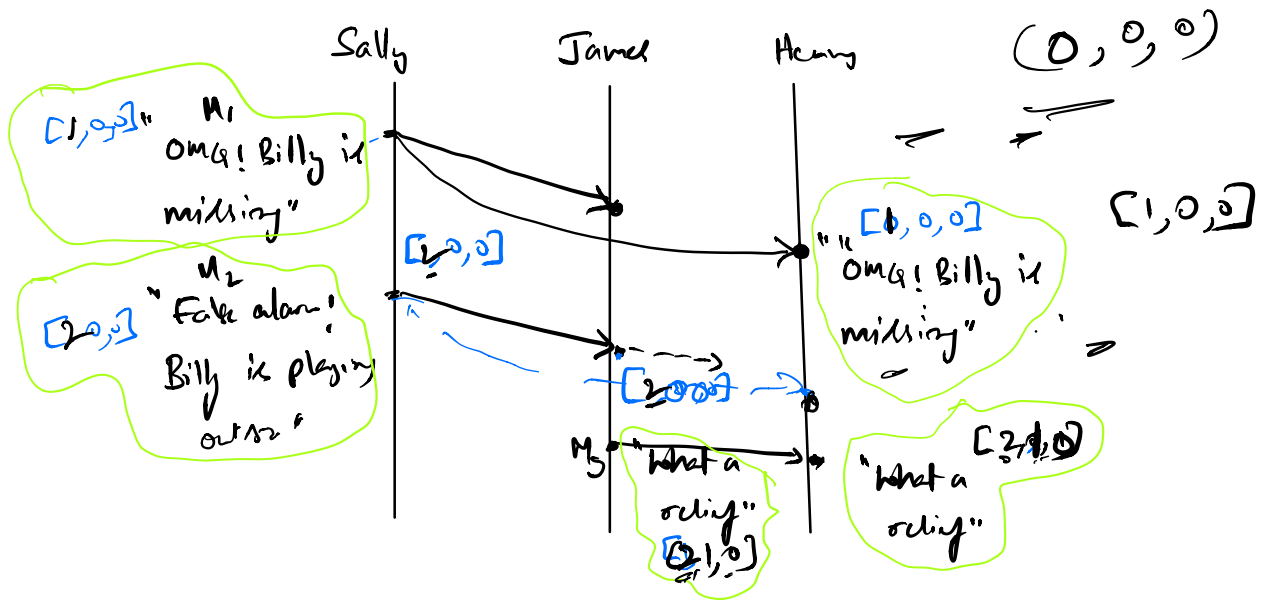


Welcome to CSCI 7000-001 Lec 4 (Jan 26)!

Recap:



Tag each message with a vector clock timestamp representing its dependencies

Today:

1. Distributed Consensus
2. FLP impossibility



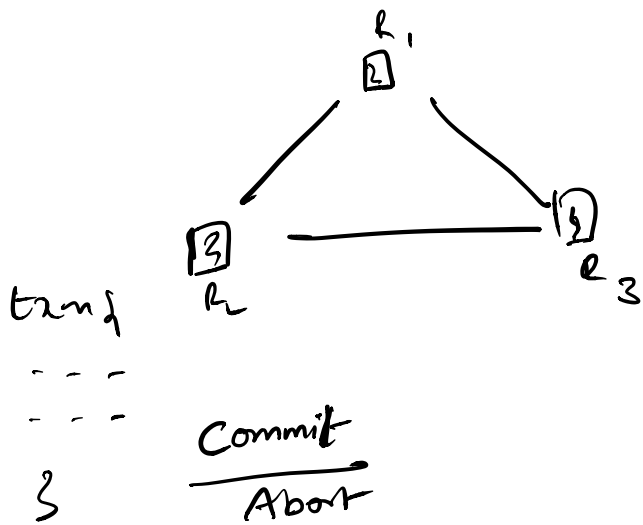
Fischer



Lynch

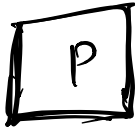


Paterson



Transaction
Commit;
Typical scenario for
Consensus

Consensus Program * Decision has to be
received eventually



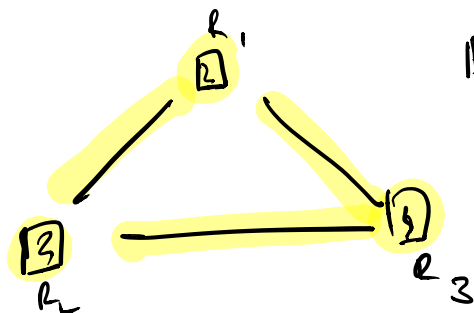
Distributed Consensus is
impossible with one
faulty process

FLP Theorem

* Every pair of nodes
has to agree on the
decision

* Decision value has to
be a proposed value
Commit / Abort

→ Model for Distributed Systems



$P = \{r_1, r_2, r_3\}$

Control
Communication

$R_1 \quad R_2 \quad R_3$

- R_1 has sent a $\overbrace{\text{message}}^m$ to R_2
- R_2 has consumed the message m .

→ Relation : $P \times P \times M$ $\{(R_1, R_2, m)\}$
 (R_1, R_2, m)

$(\underbrace{\langle R_1, R_2, R_3 \rangle}_P, \underbrace{\langle \rangle}_{\text{msg buffer}}) \xrightarrow{\text{Transition relation}} (\langle R_1, R_2, R_3 \rangle, \langle CR_2, m \rangle)$

Relation : Configuration \rightarrow Configuration
 $(P \times \text{msg+buffer})$

$P \quad R_1, R_2, R_3$

C/A
 0 1

x_p : P 's proposal

y_p : P 's decision

inf output
 (z_p, y_p, PC_p, ST_p)

(X, Y, PC, ST)

- $X : P \rightarrow \{0, 1\}$
- $Y : P \rightarrow \{0, 1, b\}$
- $PC : P \rightarrow \mathbb{N}$
- $ST : P \rightarrow _$

$$\begin{aligned}
 & \left(\overbrace{[P_1 \rightarrow 0; P_2 \rightarrow 2; P_3 \rightarrow 1]}^x, \right. \\
 & \vee [P_1 \rightarrow b, P_2 \rightarrow b, P_3 \rightarrow b], \\
 & [P_1 \rightarrow 2, P_2 \rightarrow 4, P_3 \rightarrow 8] \\
 & \left. [P_1 \rightarrow 2^s, P_2 \rightarrow 2^s, P_3 \rightarrow 2^s] \right) \quad m_1, m_2, m_3
 \end{aligned}$$

$$\left\{ (C, X, Y, PC, ST), \underbrace{\text{msg-buffer}}_{\text{Network}} \right\} \xrightarrow{P \times \text{msg}} \underline{2^{P \times \text{msg}}}$$

$1, 2, 3 \in \mathbb{N}$: Set of networks

$2^1, 2^2, 2^3 \in 2^{\mathbb{N}}$: Set of all sets of networks

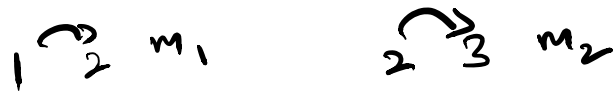
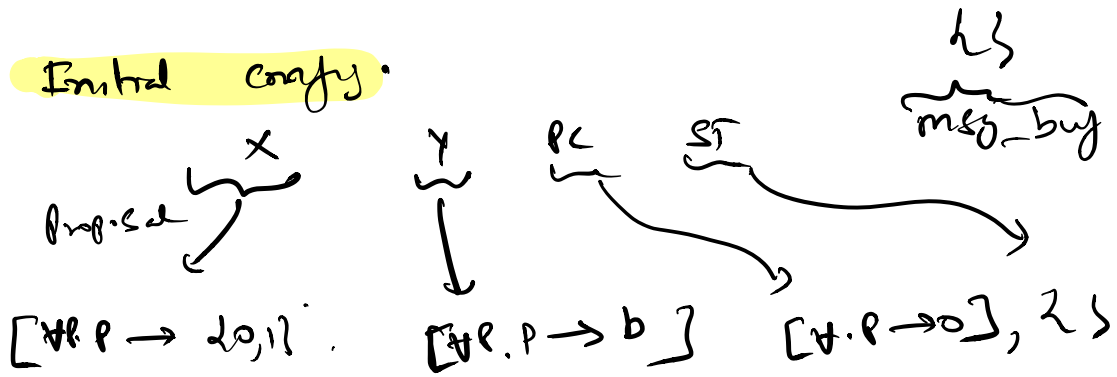
→ Config of the system at any given point.

$T : \text{Config} \rightarrow \text{Config}$

Step: at process P

- 1) Read a msg from msg-buffer \emptyset
- 2) do local computation
- 3) Send finite # of msgs to other processes.

Initial config.



$$\{ C_2, m_1, C_3, m_2 \}$$

- Step: at process P
- $C(P, m)$
- 1) Read a msg from msg_buy \emptyset
 - 2) do local computation
 - 3) Send finite # of msgs to other processes.

$$C_0 \xrightarrow{e=C(P, m)} C_1 \xrightarrow{e'=(P', m')} C_2 \dots$$

$$C_1 = e(C_0)$$

$$\rightarrow C_0^{init} (1,0) [1 \rightarrow 0, 2 \rightarrow 1, 3 \rightarrow 0, \dots]$$


$$C_1^{init} (1,0) [1 \rightarrow 1, 2 \rightarrow 0, 3 \rightarrow 0, \dots]$$

→ $2^{|P|}$ |P|-bit identifier for each initial conf.

Note 1: Each initial configuration is identified by a |P|-bit number

Note 2: $e = (p, m)$

msg - buffer = $\{ \dots (p, m), \dots \}$



e is applicable to some configuration c_0

$e = (p, m) \in c_0 \cdot \text{msg_buf}$

e applicable
 $c_0 \xrightarrow{e_1} c_1 \xrightarrow{e_2} c_2 \rightarrow \dots \rightarrow c_n$

Is e applicable to c_n ? Yes.

Note 2:

If an event e is applicable to config c

then e is applicable to every successor g of c

Lemma 1: Commutativity of Schedules

Schedule (σ) is a sequence of events

$$\sigma: \sigma = e_1 \cdot e_2 \dots e_n$$

$$\sigma(C_0) = e_1(e_2(\dots(e_n(C_0))))$$

$$C_0 \xrightarrow{e_1} C_1 \xrightarrow{\dots} \xrightarrow{e_n} C_n$$

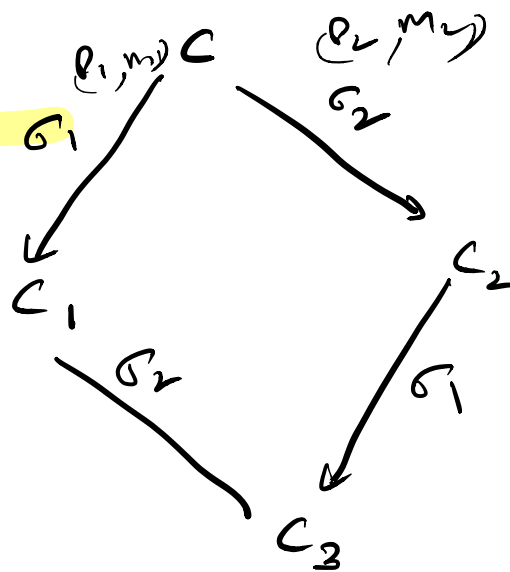
$$C_0 \xrightarrow{\sigma} C_n$$

$$(p_1, m_1) (p_2, m_2) \dots (p_n, m_n)$$

$$\sigma = e_1 \cdot e_2 \dots e_n$$

$$\text{dom}(\sigma) = \{p_1, p_2, \dots, p_n\}$$

$$\text{dom}(\sigma_1) \cap \text{dom}(\sigma_2) = \emptyset$$



$$\sigma_1 \cdot \sigma_2 (C)$$

$$= \sigma_2 \cdot \sigma_1 (C)$$