Electronic Colloquium on Computational Complexity, Report No. 334 (2026)

Multi-Party Error Resilience beyond $\frac{1}{n}$

Klim Efremenko*
Ben-Gurion University

Gillat Kol†
Princeton University

Raghuvansh R. Saxena‡
Tata Institute of
Fundamental Research

Zhijun Zhang§
INSAIT, Sofia University
"St. Kliment Ohridski"

Abstract

In the *information dissemination* problem, n parties each hold a unique piece of information and wish to learn all the information. They communicate over an *adversarial broadcast channel*, where an adversary may corrupt a fraction of the received transmissions. What is the largest fraction of errors to which an information dissemination protocol can be resilient?

In adversarial multi-party communication models, $\frac{1}{n}$ is a natural upper bound on error resilience, and prior work has therefore largely avoided these models. The reason is a simple attack: with only a $\frac{1}{n}$-fraction of errors, the adversary can corrupt all messages sent by the party that broadcasts least, making information dissemination impossible.

We observe, however, that this attack applies only to *non-adaptive* protocols, a restricted class of protocols with a fixed speaking order. Can an *adaptive* protocol tolerate a constant fraction of errors, or does the $\frac{1}{n}$ barrier persist?

We find that the truth lies in the middle. Our main result shows that the optimal error resilience of general adaptive information dissemination protocols over the broadcast channel is $\Theta(\frac{1}{\sqrt{n}})$. We give a protocol achieving this resilience and a matching impossibility result that holds even against erasure noise. To the best of our knowledge, this is the first impossibility result for multi-party adversarial noise in the adaptive model.

*klimefrem@gmail.com. Supported by the European Research Council Grant number: 949707.
†gillat.kol@gmail.com.
‡raghuvansh.saxena@gmail.com. Supported by the Department of Atomic Energy, Government of India, under project no. RTI4014.
§zhijun.zhang@insait.ai. This research was partially funded by the Ministry of Education and Science of Bulgaria (support for INSAIT, part of the Bulgarian National Roadmap for Research Infrastructure).

ISSN 1433-8092

Contents

1 Introduction **1**
- 1.1 Our Results . . . 2
- 1.2 Our Technique . . . 3
- 1.3 Related Work . . . 5
- 1.4 Additional Discussion . . . 7

2 Proof Overview **8**
- 2.1 Overview of Our Protocol . . . 8
  - 2.1.1 An $\mathcal{O}(n^2)$ Protocol for Erasures . . . 8
  - 2.1.2 Resilience to Erasures with Improved Rate . . . 9
  - 2.1.3 From Erasures to Corruptions . . . 11
- 2.2 Overview of the Impossibility Result . . . 15

3 Model and Preliminaries **17**
- 3.1 Notation and Preliminaries . . . 17
- 3.2 Error Correcting Codes . . . 18
- 3.3 Communication Model . . . 18

4 The Protocol for InfoDis Against Corruptions **20**
- 4.1 Proof of Lemma 4.5 . . . 25
- 4.2 Generalization to Multi-Bit Inputs . . . 29

5 Hardness of InfoDis Against Erasures **30**

A Interactive Coding **37**

1 Introduction

Our goal in this paper is to advance the study of *multi-party coding* in *adversarial* settings. We consider n parties communicating over a *broadcast channel*, where each message sent by one party is transmitted to all the others. An adversary with full knowledge of the parties' inputs and the communication transcript may corrupt a fixed fraction of the received symbols.

Recent work has explored multi-party channels under a variety of noise models, predominantly assuming *stochastic* noise, where each received bit is erased or corrupted with a small probability. Adversarial noise has received less attention, largely due to a key barrier highlighted by several works: no scheme, regardless of its length, could tolerate an adversarial corruption rate, or even an erasure rate, above $\frac{1}{n}$. As noted in [CHGH18], "*it is easy to show that the maximal adversarial noise fraction that any resilient protocol can cope with is* $\Theta(\frac{1}{n})$*. Indeed, if more than* $\frac{1}{n}$*-fraction of the messages are corrupted, then the noise can completely corrupt all the communication of the node that sends the least number of messages*"; also see [JKL15, HS16, ADHS20]. Indeed, with an erasure budget of $\frac{1}{n}$, the adversary can erase every message sent by the party that communicates least, preventing the parties from solving any task whose output depends on that party's input.

In this context, the coding literature distinguishes between two main broadcast models: *non-adaptive* (also called *oblivious* or *static*) and *adaptive*. In a non-adaptive protocol, exactly one party broadcasts in each round, and the speaking order is fixed in advance, independently of the parties' inputs and received transcripts [Gam87]. Thus, the party that speaks least can be identified before the protocol begins, and the attack above succeeds.

We instead study the more general model of adaptive protocols, in which a party's decision to broadcast, and the content of its message, may depend on its input and received transcript. This model goes back to the well-studied *radio networks* framework in distributed computing [CK85], which was introduced to capture *wireless communication*. Here, the identity of the least-speaking party may itself depend on the adversary's actions, and the attack above no longer applies. This raises our central question: can adaptive broadcast protocols tolerate an adversarial error rate of $\omega(\frac{1}{n})$?

We study this question through the *information dissemination* problem, in which each party begins with one input and every party must learn all n inputs. Information dissemination is perhaps the most basic multi-party communication problem and has been studied in various models. It is also "complete" for the error resilience question: a protocol that disseminates all inputs while tolerating some fraction of adversarial errors immediately yields a protocol for any other communication task with the same resilience. The parties first run the dissemination protocol to learn all inputs and then locally compute the desired output.

This generic reduction may incur substantial communication overhead when the inputs are much longer than the communication required by the original task. However, in Section 1.4 (also see Appendix A), we argue that this overhead is generally unavoidable, further motivating information dissemination as a natural benchmark for studying maximum error

resilience in multi-party communication.

1.1 Our Results

The main result gives a tight asymptotic characterization of the maximum error resilience achievable for adaptive information dissemination.

Theorem 1.1 (informal, see Theorems 4.1 and 5.1)**.** *The maximum fraction of adversarial corruptions that can be tolerated by an information dissemination protocol is* $\Theta(\frac{1}{\sqrt{n}})$.

The proof of the impossibility result in Theorem 1.1 is stronger than stated: it holds even in the simpler setting of erasure noise and information dissemination with single-bit inputs. To the best of our knowledge, this is the first impossibility result for multi-party adversarial noise in the adaptive model. As is often the case, proving impossibility results for adaptive protocols is significantly more challenging because of the circularity issue discussed above.

Our protocol in Theorem 1.1 does not have constant rate: information dissemination takes only n rounds over a noiseless broadcast channel, whereas our protocol takes $\tilde{O}_\eta(n^{3/2+\eta})$ rounds, for any $\eta > 0$.

The communication channel. As discussed above, we study adaptive protocols, in which each party decides in every round whether to broadcast based on all information available to them. A round may therefore contain a single broadcast, multiple simultaneous broadcasts (a collision), or no broadcasts (silence).

We work in the most commonly studied *collision-as-silence* (CAS) model, also known as the *radio-network model without collision detection* [CK85]; also see [PR09]. Among standard radio-network models, CAS makes a minimal assumption about the feedback available to receivers: when exactly one party broadcasts, its message is received; otherwise, the parties receive only the special symbol '$\perp$'. Thus, a collision conveys no information beyond the failure to decode a unique message. This conservative abstraction models a collision as interference, without making any assumptions about the information conveyed by that interference or even about the ability to distinguish it from silence.

The information-dissemination protocol in Theorem 1.1 operates in the CAS model and allows the adversary to corrupt the symbols received by different parties differently. Subject to its budget, the adversary may flip any subset of the received bits, replace received bits with '$\perp$', or replace '$\perp$' with bits.

Observe that once the noise fraction reaches $\frac{1}{n}$, even if restricted entirely to erasures, no protocol can guarantee correctness for all parties: the adversary can erase every symbol received by a single party, leaving its transcript independent of the other parties' inputs. Thus, no meaningful correctness guarantee can be imposed on parties whose incoming communication is too heavily corrupted. Following [EKPS21a], we therefore say that a protocol tolerates a δ-fraction of adversarial errors if, whenever the total fraction of errors is at most δ, every party whose received transcript is corrupted in no more than a 10δ-fraction

of the rounds produces the correct output. The constant 10 is arbitrary; by averaging, this choice guarantees that at least 90% of the parties produce the correct output.

Fix an execution of an information-dissemination protocol in which the total fraction of errors is at most δ, and let S be the set of parties whose received transcripts are corrupted in no more than a 10δ-fraction of the rounds. The correctness condition above requires every party in S to learn the inputs of all n parties, rather than only the inputs of the parties in S. This distinction is important because it allows the parties in S to locally compute any desired function of the *entire input vector*, including functions that are sensitive to a change in even one or a few inputs, such as the AND and OR functions.[1] This would not be possible if they learned only the inputs of the parties in S.

1.2 Our Technique

We next highlight a few main ideas in our protocol construction. Detailed overviews of our protocol and impossibility result can be found in Section 2.

Blueprint: giving endangered inputs a voice. At a high level, our protocol gives "endangered inputs", inputs that have not been heard often enough by other parties, more airtime, even when nobody knows which inputs are endangered.

We start with a simple three-phase *erasure* protocol that achieves resilience $\Omega(\frac{1}{\sqrt{n}})$ in $O(n^2)$ rounds. In the first phase, each party repeatedly broadcasts its input bit. In the second phase, each party broadcasts an encoding of its current view of all inputs. Finally, only $O(\sqrt{n})$ parties, "the chosen few", rebroadcast, including every party whose input has not spread widely enough. A party joins this set if the encodings it decodes in the second phase indicate that its bit was not sufficiently disseminated in the first phase, or if it cannot decode enough encodings to make this determination.

The main technical work lies in handling corruptions and reducing the protocol's length.

Better rate: fewer speakers, larger encodings. The bottleneck is the exchange of entire views in the second phase. We instead have parties forward an encoding of a selected subset of their bits. We want each bit to be heard roughly the same number of times. However, if we fix these subsets in advance, the adversary can target the parties assigned to forward a particular bit i by corrupting the messages they receive from party i during the first phase. We therefore select the bits dynamically by the *least-heard-first* ranking rule: each party prioritizes the bits it has heard least often among those it knows.

To reduce the length of the protocol, we gradually trade fewer speakers for larger encodings. Larger encodings are more expensive to erase, and each well-disseminated bit continues to appear in many encodings. Since a party learns a bit as soon as it decodes any encoding

[1] For example, suppose that each input is a bit indicating whether the corresponding component of a distributed system has detected a fault. A party in S can conclude that the system is fault-free only after learning all n bits, since a single missing bit could conceal the only fault.

containing it, the adversary can keep each such bit unknown to fewer receivers. We can therefore restrict the next iteration to a smaller group in which each such bit is still known to many parties. With fewer speakers, each can encode a larger subset within the same round budget. We repeat this process until the remaining speakers can encode all n bits.

Only $O(\sqrt{n})$ bits can be poorly disseminated after the first phase, since each requires erasing an $\Omega(\frac{1}{n})$-fraction of all received symbols. Every other bit is known to a strong majority of the final speakers, so receivers that decode enough of their encodings learn all but these $O(\sqrt{n})$ exceptional bits.

Corruptions: counting votes and silences. Moving from erasures to corruptions turns the task of balancing how often each bit is heard into a voting problem. Fix a party j and a bit i, and let b denote party j's current estimate of bit i, initially set to the majority value received from party i in the first phase. In each iteration of the second phase, party j ranks bit i by its *vote cushion*: the number of broadcasts it has received so far in that iteration supporting the value b for bit i, minus the number supporting $\bar{b}$. Bits with the smallest vote cushions receive priority, as the party considers them most at risk of being set to the wrong value.

However, a difficulty arises when the correct value of bit i has already built a strong lead. As in real elections, parties holding this value may refrain from broadcasting it because it is already winning, leaving parties with the incorrect value free to overturn the vote. Our remedy is to interpret *silence as compliance*: we also track how long each value maintains a strong lead. A long stretch without successful objections provides evidence in favor of the leading value that a late burst of incorrect votes cannot erase.

Scheduling: aiming for disagreement. At the end of the second phase, almost all parties know every bit, except for a set of $O(\sqrt{n})$ troublesome bits, which we call *brats*. We design a scheduling mechanism that allows brats to be heard again without first identifying their owners. To do so, we use a family of overlapping groups of size $O(\sqrt{n})$ such that, for each brat, an $\Omega(\frac{1}{\sqrt{n}})$-fraction of the groups contain its owner and every other member knows the correct values of all the group's inputs except possibly the brat's.

To illustrate our scheduling mechanism, suppose that all other members of such a group agree on the inputs of all other group members but hold the wrong value for the brat. Each member uses a balanced encoding of its current view of the group's inputs as a speaking schedule, broadcasting at the encoding's 1-positions. Members with the same view broadcast together, causing collisions. Since the brat's owner knows its own input, its view differs from theirs. The balanced code guarantees a constant fraction of positions where the owner's encoding is 1 and the others' encoding is 0, allowing the owner to broadcast alone. Thus, agreement creates collisions while disagreement creates airtime, without anyone identifying who needs to be heard.

Supermajority: embracing certainty. Whenever a brat's input still needs to be heard, the scheduling mechanism requires its owner's view to differ from every view held by the other group members. Even if these members agree on all other inputs, they may split between 0 and 1 for the brat. The owner's view may then coincide with that of the members holding the correct value. Members with identical views follow the same speaking schedule, causing collisions whenever the owner broadcasts.

We therefore introduce a third value, *unsure*. At the end of the second phase, a party j sets its view of bit i to 0 or 1 only if that value has at least a $\frac{2}{3}$-majority in the final iteration; otherwise, it marks the bit as unsure. In a suitable group, if any member other than party i has the correct value of bit i, that value has enough support to be recovered by the final majority vote, so no additional airtime is needed for this bit. In the remaining case, all other members hold either the incorrect value or unsure, so the owner's view differs from all of theirs.

1.3 Related Work

Broadcast with limited adversaries. Adversarial noise in multi-party broadcast channels was recently studied by [EKP+24], with the goal of circumventing the inherent $\frac{1}{n}$ barrier on resilience described earlier. To bypass this barrier, [EKP+24] restrict the adversary to corrupting at most a constant fraction of both the incoming and the outgoing communication of every party. After ruling out these "trivial" isolation attacks, they give a simple information-dissemination protocol that tolerates a constant fraction of adversarial corruptions. Their protocol requires $\Theta(n^2)$ rounds, a factor of $\Theta(n)$ more than the noiseless n-round protocol, and their main result shows that this quadratic round complexity is optimal.

A key difference between our work and [EKP+24], in addition to their restriction on the adversary, is that [EKP+24] consider only the *non-adaptive* model. As they note, it is unclear how to extend their definition of resilience to adaptive protocols. In particular, their outgoing-corruption constraint requires that, for each party, the adversary corrupt at most a δ-fraction of all symbols received during rounds in which that party broadcasts. In an adaptive protocol, however, the number of rounds in which a party broadcasts may itself depend on the noise encountered during the execution, so the denominator in this constraint is no longer fixed independently of the adversary.

Information dissemination over the stochastic broadcast channel. The information-dissemination problem studied in this work generalizes the classical coding task to a multi-sender, multi-receiver setting. In traditional coding, there is a single sender and a single receiver, and the goal is to transmit the sender's message to the receiver.

El Gamal [Gam87] initiated the study of information dissemination in the non-adaptive noisy broadcast model. He considered stochastic noise, where each bit received by each party is flipped independently with some constant probability $\delta \in (0, 1/2)$. El Gamal asked whether information dissemination could be solved with low communication overhead in this

model. Gallager [Gal88] answered this question with an elegant $\mathcal{O}(n \log \log n)$-round protocol, which was later shown to be optimal by [GKS08]. [GHM18] studied the corresponding stochastic-erasure model and gave an $\mathcal{O}(n \log^* n)$-round protocol. Many subsequent works have studied variants of this stochastic broadcast setting and related noisy radio channels; see, for example, [Gal88, Yao97, KM05, FK00, New04, GKS08, GHM18, CHHHZ17, EKS18, CHHHZ19, EKS19, EKS20b, AGL22, EKPS21a, MG21, EKPS21b, EKPS23b].

Interactive coding. The field of interactive coding, introduced by Schulman in his seminal work [Sch92], studies how to transform protocols designed for noiseless channels into protocols that remain correct in the presence of noise; see [Gel17] for a survey. Schulman's initial work considered the two-party setting with a fixed speaking schedule. Motivated in part by radio-network models from distributed computing, subsequent works studied two-party settings with adaptive speaking patterns or termination rules; see, for example, [Hae14, GHS14, GH14, AGS16, EKS20a, EKS21]. These works show that adaptivity can improve the achievable resilience and, in some settings, the communication efficiency of interactive codes. Although several of these works give lower bounds in adaptive two-party models with adversarial noise, our work is the first to establish such a lower bound in the multi-party broadcast model with adversarial noise.

Noise-resilient multi-party communication has also been studied in a variety of distributed settings, including peer-to-peer networks [RS94, GMS11, JKL15, LV15, HS16, ABE+16, BEGH16, GK19, CHGH19, GKR21, ADHS20, GKR22] and broadcast networks [Gal88, Yao97, KM05, FK00, New04, GKS08, GHM18, CHHHZ17, EKS18, CHHHZ19, EKS19, EKS20b, AGL22, EKPS21a, MG21, EKPS21b, EKPS23a, EKPS23b, EKP+24, EKPS25]. Our work contributes to the latter line.

Peer-to-peer with adversarial noise. In this work, we consider a broadcast channel, where each broadcasting party sends the same message to all other parties. Adversarial noise has previously been studied in various *peer-to-peer* settings, where the parties are nodes of a graph and each party may send different messages to different neighbors.

[HS16] study the synchronous, fully utilized model, in which every node sends one symbol over each incident edge in every round. They give a constant-overhead coding scheme that tolerates a $\Theta(\frac{1}{|E|})$-fraction of adversarial corruptions on an arbitrary graph. When the topology is known to all parties, they route communication over a sparse spanning subgraph and obtain resilience $\Theta(\frac{1}{n})$ with multiplicative communication overhead $\mathcal{O}(\frac{|E| \log n}{n})$.

[JKL15] consider a synchronous, non-fully-utilized communication model. They show that, when the communication graph contains a spanning star, a noiseless protocol can be transformed into one that tolerates a $\Theta(\frac{1}{n})$-fraction of adversarial errors while increasing its length by only a constant factor. [LV15] improve the communication balance of this scheme. [CHGH19] study the asynchronous setting and give a coding scheme with resilience $\Theta(\frac{1}{n})$ and multiplicative communication overhead $\mathcal{O}(n \log^2 n)$.

For the $\Theta(\frac{1}{n})$-resilient schemes above, the dependence on n is optimal up to constant factors in the respective communication models [JKL15, HS16, CHGH19]: an adversary with an error budget of this order can effectively disconnect one party from the rest of the network.

The stronger insertion-deletion-substitution noise model, in which the adversary may insert, delete, or modify messages, is studied by [GKR21, GKR22]. The work of [GKR21] gives an efficient constant-rate scheme tolerating a $\Theta(\frac{1}{|E|})$-fraction of errors when the adversary is oblivious to the parties' shared randomness. The work of [GKR22] removes this obliviousness assumption at the cost of reducing the resilience to $\Theta(\frac{1}{|E|\log|E|})$. Related schemes for adversarial noise over private point-to-point channels are studied in [ADHS20], where the adversary cannot observe the transmitted bits or the parties' private randomness.

1.4 Additional Discussion

Randomness in adversarial models. Much of the work on error-correcting codes over adversarial channels, including the upper and lower bounds in this paper, considers a *deterministic* setting. One may instead consider randomized models in which the parties share a random string. If this string is revealed to the adversary before the protocol begins, then, once the string is fixed, the protocol is effectively deterministic.

At the other extreme, if the string remains hidden from the adversary throughout the protocol, the parties can use portions of it as one-time pads to create a private channel, making the adversary *oblivious* to the contents of their messages. Such adversaries are relatively weak, at least in the peer-to-peer setting: any noiseless protocol can be simulated against them with only constant-factor communication overhead; see, for example, [ADHS20, GKR21].

An interesting direction for future research is to study *intermediate models* in which, for example, fresh shared randomness is generated in each round and immediately revealed to the adversary, but future randomness remains hidden when the adversary chooses which symbols to corrupt or erase. Can such models support protocols with improved noise resilience?

Interactive coding has exponentially small rate. Although general interactive coding and information dissemination have the same optimal resilience, their achievable rates differ sharply. In our model, any general interactive coding scheme achieving the optimal $\Theta(\frac{1}{\sqrt{n}})$ resilience, or even resilience $\omega(\frac{1}{n})$, must incur an exponential blowup over the length of the noiseless protocol, even against erasures alone. We give a sketch of this argument in Appendix A. Therefore, any such coding scheme has exponentially small rate. Put differently, for some protocols, the negligible rate of the naive scheme, which first disseminates all inputs and then performs the computation locally, is essentially optimal. Thus, in the high-noise regime, information dissemination appears to be a more natural object of study.

2 Proof Overview

2.1 Overview of Our Protocol

In this section, we overview the construction of our protocol, building up to it in steps.

2.1.1 An $\mathcal{O}(n^2)$ Protocol for Erasures

We first present a simple information-dissemination protocol that serves as a warm-up and highlights the power of adaptivity. Suppose that each of the n parties initially holds one input bit, and the goal is for every party to recover all n bits. The following three-phase protocol achieves this in $\mathcal{O}(n^2)$ rounds and tolerates an $\Omega\left(\frac{1}{\sqrt{n}}\right)$ fraction of *erasures*.

At a high level, the protocol first has the parties send their inputs and then forward the inputs they have received. It subsequently identifies $\Theta(\sqrt{n})$ parties whose inputs have not been sufficiently disseminated, and only these parties broadcast their inputs again.

Phase 1: "Roll call". Each party i broadcasts its input x_i to all other parties $\mathcal{O}(n)$ times, so the total length of this phase is $\mathcal{O}(n^2)$. After this phase, any party i that received any unerased bit from another party j knows the input x_j of party j. Define the vector $x^{(i)}$ to be an n-length bit vector whose coordinate j is x_j if party i received an unerased bit from party j in this phase, and is $\perp$ if all the bits from party j were erased for party i.

Phase 2: "What's your view?". In the second phase, all the parties i broadcast an encoding of the vector $x^{(i)}$ computed in the first phase using an error-correcting code. Since this vector has length n, transmitting its encoding requires $\mathcal{O}(n)$ rounds per party, for a total of $\mathcal{O}(n^2)$ rounds in this phase.

At the end of this phase, if a party i does not know the input of another party j, there are two possibilities. It may be the case that party i could not decode the encodings of at least a constant fraction of parties, which means that a constant fraction of the symbols received by party i were erased. Recall that in this case party i is not required to output correctly, and therefore this case can be ignored. Nonetheless, note that since we have at most an $\mathcal{O}\left(\frac{1}{\sqrt{n}}\right)$-fraction of erasures, this can happen for at most $\mathcal{O}(\sqrt{n})$ parties i. This bound will be useful later.

The other case is that party i did decode the encodings of a constant fraction of parties, but all these parties did not know x_j after the first phase. This means that a constant fraction of the parties did not receive the bit of party j in the first phase. With an $\mathcal{O}\left(\frac{1}{\sqrt{n}}\right)$-fraction of erasures, this can happen for at most $\mathcal{O}(\sqrt{n})$ parties j.

Phase 3: "The chosen few". Let S be the set of parties whose incoming or outgoing communication suffers a constant fraction of erasures. Formally, define S as the union of the $\mathcal{O}(\sqrt{n})$ parties in the first case above, namely, parties i that received a constant fraction of

erasures in the second phase, and the $\mathcal{O}(\sqrt{n})$ parties in the second case, namely, parties j for which, after the first phase, a constant fraction of the parties are still missing x_j.

Note that each party knows whether or not they are in S: they can check whether or not they are in the first case by computing the fraction of erasures they received in the second phase. If this fraction is small, they can check whether or not a constant fraction of the parties received their bit in the first phase by decoding the encodings they received from those parties in the second phase.

Unlike the previous two phases, the final phase of our protocol is *adaptive*. In this phase there are $\mathcal{O}(n^2)$ rounds and only the parties who are in S may broadcast in these rounds. We divide these rounds into chunks of size $\mathcal{O}(\log n)$ each and allocate chunks to parties so that each party i gets an $\Omega\left(\frac{1}{\sqrt{n}}\right)$-fraction of chunks. Moreover, this allocation has the property that if all the parties in S broadcast in the chunks allocated to them, and the other parties stay silent throughout, then, for all $i \in S$, there exists $\Omega\left(\frac{1}{\sqrt{n}}\right)$-fraction of chunks such that party i is the only party broadcasting in these chunks.

Then, if party i broadcasts an encoding of the pair (i, x_i), by the fact that there is at least $\Omega\left(\frac{1}{\sqrt{n}}\right)$-fraction of chunks where this party is the only one to speak, we know that these broadcasts will be heard by all parties who received no more than an $\mathcal{O}\left(\frac{1}{\sqrt{n}}\right)$-fraction of erasures. So, those parties will know the input bits of all the parties in S. As after the second phase, those parties already knew the inputs of all parties outside of S, they now know all input bits, and this finishes the proof.

2.1.2 Resilience to Erasures with Improved Rate

Now, we show how to get, for any constant $\eta > 0$, a protocol resilient to an $\Omega\left(\frac{1}{\sqrt{n}}\right)$-fraction of erasures (where the constant depends on η) and rate $n^{-\frac{1}{2}-\eta}$. In words, we can improve the rate of the above protocol by a factor of almost $\sqrt{n}$ while keeping the same resilience up to constants. For this, we note first that the first and the third phase of the above protocol are not the bottleneck when it comes to rate. For example, the first phase involved all the parties broadcasting their bit $\mathcal{O}(n)$ times. Broadcasts are just repetitions of the same bit, and the only reason we need so many repetitions is to have the length of the first phase close to the length of the second phase. This is needed as if the first phase is much smaller than the second phase, the adversary can target it relatively cheaply and fail the protocol. A similar argument applies for the third phase implying that if we can somehow reduce the length of the second phase, we will get a protocol with improved rate.

Let us examine the second phase a bit more closely. As written, this phase needs $\mathcal{O}(n^2)$ rounds as all the n parties broadcast an encoding of all the n bits they received in the first phase. This is done so that at the end of this phase, the set S of parties that need to speak in the last phase has size $\mathcal{O}(\sqrt{n})$ and that all parties know whether or not they are in S. We ask the question: Do we really need *all* the parties to broadcast *all* the bits to get these guarantees?

At first, it may seem like we really do. To see this, consider a situation where only s of the parties are broadcasting only t bits each during the second phase. Let us try to find s, t that allow us to re-derive the size bound, that is, get the number of parties that need to speak in the third phase to be $\mathcal{O}(\sqrt{n})$.

We first claim that s needs to be roughly n for the size bound to hold. The reason is that, no matter how many times each party broadcasts their bit in the first phase, for any party j, the adversary can erase all the received bits for j's transmissions by each of the s parties, by only investing $\mathcal{O}\left(\frac{s}{n^2}\right)$-fraction of erasures. With a budget of $\mathcal{O}\left(\frac{1}{\sqrt{n}}\right)$, this means that there may be $\Omega\left(\frac{n^{1.5}}{s}\right)$ parties that were not heard by any of the parties that are broadcasting, and therefore must speak in the last phase. For $s = o(n)$, this breaks the size bound we desire, and therefore, up to constant factors, we can assume that $s = n$. That is, all the parties are broadcasting in the second phase.

We next claim that t needs to be close to n. The reason is that if all n parties broadcast t bits each, then the bit of a typical party j is only broadcast by t different parties. We claim that in this case, the party j cannot know whether or not they are in the set S. Indeed, consider a situation where the messages of the t parties broadcasting the bit of party j were all erased for party j. This requires an $\mathcal{O}\left(\frac{t}{n^2}\right)$-fraction of erasures, and, with a budget of $\mathcal{O}\left(\frac{1}{\sqrt{n}}\right)$, this means that there may be $\Omega\left(\frac{n^{1.5}}{t}\right)$ many parties j for which this occurs.

All such parties cannot decide if the t parties indeed broadcast their bit and they did not hear the broadcast (in which case party j does not need to be in S and broadcast again), or if the t parties did not broadcast their bit as they did not receive it in the first phase (in which case party j should be in S). As our third phase requires the parties to know whether or not they are in S, this is a problem. Note that it is not a problem when $t = n$ as then the number of such j is $\mathcal{O}(\sqrt{n})$, and we can simply include all the unsure parties in the set S without breaking the size bound.

The key insight. What saves us here is that when party j is unaware that it needs to broadcast in the last phase, then its bit is unknown, after the second phase, to only a small number of parties i. Indeed, suppose that there are u different parties i that are all unaware of the bit of party j. This means that an $\Omega\left(\frac{t}{n}\right)$-fraction of messages in the second phase was erased for u parties implying that the fraction of messages erased is $\Omega\left(\frac{tu}{n^2}\right)$. For this to be within the adversary's budget, we must have $tu = \mathcal{O}(n^{1.5})$. In particular, $u = o(n)$, whenever $t = \omega(\sqrt{n})$.

This leads to the following iterative procedure. We start with the values $s = n$ and $t = n^{\frac{1}{2}+\eta}$, for a small constant $\eta > 0$. This means that $u \leq n^{1-\eta}$ implying that the bit of any party j (that was known to $\Omega(n)$ parties after the first phase) is now known to all but $u \leq \mathcal{O}(n^{1-\eta})$ parties. Since for any set of $10u$ parties and for any $j \in [n]$, almost all parties in the set know the input of party j, we can now decrease s from n to $10u \leq \mathcal{O}(n^{1-\eta})$ without running into the original problem of having a smaller s, namely, that none of the s parties would know the bit of party j. Since s is now smaller, the same number of rounds

now allows the parties to send $t = n^{\frac{1}{2}+2\eta}$ bits each, which would give us that any bit j is now known to all but $\mathcal{O}(n^{1-2\eta})$ parties. Continuing this way for $\mathcal{O}(1/\eta)$ iterations, we get to a stage where $s = n^{\frac{1}{2}+\eta}$ and $t = n$. At this stage, just like earlier, we can conclude that the set S has size $\mathcal{O}(\sqrt{n})$ and the parties all know whether or not they are in S, as desired.

Selecting the t bits. We still have not specified how the parties select which t of the n bits they are going to broadcast when it is their turn to speak. The trivial approach for handing every party a set of size t fixed in advance does not work, as then the adversary knows beforehand which party is in charge of broadcasting which bit, and can erase exactly those bits to that party in the first phase.

This means that the selection of t bits must happen "online" and should additionally satisfy the property that all the bits are broadcast roughly t times each. For this, we use the following idea: The parties in each iteration take turns broadcasting. Each party j keeps track of the number of times they heard bit i in this iteration of the procedure. This count is only kept for bits i that party j knew before the start of this iteration. When it is the turn of party j to broadcast, they broadcast (i, x_i) for the t values of i with the smallest possible counts.

Intuitively, the reason this should work is that the parties send the bits that have so far been broadcast the least, so every bit should be broadcast roughly the same number of times. However, making this work formally requires work, as the adversary may erase some of the bits, so the counts computed by the parties may not correspond to the true count, or some of the bits may be known to very few parties and the others simply cannot broadcast those bits. We work these details out and develop this idea to a full-fledged protocol.

2.1.3 From Erasures to Corruptions

Despite the improved rate, the foregoing ideas only work in the presence of erasure noise. We use this crucially throughout the argument, notably when claiming that the parties can check whether or not they are in S by looking at the fraction of erasures received by them, and when we implicitly assume that any bit received by the parties is correct. Getting a protocol resilient to corruptions requires new ideas, which we describe now.

An improved second phase. The second phase of the protocol described above works under erasure noise, as if party j knows bit i, then this bit must be correct. Therefore, there is no risk of party j spreading mistakes. However, when considering corruption noise, party j has no way of telling which of his bits are correct and only broadcast those.

Thus, we need a way to make the second phase work even in the presence of such incorrect broadcasts. For this, we adopt a two-step approach. First, suppose that party j had the value x_i for bit i at the beginning of some iteration. During the iteration, instead of party j simply keeping track of the number of times bit i was broadcast, party j keeps track of the number of times (i, x_i) was broadcast by the other parties, minus the number of times $(i, \bar{x}_i)$

that was broadcast. When it is the turn of party j to broadcast, they broadcast (i, x_i) for the t values of i with the smallest new counts. At the end of each iteration, each party j updates their value for x_i according to the majority of the broadcasts, for all $i \neq j$.

This is done with the following idea in mind. As the fraction of corruptions is small, for a typical bit i and almost all the parties j, party j has the correct value of bit i. When it is the turn of such a party j to broadcast, they will broadcast the correct value of bit i, for bits i for which the number of times the correct value was broadcast minus the number of times the incorrect value was broadcast is smallest. Such i are at the highest risk of having the majority vote giving the incorrect value.

However, this first step by itself is not sufficient, as the following example shows. Consider an iteration where, say, $t = n^{0.6}$. We expect each bit i to be broadcast roughly t times and want the number of times the correct version is broadcast to exceed the number of times the incorrect version is broadcast by $\Theta(t)$. For this iteration, consider a situation where all the parties with incorrect versions broadcast after all the parties with the correct versions. Then, when the parties with the correct versions are broadcasting, they will never see an incorrect version being broadcast and will broadcast this bit roughly t times. Now, if the number of parties with incorrect versions at the end exceeds $2t$, from the perspective of these parties, this bit has been broadcast "incorrectly" t times, and they will try to "correct" these by broadcasting the incorrect version around $2t$ times. Thus, overall, the incorrect version will be broadcast much more often than the correct version.

At a high level, the message is that, after the parties with incorrect versions have broadcast, there should be enough parties with the correct version that can correct these incorrect broadcasts. One might try to use adaptivity to fix this and try to ensure that the parties who broadcast last always have the correct version. However, ensuring this is hard as the parties do not know whether or not they have the correct version, so they cannot decide when to broadcast. We fix this differently (and maintain the non-adaptivity of the second phase).

We observe that in the example above, after the correct version was broadcast roughly t times, many parties simply chose not to broadcast their version of this bit. This means that, for these parties, their version of this bit must have agreed with the version broadcast so far. Thus, the very fact that only the current version was broadcast for a long time is a signal that it is correct. In our actual protocol, we keep track of exactly this. We have the parties determine not just which version was broadcast more, but for how long the parties avoided broadcasting it in an attempt to "correct" it.

Making this idea work formally requires a careful analysis that complicates our protocol significantly. This is because the adversary can corrupt bits throughout the protocol, making the quantities that the parties are tracking inexact. A big part of our analysis is showing that, for all these quantities, they are still close enough to what they are supposed to be, allowing the protocol to run smoothly.

An improved third phase. Recall that the third phase of our protocol relied on the noise being erasures even more crucially. Indeed, one of the key properties that we maintain in this phase is that all the parties know whether or not they are in the set S. For this, parties kept track of the number of erasures seen so far, something that is impossible now.

As a result, we have to redesign this phase from scratch. For simplicity, in this sketch, we describe the ideas behind this new third phase for the $\mathcal{O}(n^2)$ round protocol described in Section 2.1.1, but for corruptions instead of erasures. We note that all these ideas, with appropriate tweaks, can be used in the iterative protocol with improved rate.

Recall from Section 2.1.1 that the third phase starts with a set S of parties that are the only parties that will broadcast in this phase. This set S of parties consists of two types of parties, both with at most $\mathcal{O}(\sqrt{n})$ parties each: (1) Parties who could not decode the encodings of a constant fraction of parties during the second phase. (2) Parties that did decode enough encodings and realized that the encodings contained an erased version of their bit. This classification was written for erasure noise.

In the case of corruption noise, the parties encode their local versions of all bits in the second phase. Therefore, by the end of the second phase, the local versions of all the parties agree, except for the local versions of the following two types of parties, again of size at most $\mathcal{O}(\sqrt{n})$ each: (1) Parties whose received messages were corrupted a lot in the second phase and therefore, their local versions do not match the local version of the other parties. (2) Parties whose sent messages were corrupted a lot in the first phase and therefore, the local versions the other parties agree on do not have their actual inputs.

Note that the former corresponds to a set of $\mathcal{O}(\sqrt{n})$ parties, we call "pirates", being wrong about possibly all bits, and the latter corresponds to a set of $\mathcal{O}(\sqrt{n})$ bits, called "brats", that possibly all parties are wrong about. Also note that the same party can be both a pirate and a brat. When a party is a pirate (but not a brat), it means that the party received a lot of corruptions in the second phase, although their own bit was received correctly by the other parties. As we only require the parties who did not receive a lot of corruptions to output correctly, these pirates do not actually affect the correctness of our protocol.

The brats do, as these are the parties whose bit was received incorrectly by many other parties. The goal of the third phase is to make sure that every such brat gets to broadcast their own bit in at least an $\Omega\left(\frac{1}{\sqrt{n}}\right)$ fraction of the rounds without collisions from the other parties. As this exceeds the number of rounds the adversary can corrupt, it ensures that at the end of the third phase, the bits of all the brats are received correctly. In fact, our protocol will show a stronger guarantee, and will ensure that both the pirates and the brats get to broadcast in $\Omega\left(\frac{1}{\sqrt{n}}\right)$ fraction of the rounds.

We now describe our procedure: We take a random subset of $\mathcal{O}(\sqrt{n})$ parties and restrict attention to the versions that the parties in the subset have for bits of the other parties in the subset. We next describe two events, each occurring with at least a constant probability. When the first event happens, the protocol allows one of the pirates to broadcast their bits

in many of the rounds, without having the other parties colliding with them. When the second event happens, the protocol allows one of the brats to do so. Note that the protocol does not need to know which event happened, if any.

- *The selected subset has at most one pirate and no other brat.* For this case, note that all the non-pirates agree on all the bits, and the pirate, if it exists, may disagree on these bits. Using this fact, even though the pirate does not know that they are the pirate, we design a protocol that implicitly identifies them as follows. The parties compute an encoding of all the bits in the selected set, but instead of broadcasting this encoding, they use this as a "selector". Specifically, they devote a set of $\mathcal{O}(\log n)$ rounds for each bit of the encoding and speak in those rounds if and only if the bit is 1. Whenever party i speaks, they broadcast an encoding of the pair (i, x_i).

  Note that all parties other than the pirate agree on all the bits, and thus have the same encoding. Thus, they will all speak in the same rounds, creating a massive collision. On the other hand, the pirate has different bits, and thus (for a balanced code), there will be a constant fraction of the locations where its encoding is 1 but the other encodings are 0. In the rounds corresponding to these locations, the pirate will be the only party broadcasting even without knowing that it is the pirate, as desired.

- *The selected subset has at most one brat and no other pirate.* The foregoing idea does not work for this case, as the presence of the brat may be such that half the parties have one version of the brat and the other half has the other version. Thus, regardless of the version, many parties will be encoding that version creating collisions throughout. For this case, we use a different idea. We observe that this case can only occur if many of the parties were "unsure" about the brat, that is, in the second phase, they received the brat as 0 and 1 roughly the same number of times.

  To fix this, whenever the parties are unsure about the value of a bit at the end of the second phase, instead of making them decide on whether their local version for it should be 0 or 1, we simply have them mark it as "unsure". Formally, a bit is marked as unsure if the fraction of times it is received as 0 lies in the interval $\left(\frac{1}{3}, \frac{2}{3}\right)$. While marking bits as unsure does not fix the problem, it simplifies it. Now parties can be confused between the brat being 0 or unsure, and can be confused about the brat being 1 or unsure. However, they are never confused about the brat being 0 or 1. At first, this might seem like making the problem twice as bad, but both these cases are manageable.

  Assume without loss of generality that 1 is the correct value of the brat. In the first case, when the parties are confused between the brat being 0 or unsure, they are confused between the incorrect value of the brat and being unsure about the brat. In either case, their version of the brat is different from the correct version. Thus, the party who has the brat as input would have an encoding that is different from both

these versions. This different encoding would allow the party holding the brat to be the only party broadcasting in a constant fraction of the rounds as before.

In the other case, when the parties are confused between the brat being 1 or unsure, they are confused between the correct value of the brat and being unsure about the brat. When this happens, it means that parties do receive the correct value more than the incorrect value, but not so much so as to make them completely sure. Thus, this case can be fixed at the output stage of the protocol, where any bit a party is unsure about is decoded by majority, and we are done.

Given that both these events happen with at least a constant probability, we get that if we sample many sets, then with extremely large probability, we will have these events happening in constant fraction of the rounds. Thus, the other rounds can be ignored at the cost of just a constant in the error resilience.

Finally, we prove that we actually do not even need randomness, as we can derandomize this and show the existence of set systems that have the above properties. These set systems are what we use in our protocol.

2.2 Overview of the Impossibility Result

We now give a very high level intuition for the proof of our impossibility result. The starting point of the proof is the simple attack on non-adaptive protocols, using only a $\frac{1}{n}$ fraction of erasures: erase all messages from the party that communicates the least. As a result, the other parties gain no information about that party's input bit. This attack is well-defined because the speaking order is predetermined and does not depend on the parties' inputs.

Two-way isolation. In trying to extend the previous attack to adaptive protocols, the goal remains to isolate a party i such that no other party learns anything about party i's input bit. However, a challenge arises in identifying such a party i that can be isolated "cheaply" in the sense that they do not broadcast too much: party i may receive feedback from other parties who realize that its messages were not received in earlier rounds, prompting party i to broadcast more frequently. With adaptivity, targeting any fixed party i in this way can lead to them broadcasting in a constant fraction of the rounds. A simple solution is to erase all of party i's incoming messages as well. In other words, we aim to create *two-way isolation* for party i, which can be achieved with a minor overhead of a $\frac{1}{n}$ fraction of erasures.

However, erasing the messages received by party i has a significant side effect: by noticing an excessive number of erasures in their received transcript, party i can realize they might be under attack and respond by increasing their broadcast frequency. One concern is that all parties may adopt such a "defensive" strategy, broadcasting more to compensate for the erasures. On further reflection, though, having multiple parties behave this way actually benefits the adversary. The reason is that when at least two parties broadcast simultaneously,

there is a collision—essentially making it so that all messages in that round are erased without any additional effort from the adversary.

Why $\frac{1}{\sqrt{n}}$? To understand how many parties the adversary needs to corrupt, fix an arbitrary protocol Π with T rounds. For a parameter $0 < f < 1$, consider the iterative adversary that first selects a party that speaks in at least fT rounds (for some input) when its received transcript is completely erased, then removes these rounds from consideration, and then again selects a party that speaks in fT out of the remaining rounds. Either the adversary can keep doing this and identify $1/f$ parties that cover all rounds[2], or it reaches a stage where all parties speak in at most fT of the remaining rounds. In the former case, the selected parties are enough to make every round a collision implying a failed protocol, and in the latter case, the adversary simply isolates any one of the remaining parties, which fails the protocol. The total number of erasures needed for this adversary is T/f to completely erase the selected parties and fTn to erase all messages from the party that speaks in at most fT rounds (to all other parties). The sum is minimized when $f = \frac{1}{\sqrt{n}}$ and this gives us our noise threshold.

The challenge of signaling. Another challenge we face involves *signaling*. In adaptive protocols, parties may behave very differently depending on their input. For instance, a party might mostly listen (stay silent) when its input bit is 0, but broadcast frequently when its input is 1. In this case, receiving many erasures from that party could suggest its input is 0, as the adversary does not have the budget to erase all the rounds the party broadcasts in when the input is 1. To prevent such "signaling" of the input based on when a party chooses to broadcast, the adversary would need to erase all rounds in which the party might broadcast, regardless of whether their input is 0 or 1.

Preventing signaling. Thus, the only way to fail a protocol is to erase the same party i regardless of whether its input is 0 or 1, and we need to ensure that the number of erasures required in either case is within the adversary's budget. For this, we consider for all parties $i \in [n]$ and inputs $b \in \{0, 1\}$, the set $S_{i,b}$ of the rounds party i speaks in when its input is b and all its received messages are erased. In this notation the goal of the adversary is to find a party i such that $|S_{i,0} \cup S_{i,1}|$ is only a $\Theta(\frac{1}{\sqrt{n}})$ fraction of all the rounds. Indeed, this means that the adversary can erase party i regardless of its input and still stay within its budget.

The adversary does this via the following "greedy" strategy: Either such a party i exists, or for all parties i', we have that at least one of $|S_{i',0}|$ and $|S_{i',1}|$ is larger than a $\Theta(\frac{1}{\sqrt{n}})$ fraction of all the rounds. Let $b_{i'}$ denote the bit for which the set is larger, we pick an arbitrary such i' and fix its input to $b_{i'}$. This means that, if the received transcript for party i' is erased completely, then it will broadcast in at least a $\Theta(\frac{1}{\sqrt{n}})$ fraction of the rounds, and no other party can broadcast in these rounds without causing collisions.

[2]In the actual proof, we cover every round twice so that we can guarantee a collision. This does not affect the idea discussed here except for an extra factor of two.

Thus, we can pretend that this set of rounds has been "removed" from all the sets $S_{i,b}$ that we constructed[3]. We now recurse: If the removal resulted in an i such that $|S_{i,0} \cup S_{i,1}|$ is now $\Theta(\frac{1}{\sqrt{n}})$ fraction of all the rounds, we are done, otherwise, we can find a different i' and continue the argument. As we remove a new $\Theta(\frac{1}{\sqrt{n}})$ fraction of rounds each time we select a new i', the process must stop after at most $\mathcal{O}(\sqrt{n})$ parties i' are selected. We should then be able to find a desired party $i^\star$ for whom $S_{i^\star,0} \cup S_{i^\star,1}$ consists of $\Theta(\frac{1}{\sqrt{n}})$ fraction of all rounds. To fail the protocol, the adversary can therefore erase all the incoming messages to all parties i' selected by this iterative process, and also the outgoing messages from party $i^\star$ to all other parties in rounds $S_{i^\star,0} \cup S_{i^\star,1}$. This is within the adversary's budget, as this only entails erasing $\Theta(\frac{1}{\sqrt{n}})$ fraction of the total received messages, finishing the proof.

3 Model and Preliminaries

3.1 Notation and Preliminaries

For integer $n \geq 1$, $[n]$ is used as a shorthand for $\{1, \ldots, n\}$. For set S, 2^S denotes the power set of S. For tuple $x = (x_1, \ldots, x_n)$ and $i \in [n]$, define $x_{-i} = (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$ and $x_{<i} = (x_1, \ldots, x_{i-1})$. $x_{\leq i}, x_{>i}, x_{\geq i}$ are defined similarly. $\Delta(\cdot, \cdot)$ denotes (relative) Hamming distance. The majority function $\mathsf{maj}(\cdot)$ and the super majority function $\mathsf{smaj}(\cdot)$ are defined as follows:

$$\mathsf{maj}(w) = \begin{cases} 0, & w \in [0, 1/2) \\ 1, & w \in [1/2, 1] \end{cases}, \qquad \mathsf{smaj}(w) = \begin{cases} 0, & w \in [0, 1/3] \\ *, & w \in (1/3, 2/3) \\ 1, & w \in [2/3, 1] \end{cases}.$$

Our protocol will also need the following technical claim.

Lemma 3.1. *For $n \geq 1$ and sufficiently large m such that $\Omega(n \log n) \leq m \leq \mathrm{poly}(n)$, there exists $S^{(t)} \subset [n]$ of size at most $\sqrt{n}$ for $t \in [m]$, such that for all $S' \subset [n]$ of size at most $\sqrt{n}$ and $i \in S'$, it holds that $S^{(t)} \cap S' = \{i\}$ for at least $1/(10\sqrt{n})$ fraction of $t \in [m]$.*

Proof. We prove the lemma using the probabilistic method. Specifically, each $S^{(t)}$ includes each i independently with probability $1/(2\sqrt{n})$. For any fixed t, we have

$$\mathbb{E}\left[\left|S^{(t)}\right|\right] = \frac{1}{2\sqrt{n}} \cdot n = \frac{\sqrt{n}}{2}.$$

Since each i is sampled independently, by Chernoff bound, we can get

$$\Pr\left(\left|S^{(t)}\right| > \sqrt{n}\right) \leq \exp(-\Omega(\sqrt{n})).$$

[3]In the actual proof, a round is erased when at least two parties i' broadcast in this round, guaranteeing a collision. We omit this detail in the sketch.

As $m \leq \text{poly}(n)$, a union bound implies that with high probability, $|S^{(t)}| \leq \sqrt{n}$ for all $t \in [m]$.

For any fixed $S' \subset [n]$ of size at most $\sqrt{n}$ and $i \in S'$, we have

$$\begin{aligned}\mathbb{E}\left[\frac{1}{m} \cdot \left|\{t \in [m] \mid S^{(t)} \cap S' = \{i\}\}\right|\right] &= \frac{1}{2\sqrt{n}} \cdot \left(1 - \frac{1}{2\sqrt{n}}\right)^{|S' \setminus \{i\}|} \\ &\geq \frac{1}{2\sqrt{n}} \cdot \left(1 - \frac{1}{\sqrt{n}}\right)^{\sqrt{n}-1} \\ &\geq \frac{1}{6\sqrt{n}}.\end{aligned}$$

Since each $S^{(t)}$ is sampled independently, by Chernoff bound, we can get

$$\Pr\left(\frac{1}{m} \cdot \left|\{t \in [m] \mid S^{(t)} \cap S' = \{i\}\}\right| < \frac{1}{10\sqrt{n}}\right) \leq \exp(-\Omega(m/\sqrt{n})).$$

The total number of possible choices for S' and i is at most

$$\sqrt{n} \cdot \sum_{s=1}^{\sqrt{n}} \binom{n}{s} \leq n \cdot \binom{n}{\sqrt{n}} \leq \exp(O(\sqrt{n} \log n)).$$

The lemma then follows from a union bound for sufficiently large $m = \Omega(n \log n)$. □

3.2 Error Correcting Codes

We use the following standard results for error correcting codes; see for example [GI05] and references therein.

Lemma 3.2. *For $n \geq 1$ and $\epsilon > 0$, there exists a constant-rate binary error correcting code of input length n that has (relative) Hamming distance $1/2 - \epsilon$.*

Definition 3.3 (Balanced Error Correcting Codes)**.** *For $d \geq 1$ and $\zeta > 0$, an error correcting code C of codeword length ℓ is (d, ζ)-balanced if for all distinct messages $x_1, \ldots, x_d$ and all bits $b_1, \ldots, b_d$, it holds that*

$$\mathop{\mathbb{E}}_{i \sim [\ell]} \mathbb{1}((\mathsf{C}_i(x_1), \ldots, \mathsf{C}_i(x_d)) = (b_1, \ldots, b_d)) \geq \zeta/2^d.$$

Lemma 3.4 ([EKSZ22], Lemma IV.2)**.** *For $n \geq 1$ and $\epsilon > 0$, there exists a constant-rate binary error correcting code of input length n that is $(3, 1 - \epsilon)$-balanced.*

3.3 Communication Model

Inspired by the *collision-as-silence-as-erasures* model of [CHHHZ19] (see also [EKPS23b]), which extends the collision-as-silence model to include stochastic noise, we define its ad-

versarial analogue and prove the impossibility result in Theorem 1.1 for this model. Here, adversarial erasures, like collisions and silent rounds, are received as '$\perp$'.

A protocol in our model is specified by a tuple

$$\Big(n, T, L_{\text{in}}, L_{\text{out}}, \{g_i^{(j)}\}_{i\in[n],j\in[T]}, \{h_i\}_{i\in[n]}\Big),$$

where (1) n is the number of parties; (2) T is the number of rounds; (3) L_{in} is the length of input for each party; (4) L_{out} is the length of output for each party; (5) $g_i^{(j)} : \{0,1\}^{L_{\text{in}}} \times \{0,1,\perp\}^{j-1} \to \{0,1,\perp\}$ is the message function computed by the party i in round j, for $i \in [n]$ and $j \in [T]$; (6) $h_i : \{0,1\}^{L_{\text{in}}} \times \{0,1,\perp\}^{T} \to \{0,1\}^{L_{\text{out}}}$ is the output function computed by party i at the end of the protocol, for $i \in [n]$. We consider adversaries with independent errors across parties, defined by $\mathcal{A} : [T] \to \{0,1,\perp\}^n$.

Execution of protocols starts with each party i having input x_i. Communication proceeds in synchronized rounds. Let $M^{(j)}_{\text{send},i}$ and $M^{(j)}_{\text{recv},i}$ denote the message sent and received, respectively, by party i in round j. In every round j, each party i computes $M^{(j)}_{\text{send},i} = g_i^{(j)}(x_i, M^{(<j)}_{\text{recv},i})$, and either *listens*, if $M^{(j)}_{\text{send},i} = \perp$, or *speaks*, if $M^{(j)}_{\text{send},i} \neq \perp$. After that, each party i receives $M^{(j)}_{\text{recv},i} = \mathcal{A}_i(j)$. At the end of the protocol, each party i outputs $h_i(x_i, M^{(\le T)}_{\text{recv},i})$.

For $j \in [T]$, let $s^{(j)}$ be the unique speaking party in round j and set $s^{(j)} = \perp$ if there is no such party, i.e., if there is no party speaking, namely *silence*, or more than one party speaking, namely *collision*. For $i \in [n]$, the fraction of errors by adversary $\mathcal{A}$ against party i is defined as

$$\delta_i = \frac{1}{T} \cdot \sum_{j=1}^{T} \mathbb{1}\Big[\Big(s^{(j)} \neq \perp \wedge \mathcal{A}_i(j) \neq M^{(j)}_{\text{send},s^{(j)}}\Big) \vee \big(s^{(j)} = \perp \wedge \mathcal{A}_i(j) \neq \perp\big)\Big].$$

Note that when silence or collision happens, the adversary has to pay if it produces a non-erasure symbol. $\mathcal{A}$ has *budget* δ if δ_i averages to at most δ over $i \in [n]$.

For $\delta \in [0,1]$, we say a protocol solves a function $f : \big(\{0,1\}^{L_{\text{in}}}\big)^n \to \{0,1\}^{L_{\text{out}}}$ and is resilient to a δ-fraction of adversarial corruptions, or solves f with *error resilience* δ for short, if on all inputs $x = (x_1, \ldots, x_n)$ and against all adversaries with budget δ, all parties with $\delta_i \le 10\delta$ output $f(x)$ (so other parties may have arbitrary outputs). As mentioned in the introduction, there are at least 90% of such parties. The *erasure resilience* of a protocol is similarly defined against erasure adversaries, where an adversary is said to be an erasure adversary if $\mathcal{A}_i(j) = \perp$ when $s^{(j)} = \perp$, and $\mathcal{A}_i(j) \in \{M^{(j)}_{\text{send},s^{(j)}}, \perp\}$ when $s^{(j)} \neq \perp$. Observe that the erasure resilience of a protocol is at least as large as its error resilience.

Information Dissemination. In this paper, we study the information dissemination problem InfoDis$_{n,m}$ where $L_{\text{in}} = m$, $L_{\text{out}} = nm$, and $f(x_1, \ldots, x_n) = (x_1, \ldots, x_n)$ is the identity function, so every party needs to learn all bits. The rate of an InfoDis$_{n,m}$ protocol is nm/T.

4 The Protocol for InfoDis Against Corruptions

In this section, we present the protocol for solving information dissemination, as formalized in Theorem 4.1. Note that the protocol also works in the easier erasure setting. We first handle the case of $m = 1$ and prove Lemma 4.2. In Section 4.2, we generalize it to the case of $m > 1$.

Theorem 4.1. *For $n, m \geq 1$ and $\eta > 0$, there is a protocol for solving $\textsc{InfoDis}_{n,m}$ that has resilience $\Omega_\eta(1/\sqrt{n})$ against adversarial corruptions and rate $\tilde{\Omega}_\eta(n^{-1/2-\eta})$.*

Lemma 4.2. *For $n \geq 1$ and $\eta > 0$, there is a protocol for solving $\textsc{InfoDis}_{n,1}$ that has resilience $\Omega_\eta(1/\sqrt{n})$ against adversarial corruptions and rate $\tilde{\Omega}_\eta(n^{-1/2-\eta})$.*

Fix sufficiently large $n \geq 1$, and $\eta > 0$. Let $R = 1/(2\eta)$. Assume without loss of generality that R is an integer, as otherwise simply replace η with $1/(2\lceil 1/2\eta \rceil)$. For $r \in [0, R+1]$, let $L_r = n^{1/2+r\eta}$. Note that $L_0 = n^{1/2}$ and $L_R = n$. Also let $L = L_1 = n^{1/2+\eta}$. Let C be a constant-rate binary error correcting code of length $\Theta(\log n)$ that has (relative) Hamming distance 0.49, and D be the corresponding decoder. For $r \in [R]$, let $\mathsf{C}^{(\mathsf{r})}$ be a constant-rate binary error correcting code of length $\Theta(L_r \cdot \log n)$ that has (relative) Hamming distance 0.49, and $\mathsf{D}^{(\mathsf{r})}$ be the corresponding decoder. Note that all decoders $\mathsf{D}, \mathsf{D}^{(\mathsf{r})}$ decode over alphabet $\{0, 1, \perp\}$, and at least 0.49/2 fraction of corruptions (i.e., bit flips or erasures) are needed to cause incorrect or failed decoding (i.e., returning $\perp$). Also let $\mathsf{C}^{\mathsf{bal}}$ be a $(3, 0.99)$-balanced constant-rate binary error correcting code of length $M = \Theta(n^{1/2})$. The existence of $\mathsf{C}, \mathsf{C}^{(\mathsf{r})}$ and $\mathsf{C}^{\mathsf{bal}}$ is guaranteed by Lemmas 3.2 and 3.4 respectively. In addition, let $S^{(r)}$ be the sets promised by Lemma 3.1 for $m = n^{1+\eta}$.

The protocol for proving Lemma 4.2 is given in Algorithm 1. In the following, to avoid confusion, we will write $\mathsf{var}^{(i)}$ to denote the value of variable var for its copy owned by party i. The rest of this section constitutes the proof of Lemma 4.2.

Note that C is used in Line 31, where at most $1 + \log n$ bits are encoded, that for $r \in [R]$, $\mathsf{C}^{(\mathsf{r})}$ is used in Line 14, where at most $L_r \cdot (1 + \log n)$ bits are encoded, and that $\mathsf{C}^{\mathsf{bal}}$ is used in Line 30, where at most $n^{1/2} \cdot \log 3$ bits[4] are encoded by definition of $S^{(r)}$. Thus, Algorithm 1 is well-defined.

Fix $n \geq 1$ and $\eta > 0$. Also fix an input x and an adversary with budget $c/\sqrt{n}$ for sufficiently small c (which might depend on η, but not n). Note that phase 1 has length $nL \cdot \log n = n^{3/2+\eta} \log n$, that for $r \in [R]$, iteration r of phase 2 has length $n^{3/2+\eta}/L_r \cdot \Theta(L_r \cdot \log n) = \Theta(n^{3/2+\eta} \log n)$, and that phase 3 has length $n^{1+\eta} \cdot M \cdot \Theta(\log n) = \Theta(n^{3/2+\eta} \log n)$. Thus, the entire protocol is of overall length $T = \Theta_\eta(n^{3/2+\eta} \log n)$, as desired, due to $R = \Theta_\eta(1)$. It suffices to show its error resilience. Note that the maximum number of corruptions is

$$c/\sqrt{n} \cdot nT = O(cn^{2+\eta} \log n). \tag{1}$$

[4]Technically, $x^{(i)}_{S^{(r)}}$ has a ternary alphabet of $\{0, 1, *\}$. Nevertheless, it can be straightforwardly converted into binary strings before $\mathsf{C}^{\mathsf{bal}}$ encodes it.

Algorithm 1 Party i of the protocol for $\textsc{InfoDis}_{n,1}$.

Phase 1:

for $j \in [n]$ **do**
  if $j = i$ **then**
    Broadcast x_i for $L \cdot \log n$ rounds.
  else
    Initialize $\alpha_j \leftarrow 0$.
    for $\ell \in [L \cdot \log n]$ **do**
      Receive x' from party j, and set $\alpha_j \leftarrow \alpha_j + \mathbb{1}(x' = 1)$.
    Set $x_j \leftarrow \mathsf{maj}(\alpha_j/(L \cdot \log n))$.

Phase 2:

for $r \in [R]$ **do**
  Initialize $\gamma_k, \xi_k \leftarrow 0$ for $k \in [n]$.
  for $j \in [n^{3/2+\eta}/L_r]$ **do**
    if $j = i$ **then**
      Sort $k \in [n]$ in increasing order of $(-1)^{1-x_k}\gamma_k$ (breaking ties arbitrarily), and let $\mathcal{K}$ be the set of the first L_r indices.
      Broadcast $\mathsf{C}^{(\mathsf{r})}(\{(k, x_k) \mid k \in \mathcal{K}\})$, and set $\gamma_k \leftarrow \gamma_k + (-1)^{1-x_k}$ for $k \in \mathcal{K}$.
    else
      Receive y' from party j.
      if $\mathsf{D}^{(\mathsf{r})}(y') = \{(k', x'_{k'}) \mid k' \in \mathcal{K}'\}$ for $K' \subseteq [n]$ of size L_r and $x'_{k'} \in \{0, 1\}$ **then**
        Set $\gamma_{k'} \leftarrow \gamma_{k'} + (-1)^{1-x'_{k'}}$ for $k' \in \mathcal{K}'$.
    for $k \in [n] \setminus \{i\}$ **do**
      if $(-1)^{1-b}\gamma_k \geq L/100$ for some $b \in \{0, 1\}$ **then**
        Set $\xi_k \leftarrow \xi_k + (-1)^{1-b}$.
  for $k \in [n] \setminus \{i\}$ **do**
    if $(-1)^{1-b}\xi_k > n^{3/2+\eta}/(2L_r)$ for some $b \in \{0, 1\}$ **then**
      Set $x_k \leftarrow b$.
for $k \in [n] \setminus \{i\}$ **do**
  Set $\beta_k \leftarrow (1 + \gamma_k/L)/2$ and $x_k \leftarrow \mathsf{smaj}(\beta_k)$.

Phase 3:

Initialize $\sigma_k \leftarrow 0$ for $k \in [n]$.
for $r \in [n^{1+\eta}]$ **do**
  for $\ell \in [M]$ **do**
    if $i \in S^{(r)}$ **and** $\mathsf{C}^{\mathsf{bal}}_\ell(x_{S^{(r)}}) = 1$ **then**
      Broadcast $\mathsf{C}(i, x_i)$.
    else
      Receive y' (from some party or the adversary).
      if $\mathsf{D}(y') = (k', x')$ for $k' \in [n]$, and $x' \in \{0, 1\}$ **then**
        Set $\sigma_{k'} \leftarrow \sigma_{k'} + (-1)^{1-x'}$.
for $k \in [n] \setminus \{i\}$ **do**
  if $(-1)^{1-b}\sigma_k > LM/100$ for some $b \in \{0, 1\}$ **then**
    Set $x_k \leftarrow b$.
  else
    Set $x_k \leftarrow \mathsf{maj}(\beta_k)$.

To start with, intuitively, most input bits should have been correctly received by most parties during phase 1. This is formalized in Lemma 4.3.

Lemma 4.3. *At the end of phase 1, there exists $S_1 \subset [n]$ of size at most $n^{1/2}/4$ such that for all $k \in [n] \setminus S_1$, it holds that $x_k^{(i)} = x_k$ for more than $99/100$ fraction of parties $i \in [n]$.*

Proof. Let S_1 be the set of all $k \in [n]$ such that $x_k^{(i)} \neq x_k$ for at least $1/100$ fraction of parties $i \in [n]$ at the end of phase 1. Observe from Lines 5 and 7 that $\alpha_j^{(i)}$ simply keeps track of the number of times bit "1" is received by party i for the j-th bit. By Line 8, $x_k^{(i)} \neq x_k$ means that at least half of the $L \cdot \log n$ bits sent from party k to party i in Line 3 have been corrupted. Thus, $k \in S_1$ implies at least $n/100 \cdot 1/2 \cdot L \cdot \log n = \Omega(n^{3/2+\eta} \log n)$ corruptions. For sufficiently small c, the lemma follows from Equation (1) as these corruptions are disjoint for different k. □

At a high level, phase 2 repeatedly gathers and exchanges local information. Lemma 4.4 shows that each iteration indeed provides a stronger guarantee by enforcing majority correctness on a smaller subset of parties. This way, fewer parties are required to participate in the next iteration and each of them can then send longer messages.

Lemma 4.4. *During phase 2, for $r \in [R]$ and $k \in [n] \setminus S_1$, it holds that $x_k^{(i)} = x_k$ for more than $99/100$ fraction of parties $i \in [n^{3/2+\eta}/L_{r+1}]$ at the end of iteration r.*

We will prove Lemma 4.4 by induction on r. The base case of $r = 0$ follows from Lemma 4.3 as $n^{3/2+\eta}/L_1 = n$.

Now consider $r \in [R]$. Observe from Lines 10, 14 and 18 that $\gamma_k^{(i)}$ keeps track of the difference in the numbers of times bit "1" is received vs. bit "0" is received, by party i for the k-th bit. Let γ_k^* keep track of the actual difference in the numbers of times bit "1" is broadcast vs. bit "0" is broadcast, for the k-th bit. In other words, it is the number of parties broadcasting $(k, 1)$, minus the number of parties broadcasting $(k, 0)$, in Line 14 so far in the current iteration.

Also observe from Lines 10 and 21 that $\xi_k^{(i)}$ keeps track of the difference in the numbers of times $\gamma_k^{(i)} \geq L/100$ (i.e., received bits heavily dominated by "1") vs $\gamma_k^{(i)} \leq -L/100$ (i.e., received bits heavily dominated by "0"), for $i \neq k$. Similarly, let ξ_k^* keep track of the actual difference in the numbers of times $\gamma_k^* \geq L/90$ vs $\gamma_k^* \leq -L/90$. Note that a slightly different threshold is needed for analysis purposes.

The following helper lemma is the key to the proof of Lemma 4.4. Intuitively, a tiny fraction of incorrect bits should rarely be able to bias the broadcast majority, while a strong majority of correct bits should almost always be able to control the broadcast majority, even by a large margin.

Lemma 4.5. *During phase 2, for $r \in [R]$ and $k \in [n] \setminus S_1$, it holds that $(-1)^{1-x_k}\xi_k^* > 3/4 \cdot n^{3/2+\eta}/L_r$ at the end of iteration r.*

The proof of Lemma 4.5 is deferred to Section 4.1. Assuming Lemma 4.5, we are ready to prove Lemma 4.4.

Proof of Lemma 4.4. Fix any $k \in [n] \setminus S_1$. By Lemma 4.5, we have that $(-1)^{1-x_k}\xi_k^* > 3/4 \cdot n^{3/2+\eta}/L_r$ at the end of iteration r. Let $\mathcal{J}$ be the set of parties $j \in [n^{3/2+\eta}/L_r]$ such that $(-1)^{1-x_k}\gamma_k^* \geq L/90$ immediately after party j broadcasts in iteration r. By definition of ξ_k^*, we have that $|\mathcal{J}| > 3/4 \cdot n^{3/2+\eta}/L_r$.

We claim that for more than 99/100 fraction of parties $i \in [n^{3/2+\eta}/L_{r+1}]$, either $i = k$ or it holds that $(-1)^{1-x_k}\gamma_k^{(i)} \geq L/100$ immediately after party j broadcasts in iteration r, for all $j \in \mathcal{J}$. This is sufficient for proving the lemma because for all such $i \neq k$, by Lines 20 and 21, we can get

$$(-1)^{1-x_k}\xi_k^{(i)} \geq |\mathcal{J}| - (n^{3/2+\eta}/L_r - |\mathcal{J}|) > 1/2 \cdot n^{3/2+\eta}/L_r,$$

and that the choice of b is unambiguous, if possible, in Line 23, as exactly $n^{3/2+\eta}/L_r$ parties broadcast in iteration r.

To see the claim, for any party i not satisfying the claim, there exists $j \in \mathcal{J}$ such that $(-1)^{1-x_k}\gamma_k^{(i)} < L/100$ immediately after party j broadcasts in iteration r. On the other hand, by definition of $\mathcal{J}$, we have that $(-1)^{1-x_k}\gamma_k^* \geq L/90$. Furthermore, by definition of γ_k^*, at least $(L/90 - L/100)/2 = L/1800$ messages received by party i so far in iteration r must have decoded incorrectly or failed to decode, because each such error causes $\gamma_k^{(i)}$ to deviate further from γ_k^* by at most 2. By guarantee of $\mathsf{C}^{(r)}$, this implies at least $L/1800 \cdot 0.49/2 \cdot \Theta(L_r \cdot \log n) = \Theta(L_r \cdot n^{1/2+\eta} \log n)$ corruptions. Since all these corruptions are disjoint for different i, by Equation (1), less than $n^{3/2}/(100L_r) = n^{3/2+\eta}/(100L_{r+1})$ such i exist for sufficiently small c, as claimed. □

After all R iterations of phase 2, all errors are now concentrated on not only a small fraction of input bits (which has already been shown by Lemma 4.3) but also a small fraction of parties, as formalized in Lemma 4.6.

Lemma 4.6. *At the end of phase 2, there exists $S_2 \subset [n]$ of size at most $n^{1/2}/4$ such that for all $i \in [n] \setminus S_2$ and $k \in [n] \setminus S_1$, it holds that $x_k^{(i)} = x_k$.*

Proof. Let S_2 be the set of all $i \in [n]$ such that $x_k^{(i)} \neq x_k$ for some $k \in [n] \setminus S_1$ at the end of phase 2. Fix any $i \in S_2$ and such an index $k \in [n] \setminus S_1$. By Lemma 4.4, more than 99/100 fraction of the first $n^{3/2+\eta}/L_R = L$ parties j hold $x_k^{(j)} = x_k$ at the beginning of iteration R. Also observe that each of the first L parties broadcasts all $L_R = n$ bits in iteration R. As a result, it holds that $(-1)^{1-x_k}\gamma_k^* \geq 99L/100 - (L - 99L/100) = 49L/50$ at the end of iteration R. By Line 26, $x_k^{(i)} \neq x_k$ implies $|\beta_k^{(i)} - x_k| \geq 1/3$ and thus $(-1)^{1-x_k}\gamma_k^{(i)} \leq L/3$. Consequently, at least $(49L/50 - L/3)/2 = 97L/300$ messages received by party i in iteration R must have decoded incorrectly or failed to decode, because each such error causes $\gamma_k^{(i)}$ to deviate further from γ_k^* by at most 2. By guarantee of $\mathsf{C}^{(R)}$, this implies at least $97L/300 \cdot 0.49/2 \cdot \Theta(L_R \cdot \log n) = \Theta(n^{3/2+\eta} \log n)$ corruptions. Since all these corruptions

are disjoint for different i, the number of such $i \in S_2$ is at most $n^{1/2}/4$ by Equation (1) for sufficiently small c, concluding the proof. □

We are now ready to prove Lemma 4.2.

Proof of Lemma 4.2. Fix any party i with no more than $10c/\sqrt{n}$ fraction of corruptions. Then, the number of corruptions for party i is at most

$$10c/\sqrt{n} \cdot T = O(cn^{1+\eta} \log n). \tag{2}$$

Consider the k-th bit. Observe from Lines 27 and 35 that $\sigma_k^{(i)}$ keeps track of the difference in the numbers of times bit "1" is received vs. bit "0" is received, by party i for the k-th bit.

First suppose Line 38 is executed for $b \in \{0, 1\}$. By Line 37, this means the received message decodes to (k, b) by party i at least $LM/100 = \Omega(n^{1+\eta})$ times in Line 34 of phase 3. Each of these messages either is actually sent from party k in Line 31, in which case $b = x_k$, or contains at least $0.49/2 \cdot \Theta(\log n)$ corruptions by guarantee of C. By Equation (2), at least one of these messages is indeed sent from party k, for sufficiently small c. As a result, party i correctly outputs $x_k^{(i)} = b = x_k$ in Line 38.

On the other hand, suppose Line 40 is executed instead. We claim that $(-1)^{1-x_k}\gamma_k^* \geq L/6$ at the end of (iteration R of) phase 2. Assuming the claim for now, we can get $(-1)^{1-x_k}\gamma_k^{(i)} > 0$ at the end of (iteration R of) phase 2. Indeed, suppose for the purpose of contradiction that $(-1)^{1-x_k}\gamma_k^{(i)} \leq 0$. Observe that at least $L/12$ messages received by party i in iteration R of phase 2 must have decoded incorrectly or failed to decode, because each such error causes $\gamma_k^{(i)}$ to deviate further from γ_k^* by at most 2. This implies at least $L/12 \cdot 0.49/2 \cdot \Theta(L_R \cdot \log n) = \Theta(n^{3/2+\eta} \log n)$ corruptions for party i by guarantee of $\mathsf{C}^{(R)}$, contradicting Equation (2). Consequently, it holds that $|\beta_k^{(i)} - x_k| < 1/2$ by Line 26. Party i then correctly outputs $x_k^{(i)} = \mathsf{maj}(\beta_k^{(i)}) = x_k$ in Line 40.

It now remains to show the claim. We also prove it by contradiction. Suppose $(-1)^{1-x_k}\gamma_k^* < L/6$ at the end of (iteration R of) phase 2. Let S_3 be the set of all $j \in [n] \setminus \{k\}$ such that $x_k^{(j)} = x_k$ at the end of phase 2. By Line 26, $j \in S_3$ means $|\beta_k^{(j)} - x_k| \leq 1/3$ and thus $(-1)^{1-x_k}\gamma_k^{(j)} \geq L/3$. By an argument similar to the above, this further implies at least $\Omega(n^{3/2+\eta} \log n)$ corruptions for party j. Since all these corruptions are disjoint for different j, we can get $|S_3| \leq n^{1/2}/4$ due to Equation (1), for sufficiently small c.

Let $S' = \{k\} \cup S_1 \cup S_2 \cup S_3$. By Lemmas 4.3 and 4.6, we now have $|S'| \leq n^{1/2}$. As promised by Lemma 3.1, $S^{(r)} \cap S' = \{k\}$ for at least $1/(10\sqrt{n})$ fraction of $r \in [n^{1+\eta}]$. At the end of phase 2, for any such r and $j \in S^{(r)} \setminus \{k\}$, it holds that $x_k^{(j)} \neq x_k$ due to $j \notin S_3$. In the meantime, we also have $x_{k'}^{(j)} = x_{k'}$ for all $k' \in S^{(r)} \setminus \{k\}$ because $j \notin S_2$ and $k' \notin S_1$. As a result, $x_{S^{(r)}}^{(j)}$ for all $j \in S^{(r)} \setminus \{k\}$ can have at most two possible values (differing only in $x_k^{(j)}$). Additionally, it holds that $x_{S^{(r)}}^{(j)} \neq x_{S^{(r)}}^{(k)}$ due to $x_k^{(k)} = x_k \neq x_k^{(j)}$. Recall that $\mathsf{C}^{\mathsf{bal}}$ is $(3, 0.99)$-balanced, so for at least $0.99/2^3 > 1/9$ fraction[5] of $\ell \in [M]$, it

[5]If there are less than two distinct values of $x_{S^{(r)}}^{(j)}$ for $j \in S^{(r)} \setminus \{k\}$, the fraction can only be larger as

holds that $\mathsf{C}_\ell^{\mathsf{bal}}(x^{(k)}_{S^{(r)}}) = 1$ while $\mathsf{C}_\ell^{\mathsf{bal}}(x^{(j)}_{S^{(r)}}) = 0$ for all $j \in S^{(r)} \setminus \{k\}$. Consequently, party k is the unique broadcasting party in Line 31 of the round indexed by (r, ℓ), due to Line 30. Overall, (k, x_k) is broadcast without collision at least $n^{1+\eta}/(10\sqrt{n}) \cdot M/9 = LM/90$ times. In other words, $(-1)^{1-x_k}\sigma_k^{(i)} \geq LM/90 = \Omega(n^{1+\eta})$ if there were no corruption for party i at all. By Equation (2), it still holds that $(-1)^{1-x_k}\sigma_k^{(i)} > LM/100$ for sufficiently small c, as each corrupted broadcast requires at least $0.49/2 \cdot \Theta(\log n)$ corruptions and deviates $\sigma_k^{(i)}$ by at most 2. However, this implies Line 38 should be executed, a contradiction. □

4.1 Proof of Lemma 4.5

This section constitutes the proof of Lemma 4.5. We will use the inductive hypothesis for the statement of Lemma 4.4. Fix any $k \in [n] \setminus S_1$.

We start with a few more definitions needed for the proof. For $i \in [n^{3/2+\eta}/L_r]$, define $\mathcal{E}_i$ to be the set of $j \in [i-1]$ such that the message sent from party j to i in iteration r decodes incorrectly or fails to decode. For $k' \in [n]$, define $\mathcal{E}'_{k'}$ to be the set of $i \in [n^{3/2+\eta}/L_r]$ such that $x^{(i)}_{k'} \neq x_{k'}$ at the beginning of iteration r. Moreover, for $\tau \in [-n^{3/2+\eta}/L_r - 1, n^{3/2+\eta}/L_r + 1]$, also define $\mathcal{E}'_{k',\tau}$ to be the set of $i \in \mathcal{E}'_{k'}$ such that $(-1)^{1-x_{k'}}\gamma^*_{k'} = \tau$ immediately after party i broadcasts. Note that $\gamma^*_{k'}$ changes by at most ± 1 for each of the broadcasting parties $i \in [n^{3/2+\eta}/L_r]$, so $|\gamma^*_{k'}| \leq n^{3/2+\eta}/L_r$. As a result, $\mathcal{E}'_{k',\tau}$ must be a partition of $\mathcal{E}'_{k'}$.

Now, for $\tau \in [-n^{3/2+\eta}/L_r, n^{3/2+\eta}/L_r]$, define $F(\tau)$ to be the number of parties $i \in [n^{3/2+\eta}/L_r]$ satisfying all of the following immediately after i broadcasts in iteration r:

1. $k \in \mathcal{K}^{(i)}$ or $x^{(i)}_k \neq x_k$;
2. $(-1)^{1-x_k}\gamma^*_k = \tau$.

For $k' \in [n]\setminus\{k\}$ and $\tau' \in [-n^{3/2+\eta}/L_r, n^{3/2+\eta}/L_r]$, $\phi \in [0, n^{3/2+\eta}/L_r]$, also define $G_{k'}(\tau, \tau', \phi)$ to be the number of parties $i \in [n^{3/2+\eta}/L_r]$ satisfying all of the following immediately after i broadcasts in iteration r:

1. $k \notin \mathcal{K}^{(i)}$ and $x^{(i)}_k = x_k$;
2. $(-1)^{1-x_k}\gamma^*_k = \tau$;
3. $k' \in \mathcal{K}^{(i)}$;
4. $(-1)^{1-x_{k'}}\gamma^*_{k'} = \tau'$;
5. $|\mathcal{E}_i| = \phi$.

We write $F(\leq \tau)$ as a shorthand for $\sum_{\hat{\tau} \leq \tau} F(\hat{\tau})$, and $F(\cdot)$ for $\sum_\tau F(\tau)$. Similarly for G.

Essentially, $\mathcal{E}, \mathcal{E}'$ capture two major possibilities of adversely biasing γ^* over the time, namely adversarial corruptions and incorrect bits. In fact, the following technical claims show that these are actually the only possibilities.

balance for three messages implies balance for fewer messages.

Claim 4.7. *For $\tau \in [-n^{3/2+\eta}/L_r, n^{3/2+\eta}/L_r]$, it holds that*

$$F(\tau) \le \mathbb{1}(\tau > 0) + |\mathcal{E}'_{k,\tau}| + |\mathcal{E}'_{k,\tau-1}|.$$

Proof. Observe that by definition, γ_k^* changes (by ± 1) if and only if party i broadcasts k in iteration r. By definition of F, Item 1 means that either $x_k^{(i)} \neq x_k$ or party i broadcasts k. If $x_k^{(i)} \neq x_k$, we have that $i \in \mathcal{E}'_k$. Because of Item 2, this occurs at most $|\mathcal{E}'_{k,\tau}|$ times. Otherwise, $x_k^{(i)} = x_k$ and party i broadcasts k, increasing $(-1)^{1-x_k}\gamma_k^*$ by 1. Moreover, note that γ_k^* starts at 0 at the beginning of iteration r. Then, for $\tau > 0$, starting from the second time such case occurs, each occurrence i must be associated with a unique $j \in \mathcal{E}'_{k,\tau-1} \cap [i-1]$, also due to Item 2. In the meantime, for $\tau \le 0$, every occurrence i, including the first one, of the case $(-1)^{1-x_k}\gamma_k^*$ increasing from $\tau - 1$ to τ is uniquely associated with some $j \in \mathcal{E}'_{k,\tau-1} \cap [i-1]$. Thus, such case occurs at most $\mathbb{1}(\tau > 0) + |\mathcal{E}'_{k,\tau-1}|$ times. Combining the two cases concludes the proof. □

Claim 4.8. *For $k' \in [n] \setminus \{k\}$ and $\tau, \tau' \in [-n^{3/2+\eta}/L_r, n^{3/2+\eta}/L_r]$, $\phi \in [0, n^{3/2+\eta}/L_r]$, it holds that*

$$G_{k'}(\le \tau, \tau', \le \phi) \le \mathbb{1}(0 < \tau' \le 1 + \tau + 4\phi) + |\mathcal{E}'_{k',\tau'}| + |\mathcal{E}'_{k',\tau'-1}|.$$

Proof. Consider the status right before party i broadcasts in iteration r. Observe that each corrupted message sent from some party $j \in [i-1]$ to i can cause $\gamma_k^{(i)}$ to deviate further from γ_k^* by at most 2, so we can get

$$|\gamma_k^{(i)} - \gamma_k^*| \le 2|\mathcal{E}_i| \le 2\phi, \tag{3}$$

and similarly

$$|\gamma_{k'}^{(i)} - \gamma_{k'}^*| \le 2|\mathcal{E}_i| \le 2\phi. \tag{4}$$

By definition of G, we also have

$$(-1)^{1-x_k}\gamma_k^* \le \tau, \tag{5}$$

and

$$(-1)^{1-x_{k'}}\gamma_{k'}^* \ge \tau' - 1, \tag{6}$$

right before party i broadcasts in iteration r, since k is not broadcast by i.

Now, if $x_{k'}^{(i)} \neq x_{k'}$ right before party i broadcasts in iteration r, then $i \in \mathcal{E}'_{k',\tau'}$. Otherwise, $x_{k'}^{(i)} = x_{k'}$ and $(-1)^{1-x_{k'}}\gamma_{k'}^*$ is increased by 1 due to Item 3. Also, Equations (3) and (5), together with Item 1, imply that

$$(-1)^{1-x_k^{(i)}}\gamma_k^{(i)} = (-1)^{1-x_k}\gamma_k^{(i)} \le (-1)^{1-x_k}\gamma_k^* + 2\phi \le \tau + 2\phi, \tag{7}$$

while Equations (4) and (6) together imply that

$$(-1)^{1-x_{k'}^{(i)}}\gamma_{k'}^{(i)} = (-1)^{1-x_{k'}}\gamma_{k'}^{(i)} \geq (-1)^{1-x_{k'}}\gamma_{k'}^{*} - 2\phi \geq \tau' - 1 - 2\phi. \tag{8}$$

In the meantime, by definition of $\mathcal{K}^{(i)}$, Items 1 and 3 also mean that

$$(-1)^{1-x_{k'}^{(i)}}\gamma_{k'}^{(i)} \leq (-1)^{1-x_{k}^{(i)}}\gamma_{k}^{(i)}.$$

This is compatible with Equations (7) and (8) only if $\tau' - 1 - 2\phi \leq \tau + 2\phi$, or equivalently $\tau' \leq 1 + \tau + 4\phi$. Assuming this, an argument similar to the one for Claim 4.7 applies. Concretely, we know that every occurrence i of such case $(-1)^{1-x_{k'}}\gamma_{k'}^{*}$ increasing from $\tau' - 1$ to τ' is uniquely associated with some $j \in \mathcal{E}'_{k',\tau'-1} \cap [i-1]$, except for the first occurrence when $\tau' > 0$.

The claim follows by combining all cases above. □

Claim 4.9. *For $\phi \in (0, n^{3/2+\eta}/L_r]$, it holds that*

$$\sum_{k' \in [n]\setminus\{k\}} G_{k'}(\cdot,\cdot,\geq\phi) \leq L_r \cdot \sum_{i \in [n^{3/2+\eta}/L_r]} \mathbb{1}(|\mathcal{E}_i| \geq \phi) \leq \frac{L_r}{\phi} \cdot \sum_{i \in [n^{3/2+\eta}/L_r]} |\mathcal{E}_i|.$$

Proof. The claim follows directly from definition of G. Indeed, at most L_r indices $k' \in [n] \setminus \{k\}$ can satisfy Item 3 for each party $i \in [n^{3/2+\eta}/L_r]$ satisfying $|\mathcal{E}_i| \geq \phi$. □

The following Corollary 4.10 is a direct consequence of Claims 4.7 to 4.9. It basically reduces the proof of Lemma 4.5 to upper bounding the sizes of $\mathcal{E}, \mathcal{E}'$, which is taken care of by Claims 4.11 and 4.12 respectively.

Corollary 4.10. *During phase 2, for $r \in [R]$, $k \in [n] \setminus S_1$, and $\tau \in [n^{3/2+\eta}/L_r]$, the number of parties $i \in [n^{3/2+\eta}/L_r]$ such that $(-1)^{1-x_k}\gamma_k^{*} \leq \tau$ immediately after i broadcasts in iteration r, is at most*

$$\begin{aligned} &F(\leq\tau) + \frac{1}{L_r} \cdot \sum_{k' \in [n]\setminus\{k\}} G_{k'}(\leq\tau,\cdot,\cdot) \\ &\quad\leq \frac{4n\tau}{L_r} + 2|\mathcal{E}'_k| + \frac{4}{\tau} \cdot \sum_{i \in [n^{3/2+\eta}/L_r]} |\mathcal{E}_i| + \frac{2}{L_r} \cdot \sum_{k' \in [n]\setminus\{k\}} |\mathcal{E}'_{k'}|. \end{aligned}$$

Proof. Observe that F counts the number of occurrences i with $k \in \mathcal{K}^{(i)}$ or $x_k^{(i)} \neq x_k$. For any other occurrence of i, $\mathcal{K}^{(i)}$ contains exactly L_r indices $k' \in [n] \setminus \{k\}$ by definition. Therefore, the total number of occurrences is at most

$$\sum_{\hat{\tau}\leq\tau} F(\hat{\tau}) + \frac{1}{L_r} \cdot \sum_{k'} \sum_{\hat{\tau}\leq\tau} \sum_{\tau'} \sum_{\phi} G_{k'}(\hat{\tau},\tau',\phi)$$

$$
\begin{aligned}
&\leq \tau + 2 \cdot \sum_{\hat{\tau} \leq \tau} |\mathcal{E}'_{k,\hat{\tau}}| + \frac{1}{L_r} \cdot \sum_{k'} \sum_{\hat{\tau} \leq \tau} \sum_{\tau'} \sum_{\phi} G_{k'}(\hat{\tau}, \tau', \phi) && \text{(by Claim 4.7)} \\
&\leq \tau + 2|\mathcal{E}'_k| + \frac{1}{L_r} \cdot \sum_{k'} G_{k'}(\cdot, \cdot, > \tau/4) + \frac{1}{L_r} \cdot \sum_{k'} \sum_{\tau'} G_{k'}(\leq \tau, \tau', \leq \tau/4) \\
&\leq \tau + 2|\mathcal{E}'_k| + \frac{4}{\tau} \cdot \sum_{i} |\mathcal{E}_i| + \frac{1}{L_r} \cdot \sum_{k'} \sum_{\tau'} G_{k'}(\leq \tau, \tau', \leq \tau/4) && \text{(by Claim 4.9)} \\
&\leq \tau + 2|\mathcal{E}'_k| + \frac{4}{\tau} \cdot \sum_{i} |\mathcal{E}_i| + \frac{1}{L_r} \cdot \sum_{k'} \left(1 + 2\tau + 2 \cdot \sum_{\tau'} |\mathcal{E}'_{k',\tau'}|\right) && \text{(by Claim 4.8)} \\
&\leq \frac{4n\tau}{L_r} + 2|\mathcal{E}'_k| + \frac{4}{\tau} \cdot \sum_{i} |\mathcal{E}_i| + \frac{2}{L_r} \cdot \sum_{k'} |\mathcal{E}'_{k'}|,
\end{aligned}
$$

as claimed (where ranges of summations are omitted when they are clear from context). □

Claim 4.11. *It holds that*

$$\sum_{i \in [n^{3/2+\eta}/L_r]} |\mathcal{E}_i| = O(n^{2+\eta}/L_r).$$

Proof. Each $j \in \mathcal{E}_i$ implies at least $0.49/2 \cdot \Theta(L_r \cdot \log n)$ corruptions in the message sent from party j to i, due to guarantee of $\mathsf{C}^{(r)}$. The claim then follows from Equation (1) because all these corruptions are disjoint. □

Claim 4.12. *It holds that*

$$\sum_{k' \in [n] \setminus \{k\}} |\mathcal{E}'_{k'}| = O(n^{3/2}).$$

Proof. Consider any $k' \in [n] \setminus S_1$. For $r = 1$, the proof of Lemma 4.3 actually shows that each occurrence of $x^{(i)}_{k'} \neq x_{k'}$ at the end of phase 1, or equivalently at the beginning of (iteration 1 of) phase 2, implies at least $L/2 \cdot \log n$ corruptions in the message sent from party k' to i. The claim then follows from Equation (1) in this case.

For $r > 1$, by the inductive hypothesis for the statement of Lemma 4.4, it holds that $x^{(i)}_{k'} = x_{k'}$ for more than $99/100$ fraction of parties $i \in [n^{3/2+\eta}/L_{r-1}]$ at the beginning of iteration $r-1$. Furthermore, the proof of Lemma 4.4 actually shows that any party $i \in [n^{3/2+\eta}/L_r]$ can hold $x^{(i)}_{k'} \neq x_{k'}$ at the end of iteration $r-1$, or equivalently $i \in \mathcal{E}'_{k'}$, only if at least $L/1800$ messages involving $x_{k'}$ received by party i in iteration $r-1$ have decoded incorrectly or failed to decode. Each such error implies at least $\Omega(L_{r-1} \cdot \log n)$ corruptions. These corruptions are not necessarily disjoint across k'. Nevertheless, since each party broadcasts L_{r-1} indices, at most $2L_{r-1}$ indices could be involved by any corrupted message. As a result, we can always associate $\Omega(\log n)$ corruptions to each such error that are unique for different k'. In total, this means at least

$$L/1800 \cdot \Omega(\log n) \cdot \sum_{k' \in [n] \setminus S_1} |\mathcal{E}'_{k'}|$$

corruptions. Combined with Equation (1), it implies that

$$\sum_{k' \in [n] \setminus S_1} |\mathcal{E}'_{k'}| = O(n^{3/2}).$$

The proof concludes by observing that

$$\sum_{k' \in S_1} |\mathcal{E}'_{k'}| \leq |S_1| \cdot n = O(n^{3/2}),$$

due to Lemma 4.3. □

Finally, we are ready to prove Lemma 4.5.

Proof of Lemma 4.5. By the inductive hypothesis for the statement of Lemma 4.4, it holds that $x_k^{(i)} = x_k$ for more than 99/100 fraction of parties $i \in [n^{3/2+\eta}/L_r]$ at the end of iteration $r-1$. In other words, we have $|\mathcal{E}'_k| \leq n^{3/2+\eta}/(100L_r)$. Corollary 4.10, together with Claims 4.11 and 4.12, implies that the fraction of parties $i \in [n^{3/2+\eta}/L_r]$ such that $(-1)^{1-x_k}\gamma_k^* \leq \tau$ immediately after i broadcasts in iteration r, is at most

$$\begin{aligned}
&\frac{L_r}{n^{3/2+\eta}} \cdot \left(\frac{4n\tau}{L_r} + 2|\mathcal{E}'_k| + \frac{4}{\tau} \cdot \sum_{i \in [n^{3/2+\eta}/L_r]} |\mathcal{E}_i| + \frac{2}{L_r} \cdot \sum_{k' \in [n] \setminus \{k\}} |\mathcal{E}'_{k'}| \right) \\
&\quad \leq \frac{L_r}{n^{3/2+\eta}} \cdot \left(\frac{4n\tau}{L_r} + \frac{n^{3/2+\eta}}{50L_r} + O\left(\frac{n^{2+\eta}}{\tau L_r}\right) + O\left(\frac{n^{3/2}}{L_r}\right) \right) \\
&\quad \leq \frac{4\tau}{L} + \frac{1}{50} + O\left(\frac{n^{1/2}}{\tau} + \frac{1}{n^\eta}\right) \\
&\quad < \frac{1}{10},
\end{aligned}$$

where $\tau = L/90$ is plugged in. Consequently, by definition, we can get

$$(-1)^{1-x_k}\xi_k^* > ((1 - 1/10) - 1/10) \cdot n^{3/2+\eta}/L_r > 3/4 \cdot n^{3/2+\eta}/L_r,$$

as claimed. □

4.2 Generalization to Multi-Bit Inputs

Lemma 4.2 has focused on single-bit inputs. One natural question to ask is whether we can deal with multi-bit inputs, i.e., $\textsc{InfoDis}_{n,m}$ for $m > 1$, with similar resilience and rate. The trivial attempt is to repeat the protocol for single-bit inputs m times. However, the adversary may choose to attack only one execution of the protocol, resulting in an m factor loss in resilience.

Algorithm 2 Party i of the protocol for $\textsc{InfoDis}_{n,m}$.

Initialize $y_i \leftarrow \mathsf{C}(x_i)$.
for $\ell = 1, \ldots, L$ **do**
    Simulate the T-round protocol for $\textsc{InfoDis}_{n,1}$ with input $y_{i,\ell}$, and obtain output $(y^{(i)}_{1,\ell}, \ldots, y^{(i)}_{n,\ell})$.
Output $(\mathsf{D}(y^{(i)}_1), \ldots, \mathsf{D}(y^{(i)}_n))$.

Nevertheless, we answer the question in the affirmative using the following reduction between protocols. In a nutshell, each party encodes its input using an error correcting code to ensure the aforementioned attack will not work.

Lemma 4.13. *For $n, m, T \geq 1$ and $\delta \in (0, 1)$, if there is a T-round protocol that solves $\textsc{InfoDis}_{n,1}$ with error resilience δ, then there is an $O(mT)$-round protocol that solves $\textsc{InfoDis}_{n,m}$ with error resilience $\Omega(\delta)$.*

Proof. Let C be a constant-rate binary error correcting code of length $L = O(m)$ that has (relative) Hamming distance 0.49, and D be the corresponding decoder. The existence of C is guaranteed by Lemma 3.2.

We prove the lemma by showing that the protocol as defined in Algorithm 2 solves $\textsc{InfoDis}_{n,m}$ with error resilience 0.1δ. Note that the protocol has $LT = O(mT)$ rounds as claimed. Fix any adversary with overall corruption budget 0.1δ. It suffices to show that every party i correctly recovers $(x_1, \ldots, x_n)$ so long as there is no more than δ fraction of corruptions against party i.

To see this, suppose the fraction of corruptions against party i is at most δ. By averaging arguments, for at least $1 - 0.1 - 0.1 = 0.8$ fraction of iterations $\ell \in [L]$, the overall fraction of corruptions across all parties in the simulated protocol is less than δ, and the fraction of corruptions against party i in the simulated protocol is less than 10δ. By assumption of the lemma, the correctness of the simulated protocol means party i successfully recovers $(y^{(i)}_{1,\ell}, \ldots, y^{(i)}_{n,\ell}) = (y_{1,\ell}, \ldots, y_{n,\ell})$ in all such iterations. It further implies that $\Delta(y^{(i)}_k, y_k) \leq 1 - 0.8 = 0.2 < 0.49/2$ and thus $D(y^{(i)}_k) = x_k$ for all $k \in [n]$. This concludes the proof. □

Finally, observe that Theorem 4.1 is a direct consequence of Lemmas 4.2 and 4.13.

5 Hardness of InfoDis Against Erasures

We prove Theorem 5.1 in this section. It naturally extends to the corruption setting, which is harder for designing protocols.

Theorem 5.1. *For $n, m \geq 1$, any protocol for solving $\textsc{InfoDis}_{n,m}$ has resilience at most $O(1/\sqrt{n})$ against adversarial erasures, regardless of the rate.*

Without loss of generality, assume $m = 1$. Hardness for $m > 1$ follows directly from the $m = 1$ case. Fix any T-round protocol, where $T \geq 1$. As the first step, we define an all-erasure adversary by $\mathcal{A}_\perp(j) = \perp^n$ for all $j \in [T]$. For $i \in [n]$ and $b \in \{0,1\}$, let $S_{i,b} \subseteq [T]$ be the set of rounds j in which party i speaks against $\mathcal{A}_\perp$ when $x_i = b$. Note that it is independent of x_{-i} and well-defined since party i receives only $\perp$. In the rest of this section, we find the actual adversary $\mathcal{A}$ that fails the protocol. It consists of the following two types of erasures:

1. **Party erasure:** $\mathcal{A}$ completely erases all received messages of some subset $P \subseteq [n]$ of parties. Roughly, these are the parties speaking a lot so that they cause collisions in most rounds. In other words, messages sent in all those rounds are erased for free.

2. **Round erasure:** $\mathcal{A}$ also completely erases all received messages in some subset $R \subseteq [T]$ of rounds. At the high level, the goal is to ensure that some party i^*, which speaks in very few rounds, can convey absolutely zero information to all remaining parties. So no other party can learn anything about the input bit of this party.

Overall, $\mathcal{A}$ makes at most a $|P|/n + |R|/T$ fraction of erasures.

Concretely, we first perform party erasure by setting $P = Q \cup \{i^*\}$, where i^* is an arbitrary party not in Q and Q is given by the following procedure with integer parameter $k < n/2$ to be determined:

1. Initially, let $Q = \emptyset$ and Z be the multi-set containing each round $j \in [T]$ twice.

2. While there exists $i \notin Q$ and $b \in \{0,1\}$ such that $|S_{i,b} \cap Z| > T/k$, set $x^*_i = b$ and update $Q \leftarrow Q \cup \{i\}$ and $Z \leftarrow Z \setminus S_{i,b}$.

3. Finally, let $\mathsf{supp}(Z)$ be the support of Z.

Assuming no round erasures for now, Lemma 5.2 summarizes the useful properties of party erasure.

Lemma 5.2. *It holds that:*

1. $|P| \leq 2k$.

2. *For all* $b \in \{0,1\}$, $|S_{i^*,b} \cap \mathsf{supp}(Z)| \leq T/k$.

3. *For all rounds* $j \notin \mathsf{supp}(Z)$, *at least two parties in* Q *speak against (partial) adversary* $\mathcal{A}$ *on (partial) input* x^*_Q.

Proof. Observe that the inclusion of each i into Q is accompanied by a decrease of more than T/k in the size of Z. The first statement then follows as Z initially has size $2T$ and thus $|Q| < 2T/(T/k) = 2k$. The second statement is true because i^* would be included in Q otherwise. To see the third statement, note that all $S_{i,b}$ have multiplicity 1. So $j \notin \mathsf{supp}(Z)$ implies existence of $i_1, i_2 \in Q$ such that $j \in S_{i_1,x^*_{i_1}}$ and $j \in S_{i_2,x^*_{i_2}}$, meaning that parties i_1, i_2

both speak against adversary $\mathcal{A}_\perp$ on input x^*_Q. Since $\mathcal{A}$ erases all received messages of all parties in Q, all parties in Q have exactly the same transcript against both adversaries $\mathcal{A}_\perp$ and $\mathcal{A}$ on input x^*_Q. The third statement follows as the protocol is deterministic. □

Now that all rounds not in $\mathsf{supp}(Z)$ are erased for free due to collisions, we next perform round erasure by setting $R = \mathsf{supp}(Z) \cap (S_{i^*,0} \cup S_{i^*,1})$. This completes the definition of $\mathcal{A}$. The second statement of Lemma 5.2 ensures $|R| \leq 2T/k$. The following Lemma 5.3 formalizes the idea of isolating party i^*.

Lemma 5.3. *Let $x^*_{-i^*}$ be an arbitrary extension of x^*_Q (i.e., assign an arbitrary bit to each coordinate not in P). All parties have received exactly the same transcript against adversary $\mathcal{A}$ on both inputs $(x^*_{i^*} = 0, x^*_{-i^*})$ and $(x^*_{i^*} = 1, x^*_{-i^*})$.*

Proof. The proof is by induction on number of rounds. The base case of 0 rounds holds vacuously. Now suppose all parties have exactly the same transcript up to round $j-1$ against adversary $\mathcal{A}$ on both inputs. Consider round j. If $j \notin \mathsf{supp}(Z)$, by the third statement of Lemma 5.2, all parties receive $\perp$ in round j because there is a collision regardless of what party i^* does. For $j \in \mathsf{supp}(Z)$, if $j \in R$, all parties also receive $\perp$ in round j due to round erasure. Otherwise, $j \notin R$ and thus $j \notin S_{i^*,0} \cup S_{i^*,1}$. Party i^* listens in round j in both cases. By the inductive hypothesis, all parties will also have exactly the same transcript up to round j against adversary $\mathcal{A}$ on both inputs, concluding the proof. □

Finally, we are ready to prove Theorem 5.1.

Proof of Theorem 5.1. Since $|P| \leq 2k < n$ by the first statement of Lemma 5.2, there exists a party i' not in P. Lemma 5.3 implies, in particular, that party i' cannot distinguish the two cases where party i^* has different input bits. As a result, party i' must output incorrectly in at least one of the two cases. To prove the theorem, it suffices to bound the budget of $\mathcal{A}$ as follows:

$$|P|/n + |R|/T \leq 2k/n + (2T/k)/T = O(1/\sqrt{n}),$$

if we set $k = \lceil \sqrt{n} \rceil$. Note that the fraction of erasures by adversary $\mathcal{A}$ against party i' is also with the same budget ($(2T/k)/T = 2/k$ to be precise as party i' is only subject to round erasure). Overall, regardless of T, an $O(1/\sqrt{n})$ fraction of adversarial erasures is sufficient to fail the protocol. □

AI Methodology

The technical and conceptual contributions presented in this paper are due to the authors. ChatGPT-6.1 Sol was used only after the manuscript had been written, to check the proofs for errors and to copy edit the introduction.

While the primary goal of this paper was to show that nontrivial resilience is possible even in multi-party settings, we also hoped to obtain a protocol with constant rate. After completing the manuscript, we supplied it to ChatGPT-6 Astra and asked it to improve the

rate of our protocol. It proposed a protocol achieving the optimal resilience $\Omega(\frac{1}{\sqrt{n}})$ with an inverse-logarithmic rate. The construction can also be formulated to achieve resilience $\tilde{\Omega}(\frac{1}{\sqrt{n}})$ with constant rate.

We chose not to include these constructions because they were substantially more complex and primarily extended our ideas through an extensive collection of "brute force" techniques. In our judgment, the resulting improvements offered limited additional conceptual insight into the problem.

References

[ABE+16] Noga Alon, Mark Braverman, Klim Efremenko, Ran Gelles, and Bernhard Haeupler. Reliable communication over highly connected noisy networks. In *Symposium on Principles of Distributed Computing (PODC)*, pages 165–173, 2016. 6

[ADHS20] Abhinav Aggarwal, Varsha Dani, Thomas P. Hayes, and Jared Saia. A scalable algorithm for multiparty interactive communication with private channels. In *International Conference on Distributed Computing and Networking (ICDCN)*, pages 8:1–8:15, 2020. 1, 6, 7

[AGL22] Yagel Ashkenazi, Ran Gelles, and Amir Leshem. Noisy beeping networks. *Information and Computation*, 289:104925, 2022. 6

[AGS16] Shweta Agrawal, Ran Gelles, and Amit Sahai. Adaptive protocols for interactive communication. In *International Symposium on Information Theory (ISIT)*, pages 595–599, 2016. 6

[BEGH16] Mark Braverman, Klim Efremenko, Ran Gelles, and Bernhard Haeupler. Constant-rate coding for multiparty interactive communication is impossible. In *Symposium on Theory of Computing (STOC)*, pages 999–1010, 2016. 6

[CHGH18] Keren Censor-Hillel, Ran Gelles, and Bernhard Haeupler. Making asynchronous distributed computations robust to channel noise. In *Innovations in Theoretical Computer Science Conference (ITCS)*, volume 94, pages 50:1–50:20, 2018. 1

[CHGH19] Keren Censor-Hillel, Ran Gelles, and Bernhard Haeupler. Making asynchronous distributed computations robust to noise. *Distributed Computing*, 32:405–421, 2019. 6, 7

[CHHHZ17] Keren Censor-Hillel, Bernhard Haeupler, D Ellis Hershkowitz, and Goran Zuzic. Broadcasting in noisy radio networks. In *Symposium on Principles of Distributed Computing (PODC)*, pages 33–42, 2017. 6

[CHHHZ19] Keren Censor-Hillel, Bernhard Haeupler, D Ellis Hershkowitz, and Goran Zuzic. Erasure correction for noisy radio networks. In *International Symposium on Distributed Computing (DISC)*, 2019. 6, 18

[CK85] Imrich Chlamtac and Shay Kutten. On broadcasting in radio networks-problem analysis and protocol design. *IEEE Trans. Communications*, 33(12):1240–1246, 1985. 1, 2

[EKP+24] Klim Efremenko, Gillat Kol, Dmitry Paramonov, Ran Raz, and Raghuvansh R. Saxena. Information dissemination via broadcasts in the presence of adversarial noise. In Rahul Santhanam, editor, *Conference on Computational Complexity (CCC)*, volume 300 of *LIPIcs*, pages 19:1–19:33, 2024. 5, 6

[EKPS21a] Klim Efremenko, Gillat Kol, Dmitry Paramonov, and Raghuvansh R. Saxena. Computation over the noisy broadcast channel with malicious parties. In *Innovations in Theoretical Computer Science Conference (ITCS)*, volume 185, pages 82:1–82:19, 2021. 2, 6

[EKPS21b] Klim Efremenko, Gillat Kol, Dmitry Paramonov, and Raghuvansh R. Saxena. Tight bounds for general computation in noisy broadcast networks. In *Symposium on Foundations of Computer Science (FOCS)*, pages 634–645, 2021. 6

[EKPS23a] Klim Efremenko, Gillat Kol, Dmitry Paramonov, and Raghuvansh R. Saxena. Noisy radio network lower bounds via noiseless beeping lower bounds. In *Innovations in Theoretical Computer Science Conference (ITCS)*, volume 251 of *LIPIcs*, pages 46:1–46:20, 2023. 6

[EKPS23b] Klim Efremenko, Gillat Kol, Dmitry Paramonov, and Raghuvansh R. Saxena. Protecting single-hop radio networks from message drops. In Kousha Etessami, Uriel Feige, and Gabriele Puppis, editors, *International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 261 of *LIPIcs*, pages 53:1–53:20, 2023. 6, 18

[EKPS25] Klim Efremenko, Gillat Kol, Dmitry Paramonov, and Raghuvansh R. Saxena. Constant rate codes for adaptive broadcasts do not exist. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 565–584, 2025. 6

[EKS18] Klim Efremenko, Gillat Kol, and Raghuvansh Saxena. Interactive coding over the noisy broadcast channel. In *Symposium on Theory of Computing (STOC)*, pages 507–520. ACM, 2018. 6

[EKS19] Klim Efremenko, Gillat Kol, and Raghuvansh Saxena. Radio network coding requires logarithmic overhead. In *Foundations of Computer Science (FOCS)*, pages 348–369, 2019. 6

[EKS20a] Klim Efremenko, Gillat Kol, and Raghuvansh Saxena. Interactive error resilience beyond 2/7. In *Symposium on Theory of Computing (STOC)*. ACM, 2020. 6

[EKS20b] Klim Efremenko, Gillat Kol, and Raghuvansh R. Saxena. Noisy beeps. In Yuval Emek and Christian Cachin, editors, *Symposium on Principles of Distributed Computing (PODC)*, pages 418–427, 2020. 6

[EKS21] Klim Efremenko, Gillat Kol, and Raghuvansh R. Saxena. Optimal error resilience of adaptive message exchange. In *Symposium on Theory of Computing (STOC)*, pages 1235–1247. ACM, 2021. 6

[EKSZ22] Klim Efremenko, Gillat Kol, Raghuvansh R. Saxena, and Zhijun Zhang. Binary codes with resilience beyond 1/4 via interaction. In *63rd IEEE Annual Symposium on Foundations of Computer Science, FOCS 2022, Denver, CO, USA, October 31 - November 3, 2022*, pages 1–12. IEEE, 2022. 18

[FK00] Uriel Feige and Joe Kilian. Finding OR in a noisy broadcast network. *Information Processing Letters*, 73(1-2):69–75, 2000. 6

[Gal88] Robert G. Gallager. Finding parity in a simple broadcast network. *IEEE Transactions on Information Theory*, 34(2):176–180, 1988. 6

[Gam87] Abbas El Gamal. Open problems presented at the 1984 workshop on specific problems in communication and computation sponsored by bell communication research. *"Open Problems in Communication and Computation", by Thomas M. Cover and B. Gopinath (editors). Springer-Verlag*, 1987. 1, 5

[Gel17] Ran Gelles. Coding for interactive communication: A survey. *Foundations and Trends in Theoretical Computer Science*, 13(1–2):1–157, 2017. 6

[GH14] Mohsen Ghaffari and Bernhard Haeupler. Optimal error rates for interactive coding ii: Efficiency and list decoding. In *Foundations of Computer Science (FOCS)*, pages 394–403, 2014. 6

[GHM18] Ofer Grossman, Bernhard Haeupler, and Sidhanth Mohanty. Algorithms for noisy broadcast with erasures. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 107 of *LIPIcs*, pages 153:1–153:12, 2018. 6

[GHS14] Mohsen Ghaffari, Bernhard Haeupler, and Madhu Sudan. Optimal error rates for interactive coding i: Adaptivity and other settings. In *Symposium on Theory of computing (STOC)*, pages 794–803, 2014. 6

[GI05] Venkatesan Guruswami and Piotr Indyk. Linear-time encodable/decodable codes with near-optimal rate. *IEEE Transactions on Information Theory*, 2005. 18

[GK19] Ran Gelles and Yael T Kalai. Constant-rate interactive coding is impossible, even in constant-degree networks. *IEEE Transactions on Information Theory*, 65(6):3812–3829, 2019. 6

[GKR21] Ran Gelles, Yael Tauman Kalai, and Govind Ramnarayan. Efficient multiparty interactive coding—part i: Oblivious insertions, deletions and substitutions. *IEEE Transactions on Information Theory*, 67(6):3411–3437, 2021. 6, 7

[GKR22] Ran Gelles, Yael Tauman Kalai, and Govind Ramnarayan. Efficient multiparty interactive coding—part ii: Non-oblivious noise. *IEEE Transactions on Information Theory*, 68(7):4723–4749, 2022. 6, 7

[GKS08] Navin Goyal, Guy Kindler, and Michael Saks. Lower bounds for the noisy broadcast problem. *SIAM Journal on Computing*, 37(6):1806–1841, 2008. 6

[GMS11] Ran Gelles, Ankur Moitra, and Amit Sahai. Efficient and explicit coding for interactive communication. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, pages 768–777. IEEE, 2011. 6

[Hae14] Bernhard Haeupler. Interactive channel capacity revisited. In *Foundations of Computer Science (FOCS)*, pages 226–235. IEEE, 2014. 6

[HS16] William M. Hoza and Leonard J. Schulman. The adversarial noise threshold for distributed protocols. In *Symposium on Discrete Algorithms (SODA)*, pages 240–258, 2016. 1, 6, 7

[JKL15] Abhishek Jain, Yael Tauman Kalai, and Allison Bishop Lewko. Interactive coding for multiparty protocols. In *Symposium on Theory of computing (STOC)*, pages 1–10, 2015. 1, 6, 7

[KM05] Eyal Kushilevitz and Yishay Mansour. Computation in noisy radio networks. *SIAM Journal on Discrete Mathematics (SIDMA)*, 19(1):96–108, 2005. 6

[LV15] Allison Lewko and Ellen Vitercik. Balancing communication for multi-party interactive coding. *arXiv preprint arXiv:1503.06381*, 2015. 6

[MG21] Manuj Mukherjee and Ran Gelles. Multiparty interactive coding over networks of intersecting broadcast links. *IEEE Journal on Selected Areas in Information Theory*, 2(4):1078–1092, 2021. 6

[New04] Ilan Newman. Computing in fault tolerance broadcast networks. In *Computational Complexity Conference (CCC)*, pages 113–122, 2004. 6

[PR09] David Peleg and Tomasz Radzik. Time-efficient broadcast in radio networks. In *Graphs and Algorithms in Communication Networks*, pages 311–334. Springer, 2009. 2

[RS94] Sridhar Rajagopalan and Leonard J. Schulman. A coding theorem for distributed computation. In *Symposium on the Theory of Computing (STOC)*, pages 790–799, 1994. 6

[Sch92] Leonard J Schulman. Communication on noisy channels: A coding theorem for computation. In *Foundations of Computer Science (FOCS)*, pages 724–733. IEEE, 1992. 6

[Yao97] Andrew Chi-Chih Yao. On the complexity of communication under noise. *invited talk in the 5th ISTCS Conference*, 1997. 6

A Interactive Coding

We sketch an informal proof showing that any interactive coding scheme either has exponentially small rate or erasure resilience at most $\mathcal{O}(\frac{1}{n})$. For this, consider the following "pointer-chasing" task. Player 1 sends an n-bit message, then player 2 sends an n-bit message depending on the message of player 1, then player 3 sends an n-bit message depending on both these messages, and so on.

Consider a protocol Π' for this task that is resilient to a $\frac{1}{n}$-fraction of erasures. In particular, the protocol Π' outputs correctly if the first $\frac{2}{n}$-fraction of received messages are erased for player 2, then the next $\frac{2}{n}$-fraction are erased for player 3 and so on. This means that player 2 cannot hear the n-bit message of player 1 in the first $\frac{2}{n}$-fraction of the protocol. Without this message, the only way the pointer-chasing task can "progress" in the first $\frac{2}{n}$-fraction of the protocol is if player 2 broadcasts its response to all possible messages of player 1, which would require an exponentially small rate.

Thus, if the protocol does not have an exponentially small rate, player 2 must wait out the first $\frac{2}{n}$-fraction of the protocol before broadcasting its message. As the next $\frac{2}{n}$-fraction of the protocol is erased for player 3, it must wait out this part of the protocol before sending its message. Continuing this way, we get that the parties run out of rounds of the protocol before completing the pointer-chasing task, finishing the proof.

ECCC ISSN 1433-8092
https://eccc.weizmann.ac.il