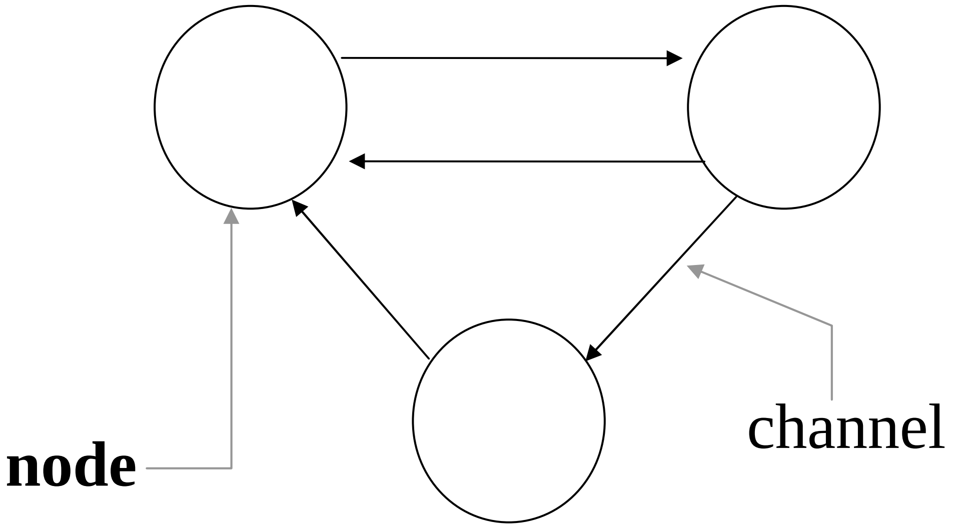
Uncoordinated Checkpointing

The Global State Recording Algorithm

The Model



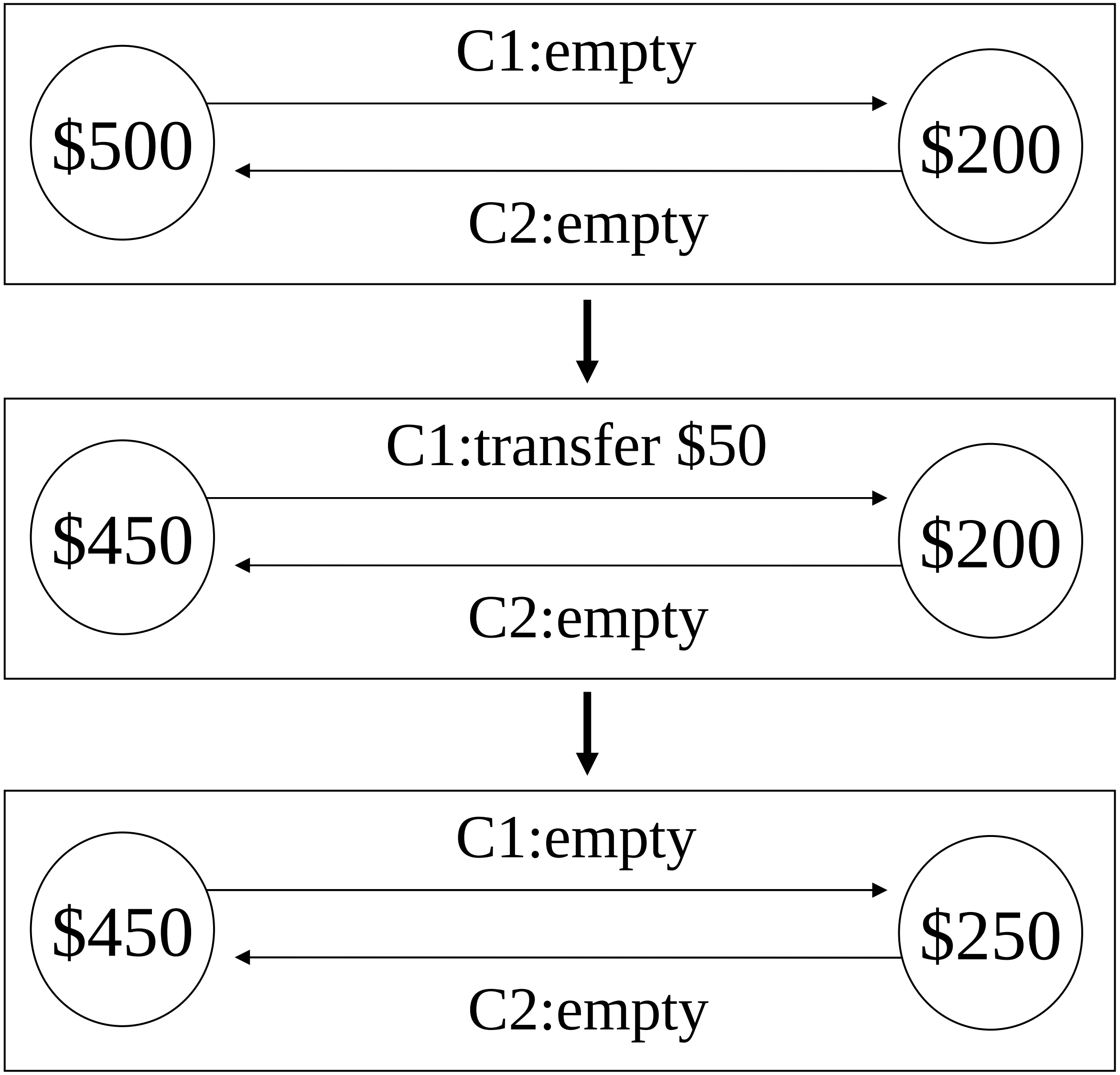
Node properties

- No shared memory
- No global clock

Channel properties:

- FIFO
- loss free
- non-duplicating

The Problem


C1:empty
$500
$200
C2:empty
C1:transfer $50
$450
$200
C2:empty
C1:empty
$450
$250
C2:empty

Distributed Snapshot (Global State Recording)

- Motivation for recording a "consistent" state of the global computation:
  - checkpointing for fault tolerance (rollback, recovery)
  - testing and debugging
  - monitoring and auditing
- Method: detecting stable properties in a distributed system via snapshots. A property is "stable" if, once it holds in a state, it holds in all subsequent states.
  - termination
  - deadlock
  - garbage collection

Definitions

Local State and Actions:

local state:	LS_i
message send:	$send(m_{ij})$
message receive:	$rec(m_{ij})$
time:	$time(x)$

$send(m_{ij}) \varepsilon LS_i$ iff $time(send(m_{ij})) < time(LS_i)$

$rec(m_{ij}) \varepsilon LS_j$ iff $time(rec(m_{ij})) < time(LS_j)$

Predicates:

$transit(LS_i, LS_j) =$

$\{m_{ij} \mid send(m_{ij}) \varepsilon LS_i \wedge !(rec(m_{ij}) \varepsilon LS_j) \}$

$inconsistent(LS_i, LS_j) =$

$\{m_{ij} \mid !(send(m_{ij}) \varepsilon LS_i) \wedge rec(m_{ij}) \varepsilon LS_j \}$

Consistent Global State:

$\forall\, i, \forall\, j : 1 <= i, j <= n :: inconsistent(LS_i , LS_j) = \Phi$

Global-State-Recording Algorithm

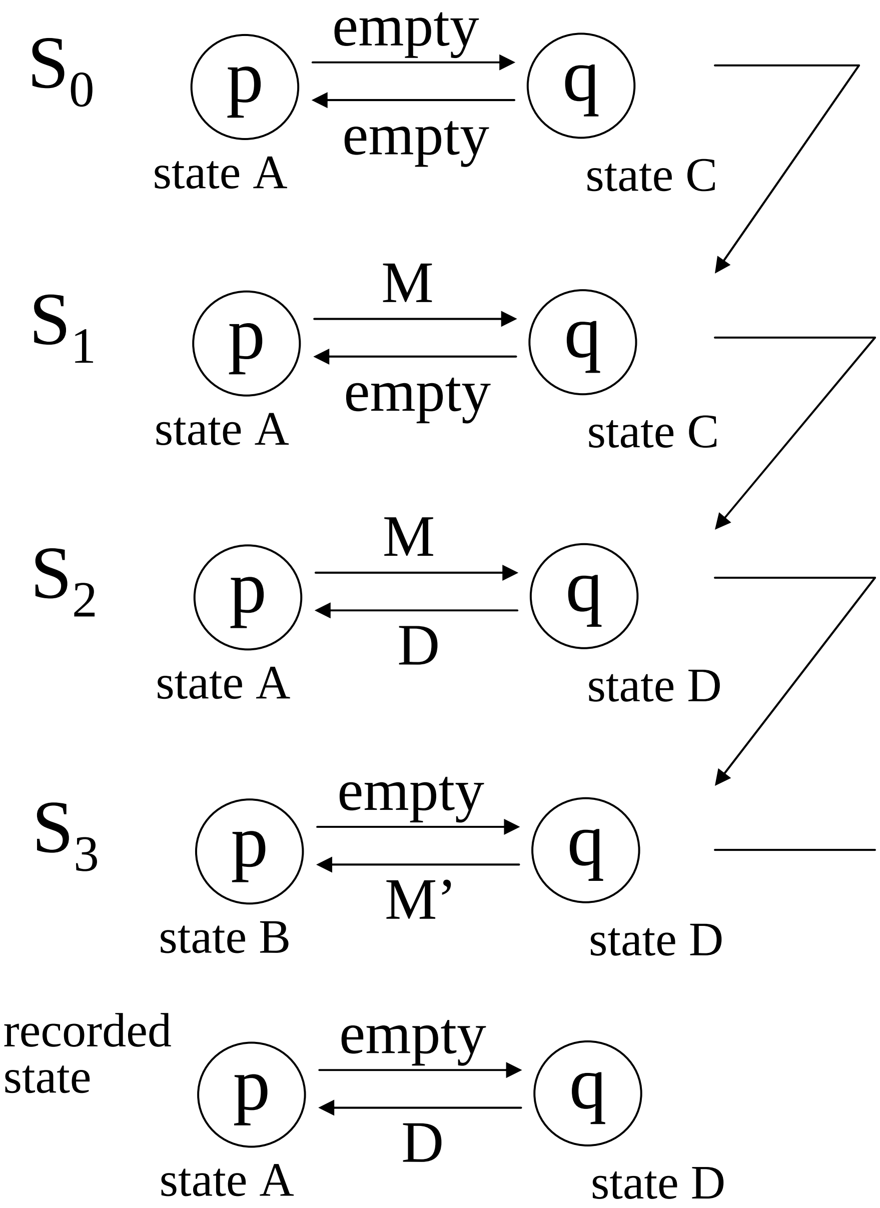
Marker-Sending Rule for a Process p:

```
for (each channel c, incident on, and directed away from p)
        { p sends one marker along c after p records its state
          and before p sends further messages along c; }
```

Marker-Receiving Rule for a Process q:

```
if (q has not recorded its state) then
        { q records its state;
          q records the state of c as the empty sequence;
        }
  else  { q records the state of c as the sequence of message
          received along c after q's state was recorded and before
          q received the marker along c.
      }
```

Global State Recording



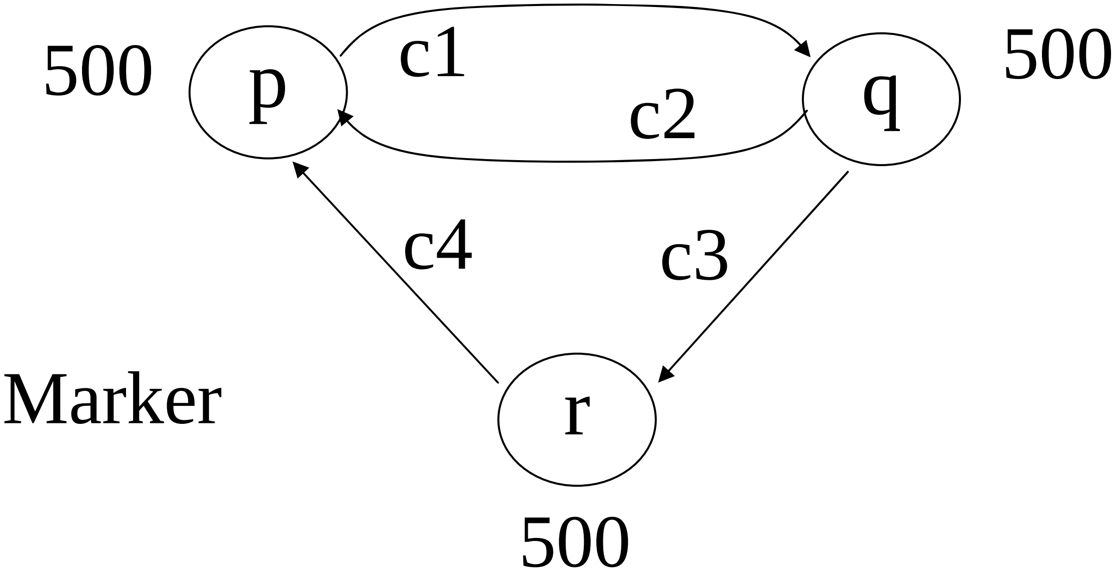
p records its state (A) and sends marker M on channel

before receiving the marker, q changes its state and sends message D.

q receives the marker and records its state (D) and the incoming channel as empty; q send marker M' on its outgoing channel.

on receiving the marker, p records the channel as having message D

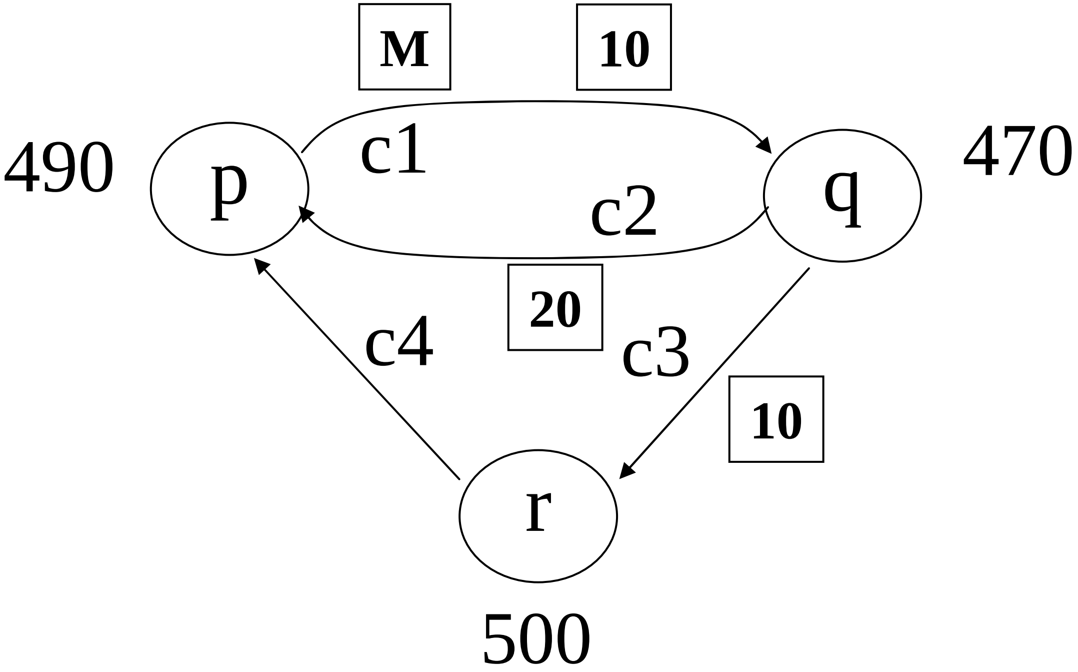
Snapshot/State Recording Example



$\boxed{\textbf{M}}$ = Marker

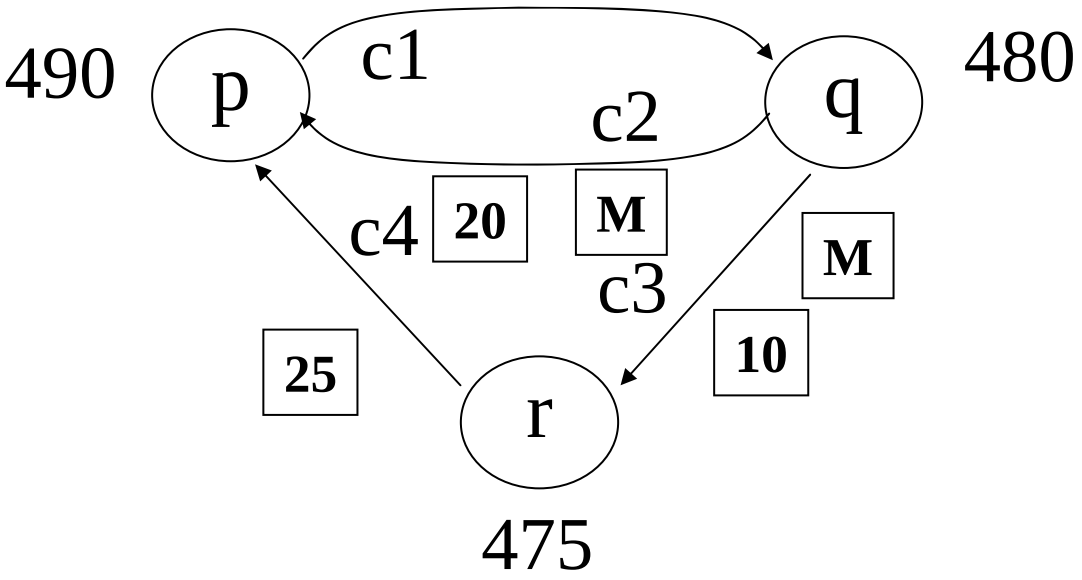
Node	Recorded state				
		c1	c2	c3	c4
p			{ }		{ }
q		{ }			
r				{ }	

Snapshot/State Recording Example (Step 1)



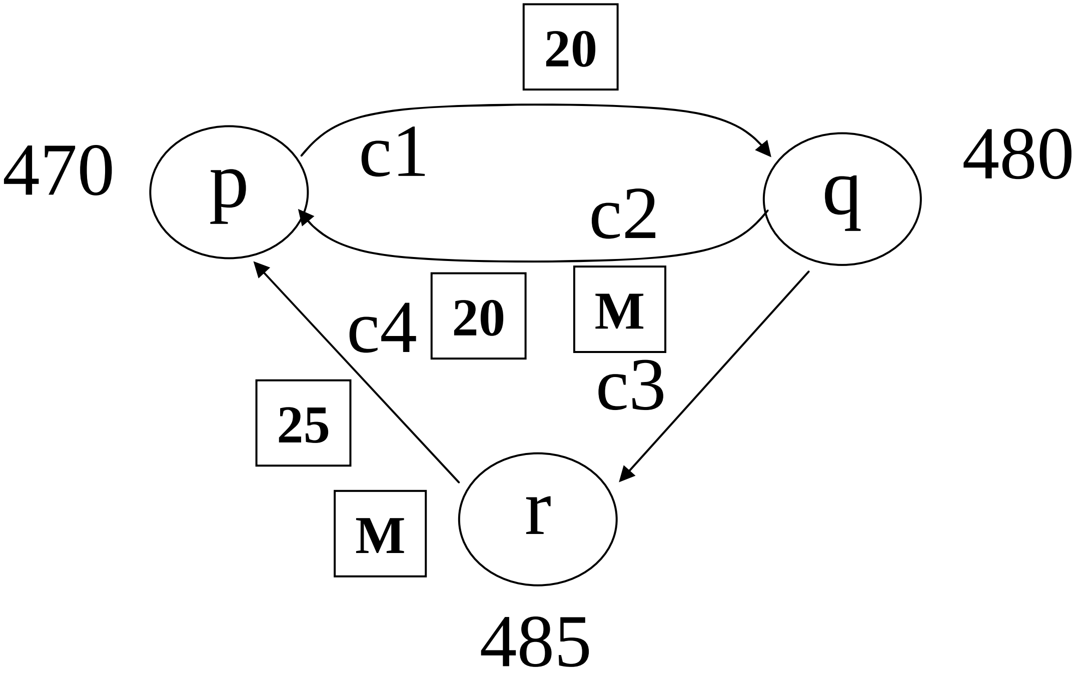
Node	Recorded state				
	state	c1	c2	c3	c4
p	490		{}		{}
q		{}			
r				{}	

Snapshot/State Recording Example (Step 2)



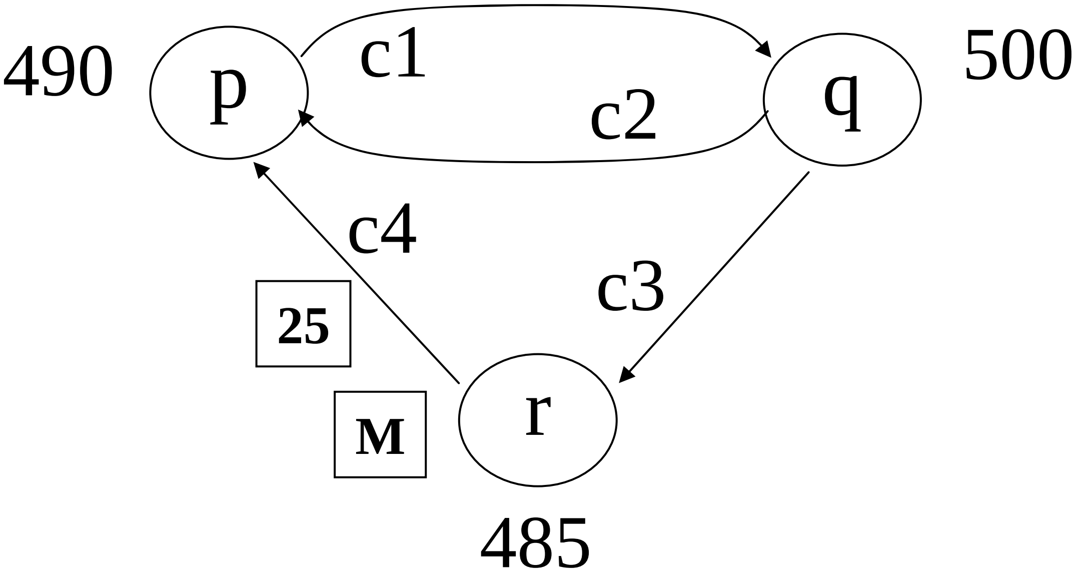
Node	Recorded state				
	state	c1	c2	c3	c4
p	490		{}		{}
q	480	{empty}			
r				{}	

Snapshot/State Recording Example (Step 3)



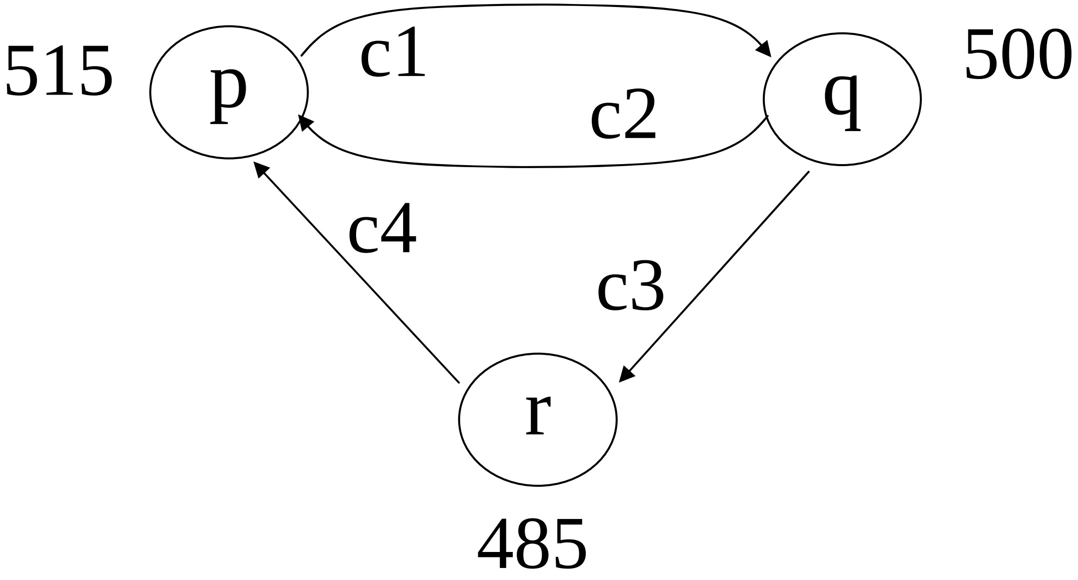
Node	Recorded state				
	state	c1	c2	c3	c4
p	490		{}		{}
q	480	{empty}			
r	485			{empty}	

Snapshot/State Recording Example (Step 4)



Node	Recorded state				
	state	c1	c2	c3	c4
p	490		{20}		{}
q	480	{empty}			
r	485			{empty}	

Snapshot/State Recording Example (Step 5)



Node	Recorded state				
	state	c1	c2	c3	c4
p	490		{20}		{25}
q	480	{empty}			
r	485			{empty}	