Expected Constant Round Byzantine Broadcast under Dishonest Majority

Jun Wan (junwan@mit.edu)
Hanshen Xiao (hsxiao@mit.edu)
Elaine Shi (runting@gmail.com)
Srini Devadas (devadas@csail.mit.edu)

Byzantine Broadcast [Lamport et al. 82]

- A set of users aim to reach consensus, one of them is the designated sender.
- The sender is given an input bit $b \in \{0, 1\}$
  - *Consistency*: all honest users must output the same bit; and
  - *Validity*: all honest users output the sender's input bit if the sender is honest.

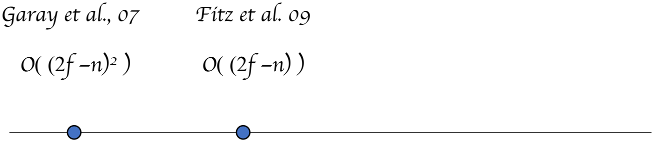
Background and Previous Work

Under synchronous setting,

- [Dolev and Strong, 83]: no deterministic protocol can achieve Byzantine Broadcast within $f + 1$ rounds, where f is the number of corrupted users.
- Focus on randomized protocols

Previous work

- Honest majority: expected constant rounds protocols exist (even under adaptive adversary) [Katz and Koo 09, Abraham et al. 19].
- Dishonest majority:



n: # total users

f: # corrupted users

Previous work

- Honest majority: expected constant rounds protocols exist (even under adaptive adversary) [Katz and Koo 09, Abraham et al. 19].
- Dishonest majority:

Garay et al., 07 — $O((2f-n)^2)$

Fitz et al. 09 — $O((2f-n))$

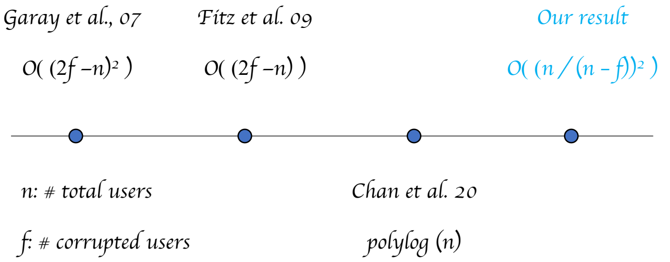
Chan et al. 20 — $polylog(n)$

n: # total users

f: # corrupted users

Previous work

- Honest majority: expected constant rounds protocols exist (even under adaptive adversary) [Katz and Koo 09, Abraham et al. 19].
- Dishonest majority: can we also achieve expected constant round complexity?



Byzantine Broadcast: main theorem

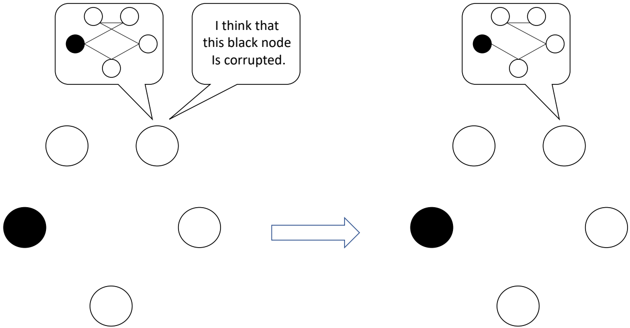
- Synchronous, assume trusted cryptographic setup.
- Weakly adaptive adversary.

Theorem

There exists a BB protocol with expected $O((\frac{n}{n-f})^2)$ round complexity under any weakly adaptive adversary that can adaptively corrupt $f < n-1$ nodes.

Novelty and new techniques: Trust graph

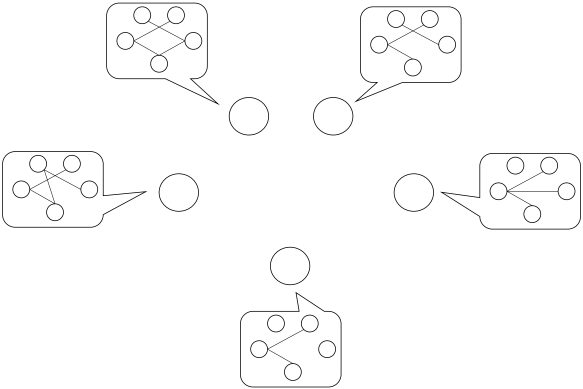
- Use a new graph idea: the trust graph.



- Vertices: users
- Edges: (u, v) indicates that u and v mutually trust each other

Novelty and new techniques: Trust graph

- Each user maintains its own trust graph
- An edge (u, v) belongs to user w's trust graph iff w thinks that u and v mutually trust each other



Trust graph: challenge

- Challenge 1: not all misbehavior leave behind cryptographic evidence.
- If a user u accuses v of not sending any message:
  - u is corrupt and intentionally drops v's messages
  - v is corrupt and does not send message
- Observation: can't tell which of $\{u, v\}$ is corrupt, but at least one of u and v must be corrupt!

Trust graph: challenge

- Challenge 1: not all misbehavior leave behind cryptographic evidence.
- Allow u to complain about v without providing an evidence.
  - a receiver of this complaint can be convinced that at least one node among u and v is corrupt
- Do not allow u to express distrust about an edge (v, w) that does not involve itself.
- Honest users are always fully connected.

Trust graph: challenge

- Challenge 2: users do not share the same trust graph.
- Solution: if honest users always share their knowledge to others, then for any honest nodes u, v:
    - u's trust graph in round $r + 1$ is a subgraph of v's trust graph in round r
- Work with this imperfect condition to derive agreement.

Properties of the trust graph

- If a node u is distrusted by more than $f+1$ other users,
  - can prove that u is a corrupt user and remove u from trust graph.
  - the proof: $f+1$ distrust signatures
- A stronger check condition:

honest users always remain fully connected

$\Longrightarrow$ honest users are always in a clique of size $n-f$

$\Longrightarrow$ if a node is not in any clique of size $n-f$, it must be corrupt

Properties of the trust graph

- Unfortunately, checking whether a node is in any clique of size $h = n - f$ is NP-hard.
- Let $N(u)$ denotes the set of neighbors of u (including u itself).
- A relaxed check condition:

$$
\begin{aligned}
& \text{honest users always remain fully connected} \\
\Longrightarrow\ & \text{honest users are always in a clique of size } h \\
\Longrightarrow\ & \text{if an edge } (u, v) \text{ is not in any } h\text{-clique, one of } u, v \text{ must be corrupt} \\
\Longrightarrow\ & \text{if } |N(u) \cap N(v)| < h, \text{ one of } u, v \text{ must be corrupt}
\end{aligned}
$$

Maintaining the trust graph

- Remove edge (u, v) if either u or v complains about the other.
- Remove edge (u, v) if $|N(u) \cap N(v)| < h$.
- Remove a node u if degree of u is less than $h - 1$.

Theorem

The diameter of any honest user's trust graph is upper bounded by $d = \lceil n/h \rceil + \lfloor n/h \rfloor - 1$

Trust graph: conclusion

We create a trust graph maintenance scheme that guarantees:

- **Honest clique invariant**: all honest users are fully connected in any honest user's trust graph.
- **Small diameter invariant**: honest users' trust graphs must have small diameter.
- **Monotonicity invariant**: for any honest users u and v, u's trust graph in round $t > r$ must be a subgraph of v's trust graph in round r.

Using trust graph: TrustCast protocol

- TrustCast: a weaker primitive of consensus,
- Use the trust graph to keep track of corrupt users during the broadcast.

Using trust graph: TrustCast protocol

A sender s wants to send a message to all other users.

- If an honest user u has not received from the sender s, then s is removed from u's trust graph.

Difference between:

- u distrusts v: u knows that v is corrupt.
- u removes v from its trust graph: u has proof that v is corrupt.

Intuition of TrustCast

A sender s wants to send a message to all other users.

- If an honest user u has not received from the sender s, then s is removed from u's trust graph.

To achieve this, we will show that

- If an honest user u does not receive from the sender s **by the** i^{th} **round**, then

$$d(u, s) > i,$$

  i.e., the distance between u and s in u's trust graph is larger than i.

Intuition of TrustCast

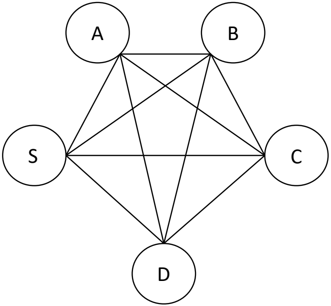
- Desired property: if u does not receive from s by the i^{th} round, then
$$d(u, s) > i,$$
- Combined with: diameter of an honest user's trust graph is upper bounded by $d = \lceil n/h \rceil + \lfloor n/h \rfloor - 1$.
- Imply: at the end of the d^{th} round, if u has not received from the sender s, then s is removed from u's trust graph.

Intuition of TrustCast

- Desired property: if u does not receive from s by the i^{th} round, then
$$d(u, s) > i.$$
- How to make it happen: if u does not receive from s by the i^{th} round, then
  - u distrusts any user v such that $d(v, s) < i$.
- Immediately implies that $d(u, s) > i$.

TrustCast: example

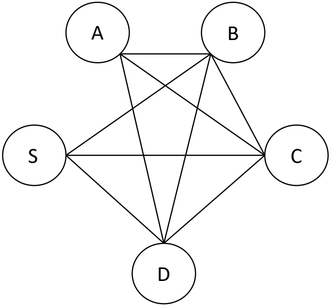
The sender s shares a message m with A, B, C, D.
The trust graph of an honest user A is a complete graph at the beginning.



TrustCast: example

In round 1, A does not receive anything from s.

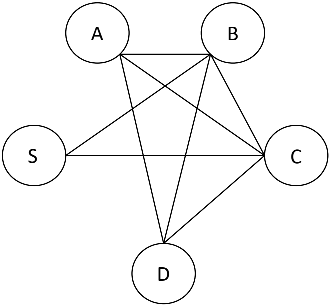
- By the protocol: A distrusts any user v such that $d(s, v) = 0$.
- Imply: A distrusts the sender s.



TrustCast: example

In round 2, B and C does not send anything. D complains to A that s did not send anything in round 1.

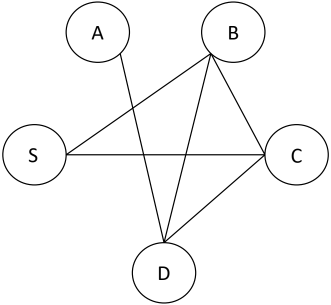
- Deal with complaints: A removes the edge (s, D) from its trust graph.



TrustCast: example

In round 2, B and C does not send anything. D complains to A that s did not send anything in round 1.

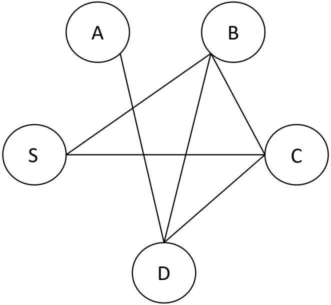
- By the protocol: A distrusts any user v such that $d(s, v) = 1$.
- Imply: A distrusts users B and C.



TrustCast: example

Why A should distrust B and C?

- If B, C have received from s: would relay it to A.
- If B, C have not received from s: would complain about s to A.



TrustCast Protocol: Conclusion

By the end of the TrustCast protocol, u either receives a message signed by s or removes s from its trust graph.

Protocol $\mathsf{TrustCast}^{\mathsf{Vf},s}(m)$

Input: The sender s receives an input message m and wants to propagate the message m to everyone.

Protocol: In round 0, the sender s sends the message m along with a signature on m to everyone.

Let $d = \lceil n/h \rceil + \lfloor n/h \rfloor - 1$, for each round $1 \leq r \leq d$, every node $u \in [n]$ does the following:

($\star$) If no message m signed by s has been received such that $u.\mathsf{Vf}(m) = \mathsf{true}$ in round r, then for any v that is a direct neighbor of u in u's trust graph: if v is at distance less than r from the sender s, call $\mathsf{Distrust}(v)$.

Outputs: At the beginning of round $d+1$, if (1) the sender s is still in u's trust graph and (2) u has received a message m such that $u.\mathsf{Vf}(m) = \mathsf{true}$, then u outputs m.

Round complexity: same as diameter, $\Theta(n/h)$.

Byzantine Broadcast

Elect a new leader in each epoch.

- Propose: the leader *trustcasts* a proposal.
- Vote: each user *trustcasts* their votes on leader's proposals.
- Commit: each user *trustcasts* commit message.

Byzantine Broadcast: why Trust Graph matters

- Only nodes on my trust graph matters.
  - Corrupt users have to send something to remain on honest users' trust graphs.

Byzantine Broadcast: why Trust Graph matters

- Trust graph Monotonicity property helps!
  - Proposals / Votes / Commit proofs valid to an honest user u in round r, will always be valid to all other honest users in round $r + 1$.
  - Once an honest user commits, future leaders cannot propose a different bit.

Byzantine Broadcast: round complexity

- $\Theta(1)$ TrustCasts in each epoch $\implies$ $\Theta(n/h)$ rounds per epoch
- Terminates when we have an honest leader, happens in expected $\Theta(n/h)$ epochs.
- Round complexity: $\Theta(n^2/h^2)$ in expectation.

Adaptive adversary

- An adaptive adversary can corrupt the leader in each epoch.
- If all leaders are corrupt, need $f + 1$ epochs.

Adaptive adversary: solution

- Postpone leader election: everyone pretends to be leader.
- Make proposals unforgeable even after corrupting the leader:
  - modify and upgrade the TrustCast protocol.

Conclusion

A Byzantine Broadcast protocol under dishonest majority and weakly adaptive adversary:

- expected $\Theta(1)$ round complexity.
- $\Theta(n^4)$ the communication complexity.

Open problems and Acknowledgement

- How to achieve expected constant round complexity under a strongly adaptive adversary?

Open problems and Acknowledgement

- How to achieve expected constant round complexity under a strongly adaptive adversary?

We would like to thank

- Zachary Newman, Ling Ren and the anonymous TCC reviewers;
- our awesome shepherd Ran Cohen.

Acknowledgement

Thank you!