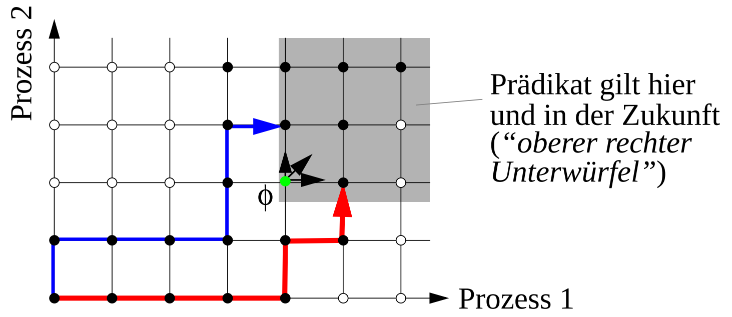
Stabile Prädikate

- Informell: Monoton - "einmal wahr, immer wahr"

- Def.: wenn c1 < c2, dann $\Phi(c1) \Longrightarrow \Phi(c2)$

Halbordnung im Zustandsverband



- Sind *beobachterunabhängig* (possibly Φ = definitely Φ)!
  - jeder Beobachter muss durch den oberen rechten Würfel (Fairness...)
  - lassen sich daher einfach mit einer einzigen Beobachtung feststellen (jede andere Beobachtung wird Φ früher oder später ebenfalls entdecken)

- Für zwei Beobachtungen B_1, B_2 gilt: Falls B_1 Φ "entdeckt", dann gibt es einen *gemeinsamen späteren Zustand* (Verbandseigenschaft!) von B_1, B_2, bei dem Φ gilt
  - spätestens der Endzustand (bei endlichen Berechnungen)
  - B_1 kann z.B. die "echte" Ereignisfolge in Realzeit sein, B_2 eine Beobachtung

- Ein *gelegentlicher* (konsistenter) Schnappschuss genügt!
  - wenn der Schnappschussalgorithmus die Gültigkeit von Φ ermittelt, dann gilt Φ "jetzt" tatsächlich
  - wenn Φ "jetzt" gilt, dann meldet dies ein (jetzt gestarteter) Schnappschussalg.

- Es gibt wichtige stabile Prädikate, z.B. Terminierung, Garbage, Deadlock...
- Aber woher weiss man eigentlich, ob bzw. dass ein Prädikat stabil ist?

Logische Zeit in verteilten Systemen

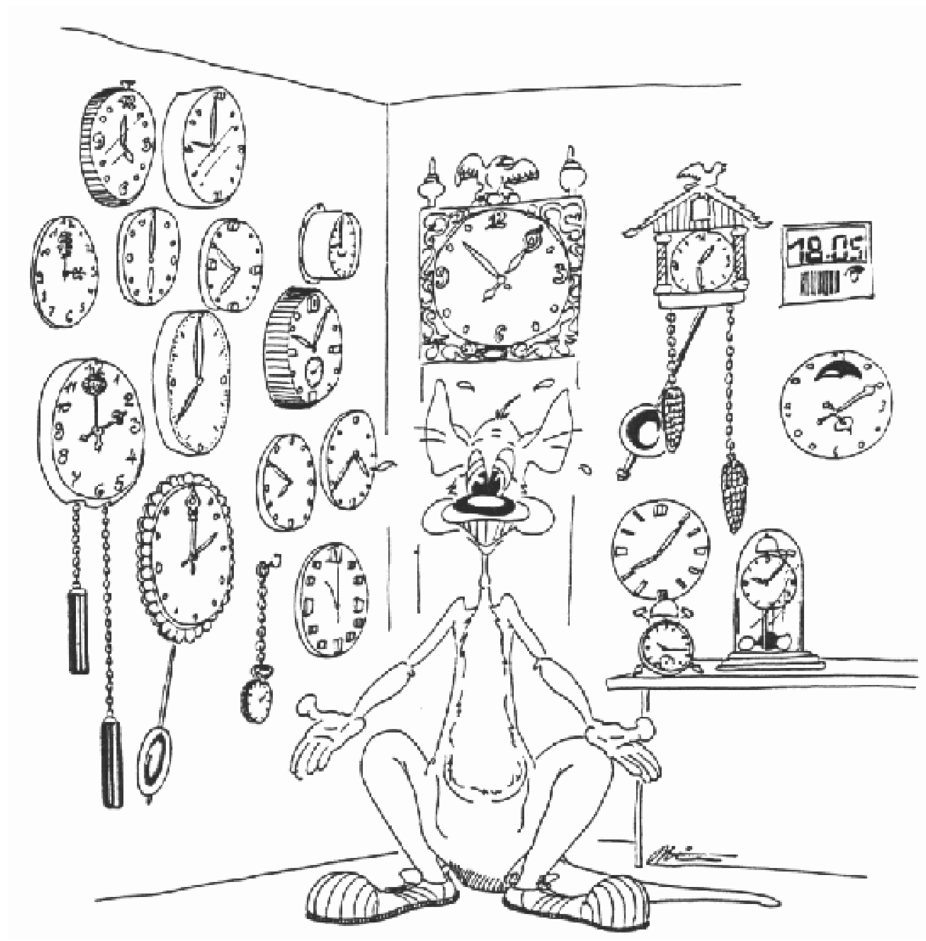


Bild: R. G. Herrtwich, G. Hommel

Zeit ?

Quid est ergo tempus?
Si nemo ex me quaerat,
scio,
si quaerenti explicare velim,
nescio.

Augustinus (354-430)

Was also ist die Zeit?
Wenn niemand mich danach fragt,
weiss ich’s,
will ich’s aber einem Fragenden erklären,
weiss ich’s nicht.

Time is money.

Benjamin Franklin (1706-1790)

Time is how long we wait.

Richard Feynman (1918-1988, Nobelpreis Physik 1965)

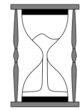
Das im menschlichen Bewusstsein verschieden erlebte Vergehen von Gegenwart zu Vergangenheit sowie von erwarteter Zukunft zu Gegenwart.

Brockhaus 1983

Der Pfeil der Zeit - Vergangenheit, Gegenwart, Zukunft

Die Zeit geht nicht, sie steht still,
Wir ziehen durch sie hin...

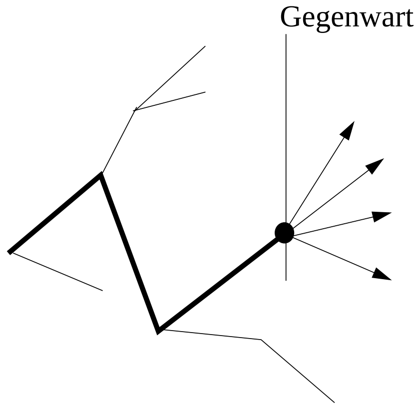
Gottfried Keller, Die Zeit geht nicht



tempus
fugit

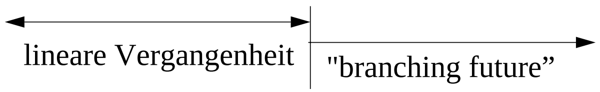
Time goes, you say?
Ah no! Alas, time stays, we go.

Austin Dobson, The Paradox of Time



Two roads diverged in a yellow wood,
And sorry I could not travel both.
And be one traveler, long I stood
And looked down one as far as I could
To where it bent in the undergrowth;
...
Then took the other, as just as fair,
...
I shall be telling this with a sigh
Somewhere ages and ages hence:
Two roads diverged in a wood, and I -
I took the one less traveled by,
And that has made all the difference.

Robert Frost (1874-1963)
The Road Not Taken (1916)



- *Rückblickend* scheint Zeit immer linear zu sein...

Zeit

...ist das häufigste (?) Substantiv der deutschen Sprache

- kommt aber in manchen Sprachen überhaupt nicht vor (z.B. Hopi)

Zeitalter	Arbeitszeit
Zeitdruck	Atomzeit
Zeitfrage	Endzeit
Zeitgeber	Frühzeit
Zeitgeist	Gleitzeit
Zeitgenosse	Gründerzeit
Zeitlupe	Halbwertszeit
Zeitnehmer	Jahreszeit
Zeitpunkt	Lebenszeit
Zeitraffer	Neuzeit
	Ortszeit
Zeitraum	Raumzeit
Zeitwende	Regierungszeit
Zeitzeichen	Spätzeit
	Steinzeit
	Tageszeit
	Teilzeit
	Uhrzeit
	Weltzeit

- Zeit kann man sparen, stehlen, rauben, nehmen, gewinnen, verlieren, absitzen, stoppen, verschwenden, vertreiben, totschlagen,...

"Die Zeit" über die "Zeit"

Stimmt es, dass "Zeit" das am häufigsten benutzte Substantiv der deutschen Sprache ist?

Jan Kny, Hamburg

Sosehr es dieser Zeitung schmeichelt: Es kommt darauf an, wer da was zählt. Tatsächlich hat ein Hobbylinguist namens Helmut Meier 1967 eine *Deutsche Sprachstatistik* veröffentlicht, in der auf Platz 90 der Liste der meistbenutzten deutschen Wörter und als erstes Substantiv das Wort Zeit steht - als nächste Hauptwörter folgen Herr, Jahre, Mann und Paragraph.

Fragt man dagegen beim Mannheimer Institut für deutsche Sprache nach, so erhält man eine ganz andere Liste, die aus einem 400 Millionen Wörter umfassenden Textkorpus ermittelt wurde: Die Top ten darin sind Jahr, Mark, Uhr, Prozent, Frau, Land, Million, Mensch, Tag und Stadt.

Paragraph und Prozent lassen stutzen und darauf schliessen, was für Texte die Sprachforscher in ihre Untersuchungen einbezogen haben mögen. Bei Herrn Meier findet man Gott und die Welt, Glaube und Gesetz ziemlich weit vorne - ein Hinweis auf juristische und theologische Quellen, wahrscheinlich hat der Mann die Bibel ausgezählt. Die Mannheimer Forscher haben vor allem Zeitungstexte ausgewertet. Was dagegen der Mann auf der Strasse sagt, protokolliert zum Glück niemand.

Christoph Drösser

Aus: "Die Zeit", 29.04.1999

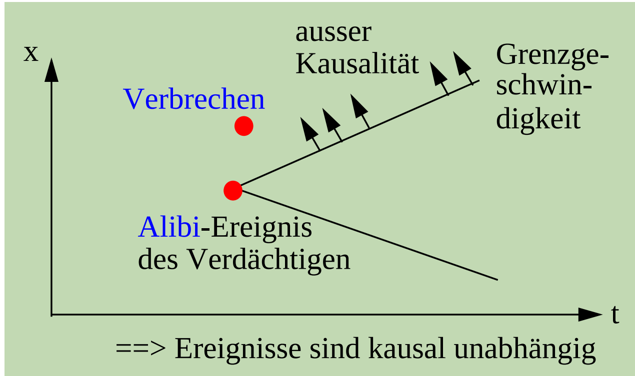
Uhren

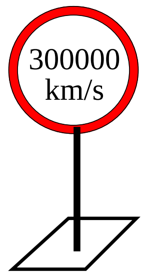
> *Ich halte ja eine Uhr für überflüssig.*
> *Sehen Sie, ich wohne ja ganz nah beim Rathaus. Und jeden Morgen, wenn ich ins Geschäft gehe, da schau ich auf die Rathausuhr hinauf, wieviel Uhr es ist, und da merke ich's mir gleich für den ganzen Tag und nütze meine Uhr nicht so ab.*
>
> Karl Valentin

Kommt Zeit, kommt Rat

1. Volkszählung: Stichzeitpunkt in der Zukunft
   - liefert eine gleichzeitige, daher kausaltreue "Beobachtung"

2. Kausalitätsbeziehung zwischen Ereignissen ("Alibi-Prinzip")
   - wurde Y später als X geboren, dann kann Y unmöglich Vater von X sein
   --> Testen verteilter Systeme: Fehlersuche/ -ursache





"speed limit of causality"
(P. Langevin)

3. Wechselseitiger Ausschluss
   - bedient wird, wer am längsten wartet

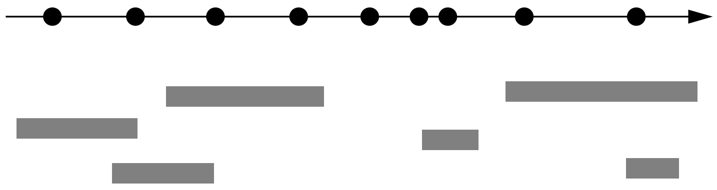
4. Viele weitere nützliche Anwendungen in unserer "verteilten realen Welt"
   - z.B. kausaltreue Beobachtung durch "Zeitstempel" der Ereignisse

> *Le temps est un grand maître, il règle bien des choses.*
> Corneille, Sertorius

Zeitmodelle

- *Punkte* “in der Zeit” zusammen mit einer Relation “später”

- Oder: Zeit- *Intervalle* zusammen mit “später”, “überlappt”...



Was ist das korrekte / geeignete Modell?

- Sind die beiden Modelle / Sichten kompatibel?
  - z.B. Start- und Endzeitpunkt bei Intervallen --> Punktmodell?

- Ist das Zeitpunktmodell besser? Oder Zeitintervallmodell?
  - vgl. passe simple / imparfait im Französischen
  - wann tritt das Ereignis "Sonne wird rot" am Abend ein?

==> *was* möchte man mit so einem Modell *wie* und *wozu* modellieren?

Eigenschaften der "Realzeit"

Formale Struktur eines Zeitpunktmodells:

- transitiv
- irreflexiv --> lin. Ordnung
- linear
- unbeschränkt ("Zeit ist ewig": kein Anfang oder Ende)
- dicht (es gibt immer einen Zeitpunkt dazwischen)
- kontinuierlich
- metrisch
- homogen
- vergeht "von selbst" --> jeder Zeitpunkt wird schliesslich erreicht

Typische Modelle: reelle Zahlen, rationale Zahlen (?)

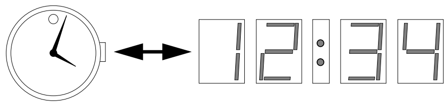
Welche Eigenschaften benötigen wir wirklich?

- was wollen wir mit "Zeit" anfangen?

--> "billigeren" Ersatz für fehlende globale Realzeit!

Zeit und Uhren in der Informatik

- Hardwarezähler als Uhren
  --> Zeit ist *diskret* (nicht dicht)



- Uhrenüberlauf z.B. bei langen Simulationsläufen
  --> Zeit ist nicht ewig sondern *beschränkt*

- Bei ereignisorientierter Sicht:

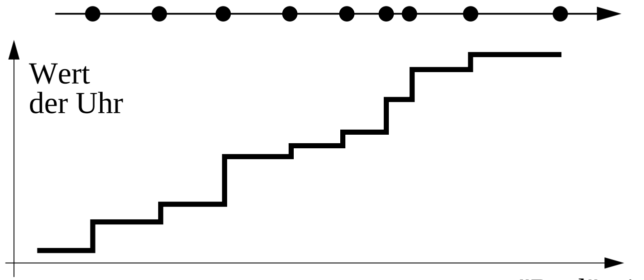
  - Zwischen zwei Ereignissen geschieht nichts

    --> Uhren brauchen nicht kontinuierlich zu laufen

    --> Uhren nur bei Ereignissen verändern

- "Weltsicht": Zeit *ist* das Stattfinden von Ereignissen

- Beispiel für diese Weltsicht: Ereignisgesteuerte Simulation



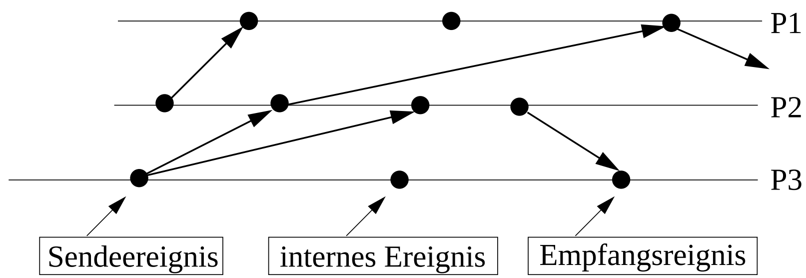
Nur bei den *Ereignissen* wird der Zustand der Welt betrachtet und gegebenenfalls verändert!

Wir nennen also Konzepte und Dinge "Zeit" und "Uhren", obwohl diese nicht alle idealen Eigenschaften besitzen!

was aber sind die *wesentlichen* Eigenschaften?

Kausalrelation ("Happened Before", Lamport 1978)

- *Zur Wiederholung:*



- interessant: von links nach rechts verlaufende "Kausalitätspfade" (bestehend aus Nachrichtenpfeilen + Teilstücken auf Prozessachsen)

- *Kausalrelation* ‘<‘ auf der Menge E aller Ereignisse:

> “Kleinste” transitive Relation auf E, mit **x** < **y** wenn:
>
> 1) **x** und **y** auf dem gleichen Prozess stattfinden und **x** vor **y** kommt, *oder*
>
> 2) **x** ist ein <u>Sendeereignis</u> und **y** ist das korrespondierende <u>Empfangsereignis</u>

- In einem Zeitdiagramm gilt für je zwei Ereignisse e, e’ die Relation e<e’ genau dann, wenn es einen Kausalitätspfad von e nach e’ gibt

Logische Zeitstempel von Ereignissen

- Verteilte Berechnung abstrakt: n Prozesse, halbgeordnete Ereignismenge E, Nachrichten (Sende- / Empfangsereignis)
- Zweck: Ereignissen eine Zeit geben ("dazwischen" egal)
- Gesucht: Abbildung C: E --> H

Clock

"Zeitbereich":
Halbgeordnete Menge
--> "füher", "später"

- Für $e \in E$ heisst C(e) *Zeitstempel* von e
- C(e) bzw. e *früher* als C(e') bzw. e', wenn C(e) < C(e')
- Wie soll H aussehen?

z.B.:

- **N** (lineare Ordnung)
- **R** (bzw. REAL-Datentyp)
- Potenzmenge von E
- $\mathbf{N}^n$ (d.h. n-dim. Vektoren)

- Sinnvolle Forderung:

Kausalrelation ("Pfad im Diagramm")

Uhrenbedingung: e < e' ==> C(e) < C(e')

Ordnungshomomorphismus

Zeitrelation "früher"

Interpretation ("Zeit ist kausaltreu"):

Wenn ein Ereignis e ein anderes Ereignis e' beeinflussen kann, dann muss e einen kleineren Zeitstempel als e' haben

Logische Uhren von Lamport

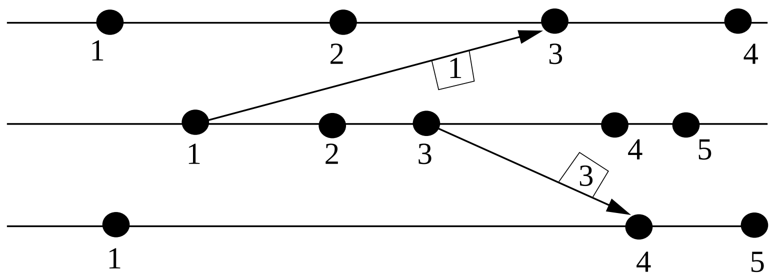
Commun. ACM 1978:
Time, Clocks, and the Ordering of Events in a Distributed System

C: (E,<) --> (**R**,<)   Zuordnung von Zeitstempeln

Kausal-
relation

(oder **N**)

e < e' ==> C(e) < C(e')   Uhrenbedingung



Protokoll zur Implementierung der Uhrenbedingung:

- Lokale Uhr (= Zähler) *tickt* "bei" *jedem* Ereignis
- Sendeereignis: Uhrwert mitsenden (*Zeitstempel*)
- Empfangsereignis: *max*(lokale Uhr, Zeitstempel)

zuerst! danach "ticken"

Behauptung:
Protokoll respektiert Uhrenbedingung

Beweis: Kausalitätspfade sind monoton...

Eigenschaften der Lamport-Zeit

- Was bleibt von den Eigenschaften der Realzeit?

    + lin. Ordnung, unbeschränkt
    + respektiert Kausalität
    - diskret
    - vergeht nicht von alleine

"Critical path" --> Parallelitätsmass

- Zeitstempel = Länge der längsten vorangehenden Kette

- Uhrenbedingung ==>

    - lokal aufsteigende Zeitstempel
    - Sendeereignis kleineren Zeitstempel als Empfang
    - $C(a) < C(b) \implies \neg (b < a)$

Zukunft kann Vergangenheit nicht beeinflussen!

Bew.: $b < a \implies C(b) < C(a)$

$\implies \neg(C(a) < C(b))$

Kausale Unabhängigkeit

- *Def*. ["kausal unabhängig"]: $a \| b \Leftrightarrow: \neg(a < b) \wedge \neg(a > b)$

Beachte: $\|$ ist *nicht transitiv*! (Gilt eigentlich $a \| a$?)

- *Behauptung*: $C(a) = C(b) \implies a \| b$

Bew.:

$C(a) = C(b) \implies \neg(C(a) < C(b)) \wedge \neg(C(a) > C(b))$

$\implies \neg(a < b) \wedge \neg(a > b)$

$\implies a \| b$ (Def.)

Verwende $\neg(C(x) < C(y)) \implies \neg(x < y)$ aus der Uhrenbedingung!

- Gilt die Umkehrung der Uhrenbedingung?

Nein, $C(e) < C(e') \implies e < e'$ gilt nicht!

Es gilt nur: $C(e) < C(e') \implies e < e'$ *oder* $e \| e'$

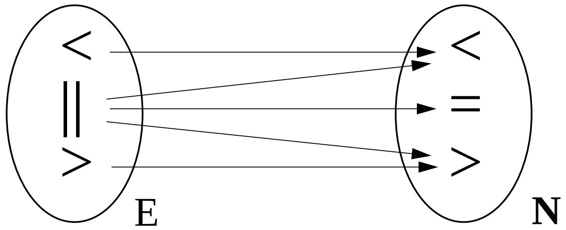
vgl. Beispiel

Das heisst:

Aus den Zeitstempeln lässt sich nicht (immer) schliessen, ob zwei Ereignisse kausal voneinander abhängig sind, oder ob sie unabhängig sind!

==> Aber wozu sind solche Zeitstempel dann gut?

Lamport-Zeit: Strukturverlust



- Negation geht verloren
- Ordnungshomomorphismus, jedoch nicht Isomorphism
- E ist Halbordnung, N ist lineare Ordnung

--> zwei kausal unabhängige Ereignisse werden vermöge der Abbildung plötzlich (scheinbar) vergleichbar!

Strukuturverlust ist ein wichtiger *Defekt*: Zweck der Zeitstempel ist, auf die Beziehung der Ereignisse zurück- zuschliessen! Geht das besser? Anderer "Zeitbereich"?

Lamport-Zeit: Nicht-Injektivität

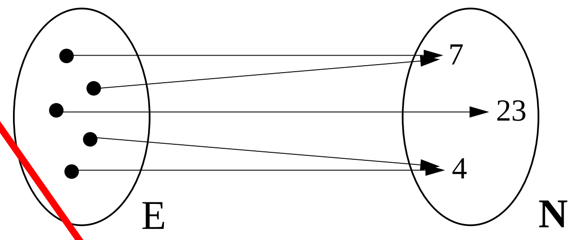


Abbildung ist nicht injektiv

- Wichtig z.B. für: "Wer die kleinste Zeit hat, gewinnt"

- Lösung:

Lexikographische Ordnung (C(e),i), wobei i die Prozessnummer bezeichnet, auf dem e stattfindet

Ist injektiv, da alle lokalen Ereignisse verschiedene Zeitstempel C(e) haben ("break ties")

- *lin*. Ordnung $(a,b) < (a',b') \Leftrightarrow a<a' \vee a=a' \wedge b<b'$

--> alle Ereignisse haben *verschiedene* Zeitstempel

--> Kausalitätserhaltende Abb. (E,<) **-->** (**N**$\times$**N, <)**

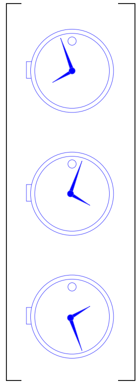
Zu jeder Menge von Ereignissen gibt es nun ein eindeutig "frühestes"!

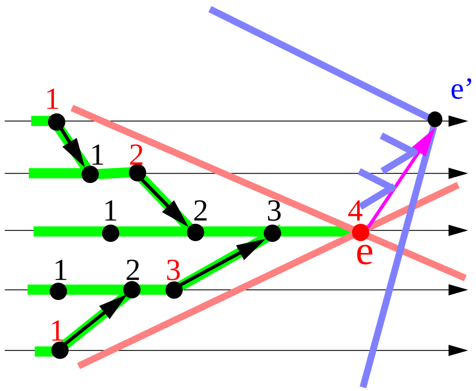
Vektorzeit

Andere Zeiten, andere Sitten

Quot tempora tot astra
G. Bruno (1548-1600)

"relativistische" Weltsicht, vgl. auch
- Nicolaus Kopernicus
- Galileo Galilei





Vektorzeit: Motivation

Umkehrung der Uhrenbedingung gilt nicht für Lamport-Zeit
- C(e) < C(e') ==> e < e' gilt nicht!
- es gilt nur: C(e) < C(e') ==> e < e' oder e||e'

Zeit := vergangene Zeit
:= Vergangenheit
:= Menge vergangener Ereignisse

viele Uhren messen die Zeit, indem sie *vergangene* Sekunden zählen

vgl. dies mit der *Lamport-Zeit* ("lokal vergangen")

Kausalrelation

Zeit(e) := {e'| e' ≤ e} = *Kegel* von e

Genauer: Zeitstempel eines Ereignisses

Kann durch lokal späteste Ereignisse repräsentiert werden (linksabgeschlossen)

Hiervon gibt es n Stück (n = Anzahl der Prozesse)

!

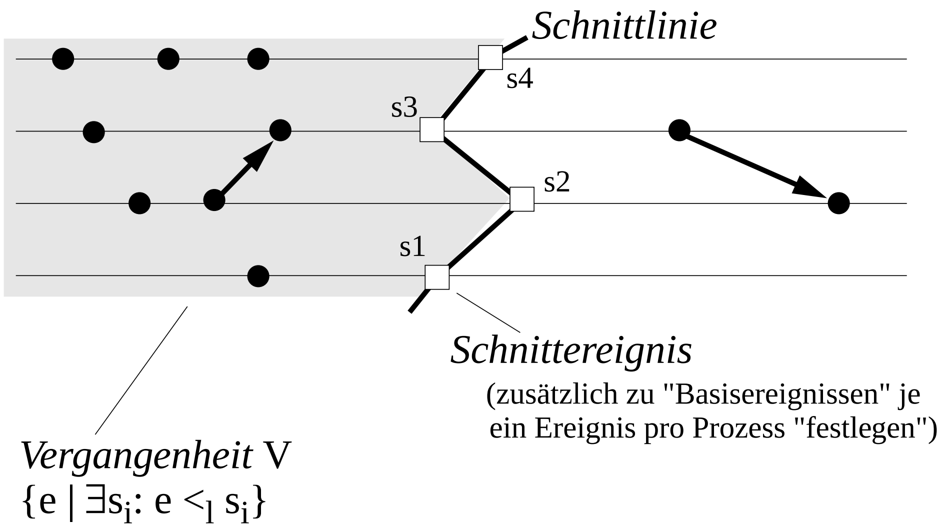
--> *Zeitstempel* ist n-dimensionaler Vektor

--> *Zeit* ist Menge n-dimensionaler Vektoren

--> *Uhr* ist ein array C[1:n]

zum Anzeigen von Zeitvektoren

Schnitt und Schnittlinie



- Schnittlinie trennt Zeitdiagramm / Ereignismenge in zwei disjunkte Mengen "Vergangenheit" / "Zukunft"

- Bemerkung: $e \in V \wedge e' <_l e \Longrightarrow e' \in V$
  (linksabgeschlossen bzgl. lokaler Kausalrelation)

Denkübung: Man vergleiche den Begriff des *Schnittes* (insbes. des *konsistenten Schnittes*, vgl. nachfolgende Folie) mit dem früher erwähnten Begriff der *Präfixberechnung*! Man beachte auch die Halbordnungsstruktur bzw. Verbandsstruktur dieser Begriffe.

Konsistente Schnitte

Def. **Schnitt**:

$S \subseteq E$ heisst *Schnitt* von E, falls $e \in S \wedge e' <_l e \Longrightarrow e' \in S$
(d.h. Schnitt wird mit seiner Vergangenheit identifiziert)

Def. **konsistenter Schnitt**:

$S \subseteq E$ heisst *konsistent*, falls $e \in S \wedge e' < e \Longrightarrow e' \in S$

Def.: Schnitt S *später* als S' :<==> $S' \subseteq S$

bzw. $\subset$ bei "strikt später"

Schreibweise auch: $S' < S$

Beh.: Jeder konsistente Schnitt ist ein Schnitt

Bew.: $<_l \subseteq <$

Bem.: Schnitt(linie) inkonsistent <==>
$\exists$ "Nachricht aus der Zukunft"

Bem.: Schnitt(linie) konsistent <==>
Schnittereignisse paarweise kausal unabhängig

Bew. als Übung

Bem.: Schnitt(linie) konsistent <==>
lässt sich senkrecht darstellen (Gummibandtransf.)

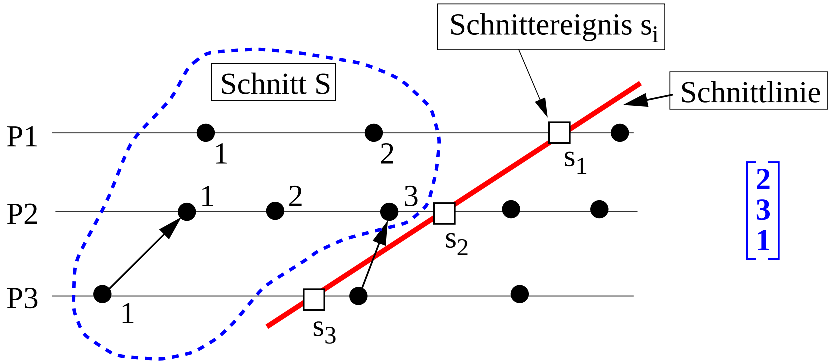
Bew. bereits bekannt: Diagramm auseinanderschneiden und versetzen

Vektorzeitstempel von Schnitten

Zeitstempel $\tau(S)$ eines Schnittes S ist ein Vektor aus $\mathbf{N}^n$

alle Ereignisse, die links von einer Schnittlinie liegen

Anzahl der Prozesse



lokal!

Def. $\tau(S)[i] := |\{e \in E_i \mid e <_l s_i\}| := |S \cap E_i|$

(für jede Komponente i mit $1 \leq i \leq n$)

Interpretation:

$\tau(S)[i]$ ist die "Stelle", wo die Prozessachse von P_i durch die Schnittlinie geschnitten wird

Beachte: man kann zu *konsistenten* und *inkonsistenten* Schnitten den zugehörigen Zeitvektor definieren!

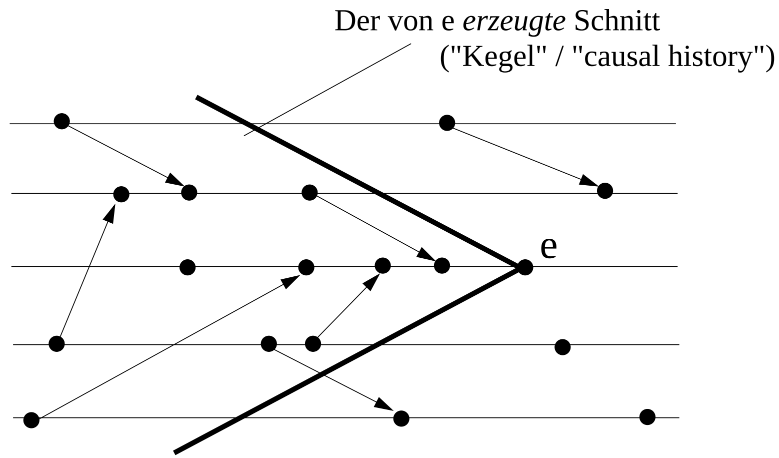
Kausale Vergangenheit eines Ereignisses

Def. *kausale Vergangenheit* $\downarrow(e)$ eines Ereignisses e:

$$\downarrow(e) = \{e' \mid e' \leq e\}$$

Beh.: $\downarrow(e)$ ist ein konsistenter Schnitt

Bew. als Übung



Beh.: $e' \leq e \iff e' \in \downarrow(e)$

Beh.: $e \parallel e' \iff \neg(e \in \downarrow(e')) \wedge \neg(e' \in \downarrow(e))$

(Bew. klar)

Vektorzeitstempel von Ereignissen

Kausale Vergangenheit

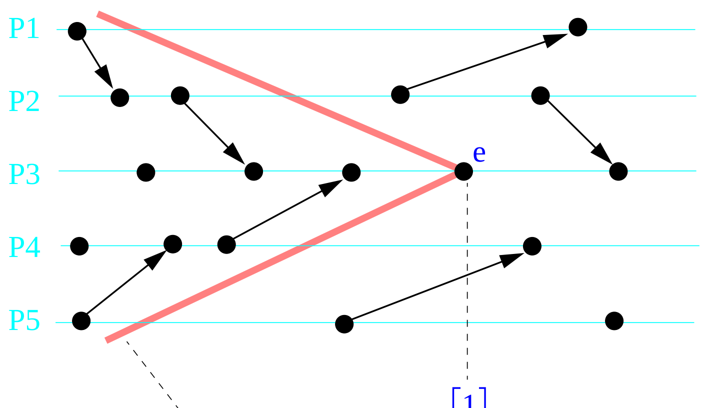
- Bezeichne $\downarrow$e den *Kegel* $\{e' \in E \mid e' \leq e\}$ von e
  - jeder Kegel ist ein konsistenter Schnitt (da linksabgeschlossen) ("Kegelmantel" könnte also als senkrechte Linie gezeichnet werden!)
  - Repräsentation durch die n lokal am weitesten rechts liegenden Ereignisse

Menge der Ereignisse von P_i

- Dann definiere $\tau(e) := \tau(\downarrow e)$

  Also: $\tau(e)[i] := |\{e' \in E \mid e' \leq e\} \cap E_i| := |\{e' \in E_i \mid e' \leq e\}|$

  - das heisst: Zeitstempel eines Ereignisses = Zeitstempel seines Kegels

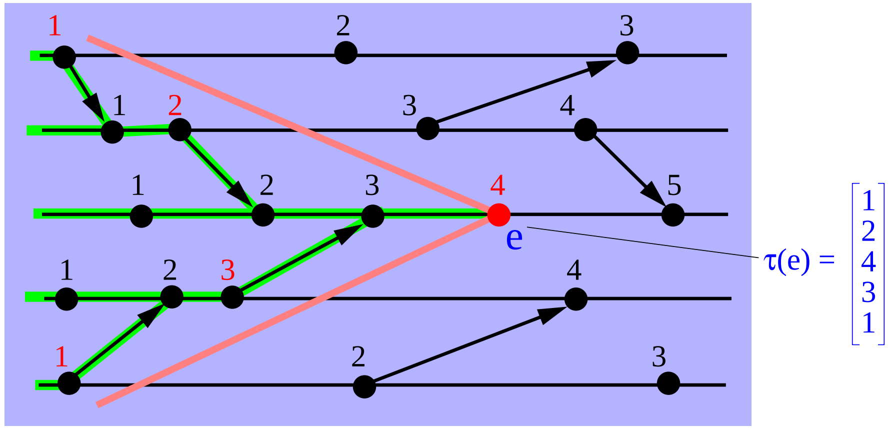


Schnittlinie in der Form eines Kegels (vgl. "Lichtkegel" der relativistischen Physik): $\downarrow$e ist ein konsistenter Schnitt

$\begin{bmatrix}1\\2\\4\\3\\1\end{bmatrix}$ τ(e): Vektorzeitstempel von e

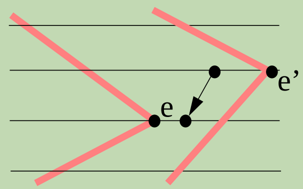
$\tau(e) < \tau(e')$

- Jeder Prozess numeriert seine Ereignisse lokal durch

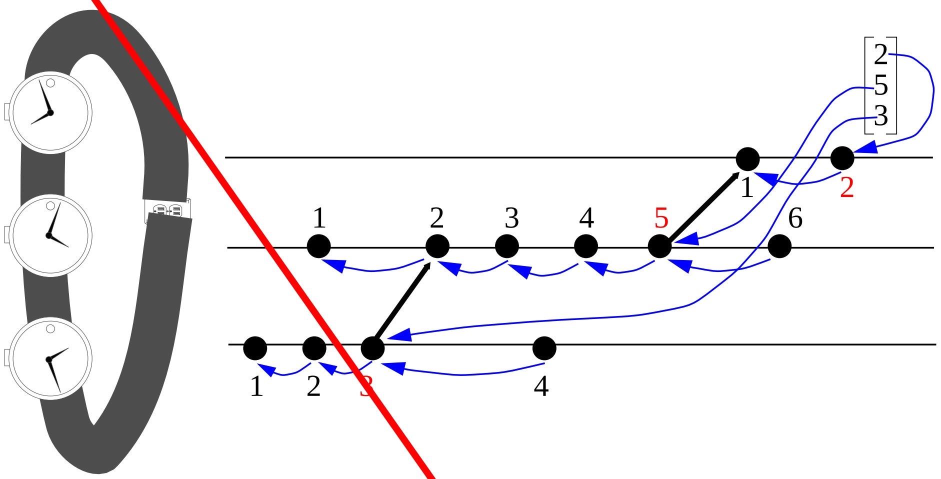


- Vektor τ(e) repräsentiert gesamte kausale Vergangenheit des Ereignisses e
  - Veranschaulichung durch einen "Kegel" $\{e' \mid e' \leq e\}$
  - Nummer des jeweils lokal letzten kausal vorangehenden Ereignisses steht in der jeweiligen Komponente des Vektors

- Interpretation von $\tau(e) < \tau(e')$:
  - e liegt in der kausalen Vergangenheit von e'
  - Kegel von e ist ganz im Kegel von e' enthalten



Vektorzeitstempel: Interpretation



- Zeigt auf jeweils jüngstes kausal vergangenes lok. Ereignis
- Damit implizit auch auf alle vorangehenden
  (wegen lokaler totaler Ordnung)
- Vektor repräsentiert *gesamte kausale Vergangenheit*
- Kodiert "*Wissen*" über (jedes einzelne) vergangene Ereignis
  Genauer: Vektorzeit repräsentiert die Kausalrelation in isomorpher Weise!

Denkübungen:

- wie stellt man fest, ob e' im Kegel von e mit τ(e) liegt?
- gibt es eine noch kompaktere Kodierung?

Def. "Zeitstempelarithmetik"

$$\begin{bmatrix}1\\3\\4\\3\\2\end{bmatrix} \leq \begin{bmatrix}1\\7\\4\\6\\2\end{bmatrix} \qquad\qquad \begin{bmatrix}1\\3\\4\\3\\7\end{bmatrix} \parallel \begin{bmatrix}5\\3\\8\\3\\2\end{bmatrix}$$

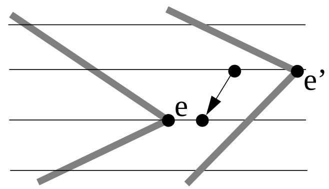
vergleichbar
(komponentenweise ≤)

"konkurrent"

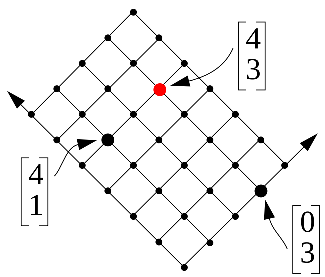
‘<‘ definiert als “≤ aber ≠”

Interpretation von $\tau(e) < \tau(e')$:

- e liegt in der kausalen Vergangenheit von e’
- Kegel von e ist im Kegel von e’ enthalten



$$\sup\left(\begin{bmatrix}1\\4\\2\\3\\7\end{bmatrix},\begin{bmatrix}8\\3\\4\\3\\2\end{bmatrix}\right)=\begin{bmatrix}8\\4\\4\\3\\7\end{bmatrix}$$

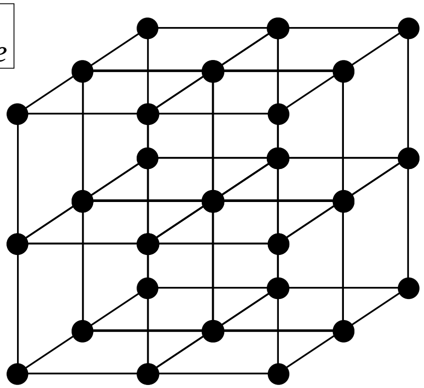


sup = komponentweises Maximum

Der Zeitverband

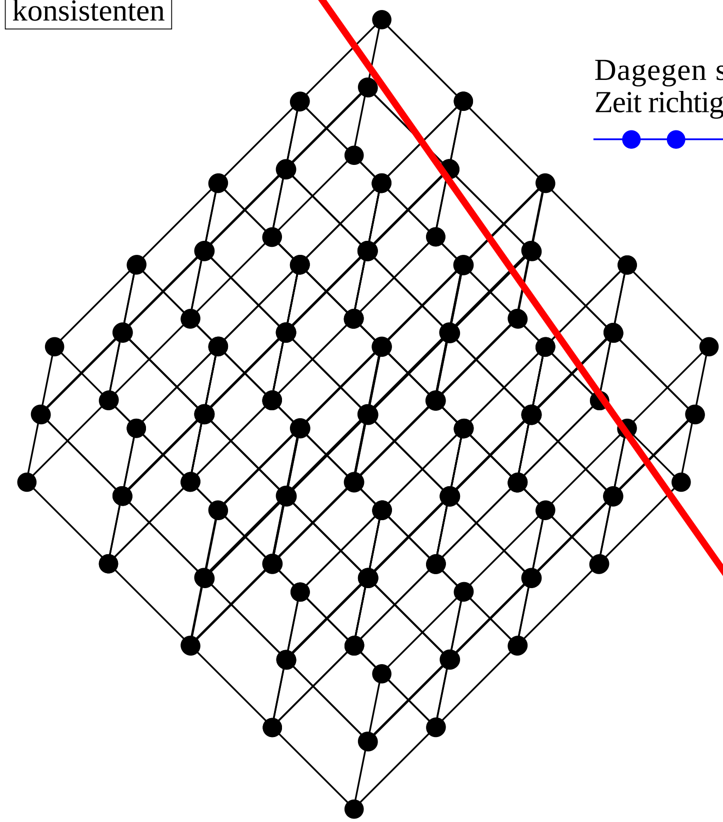
Zeitvektoren bilden bzgl. sup bzw. ≤ einen *Verband*, der sich in kanonischer Weise als *n-dimensionales Gitter* darstellen lässt

engl.: *lattice*



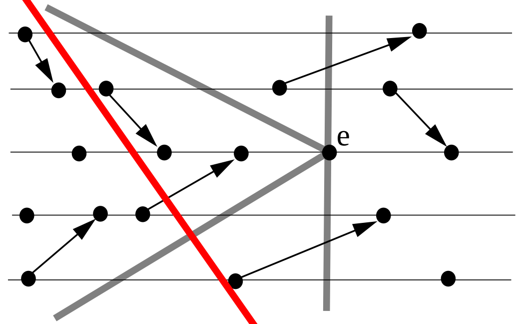
Er entspricht dem *Verband aller Schnitte* der Berechnung

nicht nur der konsistenten



Dagegen sieht lineare Zeit richtig langweilig aus!

Vektorzeit und ideale Beobachter



$$\begin{bmatrix}0\\0\\0\\0\\0\end{bmatrix} \quad \dots \quad \begin{bmatrix}2\\4\\5\\4\\3\end{bmatrix}$$

Wahrnehmungen des idealen Beobachters

$$\tau(e) = \begin{bmatrix}1\\2\\4\\3\\1\end{bmatrix} \qquad id(e) = \begin{bmatrix}1\\3\\4\\3\\2\end{bmatrix}$$

- Numeriere Ereignisse lokal
- Idealisierter Beobachter nimmt "Ticken" sofort wahr
- Geeignete Datenstruktur hierfür: Vektor / array
- Für *jeden* Beobachter gilt stets: $\tau(e) \leq id(e)$ $(\forall e)$
  (komponentenweise '≤' ; idealer Beobachter sieht stets die gesamte kausale Vergangenheit und evtl. einige weitere Ereignisse)
- $\tau(e)$ = Infimum aller idealen Sichten id(e)
- Beachte: id(e) hängt vom Zeitdiagramm ab!
- Aber $\tau(e)$ ist invariant bzgl. Gummibandtransformation!

Daher nur zur Motivation

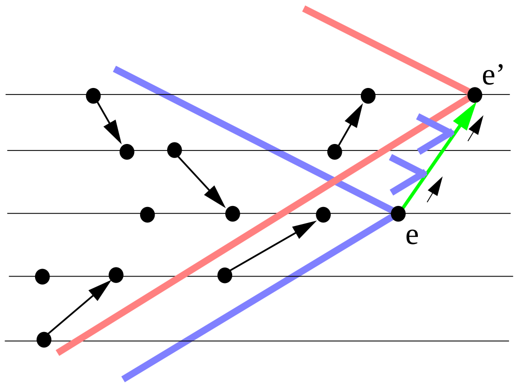
Implementierung der Vektorzeit

- Idee: Analog zur Lamport-Zeit
  (hier allerdings stets vektoriell!)

- Nachrichten enthalten die gesamte kausale Vergangenheit des Senders ==> Zeitvektor des Sendeereignisses

- Bei Empfang einer Nachricht:
  - Vereinigung der Kegel
  - ==> Supremum der Vektoren

Wissen über vergangene Ereignisse vereinigen



Mitschleppen des Kegels des Sende-ereignisses und Vereinigung mit dem Kegel des Empfangsereignisses

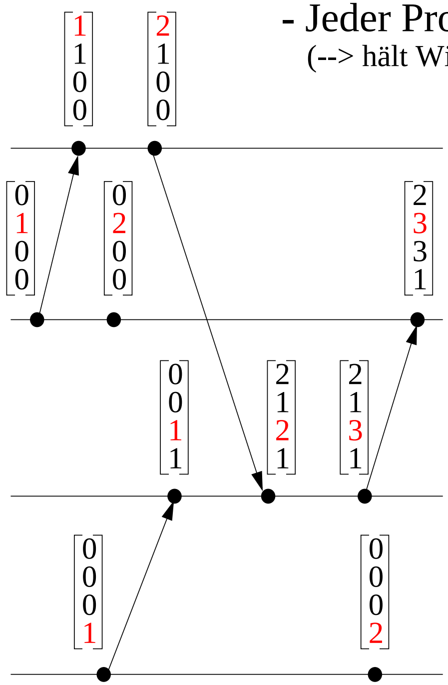
--> "induktiv": ein Ereignis hat ein "vollständiges Wissen" über alle seine vergangenen Ereignisse

Propagieren des Zeitwissens

(--> Implementation der Vektorzeit)

Andere Zeiten, andere Sitten

- Jeder Prozess besitzt eine *Vektoruhr*
  (--> hält Wissen über vergangene Ereignisse)



- *bei jedem Ereignis:*
  eigene Komponente erhöhen
- *beim Senden:*
  neuen Vektor mitsenden
- *beim Empfangen:*
  komponentenweises Maximum der beiden Vektoren

Vereinigung der beiden Kegel

Kausalrelation

bzgl. Zeitvektor

- *Behauptung*: $e < e' \Leftrightarrow \tau(e) < \tau(e')$

- *Anschauliche Interpretation*:
  - $\tau(e) \leq \tau(e') \Leftrightarrow$ es gibt eine Kausalkette von e zu e’

monoton bzgl. Zeitvektoren!

- *Korollar*: $e \parallel e' \Leftrightarrow \tau(e) \parallel \tau(e')$

Interpretation: Genau die "gleichzeitigen" Ereignisse beeinflussen sich nicht ggs.

Kausal- und Zeitstruktur: Isomorphie

"Hauptsatz": $e < e' \Leftrightarrow \tau(e) < \tau(e')$

- Umkehrung der Uhrenbedingung
- gilt nicht für Lamport-Zeit

Beweis:

(1) $e \leq e' \Leftrightarrow e \in \downarrow e'$
wegen Def. von Kegel

(2) $e \in \downarrow e' \Leftrightarrow \downarrow e \subseteq \downarrow e'$
klar nach Def. von Kegel (= kons. Schnitt)

(3) $\downarrow e \subseteq \downarrow e' \Leftrightarrow \tau(\downarrow e) \leq \tau(\downarrow e')$
weil "später" sich jeweils überträgt

(4) $\tau(\downarrow e) \leq \tau(\downarrow e') \Leftrightarrow \tau(e) \leq \tau(e')$
nach Def. von $\tau(e)$

Irreflexivität folgt aus Injektivität von τ

Anschauliche Interpretation: Wissen über lokale Zeit über Kausalkette propagiert

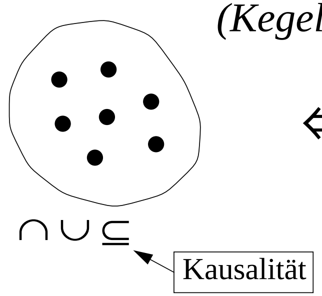
---

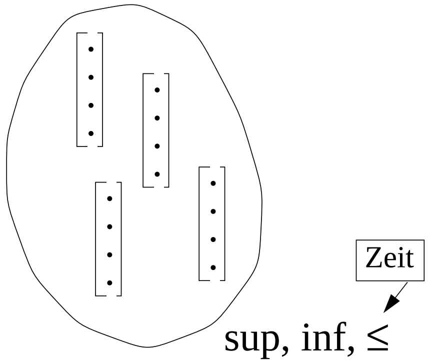
Verschärfung: Falls $e \in E_i$ (und $e \neq e'$):
$e < e' \Leftrightarrow \tau(e)[i] < \tau(e')[i]$

Effizienter: Nur eine Komponente testen

Rechnen mit Ereignismengen

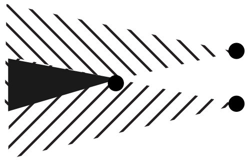
Mengen von Ereignissen (Kegel) $\Leftrightarrow$ *Zeitvektoren*

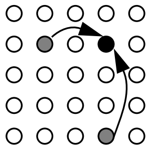




Mengentheoretische Operationen $\Leftrightarrow$ Algebraische Operationen (--> “rechnen”)

Verbandsstruktur auf 2^E (Ideale) $\Leftrightarrow$ Produduktverband auf $\mathbf{N}^n$





Ordnungstheoretische Eigenschaften $\Leftrightarrow$ Algebraische Eigenschaften

---

Vektorzeit und Vektoruhren ermöglichen eine "operationale Manipulation" der Kausalrelation

Eigenschaften der Vektorzeit

$\tau'(e) := \sum_i \tau(e)[i]$ hat Eigenschaften der Lamport-Zeit:

1) $e < e' \implies \tau'(e) < \tau'(e')$ (Uhrenbedingung)
2) Umkehrung gilt nicht
3) τ' ist nicht injektiv
4) $\tau'(e)$ ist ein Skalar

Beachte: $\tau'(e)$ ist das "Volumen" des Kegels, die Lamportzeit die längste vorangehende Kette von e.

Realzeitpunkte

Beh.:
$\tau(e) < \tau(e') \implies t(e) < t(e')$

Bew.: $\tau(e) < \tau(e') \implies e < e' \implies t(e) < t(e')$

Bem.: Gilt nicht für die Lamport-Zeit!

Frage: Gibt es kompaktere Zeitstempel als Vektoren der Länge n? (Für die auch die Umkehrung der Uhrenbedingung gelten.)

nicht leicht!

Wäre für die Praxis sehr wichtig (Zeitstempel in Nachrichten können unangenehm lang werden...)

Woraus stammt dieses Zitat?

Und überall hingen, lagen und standen Uhren.
Da gab es auch *Weltzeituhren in Kugelform, welche die Zeit für jeden Punkt der Erde anzeigten...*
...
...
M. schüttelte lächelnd den Kopf.
"Die Uhr allein würde niemand nützen.
Man muss sie auch lesen können."

Anwendungen der Vektorzeit

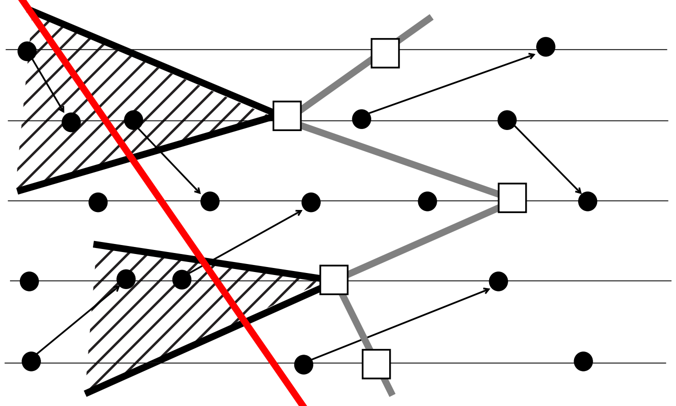
<Momo trifft Professor Hora>:

Und überall hingen, lagen und standen Uhren.
Da gab es auch *Weltzeituhren in Kugelform,*
welche die Zeit für jeden Punkt der Erde anzeigten...
"Vielleicht", meinte Momo,
braucht man dazu eben so eine Uhr."
Meister Hora schüttelte lächelnd den Kopf.
"Die Uhr allein würde niemand nützen.
Man muss sie auch lesen können."

Michael Ende, Momo

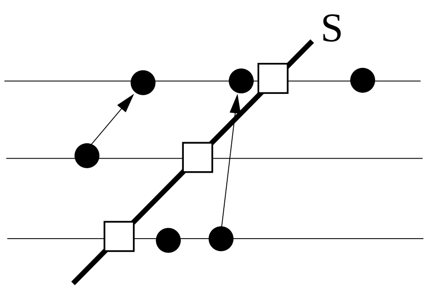
- Debugging $\boxed{!}$
  - Lokalisierung von Fehlern (“kann [nicht] Ursache sein...)
  - Race conditions; Synchronisationsfehler (kausale Unabhängigkeit)
  - Effizientes Replay
- Leistungsanalyse
  - “Flaschenhals” im Zeitverband; Synchronisationsgrad
  - Kausal unabhängige Ereignisse können parallel ausgeführt werden
- Implementierung konsistenter Schnappschüsse
  - Menge lokaler Schnappschüsse mit paarweise konkurrenten Ereignissen
- Realisierung von kausaltreuen Beobachtern
  - Causal broadcast
  - Causal order

Zeitvektoren der Schnittereignisse



- Betrachte nun die Zeitvektoren der Schnittereignisse s_i eines Schnittes S.
- Liegen die Kegel $\downarrow s_i$ ganz im Schnitt? (Gilt $\downarrow s_i \subseteq S$?)

Nicht immer:



Konsistente Hülle eines Schnittes

Für einen Schnitt S definiere $S^* := \downarrow s_1 \cup \downarrow s_2 \cup_{...} \cup \downarrow s_n$
(Vereinigung aller Schnittereigniskegel)

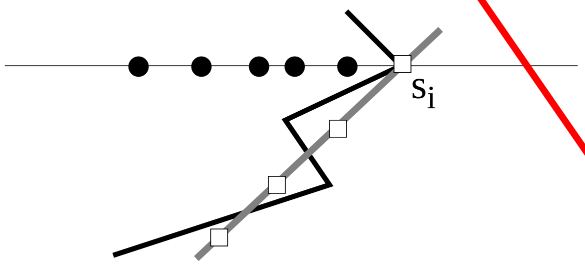
Beh.: Für jeden Schnitt S ist S*
- a) eindeutig ← klar nach Konstruktion
- b) ein Schnitt
- c) konsistent

← (b, c):
- jeder Kegel ist ein kons. Schnitt (Übung)
- kons. Schnitte sind bzgl. Vereinigung abgeschlossen (Verband --> Übung)

Beh.: Es gilt $S \subseteq S^*$ (d.h. S* ist später/gleich S)

Bew.: $S \cap E_i \subseteq \downarrow s_i$

(Alle Ereignisse des Schnittes auf dem i-ten Prozess werden durch $\downarrow s_i$ abgedeckt)



Fragen:

a) Gibt es u.U. einen Schnitt S' mit $S \subset S' \subset S^*$?

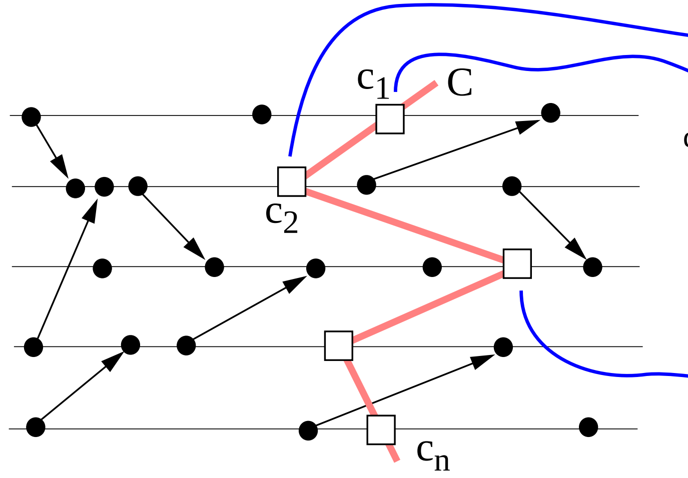
b) ... konsistenten Schnitt...?

Schnittmatrix

- Schnittmatrix \$ eines Schnittes C (mit Schnittereign. c_i):

$\$(C) := (\tau(c_1), \tau(c_2), ..., \tau(c_n))$

d.h. Schnittereignisvektoren c_i als Spaltenvektoren



$$
\begin{array}{ccccc|c}
\multicolumn{5}{l}{\text{dia}} & \text{sup} \\
3 & 1 & 1 & 0 & 0 & 3 \\
0 & 4 & 3 & 0 & 0 & 4 \\
0 & 0 & 5 & 0 & 0 & 5 \\
0 & 1 & 3 & 4 & 0 & 4 \\
0 & 0 & 1 & 1 & 3 & 3
\end{array}
$$

Frage: Kann man an den Schnittmatrizen etwas über die Schnitte erkennen? (z.B.: ob später, früher; ob konsistent...)

$\boxed{\text{C konsistent} \Leftrightarrow \text{dia}(\$) = \text{sup}(\$)}$

Diagonalvektor   Zeilenmaximum

(d.h. Maximum einer Zeile ist das Diagonalelement)

Diagonalvektor und Zeilensupremum

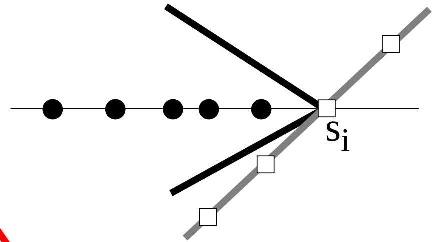
- Def.: dia(\$) ist der Vektor mit dia(\$)[i] = s_i[i]

- Beh.: Diagonalvektor von \$ ist der Schnittvektor von S,
  d.h. dia(\$) = τ(S)

- Bew.:

$$\begin{aligned} \mathrm{dia}(\$)[i] &= \tau(s_i)[i] \\ &= |\{e' \in E_i \mid e' \leq s_i\}| \\ &= |\{e' \in E_i \mid e' \leq_l s_i\}| \\ &= \tau(S)[i] \end{aligned}$$

wieso?



- Def.: sup(\$) ist der Vektor sup($\tau(s_1),\ldots,\tau(s_n)$)
  (zeilenweise Maximum)

Beispiel:

$$\$ = \begin{matrix} 3 & 1 & 1 & 5 & 0 \\ 0 & 4 & 3 & 0 & 0 \\ 0 & 0 & 5 & 0 & 0 \\ 0 & 1 & 3 & 4 & 0 \\ 0 & 0 & 1 & 1 & 3 \end{matrix} \qquad \mathrm{dia}(\$) = \begin{bmatrix} 3 \\ 4 \\ 5 \\ 4 \\ 3 \end{bmatrix} \qquad \sup(\$) = \begin{bmatrix} 5 \\ 4 \\ 5 \\ 4 \\ 3 \end{bmatrix}$$

Verträglichkeit von τ mit $\cup$ und $\cap$

Beh.: Für zwei Schnitte S, S' einer Berechnung gilt:

$$\boxed{\tau(S \cup S') = \sup(\tau(S), \tau(S'))}$$

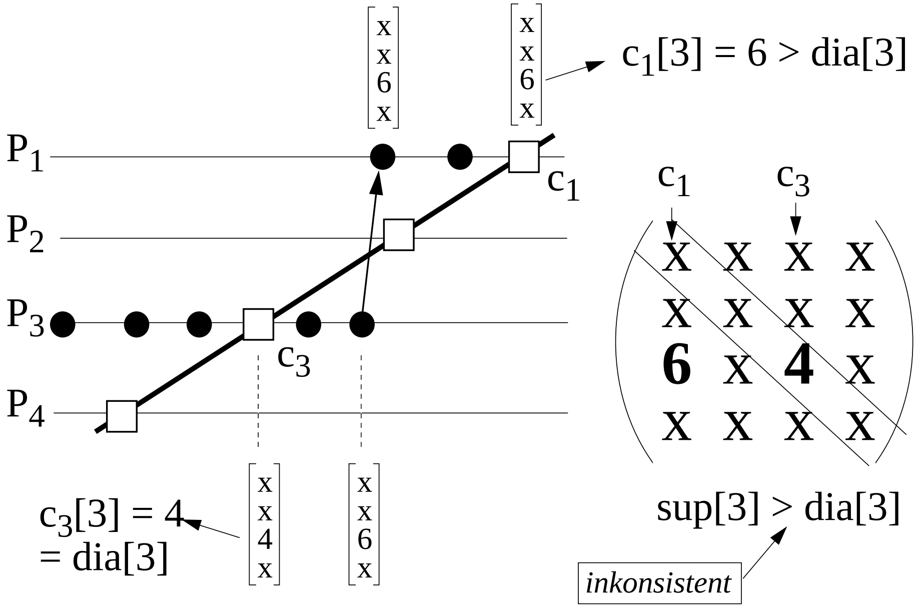
Bew.:

$$\begin{aligned} \tau(S \cup S')[i] &= |\{e \in E_i \mid e \leq_l s_i \vee e \leq_l s_i'\}| \\ &= \max(|\{e \in E_i \mid e \leq_l s_i\}|, |\{e \in E_i \mid e \leq_l s_i'\}|) \\ &= \max(\tau(S)[i], \tau(S')[i]) \end{aligned}$$

Bem.: $\tau(S \cap S') = \inf(\tau(S), \tau(S'))$ analog

Bem.: $\cup$, $\cap$, inf, sup sind assoziativ
--> $\tau(S_1 \cup S_2 \cup \ldots \cup S_k)$ ist sinnvoll.

Das “sup = dia”-Konsistenzkriterium



Ein Prozess (P_1) verschieden von P_3 weiss (bei c_1) etwas über lokale Ereignisse auf P_3, von denen P_3 selbst noch nichts weiss (d.h. die *nach* c_3 geschehen)

<==>

Es gibt einen Pfad von einem Ereignis auf P_3 *nach* c_3 zu einem Ereignis *vor* c_1

<==>

[Generalisierung über alle Indizes i≠j]

Der Schnitt ist inkonsistent

Konsistenzkriterium

Beweis hier ohne "anschauliche" Zeitdiagramme

- Beh.: S inkonsistent ==> $S \neq S^*$
- Bew.: S^* ist stets konsistent (vgl. oben)

- Korollar: S inkonsistent ==> $\tau(S) \neq \tau(S^*)$
  (Versch. Schnitte haben versch. Zeitstempel)

- Beh.: S inkonsistent ==> dia(\$) ≠ sup(\$)
- Bew.: dia(\$) = $\tau(S)$ und

  Verträglichkeit

  $sup(\$) \overset{Def.}{=} sup(\tau(s_1),\ldots,\tau(s_n)) = \tau((\downarrow s_1) \cup \ldots (\downarrow s_n)) \overset{Def.}{=} \tau(S^*)$.

  Wende nun obiges Korollar an.

- Beh.: S konsistent ==> dia(\$) = sup(\$)
- Bew.: $\downarrow s_i$ liegt ganz in S, d.h. $\downarrow s_i \subseteq S$

  Denn: 1) $x \in \downarrow s_i$ ==> $x \leq s_i$
  2) $y \in S \wedge x \leq y$ ==> $x \in S$
  Wegen $s_i \in S$: $x \in \downarrow s_i$ ==> $x \in S$
  Also $\downarrow s_i \subseteq S$

  ==> $S^* \subseteq S$. Umkehrung gilt sowieso:

  ==> $S^* = S$ vgl. vorh. Lemma

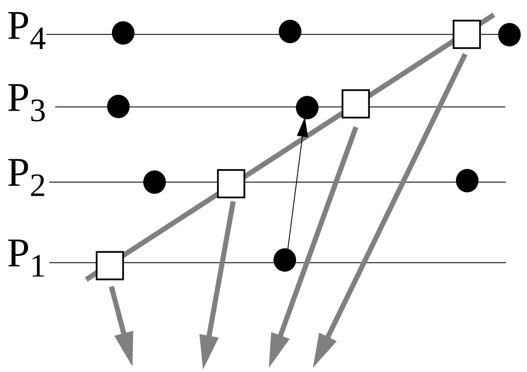
  ==> $\tau(S^*) = \tau(S)$ ==> sup(\$) = dia(\$).

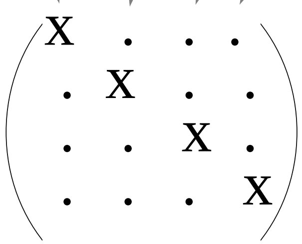
Daraus folgt das Konsistenzkriterium:

$\boxed{\text{S konsistent} \Leftrightarrow \text{dia(\$) = sup(\$)}}$

Implementierung konsistenter Schnappschüsse mit Vektorzeit?

Ein erster Ansatz: Alle Prozesse auffordern, ihren lokalen Zustand zu senden und testen, ob konsistent:





Dieses solange wiederholen, bis man einmal Glück hat...

Implementierung konsistenter Schnappschüsse mit Vektorzeit? (2)

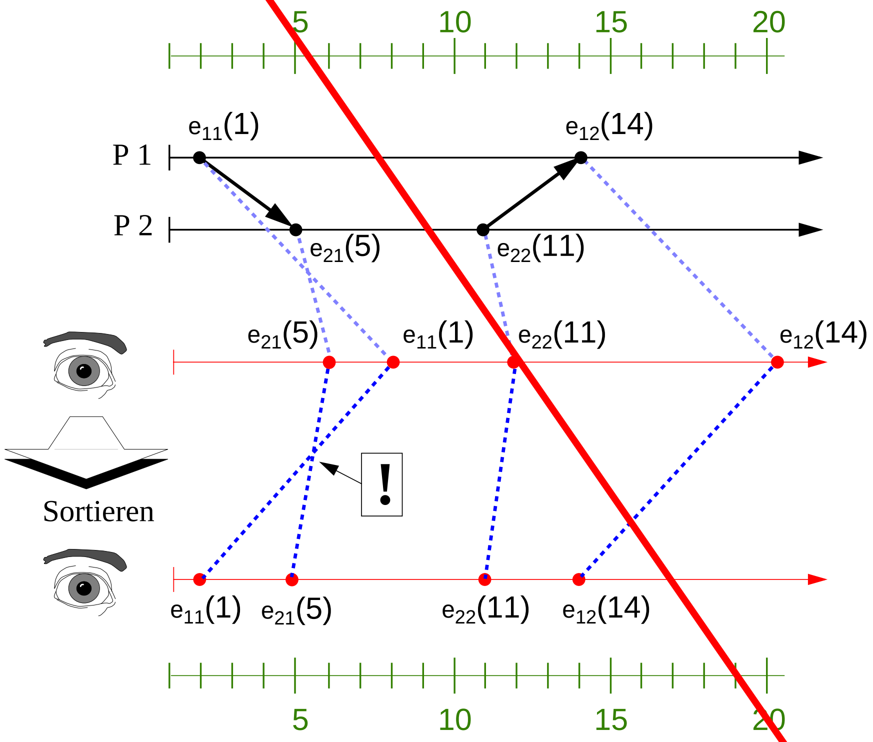
Besser: Vektoriellen *Stichzeitpunkt* in der Zukunft festlegen, bei dessen Erreichen / Überschreiten jeder Prozess seinen lokalen Zustand übermittelt

- z.Z.: so definierter Schnitt ist konsistent
- wieso ist der Stichzeitpunkt garantiert noch nicht vorbei?
- wieso wird der Stichzeitpunkt garantiert erreicht / überschritten?
  --> “konzeptioneller Trick”: Erfinde n+1-ten Prozess, dessen Uhr man voll unter Kontrolle hat... --> Optimierung...

Realisierung kausaltreuer Beobachter mit Realzeit

- Grundidee: *Zeit respektiert Kausalität*

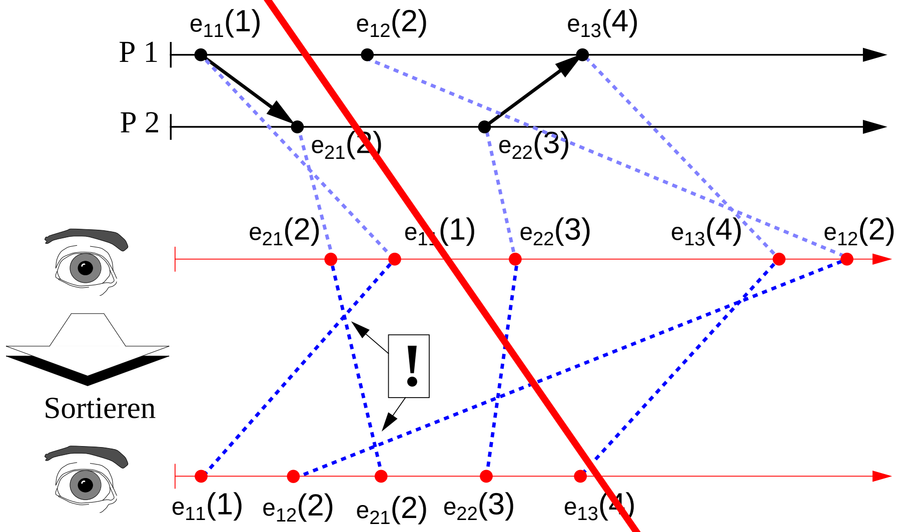
==> *Sortieren* nach globaler Zeit
= “Sortieren" nach Kausalität (--> topologisches Sortieren)



- Beobachter stellt "wahre" Berechnung wieder her
- Problem: (Globale) Realzeit wird benötigt

Realisierung kausaltreuer Beobachter mit Lamport-Zeit

- Grundidee: *Lamport-Zeit respektiert Kausalität ==>*
Sortieren liefert eine *lineare Erweiterung* der Kausalrelation

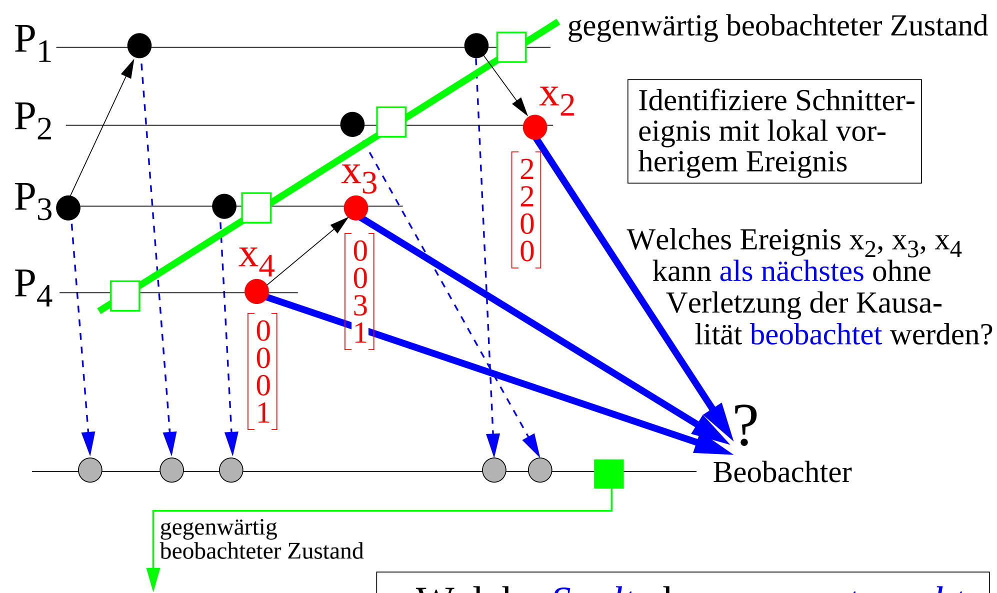


- Problem: Schlecht für *Online-Monitoring* geeignet

  - bevor man ein Ereignis "annimmt", muss man sicher sein, dass kein Ereignis mit einem kleineren Zeitstempel mehr kommt (vgl. e_{13} und e_{12})!
  - FIFO-Kanäle helfen nur beschränkt (--> lange Verzögerungen)
  - auch problematisch, wenn nur eine *Teilmenge* von Ereignissen betrachtet wird

==> Bessere Lösung?

Realisierung kausaltreuer Beobachter



$$\begin{pmatrix} \mathbf{2} & 0 & 0 & 0 \\ 0 & \mathbf{1} & 0 & 0 \\ 1 & 0 & \mathbf{2} & 0 \\ 0 & 0 & 0 & \mathbf{0} \end{pmatrix}$$

- Welche *Spalte* kann *ausgetauscht* werden? (x_2, x_4, aber nicht x_3)
- Beobachter merkt sich dia($); es muss *Zeitstempel ≤ dia($)* sein (ausgenommen die Diagonalkomponente)

- Strategie: dia($) = sup($) --> stets konsistent halten!
- Beobachter benötigt nur den Diagonalvektor, keine Matrix, um momentanen Zustand zu identifizieren

---

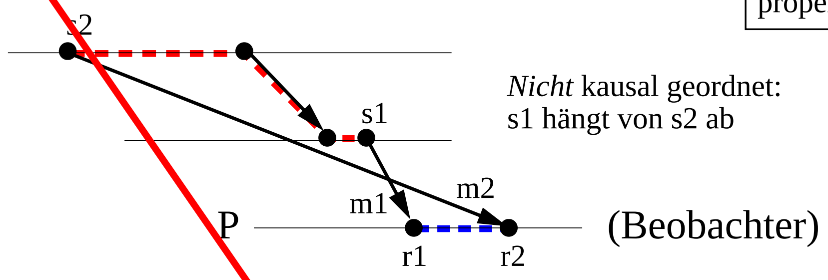
Prinzip: Verwende Vektorzeit um (indirektes) *Wissen* über "kausal frühere" Sendeereignisse zu kodieren:

"Dieses Ereignis hängt von einem anderen ab, das ich eigentlich erhalten haben müsste; also warte ich das andere erst ab..."

Kausal geordneter Nachrichtenempfang

- Empfangene Nachrichten respektieren die Kausalrelation
- Problem ähnlich zur Realisierung kausaltreuer Beobachter

causal order property



Nicht kausal geordnet:
s1 hängt von s2 ab

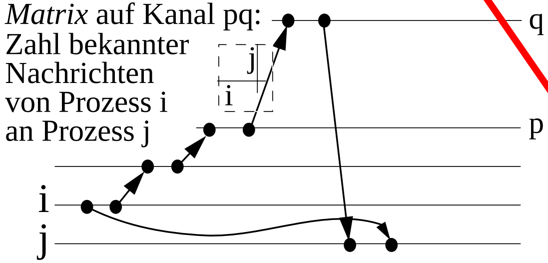
- Eine Nachricht wird nur dann an einen Prozess P ausgeliefert, wenn alle (bzgl. send-Ereignisse) kausal früheren Nachrichten an den gleichen Prozess schon ausgeliefert wurden
  - Formal: r1, r2 auf gleichem Prozess und r1 < r2 ==> s1 < s2 (wobei ri Empfangsereignis zu si ist)

!

- Kein Überholen einer einzelnen Nachricht durch eine Kette von Nachrichten ==> *"Globale FIFO-Eigenschaft"*

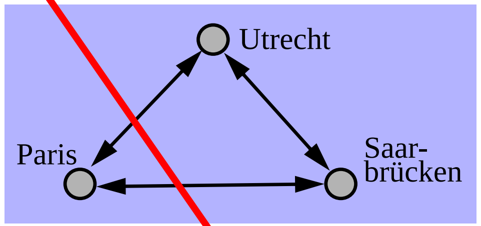
- Realisierung: *Vektor von Vektoren* ("Matrixuhr")

Matrix auf Kanal pq:
Zahl bekannter Nachrichten von Prozess i an Prozess j



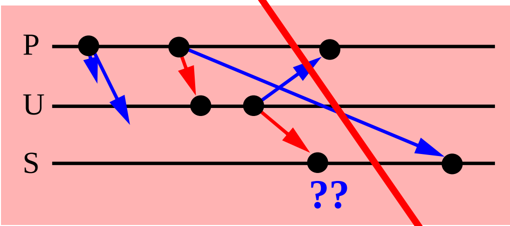
- Jeder Prozess ist ein kausaltreuer Beobachter bzgl. der Nachrichten, die er empfängt
- Schema kausaltreuer Beobachter mit n Vektoren der Länge n verwenden

Kausaltreue Nachrichtenordnung



- Diskussion von drei Personen per “broadcast”

- Verwirrungen, da indirekte Kommunikation gelegentlich schneller als die direkte ist



- Lösung: Jeder Teilnehmer soll alle relevanten Ereignisse kausaltreu wahrnehmen

- Dazu Vektorzeit verwenden (Vektor genügt, da jede Spalte der Matrix identische Elemente hat!)

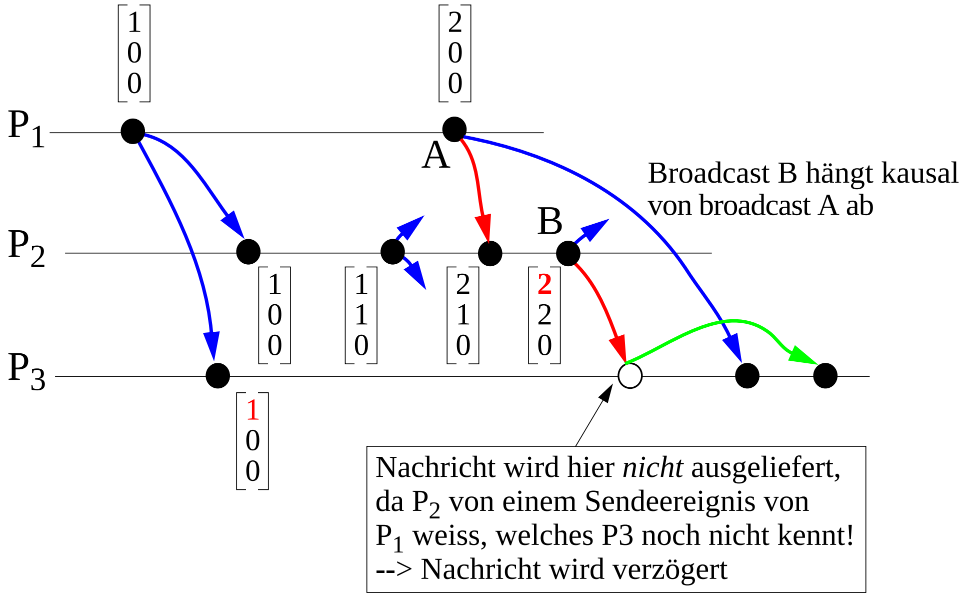
```
Date: Fri, 3 Nov 89 16:46:55 +0100
From: Bernadette Charron <charron@...fr>
To: mattern

DATE : (101,5,5)        !
Bonjour a tous,
Me revoila...

Au fait, avec vos estampilles
vectorielles, les processus “lents”
sont tout de suite detectes...On ne
peut plus dormir en silence, sans etre
repere, a moins d’accuser le reseau.

Comme j’ai BEAUCOUP reflechi, je
rajoute 100 actions internes pour ma
composante.
```


Implementierung von kausalem Broadcast



- *Prinzip*: Verwende die Vektorzeit, um (indirektes) Wissen über "kausal frühere" Ereignisse zu kodieren
  - *"dieses Ereignis hängt von einem anderen ab, das ich eigentlich erhalten haben müsste; also warte ich das andere erst ab..."*

- Nur Broadcast-Ereignisse sind relevante Ereignisse

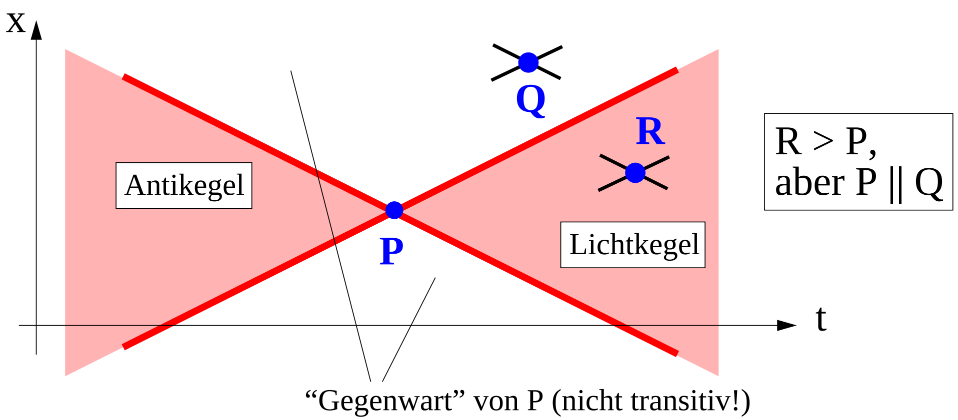
- Vektoren sind Spezialfälle der Matrixuhren
  - alle Elemente einer Spalte identisch --> Reduktion zu einem Vektor

- Verallgemeinerung des Sequenznummerverfahrens zur Implementierung von FIFO bei Nicht-FIFO-Kanälen

---

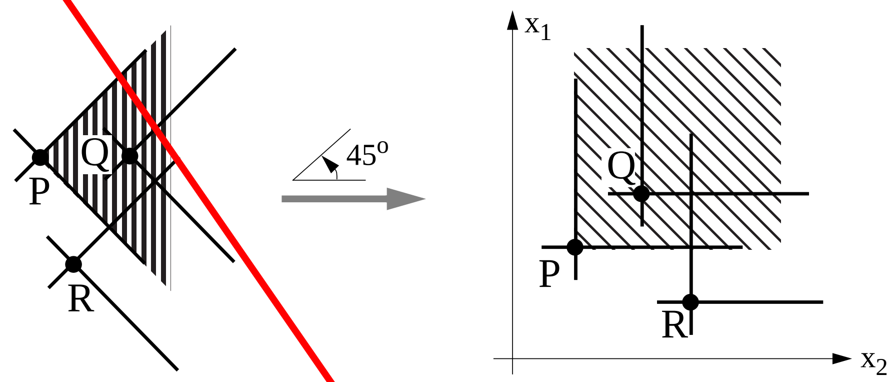
Denkübung:
Vektoren sind relativ aufwendig: Geht kausaler Broadcast, causal order, kausaltreue Beobachtung auch mit weniger aufwendigen Datenstrukturen?

Vektorzeit und Minkowski-Raumzeit



Raumzeit	*Vektorzeit*
Halbordnung	Halbordnung
2-dimensionale Kegel bilden Verband (bzgl. Schnitt)	Zeitvektoren bilden Verband (sup)
Lorentz-Transformation lässt Lichtkegel invariant	Gummiband-Transformation lässt Kausalrelation invariant
Raumzeitkoordinaten ermöglichen Test, ob potentiell kausal abhängig: Mit u= (x_1, t_1), v= (x_2, t_2) prüfe $c^2(t_2-t_1)^2 - (x_2-x_1)^2 >= 0$	Zeitvektoren ermöglichen einfachen Test, ob potentiell kausal abhängig: (prüfe, ob in allen Komponenten kleiner)

Lichtkegel- und Vektorzeit-Ordnung



90^o Lichtkegel (maximale Geschwindigkeit normiert zu “1 Raumeinheit per Zeiteinheit”, d.h.“Lichtjahr / Jahr”)

$X=(x_1,x_2)$, $Y=(y_1,y_2)$

Vektoren = Koordinaten der Punkte

- Lichtkegel von X ganz im Lichtkegel von Y enthalten (linkes Bild) $\Leftrightarrow$ $x_1 < y_1 \wedge x_2 < y_2$ (rechtes Bild) $\Leftrightarrow (x_1,x_2) < (y_1,y_2) \Leftrightarrow X < Y$.

==> 2-dimensionale Kegel ≈ 2-dimensionale Würfel

==> Zumindest bei 2 Dimensionen haben Raumzeit und Vektorzeit i.w. die gleiche Struktur!

- “später”
- potentielle Kausalität
- Verbandsstruktur

Resümee: Konzepte der Vorlesung

- Prinzipielle Phänomene und Begriffe herausarbeiten
  - Kausalität, Konsistenz, verteilte Berechnung, safety und liveness,...
- Geeignete Modelle und Abstraktionen entwickeln
  - z.B. Zeitdiagramme, Atommodell, Zustandsgitter, Gummibandtransform.
- Problemlösungs-, Analyse- und Verifikationstechniken
  - z.B. Beweis über Invarianten
- Techniken, Einsichten, Zusammenhänge,...
  - Komplexitätsanalyse
  - Transformationen zwischen Problemklassen
  - Problemverständnis von einem höheren Standpunkt

Resümee (1)

- Verteilte Systeme
- Kooperation durch Kommunikation
  - keine globale Sicht
  - keine gemeinsame Zeit
  - parallel
  - nicht-deterministisch
  - unbestimmte Nachrichtenlaufzeit
- Typische Probleme verteilter Systeme / Algorithmen:
  - Beobachtungsproblem (keine Gleichzeitigkeit)
  - Schnappschussproblem (wieviel Geld ist in Umlauf?)
  - Terminierungserkennungsproblem
  - Deadlockproblem (Phantomdeadlock?)
  - Kausalitätsproblem (indirekte Wirkung vor Ursache)
- Problem globaler Prädikate ("relativistischer Effekt")
  - es gibt i.a. mehrere "gleichberechtigte" Beobachter
  - diese stimmen i.a. bzgl. der Gültigkeit des Prädikates nicht überein!
  - gibt es beobachterinvariante Prädikate?
- Verteilter Euklidischer Algorithmus
  - als erstes Beispiel für einen verteilten Algorithmus
  - reaktives Verhalten ("nachrichtengesteuert")
  - Korrektheit der Idee / des konkreten Algorithmus? (Invarianten...)

Resümee (2)

- Zahlenrätsel
  - Parallele Constraint-Propagation
  - Abwechselnd mit Backtracking-Schritt

allgemeines Schema ("verteilte Approximation")

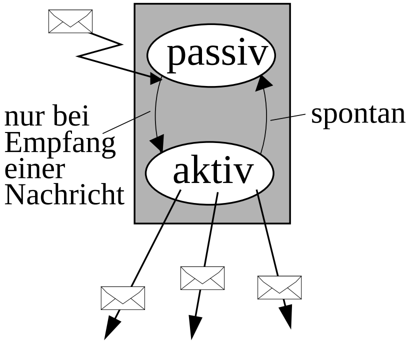
- Konzeptuelle Hilfsmittel
  - Zeitdiagramme
  - Atomare Aktionen
- Problem der verteilten Terminierung
  - Geeignete Definition?
  - Verfahren zur Feststellung?
- Flooding-Algorithmus
  - Nachrichtenzahl
  - Problem der Terminierungserkennung
  - Formalere Fassung in Pseudo-Code
- Echo-Algorithmus (Variante von Flooding)
  - Nachrichtenzahl 2e
  - Explorer- / Echo-Welle
  - Spannbaum
  - Zwei "disjunkte" Wellen (rot; grün)
  - Formalere Fassung in Pseudo-Code

Resümee (3a)

- Echo-Algorithmus
  - Verbesserung durch Mitführen von Knotenidentitäten?
- Zeitkomplexität
  - Einheitszeitkomplexität
  - Variable Zeitkomplexität
- Broadcasts auf Hypercubes
  - Hypercube: Definition und Eigenschaften
  - Einzelnachrichten: Routingverfahren, mittlere und max. Weglänge
  - Broadcast entsprechend der rekursiven Definition
  - Broadcast durch Fluten in jeweils höhere Dimensionen
  - Optimalität (Nachrichten- und Zeitkomplexität) des Broadcastproblems
  - schneller Broadcast durch paralleles Senden von Teilnachrichte
- Berechnung von Routing-Matrizen
  - verteilte Version des Bellmann-Ford-Algorithmus
  - auch wieder das bekannte Schema der verteilten Approximation
  - Anwendung in Rechnernetzen: Count to infinity-Problem, Spannbaum für lokale Netze (Zyklenfreiheit)

Resümee (3b)

- Paradigma der verteilten Approximation
  - Verallgemeinerung verschiedener ähnlicher Algorithmen
- Verteilte Terminierung
  - Problemdefinition
- Atommodell
  - Vereinfacht die Betrachtung des Wesentlichen
  - Terminierungskriterium: "keine Nachricht unterwegs"



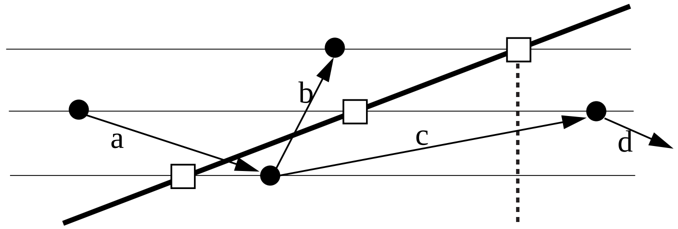
- Besprechung Übungen (1): verteilte ggT-Berechnung
  - Varianten des Algorithmus (z.B. andere Topologien)
  - Verifikationsidee (Invarianten etc.)
  - Andere Topologien (z.B. unidirektionaler Ring)

Resümee (4a)

- Schiefes Bild beim Beobachten verteilter Berechnungen

  --> Pauschales Zählen von Nachrichten genügt nicht zur Erkennung der verteilten Berechnung

  --> Suche nach den eigentlichen Ursachen für Fehlschlag des Zählkriteriums ("Kompensation" der Zähler")



- Lösungsansätze zur Terminierungserkennung
  - Durch Vermeidung der "Ursachen" für das schiefe Bild
- Terminierungserkennung: *Eindeutige Nachrichtennamen*
- Terminierungserkennung: *Kanalzählerkriterium*
  - Widerspruchsbeweis (es gibt kein frühestes Ereignis nach dem Schnitt)
- Terminierungserkennung: *Doppelzählverfahren*
  - informeller Beweis (Aussage über gedachten senkrechten Schnitt zwischen den beiden Wellen)
  - formalerer Beweis in den Folienkopien
  - Eigenschaften
- Diskussion: Terminierungserkennung durch *Einfrieren*?

Resümee (4b)

- Safety- und Liveness-Eigenschaften verteilter Algorithmen

- Kontrolltopologien zur Realisierung von Schnitten
  - Ring
  - Spannbaum
  - Echo-Algorithmus als zugrundeliegendes Basisverfahren (Hin- und Rückwelle für die beiden Schnitte des Doppelzählverfahrens!)

- Terminierungserkennung: *Zeitzonenverfahren*
  - Prinzip: *Erkenne* "Nachricht aus der Zukunft"
  - binäre "schwarz/weiss"-Zeit genügt
  - Qualitätseigenschaften im Vergleich zu anderen Verfahren

Resümee (5)

- Diskussion: Terminierungserkennung durch *Einfrieren*?
- Terminierungserkennung: Vermeiden inkonsistenter Schnitte durch geeignetes *Vorziehen der Schnittlinie*
- Synchrones / asynchrones Senden
  - synchron: senkrechte Nachrichtenpfeile sind gerechtfertigt
  - nicht alles geht synchron (z.B. Überholen von Nachrichten)
- Besprechung Teile von Übung 1
  - Formalisierung von Zeitdiagrammen und "kausal abhängig"
  - kausaltreue Beobachtungen als lineare Erweiterungen ("Einbettung") der halbgeordneten Kausalitätsrelation
- Charakterisierung synchroner Kommunikation
  - alle Nachrichtenpfeile können senkrecht gezeichnet werden; Kommunikationskanäle sind immer leer
  - es gibt eine lineare Erweiterung der Kausalitätsrelation, so dass ein Empfangsereignis immer direkt nach seinem Sendeereignis kommt
  - Senden und Empfangen bilden "atomare Einheit"
  - Nachrichten-Scheduling-Relation (m < n gdw. send(m) < receive(n)) ist zyklenfrei
  - Zyklenfreiheit der "synchronen Kausalitätsrelation <<" ("common past" / " common future"); dadurch Identifizierung von send und receive

zusammengehörige send/receive-Ereignisse sind "in gewissem Sinne" atomar

- Fragen...
  - sind die Charakterisierungen alle äquivalent?
  - kann man nun Nachrichtenlaufzeiten immer vernachlässigen?
  - funktioniert ein Algorithmus, der unter der Voraussetzung synchroner Kommunikation gemacht wurde, auch bei asynchroner Kommunikation?
  - und umgekehrt?
  - Terminierungserkennung bei synchroner Kommunikation? (das Atommodell ist dann offenbar nicht mehr adäquat, oder?)

Resümee (6a)

- Def. verteilte Terminierung bei synchroner Kommunikation

X_p: {$state_p$ = *aktiv*} $state_q$:= *aktiv* (* "atomares" aktivieren *)

I_p: $state_p$:= *passiv*

- Verhaltensmodelle verteilter Anwendungen

  - Transaktionsmodell
  - Atommodell
  - Synchronmodell

  gegenseitige Simulation bzw. Transformation der Modelle

- Modelle in der Informatik

  - nicht nur zum Erkenntnisgewinn, zur Simulation etc., sondern auch Implementierung von "ausgedachten, idealisierten Wirklichkeiten"

- Terminierungserkennung bei synchroner Kommunikation

  - z.B. Erkennen einer senkrecht von oben nach unten laufenden Nachricht, die einen schrägen Schnitt ("Welle") überquert

- Algorithmus von Dijkstra et al. ("DFG")

  - schwarz / weiss-Färbung; Token auf einem Kontrollring
  - Beschreibung durch Menge von Verhaltensregeln
  - Überlegungen zu Korrektheit, Varianten, Nachrichtenkomplexität

Resümee (6b)

- Parallele Berechnungsschemata

  - Bsp.: Integration mittels Trapezmethode
  - Lastausgleich durch Migration von Arbeitseinheiten
  - Gesamtlast = 0 <==> Terminierung

- Terminierungserkennung mit der Kreditmethode
  (Halbieren von Tickets; Einsammeln von "Krümeln")

- Safety: "Gesamtkredit" ist invariant

- Realisierung in verschiedenen Varianten möglich:

  - geeignete Darstellung der Krümel (negativer Zweierlogarithmus)
  - geeignete Realisierung des Einsammelns (Liveness!)
  - geeignete Verwaltung der Krümel bei den Prozessen
  - geeignete Informationsverwaltung im Urprozess

- Nachrichtenkomplexität: Worst-case-optimal

- Variante: direktes Nachlaufen

  - Analogie zum Echo-Algorithmus!

Resümee (7)

- Wechselseitiger Ausschluss
    - safety
    - liveness
    - fairness

- Maekawa's O($\sqrt{n}$)-Algorithmus
    - Prinzip: Gitteranordnung; Request-granting-Mengen

- Token-basierte Lösungen

- Algorithmus von Ricart / Agrawala 1983
    - Anforderungsnachrichten enthalten Zeitstempel
    - Token hat Auftragsliste und merkt sich Zeitpunkt des letzten Besuchs für alle bescuhten Prozesse

- Token-basierte Lösungen auf speziellen Topologien
    - Spannbaum / Baum: Umdrehen durchlaufener Kanten ("path reversal")
    - "Lift-Algorithmus" --> O(log n) bei "guten" Bäumen
    - Verallgemeinerung auf beliebige (gerichtete azykl.) Graphen
    - Invarianten: Zyklenfreiheit; alle Pfade führen zum Tokenbesitzer
    - Variante: Nachbarn informieren, dass Token "jetzt" hier ist
    - Request holt Token stets ein

- Besprechung von Teilen von Übung 3
    - falscher Terminierungserkennungsalgorithmus
    - es genügt nicht, nur über den Zustand seiner Nachbarn informiert zu sein

Resümee (8a)

- Wechselseitiger Ausschluss: *token-basierte* Algorithmen
    - Zurückholen des Tokens in azyklischen gerichteten Graphen
    - spezielle Topologien (Ring; Stern; lineare Kette)
    - Nachrichtenkomplexität bei starker Last ($\approx$4)

- Vergleich von Algorithmen für den wechselseitigen Ausschluss (quantitative und qualitative Kriterien)

- *Election-Problem*: Symmetriebrechung
    - Auswahl genau eines Prozesses aus mehreren (bis auf die eindeutige Identität) gleichartigen

- Election-Algorithmus mit dem Message-extinction-Prinzip
    - verteiltes Approximationsschema
    - funktioniert auf allgemeinen (zusammenhängenden) Graphen
    - aber: Problem der Terminierungserkennung

- Election-Algorithmus auf (unidirektionalem) Ring
    - nur grösste Identität schafft Ringumlauf --> ist damit "gewählt"
    - Bully-Algorithmus, oder besser:
    - message-extinction (beim Ring kein Terminierungserkennungsproblem!) ---> Chang/Roberts-Algorithmus

Resümee (8b)

- Chang/Roberts-Algorithmus auf unidirektionalem Ring
  - Worst-case-Nachrichtenkomplexität $O(n^2)$

- Chang/Roberts-Algo.: Mittlere Nachrichtenkomplexität?
  - Wahrscheinlichkeit, genau i Positionen weit zu kommen
  - Erwartungswert für die Länge der Nachrichtenkette = H_n
  - mittlere Nachrichtenkomplexität = nH_n (= ca. n ln n)

- Besprechung von Teilen von Übung 4
  - Wartezeit bis zum ersten Rekord
  - Simulation eines "unendlichen" Erwartungswertes??

Resümee (9)

- Bidirektionale Varianten des Chang/Roberts-Algorithmus
  - probabilistisch
  - mittlere Nachrichtenkomplexität

- Algorithmus von Hirschberg und Sinclair (bidir. Ring)
  - sukzessive grössere Gebiete erobern
  - worst-case Nachrichtenkomplexität < $8 \, n \log_2 n$

- Petersons Election-Algorithmus (bidir. Ring)
  - solange sukzessive Identität in beide Richtungen senden, bis man von einem grösseren Nachbarn erfährt
  - mittlere Nachrichtenkomplexität ca. $2 \, n \log_3 n$
  - Simulation ("kostenneutral"!) auf einem unidirektionalen Ring
  - Variante mit abwechselnden Richtungen
  - worst-case Nachrichtenkomplexität (ca. $1.44 \, n \log_2 n + c$) mittels Fibonacci-Folge abgeschätzt

- Election auf Bäumen
  - Explosionswellen vereinigen sich
  - Explosionswelle wird an den Blättern reflektiert
  - Kontraktionsphase endet in zwei Zentrumsknoten
  - Nachrichtenkomplexität O(n)

- Echo-Election auf allgemeinen Graphen
  - Idee wie Chang/Roberts, aber Echo-Algorithmus statt Ringumlauf

- Nachrichtenkomplexität des Election-Problems
  - mindestens e Nachrichten

Resümee (10a)

- Verteilte Spannbaumkonstruktion
    - Zusammenhang zum Election-Problem ("gleich schwierig")
- Anonyme Netze
    - De-Anonymisierung
- Election in anonymen Netzen
    - kein stets terminierender (deterministischer) Algorithmus möglich
- Probabilistische Algorithmen
    - Las Vegas (terminiert nicht immer, Ergebnis ist aber korrekt)
    - Monte Carlo (terminiert, aber ggf. mit falschem Ergebnis)
- Probabilistische Election-Algorithmen
    - Verfahren mit Zufallsidentität
    - Implementierung reellwertiger Zufallszahlen zwischen 0 und 1?

- Kausaltreue Beobachtungen
    - Beispiel: Aussterben aller Exemplare eines "Typs" (--> Terminierung)
    - Analogie: konsistente Referenzzähler (--> Garbage-Collection!)
    - Lösungen: Synchrone Kommunikation oder getrennte FIFO-Puffer pro Prozess für das Empfangen und Senden von Nachrichten

---

- Garbage-Collection: Modellierung
    - Objekte und Zeiger; Wurzelobjekte
    - nicht mehr von der Wurzel erreichbar --> Garbage
    - rekursives Freigeben (Zyklen bleiben übrig!)
    - *Mutator* (new, copy, delete: Manipulation von Zeigern)
    - *Collector* soll Garbage-Objekte identifizieren

Resümee (10b)

- Garbage-Collection: Grundverfahren
    - Paradigmen: "stop the world" / on the fly (= "parallel")
    - "Mark and sweep"-Verfahren
    - bei paralleler Variante: Problem mit "behind the back copy"
      ==> Mutator / Collector müssen sich koordinieren!
      (sonst bekäme der Collector ggf. ein "schiefes Bild")
- Verteiltes Garbage-Collection (= GC in verteilten Systemen)
    - Referenzen u.U. "in transit"
    - copy nicht mehr atomar ("send/receive copy")
    - increment / decrement per Nachricht (z.B. an den Ort des Referenzzählers)
    - inc bzw. dec daher nicht "gleichzeitig" mit copy bzw. delete
    - Unterschied zwischen lokalen und "remote" Referenzen
    - lokales und globales GC (dezentral, echt parallel, typw. hierarchisch)
- Formalisierung des GC-Problems: Operationen C_p, R_p, D_p
- Referenzzähler-Verfahren
    - Problem: "zyklischer Garbage" wird nicht entdeckt
    - bei verteilter Variante: Problem bei decrement *vor* increment

  zeitlich?
  kausal?
- Lösungen für verteiltes Reference-Counting:
    - prinzipiell: Causal Order garantieren (d.h. indirekte Überholungen vermeiden)
    - "naiv": auf Bestätigung jeder Increment-Nachricht warten
    - ...?

Resümee (11)

- Lösungen für verteiltes Reference-Counting:
  - Varianten von Lermen/Maurer und Rudalics (zwei bis vier Nachrichten pro copy-Operation)

- Weighted Reference Counting (WRC)
  - Kopieren ohne Zusatznachricht: Splitten des Reference Weight

- Analogie (verteiltes) GC <==> verteilte Terminierung

- Transformation GC-Algorithmus --> Algorithmus zur Erkennung der verteilten Terminierung
  - Umformung des Terminierungsproblems in ein GC-Problem
  - darauf gegebenen GC-Algorithmus ansetzen

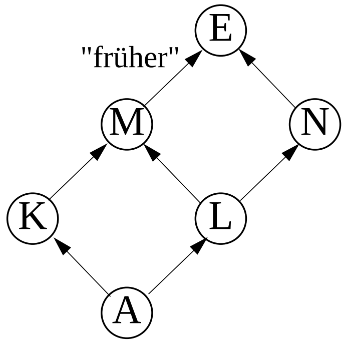
- Zum Patent der Referenzgewichtsmethode (WRC)

- Patentieren von Algorithmen

- Local Reference Counting (LRC)
  - jede Maschine besitzt für *jedes* Objekt einen (lokalen) Zähler
  - logische Baumstruktur ("Verantwortlichkeit")
  - viele interessante Eigenschaften
  - Migration von Objekten "leicht" zu unterstützen
  - lokal u.U. ein anderes GC-Verfahren (nur global LRC verwenden)
  - IRT / ORT-Tabellen ("Proxy-Objekte"; Bündelung von Referenzen)
  - Übung: Transformation in den Terminierungserkennungsalgorithmus von Dijkstra und Scholten (Literatur nachlesen)

Resümee (12a)

- *Verteilte Berechnungen*: Formale Definition (Modellierung!)
  - Partition von Ereignissen, Sende/Empfangsereignisse, Kausalrelation
  - Zeitdiagramme von verteilten Berechnungen; Gummibandtransformation
  - Globale Zustände als Endzustände von Präfixberechnungen (Präfixberechnungen sind linksabgeschlossen bzgl. der Kausalrelation)
  - Menge der Zustände (bzw. Präfixberechnungen) bilden Verband



Berechnung läuft entlang eines "unbestimmten" Weges vom Anfangszustand A zum Endzustand E.

Resümee (12b)

- Wellenalgorithmen

- Information verteilen / einsammeln; Phasen trennen; Ereignisse triggern...
- Formale Def: ... init < visit_i < conclude ...
- Visit-Ereignisse bilden einen *Schnitt* (wann senkrechte Schnittlinie möglich?)
- Bsp.: Echo-Algorithmus, Ring, Stern...
- min. n-1 Nachrichten, min e Nachrichten bei unbekannten Nachbarn
- Spannbaum = jeweils erste empfangene Nachricht eines Knotens

- Virtuell gleichzeitiges Markieren mittels flooding

- Voraussetzung: FIFO-Kanäle

- "Konsistente" Schnittlinien lassen sich senkrecht zeichnen

- konsistent: keine Nachricht läuft "rückwärts" über die Schnittlinie

- Sequentielle Traversierungsverfahren

- spezielle Wellenalgorithmen: visit-Ereignisse linear geordnet

- Algorithmus von Tarry (Labyrinth-Problem)

- Beweisskizze, dass Tarry-Algorithmus ein Traversierungsverfahren ist
- Depth-First-Search ist Spezialfall des Tarry-Algorithmus

Resümee (13a)

- Globale konsistente Schnitte / Zustände

- Schnappschussproblem und -algorithmen

(1) Färben von Prozessen / Nachrichten; Vermeiden von "Tachyonen"; In-Transit-Nachrichten durch Abgleich von Sende-/Empfangspuffern oder durch Weiterleiten von Kopien an den Initiator

(2) Chandy/Lamport-Algorithmus: Flooding; FIFO-Kanäle ("flushing"); Problem (?): einige Kanäle sind scheinbar immer leer

- Beobachten verteilter Berechnungen

- Wunsch: lückenlos konsistente Schnappschüsse anzeigen
- rekonstruiertes Bild des Beobachters
- ideale und kausaltreue Beobachter

- Kausaltreues Beobachten

- Beispiele für kausal inkonsistente Beobachtungen
- Def. kausaltreuer Beobachter
- Pfade im n-dimensionalen Zustandsgitter ("Hyperwürfel")

- Entdecken globaler Prädikate durch Beobachtung

- Abhängigkeit von konkreten Beobachtungen ("possible worlds")
- Wirkung von Handshake- und Barrier-Synchronisation

Resümee (13b)

- Stabile Prädikate

- Schnitte und Vektorzeit
  - Später- / Früher-Relation auf Schnitten
  - Definition konsistenter Schnitte als linksabgeschlossene Ereignismengen
  - Zeitstempel eines Ereignisses als Menge seiner kausalen Vorgänger (Repräsentation durch lokal letztes Ereignis --> Vektorzeit)

- Vektorzeit
  - Interpretation: repräsentiert gesamte kausale Vergangenheit
  - Zeitstempelarithmetik
  - Implementierung (Supremum beim Empfang)
  - Isomorphie der Zeit- und Kausalstruktur

- Anwendung der Vektoruhren
  - kausaltreue Beobachtungen

- Relativistische Struktur der Vektorzeit

Resümee: Themen der Vorlesung

- Beispiele für verteilte Berechnungen und Algorithmen
  - verteilte ggT-Berechnung
  - verteiltes Lösen von Zahlenrätseln
  - verteilte Approximation

- Grundalgorithmen
  - Flooding
  - Echo-Algorithmus (Wellenalgorithmus, Spannbaum)

- Verteilte Terminierung

  Grundphänomen "inkonsistenter Sicht"; nur problemspezifische Lösungen dafür

  - Doppelzählverfahren
  - Zeitzonenverfahren
  - für synchrone Kommunikation: DFG-Verfahren; "sticky flags"
  - Kreditmethode

- Wechselseitiger Ausschluss

  Synchronisation in vert. Sys. (viele wollen, einer darf; Sicherheit, Deadlockfreiheit, Fairness)

  - Grundprinzipien
  - Maekawa
  - Token-basierte Verfahren

- Election

  Symmetriebrechung in vert. Sys.: verteilte Wahl eines "Repräsentanten"

  - Chang/Roberts-Verfahren (Ring); bidirektionale Varianten
  - Hirschberg/Sinclair und Peterson's Algorithmen: O(n log n) worst case
  - Election auf Bäumen
  - untere Schranke O(e) für Nachrichtenkomplx. bei allg. Netzen
  - Election in anonymen Netzen (probabilistische Algorithmen)

Resümee: Themen (2)

- Garbage-Collection
  - Mutator, collector, Formalisierung
  - Behind the back copy
  - Verteiltes Garbage-Collection
  - Referenzzähler (verschiedene Lösungen; z.B. WRC, LRC)
  - Implementierungstechniken

- Garbage-Collection ==> Terminierungserkennung

- Wellenalgorithmen
  - Eigenschaften, Spannbäume,...

- Sequentielle Traversierungsverfahren
  - Methode von Tarry (1895)
  - Depth-first: Varianten

- Parallele Traversierungsverfahren
  - Verteilen von Information ("flooding"), Echo-Algorithmus

Resümee: Themen (3)

- Schnappschuss, Konsistenz, Beobachtungen, Prädikate,...
  - Kausalrelation, kausale Vergangenheit...
  - Halbordnung, Verband,...
  - Schnitt, globaler Zustand
  - Kausal konsistente Beobachtung
  - Globale Prädikate, stabile Prädikate
  - Schnappschussalgorithmen

- Logische Zeit
  - Uhrenbedingung
  - Lamport-Uhren
  - Vektorzeit: Eigenschaften und Implementierung
  - Schnittmatrix
  - Konsistenzkriterium
  - Implementierung von kausal konsistenten Beobachtern
  - Schnappschuss mit Vektorzeit
  - Analogie zur Raumzeit