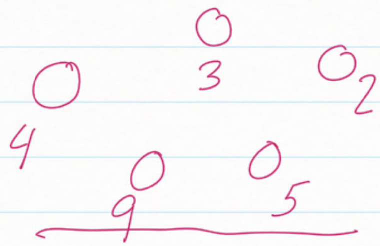
FLP result

↳

Impossibility of Distributed Consensus with One Faulty Process

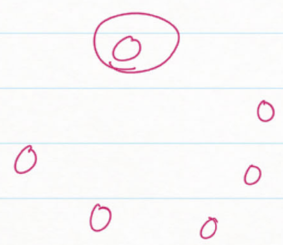
<u>Consensus</u>



Distributed systems

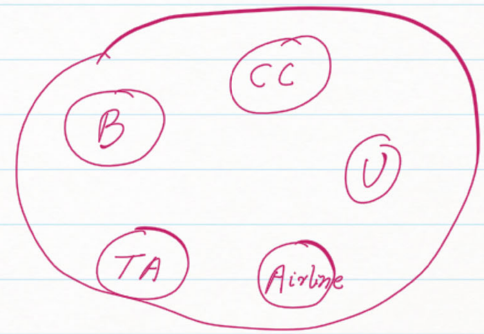
- One among the proposed values is chosen
- Everybody agrees

Leader election



Agreement

faults
delays



→ Issue the ticket

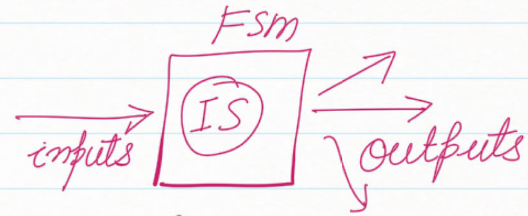
→ Agreement

Consensus

One faulty process → NO

- N processes
- Reliable message delivery, out of order delivery
- non-faulty, faulty (at max. one)
- cannot differentiate between a slow and a failed process

Process model



Step → receive a message, send messages.
delayed

$C \to C_3$

$C \to C_2$

$C_1 \quad C_2 \quad C_3$ reachable accessible

Process

$x_p \in \{0, 1\} \to \boxed{} \to y_p \in \{b, 0, 1\}$

undecided

Configuration

- Outputs
- internal states
- messages

$\underline{e} \to (p, m)$

$e(C) \to C'$

$\underbrace{e_1(e_2(e_3 \cdots\cdots}_{} (C)))) \to C^n$

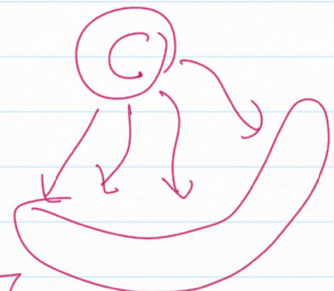
$\sigma(C) \to C^n$

send$(p, m) \to$ p : process, m : message

receive$(p) \to$ p : process

$e \to (p, m)$

$\boxed{IS} \to \boxed{IS'}$



processes
- → 1-bit input
- → output {b, 1, 0}
- → Internal state

Configuration

$\sigma(C) \rightarrow C'$

$C \rightarrow p \ (0,1)$

$C \rightarrow$ decided

$C \rightarrow C_x(0)$

$C \rightarrow C_y(1)$

• partially correct execution

1. No accessible config has more than one decision value
2. → decision value 0
   → decision value 1

Totally correct execution

faulty

non-faulty [ ]

$C \rightarrow C' \rightarrow C''$

admissible run $\longrightarrow$ all non-faulty nodes get a message eventually

One faulty

[totally correct = ① partially correct +

② Every admissible run is a deciding run]

Aim: Our protocol is not totally correct.

→ Lemma 1
→ Lemma 2
→ Lemma 3

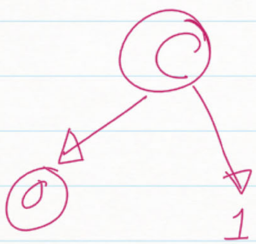
Lemma 1

$$\begin{tikzcd}
 & C \arrow[dl, "\sigma_1"'] \arrow[dr, "\sigma_2"] & \\
C_1 \arrow[dr, "\sigma_2"'] & & C_2 \arrow[dl, "\sigma_1"] \\
 & C_3 &
\end{tikzcd}$$

$P(\sigma_1) \cap P(\sigma_2) = \phi$

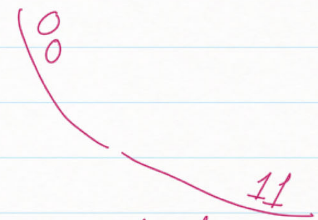
disjoint processes $\Rightarrow$ commutativity

Lemma 2: A bivalent initial configuration exists.

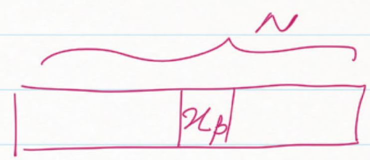


bivalent configuration

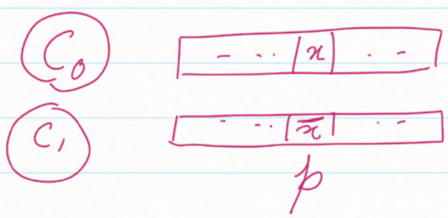
univalent configuration

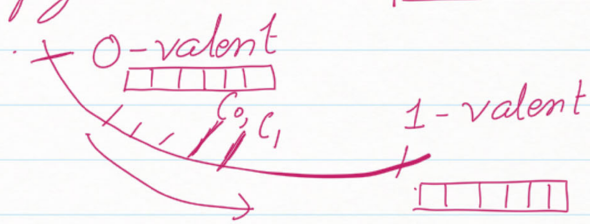


univalent



Initial configuration:





$\sigma(C_0)$

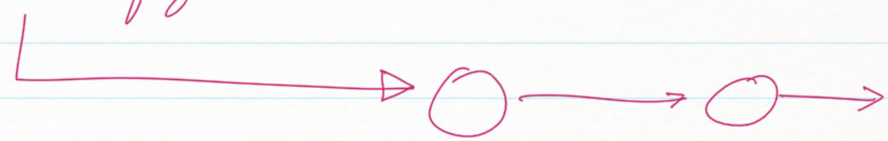
$$\begin{tikzcd}
C_0 \arrow[r, "\sigma"] & C_x \\
C_1 \arrow[ur, "\sigma"'] &
\end{tikzcd}$$

σ

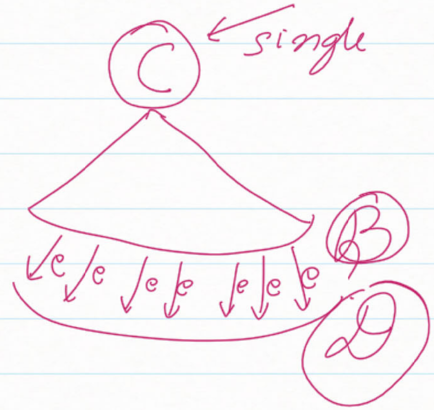
(p does not take any steps) 1-valent

<u>Contradiction</u>

Bivalent initial configuration (- . ..)



Lemma 3

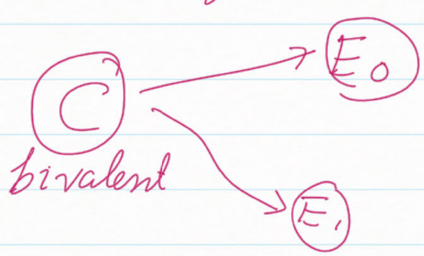


$\underline{e} = (p, m)$

$\mathcal{D}$ contains a bivalent configuration.

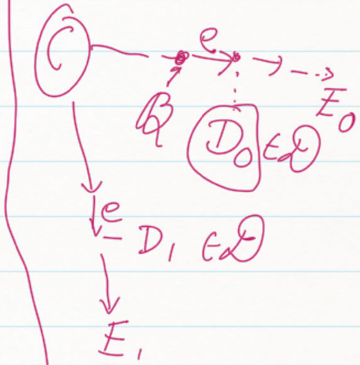
$\times\!\longrightarrow$ $\mathcal{D}$ contains only univalent configurations

$\boxed{\text{$\mathcal{D}$ contains \underline{both} 0-valent and 1-valent configurations}}$



$E_0 \in \mathcal{B}$
$\hookrightarrow_{e} D_0 \in \mathcal{D}$

$E_1 \in \mathcal{B}$
$\hookrightarrow_{e} D_1 \in \mathcal{D}$



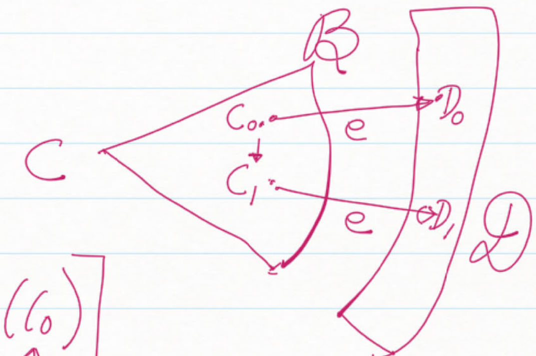
$D_0 \longleftrightarrow E_0$

$\underline{D_1 \longrightarrow E_1}$

$$C \xrightarrow{e} \mathcal{D}$$

$\boxed{0\text{-valent}}$ $\boxed{1\text{-valent}}$

$$\begin{array}{c} C \\ \downarrow \\ C_0,\ C_1 \in \mathcal{B} \\ e \swarrow \quad \downarrow e \\ D_0 \qquad D_1 \end{array}$$



$$[C_1 = e'(C_0)]$$

$$e = (p, m)$$

$$e' = (p', m')$$

Case 1: $p' \neq p$.

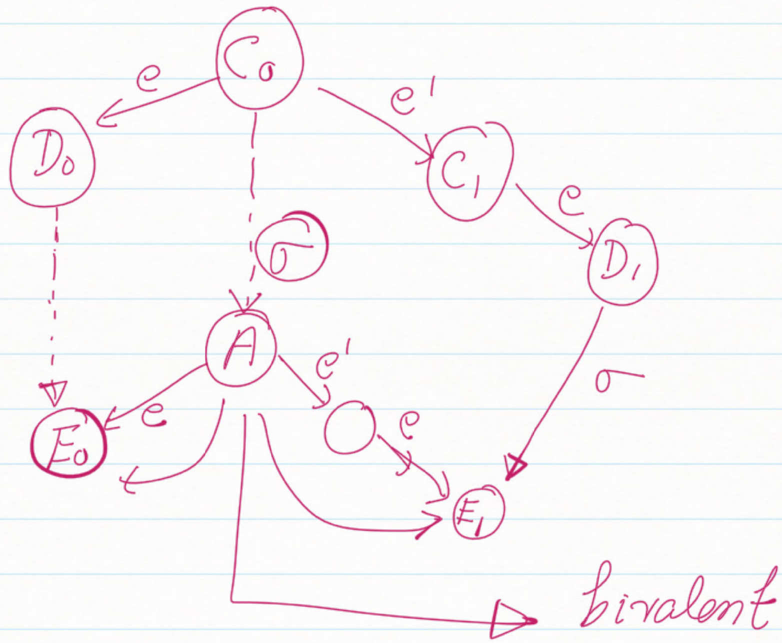
Case 2: $(p = p')$.

$$\left[\begin{tikzcd} C_0 \arrow[r, "e'"] \arrow[d, "e"'] & C_1 \arrow[d, "e"] \\ D_0 \arrow[r, "e'"] & D_1 \end{tikzcd}\right]$$

Contradiction ×

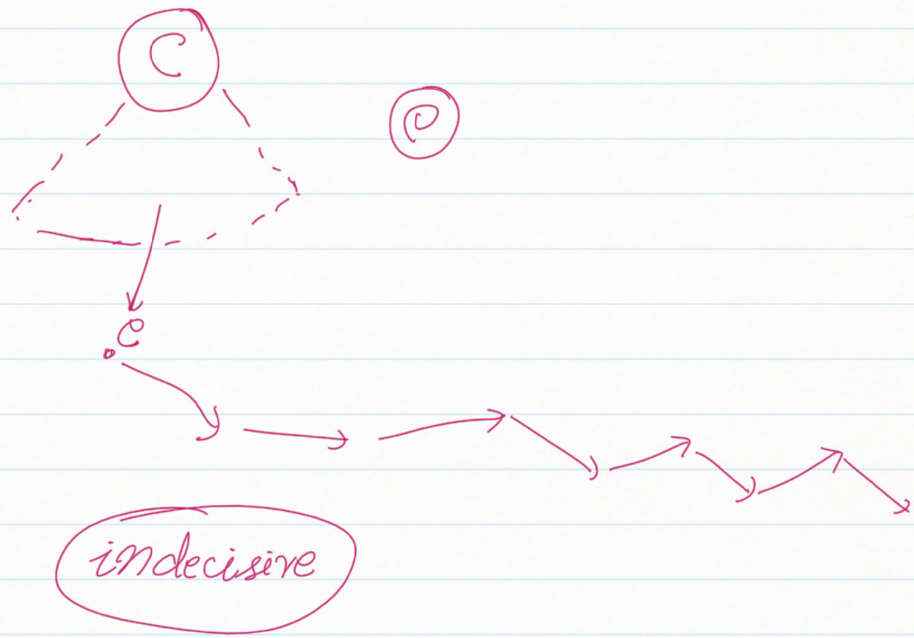
Case 2: $p = p'$

σ — $(\ldots)$
(p does not take any steps)



$\times$ Contradiction

$\mathcal{D}$ contains a bivalent configuration



Proves the theorem.

Distributed algorithm

consensus

FLP result

Paxos
Raft