Advanced Network Security 2017

Agreement and consensus II — Byzantine failures
Q2: How many messages does the crash-tolerant consensus algorithm exchange?

- n processors
- f failures (at most)
- r: is the current round

Every processor $p$, in round $r$, sends for all $1 \leq \ell = r-1$, $p \neq \sigma$, it sends $v^p_r$ to all other $q$

How many such $\sigma$ are there

$- (n-1)(n-2) \ldots \frac{(n-1)!}{(n-r)!} = \frac{(n-1)!}{(n-r)!}$

So in round $r$, $n \cdot \frac{(n-1)!}{(n-r)!} \rightarrow$ for one processor

In total all processors $n \cdot \sum_{r=1}^{f+1} \frac{(n-1)!}{(n-r)!}$
Q2: (continued): Can we optimize this?

\[ T_p \rightarrow V_p = \{ v | v \in T_p \} \]

Instead of always "forwarding" all values you receive, only forward a new value!

\[ 2n^2 \]

You only send the value 0 once to all, and the value 1 once to all.
Byzantine failures

Would the protocol for crash failures still work?

Well:

$$|\sigma| = f + 1$$

If $$v \in T_p = v \in T_q$$

to all other nodes, including $$v$$

can be faulty!!

with byzantine failures,

nodes can lie $$\Rightarrow v_{101}^p \neq v$$
Q: can we think of a protocol that tolerates arbitrary
number of byzantine failures? Or, does such a
protocol not exist?

A: No! We need to assume $f < n/3$

Example: $n = 3$, $f = 1 \Rightarrow$ impossible (if possible if $n = 4$)

\[ a \quad \text{good byzantine} \quad a \quad a \]

\[ a, b: \text{must decide 1} \quad a, c: \text{must decide 0} \]

no agreement: contradiction
A protocol for byzantine consensus if $f < n/3$

Tree $T_p$ maintained by node $p$, it contains values $V^p_0$ where if $o: q_1, q_2, \ldots, q_k$ means:
- $q_k$ told $p$ that $q_{k-1}$ told $q_k$ .... that $q$ told $q$, that
- $q$'s value is $v$

$V^p_0$ is input to $p$ itself

Let us define Majority $(S)$ to be the value that occurs most in bag $S$ (breaking ties deterministically).

For a leaf: $d^p_0 = v^p_0$
Protocol for Byzantine failures

- Initialise tree: set all \( v_p^0 = + \) and \( v_e^0 = CGpJ.in \).
- Round \( r \), \( 1 \leq r \leq f+1 \)
  - for all \( q \) with \( |\sigma| = r - 1 \) \& \( p \notin \sigma \):
    send \( v_q^p \) to all \( q \) (including \( p \)); call this message \( m_q^q \).
  - receive all \( m_q^q \) addressed to \( p \) and store in \( v_p^q \).
    (by the protocol \( x \notin \sigma \) so \( p \) receives \( n-(r-1) \) such messages from each \( x \)).
- To decide work from the leaves up the tree
  - \( d_p^0 = v_p^p \) for \( |\sigma| = f+1 \).
  - \( d_p^e = \text{Majority} (\{ d_q^p | q \notin \sigma \}) \).
  - Node \( p \) decides \( d_p^e \).

this part the same as for crash failures
Lemma 1: If \( p, q \) and \( r \) are not faulty, then for all possible \( \sigma \) we have \( v^p_{\sigma; r} = v^q_{\sigma; r} \)

Lemma 2: Let \( \sigma \) be arbitrary and let \( r \) be non-faulty. Then there is a value \( v \) such that for non-faulty \( p \) we have \( d^p_{\sigma; r} = v^q_{\sigma; r} = v \)

Proof: by induction on the length of \( \sigma; r \), starting at the leaves

base case: \( |\sigma; r| = f + 1 \)

level \( f \) \( V^r_{\sigma} = v \)

level \( f + 1 \) \( V^p_{\sigma; r} = v \) \( V^q_{\sigma; r} = v \)

\( d^p_{\sigma; r} = v \) \( d^q_{\sigma; r} = v \)

by majority decision rule \( d^q_{\sigma; r} = v \)

by induction \( V^r_{\sigma} = v \)

\( n - |\sigma; r| = n - f > 2f \)

\( \Rightarrow \) the majority of children is correct

\( \Rightarrow \) all these correct children have \( d = v \)

You need \( f < n/2 \) here.
Validity: if all non-faulty processors have input \( v \), they decide on \( v \).

Proof: If this is the case, they all send \( v \) to all other processors in the first round: \( v_q^p = v \) for all correct \( p, q \).

By lemma 2, this means that \( d_g^p = v \) for all correct \( p \) and \( q \).

\[
\text{d}_e = \text{majority} \left( \sum d_q^p \mid \text{for all } q \geq 3 \right) = v
\]
Agreement

Definition 1: \(\sigma\) is common if \(d^p_\sigma = d^q_\sigma\) (for all non-faulty \(p, q\)).

Definition 2: A subset of nodes \(C\) in a tree \(T_p\) is path cover 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.

\[\text{warning: that does not mean that for different } \sigma, \sigma' \text{ that } d^p_{\sigma} = d^q_{\sigma'}\]
Lemma 3: There exists a common path covering of the tree constructed by the consensus algorithm.

Proof: All leafs in the tree correspond to a path \( s \) of length \( f+1 \).

So \( s = s', s \geq s'' \) for some non-faulty \( r \).

By Lemma 2, \( d_{s',r} = d_{s'',r} \) for all non-faulty \( q, p \).

\( \Rightarrow s, r \) is common and on the path.
Lemma: Let \( \sigma \) be a node in a tree. If there is a common path covering of the subtree rooted at \( \sigma \), then \( \sigma \) is common itself.

Proof: By induction on the length of \( \sigma \):
- If \(|\sigma| = f+1\), the lemma holds trivially.
- Let us assume \(|\sigma| \leq f\) and there is a common path covering \( \mathcal{C} \) for the subtree rooted at \( \sigma \).

Then all children of \( \sigma \) have a common path covering (essentially \( \mathcal{C} \), split over the subtrees). And by the induction hypothesis, these children then are all common.

Hence \( d^p_{\sigma,r \mid \sigma} = d^q_{\sigma,r \mid \sigma} \) for all pairs \( p,q \) \( \Rightarrow \)

\[
d^p_{\sigma} = \text{Majority} \left( \{ d^p_{\sigma,r \mid \sigma} \mid r \neq r_0 \} \right) = \text{Majority} \left( \{ d^q_{\sigma,r \mid r_0} \} \right)
\]

\( = d^q_{\sigma} \) \( \Rightarrow \) \( \sigma \) is common.
If we assume authentication, we can build a more efficient Byzantine consensus protocol. If \( p \) is Byzantine, it can still send arbitrary messages (with valid signatures) to all other nodes.
(Binary) broadcast protocol (for one sender \( p \))

- \( p \) aka agreement protocol

- \( p \) does the following in round 1:
  - if \( \text{CEP}_p \) in \( t \) then send \( \text{CEP}_p \) to all other nodes
  - decide \( \text{CEP}_p \), in

- Other nodes \( q \):
  - for each round \( r \in [1..f+1] \)
    - if you receive a valid \( \text{CEP}_q \) \( C(1^r = r) \), \( (o = p ; o' - p) \)
      - then send \( \text{CEP}_q \) to all other nodes
      - and decide on the value 1. Then stop
  - decide on \( o \) and terminate

\[ (|r| = f + 1) \rightarrow \]
\[ 1 \leq q < f + 1 \Rightarrow q \text{ sends } [\text{CEP}_q]_q \text{ to all} \]