Impossibility of Distributed Consensus with One Faulty Process

Michael J. Fisher, *Yale University*
Nancy A. Lynch, *MIT*
Michael S. Paterson, *Warwick University*

November 4, 2014

Presented by: Theodoros Gkountouvas

Consensus Protocols

Goal of the Consensus protocols:

- Safety
  - (i) All process should decide the same value.
  - (ii) Processes do not decide an initial fixed value. Thus, there are should be runs of the protocol that decide different values.
- Liveness

  The protocol should always make a decision.

Assumptions

- Asynchronous Network:
  Messages can take arbitrarily long to arrive.
- Reliable Network:
  Messages are neither lost nor duplicated.
- Failures:
  There can be at most one crash failure amongst the processes.

Consensus State Machine Model

- State Machine Model:
  - States $(x_p, y_p) \in \{0, 1, b\}^2$
  - Initial States should restrict y_p to b.
  - You should consider y_b as a write-once variable.
  - Transitions from state C_i to C_{i+1} according to the event processed.
- Network
  - Messages are modelled as $e = (p, m)$.

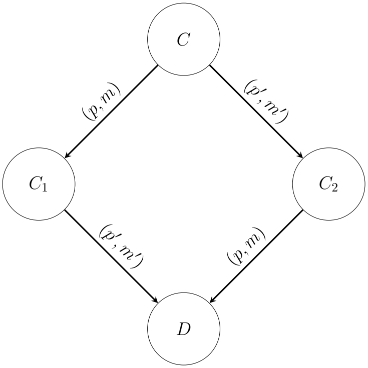
Definitions

Let C be a state.

- C is **bivalent** if there exists run s_0 from C that decides 0 and run s_1 from C that decides 1.
- C is **univalent** if all the runs of the protocol decide only one value.
  - 0-valent if they decide 0.
  - 1-valent if they decide 1.

Commutativity

Let us suppose $p \neq p'$.



Main Theorem

Theorem

No consensus protocol is totally correct in spite of one fault.

More specifically, it is proved that there can be infinite runs of any correct consensus protocol.

Bivalent Initial State

Lemma

Every correct consensus protocol P has an initial bivalent state.

Proof.

Let us suppose this is not true.

1. P should have both 0-valent and 1-valent initial states.
2. There exist two initial states C_0 (0-valent) and C_1 (1-valent) s.t. they differ only in the state of one process p (x_p).
3. Suppose p fails from the beginning and thus, C_0 and C_1 are indistinguishable for protocol P.
4. On the same run s of the protocol they decide the same value (contradiction).

□

Bivalent Intermediate State

- C is an initial bivalent state
- $e = (p, m)$ is an arbitrary event that is applicable to C
- $\mathbb{K}$ is the set of states reachable by C without applying e
- $\mathbb{L}$ be the set of states that are produced after applying e to all the states in $\mathbb{K}$.

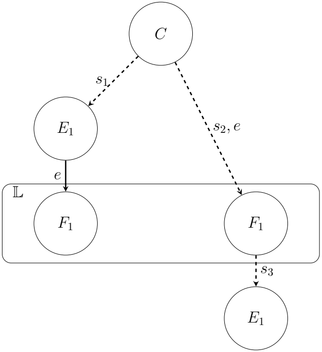
Lemma

$\mathbb{L}$ *contains a bivalent state*

Let us assume that all the states in $\mathbb{K}$ are univalent.

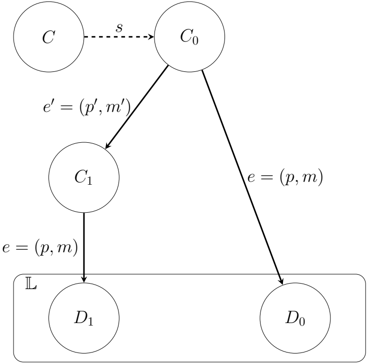
0-valent Assumption

1. Let us assume that $\mathbb{L}$ contains only 0-valent states.
2. Then since C is bivalent there exists a reachable state E_1 which is 1-valent.



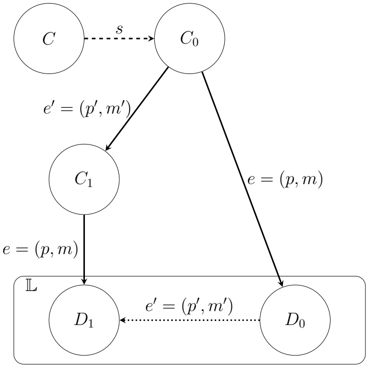
Univalent Assumption

1. There exist both 0-valent and 1-valent states in $\mathbb{L}$.
2. There exists reachable state C_0 from C such that:



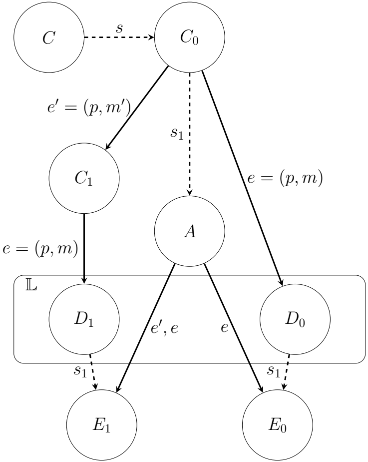
Univalent Assumption

Case 1: $p \neq p'$



Univalent Assumption

Case 2: $p = p'$



Proof of Theorem

Lemma

Every correct consensus protocol P has an initial bivalent state.

Lemma

For any event e the corresponding $\mathbb{L}$ *contains a bivalent state.*

$$\Downarrow$$

Theorem

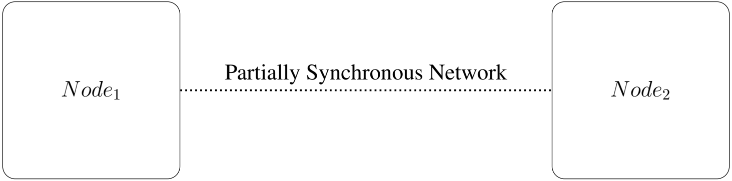
No consensus protocol is totally correct in spite of one fault.

Discussion

- Are the assumptions reasonable?
- Is this really an impossibility result?
  - Paxos
  - Virtual Synchrony
- How possible is the scenario where the system does not reach consensus?
- What is the minimum relaxation we can do in order to make consensus possible?

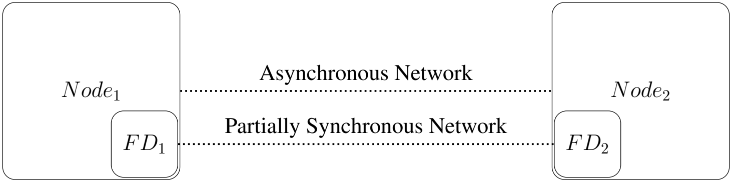
Partially Synchronous Network

- **Problem:** Consensus is impossible because of the asynchronous network!!!
- **Solution:** Relax the assumptions about the asynchronous network.



Failure Detectors

- **Problem:** Consensus is impossible because we cannot separate a faulty process from a slow one!!!
- **Solution:** Assume that there exists a failure detector that is not limited by the asynchronous environment.



Weakest Failure Detector

The weakest failure detector W for which we can achieve Consensus has the following properties:

i) There is a time after which every process that crashes is always suspected by some correct process.

ii) There is a time after which some correct process is never suspected by any correct process.

Another Failure Detector

Another failure detector B for which we can achieve Consensus has the following properties:

i) There is a time after which every process that crashes is always suspected by all correct processes (stronger).

ii) There is a time after which some correct process is never suspected by a majority of the processes (weaker).

Actually, B can be transformed into W, if the majority of processes is non-faulty. Thus, B is at least as strong as W.

Main Results

- Every failure detector B that can be used in order to achieve Consensus can be reduced to W.
- Therefore, W is indeed the weakest failure detector that can be used to solve Consensus in asynchronous systems with $n > 2f$.
- Furthermore, if $n \leq 2f$, any failure detector that can be used to solve Consensus must be strictly stronger than W.

Questions