ „Rückwärtsgehen“ einer Uhr ist möglich
  - ▶ keine Kompensation von Schwankungen in Nachrichtenlaufzeiten
  - ▶ keine Fehlerabschätzung
  - ▶ schlechte Skalierbarkeit
- Variante: Master mit ext. Referenzzeitquelle verteilt aktuelle Zeit statt berechnetem Mittelwert

# Network Time Protocol (NTP)



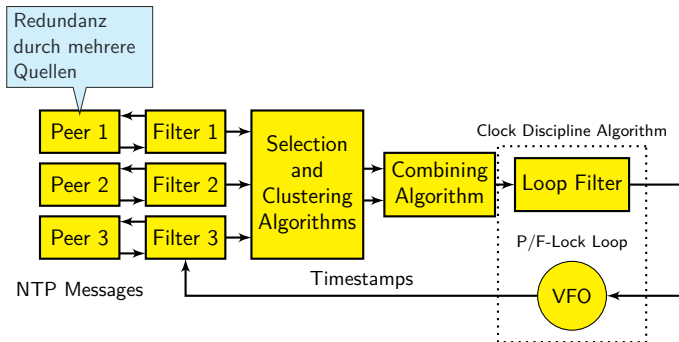
- Entwicklung primär durch D. Mills (Univ. of Delaware) getrieben
- <http://www.ntp.org>
- Ziele:
  - ▶ hohe Genauigkeit
  - ▶ Berücksichtigung schwankender Nachrichtenlaufzeiten
  - ▶ Berücksichtigung von Rechnerausfällen durch Bezug zu mehreren Zeitservern (Peers)
  - ▶ Aussortieren offensichtlich unbrauchbarer Zeitquellen (false ticker)
  - ▶ eingeschränkte Authentifizierung, Verschlüsselung
  - ▶ hohe Skalierbarkeit
- Heute Internet Standard
  - ▶ RFC 1305, 1992, frühere Version RFC 1129, RFC 958 (1985)
  - ▶ > 1.000.000 Rechner, Router, usw.
  - ▶ Nutzt UDP, Port 123
  - ▶ UNIX ntpd, xntpd (Clients aber auch für fast alle anderen Systeme)
  - ▶ Zeitserver der PTB: ptbtime1.ptb.de, ptbtime2.ptb.de
- Genauigkeit:
  - ▶ im LAN <1 ms, Internet < ca. 10 ms

# NTP(2)-Arbeitsweise



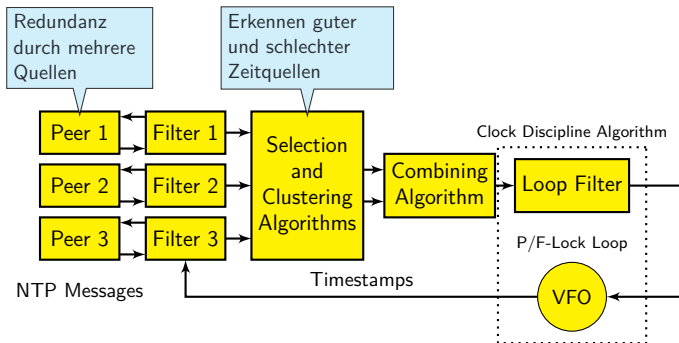
(Abbildung von Mills)

# NTP(2)-Arbeitsweise



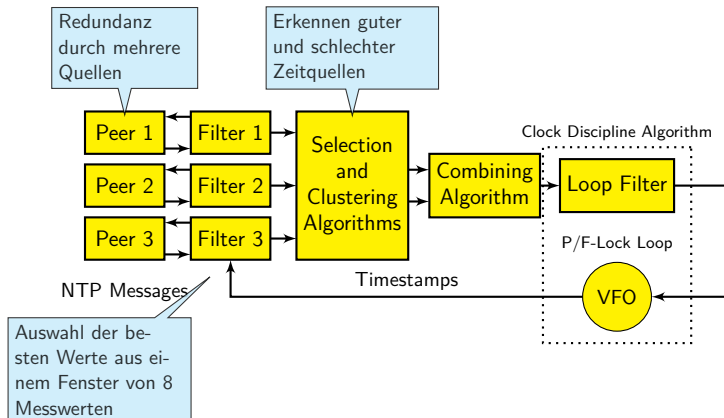
(Abbildung von Mills)

# NTP(2)-Arbeitsweise



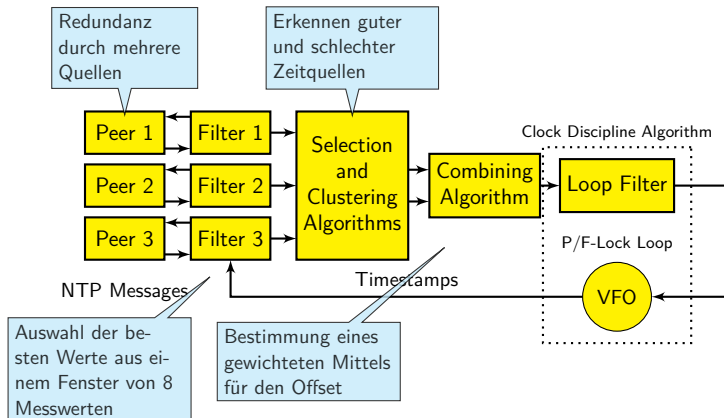
(Abbildung von Mills)

# NTP(2)-Arbeitsweise



(Abbildung von Mills)

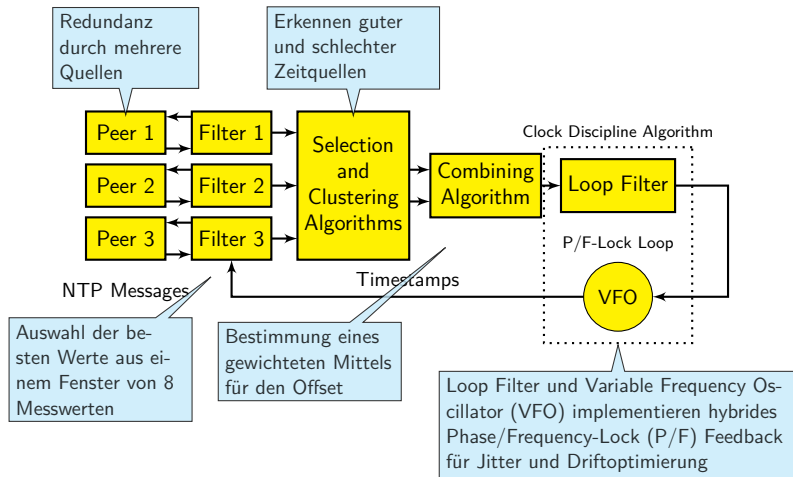
# NTP(2)-Arbeitsweise



(Abbildung von Mills)



# NTP(2)-Arbeitsweise



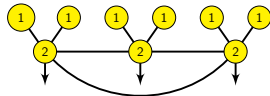
(Abbildung von Mills)

# NTP(3)

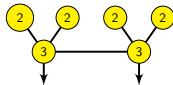


## Server legen Zeit fest, Clients beziehen Zeit Hierarchiebildung der Server durch „Stratum“-Level

- Knoten mit externen Referenzzeitquellen bilden Stratum 1 -Server (Genauigkeit:  $< 1 \mu s$  möglich)
- Stratum n - Server synchronisieren sich mit Stratum n-1 - Servern, usw.
- im Internet (2015)
  - ▶ Jeweils ca. 300 aktive Stratum-1 und Stratum-2 Server  
<http://support.ntp.org/bin/view/Servers/StratumOneTimeServers>
  - ▶ Praktisch: 4-stufige Hierarchie, Lastausgleich durch regionale NTP Pool Server
- Typische Strukturen:



Unternehmens-Zeitserver  
(fehlertolerant)



Abteilungs-Zeitserver  
(fehlertolerant)



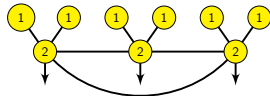
Workstation

# NTP(3)

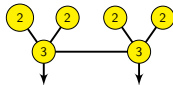


## Server legen Zeit fest, Clients beziehen Zeit Hierarchiebildung der Server durch „Stratum“-Level

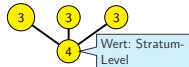
- Knoten mit externen Referenzzeitquellen bilden Stratum 1 -Server (Genauigkeit:  $< 1 \mu s$  möglich)
- Stratum n - Server synchronisieren sich mit Stratum n-1 - Servern, usw.
- im Internet (2015)
  - ▶ Jeweils ca. 300 aktive Stratum-1 und Stratum-2 Server  
<http://support.ntp.org/bin/view/Servers/StratumOneTimeServers>
  - ▶ Praktisch: 4-stufige Hierarchie, Lastausgleich durch regionale NTP Pool Server
- Typische Strukturen:



Unternehmens-Zeitserver  
(fehlertolerant)



Abteilungs-Zeitserver  
(fehlertolerant)

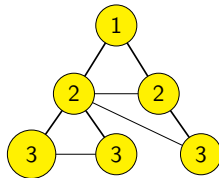


Workstation

# NTP(3)



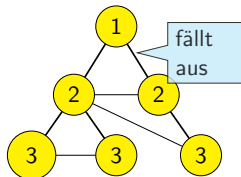
- dynamisch festgelegte logische Verbindungsstruktur mit Backup-Verbindungen
  - ▶ spannende Bäume minimalen Gewichts basierend auf Server Level und Gesamtsynchronisationsverzögerung jedes Servers zu Primary Servern
- Beispiel Verbindungstopologie
- Nachrichtenaustausch zwischen Servern zwischen 64 sec und 1024 sec (17 min) je nach Qualität der Verbindung
- 64 Bit Zeitmarken
  - ▶ 32 Bit für Sekunden seit 1.1.1900 00:00:00
  - ▶ 32 Bit für Sekundenbruchteil
- Nutzung von `settimeofday()` und `adjtime()` zur Durchsetzung großer bzw. kleiner ( $<0.128$  sec) Korrekturen.
- Kein Zurücksetzen der Uhr



# NTP(3)



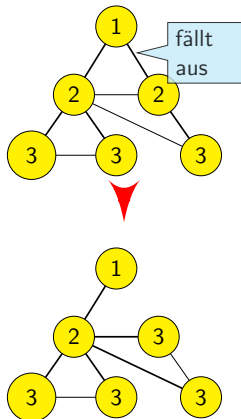
- dynamisch festgelegte logische Verbindungsstruktur mit Backup-Verbindungen
  - ▶ spannende Bäume minimalen Gewichts basierend auf Server Level und Gesamtsynchronisationsverzögerung jedes Servers zu Primary Servern
- Beispiel Verbindungstopologie
- Nachrichtenaustausch zwischen Servern zwischen 64 sec und 1024 sec (17 min) je nach Qualität der Verbindung
- 64 Bit Zeitmarken
  - ▶ 32 Bit für Sekunden seit 1.1.1900 00:00:00
  - ▶ 32 Bit für Sekundenbruchteil
- Nutzung von `settimeofday()` und `adjtime()` zur Durchsetzung großer bzw. kleiner ( $<0.128$  sec) Korrekturen.
- Kein Zurücksetzen der Uhr



# NTP(3)



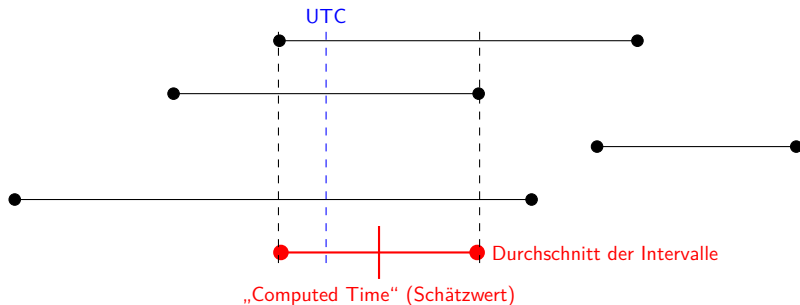
- dynamisch festgelegte logische Verbindungsstruktur mit Backup-Verbindungen
  - ▶ spannende Bäume minimalen Gewichts basierend auf Server Level und Gesamtsynchronisationsverzögerung jedes Servers zu Primary Servern
- Beispiel Verbindungstopologie
- Nachrichtenaustausch zwischen Servern zwischen 64 sec und 1024 sec (17 min) je nach Qualität der Verbindung
- 64 Bit Zeitmarken
  - ▶ 32 Bit für Sekunden seit 1.1.1900 00:00:00
  - ▶ 32 Bit für Sekundenbruchteil
- Nutzung von `settimeofday()` und `adjtime()` zur Durchsetzung großer bzw. kleiner ( $<0.128$  sec) Korrekturen.
- Kein Zurücksetzen der Uhr



# Distributed Time Service (DTS)



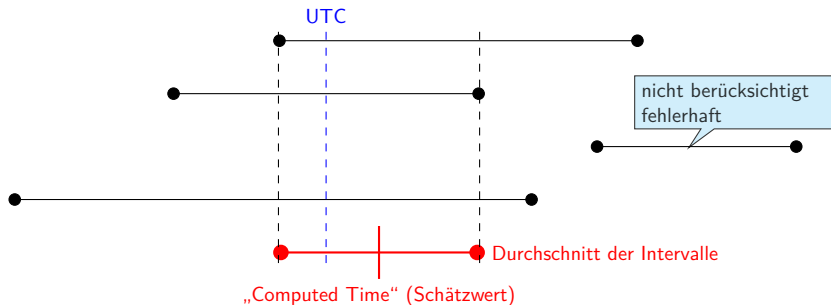
- Teil von OSF DCE, ursprünglich von Digital entwickelt
- Besonderheit: Etablieren eines Zeitintervalls, das UTC enthält und Ungenauigkeit minimiert
- adjust im Verhältnis 1:100



# Distributed Time Service (DTS)



- Teil von OSF DCE, ursprünglich von Digital entwickelt
- Besonderheit: Etablieren eines Zeitintervalls, das UTC enthält und Ungenauigkeit minimiert
- adjust im Verhältnis 1:100





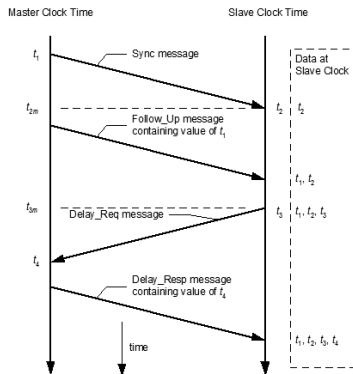
# Precision Time Protocol (PTP, IEEE 1588)



- Hauptsächlich für mess- und regelungstechnische Anwendungen
- Erreicht höhere Genauigkeit als NTP für Netze mit räumlich begrenzter Ausdehnung
- Master-Slave-Verfahren
- Automatische Wahl der besten Uhr als Grandmaster-Clock
- Primär auf Ethernet-Netzen angewendet
- Timestamping-Unit kann als Teil des Netzwerk-Controllers (in Hardware) implementiert sein

⇒ Genauigkeit im ns-Bereich, in Software im  $\mu$ s-Bereich

- Ptpd als freie Implementierung
- Verbesserte Version IEEE 1588-2008



[http://www.real-time-systems.com/ieee\\_1588/index.php](http://www.real-time-systems.com/ieee_1588/index.php)

$$t_2 - t_1 = \text{offset} + d$$

$$t_4 - t_3 = -\text{offset} + d$$

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

# Logische Zeitmarken



**Realzeit ist nicht immer notwendig**

**Beispiele:**

- Ordnen von Ereignissen (vor - nach)
- zeitmarkenbasiertes Concurrency Control in Datenbanken

# Lamport Zeitstempel



## Relation happens-before

- Notation:  $a \rightarrow b$  ( $a$  passiert-vor  $b$ )
- Ereignisse im selben Prozess sind linear geordnet
- Nachrichtenversand:
  - ▶  $a$  sei Ereignis des Versendens einer Nachricht  $m$
  - ▶  $b$  sei Empfang der Nachricht  $m$  in einem anderen Prozess
  - ▶ dann gilt:  $a \rightarrow b$
- Relation ist transitiv:
  - ▶  $a \rightarrow b, b \rightarrow c \Rightarrow a \rightarrow c$
- Nebenläufigkeit:
  - ▶ falls weder  $a \rightarrow b$  noch  $b \rightarrow a$  gilt, heißen  $a$  und  $b$  nebenläufig

## Uhrenbedingung

- $C(a)$  bezeichne die (logische) Zeit, zu der das Ereignis  $a$  stattfindet.
- $a \rightarrow b \Rightarrow C(a) < C(b)$

# Lamport-Uhren



## Algorithmus für logische Uhren nach Lamport (1978)

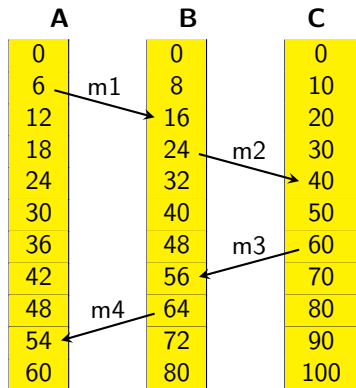
### Annahmen:

- Prozesse kommunizieren über Nachrichten (und nur über Nachrichten) miteinander
- jeder Prozess  $P$  hat eine logische Uhr  $C_P$
- jedes Ereignis  $e$  des Prozesses  $P$  erhält logischen Zeitstempel  $C_P(e)$
- zwei aufeinander folgende Ereignisse  $e_i$  und  $e_{i+1}$  eines Prozesses haben nie den gleichen Zeitstempel:  $C_P(e_i) < C_P(e_{i+1})$

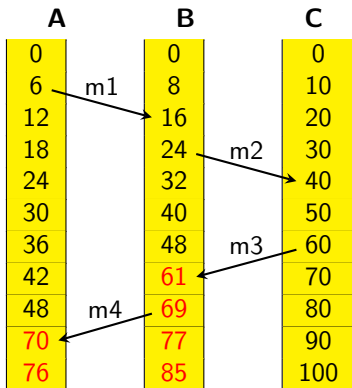
# Lamport-Uhren (2)



## Beispiel:



Uhren mit unterschiedlichen  
Geschwindigkeiten ohne Korrektur



Uhren mit unterschiedlichen  
Geschwindigkeiten **mit** Korrektur

# Lamport-Uhren (3)



## Algorithmus:

- Berücksichtigung von Kausalität im Nachrichtenversand!
- Sendeereignis  $s$  einer Nachricht  $m$  in Prozess  $A$ :
  - ▶ Zeitmarke  $C_A(s)$
  - ▶ Versende Nachricht  $m$  zusammen mit aktuellem Zeitstempel des sendenden Prozesses  $t = C_A(s)$
- Empfangsereignis  $e$  der Nachricht  $m$  in Prozess  $B$ :
  - ▶ sei  $C_B(alt)$  die Zeitmarke des letzten Ereignisses in  $B$
  - ▶ Setze  $C_B(e) := \max\{C_B(alt), t\} + 1$
- Falls zwei Ereignisse in verschiedenen Prozessen die gleiche Zeitmarke haben sollten, ordne sie anhand der Prozessordnung
- Algorithmus erfüllt Uhrenbedingung
- Umkehrung gilt **nicht**:  
 $C(a) < C(b) \Rightarrow a \rightarrow b$  ist falsch !
- Lamport-Uhren lösen nicht das Kausalitätsproblem

# Vector Clocks



## Vektor-Uhren, Mattern (Uni Kaiserslautern, 1989)

### Vektor-Uhren lösen o.a. Kausalitätsproblem

#### Algorithmus:

- nachrichtenbasierte Kommunikation
- jeder Prozess  $P_i$  besitzt Uhr  $VC_i$  als Vektor von Zeitmarken
- lokales Ereignis in  $P_i$ :
  - ▶  $VC_i[i] := VC_i[i] + 1$  , sonst unverändert
- Sendeereignis in  $P_i$ :
  - ▶  $VC_i[i] := VC_i[i] + 1$  (Erhöhe eigenen Ereigniszähler)
  - ▶ Versende Nachricht mit eigener Vektorzeit  $vt = VC_i$
- Empfangsereignis in  $P_k$ :
  - ▶  $VC_k[j] := \max\{VC_k[j], vt[j]\}$  für alle  $j$
  - ▶  $VC_k[k] := VC_k[k] + 1$  (Erhöhe eigenen Ereigniszähler)

# Vector Clocks (2)



## Vergleich von Zeitmarkenvektoren

- $S \leq T \Rightarrow S[i] \leq T[i]$  für alle  $i$
- $S < T \Rightarrow S \leq T$  und  $S \neq T$
- $S \parallel T \Rightarrow \neg(S < T)$  und  $\neg(T < S)$

## Nebenläufigkeit

- Ereignisse  $a$  und  $b$  sind nebenläufig  $\Leftrightarrow VC(a) \parallel VC(b)$

## Kausalität

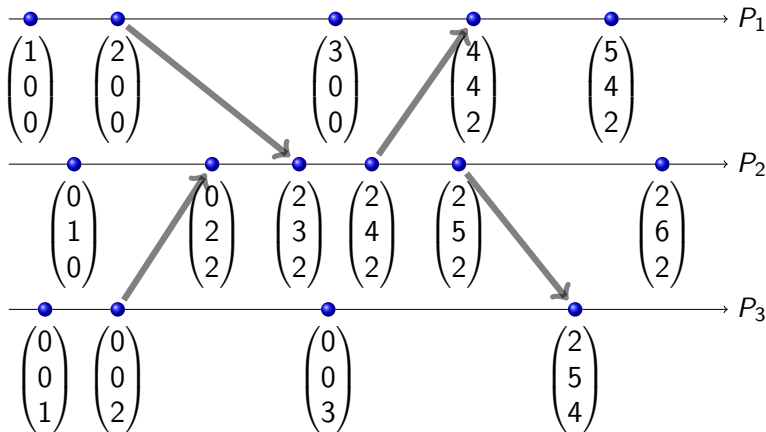
- $a \rightarrow b \Leftrightarrow VC(a) < VC(b)$



# Vector Clocks (3)



## Beispiel



- kausal abhängige Ereignisse, z.B.  $(0, 0, 1) \rightarrow (5, 4, 2)$ ,  $(1, 0, 0) \rightarrow (2, 6, 2)$
- nebenläufige Ereignisse, z.B.  $(0, 0, 3) || (5, 4, 2)$