Consensus for Byzantine failures

- Remember: Byzantine processors may lie...
- What goes wrong in the protocol for crash failures?
  - If \( |\sigma| = f + 1 \) then there is a non faulty processor \( z \) with \( \sigma = \alpha; z; \beta \).
  - Then at round \( |\sigma| + 1 \) processor \( z \) sent \( v = v_z \) to \( q \) as well (as the first processor in \( \beta \)). By construction \( v_z = v_{\alpha} \) as the processors in \( \beta \) forward this value to each other to finally deliver it to \( p_3 \). Again \( v \in V_2 \).

Byzantine failures: \( f < n/3 \) is necessary

- Suppose \( n = 3 \) and \( f = 1 \)
  - a and b must decide 1
  - b and c must decide 1
  - a and c must decide 1

- Suppose \( n = 3 \) and \( f = 1 \)
  - a and b must decide 0
  - b and c must decide 0
  - a and c must decide 0
  - a and b see the same messages, yet c decides 0 as \( v \) saw the same messages.
A protocol tolerating $f < n/3$ byzantine failures

- Again each processor $p$ builds the following tree $T_p$

![Diagram of a tree with levels labeled from 0 to $r+1$. Each level contains nodes with values, and arrows indicating parent-child relationships.]

- Lamport's OM protocol for building the tree

![Diagram illustrating the OM protocol with arrows and labels for message passing and value propagation.]

Byzantine failures: decision more complex

- Associate a decision value $d'_p$ to each node in the tree
  - After tree is filled with values top down, it is filled with decision values bottom up
  - $d'_p$ is the value for $C[p].decision$ that $p$ decides on

- Define $\text{Majority}(S)$ be the value that occurs most in a set $S$, using some constant $\bot$ to break ties

Lamport's OM protocol for building the tree

- We write $OM'_p(m, v)$ to make clear processor $p$ executes this to propagate $v$ and to keep track of "stack trace" $m$
  - $OM'_p(m, v)$ is executed by $p$ for all $\sigma$ s.t. $|\sigma| = f - m$ and $p \notin \sigma$
  - It sends $v = d'_p$ to all nodes $q$ (as message $m^*_{p,q}$) stored by $q$ as $v^*_{p,q}$, and instructs them to propagate the value through recursion
  - It essentially builds $p$'s part of the subtrees rooted at $v$ for all processors, together with the other $OM'_p(\cdot, \cdot)$ the whole subtrees rooted
  - The protocol starts with $OM'_p(\cdot, C[p].in)$ for all $p$
Lamport's OM protocol

- OM₀(0, 0)
- Send v₀ as m₀₀ to all q
- All processors q that receive it set v₀ = m₀₀; set ⊥ if no value received
- Set d₀ = Majority(v₀, q ∉ σ)

- OMₙₙₑ(ₙₙₑ, 0) for 0 ≤ n ≤ f
- Send v as m₀₀ to all q
- All processors q that receive it set v₀ = m₀₀; set ⊥ if no value received
- Trigger OM₀(ₙₙₐ, 4) for all q ∉ σ

Start as OM₀(₀, 0, 4) for all p
- Store v₀₀ in m₀ₐ

A protocol tolerating \( f < \frac{n}{3} \) byzantine failures

- Again each processor \( p \) builds the following tree \( T_p \)

One step in detail
So building the tree is the same protocol as for crash failures.

- **Before round 1**
  - Initialise tree. Set all $v^0 = -1$ and $v^K = C(p)$ in

- **Round** $r, 1 \leq r \leq f + 1$
  - For all $\sigma$ with $|\sigma| = r - 1, p \notin \sigma$, send $v^K$ to all processors $q$ (including $p$)
    - Call this message $m^*_{q}$
    - Receive all $m^*_{q}$ addressed to $p$ and store in $v^0$ for
      - By the protocol $x \notin \sigma$ so $p$ receives $n - (r - 1)$ such messages from each $x$

Deciding on a value

- **Work from the leaves upwards**
  - $d^0 = v^0$ for $|\sigma| = f + 1$
  - $d^0 = \text{Majority}(v^0_{q} | q \notin x)$ otherwise
  - Node $p$ decides on $d^K$

Correctness

- **Lemma 1**: If $p, q, r$ are non faulty, then for all $\sigma$ we have $v^K_{p, q} = v^K_{r}$
  - If $r$ is non faulty, it sends the same value to $p$ and $q$

- Set $d^K = v^K$ for all leaves, ie $|\sigma| = f + 1$
**Correctness**

**Lemma 1:** If \( p \in r \) are non faulty, then for all \( n \) we have \( c^n_p = c^n_r \).

**Lemma 2:** Let \( v \) be arbitrary and let \( r \) be non faulty. Then there is a value \( v \) such that for all non faulty \( p \) we have \( d^n_r = d^n_p = v \).

- By induction on the length of \( p \); starting with the leaves (length \( f + 1 \))

- The base case follows from lemma 1 and the fact that for \( |p \cap r| = f + 1 \) we have \( c^n_p = c^n_r \).

- Now suppose \( 0 < |p \cap r| < f + 1 \). By lemma 1, all non faulty processors have \( d^n_p = v \). Then all non faulty processors \( p \neq r \) send \( v \) to all other processors \( q \). If non faulty, \( q \) sets \( c^n_q = v \).

- By the induction hypothesis we have \( c^n_{r_1} = c^n_{r_2} = v \) for all non faulty \( q \).

- The number of children of a node with label \( x \) is \( n - |p \cap r| \geq n - f - 2f / m \).

- Hence the majority of children is non faulty, and so \( d^n_r = \text{Majority}(c^n_{r_1}, c^n_{r_2} | q \neq q) = v \) as required.

---

**Lemma 2**

**Base case** \( |p \cap r| = f + 1 \)

**Induction** \( |p \cap r| = f + 1 < f + 1 \)

---

**Validity**

**Theorem:** If all non faulty processors have input \( v \) they decide on \( v \).

- If all non faulty processors have input \( v \), they send \( v \) to all other nodes in the first round. As a result \( c^0_p = v \) for all correct \( p \) and \( q \).

- By lemma 2, \( d^n_q = v \) for all correct \( p \) and \( q \); and hence \( d^n_r = \text{Majority}(d^n_q | \text{for all } q) = v \).
Agreement

Definition 1. \( \sigma \) is common if \( d_\sigma^p = d_\sigma^q \) for all pairs of non faulty \( p,q \).

Definition 2. A subset \( C \) of nodes in a tree \( T \) is a path cover of \( T \) if all paths from the leaves to the root visit at least one node in \( C \).

Definition 3. A path cover \( C \) is common if all nodes in \( C \) are common. (Note: this does not require \( d_\sigma^p = d_\sigma'^q \) for different \( \sigma, \sigma' \).)

Lemma 3. There exists a common path covering of the tree constructed by the consensus algorithm:

- All paths from the root to a leaf correspond to a label \( \sigma \) with length \( f = 2k \).
- Then \( \sigma = \sigma'; \sigma'' \) for some non faulty \( \sigma \).
- By lemma 2 \( d_\sigma^p = d_\sigma^q \) for all non faulty \( p,q \) and so \( \sigma; \sigma'' \) is common and on the path.

Lemma 4. Let \( \sigma \) be a node. If there is a common path covering of the subtree rooted at \( \sigma \), then \( \sigma \) is common itself.

- By induction on the length of \( \sigma \).
- For \( \sigma' = \sigma \) the lemma trivially follows.
- Let \( 0 \leq |\sigma'| < f = 2k + 1 \) and assume there is a common path covering \( C \) of the subtree rooted at \( \sigma \). If \( \sigma \in C \) we are done. If not, then the trees rooted in all children have a common path covering and by the induction hypothesis then all children \( \sigma ; \sigma' \) of \( \sigma \) are common.

Hence \( d_\sigma^p = d_\sigma^q \) for all pairs of non faulty \( p,q \). Hence \( d_\sigma^p = \text{Majority}(d_\sigma^p, \sigma \not\in \sigma) \) and hence \( \sigma \) is common as well.

Theorem: All non faulty nodes decide on the same value

- Follows from lemma 3 and 4.
Using authentication

Signing messages

- Every processor $p$ has a private signing key. The corresponding signature verification key is known to all processors.
- Let us write $[m]_p$ for a message $m$ signed by $p$. Write $[m]_0$ for $[...[m]_0,...]$ with $s = p;...;r$
- Processors reject any messages with incorrect signatures.
  - Byzantine nodes cannot forge values pretending they heard another value from a correct processor
  - But they can send conflicting initial values in the first round!
- Now consider the weak broadcast protocol

(Binary) Broadcast (aka agreement)

- Sender $p$ in round 1
  - If $c(p) = 1$ then send $[1]_p$ to all, otherwise stay silent
  - Decide on $c(p)$
- Other nodes $q$
  - For each round $r \in (1,...,f+1)$
    - If you receive a valid $[1]_r$ message (note $s = r$) with $s = p;...;r$ then send $[1]_r$ to all, decide on 1 and terminate
    - Decide on 0 and terminate
Correctness

- **Agreement**
  - Suppose a correct node decides on 1 in round $r$. This means it received a valid [1] message. If $r < f + 1$ then $p$ sends a valid $[0]_{q} = [1]_{q}$ message to all correct $q$ who therefore decide on 1 too. If $r = f + 1$ then $p$ decides 1 hence $s = c'_v$ for some correct $q$ that sent a valid $[1]_{q}$ message to all correct nodes that therefore decided on 1 in round $r = f + 1$.

- **Validity**
  - Suppose $p$ is correct. Either it sends $[1]_{q}$ to all, and all correct nodes decide 1 in round 1. Or it does not send anything. As a result no correct node receives a valid [1] message, so all correct nodes decide 0 in round $f + 1$.

Reaching consensus

- Each node uses the broadcast algorithm to send its input value to all other nodes.
- All other nodes obtain (by the agreement property of the broadcast) the same vector of input values.
- All nodes decide on the majority of values in this vector (breaking ties in a deterministic way).
- If $f < n/2$ then if all nodes have the same input value, all nodes decide on this value.