Verteilte Systeme

Globale Systemzustände

18. November 2005

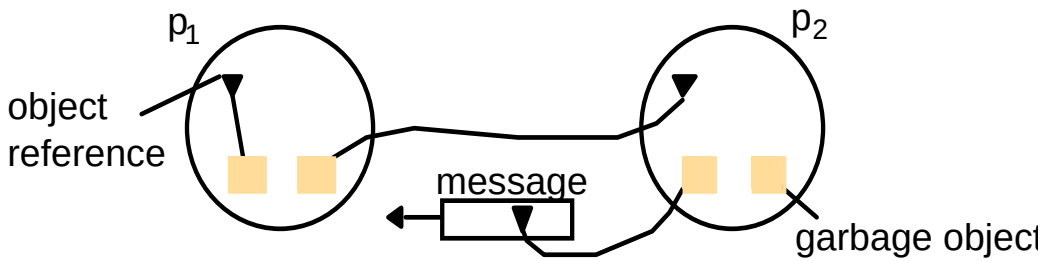
- Globale Zustände und deren Anwendung
- Distributed Snapshot
- Der Begriff des Cut
- Der Algorithmus von Lamport und Chandy
- Beispiel
- Zusammenfassung

Globale Systemzustände

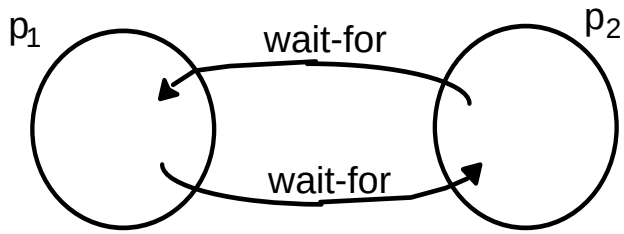
- Es gibt eine Reihe von Gelegenheiten, bei denen man gern über den Gesamtzustand des verteilten Systems Bescheid wüsste
- Der Gesamtzustand des Systems besteht aus
  - Den lokalen Zuständen der Einzelkomponenten (der Prozesse) und
  - Allen Nachrichten, die sich zur Zeit in der Übertragung befinden.
- Diesen Zustand exakt zur selben Zeit bestimmen zu können ist so unmöglich wie die exakte Synchronisation von Uhren – es lässt sich kein globaler Zeitpunkt festlegen, an dem alle Prozesse ihre Zustände festhalten sollen.

Anwendungen des globalen Zustands

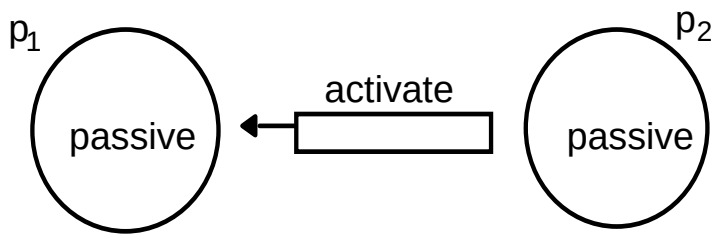
a. Garbage collection



b. Deadlock



c. Termination

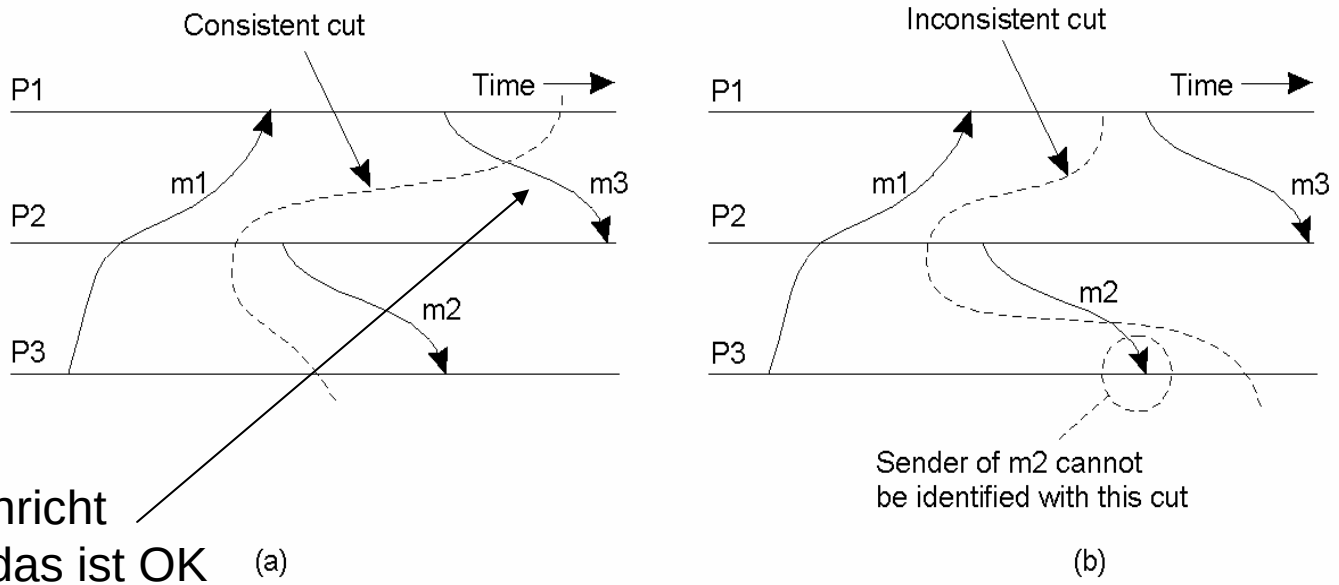


Distributed Snapshot

- Wie kann man nun den globalen Zustand eines verteilten Systems ermitteln?
- Lösung von Chandy und Lamport (1985): Distributed Snapshot:
  - ermittle einen Zustand, in dem das System möglicherweise war,
  - der aber auf jeden Fall konsistent ist
- Konsistenz bedeutet insbesondere: wenn festgehalten wurde, dass Prozess P eine Nachricht m von einem Prozess Q empfangen hat, dann muss auch festgehalten sein, dass Q diese Nachricht geschickt hat. Sonst kann das System nicht in diesem Zustand gewesen sein.

Consistent und Inconsistent Cut

- Definition der Konsistenz über den sog. „cut“, der für jeden Prozess das letzte aufgezeichnete Ereignis angibt.



Formale Definition des Cut

- Gegeben sei ein System $\wp$ von N Prozessen p_i (i=1,...,N).
- Betrachtet man nun den globalen Zustand $S=(s_1,...,s_N)$ des Systems, dann ist die Frage, welche globalen Zustände möglich sind.
- In jedem Prozess findet eine Serie von Ereignissen statt, womit jeder Prozess mittels der Geschichte seiner Ereignisse charakterisiert werden kann:
  $history(p_i) = h_i = <e_i^0, e_i^1, e_i^2, ...>$
- Jeder endliche Präfix der Geschichte eines Prozesses wird bezeichnet mit
  $h_i^k = <e_i^0, e_i^1, e_i^2, ..., e_i^k>$

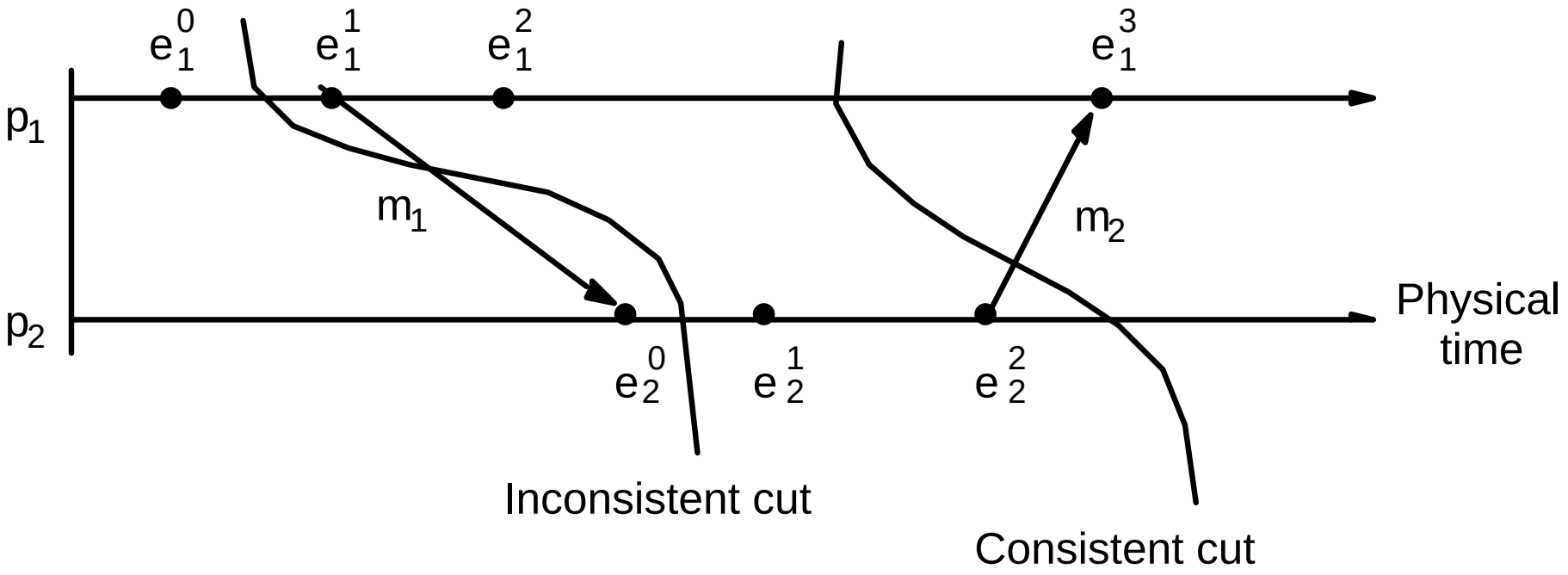
Formale Definition des Cut

- Ein Cut ist damit definiert wie folgt

$$C = h_1^{c_1} \cup h_2^{c_2} \cup ... \cup h_N^{c_N}$$

- Der Zustand s_i aus dem globalen Zustand ist dann genau derjenige von p_i, der durch das Ausführen des letzten Ereignisses im Cut erreicht wird, also von $e_i^{c_i}$
- Die Menge $\{e_i^{c_i} : i = 1,2,...N\}$ wird als Frontier des Cuts bezeichnet.

Beispiel



Frontier: $< e_1^0, e_2^0 >$ $< e_1^2, e_2^2 >$

Definition des konsistenten Cut

- Ein Cut ist dann konsistent, wenn er für jedes Ereignis, das er enthält, auch alle Ereignisse enthält, die zu diesem Ereignis in der Happened-Before-Relation (s. Zeit in verteilten Systemen) stehen:

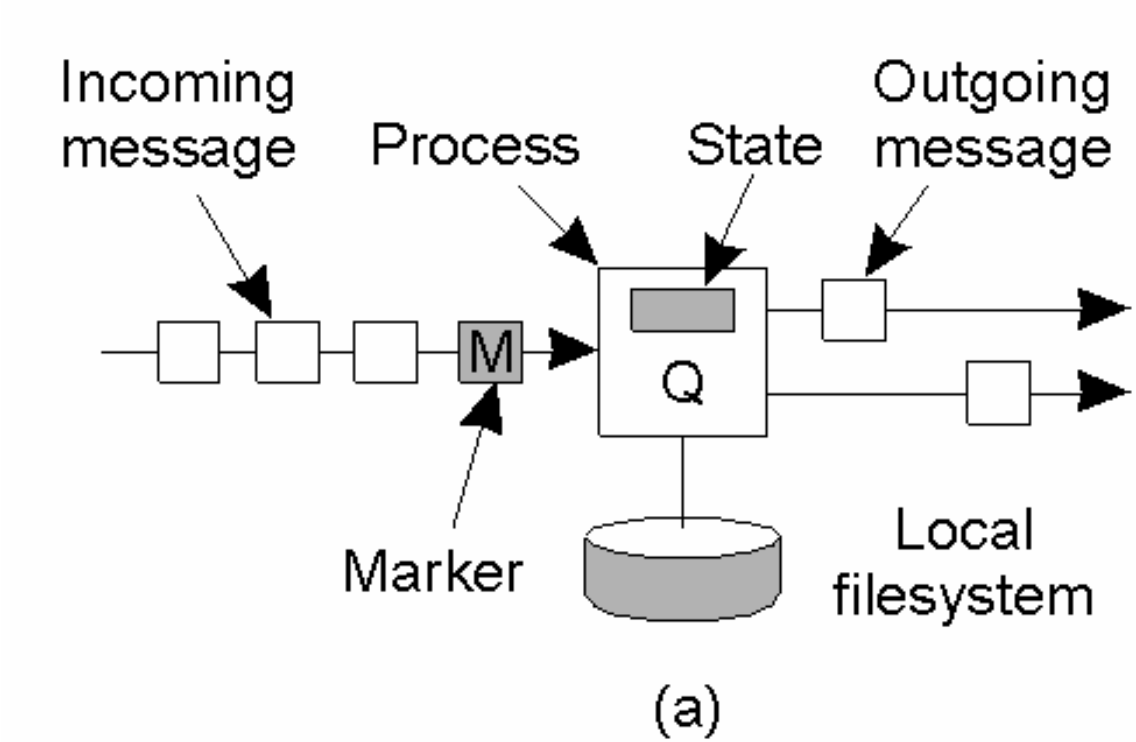
$$\forall e \in C, f \rightarrow e \Rightarrow f \in C$$

- Ein globaler Zustand ist konsistent, wenn er mit einem konsistenten Cut korrespondiert.

Lamport/Chandy-Algorithmus

- Prozesse sind mittels Punkt-zu-Punkt-Kanälen verbunden.
- Ein oder mehrere Prozesse starten den Algorithmus zur Feststellung eines Systemzustands, so dass gleichzeitig immer mehrere Snapshots erstellt werden können.
- Das System läuft unterdessen ungehindert weiter.
- Die Prozesse verständigen sich über Markierungsnachrichten über die Notwendigkeit der Speicherung eines Systemzustands.

Prozessmodell für den Algorithmus



Der Algorithmus

Marker receiving rule for process p_i

On p_i's receipt of a *marker* message over channel *c*:

if (p_i has not yet recorded its state) it

records its process state now;

records the state of *c* as the empty set;

turns on recording of messages arriving over other incoming channels;

else

p_i records the state of *c* as the set of messages it has received over *c* since it saved its state.

end if

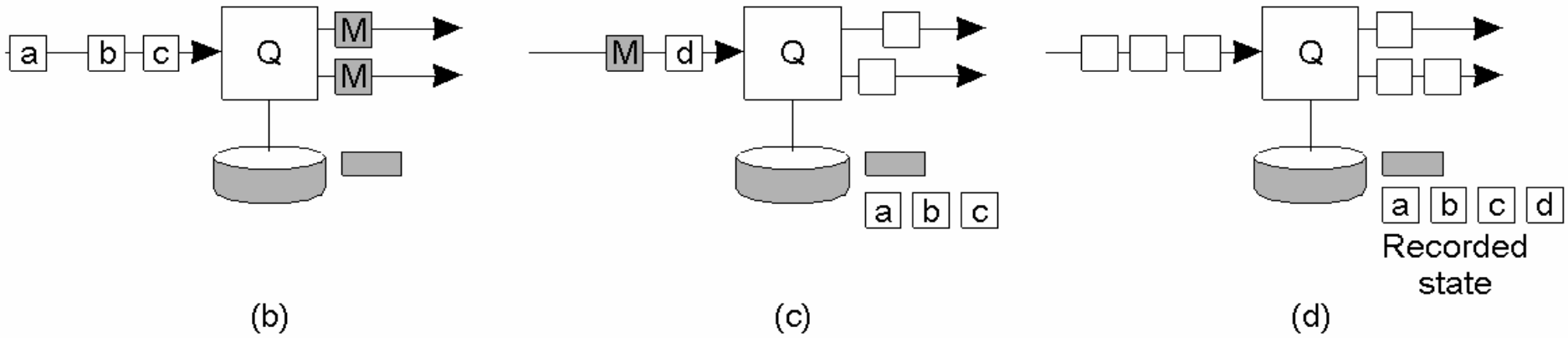
Marker sending rule for process p_i

After p_i has recorded its state, for each outgoing channel *c*:

p_i sends one marker message over *c*

(before it sends any other message over *c*).

Ablauf des Algorithmus



b) Prozess Q erhält zum ersten Mal einen Marker und hält seinen lokalen Zustand fest

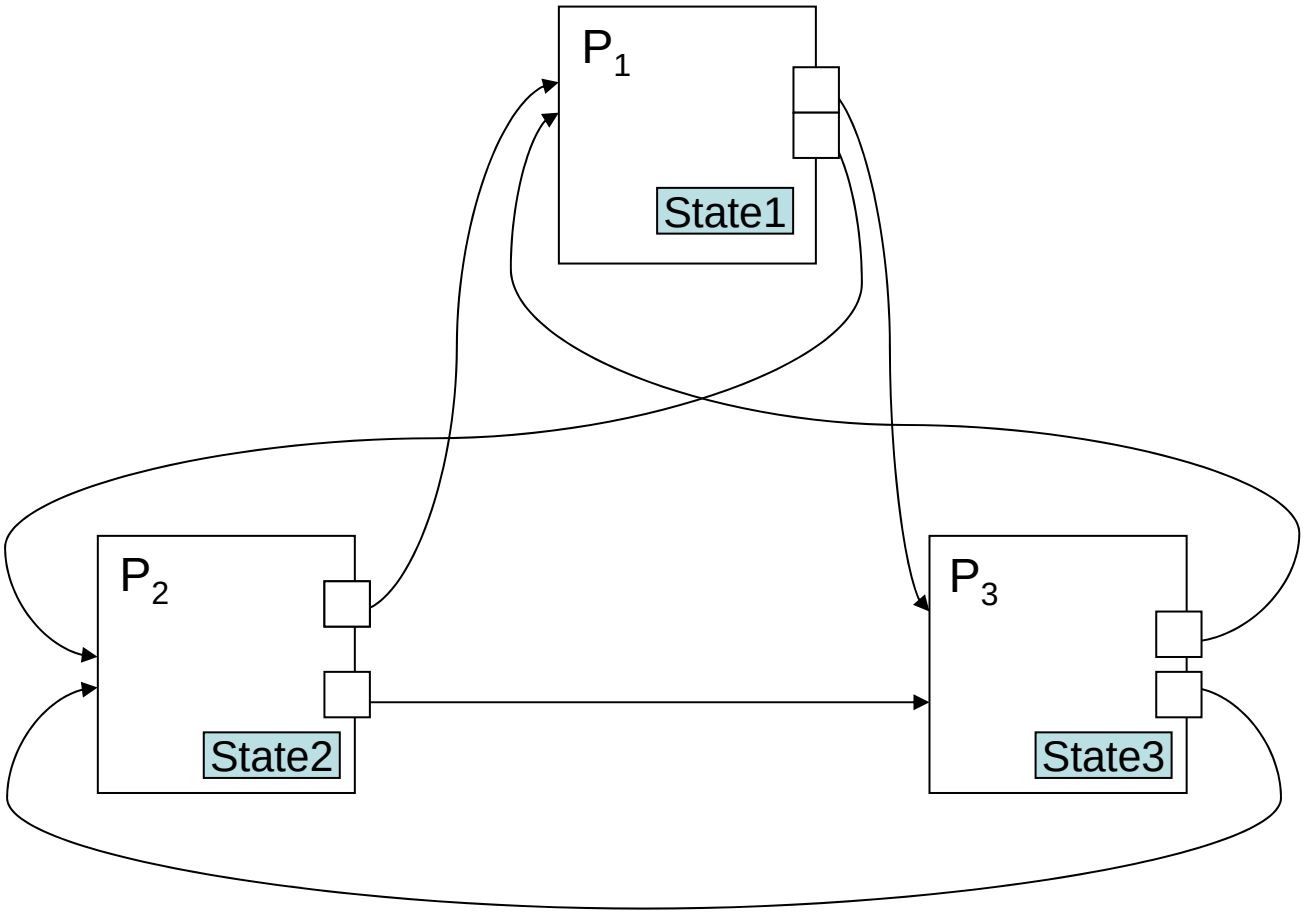
c) Q hält alle ankommenden Nachrichten fest

d) Q erhält einen Marker auf seinem Eingangskanal und stoppt die Aufzeichnung für diesen Kanal

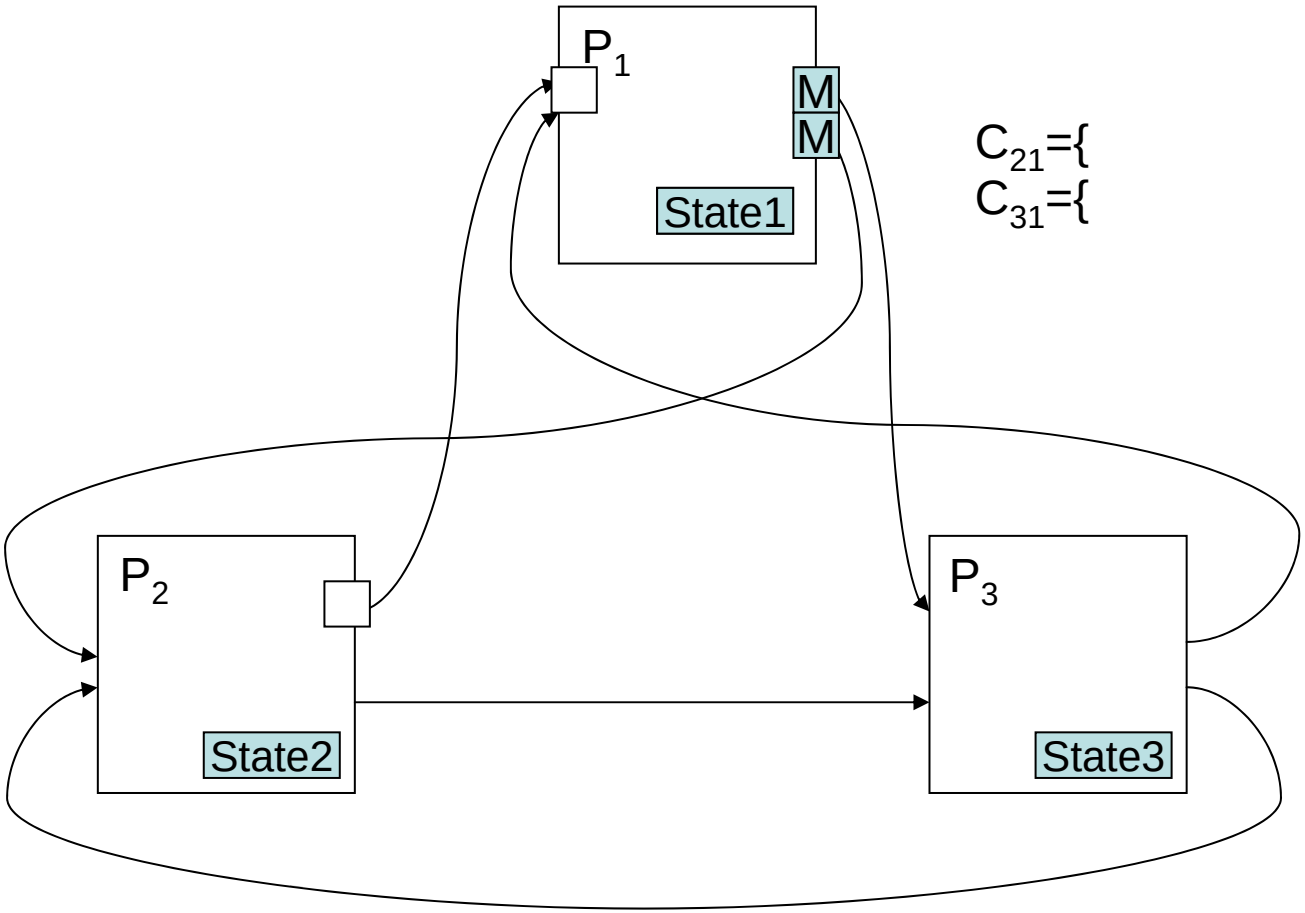
Ende des Algorithmus

- Wenn Q einen Marker auf allen Eingangskanälen erhalten und verarbeitet hat, ist für diesen Prozess der Algorithmus beendet.
- Q sendet dann seinen lokalen Zustand sowie die aufgezeichneten Nachrichten für alle Eingangskanäle an den initiierenden Prozess.
- Dieser wertet schließlich das Ergebnis entsprechend aus, analysiert also z.B. bestimmte Zustandsprädikate.
- Man kann beweisen, dass dieser Algorithmus immer einen konsistenten Cut erzeugt.

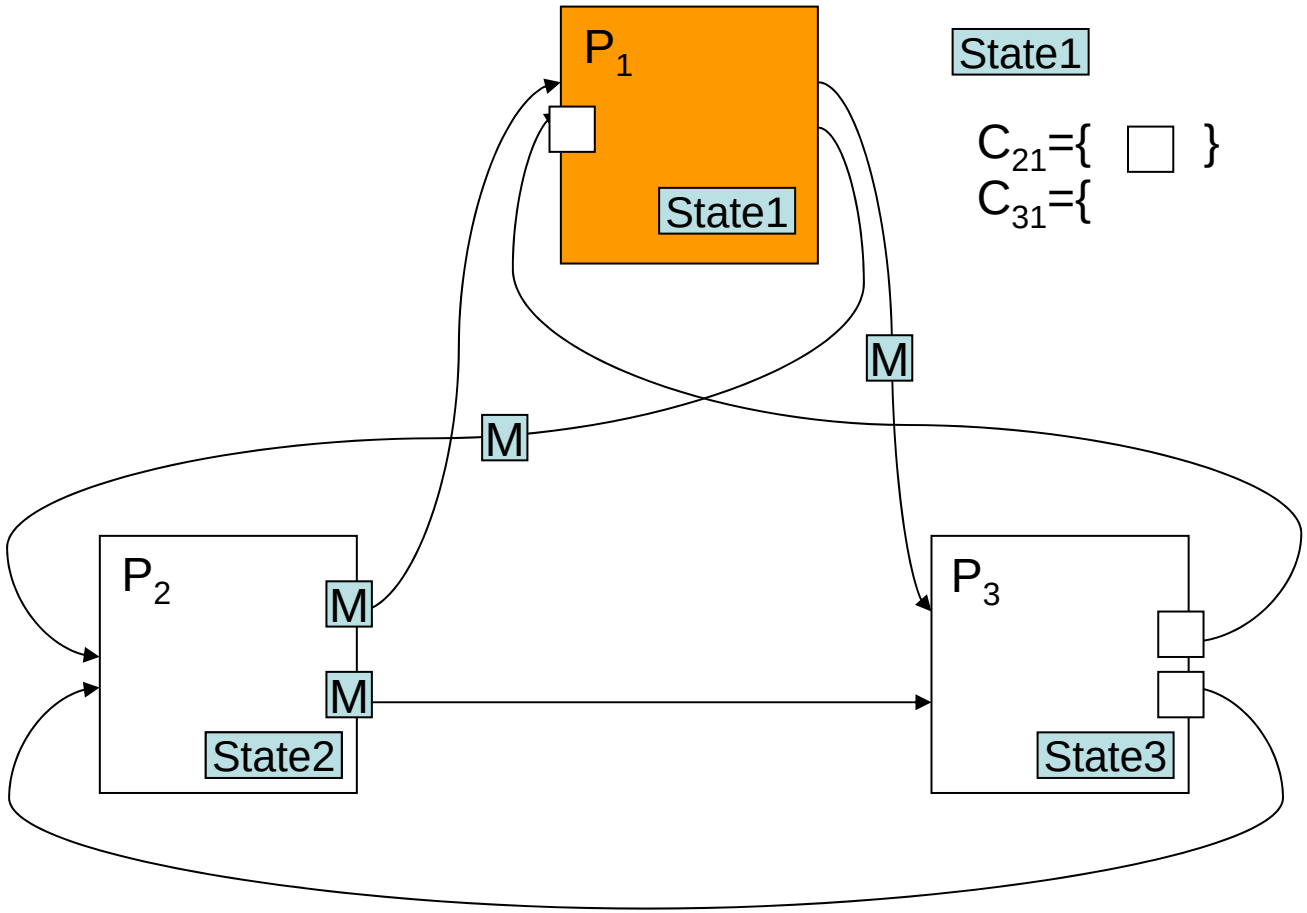
Beispiel



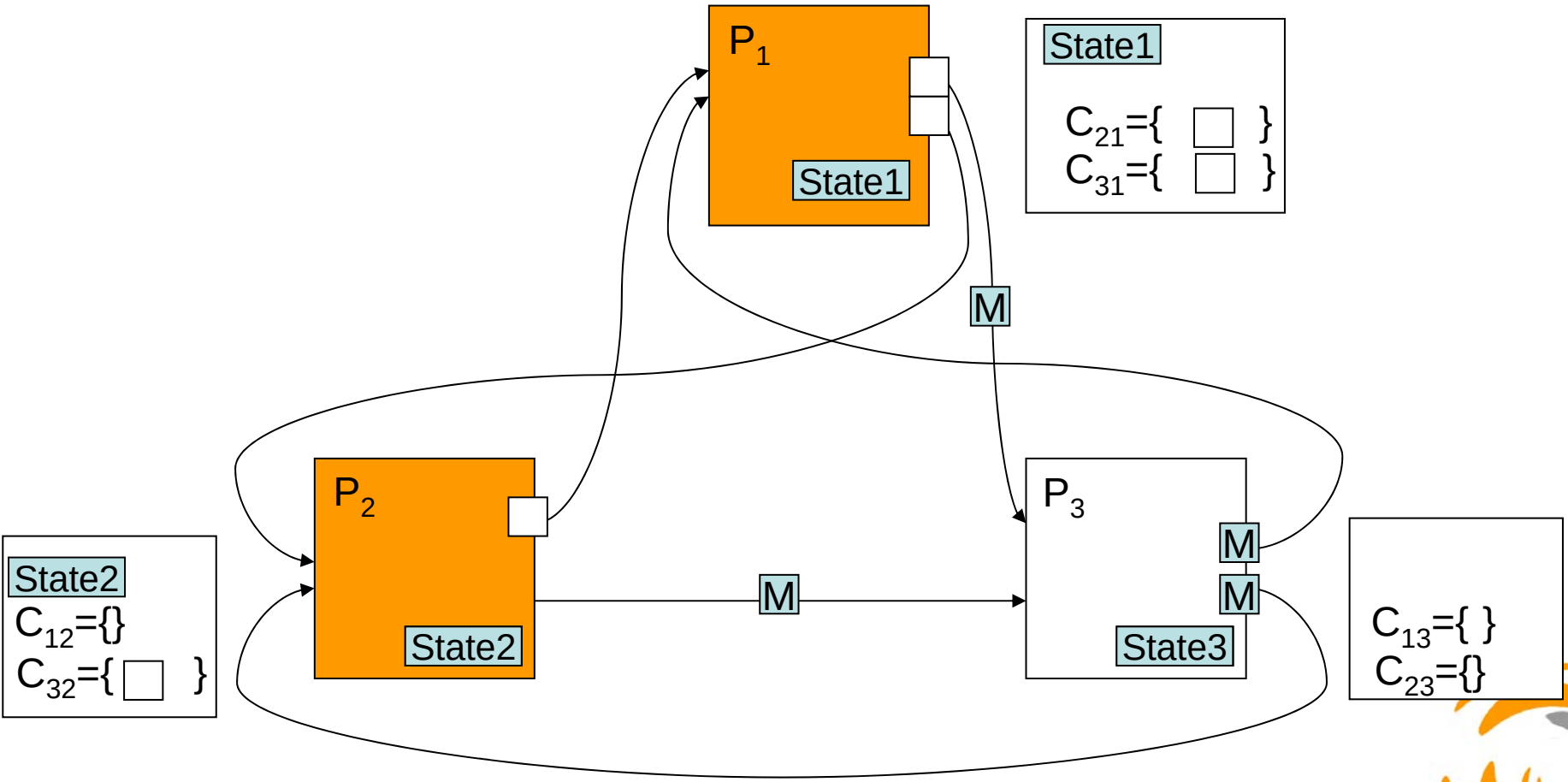
Beispiel



Beispiel



Beispiel


P1
State1
State1
C21={ }
C31={ }
M
P2
State2
M
P3
M
M
State3
State2
C12={}
C32={ }
C13={ }
C23={}

Zusammenfassung

- Es ist unmöglich, einen globalen Systemzustand „gleichzeitig“ aufzuzeichnen.
- Der Algorithmus von Lamport und Chandy macht einen „Distributed Snapshot“.
- Dieser Snapshot hat möglicherweise so nie genau als Systemzustand stattgefunden, aber er ist konsistent.