CSE 452

Distributed Systems

Vector Clocks

+

Linearizability

Clock Condition

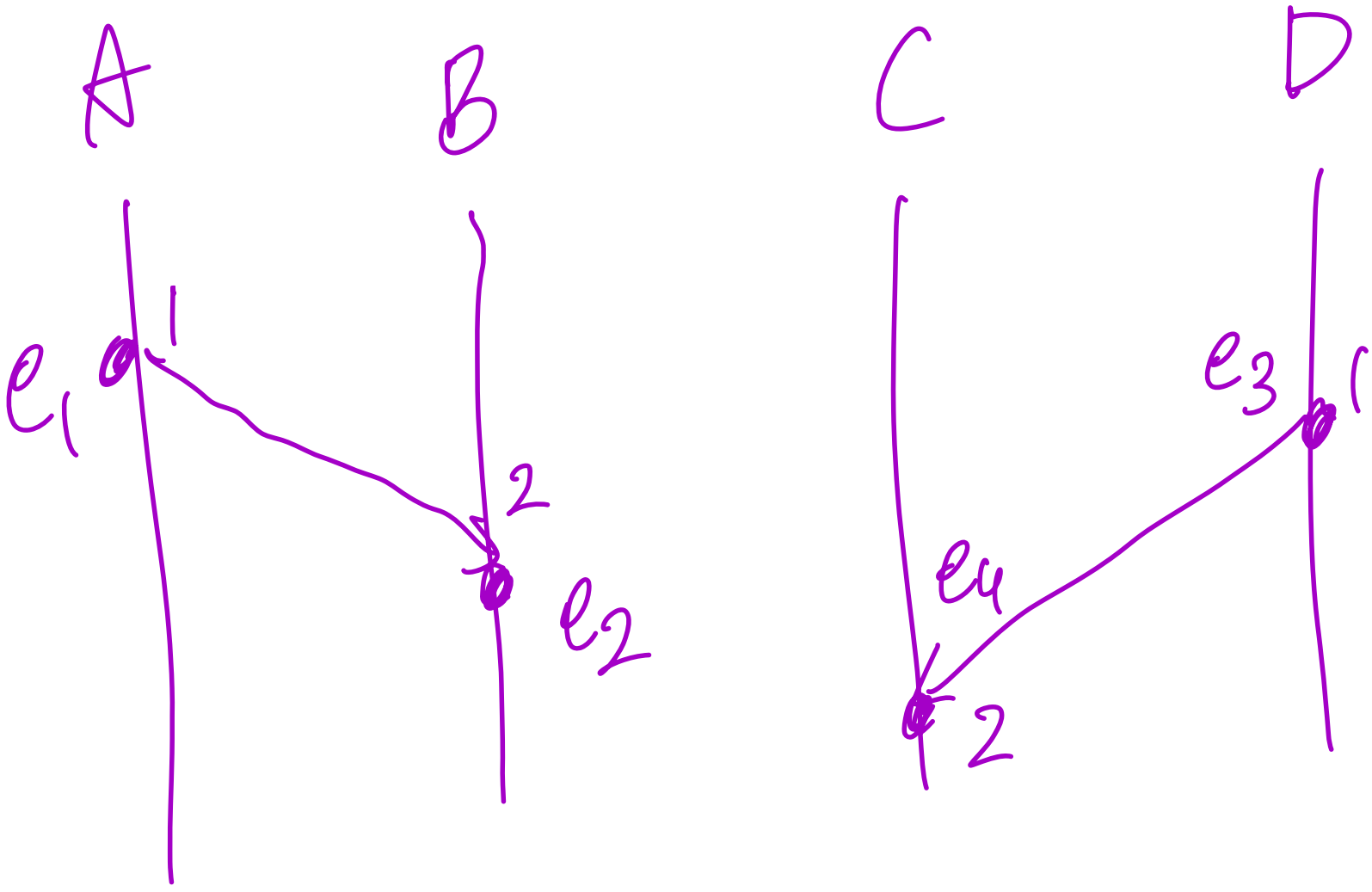
If $e_1 \; HB \; e_2$, then $C(e_1) < C(e_2)$

The converse is false:

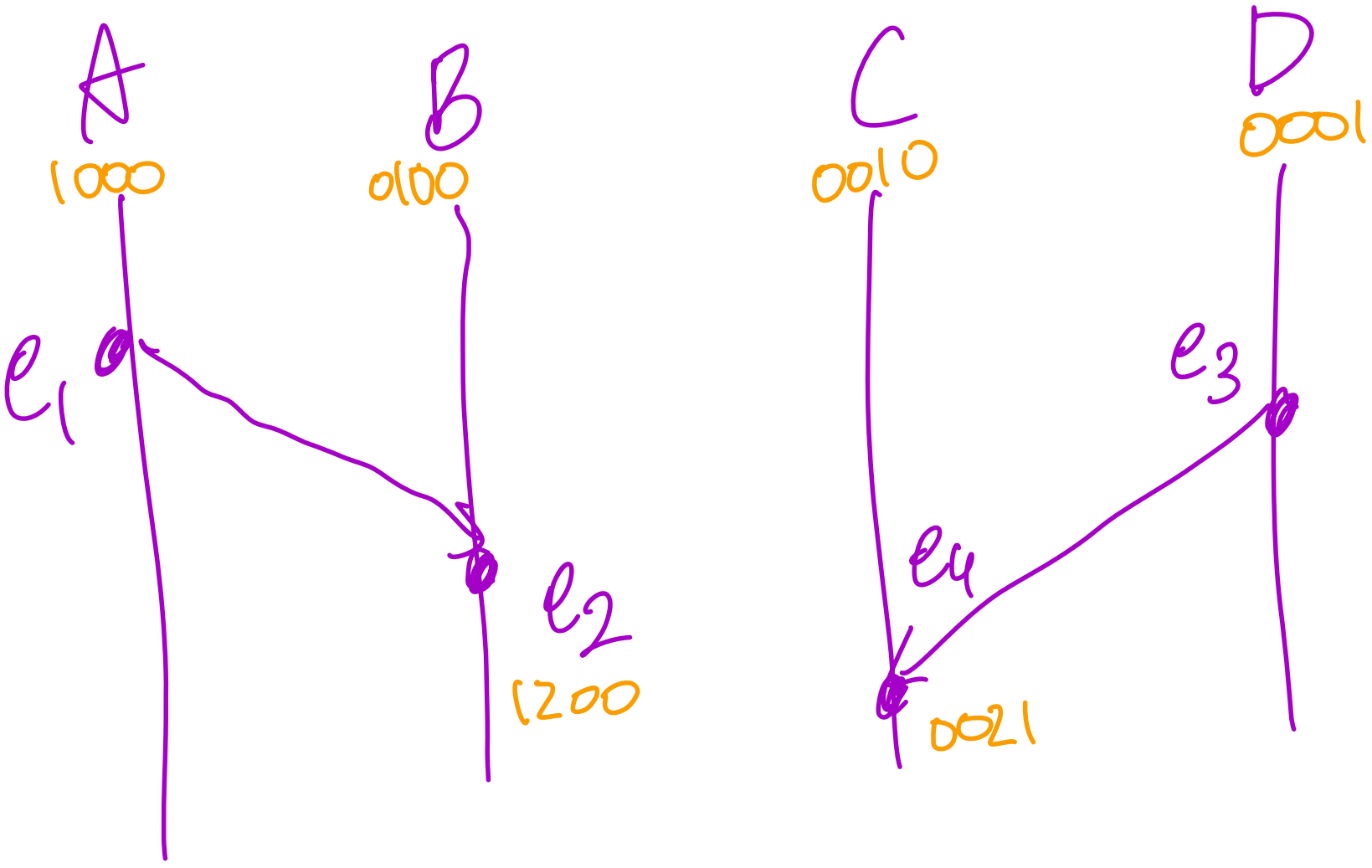
"If $C(e_1) < C(e_2)$ then $e_1 \; HB \; e_2$"

Vector Clock

- assign timestamps to events
  - redefine timestamp: vector of numbers
    list of numbers
    - length = # of nodes in system
- ith component is the highest clock value you've heard about node i

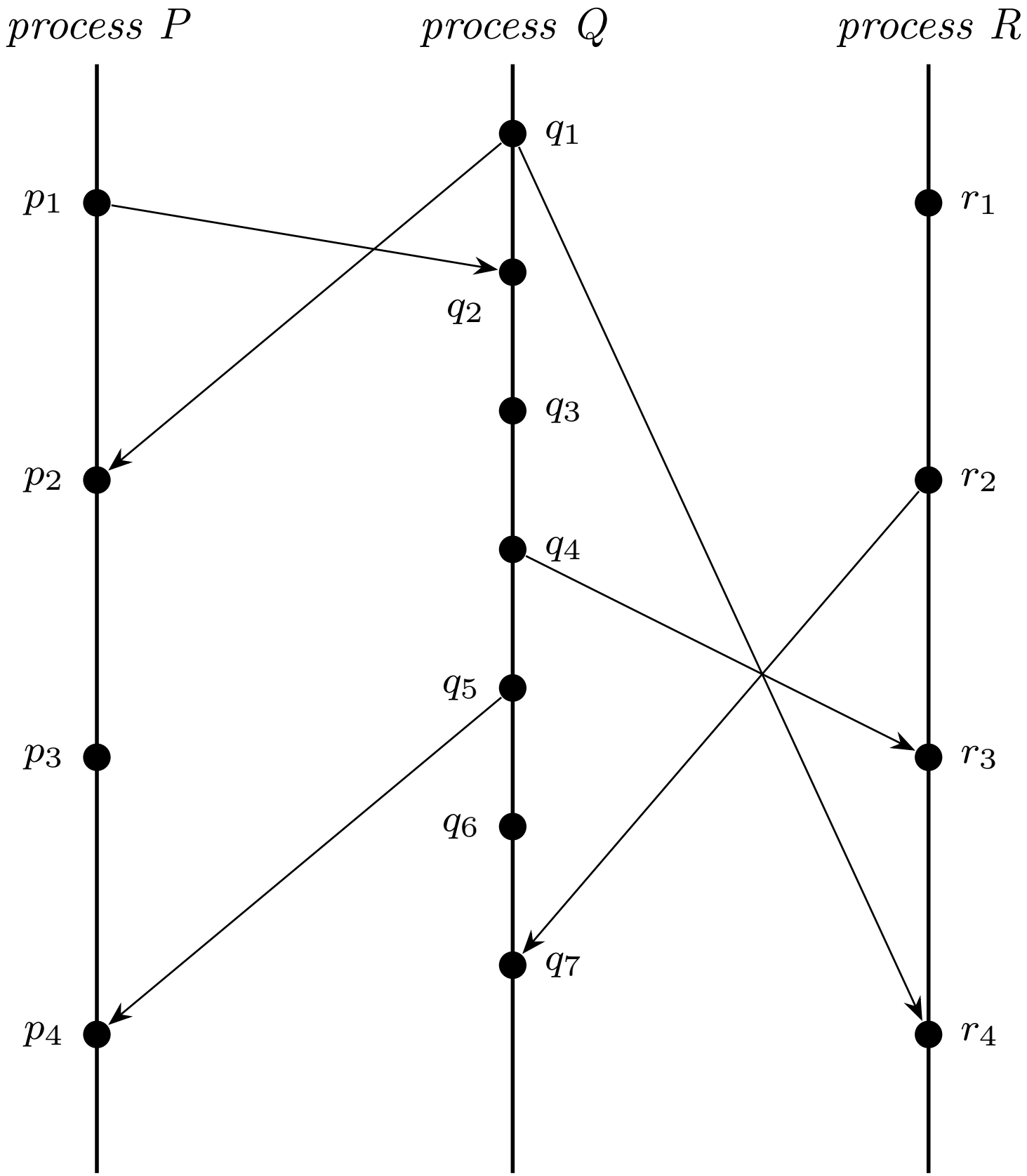


e_1 VS e_4



e_1 VS e_4

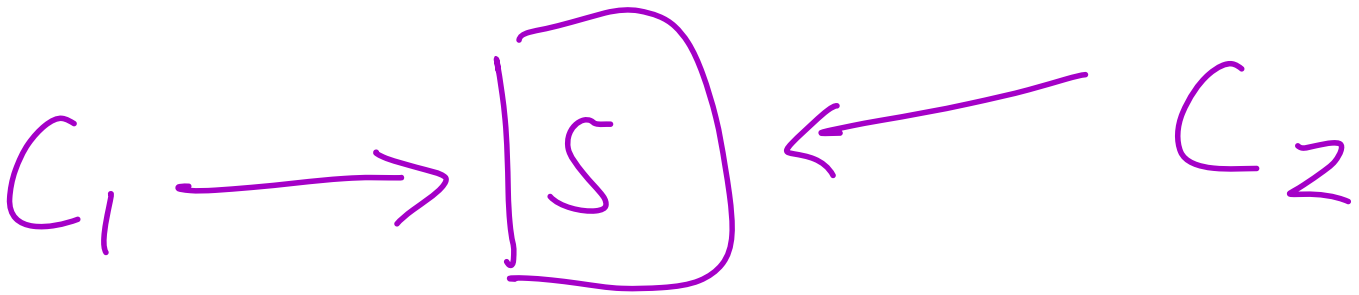
1000 0021 so concurrent

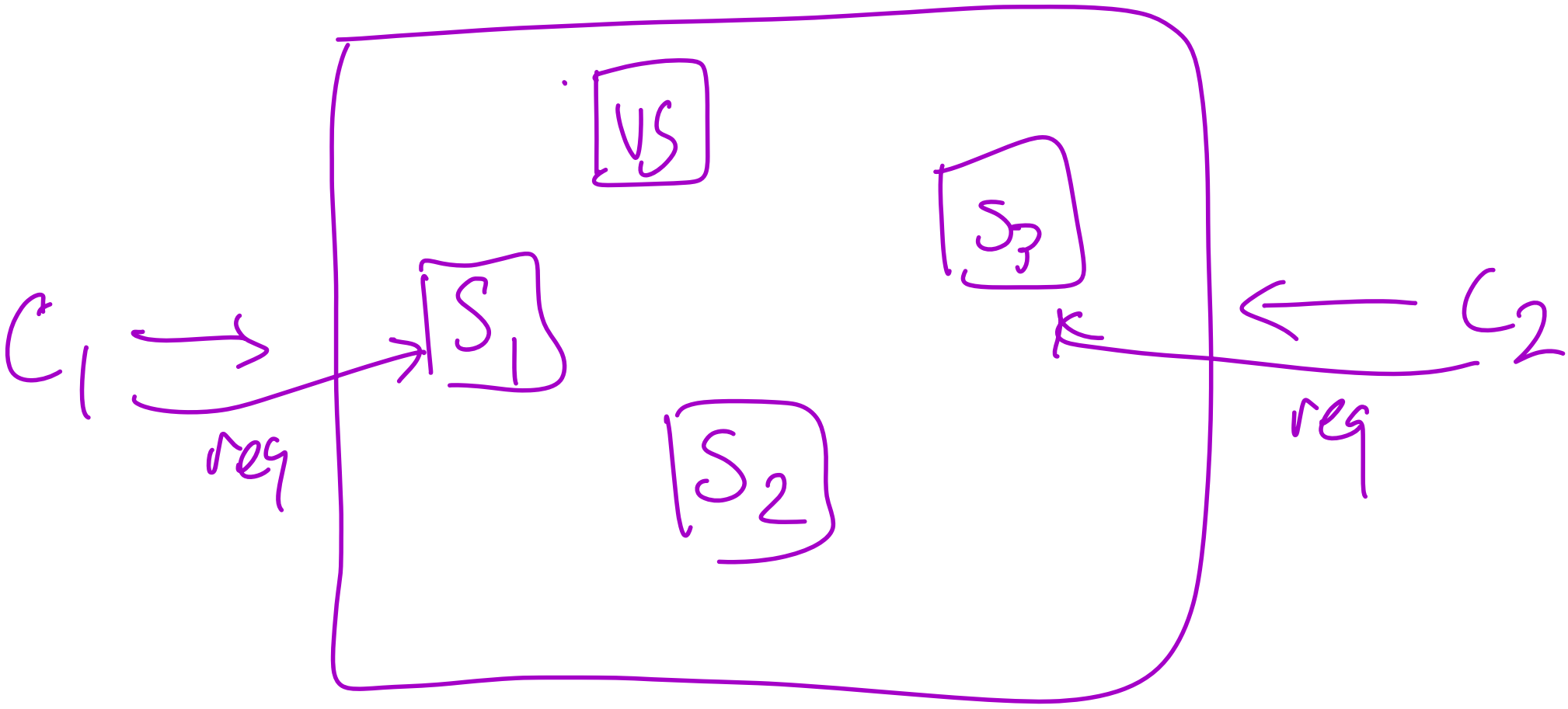

process P
process Q
process R
q_1
p_1
r_1
q_2
q_3
p_2
r_2
q_4
q_5
p_3
r_3
q_6
q_7
p_4
r_4

Vector Clock alg.

- init every node w/ 1 in its component
- send timestamps on msgs
- on recv, take the max and add 1 to receiver's component
- increment the node's component on local steps

Claim: $e_1 \; HB \; e_2$ iff $VC(e_1) \leq VC(e_2)$

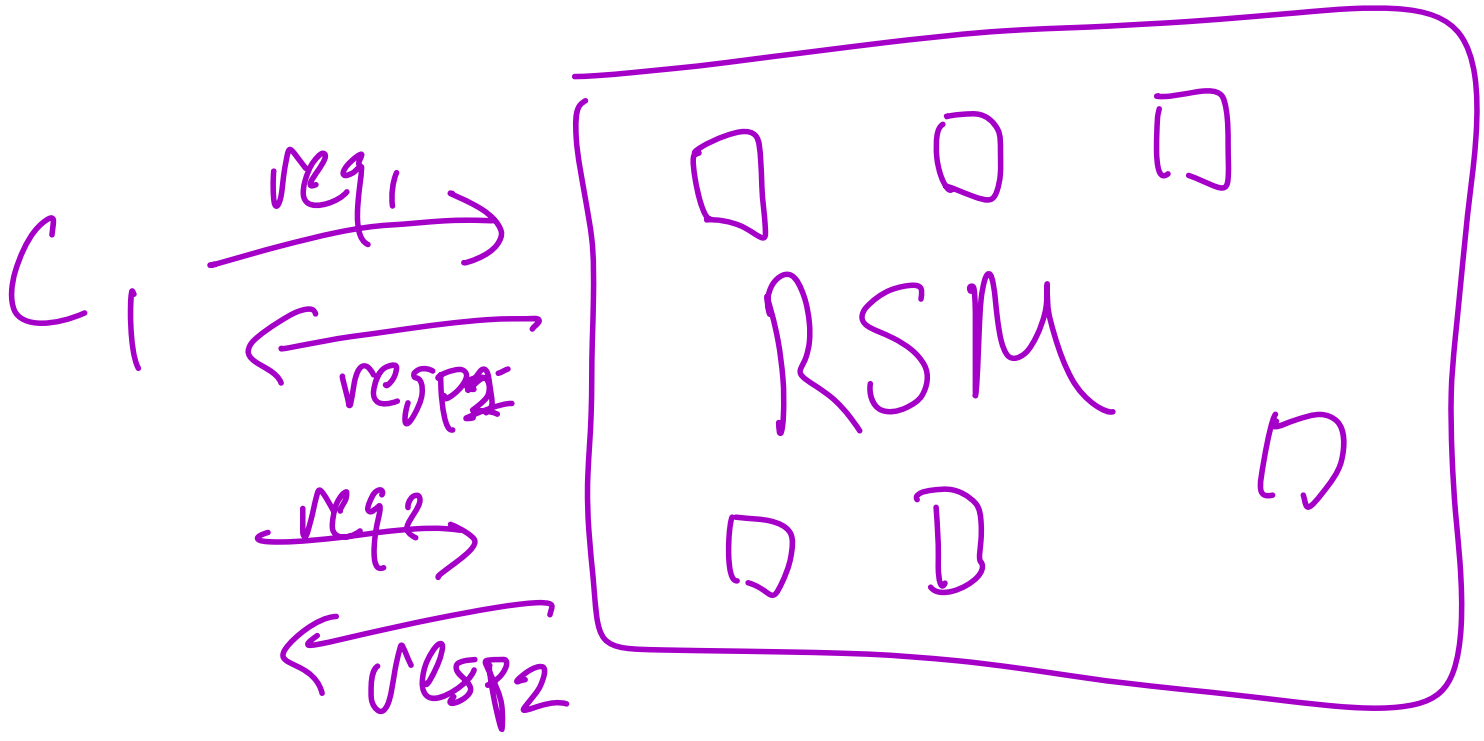

C1
S
C2


VS
S1
S2
S3
C1
C2
req
req

Consistency Model

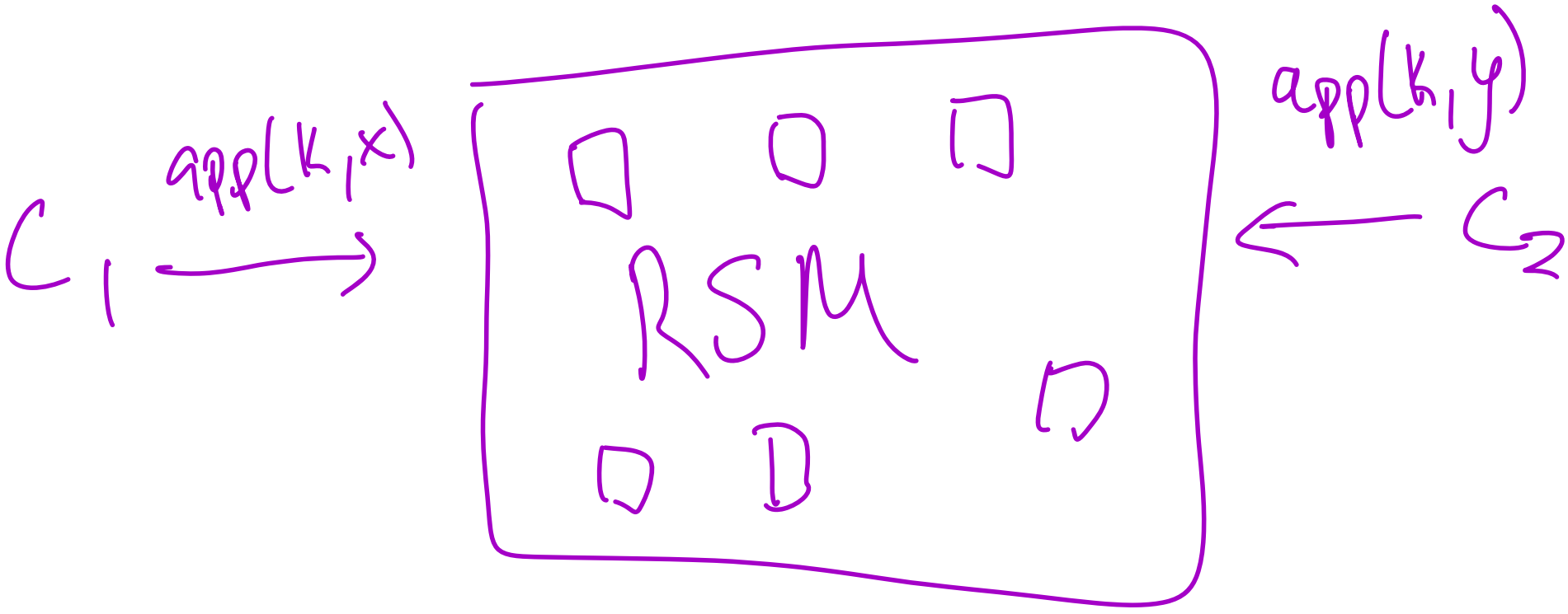
- answers the question
  what executions of a replicated state machine are correct?

- executions → bad



- one client
- one req at a time

⟶ execute in order



execute in some order

- if C_1 goes first, its response AppRes(x) and C_2 will get AR(xy)
- if C_2 goes first, it gets AR(y) and C_1 get AR(yx)

Sequential Consistency

an execution is allowed if there is an order of all the client operations in the execution such that the responses returned by the RSM agree with executing the requests in that order

Linearizability

- Sequentially consistent plus
  - if request r_2 is submitted after response to request r_1 is received then r_2 must appear after r_1 in the apparent execution order