Early Stopping: The Protocol

Let $\text{faulty}(p, k)$ be the set of processes that have failed to send a message to $p$ in any round $1, \ldots, k$.

1. if $p = \text{sender}$ then value := $m$, else value := ?

Process $p$ in round $k, 1 \leq k \leq f + 1$
2. send value to all
3. if delivered in round $k-1$ then halt
4. receive round $k$ values from all
5. $\text{faulty}(p, k) := \text{faulty}(p, k - 1) \cup \{ q \mid p \text{ received no value from } q \text{ in round } k \}$
6. if received value $v \neq ?$ then
7. $\text{value} := v$
8. deliver value
9. if $p = \text{sender}$ then value := ?
10. else if $k = f + 1$ or $|\text{faulty}(p, k)| < k$ then
11. $\text{value} := \text{SF}$
12. deliver value
13. if $k = f + 1$ then halt

Termination

Let $\text{faulty}(p, k)$ be the set of processes that have failed to send a message to $p$ in any round $1, \ldots, k$.

1. if $p = \text{sender}$ then value := $m$, else value := ?

Process $p$ in round $k, 1 \leq k \leq f + 1$
2. send value to all
3. if delivered in round $k-1$ then halt
4. receive round $k$ values from all
5. $\text{faulty}(p, k) := \text{faulty}(p, k - 1) \cup \{ q \mid p \text{ received no value from } q \text{ in round } k \}$
6. if received value $v \neq ?$ then
7. $\text{value} := v$
8. deliver value
9. if $p = \text{sender}$ then value := ?
10. else if $k = f + 1$ or $|\text{faulty}(p, k)| < k$ then
11. $\text{value} := \text{SF}$
12. deliver value
13. if $k = f + 1$ then halt

Validity

If in any round a process receives a value, then it delivers the value in that round.

If a process has received only “?” for $f+1$ rounds, then it delivers SF in round $f+1$.

Termination

If in any round a process receives a value, then it delivers the value in that round.

If a process has received only “?” for $f+1$ rounds, then it delivers SF in round $f+1$.
Validity

If the sender is correct then it sends m to all in round 1.

By Validity of the underlying send and receive, every correct process will receive m by the end of round 1.

By the protocol, every correct process will deliver m by the end of round 1.

Agreement - 1

For any \( r \geq 1 \), if a process \( p \) delivers \( m = \text{SF} \) in round \( r \), then there exists a sequence of processes \( p_0, p_1, \ldots, p_r \) such that \( p_0 = \text{sender} \), \( p_r = p \), and in each round \( k, 1 \leq k \leq r \), \( p_{k-1} \) delivers m and \( p_k \) receives it. Furthermore, all processes in the sequence are distinct; unless \( r = 1 \) and \( p_0 = p_1 = \text{sender} \).

Lemma 1:

For any \( r \geq 1 \), if a process \( p \) sets value to SF in round \( r \), then there exist some \( j \leq r \) and a sequence of distinct processes \( q_1, q_2, \ldots, q_j = p \) such that \( q_j \) only receives \( m \) in rounds \( 1 \) to \( j \), \( |\text{faulty}(q_j)| < j \), and in each round \( k, j+1 \leq k \leq r \), \( q_{k-1} \) sends SF to \( q_k \) and \( q_k \) receives SF.

Agreement - 2

Let \( \text{faulty}(p, k) \) be the set of processes that have failed to send a message to \( p \) in any round \( 1, \ldots, k \).

1. If \( p = \text{sender} \) then value := \( m \) else value := ?

Process \( p \) in round \( k, 1 \leq k \leq f+1 \):

2. send value to all
3. if delivered in round \( k-1 \) then halt
4. receive round \( k \) values from all
5. \( \text{faulty}(p, k) := \text{faulty}(p, k-1) \cup \{q \} \)
   received no value from \( q \) in round \( k \)
6. if received value \( \neq \# \) then
   value := ?
7. value := ?
8. deliver value
9. if \( p = \text{sender} \) then value := ?
10. else if \( k = f+1 \) or \( |\text{faulty}(p, k)| < k \) then
11. value := SF
12. deliver value
13. if \( k = f+1 \) then halt

Lemma 3:

It is impossible for \( p \) and \( q \), not necessarily correct or distinct, to set value in the same round \( r \) to \( m \) and SF, respectively.
**Agreement - 3**

Let \( \text{faulty}(p, k) \) be the set of processes that have failed to send a message to \( p \) in any round \( 1, \ldots, k \).

1: if \( p = \text{sender} \) then value := \( m \) else value := ?

**Process** \( p \) in round \( k \), \( 1 \leq k \leq f+1 \)

2: send value to all
3: if delivered in round \( k-1 \) then halt
4: receive round \( k \) values from all
5: \( \text{faulty}(p, k) = \text{faulty}(p, k-1) \cup \{q\} \) if \( p \) received no value from \( q \) in round \( k \)
6: if received value \( \neq \# \) then
7: value := ?
8: deliver value
9: if \( p = \text{sender} \) then value := ?
10: else if \( k = f + 1 \) or \( |\text{faulty}(p, k)| < k \) then
11: value := SF
12: deliver value
13: if \( k = f + 1 \) then halt

**Integrity**

Let \( \text{faulty}(p, k) \) be the set of processes that have failed to send a message to \( p \) in any round \( 1, \ldots, k \).

1: if \( p = \text{sender} \) then value := \( m \) else value := ?

**Process** \( p \) in round \( k \), \( 1 \leq k \leq f+1 \)

2: send value to all
3: if delivered in round \( k-1 \) then halt
4: receive round \( k \) values from all
5: \( \text{faulty}(p, k) = \text{faulty}(p, k-1) \cup \{q\} \) if \( p \) received no value from \( q \) in round \( k \)
6: if received value \( \neq \# \) then
7: value := ?
8: deliver value
9: if \( p = \text{sender} \) then value := ?
10: else if \( k = f + 1 \) or \( |\text{faulty}(p, k)| < k \) then
11: value := SF
12: deliver value
13: if \( k = f + 1 \) then halt

**Proof**

If no correct process ever receives \( m \), then every correct process delivers SF in round \( f+1 \).

**Agreement - 3**

Let \( \text{faulty}(p, k) \) be the set of processes that have failed to send a message to \( p \) in any round \( 1, \ldots, k \).

1: if \( p = \text{sender} \) then value := \( m \) else value := ?

**Process** \( p \) in round \( k \), \( 1 \leq k \leq f+1 \)

2: send value to all
3: if delivered in round \( k-1 \) then halt
4: receive round \( k \) values from all
5: \( \text{faulty}(p, k) = \text{faulty}(p, k-1) \cup \{q\} \) if \( p \) received no value from \( q \) in round \( k \)
6: if received value \( \neq \# \) then
7: value := ?
8: deliver value
9: if \( p = \text{sender} \) then value := ?
10: else if \( k = f + 1 \) or \( |\text{faulty}(p, k)| < k \) then
11: value := SF
12: deliver value
13: if \( k = f + 1 \) then halt

**Integrity**

Let \( \text{faulty}(p, k) \) be the set of processes that have failed to send a message to \( p \) in any round \( 1, \ldots, k \).

1: if \( p = \text{sender} \) then value := \( m \) else value := ?

**Process** \( p \) in round \( k \), \( 1 \leq k \leq f+1 \)

2: send value to all
3: if delivered in round \( k-1 \) then halt
4: receive round \( k \) values from all
5: \( \text{faulty}(p, k) = \text{faulty}(p, k-1) \cup \{q\} \) if \( p \) received no value from \( q \) in round \( k \)
6: if received value \( \neq \# \) then
7: value := ?
8: deliver value
9: if \( p = \text{sender} \) then value := ?
10: else if \( k = f + 1 \) or \( |\text{faulty}(p, k)| < k \) then
11: value := SF
12: deliver value
13: if \( k = f + 1 \) then halt

**Proof**

If no correct process ever receives \( m \), then every correct process delivers SF in round \( f+1 \).

**At most one**

- Failures are benign, and a process executes at most one deliver event before halting.

**If** \( m \neq \text{SF} \), **only if** \( m \) was broadcast

- From Lemma 1 in the proof of Agreement
A Lower Bound

**Theorem**

There is no algorithm that solves the consensus problem in fewer than $f+1$ rounds in the presence of $f$ crash failures, if $n \geq f + 2$

We consider a special case ($f = 1$) to study the proof technique

Views

Let $\alpha$ be an execution. The **view** of process $p_i$ in $\alpha$, denoted by $\alpha|p_i$, is the subsequence of computation and message receive events that occur in $p_i$ together with the state of $p_i$ in the initial configuration of $\alpha$

Views

Let $\alpha$ be an execution. The view of process $p_i$ in $\alpha$, denoted by $\alpha|p_i$, is the subsequence of computation and message receive events that occur in $p_i$ together with the state of $p_i$ in the initial configuration of $\alpha$

Similarity

**Definition** Let $\alpha_1$ and $\alpha_2$ be two executions of consensus and let $p_i$ be a correct process in both $\alpha_1$ and $\alpha_2$.

$\alpha_1$ is similar to $\alpha_2$ with respect to $p_i$, denoted $\alpha_1 \sim_{p_i} \alpha_2$, if $\alpha_1|p_i = \alpha_2|p_i$
Similarity

Definition Let $\alpha_1$ and $\alpha_2$ be two executions of consensus and let $p_i$ be a correct process in both $\alpha_1$ and $\alpha_2$.

$\alpha_1$ is similar to $\alpha_2$ with respect to $p_i$, denoted $\alpha_1 \sim_p \alpha_2$ if $\alpha_1[p_i] = \alpha_2[p_i]$.

Note: If $\alpha_1 \sim_p \alpha_2$ then $p_i$ decides the same value in both executions.

Lemma: If $\alpha_1 \sim_p \alpha_2$ and $p_i$ is correct, then $\text{dec}(\alpha_1) = \text{dec}(\alpha_2)$.

The transitive closure of $\alpha_1 \sim_p \alpha_2$ is denoted $\alpha_1 \approx \alpha_2$.

We say that $\alpha_1 \approx \alpha_2$ if there exist executions $\beta_1, \beta_2, \ldots, \beta_{k+1}$ such that $\alpha_1 = \beta_1 \sim_{p_i} \beta_2 \sim_{p_i} \ldots \sim_{p_i} \beta_{k+1} = \alpha_2$.

Note: If $\alpha_1 \approx \alpha_2$ then $p_i$ decides the same value in both executions.

Lemma: If $\alpha_1 \approx \alpha_2$ and $p_i$ is correct, then $\text{dec}(\alpha_1) = \text{dec}(\alpha_2)$.
Single-Failure Case

There is no algorithm that solves consensus in fewer than two rounds in the presence of one crash failure, if $n \geq 3$

The Idea

By contradiction

- Consider a one-round execution in which each process proposes 0. What is the decision value?
- Consider another one-round execution in which each process proposes 1. What is the decision value?
- Show that there is a chain of similar executions that relate the two executions.

So what?

Adjacent $\alpha^i$’s are similar!

Starting from $\alpha^i$, we build a set of executions $\alpha^j$ where $0 \leq j \leq n-1$ as follows:

$\alpha^j$ is obtained from $\alpha^i$ after removing the messages that $p_i$ sends to the $j$-th highest numbered processors (excluding itself)
The executions

Indistinguishability

Indistinguishability

Indistinguishability
Indistinguishability
Indistinguishability

\[ p_0 \quad 1 \]

\[ \cdots \]

\[ p_{i-1} \quad 1 \]

\[ p_i \quad 0 \]

\[ \vdots \]

\[ p_{n-1} \quad 0 \]

\[ \alpha_i \]

\[ \approx \]

\[ \beta_0 \]

\[ \alpha_{n-1} \]

\[ \approx \]

\[ \alpha_{i+1} \]