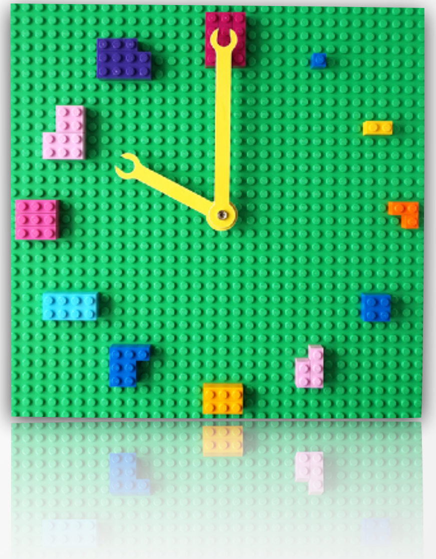
KTH-2022

Advanced Course

Distributed Systems

Logical Clocks

a short intervention



Paris Carbone

LOGICAL CLOCKS

- A clock is function **t** from the events to a totally order set such that for events a and b
  - if $a \rightarrow b$ then $\mathbf{t}(a) < \mathbf{t}(b)$

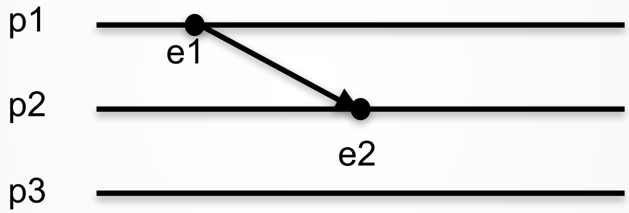
- We are interested in $\rightarrow$ being the happen-before relation

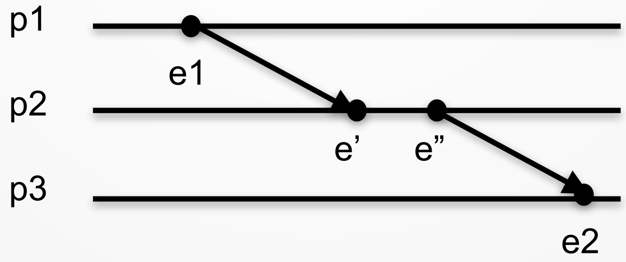
CAUSAL ORDER (HAPPEN BEFORE)

- The relation $\rightarrow_{\beta}$ on the events of an execution (or trace β), called also causal order, is defined as follows
  - If a occurs before b on the same process, then $a \rightarrow_{\beta} b$
  - If a is a send(m) and b deliver(m), then $a \rightarrow_{\beta} b$
  - $a \rightarrow_{\beta} b$ is transitive
    - i.e. If $a \rightarrow_{\beta} b$ and $b \rightarrow_{\beta} c$ then $a \rightarrow_{\beta} c$

- Two events, a and b, are concurrent if not $a \rightarrow_{\beta} b$ and not $b \rightarrow_{\beta} a$
- $a \| b$

Causal Order (happen before)





Observing Causality

So causality is all that matters…

…how to locally tell if two events are causally related?

LAMPORT CLOCKS AT PROCESS P

- Each process has a local logical clock, kept in variable $\mathbf{t_p}$, initially $\mathbf{t_p} = 0$
  - A process p piggybacks $(\mathbf{t_p}, p)$ on every message sent
- On internal event *a*:
  - $\mathbf{t_p} := \mathbf{t_p} + 1$; perform internal event *a*
- On send event message m:
  - $\mathbf{t_p} := \mathbf{t_p} + 1$; send(m, $(\mathbf{t_p}, p)$)
- On delivery event *a* of m with timestamp (t_q, q) from q:
  - $\mathbf{t_p} := \max(\mathbf{t_p}, \mathbf{t_q}) + 1$; perform delivery event *a*

Lamport Clocks (2)

Observe the timestamp (t, p) is unique

Comparing two timestamps (t_p,p) and (t_q,q)

$(t_p,p)<(t_q,q)$ iff $(t_p<t_q$ or $(t_p=t_q$ and $p<q))$

i.e. break ties using process identifiers

e.g. $(5,p_5) < (7,p_2)$, $(4,p_2) < (4,p_3)$

Lamport Clocks (2)

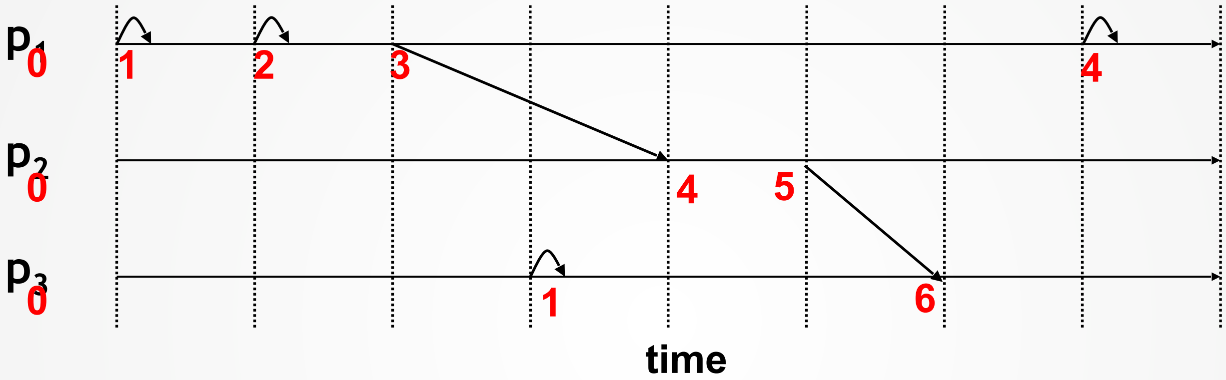
Lamport logical clocks guarantee that:

If $a \rightarrow_{\beta} b$, then $\mathbf{t}(a) < \mathbf{t}(b)$,

where $\mathbf{t}(a)$ is Lamport clock of event a

- events a and b are on the same process p, t_p is strictly increasing, so if a is before b, then t(a) < t(b)
- a is a send event with t_q and b is deliver event, t(b) is at least one larger than t_q (t(a))
- transitivity of t(a) < t(b) < t(c) implies the transitivity condition of the happen before relation

LAMPORT LOGICAL CLOCKS



Lamport logical clocks guarantee that:

If $a \rightarrow_{\beta} b$, then $\mathbf{t}(a) < \mathbf{t}(b)$,

if $\mathbf{t}(a) \geq \mathbf{t}(b)$, then not $(a \rightarrow_{\beta} b)$

Vector Clocks

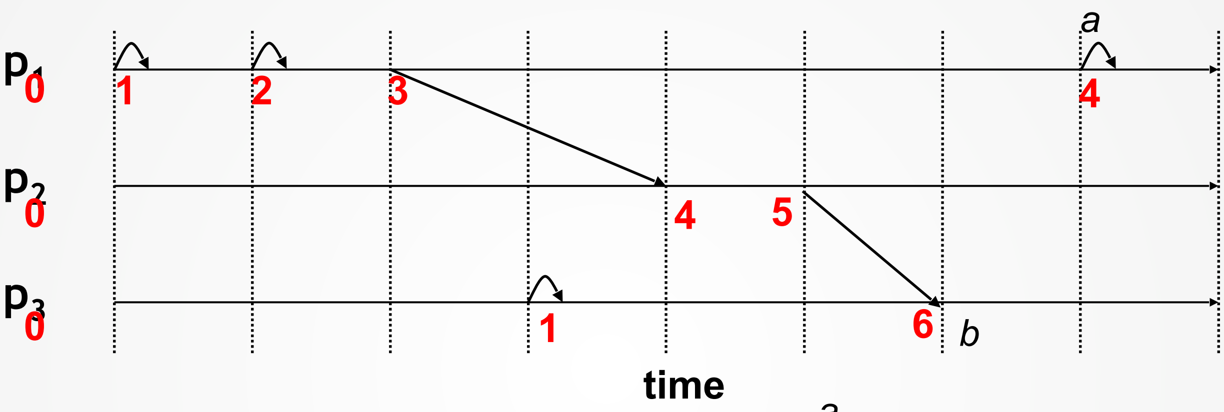
VECTOR CLOCKS

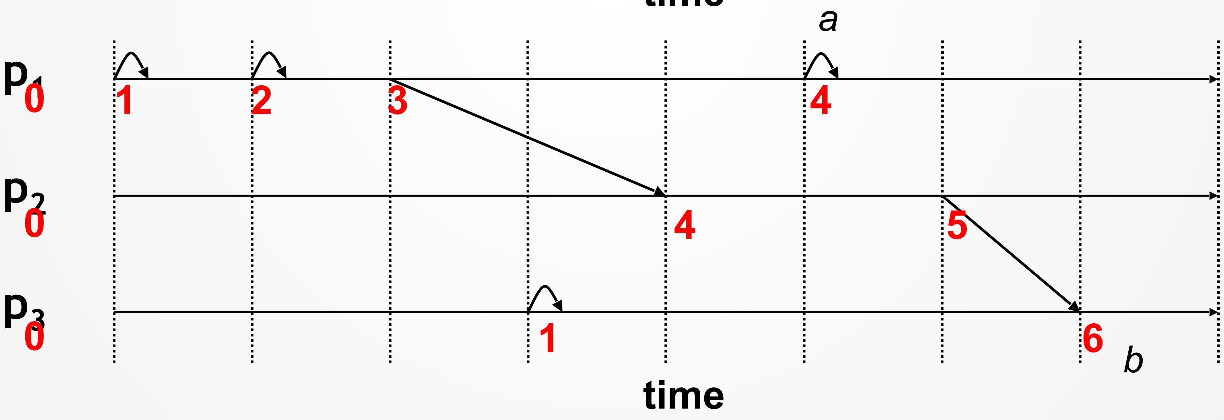
- The happen-before relation is a partial order
- In contrast logical clocks are total
  - Information about non-causality is **lost**
    - We cannot tell by looking to the timestamps of event a and b whether there is a causal relation between the events, or they are concurrent
- Vector clocks guarantee that:
  - if $\mathbf{v}(a) < \mathbf{v}(b)$ then $a \rightarrow_\beta b$, in addition to
  - if $a \rightarrow_\beta b$ then $\mathbf{v}(a) < \mathbf{v}(b)$
    - where $\mathbf{v}(a)$ is a vector clock of event a

Non-causality and Concurrent events

- Two events a and b are concurrent $(a \,||_\beta\, b)$ in an execution E (trace(E) = β) if
  - **not** $a \rightarrow_\beta b$ and **not** $b \rightarrow_\beta a$
- Computation theorem implies that if $(a \,||_\beta\, b)$ in β then there are two executions (with traces β_1 and β_2) that are similar where a occurs before b in β_1, b occurs before a in β_2

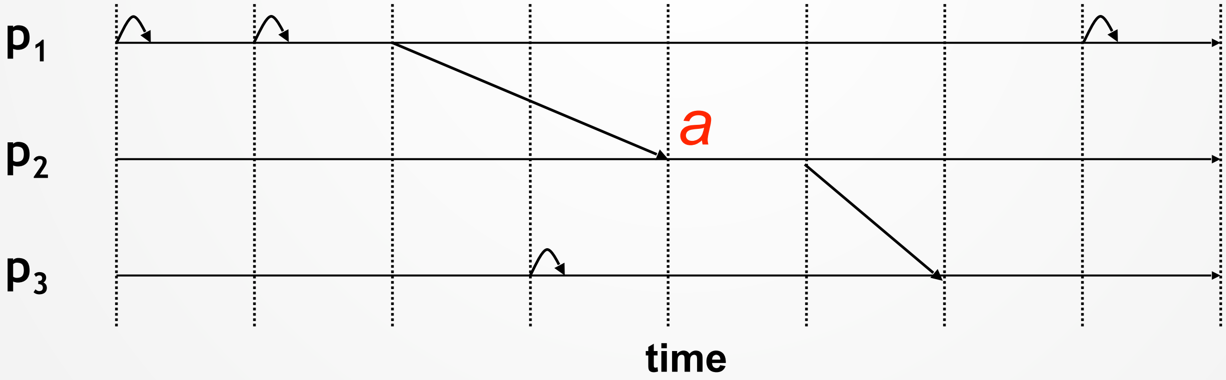
NON-CAUSALITY AND CONCURRENT EVENTS





VECTOR CLOCK DEFINITION

- Vector clock for an event ***a***
  - $\mathbf{v}(a) = (x_1,\ldots,x_n)$
  - x_i is the number of events at p_i that happens-before a
  - for each such event e: e $\rightarrow$ a



VECTOR TIMESTAMPS

- Processes $p_1, \ldots, p_n$
- Each process p_i has local vector $\mathbf{v}$ of size $\mathbf{n}$ (number of processes)
  - $\mathbf{v}[i] = 0$ for all i in 1…n
  - Piggyback $\mathbf{v}$ on every sent message
- For each transition (on each event) update local $\mathbf{v}$ at p_i:
  - $\mathbf{v}[i] := \mathbf{v}[i] + 1$ (internal, send or deliver)
  - $\mathbf{v}[j] := \max(\mathbf{v}[j], \mathbf{v}_{\mathbf{q}}[j])$, for all $j \neq i$ (deliver)
    - where $\mathbf{v}_{\mathbf{q}}$ is clock in message received from process q

ID2203

KTH-2022

Comparing Vector Clocks

- $v_p \leq v_q$ iff
  - $v_p[i] \leq v_q[i]$ for all i
- $v_p < v_q$ iff
  - $v_p \leq v_q$ and for some i, $v_p[i] < v_q[i]$
- v_p and v_q are concurrent ($v_p \parallel v_q$) iff
  - not $v_p < v_q$, and not $v_q < v_p$

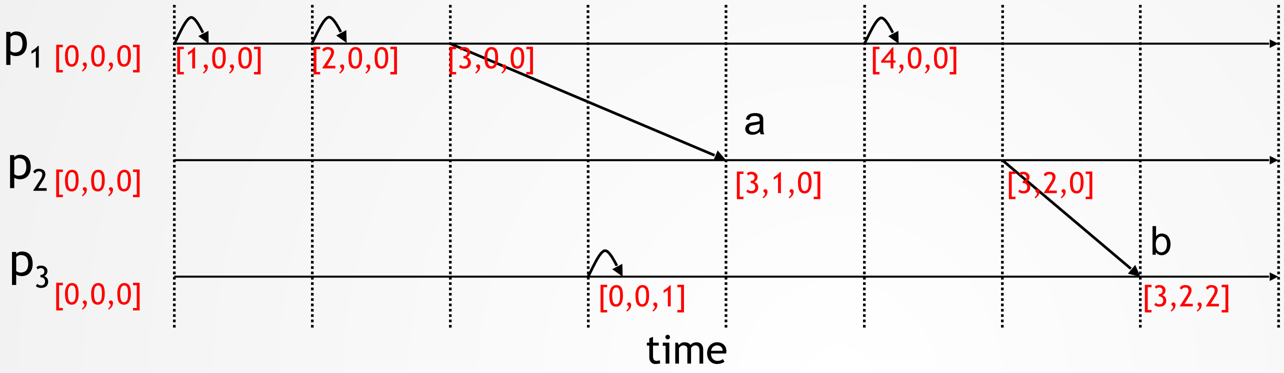
$[3,0,0] \leq [3,1,0]$

$[3,0,0] < [3,1,0]$

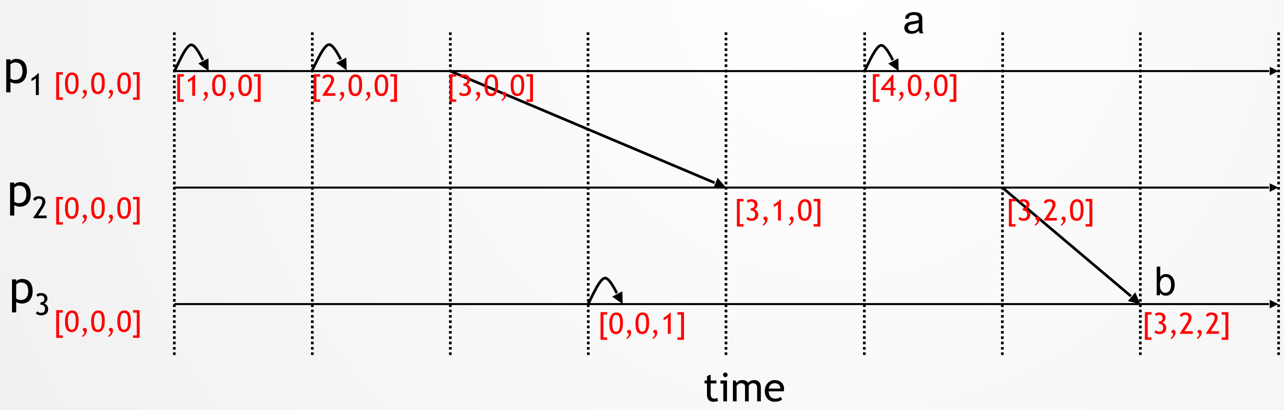
$[3,1,0] <> [4,0,0]$

- Vector clocks guarantee
  - If $v(a) < v(b)$ then $a \rightarrow b$, and
  - If $a \rightarrow b$, then $v(a) < v(b)$
    - where $v(a)$ is the vector clock of event a

Example of Vector Timestamps

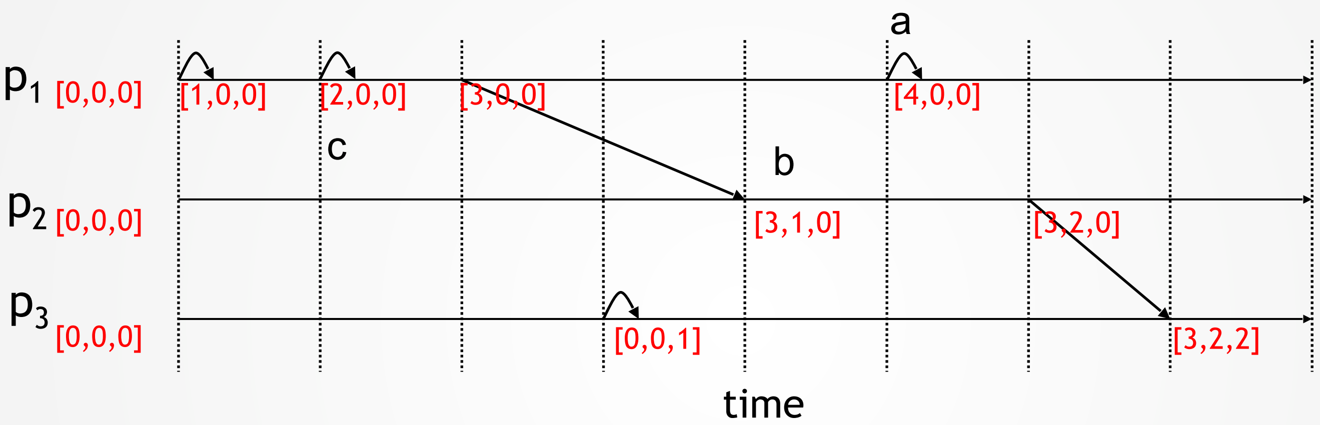


v(a) < v(b) implies a → b



v(a) <> v(b) implies a || b

Vector Timestamps

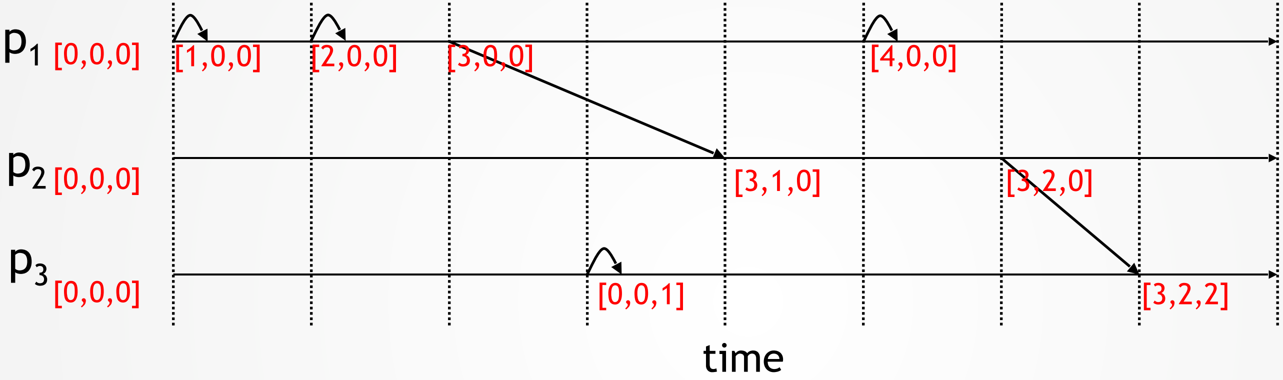


For any events a and b, and trace β :

$\mathbf{v}$(a) and $\mathbf{v}$(b) are incomparable if and only if a||b

$\mathbf{v}(a) < \mathbf{v}(b)$ if and only if $a \rightarrow b$

EXAMPLE OF VECTOR TIMESTAMPS



Great! But cannot be done with smaller vectors than size n, for n nodes

PARTIAL AND TOTAL ORDERS

- Only a partial order or a total order? [d]
  - the relation $\rightarrow_\beta$ on events in executions
    - Partial: $\rightarrow_\beta$ doesn't order concurrent events
  - the relation < on Lamport logical clocks
    - Total: any two distinct clock values are ordered (adding pid)
  - the relation < on vector timestamps
    - Partial: timestamp of concurrent events not ordered

LOGICAL CLOCK VS. VECTOR CLOCK

Logical clock

$$\text{If } a \rightarrow_{\beta} b \text{ then } t(a) < t(b) \qquad (1)$$

Vector clock

$$\text{If } a \rightarrow_{\beta} b \text{ then } v(a) < v(b) \qquad (1)$$

$$\text{If } v(a) < v(b) \text{ then } a \rightarrow_{\beta} b \qquad (2)$$

Which of (1) and (2) is more useful? [d]

What extra information do vector clocks give? [d]