Electronic Colloquium on Computational Complexity, Report No. 129 (2022)

Binary Codes with Resilience Beyond 1/4 via Interaction

Klim Efremenko[*]
Ben-Gurion University

Gillat Kol[†]
Princeton University

Raghuvansh R. Saxena[‡]
Microsoft

Zhijun Zhang[§]
Princeton University

Abstract

In the *reliable transmission problem*, a sender, Alice, wishes to transmit a bit-string x to a remote receiver, Bob, over a binary channel with adversarial noise. The solution to this problem is to encode x using an *error correcting code*. As it is long known that the distance of binary codes is at most $\frac{1}{2}$, reliable transmission is possible only if the channel corrupts (flips) at most a $\frac{1}{4}$-fraction of the communicated bits.

We revisit the reliable transmission problem in the two-way setting, where both Alice and Bob can send bits to each other. Our main result is the construction of *two-way error correcting codes* that are resilient to a constant fraction of corruptions strictly larger than $\frac{1}{4}$. Moreover, our code has constant rate and requires Bob to only send one short message. We mention that our result resolves an open problem by Haeupler, Kamath, and Velingker [APPROX-RANDOM, 2015] and by Gupta, Kalai, and Zhang [STOC, 2022].

Curiously, our new two-way code requires a fresh perspective on classical error correcting codes: While classical codes have only one distance guarantee for all pairs of codewords (*i.e.*, the minimum distance), we construct codes where the distance between a pair of codewords depends on the "compatibility" of the messages they encode. We also prove that such codes are necessary for our result.

[*] klimefrem@gmail.com
[†] gillat.kol@gmail.com
[‡] raghuvansh.saxena@gmail.com
[§] zhijunz@princeton.edu

ISSN 1433-8092

Contents

1 Introduction

As errors are everywhere, essentially any telecommunications system crucially uses *error correcting codes*. Classical "one-way" error correcting codes date back to the 40's [Sha48] and are designed to solve the *reliable transmission problem*, where a sender, Alice, wishes to send a message x to a remote receiver, Bob, but she can only communicate with him over a noisy one-way channel that corrupts some of her communication.

As the price of interaction goes down, systems are becoming more interactive. In this paper we study *two-way error correcting codes*, that are designed to solve the same problem, but work assuming a two-way channel instead of a one-way channel, allowing the parties to interact. Specifically, we consider the reliable transmission problem, where Alice and Bob are connected by a pair of *binary* channels with *adversarial corruption noise* (bit flips), one in each direction.

The two most important parameters in the study of error correcting codes are the (relative) *distance* and the *rate* of the code. Plotkin showed in the 60's [Plo60] that the minimum (or even average) relative distance of a binary error correcting code is at most $\frac{1}{2}$, which implies that binary codes can be resilient to up to $\frac{1}{4}$ fraction of adversarial errors. *Can interaction improve the error resilience of binary codes?*

1.1 Our Result

1.1.1 Main Result: Binary Two-Way Codes with Error Resilience $> 1/4$

The main result of this paper is Theorem 1.1 (see Theorem 5.1 for a formal statement), that gives a positive answer to the above question, resolving an open problem by Haeupler, Kamath, and Velingker (Section 6 in [HKV15]) and by Gupta, Kalai, and Zhang (Section 1.1 in [GKZ22]).

Theorem 1.1 (Main, Informal). *There exists a* constant rate*, deterministic, binary two-way error correcting code with error resilience* $\frac{1}{4} + 10^{-5}$*, where Bob sends a* single *message.*

We mention that in the two-way code we construct, Bob sends a single message whose length is less than 2% of the total communication, thus showing that even a minimal amount of interaction can already improve the noise resilience of binary codes, while keeping their rate constant. Finding the maximal noise tolerance of (constant rate or even zero-rate) binary two-way codes is an intriguing question we leave open.

1.1.2 Impossibility for Equally-Spaced Codes

To construct the binary two-way code promised by Theorem 1.1, we design a new binary one-way code where the guaranteed distance between certain carefully chosen pairs of "compatible" codewords is strictly greater than $\frac{1}{2}$ (at a high level, the adversary is more likely to want to confuse bewteen these pairs of codewords). Recall, however, that by the *Plotkin* *bound*, the average distance of a binary code is at most $\frac{1}{2}$, thus the distance between some of the other pairs of codewords is strictly smaller than $\frac{1}{2}$.

Theorem 1.2 below (see Theorem 7.1 for a formal statement) shows that these types of codes are required to obtain our result, as solely using *equally-spaced codes*, where the distance between every pair of codewords is roughly equal, is insufficient for breaking the $\frac{1}{4}$ error resilience barrier. This is in contrast to the one-way setting where equally-spaced codes attain the maximum resilience (*e.g.*, random binary codes are equally-spaced and have resilience of $\frac{1}{4}$). We also mention that all prior work concerned with the noise resilience of two-way channels and channels with feedback, surveyed in Section 1.2, essentially only uses equally-spaced codes (see Section 2.3).

Theorem 1.2 (Informal)**.** *The maximum error resilience of a binary two-way error correcting code (of any rate) that uses equally-spaced codes and has Bob sending a single message is* $\frac{1}{4}$.

Why non-equally-spaced? Intriguingly, when we started this project over a year ago, we believed that the answer to the above question should be negative, that is, that binary two-way codes cannot break the $\frac{1}{4}$ resilience barrier. In fact, we had a sketch of an impossibility result designed to show that any two-way code (say, with three messages, Alice, Bob, Alice) is essentially of the following form, and that codes of this form cannot have resilience better than $\frac{1}{4}$: On input x, Alice sends x encoded by a list-decodable code (for simplicity, assume lists are of size 2). Bob decodes to obtain two candidates x_1 and x_2 with the promise $x \in \{x_1, x_2\}$. Bob sends $\{x_1, x_2\}$ encoded by a list-decodable code. Alice decodes to obtain two candidates of the form $\{x, x_3\}$ and $\{x, x_4\}$. Observe that after this second message, the parties are left with the following communication task F: Bob knows $\{x_1, x_2\}$ such that $x \in \{x_1, x_2\}$, Alice knows $x, \{x_3, x_4\}$ such that $\{x_1, x_2\} \subseteq \{x, x_3, x_4\}$. Bob wishes to learn x. The two-way code then solves F with a single message from Alice to Bob.

Our strategy for proving an impossibility result was to show the following two lemmas: (1) A round elimination lemma saying that a reliable transmission protocol with resilience γ implies a one-message protocol for the task F with resilience γ. (2) A lemma showing that F cannot be solved with a single message of constant size over the *noiseless* channel.[1] We observed that under the assumption that Alice uses an equally-spaced code C to solve F,[2] the second lemma implies that the distance between any two messages she may send is at most $\frac{1}{2}$.[3] In this case F cannot have resilience better than $\frac{1}{4}$, and using the first lemma, we get our impossibility result. However, as should be expected, despite our best effort, we were unsuccessful in showing that the assumption is without loss of generality... Never-

[1] The task F is interesting on its own right. In Section 7 we show that if $x, x_1, x_2, x_3, x_4 \in [N]$, then the one-way communication complexity of F is $\Theta(\log \log N)$. We also mention that F is very related to compression efforts by [Orl90, HS16].

[2] That is, the set C of all possible messages by Alice for all possible inputs, forms an equally-spaced code.

[3] In more detail, for equally-spaced codes, the distance between every pair of codewords is roughly the minimum distance, and the minimum distance of codes of super-constant size is $\frac{1}{2}$.

theless, under this assumption, we were able to formalize this impossibility sketch, yielding Theorem 1.2 (see additional discussion in Section 2.2).

1.2 Related Work

1.2.1 Two-Way Erasure Codes

In a recent work, Gupta, Kalai, and Zhang [GKZ22] study two-way error correcting codes over the adversarial binary *erasure channel*, where the adversary may replace some of the sent bits by '?'. Their main result is a code that is resilient to a $\frac{3}{5}$ fraction of adversarial errors, improving on the noise tolerance of the one-way binary erasure channel that is known to be $\frac{1}{2}$. Gupta and Zhang [GZ22b] give a two-way code over the same channel that is also of constant rate. We mention that the two-way coding schemes of [GKZ22, GZ22b] exchange (almost) linear number of messages and are generally very different than ours.

[GKZ22] also give an upper bound of $\frac{2}{3}$ on the maximum tolerance of the two-way adversarial binary erasure channel, and an upper bound of $\frac{2}{7}$ on the maximum tolerance of the two-way adversarial binary channel (the model assumed by our work). As mentioned above, bridging the gap between our lower bound on the noise tolerance and their upper bound is a great problem.

1.2.2 Reliable Exchange and Interactive Coding

Reliable exchange. In the *reliable exchange problem*, two parties, Alice, holding a private input x, and Bob, holding a private input y, communicate over the two-way binary adversarial channel with the goal of learning each other's input. Observe that the reliable transmission problem is at least as hard as the reliable exchange problem, in the sense that a transmission protocol with resilience θ implies an exchange protocol with resilience $\frac{\theta}{2}$: Alice sends x using the transmission protocol, then Bob sends y using the transmission protocol. Now, if an adversary corrupts at most $\frac{\theta}{2}$ fraction of the total communication, it also corrupts at most θ fraction of each transmission and both transmissions go through. Since one-way codes solve the transmission problem with error resilience $\frac{1}{4}$, the exchange problem is easily solvable with error resilience $\frac{1}{8}$.

Efremenko, Kol, and Saxena [EKS20b] show how to go beyond $\frac{1}{8}$ and obtain an exchange protocol that is resilient to a $\frac{5}{39}$ fraction of adversarial errors with a constant number of rounds and constant overhead. As explained in Section 2, we use [EKS20b] as a stepping stone towards our two-way code. The resilience constant was later improved by [GZ22a] from $\frac{5}{39}$ to $\frac{1}{6}$, which was known to be optimal [Ber68, EGH16]. Note however, that the [GZ22a] protocol has linear overhead and many communication rounds.

Interactive coding. The reliable exchange problem (and therefore also reliable transmission) are special cases of the interactive coding problem: Given a two-party communication protocol Π that works assuming the noiseless channel, simulate Π by a protocol Π' that works over a noisy channel. The study of interactive coding was first suggested in seminal works by Schulman [Sch92, Sch93, Sch96], and is now an active research area, see [Gel17] for an excellent survey.

Observe that the transmission problem corresponds to the noiseless protocol Π where Alice sends her input x, and the exchange problem corresponds to Π where Alice sends her input x and Bob sends his input y. The exchange problem is also "complete" in the sense that after exchanging x and y the parties can run any other protocol without communication. Thus, an exchange protocol resilient to θ fraction of errors implies an interactive coding scheme with the same resilience. Note however that this scheme may have a huge overhead.

Braverman and Rao [BR11], building on [Sch96], gave an interactive code with constant overhead and optimal resilience of $\frac{1}{4}$ for the case where the alphabet set is large, and showed that it implies a binary interactive code with resilience $\frac{1}{8}$; see also [BE17] on asymmetric corruptions. [EKS20b] gave a binary interactive code with constant overhead that is resilient to a $\frac{5}{39}$ fraction of errors (that is, they showed that their reliable exchange scheme can be generalized to an interactive coding scheme).

The maximum resilience of interactive coding schemes was also studied for other channels, such as the erasure channel, the channel with feedback, and the insertion-deletion channel [FGOS15, EGH16, GH17, Pan13, BGMO17, SW17, HSV18]. Another channel that received quite a bit of attention in this context is the adaptive channel, where several parties may speak at the same round and collisions may occur[4] [GHS14, GH14, EKS20a, EKS21].

1.2.3 Reliable Transmission with Feedback

The works surveyed so far consider two-way channels, but are inspired by classical results from the 60's showing that the maximum resilience of one-way error correcting codes can be improved assuming the channel provides *feedback*. At a very high level, these two-way results work by implementing feedback (over channels with no build-in feedback) using interaction.

In more detail, the *feedback channel* allows Alice to communicate symbols to Bob, but upon receiving each sent symbol, Bob sends the received symbol back to Alice as feedback [Ber64]. Alice can then use it when deciding what to send next. Note that it is typically assumed that Bob's feedback is not corrupted by the channel.

Observe that any protocol that can be run over a two-way channel can also be run over the feedback version of the same channel: Given a communication protocol over the two-way channel, Alice can simulate the messages sent by Bob as she knows everything he knows (which, as Bob has no input, is just his received transcript). The two main differences between two-way channels and feedback channels are: (1) The noise in a feedback channel is one-way: while the communication from Alice to Bob may be noisy, the communication from Bob to Alice is noiseless. (2) For a feedback channel, the length of the communication is defined as the number of rounds where Alice communicates (Bob's feedback rounds do not

[4] More formally, the communication order is not predetermined. In every round, each party may decide whether to send or listen depending on his input and received transcript.

count towards the length of the protocol). In particular, the noise tolerance is measured as a fraction of Alice's rounds.

While Shannon showed that feedback does not increase the maximum noise tolerance of stochastic channels [Sha56], feedback can, in fact, increase the noise tolerance of adversarial channels [Ber68, Zig76, SWS92, ADL06]. Specifically, it was shown by Berlekamp that the noise tolerance of the binary adversarial channel increases from $\frac{1}{4}$ to $\frac{1}{3}$ given feedback [Ber68]. Haeupler, Kamath, and Velingker [HKV15] considered the setting where the feedback is partial, and showed that even if Alice receives feedback bits from Bob for an arbitrarily small constant fraction of her transmissions, resilience close to $\frac{1}{3}$ is possible.

Partial *noisy* feedback was considered by Wang, Qin, and Chang [WQC17], who constructed a binary two-way code that is resilient to any constant fraction strictly smaller than 1 of adversarial erasures from Bob to Alice, but only up to $\frac{1}{2}$ fraction of adversarial erasures from Alice to Bob (*cf.* [GKZ22], where the *total* noise tolerance is strictly greater than $\frac{1}{2}$).

2 Proof Sketch

We now give a detailed overview of our main result, a 3-message (or a 3-step) protocol for the reliable transmission problem with error resilience strictly larger than $\frac{1}{4}$. Recall that the maximum distance possible using a binary code depends on the number of codewords it has. Precisely, if one wants an even[5] number C of codewords, then the maximum distance[6] one can get equals $\frac{C}{2(C-1)}$. Put another way, this means that if Alice wants to send one of C strings to Bob over a *one-way* channel that corrupts (flips) a γ fraction of the symbols sent, she can do this if and only if $\gamma < \frac{C}{4(C-1)}$. As C goes to infinity, this bound becomes $\gamma < \frac{1}{4}$, which is the error resilience of the one-way channel. However, for smaller values of C, the error resilience is much higher: It is $\frac{1}{2}$ when $C = 2$, as demonstrated by the codewords $00\cdots0$ and $11\cdots1$, and $\frac{1}{3}$ when $C = 4$, *etc.*

As a prelude to our main result, we first overview the result of [EKS20b]. We mention that [EKS20b] were concerned with the reliable exchange problem and the interactive coding question (see Section 1.2.2), and that we are presenting a simplified and slightly modified version of their scheme for the restricted case where Bob has no input and the channel has one-way error, meaning that the adversary can only corrupt symbols sent from Alice to Bob, but cannot corrupt the ones sent from Bob to Alice[7].

[5] The bound is slightly different for odd numbers, but exhibits the same phenomenon.

[6] Recall that the distance of a code is defined as the minimum distance between any two codewords. However, the bound stated here even holds for the average distance between pairs of codewords.

[7] The one-way error setting differs from the noiseless feedback setting (see Section 1.2.3) in two respects: (1) It allows Bob to send any value it wishes as feedback. (2) The length of the protocol is the total number of bits communicated by *both* parties.

2.1 The [EKS20b] Result: Beating $\frac{1}{4}$ Given One-Way Error

At an extremely high level, the [EKS20b] protocol works by using two-way communication (with one-way error) to reduce the effective number of codewords that Alice and Bob need to consider, and then uses the higher resilience guarantees achievable by codes with small C to get an improved error resilience.

In more detail, [EKS20b] describe a 3-message protocol where in the first step, Alice sends to Bob an encoding of her input x using a *list-decodable* error correcting code. Bob's goal in this step is not to recover x exactly, which would limit the error resilience to $\frac{1}{4}$, but instead to compute a set of size 2,[8] say $S = \{x_1, x_2\}$ such that $x \in S$. This weaker guarantee, known as list-decoding, allows the parties to tolerate strictly higher than $\frac{1}{4}$ errors in this step.

Then, in the second step, Bob sends the set S to Alice, and the one-way error guarantee implies that Alice will always receive the set S correctly. Additionally, the one-way error guarantee also means that the second step can be arbitrarily short.

Overall, these two steps guarantee that before the third step begins, both Alice and Bob agree on a set S of size 2 that contains x, and in the third step Alice only needs to tell Bob which of the two elements in S is her correct input. As S is of size 2, this can be done with a high error resilience using the codes with *small* C discussed above.

2.2 Challenges in Going from One-Way to Two-Way Error

The [EKS20b] protocol described above breaks down once the adversary is allowed to corrupt the bits sent from Bob to Alice in the second step, in addition to those sent from Alice to Bob in the first and the third steps. This poses the following challenges:

No unique-decoding the second step. First and foremost, the guarantee that Alice and Bob agree on a small set S containing x that [EKS20b] crucially relied on no longer holds in the two-way error case. As the number of possible sets Bob could send in this step is large, by corrupting just a $\frac{1}{4}$ fraction of this step, the adversary can make sure that the set Alice decodes is different from the set Bob sent. This is fatal to the protocol, as without the common knowledge of S, Bob has no way to interpret the message sent by Alice in the third step, which means he cannot output x.

List-decoding the second step seems insufficient. A possible remedy that one might consider for the foregoing problem is to have Bob send the set S to Alice using a list-decodable error correcting code similarly to the way Alice sent him x in the first step. This would allow Alice to have two sets T_1, T_2, both containing x such that one of the two sets equals S. One may then hope to suitably adapt the third step of the protocol to work with this weaker guarantee while still using codes with small C.

[8] We mention that the bounds one gets for list-decoding implies that the set S needs to be of size at least 3 to get guarantees better than $\frac{1}{4}$. Nonetheless, for the sake of this sketch, we shall stick to sets of size 2.

It turns out that such an adaptation is impossible. As we show in Section 7, specifically, in Lemma 7.4, the third step in any such adaptation must use a super-constant number C of codewords (also see discussion after Theorem 1.2). However, a super-constant C means that the error resilience guarantee reduces to $\frac{1}{4}$, and we get no improvement.

List-decoding the second step is *impossible*. Not only does it seem hard to get a protocol that works given a list-decoding guarantee for the second step (*i.e.*, assuming Alice obtains two sets T_1, T_2 such that one of them is S), but it is actually *impossible* for Alice to obtain such sets T_1, T_2. The reason is that any protocol for the transmission problem with error resilience larger than $\frac{1}{4}$ must have Bob speak in at most a $\frac{1}{4}$ fraction of the communication rounds. This is because of a classical result by [Ber64, Ber68] showing that even when Alice knows everything Bob knows, the maximum error resilience possible (as a fraction of Alice's rounds) is $\frac{1}{3}$ (see Section 1.2.3). Therefore, if Bob speaks in more than a $\frac{1}{4}$ fraction of the rounds, Alice speaks in at most $\frac{3}{4}$ fraction of the rounds. Now, even if the adversary does not corrupt Bob at all and Alice knows everything Bob knows, the maximum error resilience is at most $\frac{1}{3} \cdot \frac{3}{4} = \frac{1}{4}$.[9]

However, the facts that Bob speaks in at most a $\frac{1}{4}$ fraction of the rounds, and that the adversary can corrupt strictly more than a $\frac{1}{4}$ fraction of the rounds, mean that the adversary can, if he wants, corrupt Bob's transmission entirely (that is, corrupt the communication from Bob to Alice in all 3 steps). Since Bob is effectively shut off with this attack, Alice will not be able to obtain sets T_1, T_2 such that one of them is S. Thus, even if one could adapt the third step of the [EKS20b] protocol to work when Alice has a list of 2 sets (which has its own challenges), one cannot guarantee that Alice will have this list.

2.3 Our Main Idea: Non-Equally-Spaced Codes

Although the challenges mentioned above make a pretty solid case for an impossibility result, we were able to construct a two-way scheme with resilience better than $\frac{1}{4}$ using new ideas. Our main new idea, and where our work differs most significantly from all prior work, is the use of codes where not all codewords are *equally-spaced*. Note that the distance between any two codewords sent by Alice roughly captures the amount of corruptions that the adversary needs to invest to confuse Bob between those two codewords. We observe that, in our protocol, some pairs of codewords are more prone to be corrupted by the adversary than others, and ensure that such pairs have a high distance to begin with, thereby implying that the adversary needs a high number of corruptions to confuse Bob.

Implementing this idea is not simple, as whatever code we come up with still needs to obey the aforementioned bound of $\frac{C}{2(C-1)}$ on the average distance between codewords. Thus, if we want to have a higher distance between some pairs of codewords, we must also have a

[9] We mention that this idea can also be used to get an upper bound of $\frac{2}{7}$ on the maximum error resilience, without any restriction on the number of rounds Bob speaks in, see [GKZ22].

lower distance between some other pair of codewords, and we need to ensure that these low distance pairs are carefully chosen to not affect the overall error resilience of our protocol.

Before explaining which pairs of codewords can have a lower than average distance and which pairs need to be farther apart, we mention that such a careful analysis of the distance is a novelty of our paper. Most prior work in the area of error resilience, and the area of binary codes in general, only uses one measure, the minimum distance between two codewords, when understanding the distance properties of a given code. Moreover, many constructions have a lot of "symmetry", *e.g.*, picking codewords at random, that implies the distance between any two pairs of codewords is more or less the same. This also holds for the codes with small C that were used in the [EKS20b] protocol and followup work (see Section 1.2), with [GZ22a] being a minor exception as the codes in [GZ22a], albeit not equally-spaced overall, can be seen as a union of a small number (four) of codes that are equally-spaced, and still have the issues described above.

2.4 Distance Requirements for Pairs of Codewords

We now explain why our protocol benefits if certain pairs of codewords are farther apart than other pairs of codewords. Recall that our protocol has 3 steps, and let L_1, L_2, L_3 be the lengths of these steps, and let $T = L_1 + L_2 + L_3$. Our protocol sticks to the framework in Sections 2.1 and 2.2 for the first two steps, using equally-spaced codes for these two steps.

In this section we are going to make the following two simplifying assumptions: first is that at the end of the first step Bob has a list of size two instead of three, second one is that at the second round adversary can either completely corrupt message to another codeword by investing $L_2/2$ errors or not to corrupt this message at all. Removing this assumptions is not trivial and we will discuss it in details in Section 2.6.

As a result, if x is the input that Alice starts the protocol with, then at the end of the first step Bob has a set S of size 2 that is guaranteed to contain x. Moreover, Bob knows that Alice's encodings for the two elements of S are at least $L_1/2$ apart in Hamming distance (as the list-decoding radius is $\frac{1}{2}$).

For the second step, Alice gets a set T of size 2 that contains x, as she knows that Bob encoded a set that contains x, but is otherwise arbitrary. As Alice knows both T and x, this is equivalent to her knowing an *ordered* pair (x, a), where x is her input and a is the element in T that is different from her input. Moreover, Bob knows that if Alice indeed got a set $T \neq S$, then the adversary must have invested at least $L_2/2$ corruptions in the second step.

For the third step, Alice encodes an ordered pair (x, a), and sends the encoding to Bob. As Bob knows a set S of size 2 that contains x, say $S = \{x, x'\}$, where $x \neq x'$, he knows that Alice either encoded a pair of the form (x, a) or a pair of the form (x', b), for some inputs $a \neq x, b \neq x'$, and his goal is to use the messages he received in the first and third step to figure out whether the pair sent was of the form (x, a) or the form (x', b). Note that he does not need to know what the pair was exactly as his goal is to output Alice's input (and not the pair).

If the target error resilience is $\frac{1}{4}+\theta$, Bob can achieve this goal only if it is impossible for two pairs, one of the form (x,a) and the other of the form (x',b), to give rise to the same messages (m_1,m_3) in the first and the third steps, with at most $\left(\frac{1}{4}+\theta\right)\cdot T$ corruptions. With this in mind, let us now look at the set of messages in the first and third steps that the adversary can generate using $\left(\frac{1}{4}+\theta\right)\cdot T$ corruptions from pairs of the form (x,a).

Let ECC_1 and ECC_3 denote the error correcting codes used by Alice in the first and third steps respectively, so that the messages Alice sends are $\mathsf{ECC}_1(x)$ and $\mathsf{ECC}_3(x,a)$. As explained above, we either have $a=x'$ or the adversary spent at least $L_2/2$ corruptions in the second step. Let Δ denote the Hamming distance, and let

$$f(x,x',m_1,m_3)=\Delta(\mathsf{ECC}_1(x),m_1)+\Delta(\mathsf{ECC}_3(x,x'),m_3).$$

The above discussion implies that all the pairs of messages (m_1,m_3) that the adversary can generate using $\left(\frac{1}{4}+\theta\right)\cdot T$ corruptions from the pair (x,a) must either satisfy $a=x'$ (that is, Alice's second element is one of the elements in Bob's list) and

$$f(x,x',m_1,m_3)\leq\left(\frac{1}{4}+\theta\right)\cdot T, \tag{1}$$

or satisfy $a\notin\{x,x'\}$ (that is, Alice's second element is not one of the elements in Bob's list) and

$$f(x,a,m_1,m_3)\leq\left(\frac{1}{4}+\theta\right)\cdot T-\frac{L_2}{2}, \tag{2}$$

Using these inequalities and the fact that Alice's encodings for x and x' in the first step are $L_1/2$ apart in Hamming distance, one gets that the following guarantees on the distances between codewords of ECC_3 are both necessary and sufficient for a protocol to have error resilience $\frac{1}{4}+\theta$:

1. For Bob to not be confused between messages for the pairs (x,x') and (x',x) we need there not to be a pair (m_1,m_3) that satisfies both $f(x,x',m_1,m_3)\leq\left(\frac{1}{4}+\theta\right)\cdot T$ and $f(x',x,m_1,m_3)\leq\left(\frac{1}{4}+\theta\right)\cdot T$. The reason is that if Alice has (x,x') then her input is x and her second element is in Bob's list. Similarly, if Alice has (x',x) then her input is x' and her second element is in Bob's list. Thus, on both pairs we can use Eq. (1).

   Take m_1 to be the middle point between $\mathsf{ECC}_1(x)$ and $\mathsf{ECC}_1(x')$ (a point with equal Hamming distance from both) and m_3 be the middle point between $\mathsf{ECC}_3(x,x')$ and $\mathsf{ECC}_3(x',x)$. Very roughly, we claim that these are the "worst" m_1 and m_3.

   By summing the two f inequalities for this m_1 and m_3 we get

$$\Delta(\mathsf{ECC}_1(x),\mathsf{ECC}_1(x'))+\Delta(\mathsf{ECC}_3(x,x'),\mathsf{ECC}_3(x',x))\leq 2\left(\frac{1}{4}+\theta\right)\cdot T.$$

   Recall that $\Delta(\mathsf{ECC}_1(x),\mathsf{ECC}_1(x'))=L_1/2$ and that $T=L_1+L_2+L_3$, and get that we must have:

$$\Delta(\mathsf{ECC}_3(x,x'),\mathsf{ECC}_3(x',x)) \geq \frac{L_3}{2} + 2\theta T + \frac{L_2}{2}.$$

2. For Bob to not be confused between messages for the pairs (x,a) and (x',x) (and similarly for pairs (x,x') and (x',b)), we must have:

$$\Delta(\mathsf{ECC}_3(x,a),\mathsf{ECC}_3(x',x)) \geq \frac{L_3}{2} + 2\theta T.$$

   This follows from a similar argument to Item 1 where we use Eq. (1) on (x',x) and Eq. (2) on (x,a).

3. For Bob to not be confused between messages for the pairs (x,a) and (x',b) (note that it is possible that $a=b$ but both are always different from x and x'), we must have:

$$\Delta(\mathsf{ECC}_3(x,a),\mathsf{ECC}_3(x',b)) \geq \frac{L_3}{2} + 2\theta T - \frac{L_2}{2}.$$

   This follows from similar argument to Item 1 when we use Eq. (2) on both pairs.

Importantly, as the number of pairs that require the weakest possible guarantee in Item 3 above is much larger than those requiring the stronger guarantees, if we choose $\theta > 0$ small enough so that L_2 is significantly larger than θT, say, $L_2 = 16\theta T$, these distance requirements do not violate the $L_3/2$ bound on the average distance implied by the Plotkin bound. Thus, such an ECC_3 code is theoretically possible, and we provide a construction below.

2.5 Location-Sensitive Codes: The Construction

The upshot of the discussion above is that the code Alice uses in the third step to encode pairs must ensure that the distance between the encodings of pairs that have one or two common elements *at different locations* must be large (Items 1 and 2), while the distance between the encodings of pairs that have no common elements or the common elements are at *the same location* (Item 3) can be smaller than the distance obtained by a random code.

We call such a code a *location-sensitive code* and note that its distance guarantees are very different from standard equally-spaced codes. To capture the fact that the distances need to be larger between the encodings of two pairs where the same element appears in different locations, we employ the following approach while encoding a pair (s,t): Let C be an equally-spaced code that encodes a single input s and has relative distance $\frac{1}{2}$ between pairs of codewords. We construct our location-sensitive code $\mathsf{LSC}(s,t)$ by setting each coordinate i to be coordinate i of $\mathsf{C}(s)$ with probability p and coordinate i of $\overline{\mathsf{C}(t)}$ with probability $1-p$, for some carefully selected $p > 0$. In other words, the encoding $\mathsf{LSC}(s,t)$ is *positively correlated* with $\mathsf{C}(s)$ and *negatively correlated* with $\mathsf{C}(t)$.

Roughly speaking, the above approach has the property that for pairs (x,a) and (x',x) where the same element appears at different locations, the encodings $\mathsf{LSC}(x,a)$ and $\mathsf{LSC}(x',x)$

will be positively and negatively correlated with $\mathsf{C}(x)$ respectively, and therefore should have a high distance. In contrast, for pairs (x, a) and (x', a) where the same element appears at the same location, the encodings $\mathsf{LSC}(x, a)$ and $\mathsf{LSC}(x', a)$ will both be negatively correlated with $\mathsf{C}(a)$ and will suffer from a lower distance. By choosing the parameters carefully, we are able to meet the requirements in Section 2.4.

2.6 Finalizing the Details

We finish the overview with a discussion of some remaining issues that we have not covered well so far.

Other attacks in the second step. One assumption that we made in our discussion above is that Alice always gets the encoding of one pair in the second step. This loses generality as nothing stops the adversary from giving Alice a combination of the encoding of various pairs. To get around this, we use list-decoding in the second step to give Alice a pair of pairs that contains the right pair unless there were too many corruptions.

When Alice tries to encode this pair of pairs in the third step using a location-sensitive code, she actually just encodes both the pairs separately, and simply sends a message that is positively correlated with both the encodings. The actual correlations intricately depend on the exact message received by her in the second step, with larger correlations to one of the pairs if the encoding of that pair was not too far from what Alice received in the second step.

Sets arising from list-decoding. Another assumption we made throughout the above sketch was that using list-decodable codes allows the parties to compute sets of size 2 that are guaranteed to contain the correct codeword even if there are strictly more than $\frac{1}{4}$ errors. This assumption is not correct for binary codes, and one needs to have sets of size at least 3. Correspondingly, in the actual proof, we make the entire argument above work with list-decodable codes that yield lists of size 3.

One crucial change this entails is that our location-sensitive codes must also now take triples instead of pairs. While we believe that such location-sensitive codes still exist, we found it easier to convert triple to pairs in the following way: When Alice wants to encode a triple (x, a, b), where x is her true input and $a, b \neq x$, she instead encodes the pairs (x, a) and (x, b) and sends a message positively correlated with both these encodings. We then are able to make the analysis work by carefully controlling the correlations, which forms a lot of the technical work in this paper.

3 Model and Preliminaries

3.1 Notation

All logarithms are base 2. We use $\log^{(k)} n$ to denote the k-times iterated logarithms of n. For $n \in \mathbb{N}$, $x, y \in \{0,1\}^n$, we denote by $\Delta(x,y) = |\{i \in [n] \mid x_i \neq y_i\}|$ the Hamming distance between x and y. For a set S and an integer $k \geq 0$, the notation $\binom{S}{k}$ denotes the set of all subsets of S that have exactly k elements. Also define, for a set S, the set $\mathcal{D}(S)$ to be the set of all distributions over S.

3.2 Preliminaries

We will be using the following version of the Chernoff bound:

Lemma 3.1 (Chernoff bound)**.** *For all $n \in \mathbb{N}$, and independent random variables $X_1, \dots, X_n \in [0,1]$, let $S = \sum_{i \in [n]} X_i$. For all $t > 0$, we have:*

$$\Pr(|S - \mathbb{E}[S]| \geq t) \leq 2 \cdot \exp\bigl(-2t^2/n\bigr).$$

We will also use the following well-known result about multicolor Ramsey numbers. We include a proof for completeness.

Lemma 3.2 (Multicolor Ramsey numbers)**.** *Let $k, z > 1$ be integers and $n \geq 4^{(z-1)^{k-1}}$. Any complete graph on n vertices with each edge colored using one of k colors has a monochromatic complete subgraph of size z.*

Proof. We first rephrase the lemma in terms of Ramsey numbers. For $n > 1$ and integers $r_1, \cdots, r_n > 1$, let $R(n; r_1, \cdots, r_n)$ denote the minimum number of vertices such that for any coloring of the complete graph with $R(n; r_1, \cdots, r_n)$ with colors in $[n]$, there exists an $i \in [n]$ such that there is a complete subgraph of size r_i all of whose edges have color i. In this notation, we have to show that for all integers $k, z > 1$, we have:

$$R(k; z, \cdots, z) \leq 4^{(z-1)^{k-1}}.$$

Observe that the foregoing equation follows if we show that: (1) For all $s, t > 1$, we have $R(2; s, t) \leq \binom{t+s-2}{s-1} \leq t^{s-1}$. (2) For all $n > 2, s > 1$, we have $R(n; s, \cdots, s) \leq R(2; s, R(n-1; s, \cdots, s))$.

For Item 1, we proceed by induction on $s+t$. If either s or t equals 2, the result is trivial giving us our base case $s+t = 4$. We show the result for $s+t > 4$ assuming $s, t > 2$ and the result holds for $s+t-1$. For this, we shall show that $R(2; s, t) \leq R(2; s-1, t) + R(2; s, t-1)$ implying by our induction hypothesis that $R(2; s, t) \leq \binom{t+s-3}{s-2} + \binom{t+s-3}{s-1} = \binom{t+s-2}{s-1}$. Consider a fixed vertex, say vertex 1, in a complete graph with $R(2; s-1, t) + R(2; s, t-1)$ vertices. There are $R(2; s-1, t) + R(2; s, t-1) - 1$ edges incident on vertex 1, implying that either at least $R(2; s-1, t)$ have color 1 or at least $R(2; s, t-1)$ of them have color 2. We assume the former as the argument in the other case is symmetric. Consider the subgraph formed by the $\geq R(2; s-1, t)$ vertices that have an edge colored 1 connecting them to vertex 1. By the definition of $R(2; s-1, t)$ either this subgraph has a complete subgraph of size t all of whose edges have color 2, and we are done, or this subgraph has a complete subgraph of size $s-1$ all of whose edges have color 1, then we can add the vertex 1 to this subgraph and get a complete subgraph of size s of the original graph all of whose edges have color 1, finishing the proof.

For Item 2, consider a complete graph with $R(2; s, R(n-1; s, \cdots, s))$ vertices colored using n colors. Repaint this graph by keeping color 1 as is and painting all the other edges with a "super-color". By definition of $R(2; s, R(n-1; s, \cdots, s))$, either there is complete subgraph of size s all of whose edges have color 1, and we are done, or there is a complete subgraph of size $R(n-1; s, \cdots, s)$ all of whose edges have the super-color. In the latter case, we get a complete subgraph of the original graph of size at least $R(n-1; s, \cdots, s)$ all of whose edges are colored with $n-1$ colors. By definition of $R(n-1; s, \cdots, s)$, this has a monochromatic complete subgraph of size s, finishing the proof. □

3.3 The Binary Two-Way Communication Channel

We now define (deterministic) protocols over the binary two-way communication channel. Such a protocol is defined by a tuple:

$$\Pi = \left(\mathcal{X}^A, \mathcal{X}^B, \mathcal{Y}, T, p, f^A, f^B, \mathsf{out}\right),$$

where (1) $\mathcal{X}^A$ is the set of all possible inputs for Alice, (2) $\mathcal{X}^B$ is the set of all possible inputs for Bob, (3) $\mathcal{Y}$ is the set of all possible outputs (for Bob), (4) T is the length of the protocol (the number of rounds), (5) $p \in \{A, B\}^T$ is the order of turns, (6) $f^A : \mathcal{X}^A \times \{0,1\}^{<T} \to \{0,1\}$ is the message function for Alice, (7) $f^B : \mathcal{X}^B \times \{0,1\}^{<T} \to \{0,1\}$ is the message function for Bob, (8) $\mathsf{out} : \mathcal{X}^B \times \{0,1\}^T \to \mathcal{Y}$ is the output function (for Bob).

Execution of a protocol. An adversary for such a protocol is defined by a function $\mathsf{Adv} : \mathcal{X}^A \times \mathcal{X}^B \to \{0,1\}^T$. For $i \in [T]$, we shall use $\mathsf{Adv}_i(\cdot)$ to denote the function that outputs the i^{th} bit of $\mathsf{Adv}(\cdot)$. We next define an execution of Π in the presence of an adversary Adv for Π: At the beginning of the execution, Alice starts with an input $x^A \in \mathcal{X}^A$ and Bob starts with an input $x^B \in \mathcal{X}^B$. The execution consists of T rounds and before the i^{th} round, for $i \in [T]$, Alice and Bob have transcripts $\pi^A, \pi^B \in \{0,1\}^{i-1}$ respectively. In round i, if $p_i = A$, then Alice transmits the symbol $f^A(x^A, \pi^A)$ while Bob receives the symbol $\mathsf{Adv}_i(x^A, x^B)$. Both the parties add these symbols to π^A and π^B respectively. Similarly, if $p_i = B$, then Bob transmits the symbol $f^B(x^B, \pi^B)$ while Alice receives the symbol $\mathsf{Adv}_i(x^A, x^B)$. Both the parties add these symbols to π^A and π^B respectively.

After T such rounds, Bob outputs $\mathsf{out}(x^B, \pi^B)$. Observe that this execution, and therefore π^A and π^B, are completely determined by x^A, x^B, Π, and Adv. We denote the output of Π on inputs $x^A \in \mathcal{X}^A$ and $x^B \in \mathcal{X}^B$ in the presence of adversary Adv by $\mathsf{out}_{\Pi,\mathsf{Adv}}(x^A, x^B)$.

Phases. We say that $i \in [T-1]$ is an *alternation round* of Π if $p_{i+1} \neq p_i$. Assume that Π has P alternations and let $i_1 < i_2 < i_3 < \cdots < i_P$ be all alternation rounds. Define $i_0 = 0$ and $i_{P+1} = T$. For $t \in [P+1]$, we define *Phase* t of Π as the set of rounds $(i_{t-1}, i_t]$. Informally, Phase t is the t^{th} message by one of the parties.

Corruptions. Consider an execution of Π in the presence of the adversary Adv. For $R \subseteq [T]$, $x^A \in \mathcal{X}^A$, and $x^B \in \mathcal{X}^B$, we define the number of corruptions in the rounds in R to be

$$\mathsf{corr}_{\Pi,\mathsf{Adv},R}(x^A, x^B) = \sum_{i \in R} \mathbb{1}\left(\pi_i^A \neq \pi_i^B\right).$$

Recall that π^A, π^B are completely determined by x^A, x^B, Π, and Adv and therefore corr is well defined. We omit the subscript R when $R = [T]$.

Computing a function. Let $\mathcal{I} \subseteq \mathcal{X}^A \times \mathcal{X}^B$ and let $\gamma \in [0,1]$. Let $F : \mathcal{I} \to \mathcal{Y}$ be a (possibly partial) function. Let $\Pi = \left(\mathcal{X}^A, \mathcal{X}^B, \mathcal{Y}, T, p, f^A, f^B, \mathsf{out}\right)$ be a protocol. We say that Π *computes* F against a γ fraction of corruptions if for all adversaries Adv and all inputs $(x^A, x^B) \in \mathcal{I}$, it holds that $\mathsf{out}_{\Pi,\mathsf{Adv}}(x^A, x^B) = F(x^A, x^B)$ as long as $\mathsf{corr}_{\Pi,\mathsf{Adv}}(x^A, x^B) \leq \lceil \gamma T \rceil$. Similarly, we say that Π *computes* F against a γ fraction of corruptions *per phase* if for all adversaries Adv and all inputs $(x^A, x^B) \in \mathcal{I}$, it holds that $\mathsf{out}_{\Pi,\mathsf{Adv}}(x^A, x^B) = F(x^A, x^B)$ as long as $\mathsf{corr}_{\Pi,\mathsf{Adv},(i_{t-1},i_t]}(x^A, x^B) \leq \lceil \gamma (i_t - i_{t-1}) \rceil$ for all $t \in [P+1]$ (recall that rounds $(i_{t-1}, i_t]$ constitute Phase t).

3.3.1 Protocols with Equally Spaced Code

Let $n, k \in \mathbb{N}$ and $C \subseteq \{0,1\}^n$ be such that $|C| = k$. Let $\delta > 0$. We say that C is a δ-Equally Spaced Code if for all $c_1, c_2 \in C$ it holds that

$$\Delta(c_1, c_2) \leq \begin{cases} \left(\frac{k}{2(k-1)} + \delta\right)n, & \text{if } k \text{ is even} \\ \left(\frac{k+1}{2k} + \delta\right)n, & \text{if } k \text{ is odd} \end{cases}.$$

We mention that the Plotkin bound implies that for *any* code C the average distance, $\Delta(c_1, c_2)$, between two codewords $c_1, c_2 \in C$ satisfies the last inequality, even with $\delta = 0$. Thus, a code C is an Equally Spaced Code if all the distances are at most the best possible average distance of a binary code.

Let Π be a protocol of length T over the binary two-way communication channel. For every Phase $t \in [P+1]$ of Π, let $C_{t,\mathcal{I},\gamma}$ be the set of all possible messages sent in Phase t for inputs in $\mathcal{I}$ and adversaries with at most γ fraction of corruptions per phase. More formally, if Alice is the sender in Phase t, then $C_{t,\mathcal{I},\gamma}$ is $\pi^A_{(i_{t-1},i_t]}$ for all possible transcripts

π^A obtained by taking all possible input pairs $(x^A, x^B) \in \mathcal{I}$ and adversaries Adv with $\forall t' \in [t-1] :\ \mathsf{corr}_{\Pi,\mathsf{Adv},(i_{t'-1},i_{t'}]}(x^A, x^B) \le \lceil \gamma(i_{t'} - i_{t'-1}) \rceil$. (Recall that π^A is completely determined by x^A, x^B, Π, and Adv.) Similarly, for the case that Bob is the sender in Phase t.

Let $\delta > 0$. We say that Π uses $(\mathcal{I}, \gamma, \delta)$-Equally Spaced Code if for all $t \in [P+1]$ the set $C_{t,\mathcal{I},\gamma}$ is a δ-Equally Spaced Code.

3.3.2 The Message Transfer Function

We now define the message transfer function $\mathsf{MsgTrans}$ that is the focus of this paper. Let $n \in \mathbb{N}$. The *message transfer* function $\mathsf{MsgTrans}_n : \{0,1\}^n \times \{\bot\} \to \{0,1\}^n$ is given by $\mathsf{MsgTrans}_n(x, \bot) = x$ for all $x \in \{0,1\}^n$.

4 Location-Sensitive Codes

This section is devoted to the construction of our *location-sensitive code* C, formally stated in Theorem 4.1 below. This code encodes a pair of binary strings $(x_1, x_2) \in \{0,1\}^n \times \{0,1\}^n$ such that the distance between the encodings of two different pairs, $\Delta = \Delta\Big(C(x_{1,1}, x_{1,2}), C(x_{2,1}, x_{2,2})\Big)$, satisfies:

1. If the two pairs do not share an element, *i.e.*, $|\{x_{1,1}, x_{1,2}\} \cap \{x_{2,1}, x_{2,2}\}| = 0$, then $\Delta = \frac{1}{2}$.

2. If the two pairs share a single element and the element is in the same location, *i.e.*, $|\{x_{1,1}, x_{1,2}\} \cap \{x_{2,1}, x_{2,2}\}| = 1$ and $\exists j \in [2] :\ x_{1,j} = x_{2,j}$, then Δ is a constant smaller than $1/2$.

3. If the two pairs share a single element and the element is in different locations, *i.e.*, $|\{x_{1,1}, x_{1,2}\} \cap \{x_{2,1}, x_{2,2}\}| = 1$ and $\exists j \in [2] :\ x_{1,j} = x_{2,3-j}$, then Δ is a constant greater than $1/2$.

4. If the two pairs share both elements, *i.e.*, $|\{x_{1,1}, x_{1,2}\} \cap \{x_{2,1}, x_{2,2}\}| = 2$, then Δ is a constant strictly greater than 1 minus the constant in Item 2 (and, in particular, greater than $1/2$).

Theorem 4.1. *For all $\epsilon > 0$, there exists a constant K_5 such that for all $K' \ge K_5$ and $n > 0$, there exists a code $C : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^{K'n}$ such that the following holds for all $(x_{1,1}, x_{1,2}), (x_{2,1}, x_{2,2}) \in \{0,1\}^n \times \{0,1\}^n$ satisfying $x_{i,1} \ne x_{i,2}$ for all $i \in [2]$:*

$$\left(\frac{1}{2} + \eta - \epsilon\right) \cdot K'n \le \Delta\Big(C(x_{1,1}, x_{1,2}), C(x_{2,1}, x_{2,2})\Big) \le \left(\frac{1}{2} + \eta + \epsilon\right) \cdot K'n,$$

where:

$$\eta = -\frac{1}{128} \cdot \sum_{j,j' \in [2]} (-7)^{4-j-j'} \cdot \mathbb{1}(x_{1,j} = x_{2,j'}).$$

(Note that η may be positive.)

To prove Theorem 4.1, we will need some (fairly standard) results about random codes (given in Section 4.1), as well as new ideas (given in Sections 4.2 and 4.3).

4.1 Random Coding

Define the function $\mathsf{reg} : [4] \to \mathbb{R}$ as follows:

$$\mathsf{reg}(z) = \begin{cases} 0, & z = 1 \\ \frac{1}{2}, & z = 2 \\ \frac{3}{4}, & z = 3 \\ \frac{5}{4}, & z = 4 \end{cases}. \tag{3}$$

Roughly speaking, $\mathsf{reg}(z)$ for $z \in [4]$ captures the expected sum of the fractional Hamming distance of any z codewords to the bitwise majority of the z codewords when the code is chosen uniformly at random. Using the probabilistic method, we show a code for which these fractional Hamming distances are always attained.

Lemma 4.2. *For all $\epsilon > 0$, there exists a constant K_1 such that for all $K' \geq K_1$ and $n > 0$, there exists a code $C : \{0,1\}^n \to \{0,1\}^{K'n}$ such that the following holds for all $z \in [4]$, distinct $x_1, \ldots, x_z \in \{0,1\}^n$, and (not necessarily distinct) bits $b_1, \ldots, b_z \in \{0,1\}$:*

$$\left| \frac{1}{K'n} \cdot \sum_{j \in [K'n]} \mathbb{1}\Big(\forall i \in [z] : C_j(x_i) = b_i\Big) - \frac{1}{2^z} \right| \leq \epsilon. \tag{4}$$

Proof. Set $K_1 = \frac{10}{\epsilon^2}$. We show that C exists using the probabilistic method. For any input $x \in \{0,1\}^n$, we independently sample its encoding $C(x)$ uniformly at random from $\{0,1\}^{K'n}$. For all $z \in [4]$, distinct $x_1, \ldots, x_z \in \{0,1\}^n$, and bits $b_1, \ldots, b_z \in \{0,1\}$, we have:

$$\begin{aligned} \mathbb{E}\left[\sum_{j \in [K'n]} \mathbb{1}\Big(\forall i \in [z] : C_j(x_i) = b_i\Big) \right] &= \sum_{j \in [K'n]} \Pr[\forall i \in [z] : C_j(x_i) = b_i] \\ &= \sum_{j \in [K'n]} \prod_{i \in [z]} \Pr[C_j(x_i) = b_i] \\ &= \frac{K'n}{2^z}. \end{aligned}$$

By Chernoff bound (Lemma 3.1), the probability of Eq. (4) not satisfied by any fixed $z \in [4]$, $x_1, \ldots, x_z \in \{0,1\}^n$ and $b_1, \ldots, b_z \in \{0,1\}$ is upper bounded by $2 \cdot \exp(-2 \cdot (\epsilon K'n)^2 / (K'n)) \leq \exp(-10n)$. Moreover, by the union bound, there exists some $z \in [4]$, $x_1, \ldots, x_z \in \{0,1\}^n$

and $b_1, \ldots, b_z \in \{0,1\}$ violating Eq. (4) with probability at most $4 \cdot (2^n)^4 \cdot 2^4 \cdot \exp(-10n) < 1$. This implies the existence of a code C satisfying the lemma. □

Corollary 4.3. *For all $\epsilon > 0$, there exists a constant K_2 such that for all $K' \geq K_2$ and $n > 0$, there exists a code $C : \{0,1\}^n \to \{0,1\}^{K'n}$ such that the following holds for all $z \in [4]$ and distinct $x_1, \ldots, x_z \in \{0,1\}^n$:*

$$(\mathsf{reg}(z) - \epsilon) \cdot K'n \leq \sum_{j \in [K'n]} \min_{b \in \{0,1\}} \sum_{i \in [z]} \Delta\Big(C_j(x_i), b\Big) \leq (\mathsf{reg}(z) + \epsilon) \cdot K'n.$$

Proof. Let K_1 be the constant from Lemma 4.2 for $\epsilon' = \frac{\epsilon}{20}$. Set $K_2 = K_1$. For all $K' \geq K_2$, let $C : \{0,1\}^n \to \{0,1\}^{K'n}$ be the code guaranteed by Lemma 4.2 for K', n. We show the code C satisfies the corollary.

To see this, fix $z \in [4]$ and distinct $x_1, \ldots, x_z \in \{0,1\}^n$. For all bits $b_1, \ldots, b_z \in \{0,1\}$, define:

$$\delta(b_1, \ldots, b_z) := \min_{b \in \{0,1\}} \sum_{i \in [z]} \Delta(b_i, b) = \min\left(\sum_{i \in [z]} b_i, z - \sum_{i \in [z]} b_i\right).$$

At a high level, δ simply counts the number of occurrences of the bit that appears less often. Observe that:

$$\begin{aligned}\sum_{j \in [K'n]} \min_{b \in \{0,1\}} \sum_{i \in [z]} \Delta\Big(C_j(x_i), b\Big) &= \sum_{j \in [K'n]} \delta(C_j(x_1), \ldots, C_j(x_z)) \\ &= \sum_{b_1, \ldots, b_z \in \{0,1\}} \delta(b_1, \ldots, b_z) \cdot \sum_{j \in [K'n]} \mathbb{1}\Big(\forall i \in [z] : C_j(x_i) = b_i\Big).\end{aligned}$$

As a result, by Lemma 4.2, we further get:

$$\sum_{j \in [K'n]} \min_{b \in \{0,1\}} \sum_{i \in [z]} \Delta\Big(C_j(x_i), b\Big) \in \left[\frac{1}{2^z} - \epsilon', \frac{1}{2^z} + \epsilon'\right] \cdot K'n \cdot \sum_{b_1, \ldots, b_z \in \{0,1\}} \delta(b_1, \ldots, b_z).$$

This concludes the proof by observing that:

$$\mathsf{reg}(z) = \frac{1}{2^z} \cdot \sum_{b_1, \ldots, b_z \in \{0,1\}} \delta(b_1, \ldots, b_z).$$

□

4.2 The Code Merging Operation

Lemma 4.4. *For all $\epsilon > 0, k, m \in \mathbb{N}$, and a set $D \in \binom{\mathcal{D}([k])}{m}$, there exists a constant K_3 such that for all $K' \geq K_3$, $n > 0$, a set S with $|S| \leq 2^n$, and codes $C_j : S \to \{0,1\}^n$ for all $j \in [k]$, there exists a code $C : S^k \times D \to \{0,1\}^{K'n}$ such that the following holds for all* *distinct* $\Big((s_{i,j})_{j\in[k]}, d_i\Big)_{i\in[2]} \in \big(S^k \times D\big)^2$*:*

$$\nabla - \epsilon K'n \leq \Delta\Big(C\Big((s_{1,j})_{j\in[k]}, d_1\Big), C\Big((s_{2,j})_{j\in[k]}, d_2\Big)\Big) \leq \nabla + \epsilon K'n, \tag{5}$$

where:

$$\nabla = K' \cdot \sum_{j,j'\in[k]} d_1(j)d_2(j') \cdot \Delta\Big(C_j(s_{1,j}), C_{j'}(s_{2,j'})\Big).$$

Proof. Set $K_3 = \frac{2km}{\epsilon^2}$. We show that C exists using the probabilistic method. First, associate each element of $[K'n]$ with a unique pair $(\ell, \ell') \in [n] \times [K']$. This allows us to index each coordinate of C by a pair $(\ell, \ell') \in [n] \times [K']$. For any $\Big((s_j)_{j\in[k]}, d\Big) \in S^k \times D$ and $(\ell, \ell') \in [n] \times [K']$, we independently sample coordinate (ℓ, ℓ') of C as:

$$C_{(\ell,\ell')}\Big((s_j)_{j\in[k]}, d\Big) = \begin{cases} 1, & \text{with probability } \sum_{j\in[k]} d(j) \cdot \mathbb{1}\Big(C_{j,\ell}(s_j)\Big) \\ 0, & \text{otherwise .} \end{cases}.$$

The expected distance between the encodings for distinct $\Big((s_{i,j})_{j\in[k]}, d_i\Big)_{i\in[2]} \in \big(S^k \times D\big)^2$ is therefore:

$$\begin{aligned}
K' \cdot \sum_{\ell\in[n]} &\left(\sum_{j\in[k]} d_1(j) \cdot \mathbb{1}\Big(C_{j,\ell}(s_{1,j})\Big)\right) \cdot \left(1 - \sum_{j\in[k]} d_2(j) \cdot \mathbb{1}\Big(C_{j,\ell}(s_{2,j})\Big)\right) \\
&+ K' \cdot \sum_{\ell\in[n]} \left(1 - \sum_{j\in[k]} d_1(j) \cdot \mathbb{1}\Big(C_{j,\ell}(s_{1,j})\Big)\right) \cdot \left(\sum_{j\in[k]} d_2(j) \cdot \mathbb{1}\Big(C_{j,\ell}(s_{2,j})\Big)\right) \\
&= K' \cdot \sum_{\ell\in[n]} \sum_{j,j'\in[k]} d_1(j)d_2(j') \cdot \mathbb{1}\Big(C_{j,\ell}(s_{1,j})\Big) \cdot \Big(1 - \mathbb{1}\Big(C_{j',\ell}(s_{2,j'})\Big)\Big) \\
&\qquad + K' \cdot \sum_{\ell\in[n]} \sum_{j,j'\in[k]} d_1(j)d_2(j') \cdot \Big(1 - \mathbb{1}\Big(C_{j,\ell}(s_{1,j})\Big)\Big) \cdot \mathbb{1}\Big(C_{j',\ell}(s_{2,j'})\Big) \\
&= K' \cdot \sum_{\ell\in[n]} \sum_{j,j'\in[k]} d_1(j)d_2(j') \cdot \mathbb{1}\Big(C_{j,\ell}(s_{1,j}) \neq C_{j',\ell}(s_{2,j'})\Big) \\
&= K' \cdot \sum_{j,j'\in[k]} d_1(j)d_2(j') \cdot \Delta\Big(C_j(s_{1,j}), C_{j'}(s_{2,j'})\Big) \\
&= \nabla.
\end{aligned}$$

By Chernoff bound (Lemma 3.1), Eq. (5) is not satisfied by any fixed $\Big((s_{i,j})_{j\in[k]}, d_i\Big)_{i\in[2]} \in \big(S^k \times D\big)^2$ with probability at most $2 \cdot \exp\big(-2 \cdot (\epsilon K'n)^2/(K'n)\big) \leq \exp(-2kmn)$. Moreover, by the union bound, there exists some $\Big((s_{i,j})_{j\in[k]}, d_i\Big)_{i\in[2]} \in \big(S^k \times D\big)^2$ violating Eq. (5) with probability at most $\Big((2^n)^k \cdot m\Big)^2 \cdot \exp(-2kmn) < 1$. This implies the existence of a code C satisfying the lemma. □

4.3 Proof of Theorem 4.1

Lemma 4.5 below constructs an error correcting code that encodes a binary string and a bit $(x, \ell) \in \{0,1\}^n \times \{0,1\}$ such that the distance between the encodings of two different messages, $\Delta\Big(C(x_1, \ell_1), C(x_2, \ell_2)\Big)$, is $1/2$ if $x_1 \neq x_2$ and is 1 if $x_1 = x_2$. Note that in the latter case, we must have $\ell_1 \neq \ell_2$.

Lemma 4.5. *For all $\epsilon > 0$, there exists a constant K_4 such that for all $K' \geq K_4$ and $n > 0$, there exists a code $C : \{0,1\}^n \times \{0,1\} \to \{0,1\}^{K'n}$ such that the following holds for all $x_1, x_2 \in \{0,1\}^n$ and $\ell_1, \ell_2 \in \{0,1\}$:*

$$\left(\frac{1}{2} + \kappa - \epsilon\right) \cdot K'n \leq \Delta\Big(C(x_1, \ell_1), C(x_2, \ell_2)\Big) \leq \left(\frac{1}{2} + \kappa + \epsilon\right) \cdot K'n,$$

where:

$$\kappa = \frac{1}{2} \cdot (-1)^{\ell_1 + \ell_2 + 1} \cdot \mathbb{1}(x_1 = x_2).$$

(Note that κ may be negative.)

Proof. Let K_1 be the constant from Lemma 4.2 for $\epsilon' = \frac{\epsilon}{2}$. Set $K_4 = K_1$. For all $K' \geq K_4$, let $C' : \{0,1\}^n \to \{0,1\}^{K'n}$ be the code guaranteed by Lemma 4.2 for K', n. We show the following code C satisfies the lemma:

$$C(x, \ell) := \begin{cases} C'(x), & \ell = 0 \\ \overline{C'}(x), & \ell = 1 \end{cases}.$$

The case in which $x_1 = x_2$ is straightforward. If $\ell_1 = \ell_2$, we are encoding exactly the same input so $\Delta(C(x_1, \ell_1), C(x_2, \ell_2)) = 0$ and $\kappa = -\frac{1}{2}$. Otherwise, when $\ell_1 \neq \ell_2$, we have $\Delta(C(x_1, \ell_1), C(x_2, \ell_2)) = K'n$ and $\kappa = \frac{1}{2}$ as $\overline{C'}$ negates every bit of C'.

When $x_1 \neq x_2$, which implies $\kappa = 0$, we have the observation that for all $x_1 \neq x_2 \in \{0,1\}^n$ and $\ell_1, \ell_2 \in \{0,1\}$, it holds that:

$$\Delta\Big(C(x_1, \ell_1), C(x_2, \ell_2)\Big) = K'n - \Delta\Big(C(x_1, \ell_1), C(x_2, 1 - \ell_2)\Big).$$

Consequently, it suffices to consider the case where $\ell_1 = \ell_2 = 0$. To this end, Lemma 4.2 guarantees both:

$$\left(\frac{1}{4} - \epsilon'\right) \cdot K'n \leq \sum_{j \in [K'n]} \mathbb{1}\Big(C'_j(x_1) = 0 \wedge C'_j(x_2) = 1\Big) \leq \left(\frac{1}{4} + \epsilon'\right) \cdot K'n,$$

and:

$$\left(\frac{1}{4}-\epsilon'\right)\cdot K'n \le \sum_{j\in[K'n]} \mathbb{1}\Big(C'_j(x_1)=1\wedge C'_j(x_2)=0\Big) \le \left(\frac{1}{4}+\epsilon'\right)\cdot K'n.$$

Summing the above two inequalities gives us

$$\left(\frac{1}{2}-2\epsilon'\right)\cdot K'n \le \Delta\Big(C'(x_1),C'(x_2)\Big) \le \left(\frac{1}{2}+2\epsilon'\right)\cdot K'n.$$

This concludes the proof as $\epsilon = 2\epsilon'$ and $C(x,0)=C'(x)$ for all $x\in\{0,1\}^n$ by definition. □

We are now ready to prove the main result of this section, Theorem 4.1.

Proof of Theorem 4.1. Let K_4 be the constant from Lemma 4.5 for $\epsilon' = \frac{\epsilon}{2}$, and $C' : \{0,1\}^n \times \{0,1\} \to \{0,1\}^{K_4 n}$ the code guaranteed by Lemma 4.5 for K_4, n. Define $C_1(\cdot) := C'(\cdot,0)$ and $C_2(\cdot) := C'(\cdot,1)$. Let K_3 be the constant from Lemma 4.4 for $\epsilon', k=2, m=1$ and D being the singleton set containing the distribution μ over $[2]$ that gives a probability of $\frac{7}{8}$ to 1 and a probability of $\frac{1}{8}$ to 2. Set $K_5 = K_4K_3$. For all $K' \ge K_5$, let $C_0 : (\{0,1\}^n)^2 \times D \to \{0,1\}^{K'n}$ be the code guaranteed by Lemma 4.4 for $K'/K_4, K_4n$ and codes C_1, C_2. We show the following code C satisfies the lemma:

$$C(x_1,x_2) := C_0(x_1,x_2,\mu).$$

By Lemma 4.4, we get that for all $(x_{1,1},x_{1,2}),(x_{2,1},x_{2,2}) \in \{0,1\}^n\times\{0,1\}^n$ as in the lemma statement, we have:

$$\nabla - \epsilon' K'n \le \Delta\Big(C(x_{1,1},x_{1,2}),C(x_{2,1},x_{2,2})\Big) \le \nabla + \epsilon' K'n,$$

where, using Lemma 4.5, we have:

$$(\kappa^* - \epsilon')\cdot K'n \le \nabla \le (\kappa^*+\epsilon')\cdot K'n,$$

and:

$$\kappa^* = \sum_{j,j'\in[2]} \frac{7^{4-j-j'}}{64}\cdot\left(\frac{1}{2}+\frac{1}{2}\cdot(-1)^{j+j'+1}\cdot\mathbb{1}(x_{1,j}=x_{2,j'})\right).$$

To finish the proof, note that $\eta = \kappa^* - \frac{1}{2}$ and $\epsilon = 2\epsilon'$. □

5 Our Protocol

We formalize Theorem 1.1 as Theorem 5.1:

Theorem 5.1. *Define $\theta = 10^{-5}$. There exists a constant K such that for all $n\in\mathbb{N}$ there exists a two party protocol $\Pi = \Pi_n$ with length $T = Kn$ that computes the message transfer function $\mathsf{MsgTrans}_n$ against a $\left(\frac{1}{4}+\theta\right)$ fraction of corruptions.*

In this section we give the protocol that proves Theorem 5.1. We first give the notation and definitions used by the protocol (Sections 5.1 to 5.3). The protocol itself is in Algorithm 1 (Section 5.4). The analysis of Algorithm 1 (and proof of Theorem 5.1) is in Section 6.

For the rest of this section and the next, since Bob has only one possible input $x^B = \perp$ in a protocol Π for $\mathsf{MsgTrans}_n$, we omit Bob's input from our notation (*e.g.*, we write $\mathsf{out}_{\Pi,\mathsf{Adv}}(x)$ to mean $\mathsf{out}_{\Pi,\mathsf{Adv}}(x^A = x, x^B = \perp)$).

5.1 Stories

We heavily rely on the following definition of a *story.*

Definition 5.2. *Let $n, M \in \mathbb{N}$. We define an (n, M)-*story *to be a tuple $\mathcal{Z} = (x, U, V, w) \in \{0,1\}^n \times \binom{\{0,1\}^n}{2} \times \binom{\{0,1\}^n}{4} \times [M]$ such that $U \cap V = \emptyset$ and $x \notin U \cup V$. Denote by $\mathsf{Stories}_{n,M}$ the set of all (n, M)-stories.*

The function $\mathsf{story}(\cdot)$. Recall that our algorithm starts with Alice sending her input x using a list-decodable code. This allows Bob to compute a set S of size 3 that contains x. Bob then sends this set S back to Alice using another list-decodable code, which allows Alice to compute three sets $T_1, T_2, T_3 \in \binom{\{0,1\}^n}{3}$ such that all of them contain x and (if not too many errors were introduced), one of them is S. Additionally, Alice can compute the distance $w \in [M]$ of the message she receives to the closest correct codeword (see Line 4). Thus, before Phase 3, Alice can compute a tuple $\mathcal{T} = (T_1, T_2, T_3, w)$, which is guaranteed to be an element of the set Data defined next.

Definition 5.3. *For $x \in \{0,1\}^n$, we define the set $\mathsf{Data}(x)$ to be the set containing all tuples $\mathcal{T} = (T_1, T_2, T_3, w) \in \binom{\{0,1\}^n}{3}^3 \times [M]$ such that T_1, T_2, T_3 are distinct and $x \in \bigcap_{\ell \in [3]} T_\ell$. We also define the set $\mathsf{Data} = \bigcup_{x \in \{0,1\}^n} \mathsf{Data}(x)$.*

Next, we define a function $\mathsf{story}(\cdot)$ that Alice can use to take a tuple $\mathcal{T} \in \mathsf{Data}(x)$ and her input x to output a story $\mathsf{story}(x, \mathcal{T}) \in \mathsf{Stories}_{n,M}$ she can encode in Phase 3. More precisely, we define $\mathsf{story}(x, \mathcal{T}) = (x, U, V, w)$, where $U = T_1 \setminus \{x\}$ and V is the set $(T_2 \cup T_3) \setminus T_1$ padded using dummy elements to have size 4.

5.2 Location-Sensitive Codes for Stories

Our protocol uses a location-sensitive code that encodes stories, given in Lemma 5.4, and is based on the location-sensitive code constructed in Section 4. This code uses the following functions $\mathsf{d}_1, \mathsf{d}_2, \mathsf{d}_3 : [0,1] \to [0,1]$:

$$\mathsf{d}_1(z) = z, \quad \mathsf{d}_2(z) = \max\left(z, \frac{1}{2} - z\right), \quad \mathsf{d}_3(z) = \max\left(z, \frac{1}{2} - z, \frac{5}{12} - \frac{z}{3}\right).$$

Lemma 5.4. *For all $\epsilon > 0$ and $M \in \mathbb{N}$, there exists a constant K_6 such that for all $K' \geq K_6$ and $n > 0$, there exists a code $C : \mathsf{Stories}_{n,M} \to \{0,1\}^{K'n}$ such that the following holds for all $\mathcal{Z}_1 = (x_1, U_1, V_1, w_1), \mathcal{Z}_2 = (x_2, U_2, V_2, w_2) \in \mathsf{Stories}_{n,M}$ satisfying $x_1 \neq x_2$:*

$$\Delta\Big(C(\mathcal{Z}_1), C(\mathcal{Z}_2)\Big) \geq \left(0.5511 - \epsilon - 0.1 \cdot \sum_{i \in [2]} \mathsf{d}(x_{3-i}, U_i, V_i, w_i)\right) \cdot K'n,$$

where:

$$\mathsf{d}(y, U, V, w) = \begin{cases} \mathsf{d}_1\left(\frac{w}{M}\right), & \text{if } y \in U \\ \mathsf{d}_2\left(\frac{w}{M}\right), & \text{if } y \in V \\ \mathsf{d}_3\left(\frac{w}{M}\right), & \text{if } y \notin U \cup V \end{cases}.$$

We defer the proof of Lemma 5.4 to Section 5.5.

5.3 Protocol Definitions

Constants. We shall assume that there are at least 20 "dummy" strings (strings that cannot be inputs for Alice) in $\{0,1\}^n$. This is without loss of generality as one can simply increase n by 10 and have enough dummy strings. We define $\epsilon = \theta^{10}$ and $M = \frac{480}{\epsilon}$. Let K_2 be the constant from Corollary 4.3 for this value of ϵ. Similarly, let K_6 be the constant from Lemma 5.4 for this value of ϵ and M. Define $K = 90^{10} \cdot \max(K_2, K_6)$ and the parameters:

$$L_1 = \frac{401}{500} \cdot K, \qquad L_2 = \frac{9}{500} \cdot K, \qquad L_3 = \frac{90}{500} \cdot K. \tag{6}$$

Note that all these parameters are integers divisible by 3 and larger than $\max(K_2, K_6)$.

Error correcting codes. Fix $n > 0$ for the rest of this paper. The protocol Π that we define shall use several different types of codes, which we define next. Let ECC_1 and ECC_2 be the codes promised by Corollary 4.3 for L_1, n and $L_2/3, 3n$ respectively. Also, let ECC_3 be the code promised by Lemma 5.4 for L_3, n.

The function $\mathsf{corr\text{-}lb}^{(2)}(\cdot)$. As explained above, in Phase 3, Alice sends the encoding of a story to Bob. To compute its output, Bob decodes this message together with the message he received in Phase 1. During this decoding, he will also need to estimate (actually, lower bound) the number of corruptions in Phase 2.

This is done using a function $\mathsf{corr\text{-}lb}^{(2)}(\cdot)$, parameterized by the set $S \in \binom{\{0,1\}^n}{3}$ that Bob computed in Phase 1, and takes as input $\mathcal{T} \in \mathsf{Data}$, which is Bob's candidate for what Alice may have encoded in Phase 2. Formally,

$$\mathsf{corr\text{-}lb}^{(2)}_S(\mathcal{T}) = L_2 n \cdot \begin{cases} \mathsf{d}_1\left(\frac{w}{M}\right), & \text{if } S = T_1 \\ \mathsf{d}_2\left(\frac{w}{M}\right), & \text{else if } S = T_2 \text{ or } S = T_3 \\ \mathsf{d}_3\left(\frac{w}{M}\right), & \text{otherwise} \end{cases} . \tag{7}$$

As we shall show in Lemma 6.2, $\mathsf{corr\text{-}lb}^{(2)}_S(\mathcal{T})$ is indeed a lower bound on the number of corruptions in Phase 2 when S is the set Bob computed in Phase 1 and $\mathcal{T}$ is the data Alice computed in Phase 2. Note however that none of the parties can actually compute $\mathsf{corr\text{-}lb}^{(2)}_S(\mathcal{T})$ as they do not know both S and $\mathcal{T}$.

5.4 The Protocol

We are now ready to define the protocol Π in Algorithm 1. We note that ties in all $\arg\min$ are broken arbitrarily.

Algorithm 1 The $\mathsf{MsgTrans}$ protocol Π.

```
Input: Alice has input $x \in \{0,1\}^n$.
Output: Bob outputs $y \in \{0,1\}^n$.
  Phase 1:
    Alice sends $\mathsf{ECC}_1(x)$ bit by bit over $L_1 n$ rounds.
    Bob receives $\rho \in \{0,1\}^{L_1 n}$ and computes $S = \arg\min_{S' \in \binom{\{0,1\}^n}{3}} \sum_{x' \in S'} \Delta(\mathsf{ECC}_1(x'), \rho)$.
  Phase 2:
    Bob sends $\mathsf{ECC}_2(S)$ bit by bit over $L_2 n$ rounds.
    Alice receives $\sigma \in \{0,1\}^{L_2 n}$. She orders all sets $T \in \binom{\{0,1\}^n}{3}$ containing $x$ in increasing
    order of the value $\Delta(\mathsf{ECC}_2(T), \sigma)$, and denotes by $T_1, T_2, T_3$ the first three sets in this
    ordering, with $T_1$ being the first. Let $\mathcal{T} = (T_1, T_2, T_3, w)$ where $w = \left\lceil M \cdot \frac{\Delta(\mathsf{ECC}_2(T_1), \sigma)}{L_2 n} \right\rceil$.
  Phase 3:
    Alice sends $\mathsf{ECC}_3(\mathsf{story}(x, \mathcal{T}))$ bit by bit over $L_3 n$ rounds.
    Bob receives $\tau \in \{0,1\}^{L_3 n}$ and outputs $y = \arg\min_{x' \in S} \mathsf{corr\text{-}lb}(x')$, where
      $\mathsf{corr\text{-}lb}(x') = \Delta(\mathsf{ECC}_1(x'), \rho) + \min_{\mathcal{T}' \in \mathsf{Data}(x')} \Big(\mathsf{corr\text{-}lb}^{(2)}_S(\mathcal{T}') + \Delta\big(\mathsf{ECC}_3(\mathsf{story}(x', \mathcal{T}')), \tau\big)\Big)$.
```

5.5 Proof of Lemma 5.4

This section is devoted to proving Lemma 5.4.

Let K_5 be the constant from Theorem 4.1 for $\epsilon' = \frac{\epsilon}{2}$, and $C' : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^{K_5 n}$ the code guaranteed by Theorem 4.1 for K_5, n. Define $C_j = C'$ for all $j \in [6]$. For $f \in [0,1]$, define:

$$p(f) := \frac{3-2f}{6},$$

and:

$$q(f) := \frac{1-2p(f)}{4} = \frac{f}{6}.$$

For all $f \in [0,1]$, we have:

$$2p(f) + 4q(f) = 2 \cdot \frac{3-2f}{6} + 4 \cdot \frac{f}{6} = 1,$$

and:

$$p(f) \geq \frac{1}{6} \geq q(f).$$

Let μ_f be the distribution over $[6]$ that gives a probability of $p(f)$ to each of $1, 2$ and a probability of $q(f)$ to each of $3, 4, 5, 6$. Let K_3 be the constant from Lemma 4.4 for $\epsilon', m = M, k = 6$ and $D = \big\{\mu_{w/M} \mid w \in [M]\big\}$. Set $K_6 = K_5 K_3$. For all $K' \geq K_6$, let $C_0 : (\{0,1\}^n \times \{0,1\}^n)^6 \times D \to \{0,1\}^{K'n}$ be the code guarenteed by Lemma 4.4 for $K'/K_5, K_5 n$ and codes C_j for $j \in [6]$. We show the following code C satisfies the lemma:

$$C(\mathcal{Z}) := C_0\big((x,u_1),(x,u_2),(x,v_1),(x,v_2),(x,v_3),(x,v_4),\mu_{w/M}\big),$$

where u_1, u_2 and v_1, v_2, v_3, v_4 are the elements of U and V, respectively, in some fixed order for $\mathcal{Z} = (x, U, V, w) \in \mathsf{Stories}_{n,M}$.

Because of Lemma 4.4 and Theorem 4.1, it suffices to show that the following holds for all $\mathcal{Z}_1 = (x_1, U_1, V_1, w_1), \mathcal{Z}_2 = (x_2, U_2, V_2, w_2) \in \mathsf{Stories}_{n,M}$ as in the lemma statement:

$$\eta^* + 0.1 \cdot \sum_{i \in [2]} \mathsf{d}(x_{3-i}, U_i, V_i, w_i) \geq 0.5511, \tag{8}$$

where:

$$\begin{aligned}\eta^* = &\sum_{j\in[2]}\sum_{j'\in[2]} p(f_1)p(f_2)\cdot\left(\frac{1}{2}+\eta_{\left((x_1,u_{1,j}),(x_2,u_{2,j'})\right)}\right)\\ &+\sum_{j\in[4]}\sum_{j'\in[4]} q(f_1)q(f_2)\cdot\left(\frac{1}{2}+\eta_{\left((x_1,v_{1,j}),(x_2,v_{2,j'})\right)}\right)\\ &+\sum_{j\in[2]}\sum_{j'\in[4]} p(f_1)q(f_2)\cdot\left(\frac{1}{2}+\eta_{\left((x_1,u_{1,j}),(x_2,v_{2,j'})\right)}\right)\\ &+\sum_{j\in[4]}\sum_{j'\in[2]} q(f_1)p(f_2)\cdot\left(\frac{1}{2}+\eta_{\left((x_1,v_{1,j}),(x_2,u_{2,j'})\right)}\right),\end{aligned}$$

with $f_i = \frac{w_i}{M}$ for $i \in [2]$, and:

$$\eta_{((y_{1,1},y_{1,2}),(y_{2,1},y_{2,2}))} = \begin{cases} 0, & \text{if } \{y_{1,1}, y_{1,2}\} \cap \{y_{2,1}, y_{2,2}\} = \emptyset \\ -\frac{1}{128}, & \text{if } y_{1,1} \neq y_{2,1} \text{ and } y_{1,2} = y_{2,2} \\ \frac{7}{128}, & \text{if } y_{1,1} = y_{2,2} \text{ and } y_{1,2} \neq y_{2,1}, \text{ or } y_{1,2} = y_{2,1} \text{ and } y_{1,1} \neq y_{2,2} \\ \frac{7}{64}, & \text{if } y_{1,1} = y_{2,2} \text{ and } y_{1,2} = y_{2,1} \end{cases}.$$

Note that we have assumed $x_1 \neq x_2$, so η and $\eta_{((y_{1,1},y_{1,2}),(y_{2,1},y_{2,2}))}$ are always well-defined. For convenience, we will prove the following equivalent form of Eq. (8):

$$\eta^* - 0.5 + 0.1 \cdot \sum_{i \in [2]} \mathsf{d}(x_{3-i}, U_i, V_i, w_i) \geq 0.0511. \tag{9}$$

Depending on whether $x_{3-i} \in U_i$, $x_{3-i} \in V_i$, or $x_{3-i} \notin U_i \cup V_i$, for $i \in [2]$, there are $3^2 = 9$ cases in total. Taking symmetry into account, only 6 of them are different essentially. We conclude the proof by lower bounding LHS of Eq. (9) in all these cases.

1. If $x_1 \in U_2$ and $x_2 \in U_1$, assume without loss of generality that $x_1 = u_{2,1}$ and $x_2 = u_{1,1}$. For any $y \in U_2 \cup V_2$ that is not x_1, we have $\eta_{((x_1,u_{1,1}=x_2),(x_2,y))} = \frac{7}{128}$, so the gain in LHS of Eq. (8) is $\frac{7}{128} \cdot p(f_1)(1 - p(f_2))$. Similarly, for any $y \in U_1 \cup V_1$ that is not x_2, we have $\eta_{((x_1,y),(x_2,u_{2,1}=x_1))} = \frac{7}{128}$ and thus we gain $\frac{7}{128} \cdot (1 - p(f_1))p(f_2)$. We also have $\eta_{((x_1,u_{1,1}=x_2),(x_2,u_{2,1}=x_1))} = \frac{7}{64}$, implying a gain of $\frac{7}{64} \cdot p(f_1)p(f_2)$.

   On the other hand, recall that $p(f) \geq q(f)$ for all $f \in [0, 1]$. Then by the rearrangement inequality, the worst possible case for η^* occurs when $u_{1,2} = u_{2,2}$ and $v_{1,j} = v_{2,j}$ for all $j \in [4]$. The former results in a loss of $\frac{1}{128} \cdot p(f_1)p(f_2)$ since $\eta_{((x_1,u_{1,2}),(x_2,u_{2,2}))} = -\frac{1}{128}$. Similarly, each of the latter four results in a loss of $\frac{1}{128} \cdot q(f_1)q(f_2)$. Overall, we have:

$$\begin{aligned} \text{LHS} \geq\ & \frac{7}{128} \cdot p(f_1)(1 - p(f_2)) + \frac{7}{128} \cdot (1 - p(f_1))p(f_2) + \frac{7}{64} \cdot p(f_1)p(f_2) \\ & - \frac{1}{128} \cdot (p(f_1)p(f_2) + 4q(f_1)q(f_2)) \\ & + 0.1 \cdot (\mathsf{d}(x_2, U_1, V_1, w_1) + \mathsf{d}(x_1, U_2, V_2, w_2)) \\ =\ & \frac{7}{128} \cdot (p(f_1) + p(f_2)) - \frac{1}{128} \cdot (p(f_1)p(f_2) + 4q(f_1)q(f_2)) \\ & + 0.1 \cdot (\mathsf{d}_1(f_1) + \mathsf{d}_1(f_2)) \\ =\ & \frac{-8f_2f_1 - 78f_1 - 78f_2 + 243}{4608} + 0.1 \cdot (f_1 + f_2) \\ \geq\ & \frac{27}{512} > 0.0511. \qquad \text{(Minimized at } f_1 = 0, f_2 = 0\text{)} \end{aligned}$$

   We shall remark that in this case and all the following cases, we are always seeking the minimum of a *piecewise bilinear* function in f_1, f_2. Minima of such functions can be obtained by examining its value only at the corner points of each piece.

2. If $x_1 \in U_2$ and $x_2 \in V_1$, assume without loss of generality that $x_1 = u_{2,1}$ and $x_2 = v_{1,1}$. The gain is similar to the first case except that x_2 is now chosen with probability $q(f_1)$ instead of $p(f_1)$. Regarding the loss, the worst possible case is where $u_{1,1} = u_{2,2}$, $u_{1,2} = v_{2,1}$, and $v_{1,j+1} = v_{2,j+1}$ for all $j \in [3]$. Overall, we have:

$$\begin{aligned}
\text{LHS} &\geq \frac{7}{128} \cdot q(f_1)(1 - p(f_2)) + \frac{7}{128} \cdot (1 - q(f_1))p(f_2) + \frac{7}{64} \cdot q(f_1)p(f_2) \\
&\quad - \frac{1}{128} \cdot (p(f_1)p(f_2) + p(f_1)q(f_2) + 3q(f_1)q(f_2)) \\
&\quad + 0.1 \cdot (\mathsf{d}(x_2, U_1, V_1, w_1) + \mathsf{d}(x_1, U_2, V_2, w_2)) \\
&= \frac{7}{128} \cdot (q(f_1) + p(f_2)) - \frac{1}{128} \cdot (p(f_1)p(f_2) + p(f_1)q(f_2) + 3q(f_1)q(f_2)) \\
&\quad + 0.1 \cdot (\mathsf{d}_2(f_1) + \mathsf{d}_1(f_2)) \\
&= \frac{-5f_2f_1 + 48f_1 - 81f_2 + 117}{4608} + 0.1 \cdot \left(\max\left(f_1, \frac{1}{2} - f_1\right) + f_2\right) \\
&\geq \frac{407}{7680} > 0.0511. && \text{(Minimized at } f_1 = \tfrac{1}{4}, f_2 = 0\text{)}
\end{aligned}$$

3. If $x_1 \in U_2$ and $x_2 \notin U_1 \cup V_1$, assume without loss of generality that $x_1 = u_{2,1}$. Now the only gain is coming from x_1 while the worst possible case for the loss is still $u_{1,1} = u_{2,2}$, $u_{1,2} = v_{2,1}$, and $v_{1,j+1} = v_{2,j+1}$ for all $j \in [3]$. Overall, we have:

$$\begin{aligned}
\text{LHS} &\geq \frac{7}{128} \cdot p(f_2) - \frac{1}{128} \cdot (p(f_1)p(f_2) + p(f_1)q(f_2) + 3q(f_1)q(f_2)) \\
&\quad + 0.1 \cdot (\mathsf{d}(x_2, U_1, V_1, w_1) + \mathsf{d}(x_1, U_2, V_2, w_2)) \\
&\geq \frac{7}{128} \cdot p(f_2) - \frac{1}{128} \cdot (p(f_1)p(f_2) + p(f_1)q(f_2) + 3q(f_1)q(f_2)) \\
&\quad + 0.1 \cdot (\mathsf{d}_3(f_1) + \mathsf{d}_1(f_2)) \\
&= \frac{-5f_2f_1 + 6f_1 - 81f_2 + 117}{4608} + 0.1 \cdot \left(\max\left(f_1, \frac{1}{2} - f_1, \frac{5}{12} - \frac{f_1}{3}\right) + f_2\right) \\
&\geq \frac{701}{12288} > 0.0511. && \text{(Minimized at } f_1 = \tfrac{5}{16}, f_2 = 0\text{)}
\end{aligned}$$

4. If $x_1 \in V_2$ and $x_2 \in V_1$, assume without loss of generality that $x_1 = v_{2,1}$ and $x_2 = v_{1,1}$. In this case, the probabilities of choosing x_1 and x_2 become $q(f_2)$ and $q(f_1)$, respectively. The worst possible case for the loss becomes $u_{1,j} = u_{2,j}$ for all $j \in [2]$, and $v_{1,j'+1} = v_{2,j'+1}$ for all $j' \in [3]$. Overall, we have:

$$\begin{aligned}
\text{LHS} &\geq \frac{7}{128} \cdot q(f_1)(1 - q(f_2)) + \frac{7}{128} \cdot (1 - q(f_1))q(f_2) + \frac{7}{64} \cdot q(f_1)q(f_2) \\
&\quad - \frac{1}{128} \cdot (2p(f_1)p(f_2) + 3q(f_1)q(f_2)) \\
&\quad + 0.1 \cdot (\mathsf{d}(x_2, U_1, V_1, w_1) + \mathsf{d}(x_1, U_2, V_2, w_2))
\end{aligned}$$

$$
\begin{aligned}
&= \frac{7}{128}\cdot(q(f_1)+q(f_2)) - \frac{1}{128}\cdot(2p(f_1)p(f_2)+3q(f_1)q(f_2)) \\
&\qquad + 0.1\cdot(\mathsf{d}_2(f_1)+\mathsf{d}_2(f_2)) \\
&= \frac{-11f_2f_1+54f_1+54f_2-18}{4608} + 0.1\cdot\left(\max\left(f_1,\frac{1}{2}-f_1\right)+\max\left(f_2,\frac{1}{2}-f_2\right)\right) \\
&\geq \frac{19097}{368640} > 0.0511. \qquad\qquad \text{(Minimized at } f_1=\tfrac{1}{4}, f_2=\tfrac{1}{4}\text{)}
\end{aligned}
$$

5. If $x_1 \in V_2$ and $x_2 \notin U_1 \cup V_1$, assume without loss of generality that $x_1 = v_{2,1}$. As in the third case, now only x_1 is giving us some additional gain. For the loss, the worst possible case remains $u_{1,j} = u_{2,j}$ for all $j \in [2]$, and $v_{1,j'+1} = v_{2,j'+1}$ for all $j' \in [3]$. Overall, we have:

$$
\begin{aligned}
\text{LHS} &\geq \frac{7}{128}\cdot q(f_2) - \frac{1}{128}\cdot(2p(f_1)p(f_2)+3q(f_1)q(f_2)) \\
&\qquad + 0.1\cdot(\mathsf{d}(x_2,U_1,V_1,w_1)+\mathsf{d}(x_1,U_2,V_2,w_2)) \\
&\geq \frac{7}{128}\cdot q(f_2) - \frac{1}{128}\cdot(2p(f_1)p(f_2)+3q(f_1)q(f_2)) + 0.1\cdot(\mathsf{d}_3(f_1)+\mathsf{d}_2(f_2)) \\
&= \frac{-11f_2f_1+12f_1+54f_2-18}{4608} \\
&\qquad + 0.1\cdot\left(\max\left(f_1,\frac{1}{2}-f_1,\frac{5}{12}-\frac{f_1}{3}\right)+\max\left(f_2,\frac{1}{2}-f_2\right)\right) \\
&\geq \frac{82429}{1474560} > 0.0511. \qquad\qquad \text{(Minimized at } f_1=\tfrac{5}{16}, f_2=\tfrac{1}{4}\text{)}
\end{aligned}
$$

6. If $x_1 \notin U_2 \cup V_2$ and $x_2 \notin U_1 \cup V_1$, then $U_1 \cup V_1$ and $U_2 \cup V_2$ are completely disjoint. There is just no additional gain in this case. As to the loss, the worst possible case is when $u_{1,j} = u_{2,j}$ for all $j \in [2]$, and $v_{1,j'} = v_{2,j'}$ for all $j' \in [4]$. Overall, we have:

$$
\begin{aligned}
\text{LHS} &\geq 0 - \frac{1}{128}\cdot(2p(f_1)p(f_2)+4q(f_1)q(f_2)) \\
&\qquad + 0.1\cdot(\mathsf{d}(x_2,U_1,V_1,w_1)+\mathsf{d}(x_1,U_2,V_2,w_2)) \\
&\geq 0 - \frac{1}{128}\cdot(2p(f_1)p(f_2)+4q(f_1)q(f_2)) + 0.1\cdot(\mathsf{d}_3(f_1)+\mathsf{d}_3(f_2)) \\
&= \frac{-2f_2f_1+2f_1+2f_2-3}{768} \\
&\qquad + 0.1\cdot\left(\max\left(f_1,\frac{1}{2}-f_1,\frac{5}{12}-\frac{f_1}{3}\right)+\max\left(f_2,\frac{1}{2}-f_2,\frac{5}{12}-\frac{f_2}{3}\right)\right) \\
&\geq \frac{1965}{32768} > 0.0511. \qquad\qquad \text{(Minimized at } f_1=\tfrac{5}{16}, f_2=\tfrac{5}{16}\text{)}
\end{aligned}
$$

6 Analysis of Our Protocol

In this section we prove Theorem 5.1. We fix an input x for Alice and an adversary Adv for the protocol Π such that:

$$\mathsf{corr}_{\Pi,\mathsf{Adv}}(x) \le \left(\frac{1}{4}+\theta\right)\cdot Kn. \tag{10}$$

As we fix Adv and Π is as defined in Algorithm 1, we shall omit both Π and Adv from our notations, *e.g.*, we shall write $\mathsf{corr}(\cdot)$ instead of $\mathsf{corr}_{\Pi,\mathsf{Adv}}(\cdot)$. As our protocol is deterministic, fixing x and Adv fixes the values of all the variables in Algorithm 1. Henceforth, for a variable var in Algorithm 1, we shall use var to also denote this fixed value. In this notation, in order to show Theorem 5.1, we have to show that $y = x$. We do this in the remainder of this section.

We start by defining, for $i \in [3]$, the notation for the corruptions $\mathsf{corr}^{(i)}(x)$ that happened in Phase i. Recalling the definitions in Section 3.3, we have:

$$\begin{aligned}
\mathsf{corr}^{(1)}(x) &= \mathsf{corr}_{[L_1n]}(x),\\
\mathsf{corr}^{(2)}(x) &= \mathsf{corr}_{[(L_1+L_2)n]\setminus[L_1n]}(x),\\
\mathsf{corr}^{(3)}(x) &= \mathsf{corr}_{[Kn]\setminus[(L_1+L_2)n]}(x).
\end{aligned}$$

Lemma 6.1. $x \in S$.

Proof. Suppose for the purpose of contradiction that $x \notin S$. Let $x_1, x_2, x_3 \in \{0,1\}^n$ be the three elements of S. Corollary 4.3 easily shows that:

$$\Delta\Big(\mathsf{ECC}_1(x),\rho\Big) + \sum_{i\in[3]}\Delta\Big(\mathsf{ECC}_1(x_i),\rho\Big) \ge (\mathsf{reg}(4)-\epsilon)\cdot L_1n = \left(\frac{5}{4}-\epsilon\right)\cdot L_1n.$$

Moreover, by definition of S, we have:

$$\mathsf{corr}(x) \ge \Delta(\mathsf{ECC}_1(x),\rho) \ge \frac{1}{4}\cdot\left(\frac{5}{4}-\epsilon\right)\cdot L_1n = \left(\frac{5}{16}-\frac{\epsilon}{4}\right)\cdot L_1n,$$

which, under our setting of the parameters, contradicts the assumption in Eq. (10). □

Lemma 6.2. *We have:*

$$\mathsf{corr\text{-}lb}_S^{(2)}(\mathcal{T}) \le \mathsf{corr}^{(2)}(x) + \left(\epsilon + \frac{1}{M}\right)\cdot L_2n.$$

Proof. Let $\Delta_i = \Delta(\mathsf{ECC}_2(T_i),\sigma)$ for $i \in [3]$. Also note that $\mathsf{corr}^{(2)}(x) = \Delta(\mathsf{ECC}_2(S),\sigma)$. By definition of w, we have:

$$\frac{L_2n}{M}\cdot(w-1) \le \Delta_1 \le \frac{L_2n}{M}\cdot w. \tag{11}$$

Moreover, Algorithm 1 ensures $\Delta_2, \Delta_3 \geq \Delta_1$. If $S = T_1$, then it is easy to see that by Eq. (11), we have:

$$\begin{aligned}
\mathsf{corr}^{(2)}(x) &= \Delta_1 \\
&\geq \frac{L_2 n}{M} \cdot (w-1) \\
&= \left(\mathsf{d}_1\left(\frac{w}{M}\right) - \frac{1}{M}\right) \cdot L_2 n.
\end{aligned}$$

If $S \neq T_1$ but $S = T_i$ for some $i \in \{2, 3\}$, by Corollary 4.3, we have:

$$\Delta_1 + \Delta_i \geq \left(\mathsf{reg}(2) - \epsilon\right) \cdot L_2 n = \left(\frac{1}{2} - \epsilon\right) \cdot L_2 n.$$

Again using Eq. (11), we can get:

$$\begin{aligned}
\mathsf{corr}^{(2)}(x) &= \Delta_i \\
&\geq \max\left(\Delta_1, \left(\frac{1}{2} - \epsilon\right) \cdot L_2 n - \Delta_1\right) \\
&\geq \max\left(\frac{L_2 n}{M} \cdot (w-1), \left(\frac{1}{2} - \epsilon - \frac{w}{M}\right) \cdot L_2 n\right) \\
&\geq \left(\mathsf{d}_2\left(\frac{w}{M}\right) - \epsilon - \frac{1}{M}\right) \cdot L_2 n.
\end{aligned}$$

Similarly, if $S \neq T_i$ for all $i \in [3]$, Corollary 4.3 shows:

$$\Delta\left(\mathsf{ECC}_2(S), \sigma\right) + \Delta_1 \geq \left(\mathsf{reg}(2) - \epsilon\right) \cdot L_2 n = \left(\frac{1}{2} - \epsilon\right) \cdot L_2 n,$$

as well as:

$$\Delta\left(\mathsf{ECC}_2(S), \sigma\right) + \sum_{i \in [3]} \Delta_i \geq \left(\mathsf{reg}(4) - \epsilon\right) \cdot L_2 n = \left(\frac{5}{4} - \epsilon\right) \cdot L_2 n.$$

Also observe that $\Delta(\mathsf{ECC}_2(S), \sigma) \geq \Delta_i$ for all $i \in [3]$ in this case since $x \in T_i$ for all $i \in [3]$ and $x \in S$ by Lemma 6.1. By definition of $\mathcal{T}$, Eq. (11) then implies:

$$\begin{aligned}
\mathsf{corr}^{(2)}(x) &= \Delta(\mathsf{ECC}_2(S), \sigma) \\
&\geq \max\left(\Delta_1, \left(\frac{1}{2} - \epsilon\right) \cdot L_2 n - \Delta_1, \frac{1}{3} \cdot \left(\left(\frac{5}{4} - \epsilon\right) \cdot L_2 n - \Delta_1\right)\right) \\
&\geq \max\left(\frac{L_2 n}{M} \cdot (w-1), \left(\frac{1}{2} - \epsilon - \frac{w}{M}\right) \cdot L_2 n, \left(\frac{5}{12} - \frac{\epsilon}{3} - \frac{w}{3M}\right) \cdot L_2 n\right)
\end{aligned}$$

$$\geq \left(\mathsf{d}_3\left(\frac{w}{M}\right) - \epsilon - \frac{1}{M}\right) \cdot L_2 n.$$

This concludes the proof of the lemma by definition of $\mathsf{corr\text{-}lb}_S^{(2)}(\cdot)$. □

Lemma 6.3. *We have:*

$$\mathsf{corr\text{-}lb}(x) \leq \mathsf{corr}(x) + \left(\epsilon + \frac{1}{M}\right) \cdot L_2 n.$$

Proof. By definition of $\mathsf{corr\text{-}lb}(\cdot)$ and Lemma 6.2, we have:

$$\begin{aligned}
\mathsf{corr\text{-}lb}(x) &\leq \Delta\Big(\mathsf{ECC}_1(x), \rho\Big) + \mathsf{corr\text{-}lb}_S^{(2)}(\mathcal{T}) + \Delta\Big(\mathsf{ECC}_3(\mathsf{story}(x, \mathcal{T})), \tau\Big) \\
&\leq \mathsf{corr}^{(1)}(x) + \mathsf{corr}^{(2)}(x) + \left(\epsilon + \frac{1}{M}\right) \cdot L_2 n + \mathsf{corr}^{(3)}(x) \\
&= \mathsf{corr}(x) + \left(\epsilon + \frac{1}{M}\right) \cdot L_2 n.
\end{aligned}$$

□

Lemma 6.4. *For all $x_1 \neq x_2 \in S$, we have $\mathsf{corr\text{-}lb}(x_1) + \mathsf{corr\text{-}lb}(x_2) \geq 2 \cdot \left(\frac{1}{4} + \theta + \epsilon + \frac{1}{M}\right) \cdot Kn$.*

Proof. For any $\mathcal{T}_1 = (T_{1,1}, T_{1,2}, T_{1,3}, w_1) \in \mathsf{Data}(x_1)$ and $\mathcal{T}_2 = (T_{2,1}, T_{2,2}, T_{2,3}, w_2) \in \mathsf{Data}(x_2)$, let $\mathcal{Z}_i = \mathsf{story}(x_i, \mathcal{T}_i) = (x_i, U_i, V_i, w_i)$ for $i \in [2]$. Corollary 4.3 easily gives us the following for the first phase:

$$\Delta\Big(\mathsf{ECC}_1(x_1), \rho\Big) + \Delta\Big(\mathsf{ECC}_1(x_2), \rho\Big) \geq (\mathsf{reg}(2) - \epsilon) \cdot L_1 n = (0.5 - \epsilon) \cdot L_1 n. \tag{12}$$

Regarding the third phase, by Lemma 5.4, we have:

$$\begin{aligned}
&\Delta\Big(\mathsf{ECC}_3(\mathcal{Z}_1), \tau\Big) + \Delta\Big(\mathsf{ECC}_3(\mathcal{Z}_2), \tau\Big) \\
&\quad \geq \Delta\Big(\mathsf{ECC}_3(\mathcal{Z}_1), \mathsf{ECC}_3(\mathcal{Z}_2)\Big) \\
&\quad \geq \left(0.5511 - \epsilon - 0.1 \cdot \sum_{i \in [2]} \mathsf{d}(x_{3-i}, U_i, V_i, w_i)\right) \cdot L_3 n \\
&\quad = \left(0.501 - \sum_{i \in [2]} \mathsf{d}(x_{3-i}, U_i, V_i, w_i)\right) \cdot L_2 n + (0.501 - \epsilon) \cdot L_3 n.
\end{aligned} \tag{13}$$

To finish the proof, we claim that for all $i \in [2]$, it holds:

$$\mathsf{corr\text{-}lb}_S^{(2)}(\mathcal{T}_i) \geq \mathsf{d}(x_{3-i}, U_i, V_i, w_i) \cdot L_2 n. \tag{14}$$

This is because $S = T_{i,1}$ always implies $x_{3-i} \in U_i$ by definition and thus $\mathsf{d}(x_{3-i}, U_i, V_i, w_i) = \mathsf{d}_1\left(\frac{w_i}{M}\right)$. If $S \neq T_{i,1}$ but $S = T_{i,2}$ or $S = T_{i,3}$, then we know $s_{3-i} \in U_i \cup V_i$. Therefore, $\mathsf{d}(x_{3-i}, U_i, V_i, w_i) \leq \mathsf{d}_2\left(\frac{w_i}{M}\right)$ in this case. The third case is straightforward by definition of $\mathsf{d}(\cdot)$ and $\mathsf{corr\text{-}lb}^{(2)}(\cdot)$. Summing Eqs. (12) to (14), we finally get

$$\begin{aligned}
&\mathsf{corr\text{-}lb}(x_1) + \mathsf{corr\text{-}lb}(x_2) \\
&\quad \geq (0.5 - \epsilon) \cdot L_1 n + 0.501 \cdot L_2 n + (0.501 - \epsilon) \cdot L_3 n \\
&\quad > \frac{1}{2} Kn + \left(\frac{(L_2 + L_3)/K}{1000} - \epsilon \right) \cdot Kn \\
&\quad > \frac{1}{2} Kn + 2\left(\theta + \epsilon + \frac{1}{M} \right) \cdot Kn.
\end{aligned}$$

□

6.1 Proof of Theorem 5.1

We now finish the proof of Theorem 5.1 using Lemmas 6.3 and 6.4.

Proof of Theorem 5.1. As mentioned above, we need to argue that $y = x$. Suppose for the sake of contradiction that $y \neq x$. Then, by Line 6 of Algorithm 1 and Lemma 6.1, we get that x and y are two distinct elements of S and $\mathsf{corr\text{-}lb}(y) \leq \mathsf{corr\text{-}lb}(x)$. This means:

$$\begin{aligned}
2 \cdot \mathsf{corr\text{-}lb}(x) &\geq \mathsf{corr\text{-}lb}(y) + \mathsf{corr\text{-}lb}(x) \\
&\geq 2 \cdot \left(\frac{1}{4} + \theta + \epsilon + \frac{1}{M} \right) \cdot Kn && \text{(Lemma 6.4)} \\
&> 2 \cdot \mathsf{corr}(x) + 2 \cdot \left(\epsilon + \frac{1}{M} \right) \cdot L_2 n && \text{(Eq. (10))} \\
&\geq 2 \cdot \mathsf{corr\text{-}lb}(x), && \text{(Lemma 6.3)}
\end{aligned}$$

a contradiction. □

7 Impossibility Result for Equally Spaced Code

In this section we prove Theorem 7.1 below, which is the formal version of Theorem 1.2.

Theorem 7.1. *For any $\theta > 0$, there exists $n_0 > 0$ such that for every $n \geq n_0$, there does not exist a 3-phase protocol using $\left(\{0,1\}^n, \frac{1}{4} + \theta, \frac{\theta}{2}\right)$-Equally Spaced Code that computes $\mathsf{MsgTrans}_n$ against a $\left(\frac{1}{4} + \theta\right)$ fraction of corruptions.*

7.1 Proof of Theorem 7.1

The proof of Theorem 7.1 uses the following functions:

1. $F_{1,N}$: Let $\mathcal{I}_{1,N} = [N] \times \{\bot\}$. Define $F_{1,N} : \mathcal{I}_{1,N} \to [N]$ by $F_{1,N}(x, \bot) = x$.
2. $F_{2,N}$: Let $\mathcal{I}_{2,N} = \left\{ (x, B) \in [N] \times \binom{[N]}{2} \mid x \in B \right\}$. Define $F_{2,N} : \mathcal{I}_{2,N} \to [N]$ by $F_{2,N}(x, B) = x$.
3. $F_{3,N}$: Let $\mathcal{I}_{3,N} = \left\{ \big((x, A), B\big) \in \left([N] \times \binom{[N]}{2}\right) \times \binom{[N]}{2} \mid x \in B,\ x \notin A,\ B \subseteq A \cup \{x\} \right\}$. Define $F_{3,N} : \mathcal{I}_{3,N} \to [N]$ by $F_{3,N}\big((x, A), B\big) = x$.

For $i \in [3]$, we are interested in $(4 - i)$-phase protocols using Equally Spaced Code that computes $F_{i,N}$ against a $\left(\frac{1}{4} + \theta\right)$ fraction of corruptions per phase for $\theta > 0$. Without loss of generality, we assume protocols alternate between Alice and Bob across phases with Alice always sending in the last phase.

To prove Theorem 7.1, we first note that for $N = 2^n$, $F_{1,N}$ is $\mathsf{MsgTrans}_n$, thus we need to prove a lower bound against 3-phase protocols for $F_{1,N}$. To this end, we show that a 3-phase protocol using Equally Spaced Code that computes $F_{1,N}$ against a $\left(\frac{1}{4} + \theta\right)$ fraction of corruptions per phase for $\theta > 0$, implies a similar 2-phase protocol that computes $F_{2,N'}$, for some smaller N' (Lemma 7.2). We then show that a 2-phase protocol that computes $F_{2,N'}$ implies a 1-phase protocol that computes $F_{3,N''}$, for some even smaller N'' (Lemma 7.3). Finally, we give a lower bound against any 1-phase protocol that computes $F_{3,N''}$ (Lemma 7.4).

We mention that the (noiseless) communication complexity of $F_{2,N}$ was studied by [Orl90], who showed a tight bound of $\Theta(\log N)$. Furthermore, his upper bound protocol consists of only two phases. Observe that a 2-phase protocol for $F_{2,N}$ implies a 1-phase protocol for $F_{3,N}$. Lemma 7.4 gives an $\Omega(\log \log N)$ lower bound on the communication complexity of a 1-phase protocol for $F_{3,N}$. While our result in this paper does not require an upper bound for this problem, in Section 7.2 we include a matching $O(\log \log N)$ upper bound as we believe it may be of independent interest. We also remark that the 1-phase complexity of $F_{3,N}$ is closely related to the question of deterministic compression with uncertain priors studied in [HS16].

We will use the following definition: For $n > 0$ and $x, y \in \{0,1\}^n$ we define the string $\mathsf{mid}(x, y) \in \{0,1\}^n$. Informally, $\mathsf{mid}(x, y)$ is a string whose Hamming distance from x and from y is equal (up-to ± 1). Formally, if $i_1 < i_2 < \cdots < i_{\Delta(x,y)}$ are the elements of $\{i \in [n] \mid x_i \neq y_i\}$, then, for all $i \in [n]$, coordinate i of the string $\mathsf{mid}(x, y)$ is defined as:

$$\mathsf{mid}(x, y) = \begin{cases} x_i, & \text{if } \forall j \in [\Delta(x, y)] : i \neq i_j \\ x_i, & \text{if } \exists j \in \left[\left\lceil \frac{\Delta(x,y)}{2} \right\rceil\right] : i = i_j \\ y_i, & \text{if } \exists j \in [\Delta(x, y)] \setminus \left[\left\lceil \frac{\Delta(x,y)}{2} \right\rceil\right] : i = i_j \end{cases} . \tag{15}$$

7.1.1 Going from 3 Phases to 2

Lemma 7.2. *If there exists a 3-phase protocol Π using $\left(\mathcal{I}_{1,N}, \frac{1}{4} + \theta, \frac{\theta}{2}\right)$-Equally Spaced Code that computes $F_{1,N}$ against a $\left(\frac{1}{4} + \frac{\theta}{2}\right)$ fraction of corruptions per phase, where $N, \theta > 0$, then there also exists a 2-phase protocol Π' using $\left(\mathcal{I}_{2,\theta N}, \frac{1}{4} + \theta, \frac{\theta}{2}\right)$-Equally Spaced Code that computes $F_{2,\theta N}$ against a $\left(\frac{1}{4} + \frac{\theta}{2}\right)$ fraction of corruptions per phase.*

Proof. Let ECC be the code Alice uses in Phase 1 of Π, and C the set of codewords Alice uses in Phase 1 of Π across all possible inputs. In the following, we construct Π' with the desired property in two different cases based on $|C|$.

Case 1: $|C| \leq \frac{1}{\theta}$**.** By an averaging argument, there exists a codeword $c \in C$ such that Alice sends c in Phase 1 of Π when given any one of at least $\frac{N}{|C|}$ out of all N possible inputs. Without loss of generality, assume Alice sends c upon seeing any one of $[N']$ where $N' = \frac{N}{|C|} \geq \theta N$. Let Π' be the 2-phase protocol for $F_{2,N'}$ in which both parties simulate Π with Bob pretending that he receives c from Alice at the very beginning (so Bob is actually not using his input as is in Π). Clearly, Π' is using $\left(\mathcal{I}_{2,N'}, \frac{1}{4}+\theta, \frac{\theta}{2}\right)$-Equally Spaced Code by definition. We claim that Π' computes $F_{2,N'}$ against a $\left(\frac{1}{4}+\frac{\theta}{2}\right)$ fraction of corruptions per phase.

To see this, suppose for the purpose of contradiction that an adversary Adv' for Π' is able to make Bob output $y \neq x_1^A$ on some input $A' = x_1^A$ and $B' = \left\{x_1^B, x_2^B\right\}$, while corrupting at most a $\left(\frac{1}{4}+\frac{\theta}{2}\right)$ fraction of each phase. Consider the execution of Π on the input $A = x_1^A$ and $B = \perp$. Let Adv be the adversary for Π who makes no corruption in Phase 1 and simulates Adv' in all later phases. By our assumption, Alice sends c in Phase 1, which is received by Bob without corruptions. As a result, Bob behaves in exactly the same way in both Π and Π' (with an imaginary first-phase message), and thus outputs the same answer $y \neq x_1^A$, contradicting that Π computes $F_{1,N}$ against a $\left(\frac{1}{4}+\frac{\theta}{2}\right)$ fraction of corruptions per phase.

Case 2: $|C| > \frac{1}{\theta}$**.** Since Π is using $\left(\mathcal{I}_{1,N}, \frac{1}{4}+\theta, \frac{\theta}{2}\right)$-Equally Spaced Code, we know that any two codewords in C have a relative Hamming distance of at most $\frac{1}{2}+\frac{1}{2(|C|-1)}+\frac{\theta}{2} \leq \frac{1}{2}+\theta$ by definition. This implies a $\left(\frac{1}{4}+\frac{\theta}{2}\right)$ fraction of corruptions is sufficient to corrupt any codeword $c_1 \in C$ into $\mathsf{mid}(c_1, c_2)$ for any other codeword $c_2 \in C$. On input $A = x_1^A$ and $B = \left\{x_1^B, x_2^B\right\}$, the 2-phase protocol Π' for $F_{2,N}$ proceeds by both parties simulating Π with Bob pretending that he receives $\mathsf{mid}\left(\mathsf{ECC}\left(x_1^B\right), \mathsf{ECC}\left(x_2^B\right)\right)$ from Alice at the very beginning. It is not hard to see that Π' is using $\left(\mathcal{I}_{2,N}, \frac{1}{4}+\theta, \frac{\theta}{2}\right)$-Equally Spaced Code. We claim that Π' computes $F_{2,N}$ against a $\left(\frac{1}{4}+\frac{\theta}{2}\right)$ fraction of corruptions per phase.

Again, suppose for the purpose of contradiction that an adversary Adv' for Π' is able to make Bob output $y \neq x_1^A$ on some input $A' = x_1^A$ and $B' = \left\{x_1^B, x_2^B\right\}$, while corrupting at most a $\left(\frac{1}{4}+\frac{\theta}{2}\right)$ fraction of each phase. Consider the execution of Π on the input $A = x_1^A$ and $B = \perp$. Alice sends $\mathsf{ECC}(x_1^A)$ in Phase 1. Moreover, since $x_1^A \in \left\{x_1^B, x_2^B\right\}$, by the argument above, we can construct an adversary Adv who corrupts a $\left(\frac{1}{4}+\frac{\theta}{2}\right)$ fraction of Alice's first-phase message into $\mathsf{mid}\left(\mathsf{ECC}\left(x_1^B\right), \mathsf{ECC}\left(x_2^B\right)\right)$. Adv then simulates Adv' in all later phases. Similarly to the first case, it is not hard to see that Adv forces Bob to output the incorrect answer $y \neq x_1^A$ in Π, a contradiction. This concludes the proof. □

7.1.2 Going from 2 Phases to 1

Lemma 7.3. *If there exists a 2-phase protocol* Π *using* $\left(\mathcal{I}_{2,N}, \frac{1}{4}+\theta, \frac{\theta}{2}\right)$-Equally Spaced Code *that computes* $F_{2,N}$ *against a* $\left(\frac{1}{4}+\frac{\theta}{2}\right)$ *fraction of corruptions per phase, where* $N, \theta > 0$, *then there also exists a 1-phase protocol* Π' *using* $\left(\mathcal{I}_{3,\log^\theta N}, \frac{1}{4}+\theta, \frac{\theta}{2}\right)$-Equally Spaced Code *that computes* $F_{3,\log^\theta N}$ *against a* $\left(\frac{1}{4}+\frac{\theta}{2}\right)$ *fraction of corruptions (per phase).*

Proof. Let ECC be the code Bob uses in Phase 1 of Π, and C the set of codewords Bob uses in Phase 1 of Π across all possible inputs. In the following, we construct Π' with the desired property in two different cases based on $|C|$.

Case 1: $|C| \le \frac{1}{\theta}$**.** Imagine a complete graph G with the vertex set $[N]$. For each $\{x_1, x_2\} \in \binom{[N]}{2}$, the edge (x_1, x_2) of G is colored by the codeword $\mathsf{ECC}(\{x_1, x_2\})$, so there are a total of $|C|$ colors. By Lemma 3.2, there exists a monochromatic clique of size at least $\log^{\frac{1}{|C|}} N$. Without loss of generality, assume Bob sends the same codeword $c \in C$ upon seeing $B \in \binom{[N']}{2}$ where $N' = \log^{\frac{1}{|C|}} N \ge \log^\theta N$. Let Π' be the 1-phase protocol for $F_{3,N'}$ in which both parties simulate Π with Alice pretending that she receives c from Bob at the very beginning. Clearly, Π' is using $\left(\mathcal{I}_{3,N'}, \frac{1}{4}+\theta, \frac{\theta}{2}\right)$-Equally Spaced Code by definition. We claim that Π' computes $F_{3,N'}$ against a $\left(\frac{1}{4}+\frac{\theta}{2}\right)$ fraction of corruptions.

To see this, suppose for the purpose of contradiction that an adversary Adv' for Π' is able to make Bob output $y \ne x_1^A$ on some input $A' = \left(x_1^A, \left\{x_2^A, x_3^A\right\}\right)$ and $B' = \left\{x_1^B, x_2^B\right\}$, while corrupting at most a $\left(\frac{1}{4}+\frac{\theta}{2}\right)$ fraction. Consider the execution of Π on the input $A = x_1^A$ and $B = \left\{x_1^B, x_2^B\right\}$. Let Adv be the adversary for Π who makes no corruption in Phase 1 and simulates Adv' in all later phases. By our assumption, Bob sends c in Phase 1, which is received by Alice without corruptions. As a result, Alice behaves in exactly the same way in both Π and Π' (with an imaginary first-phase message), and so does Bob, who then outputs the same answer $y \ne x_1^A$, contradicting that Π computes $F_{2,N}$ against a $\left(\frac{1}{4}+\frac{\theta}{2}\right)$ fraction of corruptions per phase.

Case 2: $|C| > \frac{1}{\theta}$**.** Since Π is using $\left(\mathcal{I}_{2,N}, \frac{1}{4}+\theta, \frac{\theta}{2}\right)$-Equally Spaced Code, we know that any two codewords in C have a relative Hamming distance of at most $\frac{1}{2}+\frac{1}{2(|C|-1)}+\frac{\theta}{2} \le \frac{1}{2}+\theta$ by definition. This implies a $\left(\frac{1}{4}+\frac{\theta}{2}\right)$ fraction of corruptions is sufficient to corrupt any codeword $c_1 \in C$ into $\mathsf{mid}(c_1, c_2)$ for any other codeword $c_2 \in C$. On input $A = \left(x_1^A, \left\{x_2^A, x_3^A\right\}\right)$ and $B = \left\{x_1^B, x_2^B\right\}$, the 1-phase protocol Π' for $F_{3,N}$ proceeds by both parties simulating Π with Alice pretending that she receives $\mathsf{mid}\left(\mathsf{ECC}\left(\left\{x_1^A, x_2^A\right\}\right), \mathsf{ECC}\left(\left\{x_1^A, x_3^A\right\}\right)\right)$ from Bob at the very beginning. It is not hard to see that Π' is using $\left(\mathcal{I}_{3,N}, \frac{1}{4}+\theta, \frac{\theta}{2}\right)$-Equally Spaced Code. We claim that Π' computes $F_{3,N}$ against a $\left(\frac{1}{4}+\frac{\theta}{2}\right)$ fraction of corruptions.

Again, suppose for the purpose of contradiction that an adversary Adv' for Π' is able to make Bob output $y \ne x_1^A$ on some input $A' = \left(x_1^A, \left\{x_2^A, x_3^A\right\}\right)$ and $B' = \left\{x_1^B, x_2^B\right\}$, while corrupting at most a $\left(\frac{1}{4}+\frac{\theta}{2}\right)$ fraction. Consider the execution of Π on the input $A = x_1^A$ and $B = \left\{x_1^B, x_2^B\right\}$. Bob sends $\mathsf{ECC}(\left\{x_1^B, x_2^B\right\})$ in Phase 1. Moreover, it holds that either

$\{x_1^A, x_2^A\} = \{x_1^B, x_2^B\}$ or $\{x_1^A, x_3^A\} = \{x_1^B, x_2^B\}$ by definition. Then by the argument above, we can construct an adversary Adv who corrupts a $\left(\frac{1}{4} + \frac{\theta}{2}\right)$ fraction of Bob's first-phase message into $\mathsf{mid}\left(\mathsf{ECC}\left(\{x_1^A, x_2^A\}\right), \mathsf{ECC}\left(\{x_1^A, x_3^A\}\right)\right)$. Adv then simulates Adv' in all later phases. Similarly to the first case, it is not hard to see that Adv forces Bob to output the incorrect answer $y \neq x_1^A$ in Π, a contradiction. This concludes the proof. □

7.1.3 Lower Bound for 1 Phase

Lemma 7.4. *For any* 1*-phase protocol* Π *computing* $F_{3,N}$*, Alice uses at least* $\log\log N$ *distinct codewords across all possible inputs.*

Proof. Let C be the set of codewords Alice uses in Π across all possible inputs. For all possible input $B = \{x_1^B, x_2^B\}$ of Bob, Bob's output function (on uncorrupted codewords) is essentially a function $\mathsf{out}_B : C \to \{0, 1\}$, where 0 represents outputting $\min\{x_1^B, x_2^B\}$ while 1 represents outputting $\max\{x_1^B, x_2^B\}$. Note that there are a total of $2^{|C|}$ possible output functions. We claim that the sets of output functions $S_x = \{\mathsf{out}_{\{x,x'\}} \mid x < x' \leq N\}$ are distinct for all $x \in [N]$.

To see this, suppose for the purpose of contradiction that $S_{x_1} = S_{x_2}$ for $1 \leq x_1 < x_2 \leq N$. By definition, $\mathsf{out}_{\{x_1,x_2\}} \in S_{x_1}$ and thus $\mathsf{out}_{\{x_1,x_2\}} \in S_{x_2}$. So there exists $x_3 \in [N]$ such that $x_3 > x_2$ and $\mathsf{out}_{\{x_2,x_3\}} = \mathsf{out}_{\{x_1,x_2\}}$. Now consider the two cases where Alice always has input $A = (x_2, \{x_1, x_3\})$ while Bob has input either $B = \{x_1, x_2\}$ or $B' = \{x_2, x_3\}$. In both cases, Alice sends the same codeword $c \in C$. With no corruptions, observe that $\mathsf{out}_B(c) = \mathsf{out}_{B'}(c)$ as the two cases share the same output function. However, this implies Bob can output the correct answer x_2 only in exactly one of the two cases since $x_1 < x_2 < x_3$. This shows S_x are indeed distinct for all $x \in [N]$.

Counting the number of possible sets of output functions, we finally get $2^{2^{|C|}} \geq N$, or equivalently $|C| \geq \log\log N$, concluding the proof. □

Finally, we are ready to prove Theorem 7.1.

Proof of Theorem 7.1. We first claim that a 3-phase protocol that computes $\mathsf{MsgTrans}_n$ (even against a 0 fraction of corruptions) must have at least $\frac{\log n}{10}$ rounds. Suppose there exists a 3-phase protocol that computes $\mathsf{MsgTrans}_n$ with $T < \frac{\log n}{10}$ rounds, then there also exists a 1-phase protocol that computes $\mathsf{MsgTrans}_n$ with 4^T rounds. But a 1-phase protocol that computes $\mathsf{MsgTrans}_n$ (one message from Alice to Bob) must communicate at least n bits, a contradiction.

Fix $\theta > 0$ and let n_0 be an arbitrary integer such that $\log\log\left(\log^{\frac{\theta}{100}}\left(\frac{\theta}{100} \cdot n_0\right)\right) \geq \frac{100}{\theta}$. Let $n \geq n_0$. We next claim that any 3-phase protocol Π that computes $\mathsf{MsgTrans}_n$ against a $\left(\frac{1}{4} + \theta\right)$ fraction of corruptions, also computes $\mathsf{MsgTrans}_n$ against a $\left(\frac{1}{4} + \theta'\right)$ fraction of corruptions *per phase*, where $\theta' = \frac{\theta}{2}$. To see this, for $t \in [3]$, let L_t be the length of Phase t of Π, and let $T = L_1 + L_2 + L_3$. Let Adv be an adversary for Π that corrupts at most $\left\lceil\left(\frac{1}{4} + \theta'\right)L_t\right\rceil$ rounds in Phase t. The total number of rounds that are corrupted by Adv is at most $\left(\frac{1}{4} + \theta'\right)T + 3$, which is no more than $\left\lceil\left(\frac{1}{4} + \theta\right)T\right\rceil$ as $T \geq \frac{\log n}{10} \geq \frac{10}{\theta}$.

Let $N = 2^n$. Observe that $F_{1,N}$ is exactly $\mathsf{MsgTrans}_n$. Thus, it remains to show that there is no 3-phase protocol Π using $\left(\mathcal{I}_{1,N}, \frac{1}{4} + \theta, \theta'\right)$-Equally Spaced Code that computes $F_{1,N}$ against a $\left(\frac{1}{4} + \theta'\right)$ fraction of corruptions per phase.

We prove the last claim by contradiction. Assume that there exists a 3-phase protocol Π using $\left(\mathcal{I}_{1,N}, \frac{1}{4} + \theta, \theta'\right)$-Equally Spaced Code that computes $F_{1,N}$ against a $\left(\frac{1}{4} + \theta'\right)$ fraction of corruptions per phase. We can apply Lemmas 7.2 and 7.3 in sequence to get a 1-phase protocol Π' using $\left(\mathcal{I}_{3,N'}, \frac{1}{4} + \theta, \theta'\right)$-Equally Spaced Code that computes $F_{3,N'}$ against a $\left(\frac{1}{4} + \theta'\right)$ fraction of corruptions, where $N' = \log^{\theta'}(\theta' N)$. Furthermore, Lemma 7.4 shows Alice must use at least $N'' = \log\log N'$ distinct codewords across all possible inputs in Π'. However, this contradicts the assumption that Π' is using $\left(\mathcal{I}_{3,N'}, \frac{1}{4} + \theta, \theta'\right)$-Equally Spaced Code since $\frac{1}{2} + \frac{1}{2(N''-1)} + \theta'$ is less than $2 \cdot \left(\frac{1}{4} + \theta'\right)$ under the above assumptions. □

7.2 Tightness of Lemma 7.4

In this section, we present a simple 1-phase protocol computing $F_{3,N}$. It shows the lower bound of Lemma 7.4 is essentially tight up to constant factors.

Theorem 7.5. *There exists a 1-phase protocol computing $F_{3,N}$ in which Alice uses $O(\log\log N)$ distinct codewords across all possible inputs.*

The protocol proving Theorem 7.5 is presented in Algorithm 2. In Algorithm 2, we identify an integer $x \in [N]$ with a $(\log N)$-bit string corresponding to its binary representation, from its most significant bit to its least significant bit. Similarly, we also identify an integer $i \in [\log N]$ with a $(\log\log N)$-bit string corresponding to its binary representation, from its most significant bit to its least significant bit. The length of the string can always be inferred from context.

It is not hard to see Alice uses $O(\log\log N)$ distinct codewords across all possible inputs. Now, we first show that the protocol is well-defined as Eq. (16) covers all possible cases. In particular, if there exists no such index j (equivalently, $i_1 = i_2$), one of the first two cases of Eq. (16) must hold.

Claim 7.6. *If $i_1 = i_2$, then it holds that either $x_1^A = \min\left(x_1^A, x_2^A, x_3^A\right)$ or $x_1^A = \max\left(x_1^A, x_2^A, x_3^A\right)$.*

Proof. By definition, for all $i \in [1, i_1)$, we have $x_{1,i}^A = x_{2,i}^A = x_{3,i}^A$ since $i_1 = i_2$. Let $b = x_{1,i_1}^A \in \{0,1\}$. Then, we also have $x_{2,i_1}^A = x_{3,i_1}^A = 1 - b$. Since the binary representations are written from the most significant bits to the least significant bits, we can get that $x_2^A, x_3^A > x_1^A$ if $b = 0$ and $x_2^A, x_3^A < x_1^A$ if $b = 1$, concluding the proof. □

Then, we finish the proof of Theorem 7.5 by showing the correctness of Algorithm 2.

Proof of Theorem 7.5. When $j' = 0$, since $x_1^A \in \left\{x_1^B, x_2^B\right\} \subseteq \left\{x_1^A, x_2^A, x_3^A\right\}$, it is clearly that either $y = \min\left(x_1^B, x_2^B\right) = \min\left(x_1^A, x_2^A, x_3^A\right) = x_1^A$ or $y = \max\left(x_1^B, x_2^B\right) = \max\left(x_1^A, x_2^A, x_3^A\right) = x_1^A$. Now consider the case where $j' = j \in [\log N]$. Assume without loss of generality

Algorithm 2 The protocol computing $F_{3,N}$.

```
Input: Alice has input $A = \big(x_1^A, \{x_2^A, x_3^A\}\big) \in [N] \times \binom{[N]}{2}$, and Bob has input $B = \{x_1^B, x_2^B\}$.
Output: Bob outputs $y \in [N]$.
  Alice:
  Alice finds the smallest index $i_1 \in [\log N]$ such that $x_{1,i_1}^A \neq x_{2,i_1}^A$ and the smallest index $i_2 \in [\log N]$ such that $x_{1,i_2}^A \neq x_{3,i_2}^A$. Alice also tries to find an index $j \in [\log\log N]$ such that $i_{1,j} \neq i_{2,j}$.
  Alice sends a codeword to Bob as follows (the first satisfying case applies):
    $$\begin{cases} (0,0,0), & x_1^A = \min\big(x_1^A, x_2^A, x_3^A\big) \\ (0,1,1), & x_1^A = \max\big(x_1^A, x_2^A, x_3^A\big) \\ \Big(j, \mathbb{1}\big(x_1^A > x_2^A\big), \mathbb{1}\big(x_1^A > x_3^A\big)\Big), & i_{1,j} < i_{2,j} \\ \Big(j, \mathbb{1}\big(x_1^A > x_3^A\big), \mathbb{1}\big(x_1^A > x_2^A\big)\Big), & i_{1,j} > i_{2,j} \end{cases}. \qquad (16)$$
  Bob:
  Bob finds the smallest index $i' \in [\log N]$ such that $x_{1,i'}^B \neq x_{2,i'}^B$.
  Bob receives $(j', b_0, b_1)$ from Alice and outputs
    $$y = \begin{cases} \min\big(x_1^B, x_2^B\big), & (j', b_1, b_2) = (0,0,0) \text{ or } b_{i'_{j'}} = 0 \\ \max\big(x_1^B, x_2^B\big), & (j', b_1, b_2) = (0,1,1) \text{ or } b_{i'_{j'}} = 1 \end{cases}. \qquad (17)$$
```

that $i_{1,j} < i_{2,j}$, implying that $i_{1,j} = 0$ and $i_{2,j} = 1$. (The other case where $i_{1,j} > i_{2,j}$ is symmetric.) Observe that Bob must have $i' = i_1$ if $\left\{x_1^B, x_2^B\right\} = \left\{x_1^A, x_2^A\right\}$ and $i' = i_2$ if $\left\{x_1^B, x_2^B\right\} = \left\{x_1^A, x_3^A\right\}$. Moreover, Bob can distinguish between the cases by simply examining the value of $i'_{j'}$: $i'_{j'} = 0$ implies the former case while $i'_{j'} = 1$ implies the latter one. By the construction of Alice's message in Eq. (16), $b_{i'_{j'}}$ always conveys the critical information about the relative order of the correct answer x_1^A in the two-candidate set $\left\{x_1^B, x_2^B\right\}$. Therefore, the correctness of y follows as Bob already figures out which one of b_0, b_1 conveys the correct information. □

Acknowledgements

Klim Efremenko is supported by the Israel Science Foundation (ISF) through grant No. 1456/18 and European Research Council Grant number: 949707. Gillat Kol is supported by a National Science Foundation CAREER award CCF-1750443 and by a BSF grant No. 2018325.

References

[ADL06] Rudolf Ahlswede, Christian Deppe, and Vladimir S. Lebedev. Non-binary error correcting codes with noiseless feedback, localized errors, or both. In *International Symposium on Information Theory (ISIT)*, pages 2486–2487, 2006. 5

[BE17] Mark Braverman and Klim Efremenko. List and unique coding for interactive communication in the presence of adversarial noise. *SIAM J. Comput.*, 46(1):388–428, 2017. 4

[Ber64] Elwyn R. Berlekamp. *Block Coding with Noiseless Feedback*. PhD thesis, Massachusetts Institute of Technology (MIT), 1964. 4, 7

[Ber68] Elwyn R Berlekamp. Block coding for the binary symmetric channel with noiseless, delayless feedback. *Error-correcting codes*, pages 61–68, 1968. 3, 5, 7

[BGMO17] Mark Braverman, Ran Gelles, Jieming Mao, and Rafail Ostrovsky. Coding for interactive communication correcting insertions and deletions. *IEEE Transactions on Information Theory*, 63(10):6256–6270, 2017. 4

[BR11] Mark Braverman and Anup Rao. Towards coding for maximum errors in interactive communication. In *Symposium on Theory of computing (STOC)*, pages 159–166, 2011. 4

[EGH16] Klim Efremenko, Ran Gelles, and Bernhard Haeupler. Maximal noise in interactive communication over erasure channels and channels with feedback. *IEEE Trans. Inf. Theory*, 62(8):4575–4588, 2016. 3, 4

[EKS20a] Klim Efremenko, Gillat Kol, and Raghuvansh Saxena. Interactive error resilience beyond 2/7. In *Symposium on Theory of Computing (STOC)*. ACM, 2020. 4

[EKS20b] Klim Efremenko, Gillat Kol, and Raghuvansh R. Saxena. Binary interactive error resilience beyond 1/8. In *Foundations of Computer Science (FOCS)*, pages 470–481, 2020. i, 3, 4, 5, 6, 7, 8

[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, 2021. 4

[FGOS15] Matthew Franklin, Ran Gelles, Rafail Ostrovsky, and Leonard J. Schulman. Optimal coding for streaming authentication and interactive communication. *IEEE Transactions on Information Theory*, 61(1):133–145, 2015. 4

[Gel17] Ran Gelles. Coding for interactive communication: A survey. *Foundations and Trends® in Theoretical Computer Science*, 13(1–2):1–157, 2017. 4

[GH14] Mohsen Ghaffari and Bernhard Haeupler. Optimal Error Rates for Interactive Coding II: Efficiency and List Decoding. In *Foundations of Computer Science (FOCS)*, FOCS, pages 394–403, 2014. 4

[GH17] Ran Gelles and Bernhard Haeupler. Capacity of interactive communication over erasure channels and channels with feedback. *SIAM Journal on Computing*, 46(4):1449–1472, 2017. 4

[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. 4

[GKZ22] Meghal Gupta, Yael Tauman Kalai, and Rachel Yun Zhang. Interactive error correcting codes over binary erasure channels resilient to 1/2 adversarial corruption. In *Symposium on Theory of Computing (STOC)*, 2022. 1, 3, 5, 7

[GZ22a] Meghal Gupta and Rachel Yun Zhang. The optimal error resilience of interactive communication over binary channels. In *Symposium on Theory of Computing (STOC)*, 2022. 3, 8

[GZ22b] Meghal Gupta and Rachel Yun Zhang. Positive rate binary interactive error correcting codes resilient to ¿1/2 adversarial erasures. *CoRR*, abs/2201.11929, 2022. 3

[HKV15] Bernhard Haeupler, Pritish Kamath, and Ameya Velingker. Communication with partial noiseless feedback. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM)*, volume 40 of *LIPIcs*, pages 881–897, 2015. 1, 5

[HS16] Elad Haramaty and Madhu Sudan. Deterministic compression with uncertain priors. *Algorithmica*, 76(3):630–653, 2016. 2, 32

[HSV18] Bernhard Haeupler, Amirbehshad Shahrasbi, and Ellen Vitercik. Synchronization strings: Channel simulations and interactive coding for insertions and deletions. In *International Colloquium on Automata, Languages, and Programming (ICALP)*, volume 107, pages 75:1–75:14, 2018. 4

[Orl90] Alon Orlitsky. Worst-case interactive communication. i. two messages are almost optimal. *IEEE Transactions on Information Theory*, 36(5), 1990. 2, 32

[Pan13] Denis Pankratov. On the power of feedback in interactive channels. *Manuscript*, 2013. 4

[Plo60] M. Plotkin. Binary codes with specified minimum distance. *IRE Transactions on Information Theory*, 6(4):445–450, 1960. 1

[Sch92] Leonard J Schulman. Communication on noisy channels: A coding theorem for computation. In *Foundations of Computer Science (FOCS)*, pages 724–733, 1992. 4

[Sch93] Leonard J Schulman. Deterministic coding for interactive communication. In *Symposium on Theory of computing (STOC)*, pages 747–756, 1993. 4

[Sch96] Leonard J. Schulman. Coding for interactive communication. *IEEE Transactions on Information Theory*, 42(6):1745–1756, 1996. 4

[Sha48] Claude E. Shannon. A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, 5(1):3–55, 2001. Originally appeared in *Bell System Tech. J.* 27:379–423, 623–656, 1948. 1

[Sha56] Claude E. Shannon. The zero error capacity of a noisy channel. *IRE Transactions on Information Theory*, 2(3):8–19, 1956. 5

[SW17] Alexander A. Sherstov and Pei Wu. Optimal interactive coding for insertions, deletions, and substitutions. In *Foundations of Computer Science (FOCS)*, pages 240–251, 2017. 4

[SWS92] Joel Spencer, Peter Winkler, and South St. Three thresholds for a liar. *Combinatorics, Probability and Computing*, 1:81–93, 1992. 5

[WQC17] Gang Wang, Yanyuan Qin, and Chengjuan Chang. Communication with partial noisy feedback. In *IEEE Symposium on Computers and Communications (ISCC)*, pages 602–607, 2017. 5

[Zig76] K.Sh. Zigangirov. On the number of correctable errors for transmission over a binary symmetrical channel with feedback. *Problems of Information Transmission*, 12:85–97, 1976. 5