Lectures 4 & 5: The Asynchronous Model and the FLP Impossibility Theorem

(Lecture Series on Foundations of Blockchains)

Relaxing the Synchronous Assumption

Synchronous model: (i) shared global clock (common notion of time);
(ii) every msg sent at time t arrives by time $t+1$.

↳ not OK for modeling the Internet (which has outages, denial-of-service (DoS) attacks, etc.)

Note: if there is a known bound Δ on the maximum message delay, still in the synchronous model.

(can port over old protocols by inflating time steps)

(similarly if a priori known bound Φ on max difference between nodes' clocks)

Unsatisfying: (i) didn't force us to deal with unexpected outages/attacks (ii) produces stupid protocols in which nodes mostly idle

The Asynchronous Model

Note: 2 adversaries:
- Byz. nodes
- adversarial msg delivery

The Basic Idea: (i) no shared clock; (ii) no bound on the max msg delay.

(minimal assumption: every msg arrives, eventually)

Precise Model:

- pool M of outstanding (not-yet-delivered) messages
- while (TRUE):
  - one message (r, m) is delivered (to recipient r) (think of as chosen by "adversary" who wants to break a consensus protocol)
  - r can add any number of messages to M

(recipient, message content)

Assume: (1) M initialized to $\{(i, \perp)\}_{i=1}^{n}$ ← dummy msgs to get started

(2) every msg eventually delivered (Otherwise trivial impossibility via starvation)

Byzantine Agreement

Protocol: specifies what messages a node sends upon receipt of a new message (as a function of what the node knows — its private input + messages it has seen thus far)

The Problem: Byzantine agreement.

Setup: every node i has a private input $v_i^* \in V_i$. (for FLP, can even take $V = \{0,1\}$)

1. **termination**: each node i eventually halts, with some output v_i.
2. **agreement**: all honest nodes output same v_i.
3. **validity**: if all honest nodes have same input v^*, all honest v_i's equal v^*.

The FLP Impossibility Theorem

Theorem: [Fischer-Lynch-Paterson 1985] For every $n \geq 2$, even for $f = 1$, no deterministic protocol for Byzantine agreement satisfies termination, agreement, and validity in the asynchronous model.

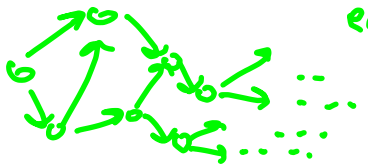
Note: formally seperates what is possible in synchronous, asynchronous settings.

Comment #1: holds even if restrict adversary to one crash fault!

Comment #2: implies same impossibility for Byzantine broadcast + SMR.

Workarounds: - allow randomized protocols (nodes can flip coins)

- use less extreme version of asynchrony (see Lecture 6, partially synchronous model)



vertices = configurations
edges = transitions / message deliveries

Configurations

Π = an allegedly correct deterministic BA protocol

Definition: a configuration C records the state of every node (private input + sequence of msgs received thus far) and the message pool M.

(note delivery of a msg (r, m) causes state transition $(\to C')$)

Definition: 0-configuration = no matter what adversary does from here, all honest nodes output 0. (Similarly, 1-configuration.)

ambiguous configuration: adversary has options for forcing all-0s, forcing all-1s.

Proof plan: show there exists an infinite sequence $C \to C_1 \to C_2 \to \cdots$ of ambiguous configurations. (i.e., might run forever, contradicting termination)

given by Lemma 1

given by Lemma 2

Π = allegedly correct protocol

An Initial Ambiguous Configuration

Lemma: There is a choice of private inputs s.t. the initial configuration C_0 is ambiguous.

Proof: Let X_i = initial configuration when private inputs of first i nodes are 1, last $n-i$ nodes are 0:

Note: (i) X_0 is a 0-configuration.
(ii) X_n is a 1-configuration.
} because Π satisfies validity!

Claim: One of the X_i's is ambiguous.

Completing the Contradiction

Claim: one of the X_i's is ambiguous.

Proof of Claim: (implies Lemma 1)

let i be such that (must exist!) (i) X_{i-1} a 0-configuration; (ii) X_i not a 0-config.

To finish proof: will argue that X_i must be ambiguous (i.e., can't be a 1-config.)

1. adversary has a strategy that forces all-1s output (since X_i not a 0-config.)
2. suppose node i Byzantine, adversary executes Π as if an honest node with private input 0

$\Rightarrow$ Π proceeds exactly as if started in X_{i-1} + all nodes honest

$\Rightarrow$ all honest nodes output 0 (because X_{i-1} a 0-configuration + honesty is one possible adversary strategy)

$\Rightarrow$ X_i is ambiguous

QED!

Extending the Ambiguous Sequence

Lemma 2: Let C_i be ambiguous, and (r, m) a message in C_i's msg pool. Then, there exists a sequence of message deliveries such that:

(i) last step = delivery of (r, m);

(ii) leads to an ambiguous configuration.

Lemma 1 + Lemma 2 ➔ FLP Theorem

Let C_0 = ambiguous configuration promised by Lemma 1.

To define C_{i+1} from C_i:

- let (r, m) = oldest message in C_i's pool
- let C_{i+1} = result of Lemma 2

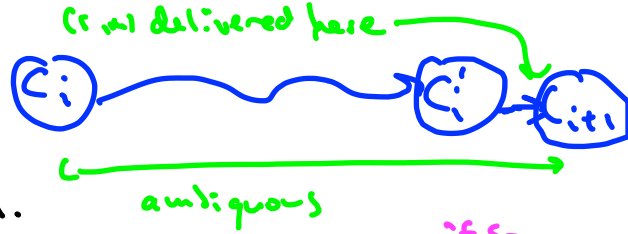
⇒ all of $C_0, C_1, C_2, C_3, \ldots$ ambiguous (by Lem 2)

(along with all of the intermediate configurations)

Note: if (r, m) added to pool at or before configuration C_i

⇒ delivered at or before $C_{i+|m|}$ → size of msg pool immediately after (r, m) added to it

Proof Setup



Fix ambiguous configuration C_i, msg (r,m) in C_i's pool.

Easy case: delivering (r,m) now (at C_i) yields an ambiguous configuration. ← if so, done!

⇒ assume delivering (r,m) now yields a 0-configuration C (say) ← analogous proof works for a 1-configuration

Definition: a configuration C is 0^* if: ← defined wrt. the protocol Π, msg (r,m)

(i) C reachable from C_i without delivering (r,m)

(ii) if (r,m) delivered at C, transition to a 0-configuration

⇒ define 1^* and ambiguous* configurations analogously

Note: Lemma 2's conclusion ⇔ there exists an ambiguous* configuration C_i'.

Note: assumption ⊛ ⇔ C_1 is a 0^*-configuration.

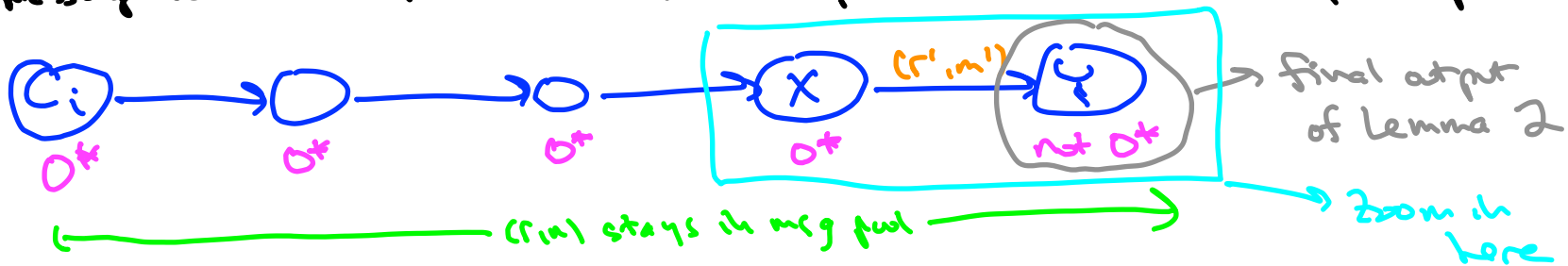
Hunting for a Non-0*-Configuration

Goal: find an ambiguous*-configuration. **Assumption**: C_i is a 0*-configuration.

Note: must exist at least 1*- or ambiguous*-configuration.
- if all reachable configurations were 0*, C_i would be a 0-config, not an ambiguous config.

adversary must eventually deliver (r,m)

Notation: Let Y = non-0* configuration closest to C_i (i.e., fewest # of message deliveries). Let X be Y's predessor on shortest $C_i \to Y$ path.



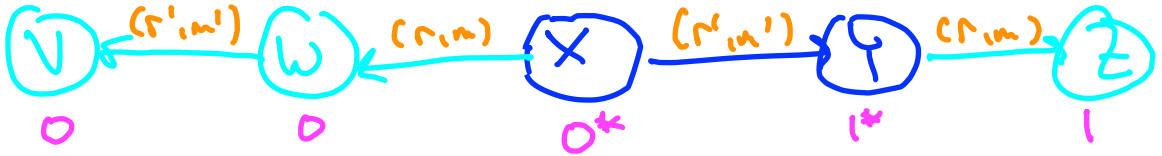
Claim: Y cannot be a 1*-configuration (so must be ambiguous*).

Completing the Contradiction

((r,m) is msg pulls of X & Y)

Proof of Claim:

Assume (for contradiction) Y is a 1*-config.



Point: (i) msg order $(r,m),(r',m') \longmapsto$ 0-config (V) (*)
(ii) msg order $(r',m'),(r,m) \longmapsto$ 1-config (Z) (**)

Case 1: $r \neq r'$. All nodes see same sequence of received msgs in (i), (ii)
⇒ all (honest) nodes behave identically (+ halt w/ same output) in (i), (ii) [contradiction]

Case 2: $r = r'$. Suppose r is Byzantine.

- Scenario #1: r receives m before m'
- Scenario #2: r receives m' before m

indistinguishable to all honest nodes (see identical sequences of messages), same output either way

(*) (⇒ final outputs = 0), (**) (⇒ final outputs = 1)

r follows Π honestly

r acts as if received m before m' + is then following Π honestly.

Contradiction!