

Folienkopien zur Vorlesung
Verteilte Algorithmen

F. Mattern

Departement Informatik, ETH Zürich

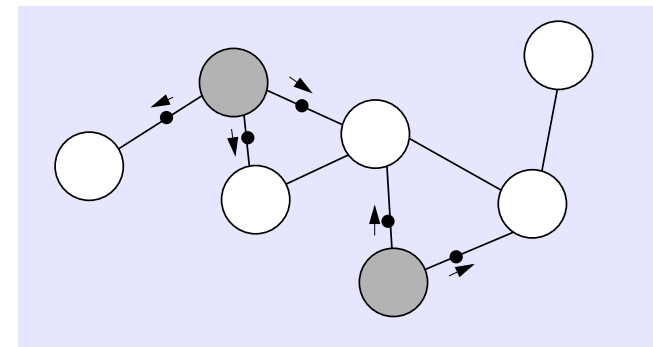
WS 03/03 (3-stündig inkl. Übungen)

© F. Mattern, 2003

Verteilte Algorithmen

F. Mattern

Departement Informatik
ETH Zürich



Verteilte Algorithmen

= Algorithmen für / in verteilten Systemen

Grundkonzept der Informatik
(praktisch / theoretisch)

Interessantes Gebiet der
praktischen Informatik
sowie wichtiges Anwendungsfeld

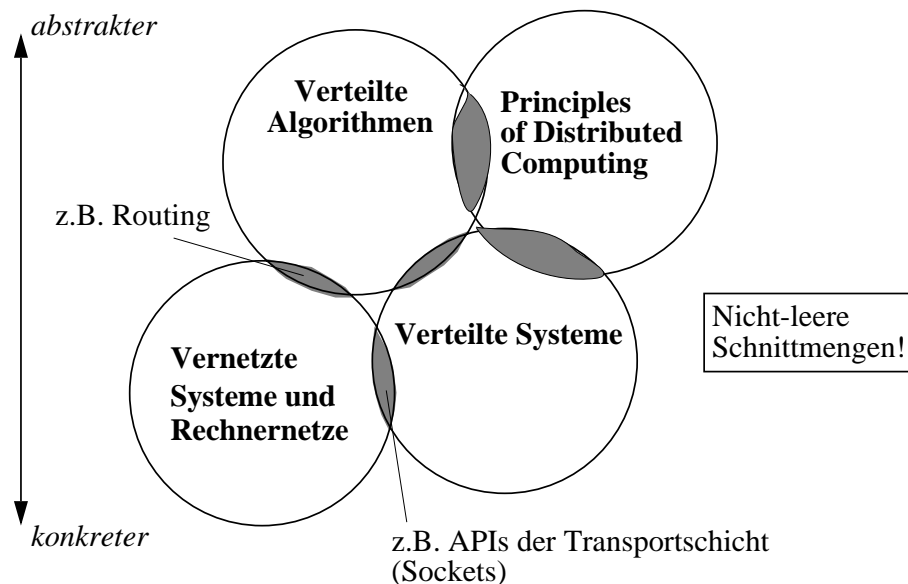
Achtung: Prüfungsrelevant ist der Inhalt der Vorlesung, nicht alleine der Text dieser Foliensammlung!

- Was ist "verteilt" an einem verteilten Algorithmus?

Vorläufige Antwort: *Zustand* und *Kontrolle* auf verschiedene "Orte"

Einordnung der Vorlesung

- 3-stündige Vorlesung (inkl. Übungen)
- Sinnvolle Vorkenntnisse:
 - Verteilte Systeme (< 10% Überschneidung)
 - Grundkenntnisse der Informatik und Mathematik (Vordiplom)
- Folienkopien jeweils in der darauffolgenden Vorlesung (später auch im WWW im .ps- bzw .pdf-Format)
 - www.inf.ethz.ch/vs/edu/WS0304/VA/



- Mobile Computing
- Ubiquitous Computing
- ...

Vorlesungsankündigung "Verteilte Algorithmen"

Verteilte Algorithmen sind Verfahren, die dadurch charakterisiert sind, dass mehrere autonome Prozesse gleichzeitig Teile eines gemeinsamen Problems in kooperativer Weise bearbeiten und der dabei erforderliche Informationsaustausch ausschliesslich über Nachrichten erfolgt. Derartige Algorithmen kommen im Rahmen verteilter Systeme zum Einsatz, bei denen kein gemeinsamer Speicher existiert und die Übertragungszeit von Nachrichten i.a. nicht vernachlässigt werden kann. Da dabei kein Prozess eine aktuelle konsistente Sicht des globalen Zustands besitzt, führt dies zu interessanten Problemen.

Im einzelnen werden u.a. folgende **Themen** behandelt:

Modelle verteilter Berechnungen;
Raum-Zeitdiagramme;
virtuelle Zeit; logische Uhren und Kausalität;
Wellenalgorithmen;
verteilte und parallele Graphtraversierung;
Berechnung konsistenter Schnappschüsse;
wechselseitiger Ausschluss;
Election und Symmetriebrechung;
verteilte Terminierung;
Garbage-Collection in verteilten Systemen;
Beobachten verteilter Systeme;
Berechnung globaler Prädikate.

Literatur:

- F. Mattern: Verteilte Basisalgorithmen. Springer-Verlag, 1989.
- G. Tel: Topics in Distributed Algorithms. Cambridge University Press, 1991.
- G. Tel: Introduction to Distributed Algorithms. Cambridge University Press, 2nd edition, 2000.
- [] V. Barbosa: An Introduction to Distributed Algorithms, MIT Press, 1996.
- N. Lynch: Distributed Algorithms, Morgan Kaufmann Pub., 1996.
- V. K. Garg: Principles of Distributed Systems, Kluwer, 1996.

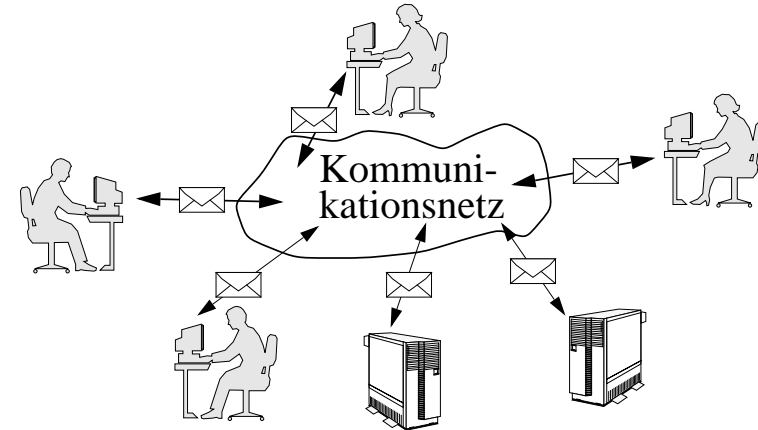
Artikel aus Fachzeitschriften (wird in der Vorlesung bekanntgegeben).

Themenspektrum

- Wellenalgorithmen, Traversierungsverfahren
- Verteilte Terminierung
- Verteiltes Garbage-Collection
- Globaler Zustand, Schnappschuss
- Konsistente Beobachtungen und globale Prädikate
- Logische Zeit, Kausalität, Konsistenz
- Wechselseitiger Ausschluss
- Election
- Kommunikationssemantik (synchron, asynchron, kausal)

Algorithmen

Verteiltes System



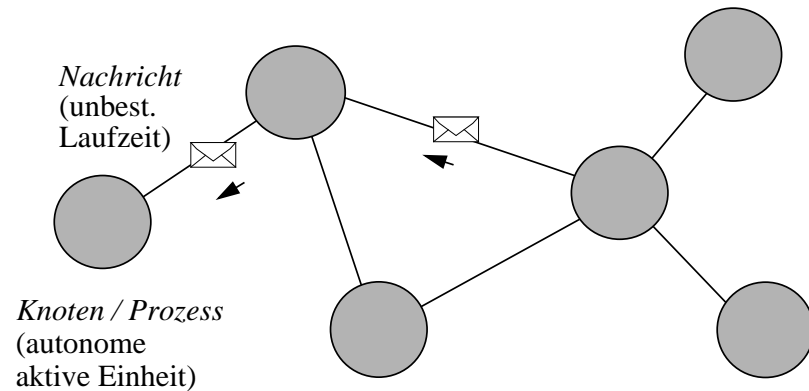
- Prinzipielle Phänomene und Begriffe
 - Kausalität, Parallelität,...
- Anwendungsbezug
 - verteilte Betriebssysteme, Datenbanken, Anwendungen, Tools...

- Rechner, Personen, Prozesse, "Agenten" sind an *verschiedenen Orten*
- Autonome Handlungsträger, die jedoch gelegentlich *kooperieren* (und dazu *kommunizieren*)

- Techniken, Einsichten, Zusammenhänge,...
- Basisverfahren, grundsätzliche Probleme,...
- Abstraktion, Modellbildung, Formalisierung,...
- Problemlösungstechniken, Analysetechniken,...
- Qualitätsbewertung, Komplexitätsabschätzung,...
- Verifikationstechniken

(verallgemeinerbare)
Erkenntnisse

Verteiltes System II



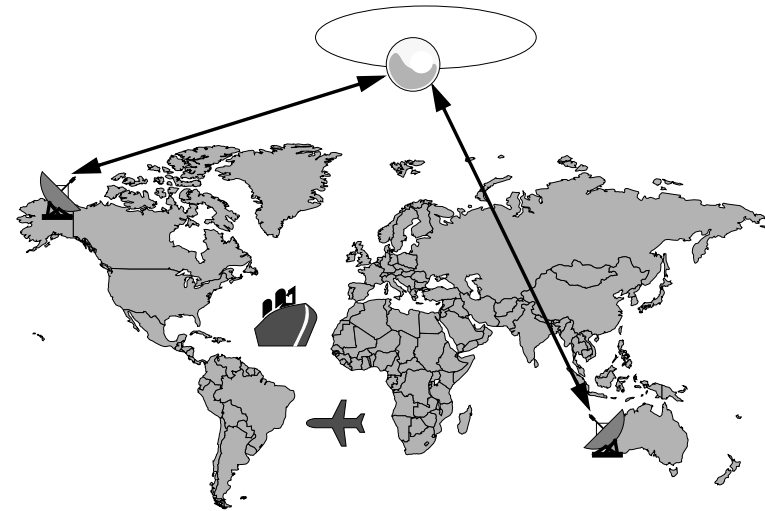
Physisch verteiltes System:

Mehrrechnersystem ... Rechnernetze

Logisch verteiltes System: Prozesse (Objekte, Agenten)

- Verteilung des Zustandes (keine globale Sicht)
- Keine gemeinsame Zeit (globale, genaue "Uhr")

Verteiltes System III



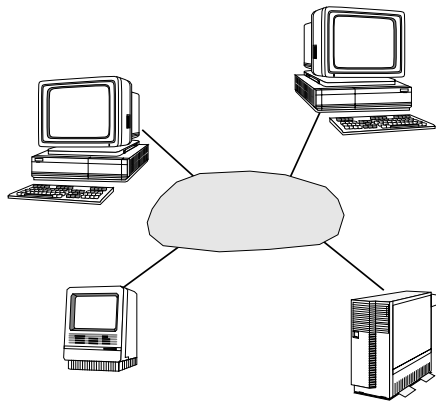
Auch die "reale Welt" ist ein verteiltes System:

- Viele gleichzeitige ("parallele") Aktivitäten
- Exakte globale Zeit nicht erfahrbar / vorhanden
- Keine konsistente Sicht des Gesamtzustandes
- Kooperation durch explizite Kommunikation
- *Ursache* und *Wirkung* zeitlich (und räumlich) getrennt

- "Inkonsistente Zustände": Kriegsende zwischen England und Frankreich war in den Kolonialgebieten erst später bekannt!

- Heute: "zeitkompakter Globus" - weitgehend synchronisierte Uhren

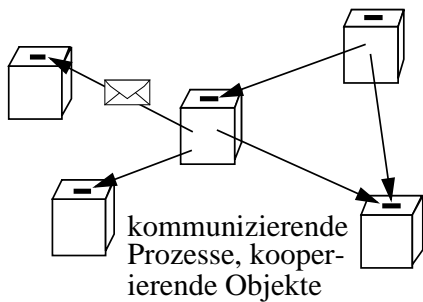
Sichten verteilter Systeme



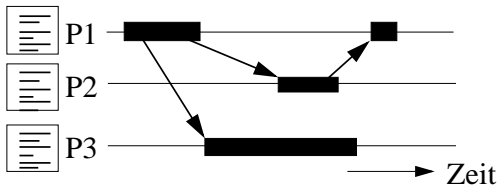
Rechnernetz mit Rechenknoten:

- Multicomputer (Parallelrechner)
- LAN = Local Area Network
- WAN = Wide Area Network
- Routing, Adressierung....

zunehmende Abstraktion



Objekte eines Betriebssystems bzw. einer Programmiersprache
 ==> "Programmierersicht" (Client, Server...)



Algorithmen- und Protokoll-ebene

- Aktionen, Ereignisfolgen
- Konsistenz, Korrektheit

Parallele <--> verteilte Algorithmen

Sequentieller Algorithmus

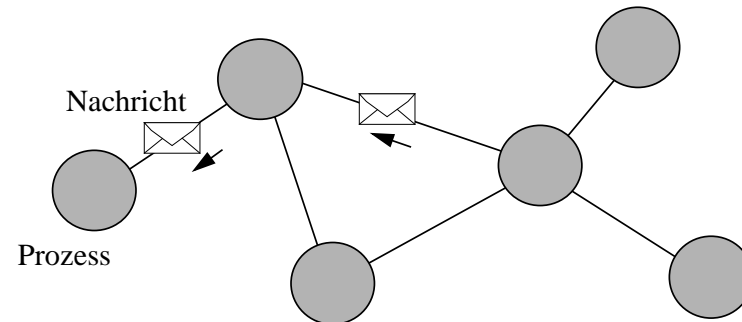
Paralleler Algorithmus

- mehrere, oft gleichartige Prozesse
- Synchronisation oft über gemeinsamen Speicher
- Zweck: Beschleunigung gegenüber seq. Algorithmus

Verteilter Algorithmus

- mehrere Prozesse kommunizieren über Nachrichten
- gemeinsames Ziel --> Kooperation
- ideal: keine zentrale Kontrolle

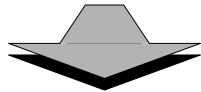
oft asynchron



Aspekte verteilter Systeme

im Vergleich zu *sequentiellen* Systemen:

- Grösse und Komplexität → jede(r) ist anders
- Heterogenität →
- Nebenläufigkeit → vieles gleichzeitig
- Nichtdeterminismus → morgen anders als heute...
- Zustandsverteilung → niemand weiss alles



- Programmierung *komplexer*
- Test und Verifikation *aufwendiger*
- Verständnis der Phänomene *schwieriger*

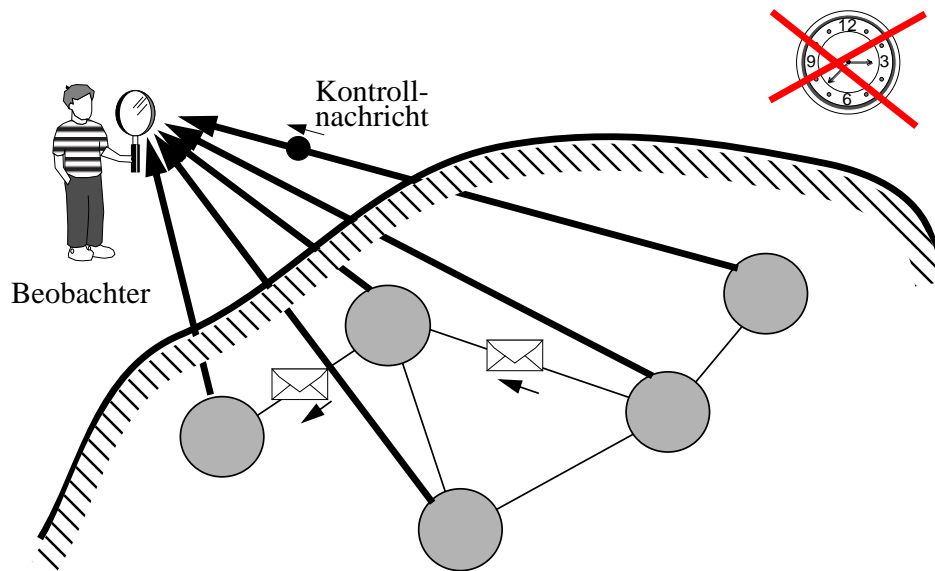
==> gute Werkzeuge (“Tools”) und Methoden
- z.B. Middleware als Software-Infrastruktur

==> adäquate Modelle, Algorithmen, Konzepte
- zur Beherrschung der neuen Phänomene

Ziel: Verständnis der grundlegenden Phänomene,
Kenntnis der geeigneten Konzepte und Verfahren

Phänomene verteilter Berechnungen

Beobachten verteilter Berechnungen



Beobachten geht nur über das Empfangen von "Kontrollnachrichten" (mit unbestimmter Laufzeit)

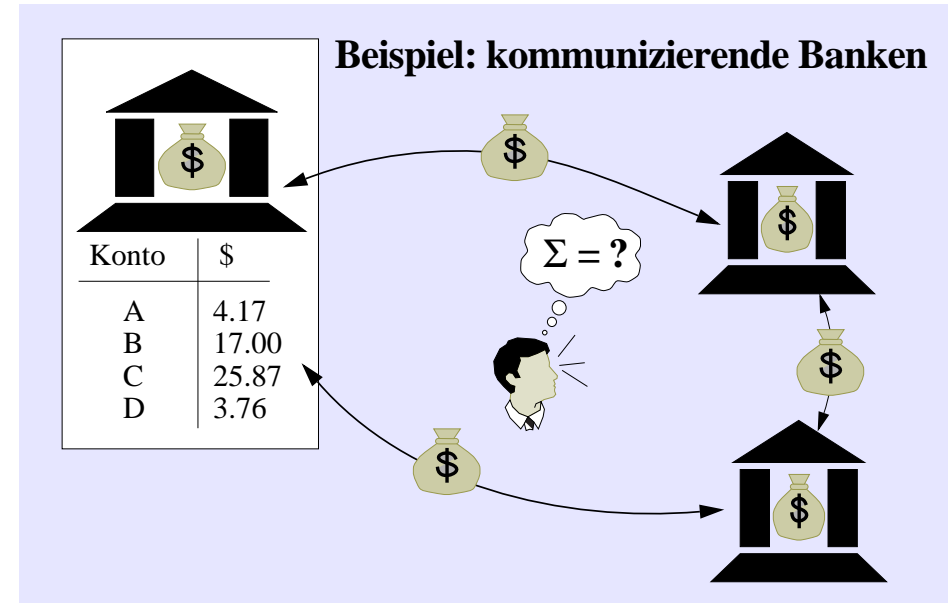
"Axiom": Mehrere Prozesse können "niemals" gleichzeitig beobachtet werden

was heisst das?

"Korollar": Aussagen über den globalen Zustand sind schwierig

Ein erstes Beispiel: Wieviel Geld ist in Umlauf?

- konstante Geldmenge, oder
- monotone Inflation (--> Untergrenze)



- Modellierung:

- verteilte Geldkonten
- ständige Transfers zwischen den Konten

- Erschwerte Bedingungen:

- niemand hat eine globale Sicht
- es gibt keine gemeinsame Zeit ("Stichtag")

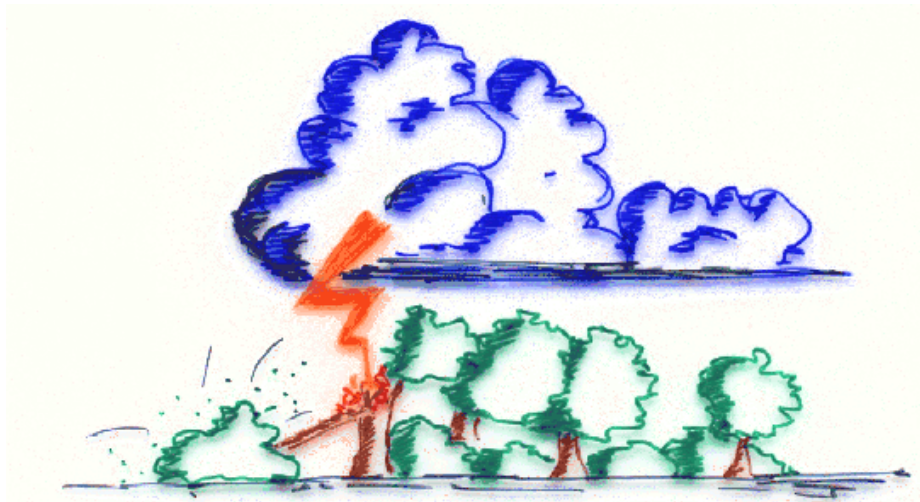
--> Geht das dann überhaupt?

--> Ist das überhaupt ein wichtiges Problem?

==> Schnappschussproblem

Beispiel: Prähistorische Gesellschaft

- Einzelne, versprengte Stämme
- Beschränktes technisches Wissen
 - Feuermachen unbekannt
 - Feuerhüter ist ein angesehener Beruf!
- Feuer erlischt--> von einem anderen Stamm holen
- Nur lokale Sicht
 - Glimmt noch ein Fünkchen Hoffnung?
 - Oder ist der Ofen aus?
- Alle Feuer erloschen und kein Feuerträger unterwegs:
 - Warten auf einen Blitz (--> grosser Feuerzauber...)



- Feststellen der Terminierung ist wichtig
 - kein warmes Essen bis zum nächsten Gewitter...

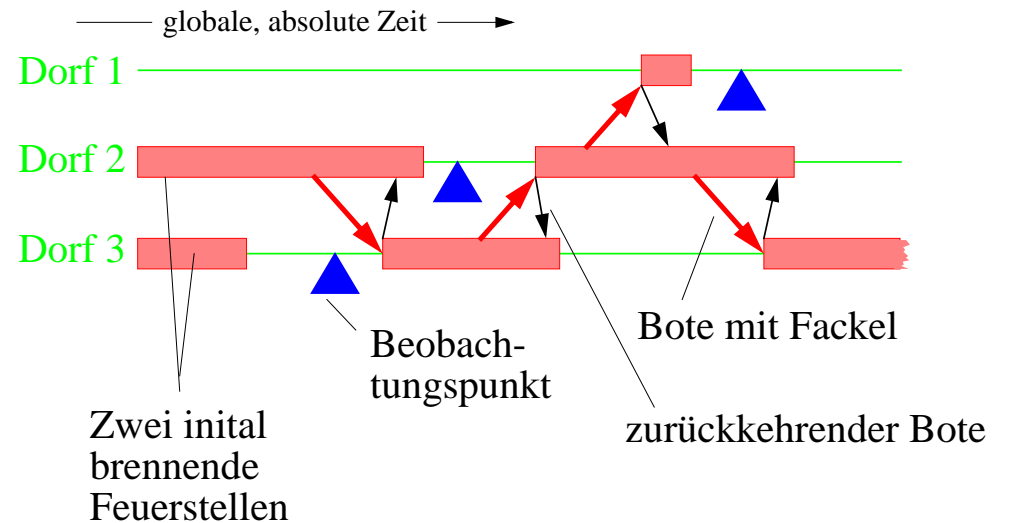




Falsche Beobachtungen



Beobachter kann **nicht** alles **gleichzeitig** betrachten!



Für alle Feuerstellen gilt (zu einem Zeitpunkt):

- kein Feuer brennt
- kein Bote ist unterwegs

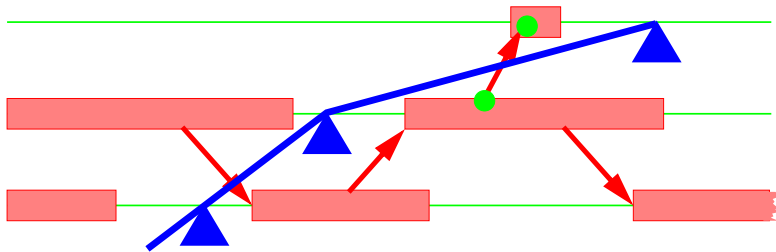
blaue Dreiecke ▲

Aber: Es gibt keinen *einzigsten Zeitpunkt*, wo kein Feuer brennt!

==> Beobachtung liefert ein *falsches* ("schiefes") Bild!

==> *Terminierungserkennungs-Problem*

Was läuft schief?



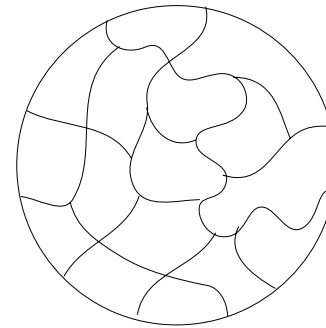
- Es scheint, als ob **alle Feuer aus sind** und **genauso-viele Boten angekommen wie aufgebrochen sind**
 - dann wäre tatsächlich "alles aus" (bis zum nächsten Blitzschlag)

- Täuschungsgrund:
Beobachtung ist **nicht kausal**treu

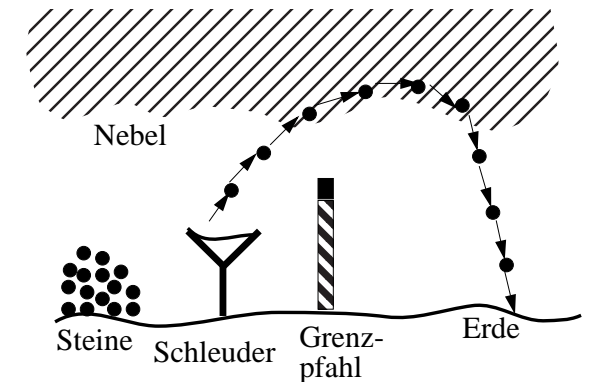


- **Wirkung** wird registriert, nicht aber die zugehörige **Ursache**
- **inkonsistente** Sicht durch schiefen Schnitt
- Würde man **alles gleichzeitig** beobachten, dann könnte man niemals eine **Wirkung** ohne ihre **Ursache** wahrnehmen!

Wie stellt man fest, dass der ewige Friede ausgebrochen ist?



Die Welt ist vollständig in einzelne Gebiete parzelliert



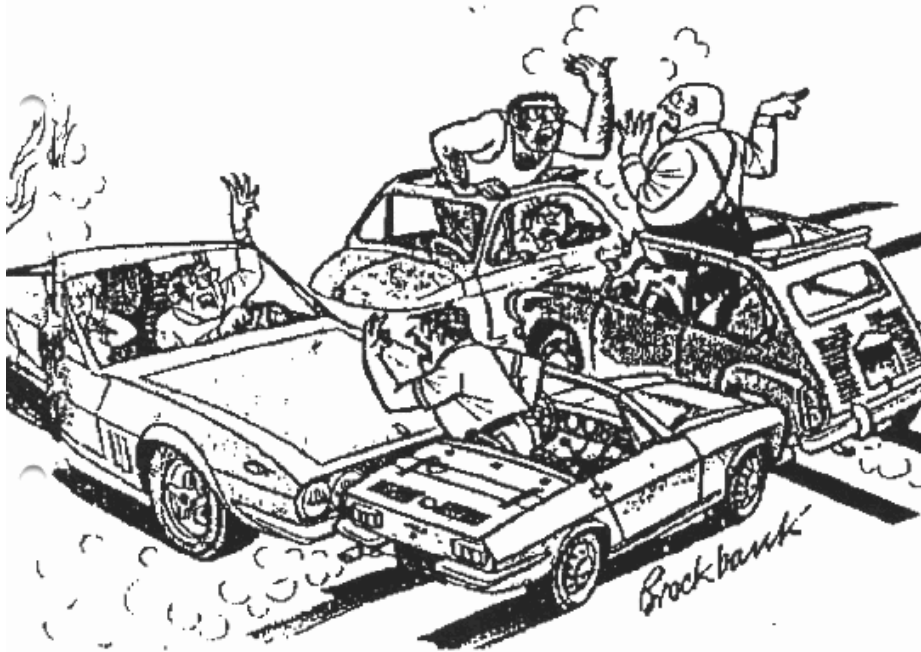
Regel: Nur wer von einem Stein getroffen wird, *darf* Steine wegschleudern

- Urknall: Ein einziger Meteorit traf die Erde...
- Steine fliegen beliebig hoch und brauchen beliebig lang...
- Niemand hat den Gesamtüberblick...

- Der Friede ist ewig ("stabil"; "monotone Eigenschaft")
- Wie stellt man sicher fest, *dass* er gilt (*wenn* er gilt)?

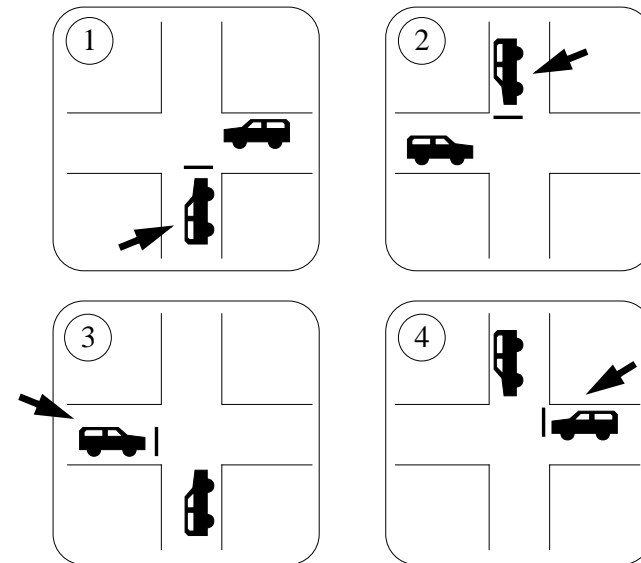
==> *Terminierungserkennungs-Problem*

Das Deadlock-Problem



„Neapolitanischer Hakenkreuzstau“
(Also sprach Bellavista, Luciano De Crescenzo)

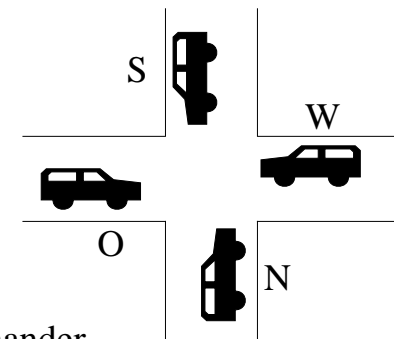
Phantom-Deadlocks



Vier Einzelbeobachtungen der Autos N, S, O, W

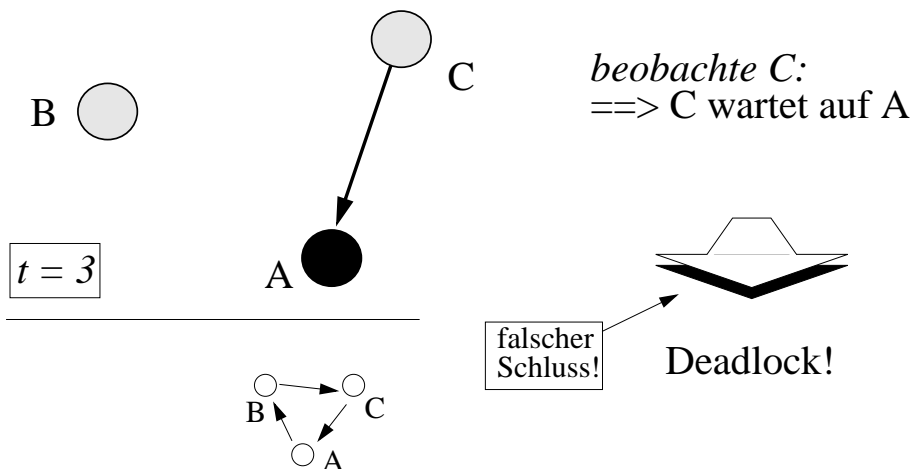
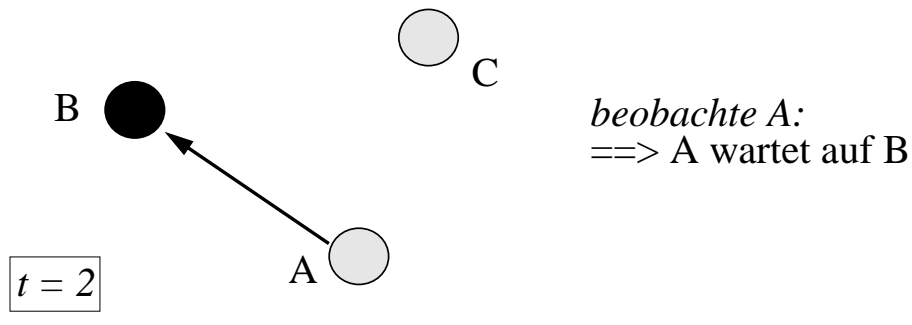
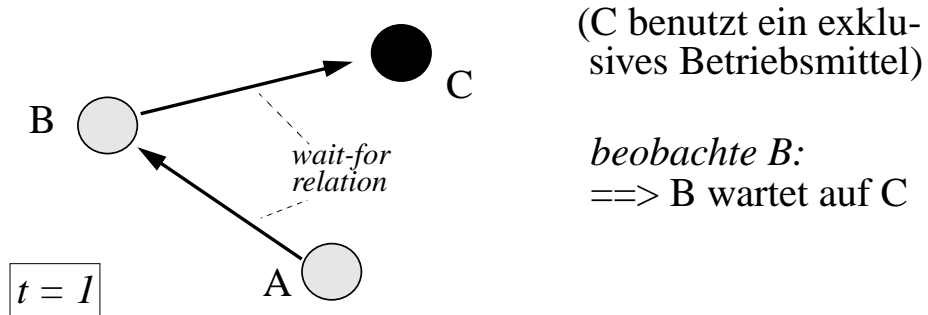
- 1) N wartet auf W
- 2) S wartet auf O
- 3) O wartet auf N
- 4) W wartet auf S

zu notwendigerweise
verschiedenen Zeitpunkten
liefert den *falschen* Eindruck,
als würden zu einem *einzigem*
Zeitpunkt alle zyklisch aufeinander
warten (--> Verklemmung)



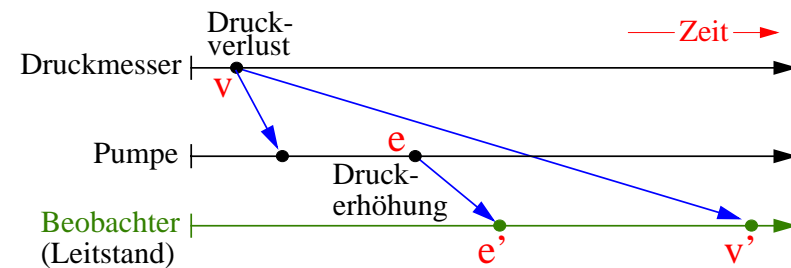
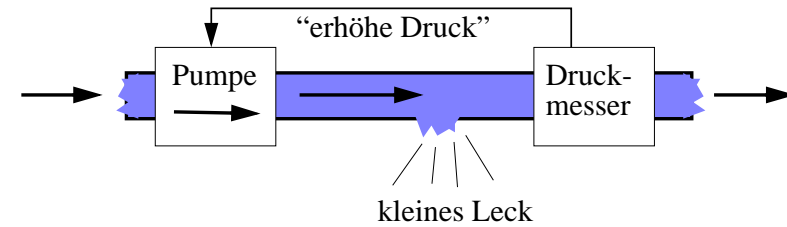
==> *verteiltes Deadlockproblem*

Phantom-Deadlocks



Kausal (in)konsistente Beobachtungen

- Gewünscht: Eine **Ursache** stets vor ihrer (u.U. indirekten) **Wirkung** beobachten



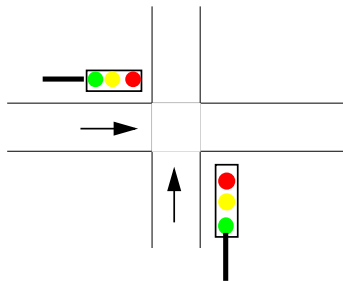
Falsche Schlussfolgerung des Beobachters:

Es erhöhte sich der Druck (aufgrund einer unbegründeten Aktivität der Pumpe), es kam zu einem Leck, was durch den abfallenden Druck angezeigt wird.

\implies *"Causal order-Problem"*

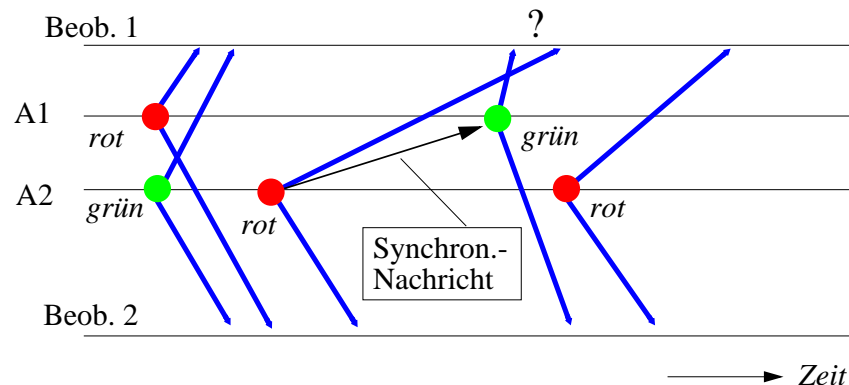
Beispiel: Verteilte Ampelsteuerung

(hier für 2 Ampeln)



Konsistenzproblem
bei *mehreren*
Beobachtern

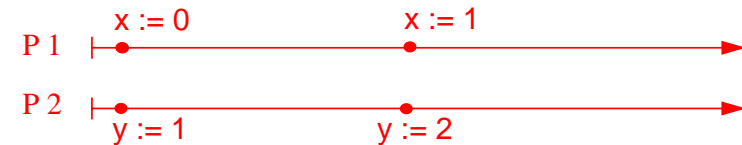
- Jede Ampel darf autonom auf **rot** schalten
- Eine Ampel darf nur dann auf **grün** schalten, wenn sie erfahren hat, dass die andere rot ist
- Umschalten der Ampel ist ein *Ereignis* ●
(*atomar*: zeitlos, nicht unterbrechbare Aktion)

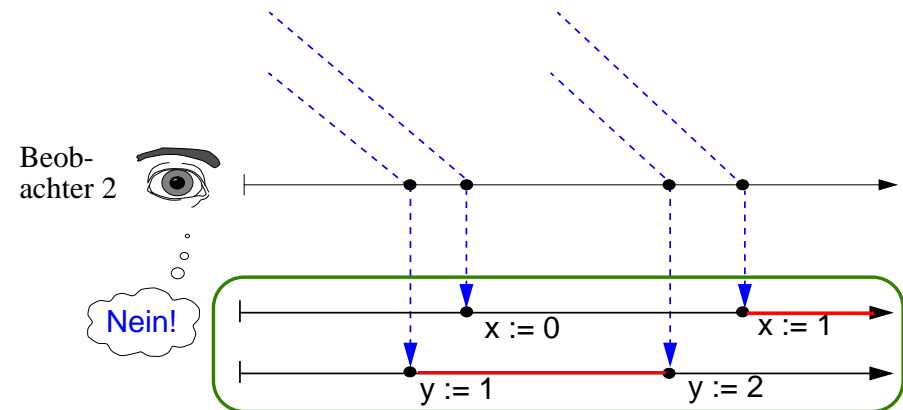
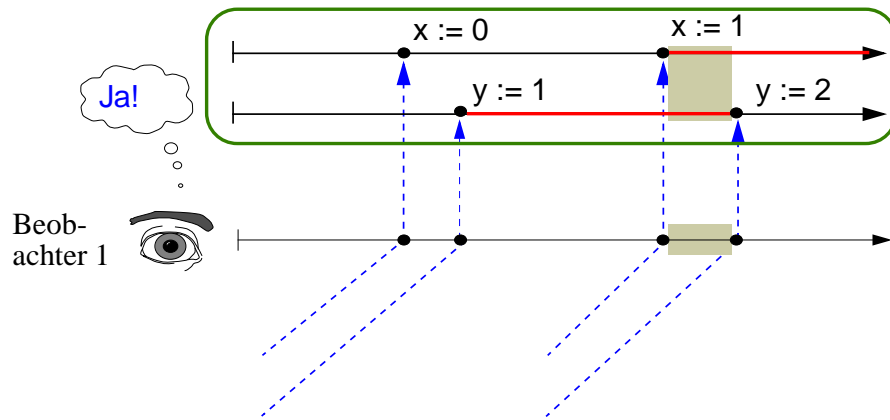


- **Welcher Beobachter hat Recht?** (Wieso?)
- Ähnliche Probleme (in komplexerem Umfeld) tauchen in der Praxis tatsächlich auf (z.B. Flugkontrollsystem)

Das Problem globaler Prädikate

Frage: Gilt in der vorliegenden Berechnung $x = y$?



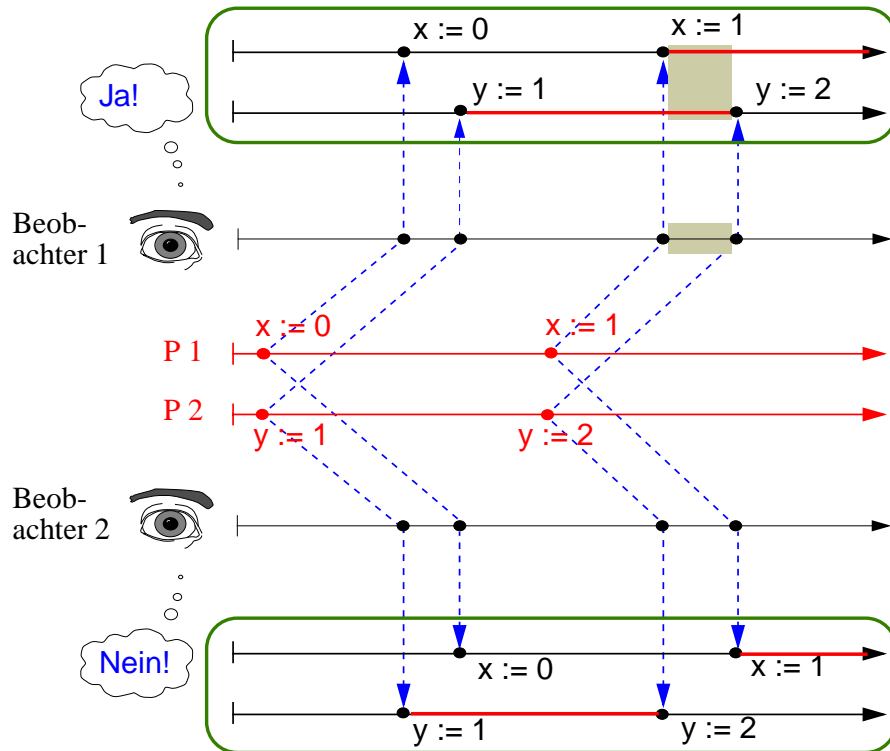


- Beide Beobachtungen sind gleich "richtig"
- Die Beobachter stimmen bzgl. $x = y$ nicht überein!

Aber was denn nun: *gilt $x=y$ in dieser Berechnung oder nicht?*

Das Problem globaler Prädikate

Frage: Gilt in der vorliegenden Berechnung $x = y$?

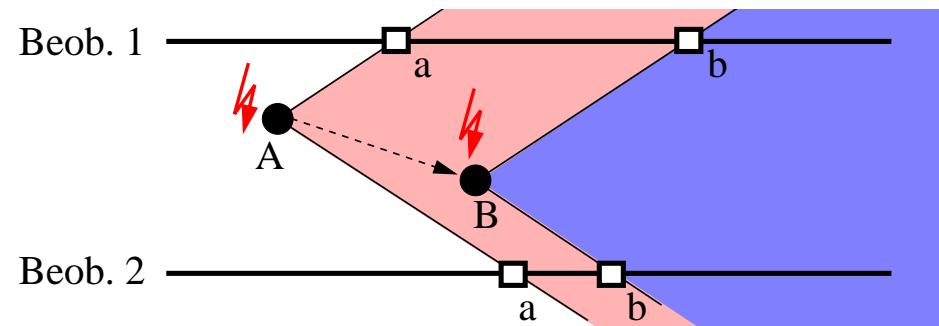
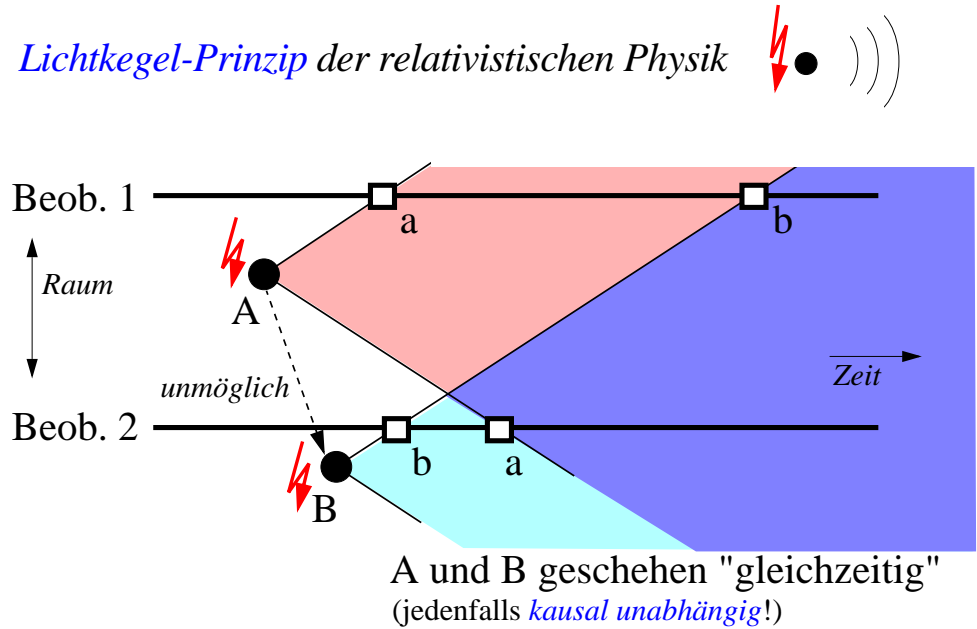


- Beide Beobachtungen sind gleich "richtig"
- Die Beobachter stimmen bzgl. $x = y$ nicht überein!

Aber was denn nun: *gilt $x=y$ in dieser Berechnung oder nicht?*

Relativierung der Gleichzeitigkeit

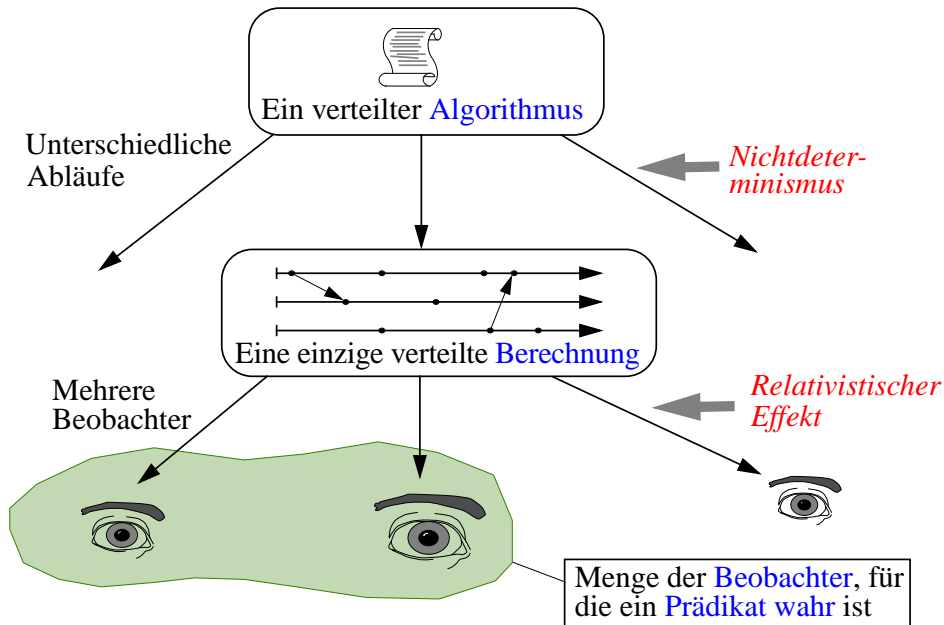
Zwei "kausal unabhängige" Ereigniss können in beliebiger Reihenfolge beobachtet werden!



beobachterinvariant
 \implies objektive Tatsache \implies Alle Beobachter sehen B nach A

Die "Beobachtungsvielfalt"

- Verschiedene Beobachter sehen *verschiedene Wirklichkeiten*



Konsequenz:

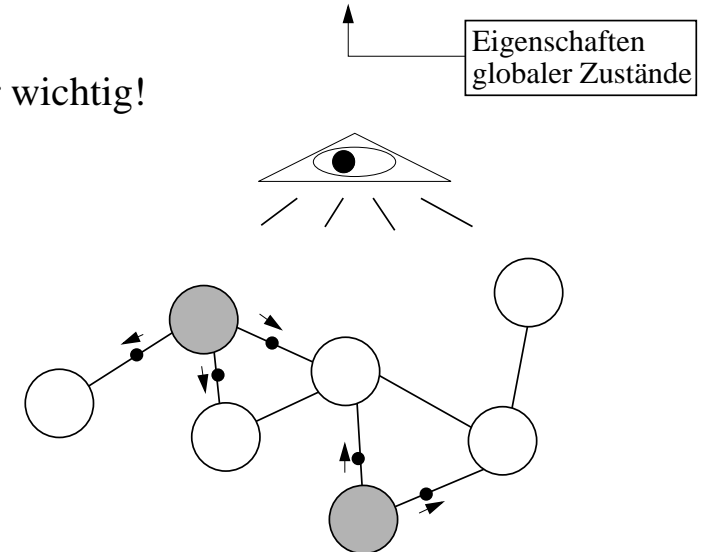
Es ist naiv (d.h. falsch!), einen **verteilten Debugger** zu entwickeln, mit dem man solche (im sequentiellen Fall "richtigen") Fragen beantworten kann!

Ursache:

Bei sequentiellen Berechnungen fallen **Berechnung** und **Beobachtung** zusammen, im verteilten Fall nicht!

Globale Prädikate...

... sind aber wichtig!



Bsp.:

- Wieviel Geld ist in Umlauf?
- Ist die parallele Approximation gut genug?
- Ist die verteilte Berechnung terminiert?
- Ist ein bestimmtes Objekt "Garbage"?
- Konsistenter Sicherungspunkt (vert. Datenbank)?
- Wie hoch ist die momentane Last?
- Liegt eine zyklische Wartebedingung vor?

Einige globale Prädikate lassen sich "eindeutig" bestimmen

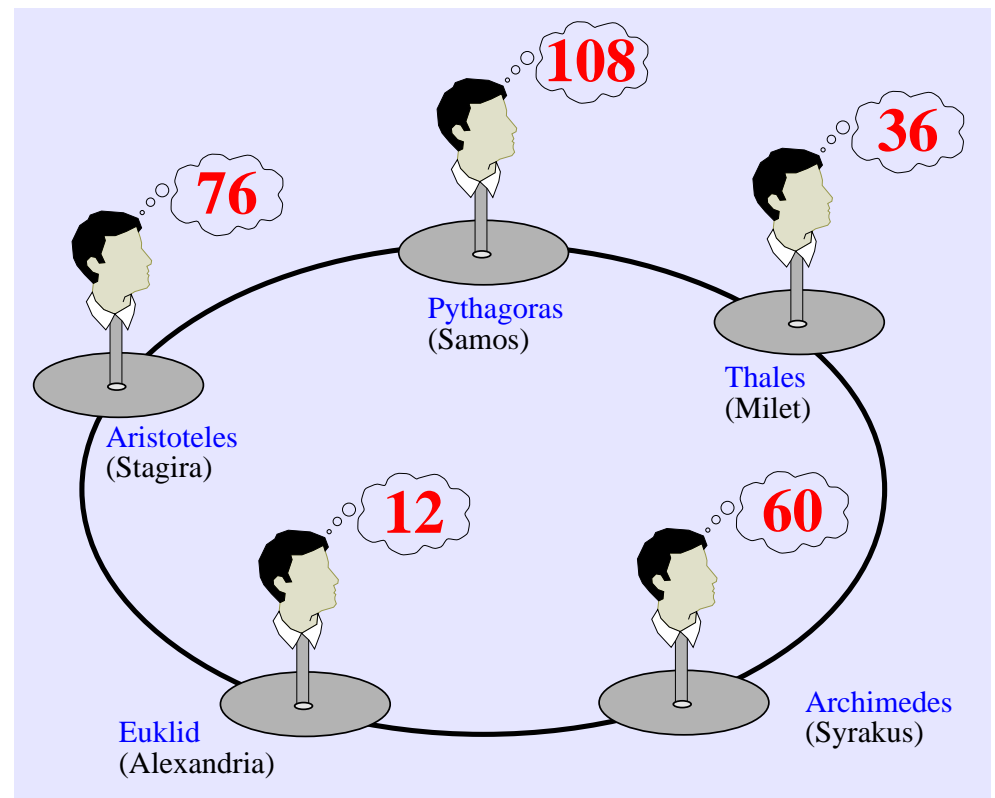
- Welche? --> fundamentale Aspekte analysieren
- Wie? --> *Algorithmen*

u.a. Thema der Vorlesung

Erste Beispiele für verteilte Algorithmen

Ein erster verteilter Algorithmus: Verteilte Berechnung des ggT

Fünf in Raum und Zeit verteilte griechische Philosophen wollen zusammen den **grössten gemeinsamen Teiler** ihres Alters berechnen...



- Was ist der **ggT** einer Menge $\{a_1, \dots, a_n\}$ nat. Zahlen?
- Wie funktioniert der *euklidische Algorithmus*?

“Elemente” Buch 7, Satz 1 und 2

Der grösste gemeinsame Teiler (ggT)

- Nachfolgend werden nur natürliche Zahlen betrachtet
- t heisst *Teiler* einer Zahl p , falls es $q \neq 0$ gibt mit $q \cdot t = p$
 - Bsp: 6 ist Teiler von 30 (da $5 \times 6 = 30$)
 - 3 ist Teiler von 30 (da $10 \times 3 = 30$)
 - aber: 8 ist kein Teiler von 30
- 1 ist Teiler jeder Zahl (klar nach Definition)

-
- t heisst *gemeinsamer Teiler* von u, v wenn t Teiler von u ist und t Teiler von v ist

- Bsp: 6 ist gemeinsamer Teiler von 30 und 18.
 - Aber 3, 2, 1 sind auch gemeinsamer Teiler.
 - Jedoch gibt es keine anderen gemeinsamer Teiler
 - \implies 6 ist der grösste gemeinsame Teiler (ggT).

- Zu je zwei Zahlen $\neq 0$ gibt es stets einen ggT, da die Menge der gemeinsamen Teiler mindestens die 1 enthält und endlich ist (was aber ist $\text{ggT}(x,0)$?)

- Kanonische Erweiterung auf $n \geq 1$ Argumente
 - Bsp: $\text{ggT}(108, 36, 60, 12, 76) = 4$

-
- *Anwendung*: Z.B. Kürzen von Brüchen

Satz von Euklid

Der ggT zweier positiver ganzer Zahlen x, y (mit $x \geq y > 0$) ist gleich dem ggT von y und dem Rest, der bei ganzzahliger Division von x durch y entsteht

- Offenbar ist $\text{ggT}(x, x) = x$ für alle x
- Man setzt nun noch $\text{ggT}(x, 0) = x$ für alle x

- Obiger Satz legt eine rekursive Realisierung nahe

$$\text{ggT}(x, y) \rightsquigarrow \text{ggT}(y, \text{mod}(x, y))$$

- für $x > y$ werden in der Rekursion beide Argumente jeweils kleiner
- Rekursion geht irgendwann mit $\text{ggT}(\dots, 0)$ zuende
- oft: iterative Formulierung (statt rekursiver)

\implies *Euklidischer Algorithmus*

Vorteil: Zur Bestimmung des ggT sind Primfaktorzerlegung ("Schulmethode") oder Probedivisionen nicht notwendig

Hinweis: Der Beweis des Algorithmus (und des Satzes) ist eine interessante Wiederholungsübung!

Das (abstrakte) Lösungsprinzip

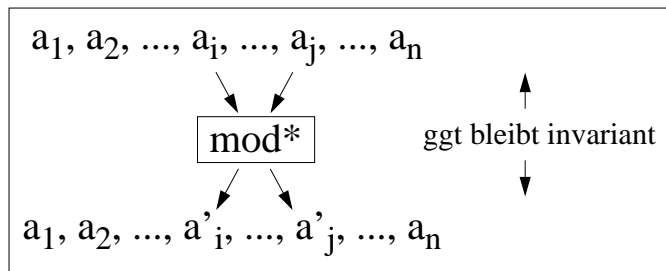
- Invariante (für $x \geq y > 0$):

$$\text{ggT}(x,y) = \text{ggT}(y, \text{mod}^*(x,y))$$

Wie Restfunktion mod , soll jedoch y liefern, falls x ganzzahliges Vielfaches von y ist.
Also: $\text{mod}^*(a,b) = \text{mod}(a-1,b)+1$

- Idee: Ersetze x durch $\text{mod}^*(x,y)$ ← Ist i.a. kleiner als x

- Erweiterung auf n Zahlen:



Frage: Wieso wurde mod zu mod^* modifiziert?

Greife zwei beliebige Elemente a_i, a_j ($i \neq j$) heraus und ersetze den grösseren Wert durch $\text{mod}^*(a_i, a_j)$

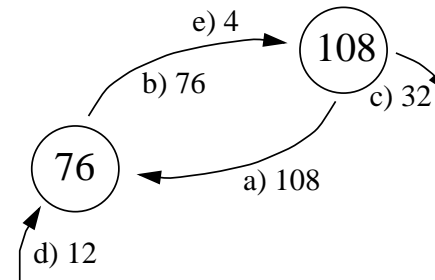
\implies konvergiert gegen den ggT

- Dies nun "verteilt" realisieren!

In "Teamarbeit", damit es schneller geht...

Nachrichtenaustausch

Prinzip: Kooperation durch Kommunikation



zwischen den Nachbarn auf dem Ring

- Nachrichten sind beliebig lange unterwegs

- Parallele Aktivitäten

Idee:

- Initial beide Nachbarn spontan informieren
- Danach nur auf eintreffende Nachrichten reagieren
- Neue Erkenntnisse an alle Nachbarn übermitteln

Verhaltensbeschreibung eines Prozesses P_i :

```
{ Eine Nachricht <y> ist eingetroffen }
if  $y < M_i$  then
     $M_i := \text{mod}(M_i-1, y) + 1;$ 
    send < $M_i$ > to all neighbours;
fi
```

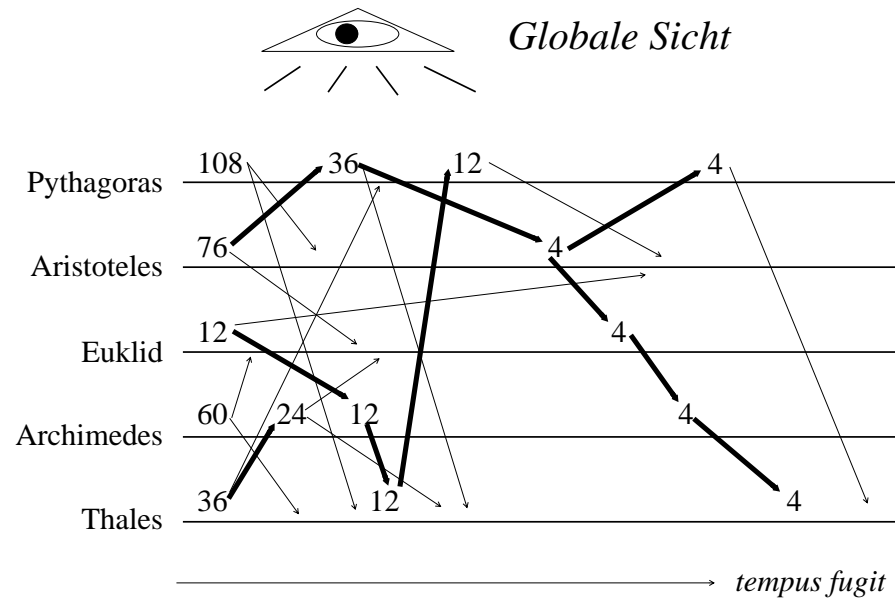
Jeder Prozess P_i hat seine eigene Variable M_i .

Jeder Prozess hat das gleiche prinzipielle Verhalten.

Atomare Aktion:

In der Zwischenzeit u.U. eintreffende Nachrichten werden im "Hausbriefkasten" zwischengepuffert...

Berechnungsablauf und Zeitdiagramm



Ablauf einer *möglichen* "verteilten Berechnung" mit einem *Zeitdiagramm*.

Nicht-deterministisch, abhängig von der Nachrichtenlaufzeit!

- *Terminiert* wenn (?):

Das ist unrealistisch!

(a) jeder Prozess den ggT kennt,

(b) alle den gleichen Wert haben,

Beweisbare math. Eigenschaft

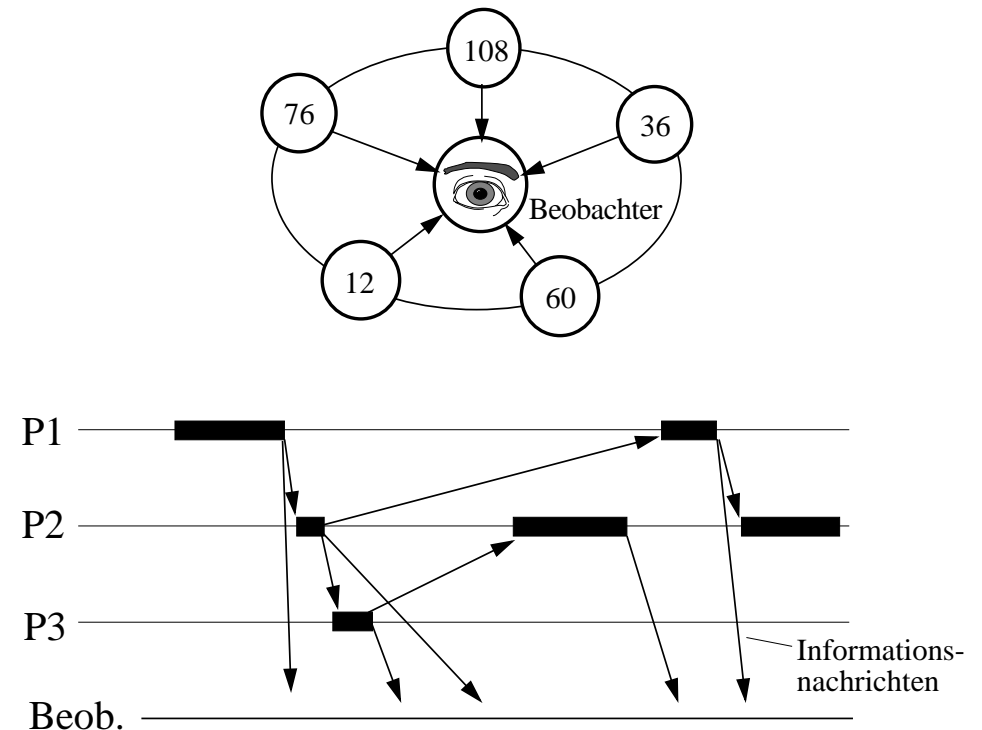
(c) alle Prozesse passiv sind und keine Nachricht unterwegs ist.

Diese "stabile" Stagnationseigenschaft ist *problemunabhängig*!

- Was ist adäquat?

- Wie feststellen?

Globaler ggT-Beobachter?



Bekommt der Beobachter ein "richtiges Bild" des Geschehens?

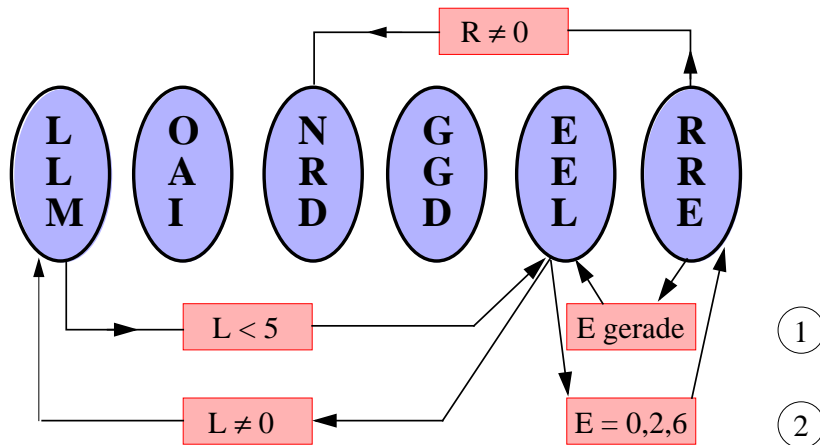
- was heisst das genau?
- und wenn Informationsnachrichten verschieden schnell sind?
- könnte der Beobachter einen Zwischenzustand (z.B. "alle haben den Wert 12") irrtümlich als Endzustand interpretieren?
- wie stellt er überhaupt das Ende der Berechnung fest?

Ein weiteres Beispielproblem: Paralleles Lösen von "Zahlenrätseln"

LONGER	207563
+ LARGER	+283563
MIDDLE	491126

Pro Spalte
ein Prozess

- Reaktives Verhalten: Auslösen atomarer Aktionen
- Propagieren neuer Erkenntnisse (= Einschränkungen) ("parallele Constraint-Propagation")

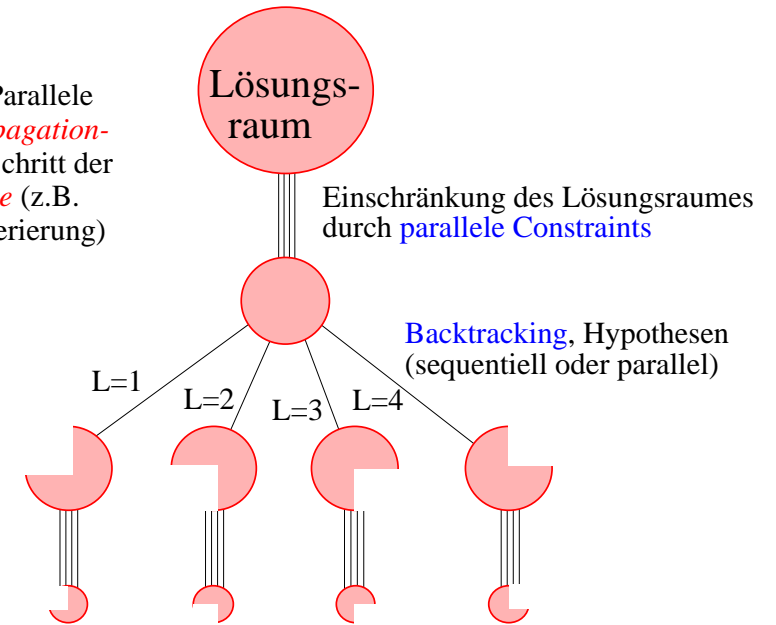


Endergebnis:
 $1 \leq L \leq 4$
 $M \geq 2$
 $R = 1,3,5,6,8$
 $E = 0,2,6$
 sonst keine Einschränkung

Probleme:
 1) Keine eindeutige Lösung
 --> Backtrack-Algorithmus
 2) Entdeckung der "Stagnation"?
 (Ende der Parallelphase)

Der aufgesetzte Backtrack-Algorithmus

Abwechselnd: Parallele
Constraint-Propagation-Phase
und ein Schritt der
Backtrack-Phase (z.B.
Hypothesengenerierung)



- **Hypothese** = beliebige Menge von Constraints (von einem "Orakel" statt von einer Spalte)
- Einheit, die die Hypothesen generiert und verwaltet, muss die **Terminierung** einer Constraint-Phase feststellen können
 - wie würde man hier die Terminierung zweckmässigerweise definieren?
 - problembezogen wie bei ggT ("alle Werte identisch") hier nicht einfach
 - "alle passiv und keine sinnvolle Nachricht mehr unterwegs"?

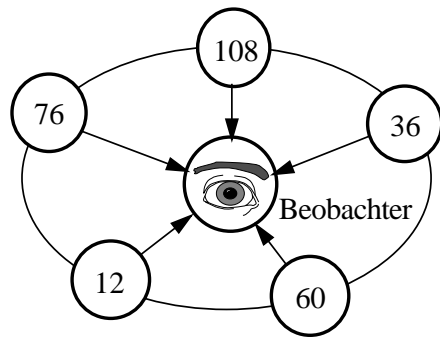
Bemerkungen:

- nicht unbedingt beste Parallelisierungsstrategie!
- Problem ist (in Verallgemeinerung) NP-vollständig

Offensichtlich *gleiches Schema* wie ggT-Berechnung!

Übungen (1) zur Vorlesung "Verteilte Algorithmen"...

- a) Man zeichne Raum-Zeit-Diagramme für verschiedene Abläufe des verteilten ggT-Algorithmus
- b) Wie kann man beweisen, dass für *jeden* denkbaren Ablauf das Endergebnis stets der ggT ist?



- c) Bleibt der Algorithmus (und/oder der Beweis) korrekt, wenn im Algorithmus $y < M_i$ durch $y \leq M_i$ ersetzt wird?

Verhaltensbeschreibung eines Prozesses P_i :

```

{Eine Nachricht  $\langle y \rangle$  ist eingetroffen}
if  $y < M_i$  then
     $M_i := \text{mod}(M_i - 1, y) + 1$ ;
    send  $\langle M_i \rangle$  to all neighbours;
fi
    
```

...Übungen (1)

- d) Man vergleiche die verteilte Berechnung des ggT-Algorithmus für zwei Zahlen mit dem üblichen sequentiellen ggT-Algorithmus für zwei Zahlen
- e) Genügt es auch, nur in Uhrzeigerrichtung eine Nachricht zu senden (anstatt an beide Nachbarn)?
- f) Kann statt des Ringes eine andere Topologie verwendet werden? Welche?
- g) Formalisieren Sie für Zeitdiagramme den Begriff (potentiell, indirekt) "kausal abhängig" als Halbordnung über "Ereignissen"
- h) Wie kann man erreichen, dass ein ggT-Beobachter (der über jede Wertänderung eines Prozesse informiert wird) eine "kausaltrue" Beobachtung macht?
- i) Beobachtungen sind eine lineare Ordnung von (beobachteten) Ereignissen. In welcher Beziehung steht die oben erwähnte Halbordnung zu dieser linearen Ordnung? Können Sie eine Vermutung darüber anstellen, was der Schnitt aller möglichen kausaltrauen Beobachtungen einer verteilten Berechnung aussagt?
- j) Wie kann der Beobachter die Terminierung erkennen?