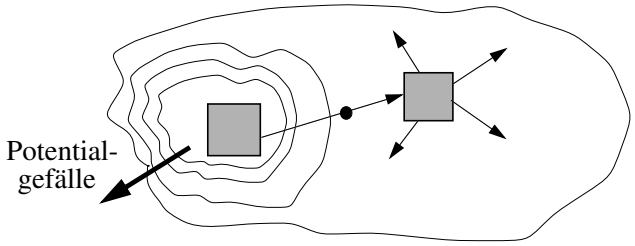
# Das Paradigma der vert. Approximation

*Prinzip:*
- Anfang: Informiere alle Nachbarn spontan
- Bei Empfang einer Nachricht:
  - berechne neue Approximation
  - falls diese "besser": informiere Nachbarn




- *Nachrichtengesteuert* (aber "Spontanstart")
- Alle Prozesse arbeiten gleich, alle sind beteiligt
- *Nichtdeterministischer* Ablauf, determin. Ergebnis
- Beliebige stark zusammenhängende Topologie
- Assoziative Operatoren (min, max, $\cap$, $\cup$, +, and, or, ...)
- *Stagnation* bei globalem Gleichgewicht ("Optimum")
  --> Potentialunterschiede ausgeglichen
  --> Terminierungsproblem

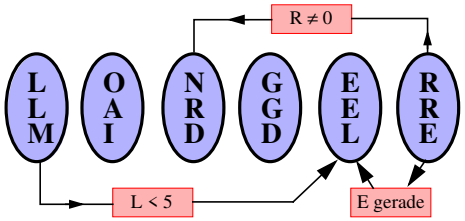
*Beispiele* ("Instanzen der Algorithmenklasse"):
- ggT
- Zahlenrätsel
- Verteilen von Information ("Wissensausgleich")
- Routingmatrizen (inkl. Spannbaum)
- Maximale Identität ("election") } (noch) nicht behandelt
- Lastausgleich (Approx. eines dyn. Optimums) } (noch) nicht behandelt
- Relaxationsverfahren (Lösen von DGL) } (noch) nicht behandelt



# Verteilte Terminierung

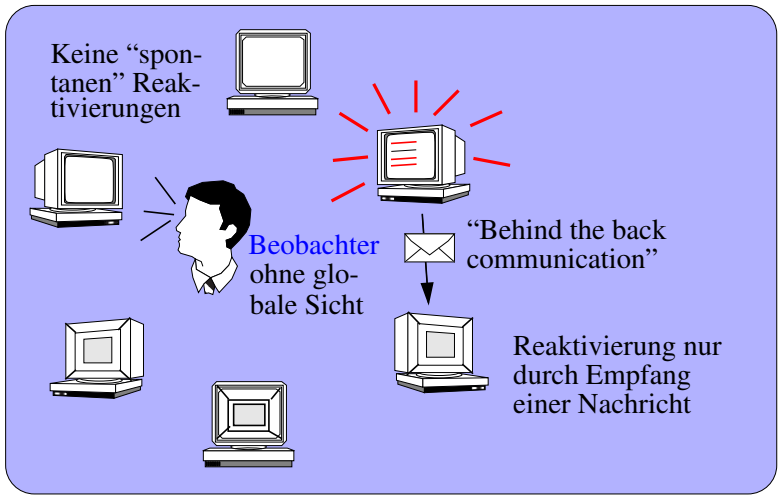
# Das Problem der Terminierung

- Bsp: Zahlenrätsel (oder ggT) auf einem PC-Cluster




- pro Spalte (bzw. "Philosoph") jeweils ein Display
- dort jeweiligen Zustand und neue Wertemengen anzeigen

*aktiv* oder *passiv*



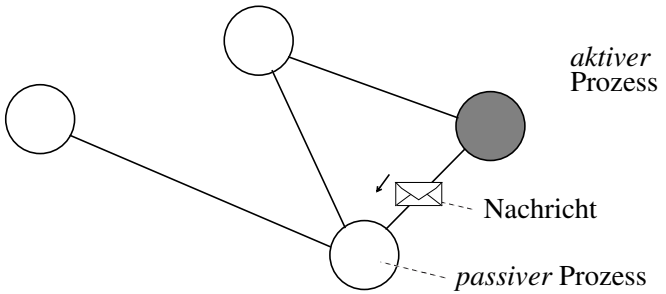

*Erkennung* der verteilten Terminierung?

alle passiv und keine Nachricht unterwegs



# Terminierungserkennung

- Dem Zahlenrätsel- und dem ggT-Beispiel gemeinsam ist ein etwas abstrakteres Modell:

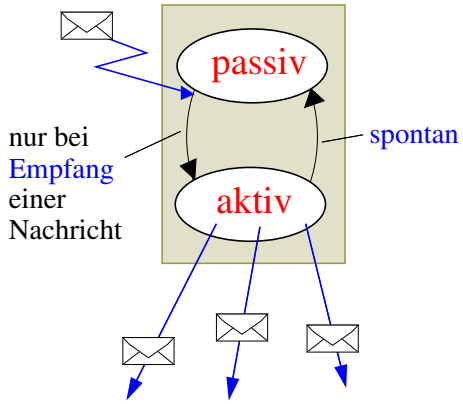



- Es gibt aktive und passive Prozesse sowie Nachrichten
  - dabei kann eine ankommende Nachricht einen passiven Prozess wieder reaktivieren
- Man möchte "irgendwie" erkennen, ob alle Prozesse passiv sind und keine Nachricht mehr unterwegs ist
  - Stagnationszustand beim verteilten Approximationsparadigma

# Verteilte Terminierung: Modell und Problemdefinition

Nachrichtengesteuertes Modell einer vert. Berechnung:

- Prozesse sind *aktiv* oder *passiv*
- Nur aktive Prozesse versenden Nachrichten
- Prozess kann "spontan" passiv werden
- Prozess wird durch ankommende Nachricht reaktiviert




Problem:

Feststellen, ob (zu *einem* Zeitpunkt)
- alle Prozesse passiv sind
- keine Nachricht unterwegs ist

- "globales Prädikat"
- "stabiler Zustand"



# Die Aktionen der Basisberechnung im nachrichtengesteuerten Modell

"guard": Prädikat über dem lokalen Zustand

Prozess p:

$S_p$: {Zustand = *aktiv*}
**send** message <M> **to** ...

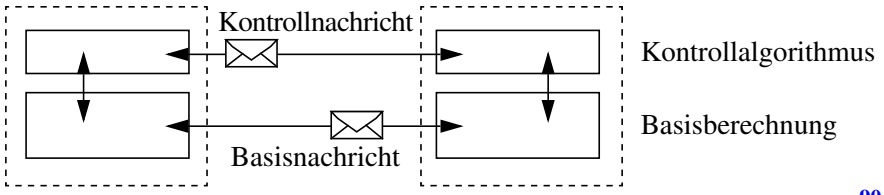
$R_p$: {Eine Nachricht ist angekommen}
**receive** <M> ; Zustand := *aktiv*

$I_p$: {Zustand = aktiv}
Zustand := *passiv*

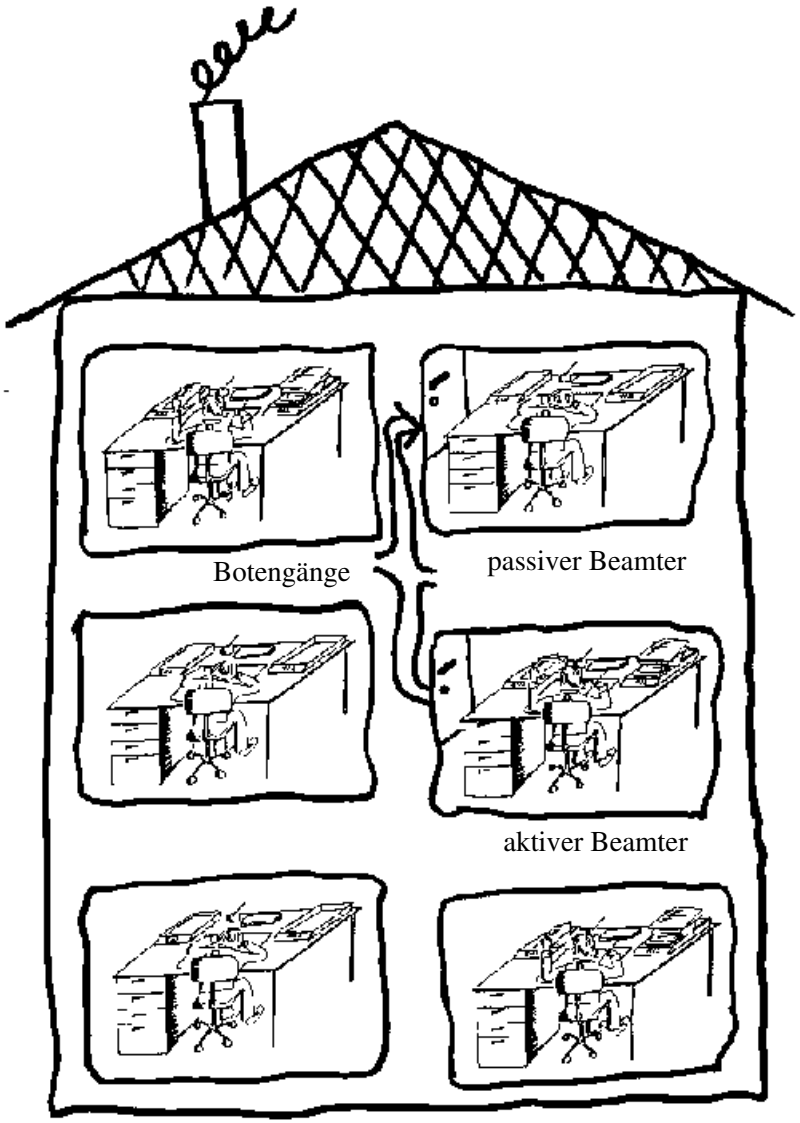
Typischerweise werden die Aktionen als "atomar" betrachtet

Abstraktes Verhalten einer verteilten Berechnung hinsichtlich Terminierung (ggf. existieren weitere Aktionen)

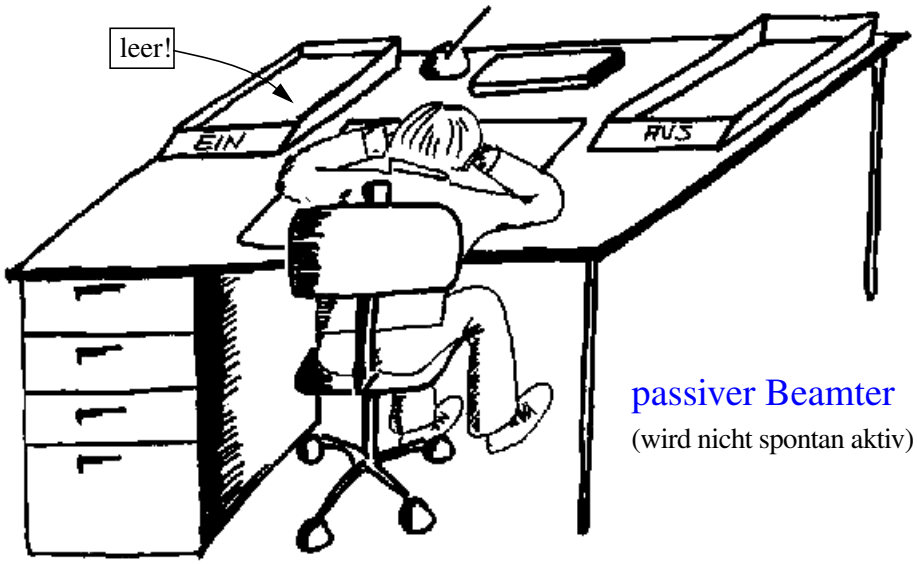
- Durch einen "überlagerten" Kontrollalgorithmus werden weitere Aktionen hinzugefügt
- "Anreichern" der Semantik der Basisberechnung für Zwecke des Kontrollalgorithmus
  - z.B. Verändern spezifischer (lokaler) Variablen
- Überlagerter Algorithmus soll Basisberechnung nicht stören
  - darf aber die Variablen, die der lokalen Kommunikation mit dem Basisalgorithmus dienen, lesen und schreiben



# Eine typische Behörde



aktiver Beamter




passiver Beamter
(wird nicht spontan aktiv)

# Die Funktionsweise der Behörde

(1) Publikumsverkehr nur bis 12 Uhr
(2) Schliesst erst, wenn alle Vorgänge bearbeitet
(3) Vorgänge werden von Beamten erledigt
(4) Die Bearbeitung eines Vorganges *kann* neue Vorgänge für andere Beamte auslösen
(5) Aktenaustausch per (bel. langsame) Boten
(6) Keiner hat den Gesamtüberblick
(7) Beamte sind *aktiv* oder *passiv*
(8) Ein Beamter wird nicht spontan aktiv

*Terminiert*, wenn alle passiv und nichts "unterwegs"

Das ist ein stabiler Zustand!

Variante: Beamter lässt sich während der Arbeit nicht stören (anklopfen/warten auf "herein") -->

! → Beamte *scheinen* immer passiv (Atommodell)

---

*Speedup* ist durch die Maximalzahl gleichzeitig aktiver Beamten begrenzt

Dieser ist oft erstaunlich niedrig...

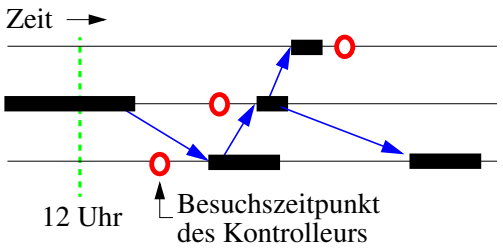
# Das schiefe Bild des Kontrolleurs

- *Kontrolleur* wandert durch die Behörde, um die Terminierung feststellen zu können

- *Problem*: Wie stellt der Kontrolleur fest, ob der stabile Terminierungszustand eingetreten ist?

---

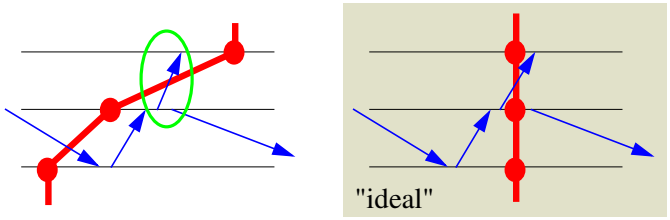
- Die *Illusion* Kontrolleurs:




"behind the back communication"

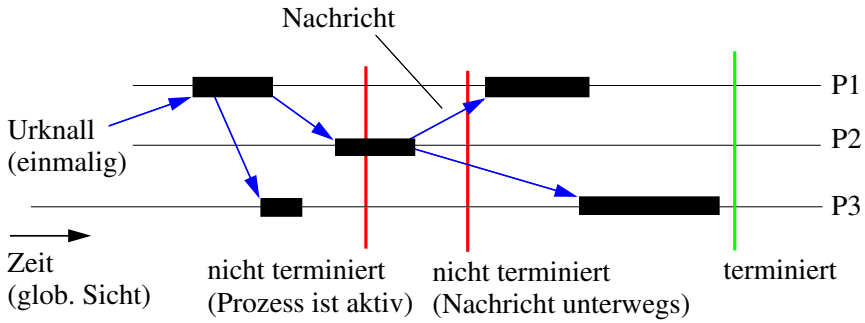
- Alle Beamten stets passiv
- $\Sigma$ Nachrichten versendet = $\Sigma$ Nachrichten empfangen

- Kontrolleur macht sich ein *schiefes Bild!*



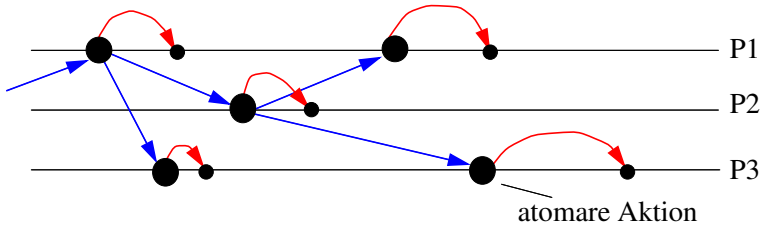



# Zeitdiagramme und Atommodell




*Idee:* Dauer der Aktivitätsphasen "gegen Null" gehen lassen

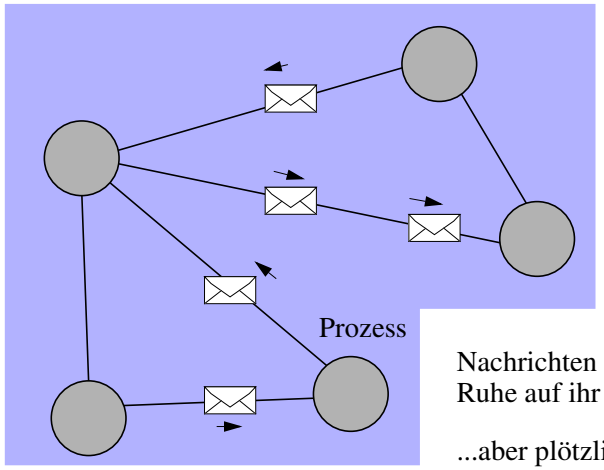
Modellierung: Prozess sendet (virtuelle) Nachricht an sich selbst, sobald er aktiv wird; ist "unterwegs", solange er aktiv ist




Terminiert (Atommodell) <==>
Keine (echte oder virtuelle) Nachricht unterwegs

Zur Lösung des Terminierungsproblems also feststellen, *ob* noch Nachrichten unterwegs sind

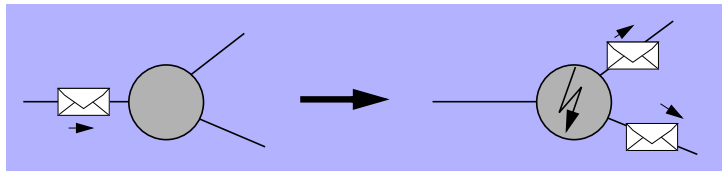
# Globale Sicht ’atomarer’ Berechnungen




idealisierter Beobachter

Nachrichten fliessen in aller Ruhe auf ihr Ziel zu...

...aber plötzlich "explodiert" ein Prozess, wenn er von einer Nachricht getroffen wird!



Terminiert, wenn in der globalen Sicht kein ✉ existiert

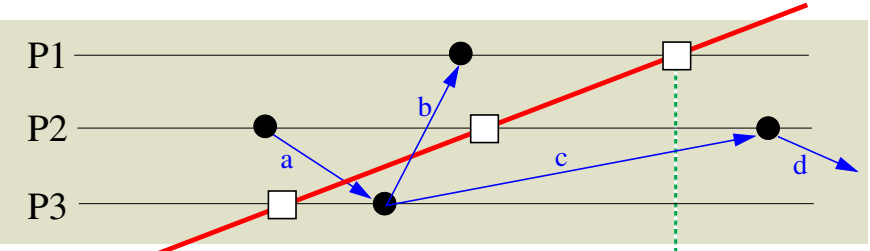
---

- Statt im "passiv/aktiv-Modell" genügt es offenbar, im Atommodell die Terminierungserkennung zu lösen (wieso?)
- Wie sehen die Aktionen der Basisberechnung in diesem Modell aus?

# Verteilte Terminierung: Lösungen durch Zählen von Nachrichten?

- Genügt das (verteilte) Zählen von gesendeten und empfangenen Nachrichten?

- Einfaches Zählen genügt nicht, Gegenbeispiel:




Schiefer Zeitschnitt
--> "Illusion"

Man erwischt nicht alle Prozesse gleichzeitig

Implementiert durch eine "Besuchswelle"

Insgesamt:

1 Nachricht gesendet,
1 Nachricht empfangen.

*Aber*: nicht terminiert!

*Ursache* (informell):

- Nachricht aus der "Zukunft"
  - kompensiert die Zähler
- Inkonsistenter Schnitt
  - ist nicht äquivalent zu einem senkrechten Schnitt

Lösung durch "Ursachenvermeidung"? Ideen vielleicht:

- Nachrichten aus der Zukunft *vermeiden* oder zumindest *erkennen*?
- Senkrechten Schnitt simulieren durch *Einfrieren* der Prozesse?