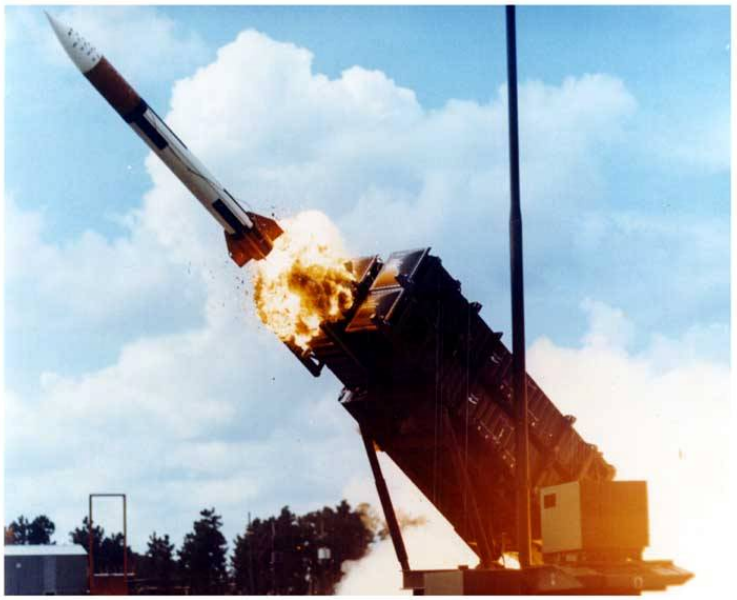
Inhalt

- Definition und Motivation
- Internationale Zeitstandards
- Zeit als Modell
- Globale Zeit in verteilten System
  - Reasonableness Condition
  - π/Δ-Präzedenz
- Synchronisation
  - Algorithmus von Christian
  - Algorithmus aus Berkeley
  - NTP-Protokoll

- **Definition**
- Historisch:
  - Jeden Tag gegen Mittag erreicht die Sonne ihren höchsten Punkt am Himmel
  - Die Zeitspanne zwischen zwei aufeinander folgenden Ereignissen dieses Typs heißt Tag (genauer gesagt: ein Sonnentag)
  - Eine Sonnensekunde ist 1/86400 dieser Spanne
- Zeitmessung heute:
  - Verwendung von Atomuhren: eine Sekunde ist die 9.192.631.770-fache Periodendauer, der dem Übergang zwischen den beiden Hyperfeinstrukturniveaus des Grundzustands von 133 Cäsium-Atomen entsprechenden Strahlung
  - Am 01.01.1958 entsprach die Atomsekunde genau einer Sonnensekunde
- "Zeit ist, was verhindert, dass alles auf einmal passiert" (John A. Wheelers)

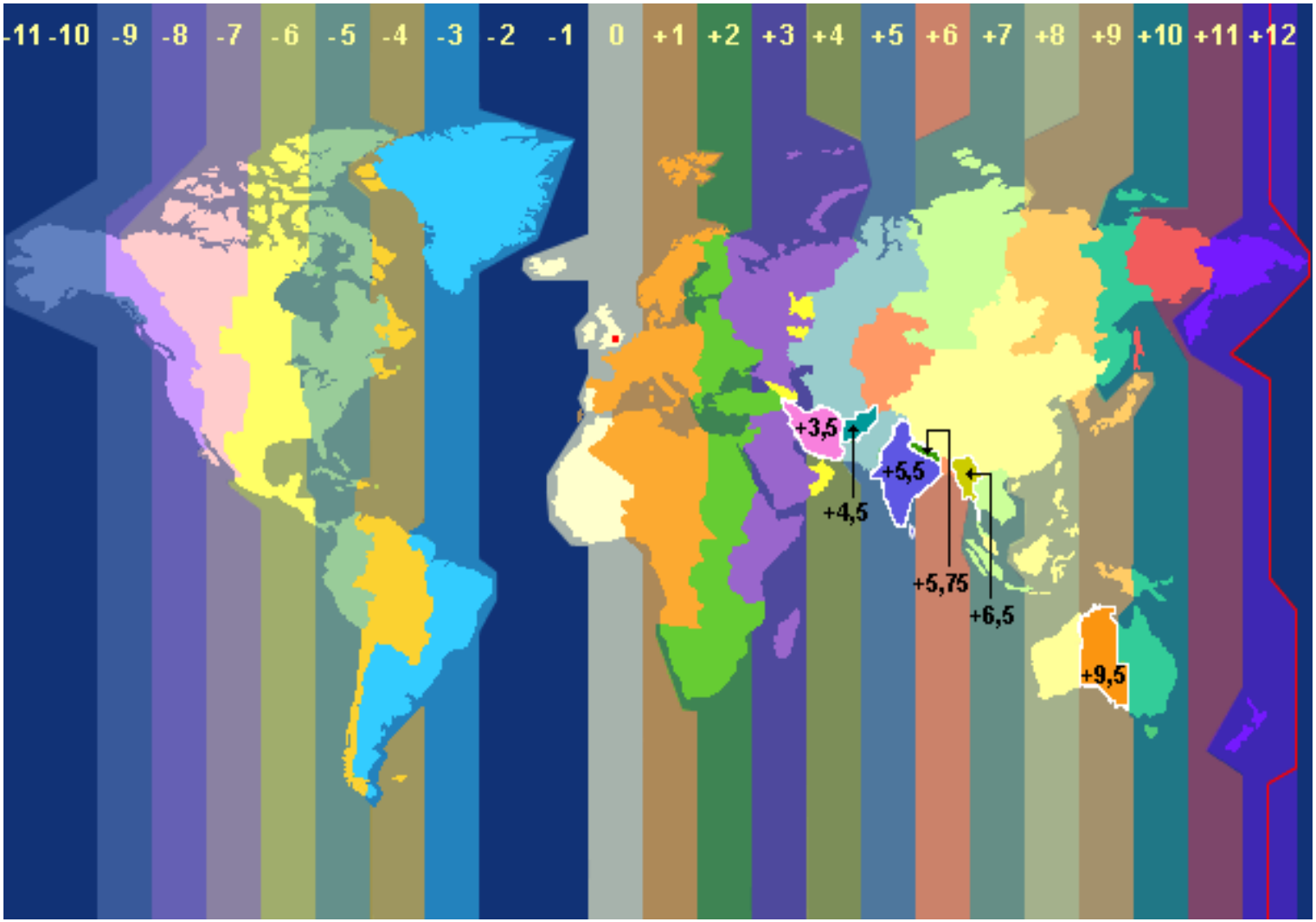
- **Motivation für Uhrensynchronisation**
  - Patriot Missile
  - Während des Golf Krieges am 25. Februar 1991 sollte eine Patriot Missile eine irakische Scud Missile abwehren. Die Patriot verfehlte aber ihr Ziel und die Scud Missile schlug in einer amerikanischen Kaserne ein, wobei 28 Soldaten getötet und 100 andere verletzt wurden. Zwei Wochen vor dem Vorfall haben die Amerikaner Daten von den Israeli erhalten, welche zeigten, dass nach 8 Stunden die Genauigkeit des Systems der Patriot nachlässt. Das Problem wurde bearbeitet und die neue Software wurde ins Kriegsgebiet geschickt. Dort kam sie aber erst ein Tag nach dem Unglück an. Das System der betroffenen Patriot Missile lief vor dem Abschuss 100 Stunden durchgehend und war deshalb bereits inakkurat.



- **Motivation für Uhrensynchronisation**
  - **Uhren in verteilten Systemen**
    - Sind unabhängig
    - Sind nicht 100% genau
    - Da jede Maschine/Rechner seine eigene Uhr hat, werden Ereignissen die nach einem anderen aufgetreten sind eventuell trotzdem einen neueren Zeitstempel erhalten
    - Wie stellt man nun die richtige Sequenz von Ereignissen sicher?
      - Notwendig ist eine globale Zeitbasis (Uhrensynchronisation)

- **Internationale Zeitstandards**
  - TAI (Temps Atomique International bzw. International Atomic Time)
    - Atomzeitskala, die zur Koordination nationaler Atomzeiten ermittelt wird
    - Beteiligung von 50 verschiedene Zeitinstituten mit ca. 250 Atomuhren
    - Zeit basiert auf der Atomsekunde
    - TAI ist chronoskopisch, d.h. eine Zeitskala ohne Diskontinuitäten
    - Referenzzeitpunkt ist der 1.Januar 1970
    - relative Genauigkeit von +/ - 10^{-15}
      - aber keine exakte Übereinstimmung mit der Sonnenzeit
      - Problem: “Atomtag” ist 3 msec kürzer als ein Sonnentag

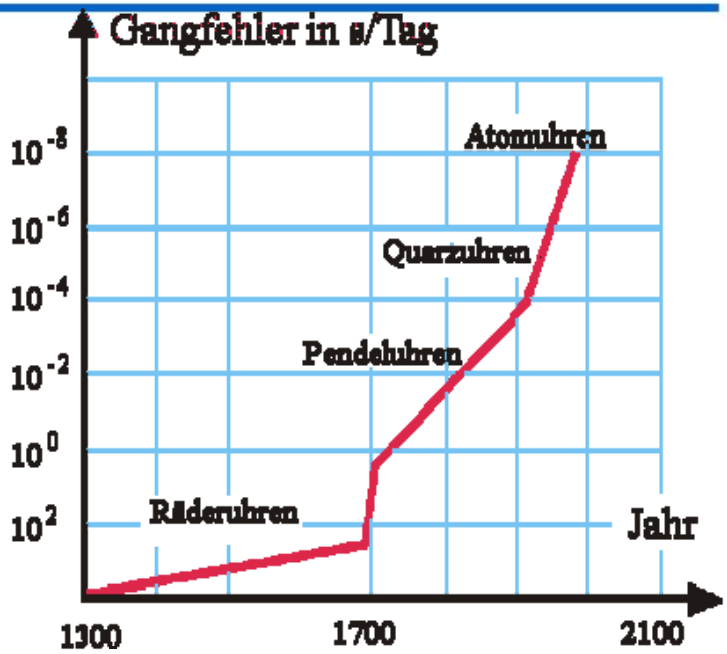
- **Internationale Zeitstandards**
  - UTC (Coordinated Universal Time)
    - realisiert durch Atomuhren, die Zeiteinheit ist die SI-Sekunde
    - hochkonstante Zeiteinheit aber nicht chronoskopisch
    - zusätzlich Übereinstimmung mit dem Sonnenlauf
    - einheitliche Grundlage zur Zeitbestimmung im täglichen Leben
    - Durch Einfügen von Schaltsekunden wird die UTC mit der universellen Sonnenzeit (UT1) synchronisiert (Unterschied zur Sonnenzeit 800 msec)
    - Anpassung erfolgt zumeist zu Ende oder Mitte des Jahres (typischer Abstand: alle 18 Monate)
    - Wird ausgestrahlt von Radio Stationen und Satelliten (GPS)
    - Problem: Verzögerungen bei der Übertragung müssen in Betracht gezogen werden (0.1-10 msec)


-11 -10 -9 -8 -7 -6 -5 -4 -3 -2 -1 0 +1 +2 +3 +4 +5 +6 +7 +8 +9 +10 +11 +12
+3,5
+4,5
+5,5
+5,75
+6,5
+9,5

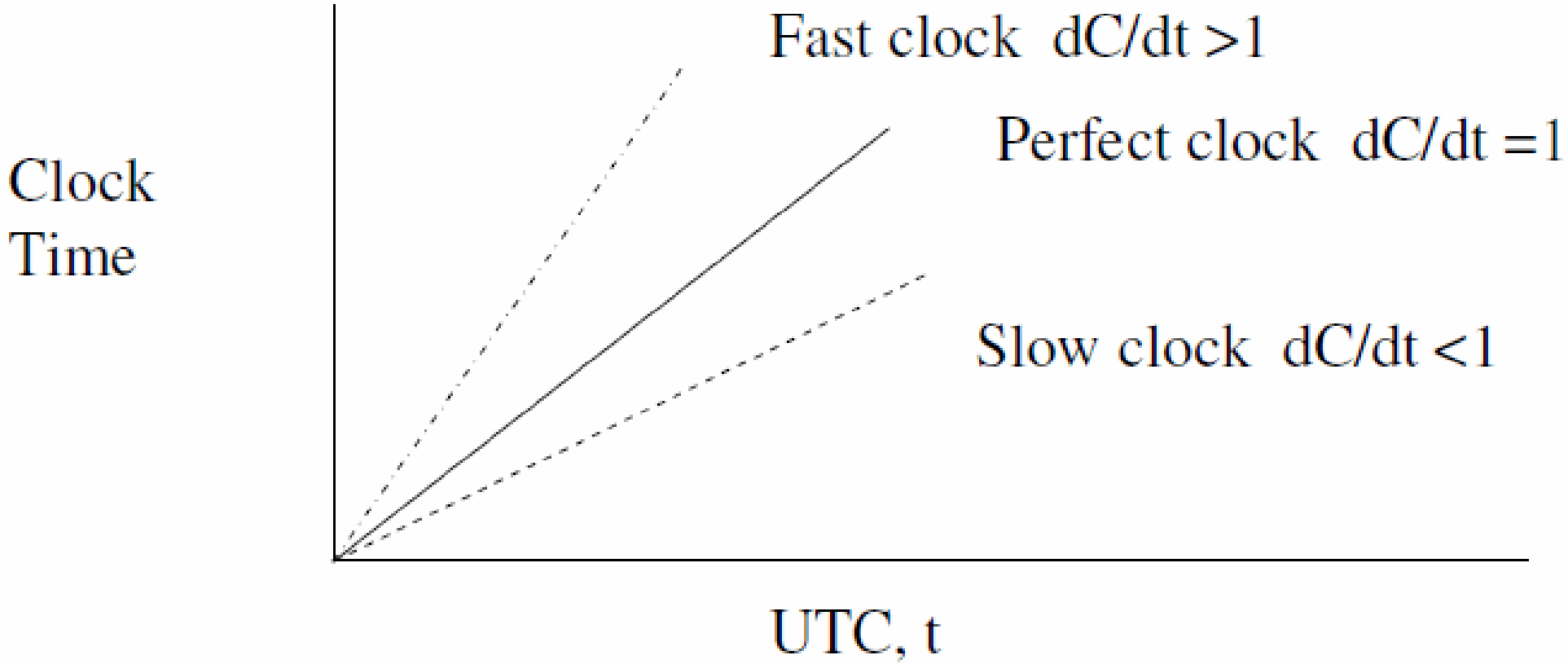
- **Internationale Zeitstandards**
  - Das PTB überträgt die aktuelle Uhrzeit über den Langwellensender DCF77
  - Versorgt funkgesteuerte Uhren in Westeuropa im Sekundenrhythmus
  - Zeitbasis am Senderstandort ist eine hochgenaue Spezialuhr welche mit den Atomuhren des PTB abgeblichen wird
  - Standardabweichung am Sendeort: 10^{-12}
  - Die Zeitinformationen werden als digitales Signal (negative Modulation Absenkung der Trägeramplitude) im Sekundentakt übertragen
  - '0' und '1' werden durch eine Absenkung um 100 ms bzw. 200 ms codiert. In der Sekunde 59 erfolgt keine Absenkung Markierung der Beginn einer neuen Minute bei nächster Amplitudenabsenkung
  - Pro Minute stehen somit 59 Bit zur Verfügung (wobei Bit 1-14 für Betriebsinformationen verwendet werden)


DCF77
500 km
1000 km
1500 km
2000 km

- **Genauigkeit von Uhren**
  - Eine Uhr arbeitet korrekt, wenn sie die vom Hersteller angegebene maximale Driftrate τ einhält, auch wenn sie dann etwas zu schnell oder zu langsam ist
  - Typische Driftraten:



Uhrentyp	Driftrate τ	Abweichung pro Jahr
Quarzuhr	10^{-5}	± 300 sec
Pendeluhr	10^{-6}	± 30 sec
Atomuhr	$1{,}5*10^{-14}$	± 0,5 Mikrosekunden
Atomuhr (lasergekühlte Atome)	10^{-15}	± 0.03 Mikrosekunden


Fast clock dC/dt >1
Perfect clock dC/dt =1
Slow clock dC/dt <1
Clock
Time
UTC, t

- **Uhrenverhalten**
  - Korrekt:
    - Absolutwert der Abweichung kleiner der zugesicherten Gangabweichung
  - Fehlerbehaftet:
    - Überschreiten der zugesicherten Gangabweichung
    - Zustandsfehler (z.B. Sprung im Zählerwert)
    - Stehenbleiben der Uhr
  - Unmöglich:
    - Rückwärtslaufende Uhr
    - Unendlich schnell laufende Uhr
  - Gangabweichung zweier korrekter Uhren kann beliebig groß werden, falls die Uhren nicht synchronisiert werden

- **Zeit als Modell**
  - kann durch unendliche Menge T von Zeitpunkten („Instants“) modelliert werden
    - T ist dann eine geordnete Menge
    - d.h. Falls p und q zwei Zeitpunkte sind so treten sie entweder:
      - Gleichzeitig oder
      - Eines nach dem anderen ein → p vor q oder q vor p (mutually exclusive)
    - Ordnung zwischen den Zeitpunkten heißt temporale Ordnung
    - T ist dicht
      - zwischen zwei Zeitpunkten p und r liegt zumindest ein weiterer Zeitpunkt q, dann und nur dann wenn p und r nicht gleichzeitig sind.
    - Intervall aus T heißt Zeitraum oder Dauer

- **Zeit als Modell**
  - (Verteilte) Echtzeitsysteme besteht aus einer Menge an Prozessen (States/Values) die miteinander kommunizieren/sich verändern (Events/Actions)
  - Sind temporal geordnet nach Zeitpunkt ihres Eintretens (partielle Ordnung) bzw. total geordnet durch Anfügen einer Prozess-ID
  - finden zu bestimmten Zeitpunkten statt und haben keine Dauer.
  - Sind eventuell, aber nicht notwendigerweise, kausal geordnet

E1: Jemand betritt einen Raum

E2: Das Telefon läutet

→ Fall 1: E1 nach E2 kausale Abhängigkeit möglich

→ Fall 2: E2 nach E1 kausale Abhängigkeit unwahrscheinlich

→ E1 vor E2 ist ist eine notwendige, aber nicht ausreichende, Bedingung

- **Zeit als Modell**
  - Betrachtung einer Referenzuhr mit sehr kleiner Granularität (z.B. 10^{-15} sec)
    - Absoluter Zeitstempel für ein Ereignis: clock(e)
    - Der Zeitpunkt des Eintretens von Ereignis e gemessen mit der Referenzuhr clock wird mit clock(e) notiert und liefert einen absoluten Zeitstempel für das Ereignis e
    - Dauer zwischen zwei Ereignissen ist die Differenz ihrer Zeitstempel, also die Anzahl der Mikroticks auf der Referenzuhr clock, die zwischen den zwei Ereignissen aufgetreten sind
    - Granularität einer Uhr k ist gegeben durch die nominale Anzahl von Mikroticks der Referenzuhr clock zwischen zwei Mikroticks der Uhr k

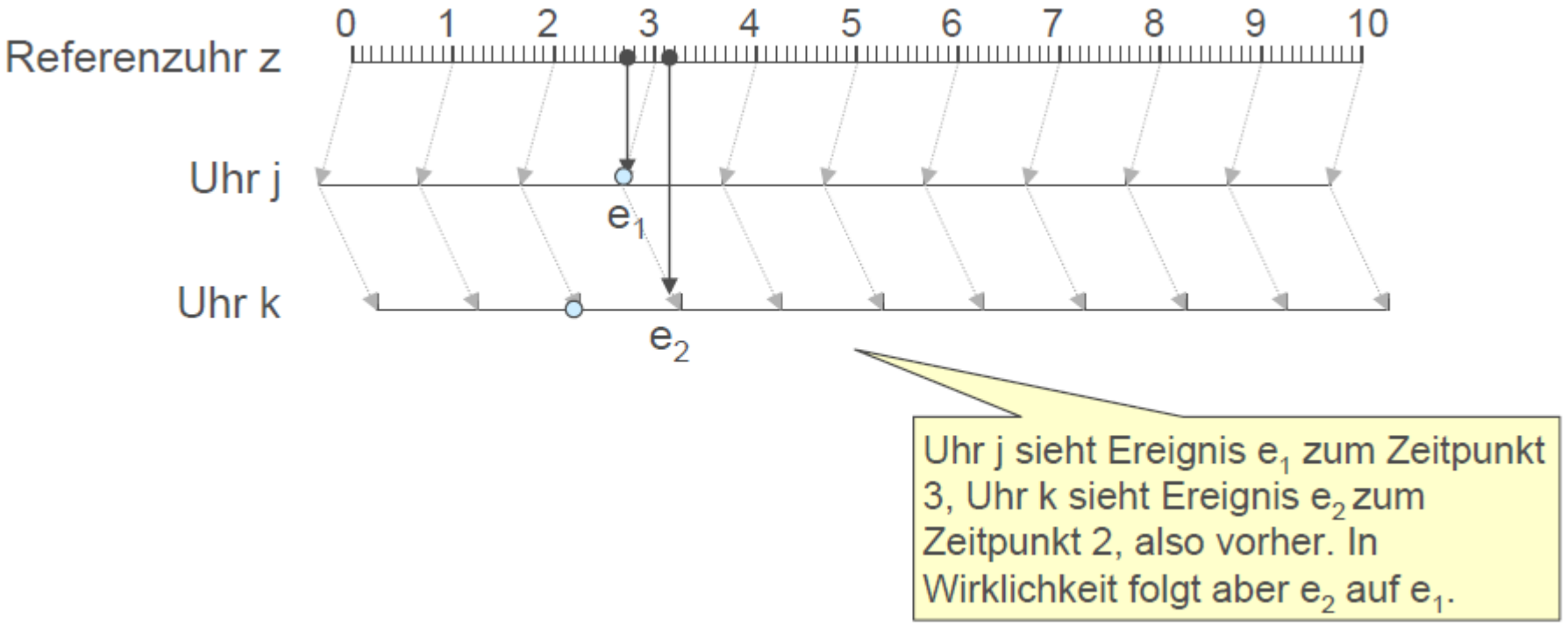
- **Zeit als Modell**
  - Clock Drift und Drift Rate
    - Die Abweichung einer Tochteruhr k von einer Referenzuhr clock bezeichnet man als clock drift
    - **$Drift^{i}_{k}$ = [clock($microtick^{i+1}_{k}$) – clock($microtick^{i}_{k}$)] / n_k**
    - wobei n_k die nominale Anzahl von Mikroticks der Referenzuhr clock zwischen 2 aufeinanderfolgenden Mikroticks der Tochteruhr k angibt
    - Eine perfekte Uhr hätte einen clock drift von 1
    - Alternativ kann auch die Abweichungsrate (drift rate) einer Tochteruhr k gegeben sein:
    - **P^{i}_{k} = | $drift^{i}_{k}$ – 1 |**
    - Eine perfekte Uhr hätte einen Abweichungsrate von 0
    - Reale Uhren haben typischerweise Abweichungsraten von 10^{-7} bis zu 10^{-2} sec/sec

- **Zeit als Modell**
  - Abweichung (offset)
    - Man betrachte zwei Uhren j und k mit gleicher Granularität
    - Die Differenz der zwei Uhren gemessen in Mikroticks der Referenzuhr wird offset genannt
      - **$Offset^i_{jk} = | clock(microtick^i_j) – clock(microtick^i_k) |$**
  - Präzision (precision)
    - Man betrachte eine Menge von Uhren {1,2, ...,n} dann ist
      - **$i = max_{1≤j,k≤n} \{ offset^i_{jk} \}$**
    - die Präzision der Uhrenmenge zum Mikrotick i
  - Das Maximum aller Π^i über einen bestimmten Beobachtungszeitraum hinweg wird Präzision Π der Uhrenmenge genannt
  - Werden die Uhren innerhalb der Menge nicht regelmäßig untereinander synchronisiert, so werden sie immer stärker voneinander abweichen („intern“)

- **Zeit als Modell**
  - Genauigkeit (accuracy)
    - Der offset einer Uhr k zur Referenzuhr zu Mikrotick i wird mit $accuracy^i_k$ bezeichnet
    - Das Maximum der Werte $accuracy^i_t$ über einen bestimmten Beobachtungszeitraum wird Genauigkeit der Uhr k ($accuracy_k$) genannt
    - Um Sicherzustellen, dass die Genauigkeit einer Uhr innerhalb bestimmter Schranken bleibt, muss sie periodisch mit der Referenzuhr synchronisiert werden (externe Synchronisation)
  - Beachte:
    - Werden alle Uhren einer Uhrenmenge regelmäßig mit der Referenzuhr innerhalb einer Genauigkeit A synchronisiert, dann ist die Menge dieser Uhren intern mit Präzision 2A synchronisiert

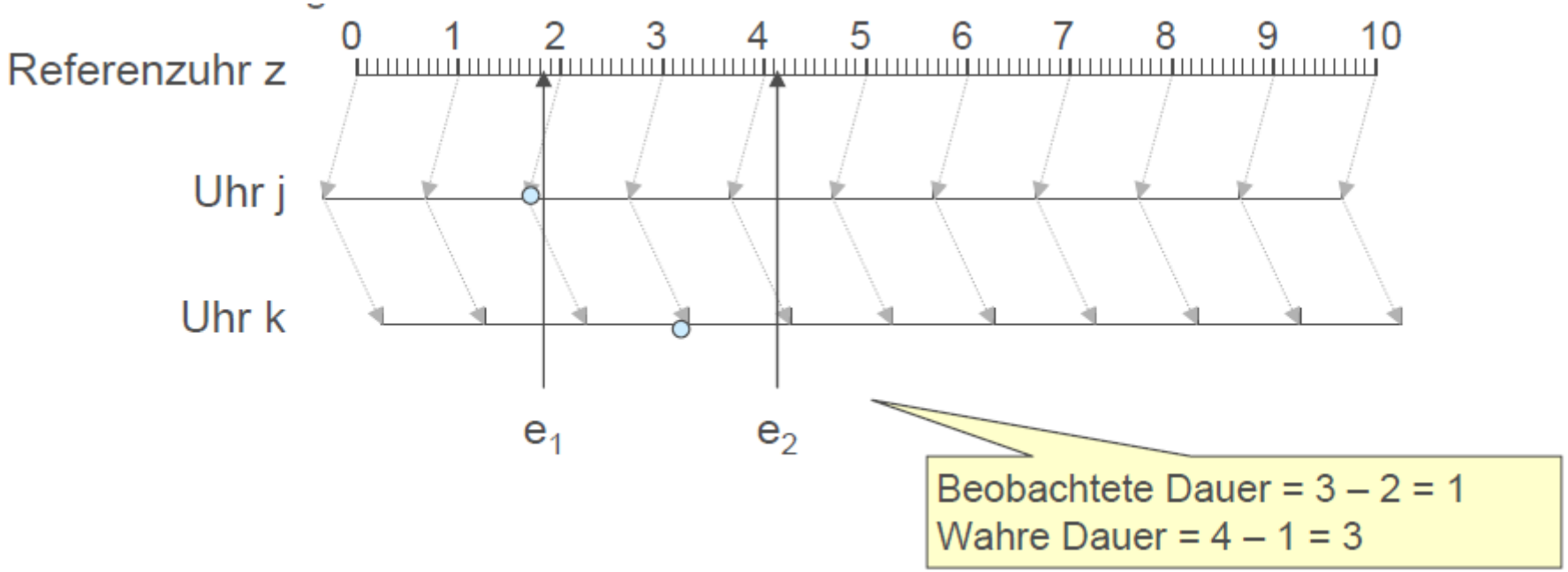
- **Globale Zeit**
  - Annahme:
    - Man betrachte eine Menge von Knoten in einem verteilten System
    - Jeder Knoten hat seine lokale Uhr c_k mit Granularität g_k
    - Die Menge aller lokalen Uhren ist intern mit Präzision Π synchronisiert, i.e.
      - **$| clock(microtick^i_j) - clock(microtick^i_k) | < \Pi$ □ i,j,k**
  - Lokale Implementierung der globalen Zeit
    - Definiere jeden n.ten Mikrotick einer Uhr als Makrotick der globalen Zeit

- **Reasonableness Condition**
  - Wenn für alle lokalen Implementierungen der globalen Zeit gilt, dass die Präzision Π der Uhren (interne Synchronisation) durch die Granularität g der globalen Zeit beschränkt ist (**g >** Π), d.h. der offset zweier beliebiger Uhren gemessen in Microticks kleiner als die Anzahl der Microticks zwischen zwei Macroticks der globalen Uhr ist, dann heißt das System „vernünftig“ (reasonable) und es gilt weiters
    - **$| t_j(e) - t_k(e) | \leq 1$**,
  - i.e. die globalen Zeitstempel eines Ereignisses ($t_j(e)$ bzw. $t_k(e)$) , das von zwei verschiedenen lokalen Uhren aufgezeichnet wird, können sich maximal um 1 unterscheiden
  - Beachte:
    - ein garantierter Unterschied von 0 kann nie erreicht werden, da es keine perfekte Synchronisation gibt und wegen der Dichte der Zeitfunktion gilt dass immer folgender Fall eintreten kann: Uhr j tickt, Ereignis e tritt ein, Uhr k tickt

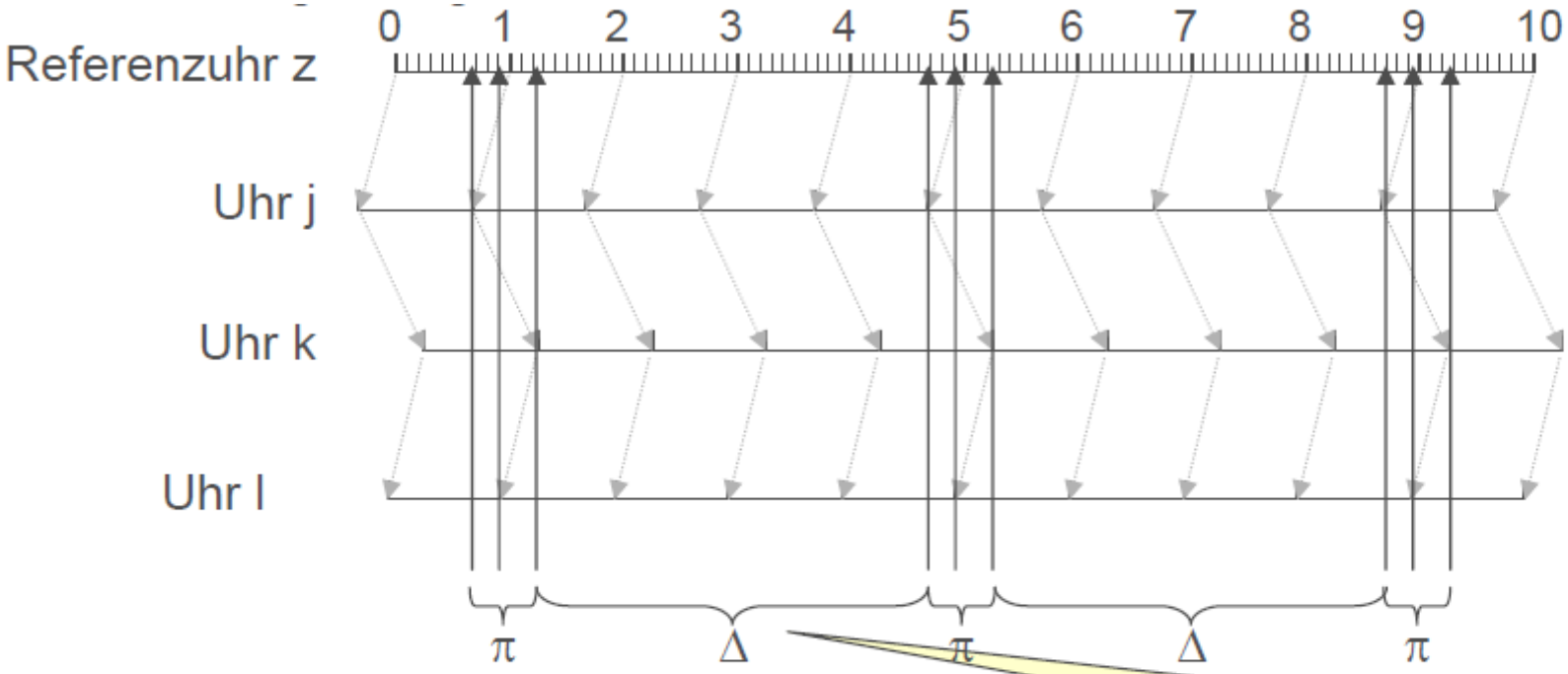

Referenzuhr z
0
1
2
3
4
5
6
7
8
9
10
Uhr j
e1
Uhr k
e2
Uhr j sieht Ereignis e1 zum Zeitpunkt 3, Uhr k sieht Ereignis e2 zum Zeitpunkt 2, also vorher. In Wirklichkeit folgt aber e2 auf e1.

- Reasonableness Condition
  - Implikation für die Ermittlung der zeitlichen Reihenfolge zweier Ereignisse
  - Zwei Ereignisse die einen um 1 unterschiedlichen Zeitstempel besitzen können nicht geordnet werden
  - Unterscheiden sich aber die Zeitstempel zweier Ereignisse um mehr als 1, so ist die zeitliche Ordnung eindeutig feststellbar

- Reasonableness Condition
  - Implikation für die Messung von Zeitintervallen
  - Ein Intervall wird durch das Eintreten zweier Ereignisse bestimmt.
  - Daher „pflanzt“ sich die Ungenauigkeit der Ermittlung des Eintrittszeitpunktes für ein Ereignis fort und die Ungenauigkeit bei der Messung der Dauer eines Intervalls ist durch 2g beschränkt.
  - Für die wahre Dauer eines Intervalls $\mathbf{d_{true}}$ gilt also
    - $\mathbf{(d_{observed} - 2g) < d_{true} < (d_{observed} + 2g)}$
  - wobei $d_{observed}$ die beobachtete Differenz der Zeitstempel des Anfangs- und Endereignisses ist


Referenzuhr z
0
1
2
3
4
5
6
7
8
9
10
Uhr j
Uhr k
e1
e2
Beobachtete Dauer = 3 – 2 = 1
Wahre Dauer = 4 – 1 = 3

- π/Δ-Reihenfolge
  - Man betrachte ein System, in dem jeder Knoten zum Zeitpunkt 1, 5, 9, … Ereignisse generiert
  - Zwei Ereignisse e1 und e2 heißen π/Δ-präzedent, wenn für sie gilt
    - $[\ |\ z(e_i) - z(e_j)\ |\ \leq \pi\ ] \Rightarrow [\ |\ z(e_i) - z(e_j)\ |\ > \Delta\ ]$

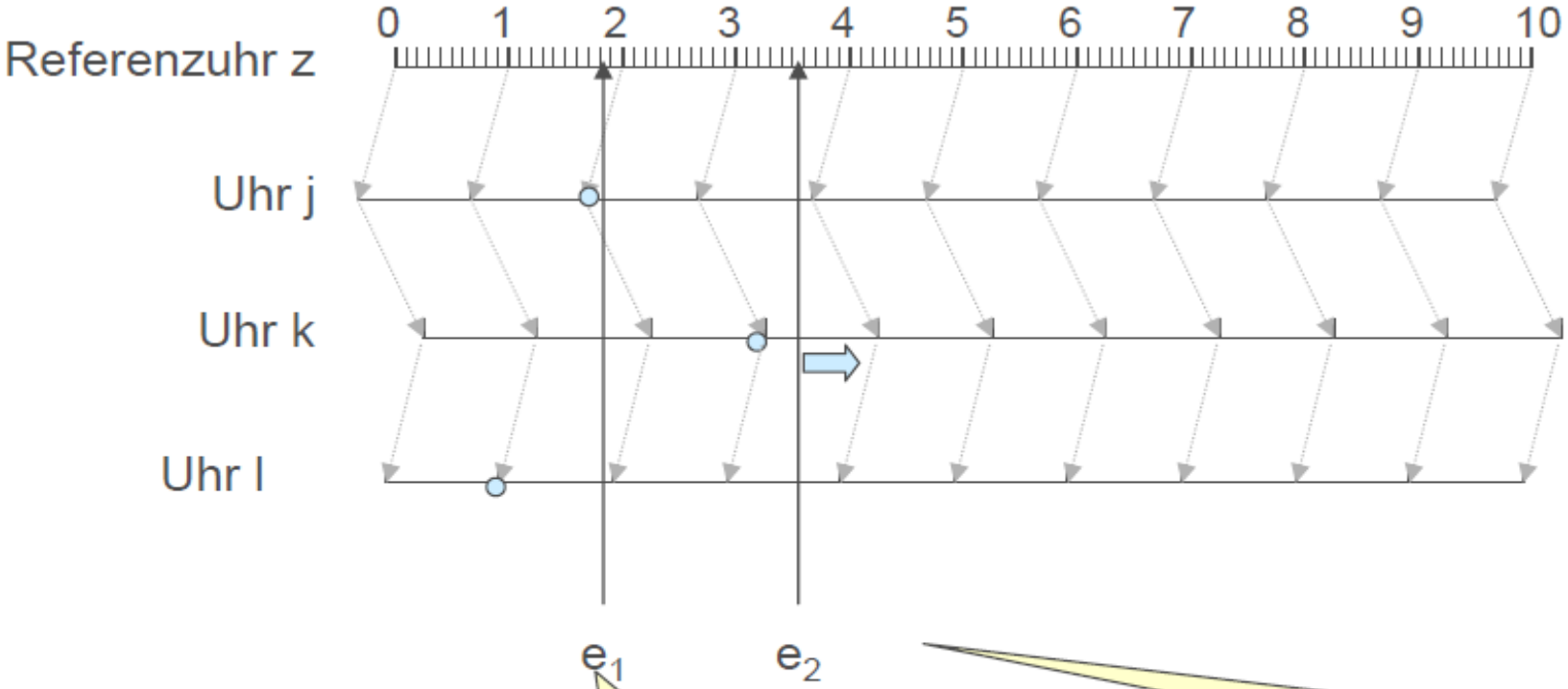


„Externer Beobachter“ sieht Menge von Ereignisse innerhalb eines kurzen Intervalls gefolgt von einer längeren Pause

- Sonderfall: π = 0
  - Dann sind e1 und e2 entweder zum gleichen Zeitpunkt aufgetreten oder mit mindestens Δ Zeiteinheiten Unterschied
  - Man betrachte eine globale Zeitbasis mit Granularität g und zwei von unterschiedlichen Knoten beobachtete Ereignisse e1 und e2, dann gilt

Event Set	$D_{observed}$ zweier nicht-gleichzeitiger Ereignisse	Zeitliche Reihenfolge feststellbar
0/1g präzedent	$\lvert t^j(e_1) - t^k(e_2) \rvert \geq 0$	nein
0/2g präzedent	$\lvert t^j(e_1) - t^k(e_2) \rvert \geq 1$	nein
0/3g präzedent	$\lvert t^j(e_1) - t^k(e_2) \rvert \geq 2$	ja
0/4g präzedent	$\lvert t^j(e_1) - t^k(e_2) \rvert \geq 3$	ja

- Problem: rasch aufeinanderfolgende Ereignisse – „dense time“ Reihenfolge kann nicht eindeutig ermittelt werden, Knoten ziehen inkonsistente Schlüsse



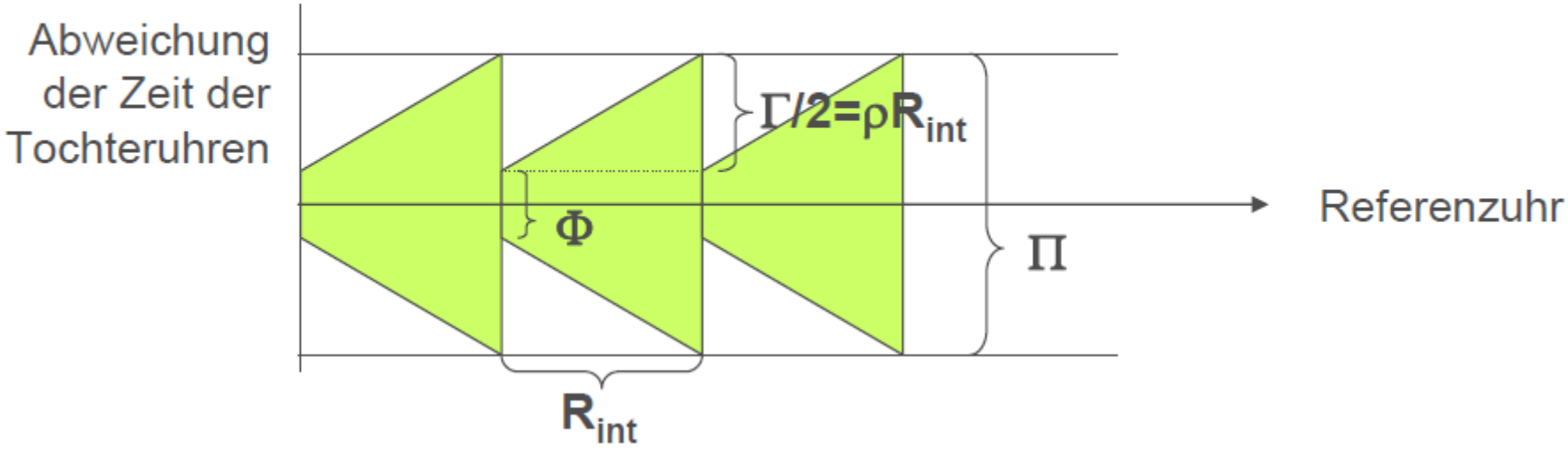
- Problem: rasch aufeinanderfolgende Ereignisse – „dense time“
  - Lösung: Verwendung eines Agreement-Protokolls, das zwar nicht korrekte zeitliche Ordnung garantieren kann, aber zumindest zu konsistentem System führt
    - Phase 1: alle Knoten tauschen Systeminformation aus und haben nach dem Austauschschritt identische Information über das System
    - Phase 2: jeder Knoten ermittelt nach deterministischen Verfahren auf Basis der identischen Information gemeinsamen, konsistenten Systemzustand
    - Nachteil des Agreement-Protokolls:
      - Mehraufwand durch globalen Systeminformationsaustausch

- Sparse Time Base
  - Ereignisse treten nur innerhalb bestimmter Intervalle auf gefolgt von längeren Phasen ohne Ereignisauftritte („spärlich“, „sparse“)
  - Man betrachte ein verteiltes System mit einem Cluster A, in dem Ereignisse generiert und einem Cluster B, der das Auftreten dieser Ereignisse beobachtet
  - Frage: Wie „spärlich“ müssen Ereignisse auftreten, sodass zeitliche Reihenfolge ohne Agreement-Protokoll vom Cluster B bestimmt werden kann?
  - Antwort:
  - Cluster A muss 1/4g Präzedenzmenge von Ereignissen erzeugen, d.h. innerhalb eines cluster-weiten Ticks treten Ereignisse gleichzeitig auf, in den folgenden 4 Ticks werden keine Ereignisse generiert
  - Cluster B klassifiziert Ereignisse, die sich um zwei oder weniger Ticks unterscheiden, als gleichzeitig, bei Unterschied ab 3 Ticks kann zeitliche Reihenfolge eindeutig bestimmt werden
  - Kann für Ereignisse verwendet werden, die innerhalb der Einflußsphäre des Systems erzeugt werden, nicht aber für externe Ereignisse

- Zusammenfassung:
  - In einem verteilten System mit reasonable global time von Granularität g gilt
    - Wenn ein einzelnes Ereignis von zwei unterschiedlichen Knoten beobachtet wird, so kann immer der Fall eintreten, dass sich die Zeitstempel um nur einen Makrotick unterscheiden.
    - Die zeitliche Reihenfolge von zwei Ereignissen kann ermittelt werden, wenn sich ihre Zeitstempel um mindestens 2 Makroticks unterscheiden.
    - Die zeitliche Reihenfolge von Ereignissen kann immer ermittelt werden, wenn die Menge der Ereignisse zumindest 0/3g präzedent sind.
    - Wenn die beobachtete Dauer eines Intervalles $d_{observed}$ ist, dann ist die tatsächliche Dauer beschränkt durch
    - $\mathbf{(d_{observed} - 2g) < d_{true} < (d_{observed} + 2g)}$.
  - Voraussetzung für das Konzept der globalen Zeit sind interne Synchronisationsverfahren, die sicherstellen, dass die lokalen Uhren eine Präzision von zumindest g haben.

- Synchronisationsbedingung
  - Man betrachte zwei Uhren mit einer Abweichungsrate von ρ, die im Intervall von R_{int} synchronisiert werden, dann ist die maximale Zeitdifferenz zwischen den Uhren beschränkt durch
    - $\Gamma = \mathbf{2}\ \rho\ \mathbf{R_{int}}$
  - Beachte: (Γ ... Drift Offset)
  - Synchronisation kann Zeitdifferenz nicht wirklich auf 0 zurückführen.
  - Sei Φ die unmittelbar nach einer Synchronisation verbleibende Zeitdifferenz und die Präzision Π die maximal erlaubte Zeitdifferenz zwischen zwei Uhren, dann muss zur Wahrung der Präzision gelten:

$$\Phi + \Gamma \leq \Pi$$



- Synchronisation mit zentraler Uhr
  - Vorgehensweise
    - Masterknoten sendet periodisch Zeitmeldungen an alle anderen Knoten
    - Differenz zwischen lokaler Zeit und übermittelter Zeit abzüglich der Laufzeit der Nachricht ergibt Korrekturwert
    - Präzision des Algorithmus ist bestimmt durch Variation in der Laufzeit

    (jitter ε)

      - **central =** ε **+** Γ
  - Nachteil:
    - Hoher Kommunikationsaufwand, wird meist nur in der Startphase eines verteilten Systems verwendet

- Verteilte Synchronisation
  - Grundprinzip
    - Phase 1: alle Knoten sammeln Zeitinformation von anderen Knoten
    - Phase 2: jeder Knoten berechnet aus Gesamtinformation Korrekturwert für lokale Uhr
    - Phase 3: lokale Uhren werden gestellt
  - Ad Phase 1
    - Genauigkeit hängt von Jitter des Nachrichtenaustausches statt
    - Dieser hängt wiederum auf der Systemeebene ab, auf der Nachrichtenaustausch durchgeführt wird
    - Z.B. auf Anwendungsebene 0,5-5 msec, auf Ebene der Hardware des Kommunikationscontrollers aber weniger als 0,01 msec
    - Nach [Lundelius & Lynch 1984] ist Präzision einer Menge von N Uhren beschränkt durch $\Pi = \varepsilon$ **$(1 - 1/N)$**

- Verteilte Synchronisation
  - Ad Phase 2:
    - Eine Uhr die in der Phase 1 unterschiedliche Zeitwerte zu anderen Uhren sendet, kann bewirken, dass in der Phase 2 keine Synchronisation erzielt werden kann
  - Diese Art von fehlerhaften Uhren nennt man malicious clocks oder byzantinischen Fehler (byzantine error)
  - Mögliche Lösung zur Toleranz von k fehlerhaften Uhren (FTA – Fault-tolerant average)
  - Alle erhaltenen Zeitwerte werden der Größe nach sortiert
  - Jeweils die k kleinsten und größten Werte werden eliminiert, der Mittelwert der verbleibenden Werte N-2k ergibt globale Zeit
  - Präzision
    - $\Pi = (\varepsilon + \Gamma) (N-2k) / (N-3k)$
  - Es lässt sich zeigen, dass Korrektur von k fehlerhaften Uhren zumindest N = 3k+1 Uhren erfordert

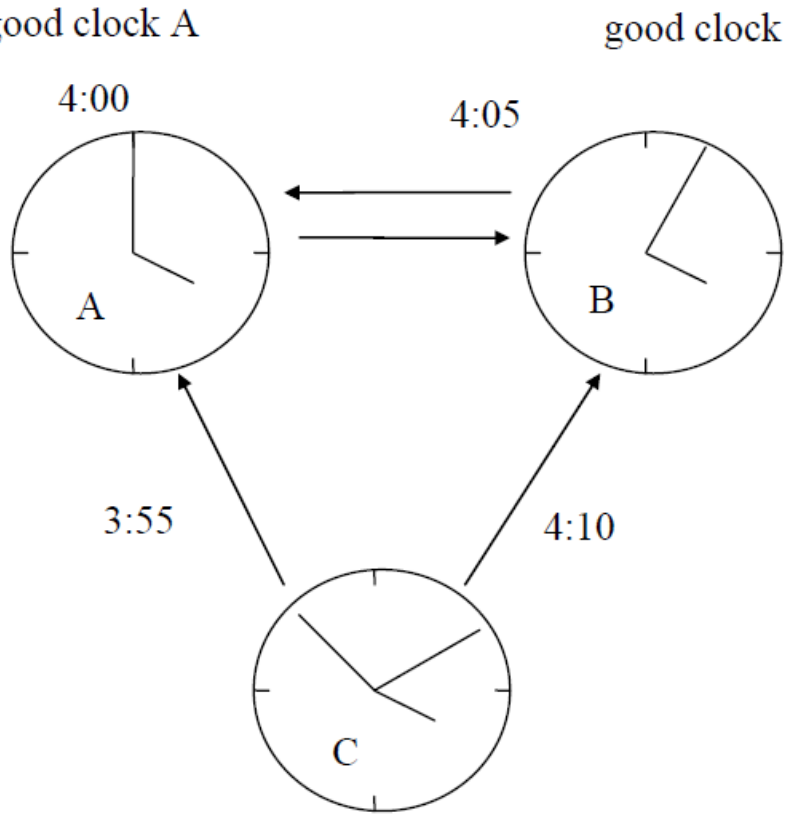
- Byzantine Error

View at A

B 4:05

A 4:00

C 3:55



View at B

A 4:00

B 4:05

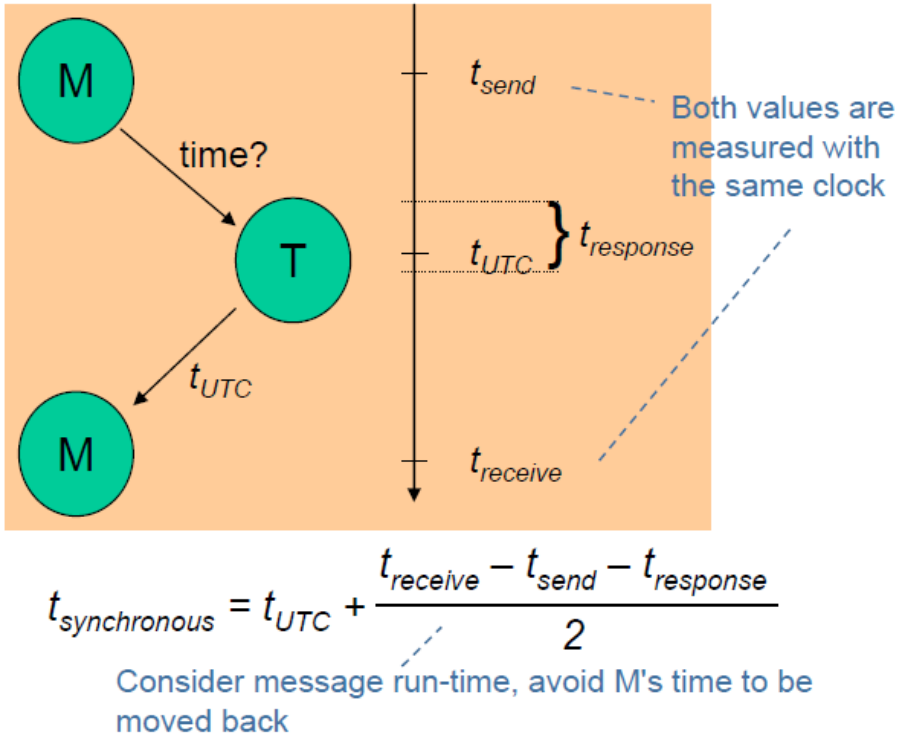
C 4:10

„two faced“ malicious clock C

- Verteilte Synchronisation
  - Ad Phase 3:
    - Zeitkorrektur kann erreicht werden durch
    - direktes Setzen („state correction“)
    - Probleme, da Sprünge in der lokalen Zeit auftreten können
    - Bei Rücksprung würde ein Zeitpunkt zweimal auftreten oder durch Anpassen der Geschwindigkeit der lokalen Uhr („rate correction“)
  - Innerhalb der erlaubten drift rate:
    - Durchschnittliche Korrektur über alle Uhren hinweg sollte null sein, um zu vermeiden, dass alle Uhren tendenziell schneller / langsamer werden

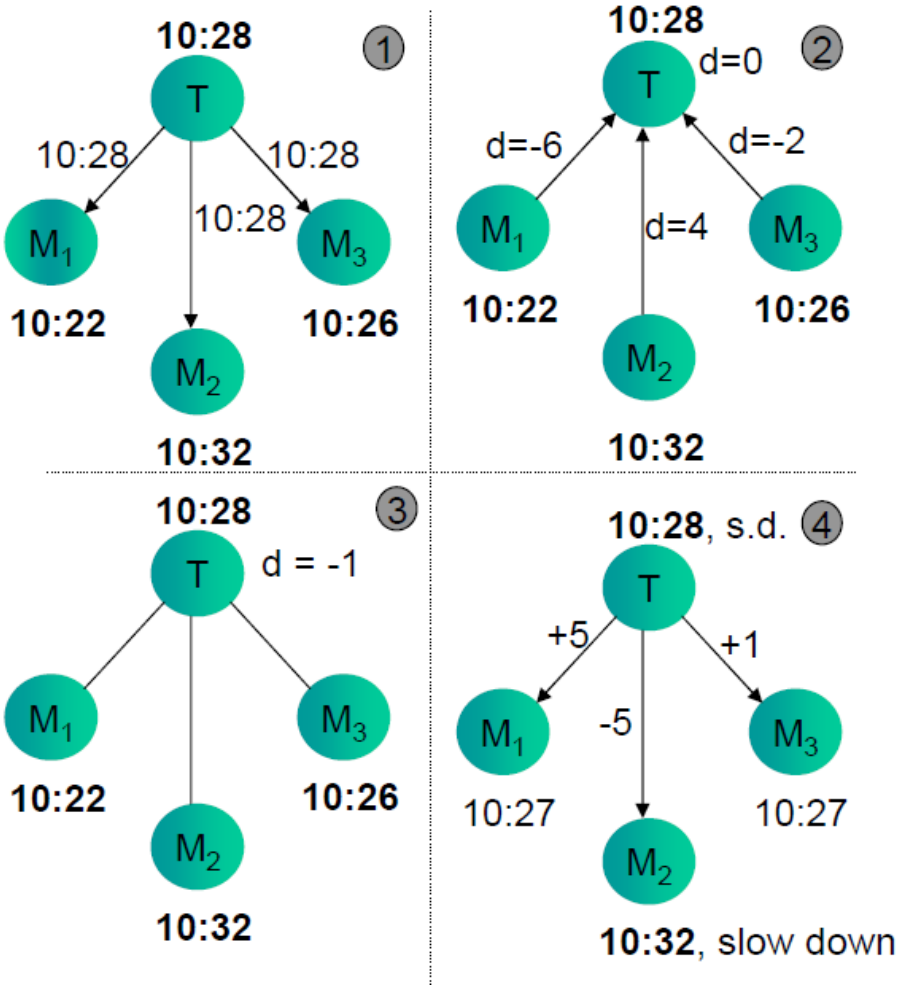
- **Uhreinsynchronisation (Algorithmen)**
  - **Algorithmus von Cristian (1989)**
    - Das Verfahren basiert auf verteilter, externer Synchronisation
    - Innerhalb des Systems existiert ein Time-Server, zumeist ein UTC-Empfänger
    - In regelmäßigen Abständen senden die anderen Einheiten einen Time-Request, der so schnell wie möglich vom Server beantwortet wird

- Rechner M fragen bei T an
- Anfrage wird von T möglichst schnell bearbeitet
- M berechnet dann die Zeit
  - Zeitstempel für die Anfrage
  - Zeitstempel wenn das Ergebnis eintrifft
  - Substrahieren der Response-Zeit
  - Durch 2 Teilen
  - Addiere t_{UTC} (Zeit die für das Senden benötigt wurde)
- $t_{synchronous}$ wird die neue Systemzeit

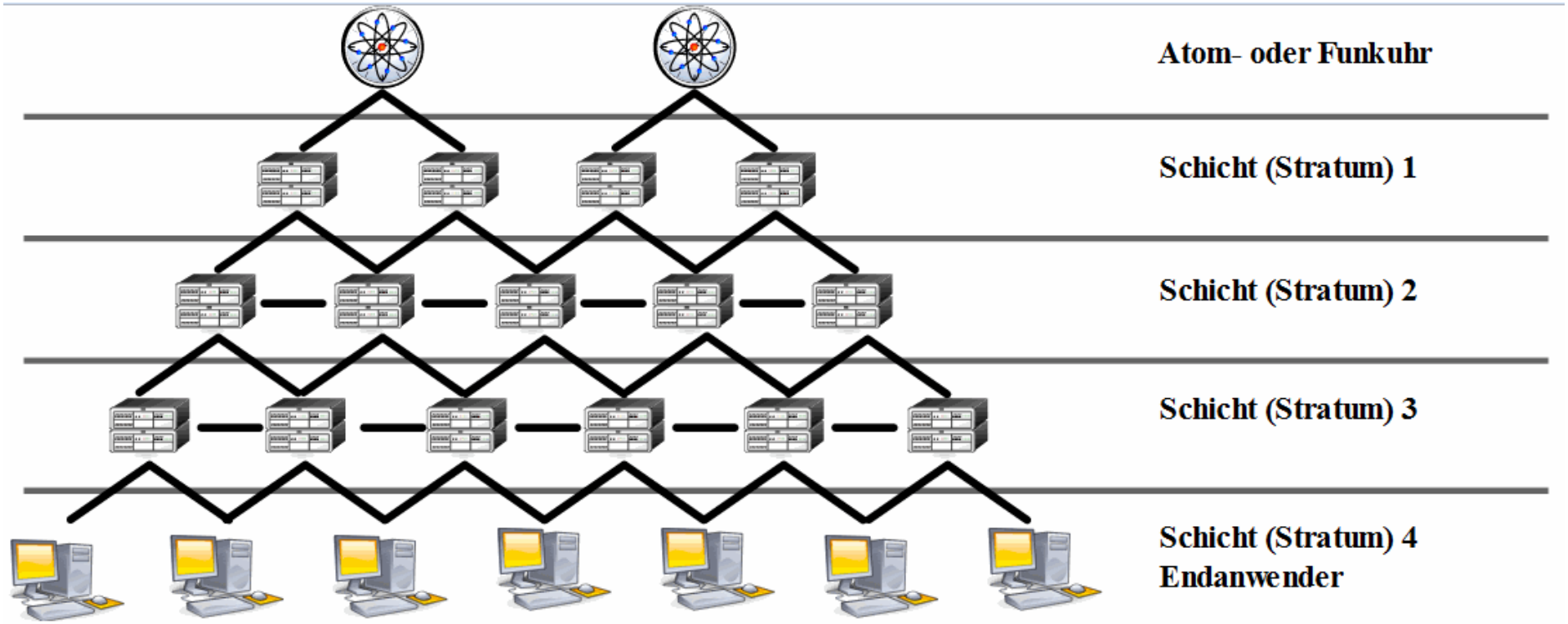


- **Uhreinsynchronisation (Algorithmen)**
  - **Algorithmus von Berkeley (1989)**
    - Annahme: kein UTC-Empfänger verfügbar
    - Algorithmus (zentral, intern):
    - ein Rechner agiert als aktiver Time-Server
    - Der Server fragt periodisch die Zeiten/Unterschiede aller anderen Rechner ab (Phase 1) und ermittelt den Durchschnittswert (Phase2)
    - In Phase 3 wird der errechnete Wert an alle anderen Uhren ausgegeben

- **Uhreinsynchronisation (Algorithmen)**
  - **Algorithmus von Berkeley (1989)**
  - Zeitserver T sendet seine eigene Zeit an alle anderen Rechner
  - Rechner antworten mit der Abweichung zur eigenen Zeit
  - T summiert alle Abweichungen auf und teilt durch die Anzahl der Maschinen (sich selbst eingeschlossen)
  - Neue Zeit für alle Rechner ist durch den Durchschnittswert gegeben
  - Wichtig: schneller laufende Uhren werden nicht zurückgestellt sondern verlangsamt



- **Uhreinsynchronisation (Algorithmen)**
  - **NTP: Network Time Protocol (1982)**
    - Problem: Die angegebenen Algorithmen funktionieren nur in kleinen statischen Netzen
    - Das NTP Protokoll bietet eine Möglichkeit in großen Netzen eine Synchronisation zu gewährleisten
    - Die Netze können dabei dynamisch konfiguriert werden, um eine zuverlässige Synchronisation zu gewährleisten
    - Die Grundstruktur von NTP ist ein hierarchisches Modell (mit verschiedenen Strata/Schichten)
    - Der Dienst wird durch ein verteiltes Serversystem geleistet
    - Primäre Server sind direkt mit einer UTC-Quelle verbunden
    - Sekundäre Server synchronisieren sich mit primären Servern usw
    - Jede zusätzliche Schicht verursacht einen zusätzlichen Zeitversatz von 10-100 ms


Atom- oder Funkuhr
Schicht (Stratum) 1
Schicht (Stratum) 2
Schicht (Stratum) 3
Schicht (Stratum) 4
Endanwender

THE END

Fragen?