Byzantine Broadcast in $\widetilde{O}(n^2)$ Communication

Yuval Efron
Institute for Advanced Study
Princeton, NJ, USA

Jovan Komatovic
Category Labs
New York, NY, USA

Ling Ren
University of Illinois
Urbana-Champaign
Urbana, IL, USA

Abstract

We present a deterministic n-party Byzantine broadcast protocol secure against $t < n$ faults, with communication complexity $O(n^2 \log^4(n)\lambda)$ and $O(t)$ rounds (where λ is a security parameter), improving upon the seminal Dolev–Strong protocol from 1983.

1 Introduction

Byzantine broadcast is one of the most fundamental problems in distributed computing, originally introduced and defined in the pioneering work of Lamport, Pease, and Shostak [PSL80; LSP82]. The most prominent Byzantine broadcast protocol is the seminal protocol of Dolev and Strong [DS83], dating back to 1983. Assuming digital signatures and a public-key infrastructure (PKI), the Dolev–Strong protocol allows a sender to reliably deliver a value to all parties, regardless of the number of corruptions in the network. It is a deterministic protocol (outside the use of signatures), and it requires $t + 1$ rounds, $O(n^2)$ messages, and $O(n^3\lambda)$ bits of communication in the worst case, where n is the total number of parties, t is an upper bound on the number of corruptions, and λ is the security parameter. For deterministic protocols, the Dolev–Strong paper shows that $t + 1$ rounds are tight, and the quadratic number of messages are also shown to be asymptotically optimal [DR85] (when $t = \Theta(n)$).

Over the following decades, Byzantine broadcast has received an enormous amount of attention, from a myriad of viewpoints, settings, and models, and has grown into a vast literature that is far too large to survey here. The overwhelming majority of work assumes an honest-majority $\left(t < \frac{n}{2}\right)$ or supermajority $\left(t < \frac{n}{3}\right)$, e.g., [Bra87; FM97; CKS05; KK09; KS11; AMS19; ADD+19; ACD+19; ER25]. Most relevant to this work, there are deterministic protocols that achieve $O(n^2\lambda)$ bits of communication, with optimal quadratic dependence on n, in synchrony [BGP92; CW92; MR21; CKS22; ETP24; CDG+24] and, after stabilization, in partial synchrony [CDG+22; CDG+25].

In the dishonest-majority setting, however, prior work has focused on randomized protocols and/or round complexity [GKK+07; CPS20; WXS+20; TLP22; ALP+25; CDL+26], including early stopping [DRS90; LN24; DKL25]. There has been essentially no progress on the communication complexity of deterministic Byzantine broadcast tolerating arbitrary $t < n$ faults. Dolev–Strong remains the state of the art in terms of (worst-case) communication complexity at $O(n^3\lambda)$, while the lower bound of Dolev and Reischuk [DR85] stands at $\Omega(n^2)$. After over four decades of research, the following question has remained open:

Is there a deterministic Byzantine broadcast protocol secure against $t < n$ faults with $o(n^3\lambda)$ communication complexity?

We resolve this question in the affirmative. Specifically, we design a deterministic Byzantine broadcast protocol under common cryptographic assumptions, with $\widetilde{O}(n^2\lambda)$ communication complexity and $O(t)$ rounds (see Theorem 1 for the formal statement).

2 Model & primitives

We consider a synchronous network of n parties, operating in rounds. Messages sent in a round are delivered at the end of that round. An adversary can corrupt up to t parties and has complete control over their behavior (in particular, corrupt parties cede their internal states to the adversary). We say that the adversary is t-bounded if it corrupts at most t parties. For simplicity and clarity of exposition, we assume throughout that n is a power of two. The results easily generalize to arbitrary n, either by adjusting constants or by simulating the protocol for the smallest power of two $n' \geq n$, in which the remaining $n' - n$ parties are crashed.

We assume a public-key infrastructure (PKI), ideal digital signatures, and an ideal vector commitment scheme: signatures cannot be forged, and a commitment cannot be opened to different values at the same position. All security statements use this idealized model. A protocol is deterministic if it makes no random choices outside calls to these cryptographic primitives. We assume signatures and commitments have size $O(\lambda)$ where $\lambda > \log n$ is a security parameter, and openings have size $O(\lambda \log L)$, where L is the length of the vector. Vector commitments with these sizes can be constructed from a collision-resistant hash function using Merkle trees [Mer80].

Definition 1 (Byzantine Broadcast (BB))**.** In the BB problem, there is a designated party ℓ, referred to as the *leader*, with an *input* $b \in \{0,1\}$. As output, each party p *decides* a value $b_p \in \{0,1\}$. For a protocol π, the following properties pertain to the BB task.

1. **Termination.** Every correct party decides and halts.
2. **Agreement.** All correct parties decide the same $b' \in \{0,1\}$.
3. **Validity.** If ℓ is correct, then all correct parties decide b, i.e., ℓ's input.

A protocol π is a t-secure BB protocol if it satisfies termination, agreement, and validity against every t-bounded adversary.

The communication complexity of a protocol is the worst-case total number of bits sent by correct parties, over all executions with a t-bounded adversary.

Theorem 1. *Let $t < n$. Assuming a PKI, ideal digital signatures, and ideal vector commitments, there exists a t-secure deterministic* BB *protocol with communication complexity $O(n^2 \log^4(n)\lambda)$ and $O(t)$ rounds.*

In our protocol, we make use of the existence of the following family of graphs.

Lemma 2. *Let $c \in \{1,\dots,n\}$. There exists a graph $G_c = (V, E_c)$, where $V = [n]$, such that $d_{G_c}(i) = O(c \log n)$ for every $i \in V$, and for all disjoint $X, Y \subseteq V$ with $|X|, |Y| \geq n/c$, there exist $x \in X$, $y \in Y$ such that $\{x, y\} \in E_c$.*

Proof. Sample $m = 3c \log n$ independent uniform permutations $\pi_1, \dots, \pi_m$ of $[n]$ and include every edge $\{v, \pi_j(v)\}$ with distinct endpoints. Every degree is at most $2m = O(c \log n)$. It suffices to consider $|X| = |Y| = K := \lceil n/c \rceil$. Fixed disjoint K-sets lack an edge with probability at most $\exp(-mK^2/n) \leq n^{-3K}$; a union bound over at most n^{2K} choices of (X, Y) gives existence. □

3 Byzantine broadcast protocol

3.1 Protocol overview

Dolev–Strong. Our starting point is the seminal protocol of Dolev and Strong [DS83], so let us first recall how it works. In round 0, the leader ℓ signs its input and multicasts it. From then on, a party accepts a bit b in round $r \leq t + 1$ if, by then, it has received signatures on b from r distinct parties, including ℓ. When a party accepts b, it signs b and multicasts all $r + 1$ signatures on b, so that the other parties accept b in the next round. After round $t + 1$, each party outputs the unique bit it accepted, or a default value if it accepted neither bit or both.

The expensive case. Dolev–Strong is simple but expensive: each correct party multicasts at most $t + 1$ signatures per bit, for $O(n^2 t\lambda)$ bits in total, which is cubic for $t = \Theta(n)$. The cubic cost arises when, late in the protocol, many correct parties accept a bit at the same time, and all of them multicast $\Theta(n)$ signatures. Any improvement must handle this case. There is, however, a potential silver lining to be explored. Note that these simultaneous acceptances create plenty of fresh evidence for b, namely the accepting parties' own signatures. Dolev–Strong ignores this fact: a party multicasts all the signatures it holds because, for all it knows, it may be the only correct party to accept b and it alone must convince everyone. But in exactly the expensive case, it is not alone, and the fresh signatures raise the hope of a cheaper way to convince the rest.

A responsibility shift. In Dolev–Strong, the burden of convincing falls on the convinced: once a party accepts b, it sends every other party the signatures that convinced it. As we saw, this is a challenging task, since, for all it knows, it may be the only party to have accepted b. The main idea behind our protocol is

to turn the tables and shift this burden to the parties that are not yet convinced. Specifically, a party that accepts b merely reports its acceptance by multicasting its own signature on b. It is then up to the parties that have not yet accepted b to collect signatures from the parties that reported their acceptances.

On the surface, this may look just as hard. However, there is an asymmetry: a party that has not yet accepted b has extra information, namely how many parties reported accepting b in this round. This information helps. If few parties reported, it needs to contact only those few to get the signatures it lacks. If many reported, it can ask each of them for a small number of signatures, spreading the load evenly. Each such unconvinced party thus faces a retrieval task as a *receiver*: it wants to collect the signatures it lacks efficiently, from the set P of *senders*, who reported accepting b. A protocol for this task must guard against two threats that pull in opposite directions:

- A corrupt sender, which pretends to have accepted b, must not waste much of the receiver's communication.
- A corrupt receiver must not be able to force a sender to blindly send all its signatures.

Addressing both at once is the core difficulty, and requires several new ideas.

The receiver's point of view. Consider a correct receiver q that hears acceptances for b in a round from a set P of senders, and wants to retrieve signatures from P. Intuitively, q would like to partition the signatures it lacks among the senders, and ask each of them for a different set of signatures. This partition task is nontrivial on its own, and we return to it shortly. For now, suppose that an oracle grants q such a partition, equal among the senders, in which each correct sender is assigned only signatures it holds. The retrieval protocol is then clear: each sender sends q its part. A sender that does not send its part is corrupt, and q blames it. This is a win-win situation: if at least half of the senders send their parts, q collects at least half of the signatures it lacks; otherwise, q blames more than half of the senders. Either way, q cuts in half either the senders or the signatures it lacks. After $O(\log n)$ attempts, q either collects all the signatures it lacks or blames all the senders. Note that this retrieval protocol is communication-efficient (outside the oracle itself): q sends nothing, and correct senders send disjoint sets of signatures to q, hence at most n signatures in total.

A binary-search-style assignment procedure. Next, we turn to the oracle, a protocol that determines which signatures each sender in P should send to q. Let A be the set of signers whose signatures q already has, W_p the set of signers whose signatures sender p holds, and D the number of additional signatures q needs. Suppose for simplicity that $D \geq |P|$ and q asks each sender for a single signature. Every correct sender has at least D signatures that q lacks, so

$$|W_p \setminus A| \;\geq\; D \geq |P| \qquad \text{for every correct sender } p.$$

The receiver's goal is to assign the senders in P to distinct signatures such that every correct sender p is assigned to a signature from $W_p \setminus A$.

To find such an assignment, the receiver uses a process that resembles a binary search. Let I^0 and I^1 be the lower and upper halves of $[n]$. Each sender p tells q $\ell_p = |W_p \cap I^0|$ and $r_p = |W_p \cap I^1|$. Then, q discounts the signatures it already has: q now knows that p claims to have at least $\ell_p - |I^0 \cap A|$ signatures that q lacks in I^0, and at least $r_p - |I^1 \cap A|$ in I^1. Next, q assigns some senders to I^0 and other senders to I^1, and continues the search with both halves. How does q decide which senders go to I^0, and which go to I^1? The key idea is to maintain the invariant at every interval of the search: if P_I is the set of senders assigned to an interval I, then

$$|(W_p \cap I) \setminus A| \;\geq\; |P_I| \qquad \text{for every correct sender } p \in P_I.$$

Initially, the invariant holds for $I = [n]$. At each step, q ensures the invariant is maintained as follows. It starts a counter z at $|I^0 \cap A|$, the number of signatures in I^0 it already has. q then goes over the senders

in decreasing order of ℓ_p: if $\ell_p > z$, it sends p to I^0 and increases z by one (to account for p); otherwise, it sends p to I^1. One can show that this maintains the invariant in both halves (Claim 16). The invariant guarantees that if a sender p is assigned to a leaf $\{i\}$, then p claims to have party i's signature and q lacks it. This procedure is also cheap: the search has $\log n$ levels, and at each level, every sender reports two counts to q, and q tells every sender in which half to continue. Each of these messages has $O(\log n)$ bits, so the search costs $O(\log^2 n)$ bits for each sender, and $O(|P| \log^2 n)$ bits for the receiver.

Relaxing assumptions. So far, we have assumed that q asks each sender for a single signature and $D \geq |P|$. In our actual protocol, however, these assumptions may fail in two ways. First, q may ask each sender for d signatures, where $1 < d \leq D$. Each sender p simulates d virtual senders $(p, 1), \ldots, (p, d)$, each holding the signatures of p and asked for a single signature. If $d|P| \leq D$, the invariant holds for these virtual senders, so the receiver runs the assignment procedure with them. The signatures assigned to $(p, 1), \ldots, (p, d)$ are pairwise distinct, and together they are the d signatures that q asks p for. This costs $O(d \log^2 n)$ bits per physical sender, both for the sender and for q's interaction with it.

Second, we may have $d|P| > D$. Then q partitions the physical senders into groups of size $\lfloor D/d \rfloor$, the last possibly smaller. Each group has at most D virtual senders, so the invariant holds within each group. The receiver q runs the assignment procedure independently with each group's virtual senders, at a cost of $O(D \log^2 n)$ bits per group. If most members of some full group reply, q collects a constant fraction of the signatures it needs; otherwise, q blames a constant fraction of the senders.

Gentle scaling. It might appear that we are done describing what the receiver does. However, there is another subtlety in the communication accounting. Recall that the binary-search assignment procedure costs the receiver $O(D \log^2 n)$ per group. Consider a receiver that still needs $D = \Theta(n)$ signatures and hears from $|P| = \Theta(\sqrt{n})$ senders. If all these senders are corrupt, they can participate in the assignment procedure but withhold their signatures. Subsequently, another set of corrupt senders can do the same, forcing the receiver to spend $\widetilde{\Theta}(n^{1.5})$ communication without receiving any of the requested signatures. Therefore, the receiver must collect signatures in an even more fine-grained manner, in which the receiver and the senders "establish trust" gradually. Even if the receiver needs many signatures from each sender, it first requests only one from each sender, and only if that goes through does it move on to request two, then four, and so on, from each sender. A sender is thus asked for c signatures only after delivering $c/2$, so a corrupt sender can waste only about as much of the receiver's communication as it has already provided.

As it turns out, gentle scaling also helps to protect against a corrupt receiver. As we explain next, senders must check that the receiver's requests to different senders are consistent, and these checks become more expensive as the number of requested signatures grows. A corrupt receiver that immediately requests $\Theta(n)$ signatures can force senders to waste too much communication on the consistency check. To prevent this, a sender performs the check for c signatures only after passing the check for $c/2$. This keeps the cost of the consistency checks (across all senders) to $\widetilde{O}(n)$ messages per receiver (Claim 15). We now explain how these consistency checks work.

Sender list consistency. Consider the following naive case: a receiver q tells a sender p that p is its only sender, and asks p for all the signatures it lacks. What should p do? Of course, naively sending q all its signatures, even gradually as described above, makes p behave as in Dolev–Strong: repeated with every receiver and every sender, this leads to $\Theta(n^3)$ communication. It is here that the graphs of Lemma 2 come to the rescue. Specifically, when a sender receives a request from a receiver (signed by the receiver) , even before engaging with the assignment procedure, it checks that the receiver is not equivocating on the sender list, for example, by telling two or more senders that each of them is its only sender. Concretely, a sender that is asked for c signatures uses the graph G_c of Lemma 2 to check that the request it receives is

consistent with what other senders receive. It simply forwards the request to all its neighbors in G_c, and stops interacting with the receiver if a neighbor forwards it a different request.

The next question is what a request should contain. The receiver cannot send each sender the entire list of senders it talks to: this list may be long, and sending it from every receiver to every sender can cost cubic communication. On the other hand, telling each sender only the length of the list is insufficient. For instance, the receiver can tell every sender that it talks to $\sqrt{n}$ senders, while dividing all n senders into $\sqrt{n}$ disjoint lists. A sender then hears the same claim from its neighbors in $G_{\sqrt{n}}$, whether or not they are on its list, and this pattern can be replicated across the entire graph, so that every sender sees consistent information, although the receiver involves all n senders.

This is where the vector commitments come in. The receiver commits to its list of senders, and sends each sender the commitment and the length of the list, together with a membership proof: an opening of the sender's position in the list. Senders compare commitments, list lengths, and requested signature counts, but not membership proofs. By binding, the receiver can provide valid membership proofs for the same request to no more distinct senders than its claimed list length. To involve more senders, the receiver must therefore send different requests, and as soon as two disjoint sets of at least n/c senders hold different requests, Lemma 2 provides an edge of G_c between them, over which the conflict is detected.

Putting it all together. With this retrieval procedure in hand, we obtain a protocol that, at a high level, resembles Dolev–Strong. In each step $r = 1, \ldots, t$, each party that accepts a bit b, holding signatures on b from r parties including ℓ, reports this by multicasting its accept message. Then, each party that has not yet accepted b runs a retrieval with the parties that reported as senders, aiming for $r + 1$ signatures on b, and accepts b in step $r + 1$ if it succeeds. The properties of Dolev–Strong still hold: if a correct party accepts b in step r, every correct party accepts b by step $r + 1$. In particular, correct parties serve as senders in at most two steps for each bit b. Up to logarithmic and λ factors, the reports then cost $O(n^2)$ bits, each receiver spends $O(n)$ bits, and the senders of each receiver send $O(n)$ bits per retrieval, for $\widetilde{O}(n^2)$ bits in total. Note, however, that running a retrieval after every step, as described so far, yields a protocol with $\widetilde{O}(t)$ rounds, as each retrieval takes polylogarithmically many rounds.

Getting to $O(t)$ rounds. To reduce the round complexity to $O(t)$, we can run the retrieval process less frequently. Let E be the number of rounds in the retrieval process; E is polylogarithmic in n. The parties run the retrieval process once every E consecutive steps of accepting/reporting. We say E consecutive steps and a retrieval process form an *epoch*. Now the only concern is that a party that accepts b in the middle of an epoch may hold too few signatures for the retrieval process at the end of the epoch. A simple observation closes it. When a party reports accepting b in a step, it also learns how many other parties accepted b in the same step. If fewer than E reported, it can simply multicast all its signatures like in Dolev–Strong; this is cheap enough as fewer than E parties would do the same. Otherwise, it receives signatures from at least E more parties than the current step requires, which suffices for the retrieval at the end of the epoch. The protocol thus alternates between E two-round steps (reporting + potentially sending all signatures), and a retrieval phase of E rounds, as illustrated in Figure 1, and thus has $O(t)$ rounds. Finally, in the corner case with $t < E$, even a single epoch takes too many rounds; but in this case Dolev–Strong is cheap enough with $O(n^2 t\lambda) = \widetilde{O}(n^2)$ communication, and we simply run it.

3.2 Byzantine Broadcast

We first present the retrieval primitive used by our protocol.

Definition 2 (Retrieval (RET))**.** In the RET problem with target M over a universe $\mathcal{V}$, there is a designated party q, referred to as the *receiver*, with input a set $A \subseteq \mathcal{V}$ and a set P of *senders*. Each sender has as input a set of $K \geq M$ values in $\mathcal{V}$. Only q outputs: a set $V \supseteq A$ of values and a set $F \subseteq P$ of parties.

1. **Delivery.** If q and some sender are correct, then $|V| \geq M$.
2. **Resolution.** If q is correct and $|V| < M$, then $P \subseteq F$.

A protocol is a t-secure RET protocol if it satisfies delivery and resolution against every t-bounded adversary.

We prove the following theorem in Section 3.3.

Theorem 3. *Let $t < n$. Assuming digital signatures, PKI and vector commitments,* Ladder *is a t-secure* RET *protocol over $\mathcal{V} = [n]$, for targets $M \leq n$, with communication complexity $O(n \log^4(n)\lambda)$ for the senders and $O((|P| + |V \setminus A|) \log^2(n)\lambda)$ for the receiver, and at most $E = O(\log^3(n))$ rounds, where E is a publicly known positive integer independent of M.*

Let E be the round bound from Theorem 3, and define

$$T = \lceil t/E \rceil, \qquad M_i = \min\{iE + 1, t + 1\}.$$

After round 0, our protocol runs T epochs (Figure 1). Epoch $i \in [T]$ consists of an acceptance phase with steps $(i-1)E + 1, \ldots, iE$, of two rounds each, followed by a retrieval phase of E rounds. In the retrieval phase, value j is j's accept message for b. The final test follows epoch T.

Notation. Every message carries a type tag, and a subscript denotes the signer; e.g., $\langle \text{accept}, b \rangle_p$ is p's signed accept message for bit b. A non-leader's accept message for b is valid only if $\langle \text{accept}, b \rangle_\ell$ has also been received. Each party p maintains $A_p(b)$ (written $A(b)$ in the pseudocode), retaining one valid accept message per signer, including its own and those carried in other messages. The set B contains accepted bits.

Protocol 1: Byzantine broadcast at party p

if $t < E$: run Dolev–Strong; halt.
$B \leftarrow \emptyset$; $F \leftarrow \emptyset$.
Round 0: **if** $p = \ell$: multicast $\langle \text{accept}, b \rangle_p$ for its input b and set $B = \{b\}$.
for epochs $i = 1, \ldots, T$ **do**
  for $s = (i-1)E + 1, \ldots, iE$ **do** ▷ acceptance phase of epoch i
    Round 1: **for** bit $b \notin B$ with $|A(b)| \geq s$ and $\langle \text{accept}, b \rangle_\ell \in A(b)$ **do**
      $B \leftarrow B \cup \{b\}$; multicast $\langle \text{accept}, b \rangle_p$ with $\langle \text{accept}, b \rangle_\ell$.
    Round 2: **for** b such that p sent $\langle \text{accept}, b \rangle_p$ in Round 1 of this step and $|A(b)| < M_i$ **do**
      Multicast $\langle \text{certificate}, b, A(b) \rangle$.
  for $b \in \{0, 1\}$ **in parallel do** ▷ retrieval phase of epoch i
    if $b \notin B$:
      $P_b \leftarrow \{p' : \langle \text{accept}, b \rangle_{p'}$ received directly from p' in epoch $i\} \setminus F$.
      Run Ladder as the **receiver** with target M_i and input $(A(b), P_b)$.
      Let (V, F') be the output. $A(b) \leftarrow V$; if $|V| < M_i$, $F \leftarrow F \cup F'$.
    else if: p sent an accept message for b in this epoch and $|A(b)| \geq M_i$:
      Run Ladder as a **sender** with target M_i and input $A(b)$ for every receiver.
for $b \in \{0, 1\}$ with $|A(b)| \geq t + 1$ and $\langle \text{accept}, b \rangle_\ell \in A(b)$: $B \leftarrow B \cup \{b\}$.
Output the sole element of B if $|B| = 1$, and 0 otherwise; halt.

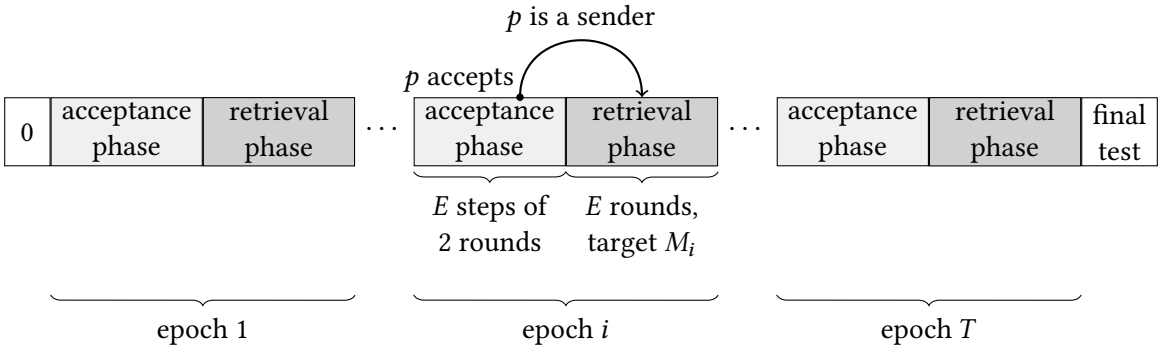

Figure 1: Each epoch consists of an acceptance phase of E two-round steps and a retrieval phase of E rounds, with target M_i in epoch i. A correct p that accepts in the acceptance phase of epoch i without sending a certificate message is a sender in its retrieval phase.

We assume $t \geq E$ in Claims 4 to 7.

Claim 4. *Let p be a correct receiver that executes line 12 of epoch i. In the retrieval phase of epoch i, every correct $p' \in P_b$ is a sender, and F contains no correct party.*

Proof. First, a correct party accepting at the last acceptance step $s = iE$ holds at least M_i accept messages: if $s \leq t$, its own new signature brings the count to at least $s + 1 = M_i$; otherwise, it already holds at least $s \geq M_i$. Thus, it sends no certificate message (line 8) in the last acceptance step. By definition of P_b (line 12), p' sent $\langle \text{accept}, b \rangle_{p'}$ at some step s of the acceptance phase of epoch i. If it also sent a certificate message, this carries at least $s + 1$ signers, so p accepts b by step $s + 1$ (line 6). Then p would not enter the receiver branch (line 11). Hence $|A_{p'}(b)| \geq M_i$, and p' is a sender (line 15). Finally, F gains $F' \subseteq P_b$ only if $|V| < M_i$ (line 14), and then, by the above and delivery (Theorem 3), P_b contains no correct party. □

Claim 5. *If ℓ is correct, every correct party accepts ℓ's input by step 1, and no correct party accepts the other bit.*

Proof. The leader accepts its input in round 0 (line 3); every other correct party receives its accept message and accepts at step 1 (line 6). Accepting requires ℓ's signature (lines 6 and 17), which the other bit lacks. □

Claim 6. *If a correct party accepts b in the acceptance phase of epoch i, every correct party accepts b by the first step of the acceptance phase of epoch $i + 1$, or by the final test (line 17) if $i = T$.*

Proof. If ℓ is correct, this follows from Claim 5. Otherwise, let p be the first correct party to accept b, at step s of the acceptance phase of epoch $i' \leq i$ (line 6); then $s \leq t$, as $t + 1$ signers include a correct one that accepted earlier. If p sends a certificate message (line 8), every correct party accepts b by step $s + 1$ in the same acceptance phase, by the argument in Claim 4. Otherwise p is a sender in the retrieval phase of epoch i' (line 15). Every correct q with $b \notin B$ includes p in P_b (line 12): it received p's accept message in that acceptance phase, and $p \notin F$ by Claim 4. By that claim and delivery (Theorem 3), each such q ends the retrieval phase of epoch i' with $|A_q(b)| \geq M_{i'}$ (line 14), and accepts at the first step of the acceptance phase of epoch $i' + 1$, or at the final test (line 17) if $i' = T$. □

Claim 7. *Our protocol has at most $6t + 1$ rounds, and correct parties send $O(n^2 \log^4(n)\lambda)$ bits.*

Proof. By Theorem 3, every retrieval invocation finishes within $E = O(\log^3(n))$ rounds. Each epoch has $2E + E = 3E$ rounds, so there are $1 + 3TE \leq 6t + 1$ rounds in total, as $TE \leq t + E \leq 2t$. We now analyze communication. Fix a bit b.

Acceptance phases. Each correct party sends an accept message for b at most once (line 6), which costs $O(n^2\lambda)$ bits in total. A party sending an accept message at a step $s > t$ holds at least $s \geq M_T$ accept messages, so it sends no certificate message (line 8). If a correct parties send accept messages for b at a step $s \leq t$ of the acceptance phase of epoch i, each holds at least $s + a$ accept messages in Round 2, so sending a certificate message requires $s + a < M_i \leq iE + 1 \leq s + E$, i.e., $a < E$. The first certificate message of a correct party, at step s_c, makes every correct party accept by step $s_c + 1$, so correct parties send certificate messages only at steps s_c and $s_c + 1$. These are fewer than $2E$ messages of $O(n\lambda)$ bits, each sent to n parties, so they cost $O(n^2 E\lambda) = O(n^2 \log^3(n)\lambda)$ bits.

Receiving. By Claim 4, in every retrieval phase of epoch j, each correct member of P_b is a sender with at least M_j values, as RET requires. Fix a correct q, and let (A_j, P_j, V_j) be its inputs and output in each epoch j in which it runs as a receiver. The sets of newly retrieved values $V_j \setminus A_j$ are disjoint, so $\sum_j |V_j \setminus A_j| \leq n$. If $P_j \neq \emptyset$ and $|V_j| < M_j$, then q adds P_j to F (line 14), by resolution; if $|V_j| \geq M_j$, then q accepts at step $jE + 1$ or at the final test (line 17), and runs no later receiver invocation for b (line 11). Hence $\sum_j |P_j| \leq 2n$, and by Theorem 3, q sends $O(n \log^2(n)\lambda)$ bits as a receiver.

Sending. By Claims 5 and 6, correct parties send accept messages for b in the acceptance phases of at most two consecutive epochs, so they are senders in the retrieval phases of at most those two epochs (line 15). In each of these, by Theorem 3, the senders of each of the n instances send $O(n \log^4(n)\lambda)$ bits.

Summing over the n receivers and the two bits gives $O(n^2 \log^4(n)\lambda)$ bits. □

Proof of Theorem 1. If $t < E$, Dolev–Strong (line 1) takes $t + 1$ rounds and $O(n^2(t + 1)\lambda)$ bits, which is $O(n^2 E\lambda) = O(n^2 \log^3(n)\lambda)$, since $t < E$ and Theorem 3 gives $E = O(\log^3(n))$. Otherwise, Claim 5 gives validity. For agreement, by Claim 6 it suffices that whenever a correct party accepts b, some correct party accepts b in an acceptance phase. This holds for ℓ's input, which correct parties accept in the acceptance phase of epoch 1, and for a bit accepted at the final test (line 17), whose $t + 1$ signers include a correct one, which accepted it in an acceptance phase or is ℓ. Claim 7 gives the bounds, and all correct parties halt after the fixed schedule. □

3.3 Retrieval

In this section, we prove Theorem 3. We use the following primitive.

Definition 3 (Splitting (SPLIT))**.** In the SPLIT problem, there is a designated party q, referred to as the *receiver*, with input a set $A \subseteq [n]$ and a set P of *senders*. Each sender p has as input a set $W_p \subseteq [n]$, where $|P| \leq |W_p| - |A|$ if p is correct. Only q outputs: $(j_p)_{p \in P}$, where $j_p \in [n] \cup \{\perp\}$.

1. **Split.** If q is correct, then the non-$\perp$ indices j_p are pairwise distinct and belong to $[n] \setminus A$.
2. **Validity.** If q is correct, then for every correct $p \in P$, $j_p \neq \perp$ and $j_p \in W_p \setminus A$.

A protocol is a t-secure SPLIT protocol if it satisfies split and validity against every t-bounded adversary.

We prove the following theorem in Section 3.4.

Theorem 8. *Let $t < n$.* Bisect *is a t-secure* SPLIT *protocol with communication complexity $O(\log^2(n))$ per sender and $O(|P| \log^2(n))$ for the receiver, and $2 \log n$ rounds.*

Set $k = \log n$ and $R = 8k + 8$. Commit commits to a list, with opening π_p for the position of p; $G_c = ([n], E_c)$ is from Lemma 2. Attempt $(s, j) \in \{0, \ldots, k\} \times [R]$ occupies rounds $(sR + j - 1)(2k + 3) + 1, \ldots, (sR + j)(2k + 3)$, numbered $1, \ldots, 2k + 3$ within the attempt. Party p runs the virtual senders $(p, 1), \ldots, (p, d)$ of Bisect.

Claim 9. *Let q be correct. In every attempt, the sets L_p of the members p of a group that are not in F after the attempt are pairwise disjoint, have size d, and are disjoint from A at the start of the attempt, and q receives their values.*

Protocol 2: Ladder **at receiver** q**, with target** M

1: $F \leftarrow \emptyset$.
2: **for** $s = 0, \ldots, k$, with $c = 2^s$ **do**
3: **for** $j = 1, \ldots, R$ **do**
4: **Round 1:** $D \leftarrow M - |A|$; **if** $D \le 0$: output (A, F) and **return**.
5: $d \leftarrow \min\{c, D\}$; $\rho \leftarrow \mathsf{Commit}(P)$.
6: Divide P into consecutive groups $\mathcal{G}_1, \ldots, \mathcal{G}_g$ of size $\lfloor D/d \rfloor$, the last possibly smaller.
7: Send $\langle \text{request}, s, j, \rho, |P|, d \rangle_q$ with opening π_p to each $p \in P$.
8: **Round 2:** Wait.
9: **Round** $3, \ldots, 2k+2$: $(j_x)_x \leftarrow \mathsf{Bisect}(A,\ \mathcal{G}_i \times [d])$ for $i \in [g]$, in parallel.
10: $F \leftarrow F \cup \{p \in P : j_{(p,e)} = \perp \text{ for some } e \in [d]\}$; $P \leftarrow P \setminus F$.
11: Send $\langle \text{query}, L_p \rangle$, $L_p = \{j_{(p,e)} : e \in [d]\}$, to each $p \in P$.
12: **Round** $2k+3$: Wait for $\langle \text{data}, \cdot \rangle$ messages from senders.
13: $A \leftarrow A \cup \bigcup \{L : \langle \text{data}, L \rangle \text{ received from some } p \in P\}$.
14: $F \leftarrow F \cup \{p \in P : \langle \text{data}, L_p \rangle \text{ not received from } p\}$; $P \leftarrow P \setminus F$.
15: Output (A, F).

Protocol 3: Ladder **at sender** p **with input** W**, for receiver** q **and target** M

16: **for** $s = 0, \ldots, k$, with $c = 2^s$ **do**
17: **for** $j = 1, \ldots, R$ **do**
18: **Round 1:** Wait to receive $\mu = \langle \text{request}, s, j, \rho, m, d \rangle_q$ with opening π from q.
19: **if** none is received: **return**; **if** $m > n$, $d \notin [1, \min\{c, M\}]$, or $cm > 2n$: **return**.
20: **Round 2:** Send $\langle \text{compare}, \mu \rangle$ to each v with $\{p, v\} \in E_c$.
21: **if** some $\langle \text{request}, s, j, \cdot \rangle_q \neq \mu$ is received: **return**.
22: **Round** $3, \ldots, 2k+2$: Run Bisect as senders $(p, 1), \ldots, (p, d)$, each with input W.
23: Wait to receive $\langle \text{query}, L \rangle$ with $|L| = d$ and $L \subseteq W$ from q; **if** none is received: **return**.
24: **Round** $2k+3$: Send $\langle \text{data}, L \rangle$ to q.

Proof. Write A for the receiver's set at the start of the attempt. For each correct $p \in \mathcal{G}_i$, $|\mathcal{G}_i| d \le \lfloor D/d \rfloor\, d \le D$ (line 6) and $D = M - |A| \le |W_p| - |A|$ for p's input W_p, so $|\mathcal{G}_i \times [d]| \le |W_p| - |A|$, which is the condition of SPLIT on correct senders for $\mathsf{Bisect}(A, \mathcal{G}_i \times [d])$ (line 9). By split (Theorem 8), the non-$\perp$ indices j_x of a group's senders are pairwise distinct and belong to $[n] \setminus A$. A member p stays outside F only if $j_{(p,e)} \neq \perp$ for all $e \in [d]$ (line 10), so $|L_p| = d$ (line 11), and only if q receives $\langle \text{data}, L_p \rangle$ from p (line 14). □

Claim 10 (Resolution)**.** *If q is correct and $|V| < M$, then $P \subseteq F$.*

Proof. If $|V| < M$, q reaches the last scale with $c = n \ge D$. Thus $d = D$ and every group is a singleton (lines 5–6). Any sender remaining after its first attempt supplies D new values by Claim 9, contradicting $|V| < M$. Hence P becomes empty. Since q removes a party from P only when adding it to F (lines 10 and 14), every party in the input P has been added to F. □

Claim 11 (No false blame)**.** *If q is correct, no correct sender is added to F.*

Proof. We first show that every request satisfies $c|P| \le 2n$, by induction over scales. Initially $c = 1$ and $|P| \le n$; within a scale, P only shrinks. Suppose a scale starts with at most $2n/c$ senders and completes all R

attempts with $|A| < M$. If its final sender set P_* had more than n/c members, every attempt would have $D > c$: otherwise, $d = D$ and a final sender would supply D new values, completing retrieval (Claim 9). Thus $d = c$. With group size $m = \lfloor D/c \rfloor \le n/c$, the number of groups is at most

$$\frac{2n}{cm} + 1 \le \frac{3n}{cm}.$$

Some group contains more than $m/3$ members of P_*, whose responses add more than $cm/3 \ge D/6$ new values (Claim 9). After R attempts, the positive deficit would be less than $n(5/6)^R < 1$, a contradiction. Hence $c|P_*| \le n$, establishing the bound when c doubles.

We can now show that each correct sender passes every check and stays outside F. By the bound above and the choice of d, the request passes the numerical checks of line 19. A correct q signs one request per attempt, so the check of line 21 passes. As in Claim 9, $|\mathcal{G}_i \times [d]| \le |W| - |A|$. By validity and split (Theorem 8), its d indices are non-$\perp$, pairwise distinct, and belong to $W \setminus A$. Thus, the sender stays outside F at line 10, receives a query with $L \subseteq W$ and $|L| = d$, and sends the requested data, staying outside F at line 14. □

Claim 12 (Delivery)**.** *If q and some sender are correct, then $|V| \ge M$.*

Proof. By Claim 11, a correct sender stays outside F. Resolution (Claim 10) therefore implies $|V| \ge M$. □

Claim 13. *Let q be correct. Suppose a scale with parameter c starts with p senders and deficit D, and completes all R attempts with p' senders and deficit $D' > 0$. Then*

$$p' \le p/4 \qquad \text{or} \qquad D' \le D - cp'.$$

Proof. Suppose $p' > p/4$ and $D' > D - cp'$. The p' final senders survive every attempt. Write δ for an attempt's deficit. If $\delta \le c$, then $d = \delta$ and every group is a singleton, so any final sender completes retrieval (Claim 9). Hence $\delta > c$ and $d = c$ in every attempt. If there is a single group, an attempt adds at least cp' new values, contradicting $D' > D - cp'$.

Consider an attempt with $p_j \le p$ senders, group size $m = \lfloor \delta/c \rfloor$, and $g = \lceil p_j/m \rceil$ groups. Since $g \ge 2$, $p_j \ge m + 1$, and

$$g \le \frac{p_j + m - 1}{m} \le \frac{2p_j}{m+1} \le \frac{2p}{m+1}.$$

Some group contains at least p'/g final senders. By Claim 9, their responses add at least

$$\frac{cp'}{g} \ge \frac{cp'(m+1)}{2p} > \frac{\delta}{8}$$

new values, using $p' > p/4$ and $\delta < c(m+1)$. Thus every attempt reduces the positive deficit by a factor smaller than $7/8$. After $R = 8k + 8$ attempts, $D' < n(7/8)^R < 1$, a contradiction. □

Claim 14 (Complexity of the receiver)**.** Ladder *has at most $E = (k+1)R(2k+3) = O(k^3)$ rounds, and a correct receiver with inputs A, P sends $O((|P| + |V \setminus A|)k^2(\lambda + k))$ bits.*

Proof. There are $(k+1)R$ attempts of $2k + 3$ rounds. This schedule bounds every correct party's execution, independently of M and the receiver's behavior, giving the stated E.

If q sends no requests, its communication is zero. Otherwise, let S be the last scale in which it sends a request. For $0 \le s \le S$, let P_s be the senders at the start of scale s, a_s the number of new values added during it, and $x_s = 2^s|P_s|$. Each scale $s < S$ completes all R attempts with $|A| < M$. Applying Claim 13 with $c = 2^s$, $p = |P_s|$, $p' = |P_{s+1}|$, and $D - D' = a_s$ gives

$$x_{s+1} \le \max\left\{\frac{x_s}{2}, 2a_s\right\} \le \frac{x_s}{2} + 2a_s.$$

Since $x_0 = |P|$, summing yields

$$\sum_{s=0}^{S} x_s \le |P| + \frac{1}{2}\sum_{s=0}^{S} x_s + 2\sum_{s=0}^{S} a_s, \qquad \sum_{s=0}^{S} x_s \le 2|P| + 4|V \setminus A|.$$

Here the sets of values added in different scales are disjoint. Each attempt at scale s requests at most x_s values, since $d \le 2^s$ and P only shrinks. Thus the total number of requested values is at most $R\sum_s x_s$, including a possibly partial last scale.

Per requested value, q sends $O(k(\lambda + k))$ bits: a signed request message and an opening of $O(\lambda k)$ bits per sender and attempt (line 7), $O(k^2)$ bits per virtual sender of Bisect (line 9, Theorem 8), and $O(k)$ bits per value in query messages (line 11). Since $R = O(k)$, the bound follows. □

Claim 15 (Complexity of the senders)**.** *For any receiver, correct senders send $O(nk^4(\lambda + k))$ bits in total.*

Proof. Fix an attempt at scale s, and let U be the set of correct senders that pass the comparison check in it (line 21). For a request message μ, let U_μ be the set of parties in U that received μ (line 18); these sets partition U. Each party in U_μ holds an opening of its position in the list of length m committed in μ, and $m \le 2n/c$ (line 19); by binding, $|U_\mu| \le m \le 2n/c$. For $\mu \ne \mu'$, no edge of G_c joins U_μ and $U_{\mu'}$: a party in U_μ sends $\langle \text{compare}, \mu\rangle$ to all its neighbors (line 20), and a neighbor in $U_{\mu'}$ would have returned on receiving it (line 21). So by Lemma 2, no two disjoint unions of sets U_μ both have at least n/c parties. If some U_μ has at least n/c parties, then $|U \setminus U_\mu| < n/c$, so $|U| < 3n/c$. Otherwise, if $|U| \ge 3n/c$, adding sets U_μ one at a time yields a union X with $n/c \le |X| < 2n/c$, and then $|U \setminus X| > n/c$, a contradiction. Hence $c|U| < 3n$.

At scale 0, at most n correct senders pass the check of line 19 in an attempt. At scale $s > 0$, every such sender passed the comparison check in the preceding scale: otherwise, it would have returned (lines 18–21). Taking the union over that scale's R attempts gives fewer than $R \cdot 3n/(c/2) = 6Rn/c$ such senders. Each of them sends a compare message, of $O(\lambda + k)$ bits, to each of its $O(ck)$ neighbors in G_c (line 20), for $O(Rnk(\lambda + k))$ bits per attempt. Each party in U sends $O(dk^2)$ bits in Bisect (line 22) and $O(d(\lambda + k))$ bits in data messages (line 24), with $d \le c$ (line 19), for $O(n(\lambda + k^2))$ bits per attempt, as $c|U| < 3n$. Correct senders send no other messages, so over the $(k + 1)R$ attempts, this is $O(nk^4(\lambda + k))$ bits. □

Proof of Theorem 3. A correct receiver outputs $V \supseteq A$ and $F \subseteq P$, since A only grows (line 13) and F only gains members of P (lines 14 and 10). Claims 10 and 12 give resolution and delivery against every t-bounded adversary with $t < n$, and Claims 14 and 15 give the bounds. □

3.4 Splitting

In this section, we prove Theorem 8. Let $I^\varepsilon = [n]$ for the empty string ε, and for $\sigma \in \{0,1\}^h$ with $h < k$, let $I^{\sigma 0}$ and $I^{\sigma 1}$ be the lower and upper halves of I^σ. For each $h \le k$, the sets I^σ with $\sigma \in \{0,1\}^h$ form *layer* h, which splits $[n]$ into 2^h intervals of size $n/2^h$. In round $2h$, q moves each sender outside F from layer $h - 1$ to layer h, breaking ties in line 20 by the order on P.

Claim 16. *Let q be correct and $h \in [k]$. After round $2h$, every $x \in P \setminus F$ claims to hold at least as many values outside A in I^{σ_x} as there are senders $y \in P \setminus F$ with $\sigma_y = \sigma_x$, i.e.:*

$$w_x - |I^{\sigma_x} \cap A| \;\ge\; |\{y \in P \setminus F : \sigma_y = \sigma_x\}|.$$

Proof. The invariant holds at initialization, at layer 0, since $w_x - |A| = |P|$. Suppose it holds before round $2h$. Fix $\sigma \in \{0,1\}^{h-1}$, and let v be the number of senders at σ after the checks of round $2h$ (line 17). Removing senders only decreases this number, so every survivor satisfies

$$\ell_x + r_x - |I^\sigma \cap A| \ge w_x - |I^\sigma \cap A| \ge v.$$

Protocol 4: Bisect

Sender x, with input W

1: $\sigma \leftarrow \varepsilon$.
2: **for** $h = 1, \dots, k$ **do**
3: **Round** $2h-1$**:**
4: **if** q sent $\langle \text{node}, \sigma' \rangle$ with $\sigma' \in \{\sigma 0, \sigma 1\}$: $\sigma \leftarrow \sigma'$.
5: $\ell \leftarrow |W \cap I^{\sigma 0}|$; $r \leftarrow |W \cap I^{\sigma 1}|$.
6: Send $\langle \text{count}, \ell, r \rangle$ to q.
7: **Round** $2h$**:**
8: Wait.

Receiver q, with input A, P

9: $F \leftarrow \emptyset$; $\sigma_x \leftarrow \varepsilon$, $w_x \leftarrow |A| + |P|$ for $x \in P$.
10: **for** $h = 1, \dots, k$ **do**
11: **Round** $2h-1$**:**
12: Wait.
13: **Round** $2h$**:**
14: **for** $x \in P \setminus F$ **do**
15: Receive $\langle \text{count}, \ell_x, r_x \rangle$ from x.
16: **if** no count message was received from x, $\ell_x > n/2^h$, $r_x > n/2^h$, or $\ell_x + r_x < w_x$:
17: $F \leftarrow F \cup \{x\}$.
18: **for** $\sigma \in \{0,1\}^{h-1}$ **do**
19: $z \leftarrow |I^{\sigma 0} \cap A|$.
20: **for** $x \in P \setminus F$ with $\sigma_x = \sigma$, by decreasing ℓ_x **do**
21: **if** $\ell_x > z$: $\sigma_x \leftarrow \sigma 0$; $w_x \leftarrow \ell_x$; $z \leftarrow z + 1$.
22: **else**: $\sigma_x \leftarrow \sigma 1$; $w_x \leftarrow r_x$.
23: **if** $h < k$: send $\langle \text{node}, \sigma_x \rangle$ to each $x \in P \setminus F$.
24: Output $(j_x)_{x \in P}$, with $I^{\sigma_x} = \{j_x\}$ for $x \notin F$, and $j_x = \perp$ for $x \in F$.

Let Z of these senders move left to $\sigma 0$.

If $Z > 0$, consider the last sender x_* sent left. At its turn, $z = |I^{\sigma 0} \cap A| + Z - 1$. The strict test $\ell_{x_*} > z$ therefore gives $\ell_{x_*} \geq |I^{\sigma 0} \cap A| + Z$. Since senders are processed in decreasing order of ℓ_x, every sender sent left has $\ell_x \geq \ell_{x_*}$, and hence $\ell_x - |I^{\sigma 0} \cap A| \geq Z$.

A sender sent right satisfies $\ell_x \leq z \leq |I^{\sigma 0} \cap A| + Z$ at its turn. Substituting this bound into the parent inequality gives

$$r_x - |I^{\sigma 1} \cap A| \geq v + |I^{\sigma 0} \cap A| - \ell_x \geq v - Z.$$

The updated w_x is ℓ_x on the left and r_x on the right, establishing the invariant in both children. □

Claim 17 (Split)**.** *If q is correct, the non-$\perp$ indices j_x are pairwise distinct and belong to $[n] \setminus A$.*

Proof. Let $x \in P \setminus F$ after round $2k$. Then $I^{\sigma_x} = \{j_x\}$, and $w_x \leq n/2^k = 1$. By Claim 16 for $h = k$, at most $1 - |\{j_x\} \cap A| \leq 1$ sender has σ_x, which is x. So the indices j_x, $x \in P \setminus F$, are pairwise distinct and belong to $[n] \setminus A$. □

Claim 18 (Validity)**.** *If q is correct, every correct $x \in P$ satisfies $j_x \neq \perp$ and $j_x \in W_x \setminus A$.*

Proof. Fix a correct sender x. We show by induction that it passes every check. Both parties start at the root, and each subsequent node message selects a child of their preceding interval. Thus x's counts describe the receiver's current interval and satisfy $\ell_x, r_x \leq n/2^h$ and $\ell_x + r_x = |W_x \cap I^{\sigma_x}| \geq w_x$. Initially, the last inequality follows from the input condition $w_x = |A| + |P| \leq |W_x|$; thereafter, w_x equals x's previous count for the selected child. Hence x passes the check of round $2h$ (line 17) and stays outside F.

After the final selection, $I^{\sigma_x} = \{j_x\}$ and $w_x = |W_x \cap \{j_x\}|$. The invariant (Claim 16) gives $w_x \geq 1$, so $j_x \in W_x$; Claim 17 gives $j_x \notin A$. Thus $j_x \neq \perp$ and $j_x \in W_x \setminus A$. □

Claim 19 (Complexity)**.** Bisect *has* $2k$ *rounds, a correct receiver sends* $O(|P|k^2)$ *bits, and each correct sender sends* $O(k^2)$ *bits.*

Proof. There are k iterations of two rounds. Per iteration, q sends at most one node message to each sender (line 23), and each correct sender sends one count message to q (line 6). The string in a node message has fewer than k bits, and the entries of a count message are at most n, so each message has $O(k)$ bits. □

Proof of Theorem 8. Claims 17 and 18 hold against every t-bounded adversary with $t < n$, and Claim 19 gives the bounds. □

Acknowledgments

Yuval Efron's research is supported by the Institute for Advanced Study and a Kovner Foundation fellowship.

AI Disclosure

ChatGPT Dot substantially assisted with protocol design and proof development. Codex and Claude Opus 5.5 assisted with manuscript revision and proof checking.

References

[ACD+19] Ittai Abraham, T.-H. Hubert Chan, Danny Dolev, Kartik Nayak, Rafael Pass, Ling Ren, and Elaine Shi. "Communication Complexity of Byzantine Agreement, Revisited". In: *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing (PODC)*. ACM, 2019, pp. 317–326. DOI: 10.1145/3293611.3331629.

[ADD+19] Ittai Abraham, Srinivas Devadas, Danny Dolev, Kartik Nayak, and Ling Ren. "Synchronous Byzantine Agreement with Expected $O(1)$ Rounds, Expected $O(n^2)$ Communication, and Optimal Resilience". In: *Financial Cryptography and Data Security (FC)*. Springer, 2019, pp. 320–334. DOI: 10.1007/978-3-030-32101-7_20.

[ALP+25] Andreea B. Alexandru, Julian Loss, Charalampos Papamanthou, Giorgos Tsimos, and Benedikt Wagner. "Sublinear-Round Broadcast without Trusted Setup". In: *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM, 2025, pp. 4132–4171. DOI: 10.1137/1.9781611978322.141.

[AMS19] Ittai Abraham, Dahlia Malkhi, and Alexander Spiegelman. "Asymptotically Optimal Validated Asynchronous Byzantine Agreement". In: *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing (PODC)*. ACM, 2019, pp. 337–346. DOI: 10.1145/3293611.3331612.

[BGP92] Piotr Berman, Juan A. Garay, and Kenneth J. Perry. "Bit Optimal Distributed Consensus". In: *Computer Science: Research and Applications*. Ed. by Ricardo Baeza-Yates and Udi Manber. Boston, MA: Springer, 1992, pp. 313–321. DOI: 10.1007/978-1-4615-3422-8_27.

[Bra87] Gabriel Bracha. "Asynchronous Byzantine Agreement Protocols". In: *Information and Computation* 75.2 (1987), pp. 130–143. DOI: 10.1016/0890-5401(87)90054-X.

[CDG+22] Pierre Civit, Muhammad Ayaz Dzulfikar, Seth Gilbert, Vincent Gramoli, Rachid Guerraoui, Jovan Komatovic, and Manuel Vidigueira. "Byzantine Consensus Is $\Theta(n^2)$: The Dolev–Reischuk Bound Is Tight Even in Partial Synchrony!" In: *36th International Symposium on Distributed Computing (DISC)*. Vol. 246. Leibniz International Proceedings in Informatics (LIPIcs). Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022, 14:1–14:21. DOI: 10.4230/LIPIcs.DISC.2022.14.

[CDG+24] Pierre Civit, Muhammad Ayaz Dzulfikar, Seth Gilbert, Rachid Guerraoui, Jovan Komatovic, and Manuel Vidigueira. "DARE to Agree: Byzantine Agreement With Optimal Resilience and Adaptive Communication". In: *Proceedings of the 43rd ACM Symposium on Principles of Distributed Computing (PODC)*. ACM, 2024, pp. 145–156. DOI: 10.1145/3662158.3662792.

[CDG+25] Pierre Civit, Muhammad Ayaz Dzulfikar, Seth Gilbert, Rachid Guerraoui, Jovan Komatovic, Manuel Vidigueira, and Igor Zablotchi. "Partial Synchrony for Free: New Upper Bounds for Byzantine Agreement". In: *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM, 2025, pp. 4227–4291. DOI: 10.1137/1.9781611978322.144.

[CDL+26] Daniel Collins, Sisi Duan, Julian Loss, Charalampos Papamanthou, Giorgos Tsimos, and Haochen Wang. "Towards Optimal Parallel Broadcast Under a Dishonest Majority". In: *Financial Cryptography and Data Security (FC 2025)*. Springer, 2026, pp. 345–363. DOI: 10.1007/978-3-032-07024-1_20.

[CKS05] Christian Cachin, Klaus Kursawe, and Victor Shoup. "Random Oracles in Constantinople: Practical Asynchronous Byzantine Agreement Using Cryptography". In: *Journal of Cryptology* 18.3 (2005), pp. 219–246. DOI: 10.1007/s00145-005-0318-0.

[CKS22] Shir Cohen, Idit Keidar, and Alexander Spiegelman. "Make Every Word Count: Adaptive Byzantine Agreement with Fewer Words". In: *26th International Conference on Principles of Distributed Systems (OPODIS 2022)*. Ed. by Eshcar Hillel, Roberto Palmieri, and Etienne Rivière. Vol. 253. Leibniz International Proceedings in Informatics (LIPIcs). Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2022, 18:1–18:21. DOI: 10.4230/LIPIcs.OPODIS.2022.18.

[CPS20] T.-H. Hubert Chan, Rafael Pass, and Elaine Shi. "Sublinear-Round Byzantine Agreement Under Corrupt Majority". In: *Public-Key Cryptography (PKC)*. Springer, 2020, pp. 246–265. DOI: 10.1007/978-3-030-45388-6_9.

[CW92] Brian A. Coan and Jennifer L. Welch. "Modular Construction of a Byzantine Agreement Protocol with Optimal Message Bit Complexity". In: *Information and Computation* 97.1 (1992), pp. 61–85. DOI: 10.1016/0890-5401(92)90004-Y.

[DKL25] Giovanni Deligios, Ivana Klasovitá, and Chen-Da Liu-Zhang. "Asymptotically Optimal Early Termination for Dishonest Majority Broadcast". In: *Advances in Cryptology (EUROCRYPT)*. Springer, 2025, pp. 116–144. DOI: 10.1007/978-3-031-91092-0_5.

[DR85] Danny Dolev and Rüdiger Reischuk. "Bounds on Information Exchange for Byzantine Agreement". In: *Journal of the ACM* 32.1 (1985), pp. 191–204. DOI: 10.1145/2455.214112.

[DRS90] Danny Dolev, Rüdiger Reischuk, and H. Raymond Strong. "Early Stopping in Byzantine Agreement". In: *Journal of the ACM* 37.4 (1990), pp. 720–741. DOI: 10.1145/96559.96565.

[DS83] Danny Dolev and H. Raymond Strong. "Authenticated Algorithms for Byzantine Agreement". In: *SIAM Journal on Computing* 12.4 (1983), pp. 656–666. DOI: 10.1137/0212045.

[ER25] Yuval Efron and Ling Ren. *Lifeline: Optimal Byzantine Agreement Under Minimal Synchrony*. Cryptology ePrint Archive, Paper 2025/1318. 2025. URL: https://eprint.iacr.org/2025/1318.

[ETP24] Fatima Elsheimy, Giorgos Tsimos, and Charalampos Papamanthou. "Deterministic Byzantine Agreement with Adaptive $O(n \cdot f)$ Communication". In: *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM, 2024, pp. 1120–1146. DOI: 10.1137/1.9781611977912.43.

[FM97] Pesech Feldman and Silvio Micali. "An Optimal Probabilistic Protocol for Synchronous Byzantine Agreement". In: *SIAM Journal on Computing* 26.4 (1997), pp. 873–933. DOI: 10.1137/S0097539790187084.

[GKK+07] Juan A. Garay, Jonathan Katz, Chiu-Yuen Koo, and Rafail Ostrovsky. "Round Complexity of Authenticated Broadcast with a Dishonest Majority". In: *48th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2007, pp. 658–668. DOI: 10.1109/FOCS.2007.44.

[KK09] Jonathan Katz and Chiu-Yuen Koo. "On Expected Constant-Round Protocols for Byzantine Agreement". In: *Journal of Computer and System Sciences* 75.2 (2009), pp. 91–112. DOI: 10.1016/j.jcss.2008.08.001.

[KS11] Valerie King and Jared Saia. "Breaking the $O(n^2)$ Bit Barrier: Scalable Byzantine Agreement with an Adaptive Adversary". In: *Journal of the ACM* 58.4 (2011), 18:1–18:24. DOI: 10.1145/1989727.1989732.

[LN24] Julian Loss and Jesper Buus Nielsen. "Early Stopping for Any Number of Corruptions". In: *Advances in Cryptology (EUROCRYPT)*. Springer, 2024, pp. 457–488. DOI: 10.1007/978-3-031-58734-4_16.

[LSP82] Leslie Lamport, Robert E. Shostak, and Marshall C. Pease. "The Byzantine Generals Problem". In: *ACM Transactions on Programming Languages and Systems* 4.3 (1982), pp. 382–401. DOI: 10.1145/357172.357176.

[Mer80] Ralph C. Merkle. "Protocols for Public Key Cryptosystems". In: *1980 IEEE Symposium on Security and Privacy*. IEEE, 1980, pp. 122–134. DOI: 10.1109/SP.1980.10006.

[MR21] Atsuki Momose and Ling Ren. "Optimal Communication Complexity of Authenticated Byzantine Agreement". In: *35th International Symposium on Distributed Computing (DISC)*. 2021, 32:1–32:16. DOI: 10.4230/LIPIcs.DISC.2021.32.

[PSL80] Marshall C. Pease, Robert E. Shostak, and Leslie Lamport. "Reaching Agreement in the Presence of Faults". In: *Journal of the ACM* 27.2 (1980), pp. 228–234. DOI: 10.1145/322186.322188.

[TLP22] Georgios Tsimos, Julian Loss, and Charalampos Papamanthou. "Gossiping for Communication-Efficient Broadcast". In: *Advances in Cryptology (CRYPTO)*. Springer, 2022, pp. 439–469. DOI: 10.1007/978-3-031-15982-4_15.

[WXS+20] Jun Wan, Hanshen Xiao, Elaine Shi, and Srinivas Devadas. "Expected Constant Round Byzantine Broadcast Under Dishonest Majority". In: *Theory of Cryptography Conference (TCC)*. Springer, 2020, pp. 381–411. DOI: 10.1007/978-3-030-64375-1_14.