Electronic Colloquium on Computational Complexity, Report No. 51 (2021)

Binary Interactive Error Resilience Beyond $1/8$

(or why $(1/2)^3 > 1/8$)

Klim Efremenko* Ben-Gurion University

Gillat Kol† Princeton University

Raghuvansh R. Saxena‡ Princeton University

Abstract

Interactive error correcting codes are codes that encode a two party communication protocol to an error-resilient protocol that succeeds even if a constant fraction of the communicated symbols are adversarially corrupted, at the cost of increasing the communication by a constant factor. What is the largest fraction of corruptions that such codes can protect against?

If the error-resilient protocol is allowed to communicate large (constant sized) symbols, Braverman and Rao (STOC, 2011) show that the maximum rate of corruptions that can be tolerated is $1/4$. They also give a *binary* interactive error correcting protocol that only communicates bits and is resilient to $1/8$ fraction of errors, but leave the optimality of this scheme as an open problem.

We answer this question in the negative, breaking the $1/8$ barrier. Specifically, we give a binary interactive error correcting scheme that is resilient to $5/39 > 1/8$ fraction of adversarial errors. Our scheme builds upon a novel construction of binary *list-decodable* interactive codes with small list size.

*klimefrem@gmail.com. Supported by the Israel Science Foundation (ISF) through grant No. 1456/18.

†gillat.kol@gmail.com. Supported by an Alfred P. Sloan Fellowship, the National Science Foundation CAREER award CCF-1750443, and by the E. Lawrence Keyes Jr. / Emerson Electric Co. Award.

‡rrsaxena@princeton.edu. Supported by the National Science Foundation CAREER award CCF-1750443.

ISSN 1433-8092

1 Introduction

We study the error resilience of *binary* interactive coding schemes [Sch96]. A *binary* interactive coding scheme with error resilience θ solves the following problem: Let Π be a two party protocol over the noiseless binary channel. Construct a protocol Π' that can simulate Π even when a θ fraction of the bits sent during Π' are adversarially corrupted. What is the largest[1] $\theta > 0$ for which there exist binary interactive coding schemes with error resilience θ?

For a related problem where the protocols are allowed to communicate symbols from a large constant sized alphabet (rather than bits), Braverman and Rao [BR11], building on the groundbreaking work of [Sch96], construct a beautiful interactive coding scheme with error resilience $\frac{1}{4}$ and prove its optimality. They also show how to derive from it a binary interactive coding scheme that is resilient to $\frac{1}{8}$ fraction of errors. Since their work, it is open if there exists a binary interactive coding scheme with error resilience larger than $\frac{1}{8}$. In this work, we construct the first such scheme, answering this question from nearly a decade ago.

We note that constructing optimal binary codes is often a much more challenging task than constructing codes over large alphabets. This holds even in the non-interactive setting of error correcting codes where optimal rate *vs.* resilience trade-offs are known for codes with large alphabets (*e.g.*, the singleton bound). However, proving such trade-offs for the binary case has been a major open problem in coding theory for over half a century.

The $\frac{1}{8}$ error resilience barrier. Not only are we lacking simulations for general binary protocols with error resilience larger than $\frac{1}{8}$, but we also do not know any such simulation for the simplest interactive task of *message exchange*, where parties wish to exchange their inputs. This is despite the fact that, for message exchange, a binary protocol with error resilience $\frac{1}{8}$ is almost obvious. Namely, both the parties simply encode their inputs with a binary error correcting code of distance $\frac{1}{2}$, and exchange these encodings[2].

In Section 2, we explain why $\frac{1}{8}$ is a natural barrier for the error resilience of binary protocols, even when restricted to the message exchange task. We argue that $\frac{1}{8}$ comes from three separate $\frac{1}{2}$ factors, each stemming from a different property that we require. The first $\frac{1}{2}$ factor comes from the fact that we want a binary protocol (classical binary error correcting codes have distance approaching $\frac{1}{2}$, whereas codes over a large alphabet can have distance approaching 1 – the same factor $\frac{1}{2}$ separation is also present in the interactive regime). The second $\frac{1}{2}$ factor comes from the requirement of unique decoding (classical unique decoding can only be performed when the fraction of corruptions is less than $\frac{1}{2}$, whereas the analogous threshold for list decoding is 1 – the same factor $\frac{1}{2}$ separation is also present in the interactive regime). The last $\frac{1}{2}$ factor comes from the fact that the message exchange task, like all

[1] Actually, supremum.

[2] A distance of $\frac{1}{2}$ is the best one can get from binary error correcting codes with positive rate. This contrasts with the large alphabet case where we know error correcting codes with distance approaching 1. Correspondingly, for this case, the natural protocol for message exchange has error resilience $\frac{1}{4}$.

interactive tasks, is 'two-sided', and in order to fail the task, the adversary only needs to corrupt the 'weaker' one of the two parties (*i.e.*, the one who transmits less), which needs at most $\frac{1}{2}$ the corruptions.

We show, perhaps surprisingly, that while a protocol for message exchange with any two out of the three requirements above (*i.e.*, a non-binary unique decodable interactive scheme *or* a binary list-decodable interactive scheme *or* a binary unique decodable classical scheme) can have error resilience at most $(\frac{1}{2})^2 = \frac{1}{4}$, there is a protocol satisfying all three requirements with error resilience strictly larger than $(\frac{1}{2})^3 = \frac{1}{8}$. After a lot more effort, we are also able to extend our scheme from message exchange to all interactive tasks.

1.1 Our Result

In this work, we show a binary interactive coding scheme with error resilience strictly greater than $\frac{1}{8}$. The following is an informal statement of our main result, a formal statement is given in Theorem 6.1.

Theorem 1.1 (Informal)**.** *Let* Π *be a two-party binary communication protocol*[3]*. For every* $\theta < \frac{5}{39}$*, there exists a binary protocol* Π' *that simulates* Π *and is resilient to* θ*-fraction of adversarial errors. Moreover, the length of* Π' *is linear in the length of* Π*.*

Our result solves a long-standing open problem, stated explicitly in [BR11] (Open Problem 3), in [Gel17] (see Version 1.3, Remark 2.1), in [BE17] (Open Problem 2), and in [EGH16]. We also note that the best known upper bound on the error resilience of binary interactive coding schemes is $\frac{1}{6}$ [EGH16]. Pinning down the optimal constant inside the range $[\frac{5}{39}, \frac{1}{6}]$ is an extremely intriguing question.

Prior to our work, it was not even known if the $\frac{1}{8}$ barrier can be crossed with a protocol Π' of *arbitrary length*. When waiving the constraint on the encoding length, the problem of finding the maximal error resilience of interactive coding schemes reduces to finding the maximal error resilience of the message exchange problem, as any communication task can be accomplished if the parties exchange their entire (possibly huge) inputs.

Binary interactive list-decodable codes. A key ingredient in the construction of our coding scheme is a novel construction of a binary interactive *list-decodable* code with small list size. Our list-decodable scheme is resilient to $\frac{5}{32}$ fraction of errors and outputs a list of size (only) 3. We stress that our scheme in Theorem 1.1 crucially relies on the list being that small, and that even a list of size 5 would not have sufficed for our result (see Section 2). An imprecise statement of this result is given in Theorem 1.2, a formal statement can be found in Theorem 5.1.

[3] We assume, without loss of generality, that Π is deterministic, as every randomized protocol is a distribution over deterministic protocols.

Theorem 1.2 (Informal)**.** *Let Π be a two-party deterministic binary communication protocol and let $\theta < \frac{5}{32}$. Then, there exists a binary protocol Π' such that, upon its termination, on every pair of inputs x, y for the players in Π, the first party obtains a list $L^A_{x,y}$ and the second party obtains a list $L^B_{x,y}$ with $|L^A_{x,y}|, |L^B_{x,y}| \leq 3$. The lists have the property that whenever at most θ fraction of the communication is adversarially corrupted, $\Pi(x, y) \in L^A_{x,y} \cap L^B_{x,y}$, where $\Pi(x, y)$ is the (noiseless) transcript of the execution of Π on inputs x, y. Moreover, the length of Π' is linear in the length of Π.*

We note that (in an informal sense) the parameters obtained by our interactive list-decoding scheme correspond to the best possible parameters of a (non-interactive) list-decoding scheme: There exist (non-interactive) binary codes that are list decodable with lists of size 3 from up to a $\frac{5}{16}$-fraction of errors but no more [Bli86] (also see [ABP18]). An extra multiplicative factor of $\frac{1}{2}$ in the error resilience is incurred as our protocol is two-way and therefore both parties need to list-decode.

1.2 Related Work

Since the study of coding for interactive communication was initiated by Schulman [Sch92, Sch93, Sch96], numerous works have been published in this area [GMS11, BR11, Bra12, KR13, BE17, BKN14, GMS14, GHK+18, EGH16, BGMO17, *e.g.*]. Out of these works, the one most related to our paper is the [BR11] paper discussed above. For a great survey of this field, see [Gel17].

The maximum error resilience of interactive protocols. The question of the maximum error resilience of interactive codes, which parallels the question of maximal distance in the study of standard codes, has been one of the central topics studied by the interactive coding literature and was considered for various interactive models.

Our work, like [BR11] and most other works in interactive coding, assumes the "standard" (non-adaptive) model of interactive communication, where parties take turns communicating and cannot both send bits in the same round. There are models in the literature that give the parties more power and therefore have a higher resilience.

For example, the maximal error resilience question was also considered over the *adaptive* channel, where parties may collide (communicate in the same round) [GHS14, AGS16, EGH16, EKS20a, EKS20b]. Over large alphabets, the error resilience of the adaptive channel was shown to be strictly higher than $\frac{1}{4}$, which is the error resilience of the non-adaptive channel [GHS14, EKS20a, EKS20b]. It may be possible to obtain binary *adaptive* protocols with error resilience higher than $\frac{1}{8}$ by using the ideas in these works and losing a factor of $\frac{1}{2}$ from the result for large alphabets. However, our work is the first one to show that one does not necessarily have to lose this factor of $\frac{1}{2}$. Consequently, we suspect that our techniques in this paper, in combination with some of our tools in [EKS20a, EKS20b], can save us from losing a factor of $\frac{1}{2}$ in those models as well.

Error resilience better than $\frac{1}{8}$ for the binary case was proved for the channel with noiseless *feedback* as well. The work of [EGH16] gives an interactive coding scheme over the non-adaptive binary channel with feedback with error resilience $\frac{1}{6}$, and an interactive coding scheme over the adaptive binary channel with feedback with error resilience $\frac{1}{3}$. Both of these results are shown to be tight [EGH16]. The maximum interactive error resilience of the erasure channel and the insertions and deletions channel was studied in [EGH16, FGOS15, SW17, HSV18].

Interactive list-decoding. The notion of list-decodable codes was extended to the interactive setting by [BE17, GHS14, GH14]. In [BE17], it is shown that for every $\varepsilon > 0$, there exists an interactive list-decoding scheme over a large constant-sized alphabet with a list of size $\text{poly}(\frac{1}{\varepsilon})$ and error tolerance $\frac{1}{2} - \varepsilon$, that only blows-up the communication linearly[4]. Computationally efficient interactive list-decoding schemes with similar guarantees are constructed in [GHS14, GH14][5].

Open Problem 2 in [BE17] remarks that "one can modify all our protocols to work over a binary channel with a loss of a factor of two in the error rates that one can handle". This modification seems to require non-trivial effort and would imply binary interactive list-decodable codes with a list of size $\text{poly}(\frac{1}{\varepsilon})$ and error tolerance $\frac{1}{4} - \varepsilon$. While the error tolerance guarantee given by this modification is better than the error tolerance promised in Theorem 1.2 ($\frac{1}{4}$ *vs.* $\frac{5}{32}$), it does not suffice for our purposes as the list size (while still constant) is too big.

Shayevitz [Sha09] studied (standard) list-decodable codes in the presence of noiseless feedback, showing that feedback can in fact improve the parameters of list-decodable codes. The noiseless feedback model used in [Sha09] is incomparable to our interactive setting.

1.3 Techniques

Recall that the [BR11] protocol for the binary alphabet has error resilience $\frac{1}{8}$, and this value stems from the fact that the *optimal* decoding radius (error resilience) of binary error correcting codes is $\frac{1}{4}$. At a high level, our protocol gets higher error resilience by artificially implementing binary codes with error resilience better than the optimal $\frac{1}{4}$. We next give a very simplified description of our protocol.

At the beginning of the protocol, the parties use a binary interactive list-decodable code with super small (size 3) lists that tolerates $\frac{5}{16} > \frac{1}{4}$ fraction of errors, which we construct for this purpose (see Subsection 1.1). After this phase, both parties have small lists of candidates for the correct output, and Bob sends his list to Alice. Since Alice knows both her list and

[4]We note that the work of [BE17] actually gives much stronger statements: It shows that interactive list-decoding with a list of size $\text{poly}(\frac{1}{\varepsilon})$ is possible as long as $\alpha + \beta < 1 - \varepsilon$, where α and β are the fractions of the communication from Alice to Bob and from Bob to Alice (respectively) corrupted by the adversary.

[5]The list decoding scheme in [GH14] is of constant rate, improving over the polynomially-small rate obtained by [GHS14].

Bob's list, she can output correctly. She then sends the index of the correct output in Bob's list to Bob. The reason Alice sends the index instead of sending the actual output itself is that the index can only take one of 3 values. While no (asymptotic) binary error correcting code has distance greater than $\frac{1}{2}$, binary codes with up to 4 codewords can have distance $\frac{2}{3}$. We use such a code for the index and exploit its larger distance to get higher error resilience overall.

To conclude, the saving in our protocol stems from a novel combination of two binary error correcting codes with "better-than-optimal" error resilience, namely, a binary list-decodable code with super short list size and a binary error correcting code with few codewords. A more detailed overview, highlighting some of the challenges that the implementation of this high-level idea entails, is found in Section 2. We believe that this high-level idea can guide the design of additional interactive codes.

2 Overview of Our Protocol

In this section, we gradually build various aspects of our simulation scheme, highlighting the roles they play.

2.1 [BR11] And The Message Exchange Problem

The work most directly related to our work is by Braverman and Rao [BR11]. In this work, the authors show that, for any $\theta < \frac{1}{4}$, any noiseless two party communication protocol can be simulated over a channel that can adversarially corrupt any θ fraction of the symbols communicated over the channel. Moreover, the authors also show that the parameter $\frac{1}{4}$ is the largest possible for which such a claim holds, as long as the channel is non-adaptive, thereby showing that the *maximal error resilience* of the non-adaptive channel is $\frac{1}{4}$.

[BR11] provides a simulation for arbitrary noiseless protocols over the noisy two party channel. Nonetheless, it is worthwhile to consider this simulation for the basic *message exchange* protocol, where both parties have a private input that they want to share with each other. Considering only this simple task already provides a simulation scheme for *all* tasks with an exponential blowup, as any communication task can be performed by the parties simply exchanging their (potentially exponentially large) inputs. Thereafter, comes the harder task of reducing the length of the simulation and making it linear in the length of the original protocol.

We incorporate this high level blueprint into our sketch and first illustrate the subset of ideas needed even if one only wants to simulate the message exchange task in Subsection 2.2 and Subsection 2.3, and then build on these ideas to get a simulation scheme for all noiseless protocols (with a constant blowup) in Subsection 2.4.

2.2 The $\frac{1}{8}$ Barrier For Message Exchange

In the binary regime, the best known error resilience of an interactive coding scheme is $\frac{1}{8}$ [BR11]. This even holds for the restricted task of message exchange. The value $\frac{1}{8}$ seems to be a natural barrier due to the following three reasons, each of which contributes a multiplicative factor of $\frac{1}{2}$.

- **Binary alphabet:** It is well known that the maximum distance of binary error correcting codes is $\frac{1}{2}$, whereas the maximum distance of *general* error correcting codes (*i.e.*, with a constant sized alphabet) can be arbitrarily close to 1. This factor of $\frac{1}{2}$ separation is also found in the best known distance parameters for binary and general *tree codes* (see for example the construction in [Sch96]).

  As error correcting codes and tree codes are commonly used as 'building blocks' in interactive coding schemes, a factor of $\frac{1}{2}$ separation between binary and general coding schemes is found in many works. Indeed, [BR11] is one such work.

- **Unique decoding:** Another factor of $\frac{1}{2}$ in the error-resilience comes from the fact that if the minimum distance between two codewords of an error correcting code is δ, then the parties can decode to a unique codeword only if the number of errors is at most $\frac{\delta}{2}$. Going beyond this threshold requires working in the list-decoding regime where the parties output a small list of values that is guaranteed to contain the correct codeword. In the list decoding regime, the fraction of errors that one can decode from is double that in the unique decoding regime, and can be made arbitrarily close to 1 (using a constant sized alphabet).

- **Two-way task:** The last factor of $\frac{1}{2}$ is due to the fact that the message exchange task is successful only if *both* the parties output each others' input. In other words, in order to derail the message exchange task, the adversary only needs to make sure that one of the parties outputs incorrectly. In particular, the adversary can invest all his errors on the party that speaks less. Since this party talks in at most half the rounds, the fraction of corruptions required to derail the protocol is lower by a factor of $\frac{1}{2}$.

The fact that we have three requirements, each of which hurt the error resilience by a factor of $\frac{1}{2}$, is not strong reason to believe that the three requirements together imply an error resilience of at most $\frac{1}{8}$. Indeed, it is possible that there are ways to combine these requirements so that the resulting error resilience is higher. The reason $\frac{1}{8}$ is a natural barrier for the error resilience is that combining any two of the above requirements (and not the third) implies an error resilience of at most $\frac{1}{4}$. Indeed, we have the following bounds:

- **No binary alphabet:** Suppose that one relaxes the binary alphabet constraint but still requires unique decoding and two-way message exchange. As mentioned above, [BR11] show that the error resilience of their simulation is optimal in this case and no protocol can get error resilience higher than $\frac{1}{4}$.

- **No unique decoding:** Suppose now that one relaxes the unique decoding constraint (and allows each party to output a small set of guesses for the input of the other party, rather than a single value), but still requires binary alphabet and two-way message exchange. For this setting, [BE17] give a protocol that has error resilience $\frac{1}{4}$[6]. It seems to be folklore that one cannot have a protocol with higher error resilience, but as we could not find a proof, we provide an informal proof of this fact in Appendix A.

- **No two-way task:** Finally, consider the setting where only one party is required to send its input to the other party using a binary alphabet and the other party needs to uniquely decode this input. This is the familiar setting of binary error-correcting codes. It is well-known that in this setting, the maximum error resilience is $\frac{1}{4}$ and is achieved, for instance, by random codes.

Having shown the significance of the $\frac{1}{8}$ barrier, we next describe our protocol and illustrate how it surpasses this barrier.

2.3 Our Message Exchange Protocol

Our protocol for the message exchange problem will have $39N$ rounds and error resilience $\frac{5}{39} > \frac{1}{8}$ (for an N linear in the size of the parties' inputs). This means that the adversary can corrupt less than $\frac{5}{39} \cdot 39N = 5N$ rounds.

Our protocol consists of three phases of different lengths. The first phase and the third phase will have lengths $16N$ and $3N$ respectively and in these phases, Alice will be sending messages to Bob. The second phase will have length $20N$ where Bob will be sending messages to Alice. The protocol is given in Algorithm 1, illustrated in Figure 1, and described below.

The protocol follows the following high level scheme (see Figure 1): In phase 1, Alice will send her input x^A to Bob encoded using a *list-decodable* error correcting code. We choose an error correcting code that can decode from up to a $\frac{5}{16}$ fraction of corruptions using a list of size 3. As the length of this phase is $16N$ and the number of corruptions is less than $5N$, using such a code allows Bob to compute a set L of values, $|L| \leq 3$, that is guaranteed to have Alice's input.

In phase 2, Bob will send his input x^B to Alice along with the set L that he computed in phase 1, encoded using a *uniquely decodable* error correcting code[7]. As the length of this phase is $20N$ and the number of corruptions is less than $5N = \frac{1}{4} \cdot 20N$, there exist codes that allow Alice to decode x^B and L uniquely from Bob's message. Thus, after phase 2, Alice knows both her output and the set L that Bob computed in phase 1.

In the third phase, Alice will help Bob identify x^A inside the set L. As Alice knows both x^A and L, after phase 2, she can do this by simply sending the index of x^A inside L in this

[6]More precisely, they give a protocol with error resilience $\frac{1}{2}$ with a constant sized alphabet but remark that their ideas can be extended to the binary regime at the cost of another $\frac{1}{2}$ in the error resilience.

[7]Note that this is the only phase where Bob speaks in our protocol, and therefore Alice must unique decode in this phase.

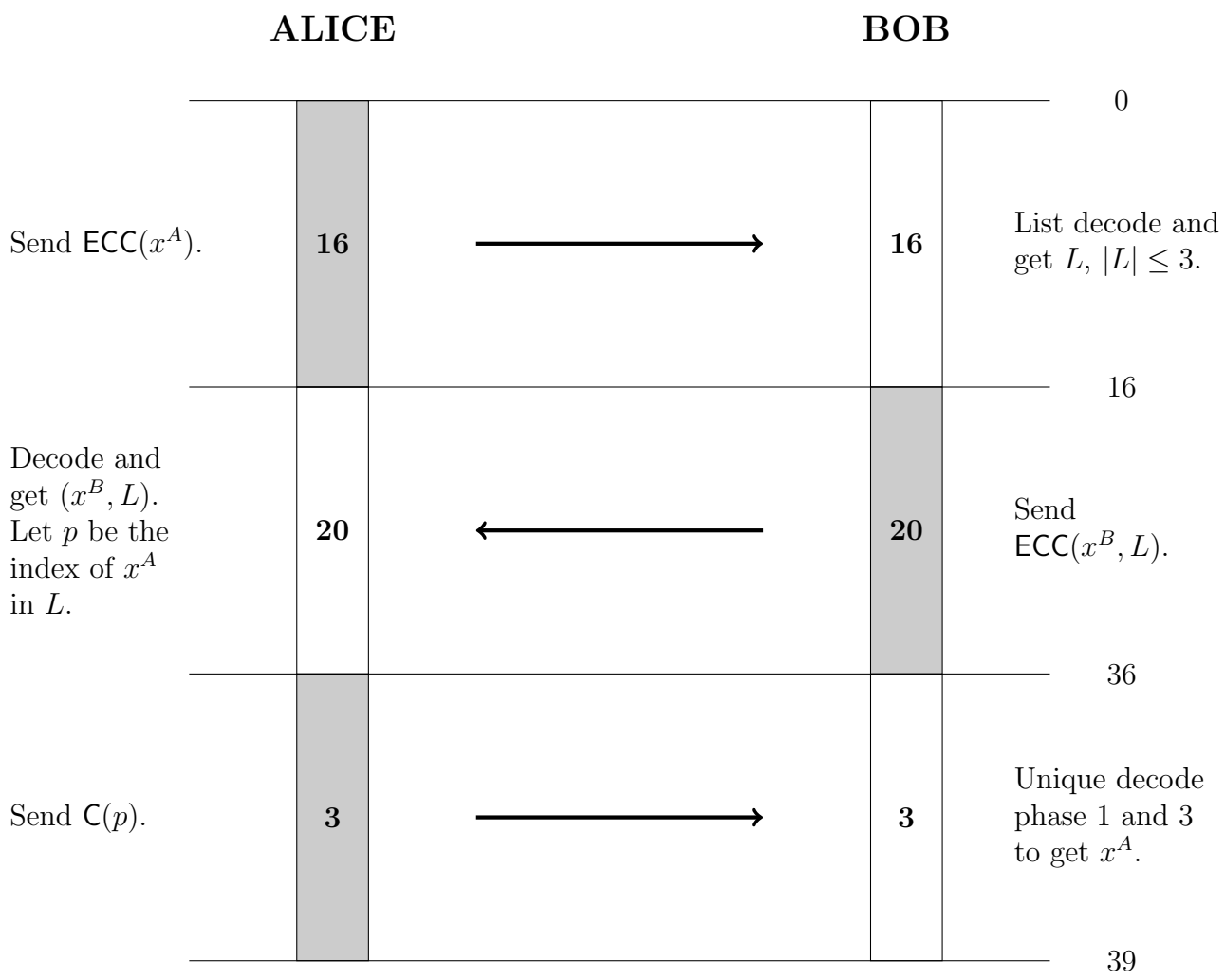

Figure 1: The 3 stages in our $\frac{5}{39}$ error resilient protocol (lengths not to scale). Here, ECC is a binary error correcting code with distance $\frac{1}{2}$ and list decodable from up to $\frac{5}{16}$-fraction of errors with a list of size 3 and C is a binary error correcting code with only 3 codewords and distance $\frac{2}{3}$. It is well known that such codes exist. In fact, codes with 4 codewords and distance $\frac{2}{3}$ also exist (see Lemma 4.2).

phase (of course, encoded using an error correcting code). However, binary error correcting codes have distance of at most $\frac{1}{2}$, implying that the distance obtained by Alice in phase 1 and phase 3 add up to $\frac{1}{2} \cdot (16N + 3N) = 9.5N$. As $9.5N < 2 \cdot 5N$, an adversary that can corrupt less than $5N$ messages can make Bob receive the same transcript for two different inputs for Alice.

Thus, it seems that in order for Bob to decode Alice's input, Alice somehow needs to encode her message in phase 3 using a binary code of distance $> \frac{1}{2}$[8]. However, as stated in Subsection 2.2, binary error correcting codes have distance at most $\frac{1}{2}$. How is it possible to attain the unattainable?

Attaining the unattainable. It turns out that there is one setting where binary error correcting codes can get distance strictly larger than $\frac{1}{2}$, and this is the setting of error correcting codes with few codewords (as opposed to asymptotic codes that are being used regularly). As an extreme example, consider the case where one wishes to have an error correcting code with only two codewords. In this case, it is actually possible to get a code with distance 1! Indeed, simply consider the code with the codewords $00 \cdots 0$ and $11 \cdots 1$.

In fact, binary codes with distance exceeding $\frac{1}{2}$ are known to exist for any constant number of codewords, with the distance approaching $\frac{1}{2}$ as the number of codewords increases. As all we need in phase 3 is a binary code with 3 codewords (recall that $|L| \leq 3$ and Alice only wants to encode an index $\leq |L|$), codes with few codewords are the way to go! We next employ the fact that there exists a code with 4 codewords that has distance $\frac{2}{3}$ (see Lemma 4.2 for an explicit construction) to get that the combined distance obtained by Alice in phase 1 and phase 3 is $\frac{1}{2} \cdot 16N + \frac{2}{3} \cdot 3N = 10N$. As the budget of the adversary is less than $5N$, this ensures that Bob can always decode Alice's input from the bits received by him.

Below, we present a slightly more formal description of our protocol and analysis. Before the description however, we briefly state why our protocol does not run into the barriers described in Subsection 2.2.

The first barrier described in Subsection 2.2 was that binary error correcting codes can only offer a distance of $\frac{1}{2}$. We get around this barrier by using codes with few codewords that offer better distance guarantees. We also note that, for larger alphabets, codes with few codewords offer the same distance guarantees as asymptotic codes (both have distance close to 1) and therefore, a similar trick cannot be used to improve the error resilience of, say, [BR11].

The second barrier in Subsection 2.2 was that message exchange requires unique decoding and unique decoding can only decode from half of the distance guaranteed by the code. We get around this barrier by using list decoding to a super small (size 3) list in phase 1. Although less powerful than full-fledged list-decoding (where lists of any constant size are

[8] We can also make phase 3 longer, but it is easily seen that the same arguments apply as long as phase 3 is less than $4N$ rounds. If phase 3 has length at least $4N$, then the error resilience is at most $\frac{5N}{16N+20N+4N} = \frac{1}{8}$ and does not give us our result.

allowed), even size 3 list-decoding is powerful enough to get around the $\frac{1}{2}$ barrier that unique decoding runs into.

Finally, the last barrier in Subsection 2.2 was that two-way decoding should (roughly) have half the error resilience of one-way decoding as maybe it can be split into two one-way schemes (one for Alice and one for Bob). We do not run into this barrier as in our scheme, Alice needs to know Bob's list in order to send him the index. Since this is not possible without Bob communicating, our scheme cannot be split into two one-way schemes, one for each party.

As is evident from the discussion, all three barriers interact to allow the saving over 1/8.

Analyzing our message exchange protocol. We describe our message exchange protocol more formally in Algorithm 1. Our protocol has a total of $39N$ rounds, where N is chosen to be large enough so that all the codes mentioned in Algorithm 1 exist. It is well known that this can be achieved by an N that is linear in the length of Alice's and Bob's inputs.

Algorithm 1 Our $\frac{5}{39}$ error resilient protocol simplified for the message exchange task.

Phase 1:

1: Alice sends $\mathsf{ECC}(x^A)$ to Bob in the first $16N$ rounds. Here, ECC is a binary error correcting code with distance $\frac{1}{2}$ that is list-decodable from up to $\frac{5}{16}$ fraction of errors with a list of size 3.

2: Bob receives $\rho \in \{0,1\}^{16N}$ from Alice and decodes ρ to get a list L of size at most 3.

Phase 2:

3: Bob sends $\mathsf{ECC}(x^B, L)$ to Alice in the next $20N$ rounds. Recall that ECC is a binary error correcting code with distance $\frac{1}{2}$ and therefore uniquely decodable from up to $\frac{1}{4}$ fraction of errors.

4: Alice unique decodes Bob's message to get x^B and L. She outputs x^B and sets $\mathsf{p} \in [3]$ to be the index of x^A in L.

Phase 3:

5: Alice sends $\mathsf{C}(\mathsf{p})$ to Bob in the final $3N$ rounds. Here, C is a binary error correcting code with only 3 codewords and distance $\frac{2}{3}$. It is well known that such codes exist.

6: Bob receives $\tau \in \{0,1\}^{3N}$ and outputs L_{q} where q minimizes $\delta(\mathsf{q}) = \Delta\left(\mathsf{ECC}(L_{\mathsf{q}}), \rho\right) + \Delta\left(\mathsf{C}(\mathsf{q}), \tau\right)$. Here, $\Delta(\cdot,\cdot)$ denotes the Hamming distance between strings and L_{q} is the q^{th} element in L.

We now argue why, at the end of Algorithm 1, if the number of corruptions is less than $\frac{5}{39} \cdot 39N = 5N$, then both Alice and Bob can output each others' inputs. The reason Alice can output Bob's input x^B is because, in Line 3 of phase 2, Bob sends x^B encoded using an error correcting code that is uniquely decodable from up to $\frac{1}{4}$ errors. As the length of phase 2 is $20N$ rounds and the number of errors is $< \frac{1}{4} \cdot 20N = 5N$, Alice will be able to output x^B (and recover L) correctly.

As Alice can compute L correctly, the correctness of Bob's output follows if we can show that $\mathsf{p} = \mathsf{q}$. For this we first observe that $\delta(\mathsf{p})$ is at most the number of corruptions and is therefore, strictly less than $5N$. As q is chosen to the minimizer of $\delta(\cdot)$, we also have that $\delta(\mathsf{q}) < 5N$ and $\delta(\mathsf{p}) + \delta(\mathsf{q}) < 10N$. Thus, in order to show that $\mathsf{p} = \mathsf{q}$, it is enough to show that $\delta(\mathsf{p}) + \delta(\mathsf{p}') \geq 10N$ for all $\mathsf{p} \neq \mathsf{p}'$. This follows simply using the triangle inequality and the distance property of our codes as:

$$\delta(\mathsf{p}) + \delta(\mathsf{p}') \geq \Delta\left(\mathsf{ECC}(L_{\mathsf{p}}), \mathsf{ECC}(L_{\mathsf{p}'})\right) + \Delta\left(\mathsf{C}(\mathsf{p}), \mathsf{C}(\mathsf{p}')\right) \geq \frac{1}{2} \cdot 16N + \frac{2}{3} \cdot 3N = 10N,$$

where the last inequality is because the distance of ECC is $\frac{1}{2}$ and the distance of C is $\frac{2}{3}$.

2.4 Extending to Interactive Coding

Building a protocol that solves the message exchange task with error resilience $\frac{5}{39} > \frac{1}{8}$ is only the first step in our general simulation. In fact, this first step was almost trivial in the work of [BR11]. It remains to extend the simulation to cover all possible noiseless protocols. This extension has several challenges that are outlined below.

Binary interactive list-decodable codes with a list of size 3. First and foremost comes the challenge that the list-decodable codes used in phase 1 of our protocol for message exchange have no analogue for general interactive tasks. In fact, the only list-decodable interactive error correcting codes for general interactive tasks are the ones described in [BE17, GH14]. These are too weak for us due to three reasons: (1) Firstly, both of the works [BE17] and [GH14] work with a large constant sized alphabet and not the binary alphabet that we work in. (2) Secondly, both of these works ignore constants in the list size they obtain. For us, however, this constant is closely connected to the distance of the codes we can use in phase 3 and therefore directly affects the maximum error resilience we can obtain. (3) Lastly, both of these works do not give the distance properties we need from our list-decodable code in phase 1. In other words, for these works, we do not have a guarantee that any two 'codewords' have a distance of $\frac{1}{2}$ as was needed in our analysis.

A lot of technical work goes into getting list-decodable interactive error correcting codes with the properties listed above. We do this in Section 5 where we build upon the ideas in [BE17] to get a list-decodable interactive code with the desired properties. An important ingredient in this protocol is our notion of 'boosted list-decodable tree codes' that provide stronger guarantees than standard list-decodable tree codes and will be crucial in ensuring the above properties. We define and show the existence of these tree codes in Subsection 4.2 and Subsection 4.3.

We also mention that parameters of the list-decodable interactive codes we construct "correspond" to the optimal parameters in the non-interactive setting in the following sense: In the non-interactive setting, it is well known [Bli86] that it is not possible to decode from more than $\frac{5}{16}$ errors with a list of size 3. Our list decodable interactive codes decode from

up to $\frac{5}{32}$ errors in using a list of size 3. This parameter corresponds to the one for classical codes, up to a factor of $\frac{1}{2}$ that we lose as in our case, both parties need to decode.

Adding a fourth codeword. Our task is not complete even after building the aforementioned list-decodable interactive codes as they turn out to be incompatible with Algorithm 1 in the following sense. Any interactive code that has a sub-exponential blowup requires Alice and Bob to interact. To allow this interaction, we must have Bob send messages to Alice in phase 1 of our scheme when they are running the list decodable interactive code. Consequently, phase 1 of our protocol will be longer, which means that in order to not lose too much in the error resilience, phase 2 of our protocol will have to be much shorter.

However, if phase 2 of our protocol is shorter, then Alice is not guaranteed to decode Bob's list L correctly, which means that the index p that Alice computes may not be the index of the correct output in Bob's actual list, and everything that Alice sends in phase 3 may be meaningless!

To fix this problem, we first observe that in case Alice does not decode L correctly in phase 2, then there must have been many errors in phase 2. As the total number of errors is limited, it means that there must have been relatively fewer errors in phase 1. We ensure that this number is small enough so that the 'most likely' output in phase 1 is the correct one. Additionally, we add an extra codeword to phase 2 that Alice sends when she thinks that phase 2 had many corruptions.

If Bob decodes to this extra codeword in phase 3, then he understands this as being a signal that there were many errors in phase 2 (or in phase 3) and he should simply output the 'most likely' outcome in phase 1. Otherwise, Bob decodes as usual. Finally, as there exists a binary code with 4 codewords and distance $\frac{2}{3}$, this does not affect the error resilience of our protocol.

3 Preliminaries and Formal Problem Definition

Our proof uses the following version of the Chernoff bound.

Lemma 3.1 (Multiplicative Chernoff bound)**.** *Suppose $X_1, \cdots, X_n$ are independent random variables taking values in $\{0,1\}$. Let X denote their sum and let $\mu = \mathbb{E}[X]$ denote the sum's expected value. Then,*

$$\Pr[X \geq (1+\delta)\mu] \leq \mathrm{e}^{-\frac{\delta^2\mu}{3}}, \qquad \forall 0 < \delta < 1,$$
$$\Pr[X \leq (1-\delta)\mu] \leq \mathrm{e}^{-\frac{\delta^2\mu}{2}}, \qquad \forall 0 < \delta < 1.$$

Throughout, we shall use $\Delta(\cdot,\cdot)$ to denote Hamming distance. We also will need the following definition of suffix distance.

Definition 3.2. *Let $n > 0$ and $s, t \in \{0,1\}^n$. Define the suffix distance, $\delta_{\mathsf{suf}}(s,t)$, between s and t as*

$$\delta_{\mathsf{suf}}(s,t) = \max_{i \in [n]} \frac{\Delta(s_{\geq i}, t_{\geq i})}{n-i+1}.$$

For $r > 1$, we extend the definition of suffix distance to strings $s, t \in (\{0,1\}^r)^n$ as follows: If $s = s_1 s_2 \cdots s_n$ where $s_i \in \{0,1\}^r$ for $i \in [n]$, then, let $s' \in \{0,1\}^{rn}$ denote the string $s' = s_1 \| s_2 \| \cdots \| s_n$. Define t' similarly. We define:

$$\delta_{\mathsf{suf}}(s,t) = \delta_{\mathsf{suf}}(s',t').$$

3.1 The Binary Two Party Communication Model

We now formally define the binary two party communication model.

A (deterministic) protocol $\Pi = \{T, p, \mathcal{X}^C, \mathcal{Y}^C, f^C, \mathsf{out}^C\}_{C \in \{A,B\}}$ in the binary two party communication model is defined by a length parameter T, an order of turns given by a sequence $p \in \{A,B\}^T$, input sets $\mathcal{X}^A$ and $\mathcal{X}^B$, output sets $\mathcal{Y}^A$ and $\mathcal{Y}^B$, transmission functions f^A and f^B, and output functions out^A and out^B. Here, for $C \in \{A,B\}$, the functions f^C and out^C are of the types:

$$f^C : \mathcal{X}^C \times \{0,1\}^{<T} \to \{0,1\}, \qquad \mathsf{out}^C : \mathcal{X}^C \times \{0,1\}^T \to \mathcal{Y}^C.$$

Such a protocol Π is executed is the presence of an adversary. We first define an adversary Adv for Π and then define an execution of Π in the presence of Adv. An adversary Adv for Π is defined by two functions Adv^A and Adv^B of the types:

$$\mathsf{Adv}^A, \mathsf{Adv}^B : \mathcal{X}^A \times \mathcal{X}^B \to \{0,1\}^T.$$

An execution of Π in the presence of Adv proceeds as follows: At the beginning of the execution, Alice and Bob start with inputs $x^A \in \mathcal{X}^A$ and $x^B \in \mathcal{X}^B$ respectively. The execution consists of T rounds and before the i^{th} rounds, 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 $\mathsf{Adv}_i^B(x^A, x^B)$. 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 $\mathsf{Adv}_i^A(x^A, x^B)$. Both the parties add these symbols to π^A and π^B respectively. After T such rounds, Alice and Bob output $\mathsf{out}^A(x^A, \pi^A)$ and $\mathsf{out}^B(x^B, \pi^B)$ respectively.

Observe that this execution, and therefore π^A, π^B are completely determined by x^A, x^B, Π, and Adv. We shall often use $\mathsf{out}^A_{\Pi,\mathsf{Adv}}(x^A, x^B)$ to denote $\mathsf{out}^A(x^A, \pi^A)$ and $\mathsf{out}^B_{\Pi,\mathsf{Adv}}(x^A, x^B)$ to denote $\mathsf{out}^B(x^B, \pi^B)$.

Corruptions. Consider an execution of Π in the presence of the adversary Adv. For $R \subseteq [T]$, 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. For $i \in [T]$, we use $\mathsf{corr}_{\Pi,\mathsf{Adv},i}(x^A, x^B)$ to denote $\mathsf{corr}_{\Pi,\mathsf{Adv},\{i\}}(x^A, x^B)$, $\mathsf{corr}_{\Pi,\mathsf{Adv},\leq i}(x^A, x^B)$ to denote $\mathsf{corr}_{\Pi,\mathsf{Adv},[i]}(x^A, x^B)$, $\mathsf{corr}_{\Pi,\mathsf{Adv},>i}(x^A, x^B)$ to denote $\mathsf{corr}_{\Pi,\mathsf{Adv},\{i+1,i+2\cdots,T\}}(x^A, x^B)$, *etc.* Finally, we omit the subscript R when $R = [T]$.

Noiseless adversary. Observe that for any protocol Π, there is a unique adversary Adv that satisfies $\mathsf{corr}_{\Pi,\mathsf{Adv},\leq T}(x^A, x^B) = 0$ for all $x^A \in \mathcal{X}^A$ and $x^B \in \mathcal{X}^B$. We call this adversary the noiseless adversary and denote it by Adv^*. It follows that when Π is executed in the presence of Adv^* and the inputs are x^A and x^B respectively, then we have $\pi^A = \pi^B$. We use $\Pi(x^A, x^B)$ to denote this common value.

4 Results From Coding Theory

4.1 Error Correcting Codes

We will need the following standard result concerning error correcting codes.

Lemma 4.1. *For all $\epsilon > 0$, there exists an integer $n_0 > 0$ such that for all $n \geq n_0$, there exists a function $\mathsf{ECC}_{n,\epsilon} : \{0,1\}^{\lfloor \frac{\epsilon^2 \cdot n}{10} \rfloor} \to \{0,1\}^n$ such that for all $s \neq t \in \{0,1\}^n$, we have*

$$\Delta\left(\mathsf{ECC}_{n,\epsilon}(s), \mathsf{ECC}_{n,\epsilon}(t)\right) \geq \left(\frac{1}{2} - \epsilon\right) \cdot n.$$

We also use the following well known lemma concerning codes with 4 codewords.

Lemma 4.2. *For every $n > 0$, there exists a function $\mathsf{C}_n : \{0,1,2,3\} \to \{0,1\}^{3n}$ such that for all $i \neq i' \in \{0,1,2,3\}$, we have*

$$\Delta\left(\mathsf{C}_n(i), \mathsf{C}_n(i')\right) = 2n.$$

Proof. Let str^z denote the string obtained by concatenating str to itself z times, *e.g.*, $(934)^5 = 934934934934934$. We define the function C_n as follows:

$$\mathsf{C}_n(i) = \begin{cases} (000)^n & , i = 0 \\ (011)^n & , i = 1 \\ (101)^n & , i = 2 \\ (110)^n & , i = 3 \end{cases}.$$

The lemma then follows straightforwardly. □

4.2 List-Decodable Tree Codes with Binary Alphabet

Throughout this section, we fix Σ be a non-empty finite set. Let $S \subseteq \Sigma^*$ be a set of strings over Σ. We define $\mathsf{pre}(S)$ to be the set of all prefixes of all strings in S, *i.e.* the set

$$\mathsf{pre}(S) = \bigcup_{i>0} \mathsf{pre}_i(S) \qquad \text{where} \qquad \mathsf{pre}_i(S) = \{s_{\leq i} \mid s \in S, i \leq |s|\}.$$

For $r > 0$ and a function $f : \Sigma^* \to \{0,1\}^r$, we use $\overline{f} : \Sigma^* \to (\{0,1\}^r)^*$ to denote the function that takes $s \in \Sigma^*$ to an $|s|$-length string over $\{0,1\}^r$ such that the i^{th} coordinate of $\overline{f}(s)$, for $i \in [|s|]$, is $f(s_{\leq i})$. Next, if $r, \alpha > 0$, $f : \Sigma^* \to \{0,1\}^r$ is a function, and $\tilde{s} \in (\{0,1\}^r)^*$ is a string, we define

$$\mathsf{near}^f_\alpha(\tilde{s}) = \bigcup_{i \in [|\tilde{s}|]} \mathsf{near}^f_{\alpha,i}(\tilde{s}) \qquad \text{where} \qquad \mathsf{near}^f_{\alpha,i}(\tilde{s}) = \{s \in \Sigma^i \mid \delta_{\mathsf{suf}}(\overline{f}(s), \tilde{s}_{\leq i}) < \alpha\}.$$

Also, define, for $S \subseteq \Sigma^*$, the value:

$$\delta^f_{\mathsf{pre}}(S, \tilde{s}) = \sum_{i \in [|\tilde{s}|]} \sum_{s \in \mathsf{pre}_i(S)} \Delta(f(s), \tilde{s}_i).$$

The following lemma captures what we need from these definitions.

Lemma 4.3. *For all $r, \alpha > 0$, $\tilde{s} \in (\{0,1\}^r)^*$ and $f : \Sigma^* \to \{0,1\}^r$, we have that*

$$\delta^f_{\mathsf{pre}}(\mathsf{near}^f_\alpha(\tilde{s}), \tilde{s}) \leq r\alpha \cdot |\mathsf{pre}(\mathsf{near}^f_\alpha(\tilde{s}))|.$$

Proof. Consider, for $i > 0$, the function $g_i : \mathsf{pre}_i(\mathsf{near}^f_\alpha(\tilde{s})) \to \mathsf{near}^f_\alpha(\tilde{s})$ that takes $s' \in \mathsf{pre}_i(\mathsf{near}^f_\alpha(\tilde{s}))$ to the lexicographically smallest $s \in \mathsf{near}^f_\alpha(\tilde{s})$ such that s' is a prefix of s. Owing to the definition of pre, at least one such s always exists and therefore g_i is well-defined. Furthermore, observe that g_i is an injection. We get the following equalities:

$$\delta^f_{\mathsf{pre}}(\mathsf{near}^f_\alpha(\tilde{s}), \tilde{s}) = \sum_{i \in [|\tilde{s}|]} \sum_{s \in \mathsf{pre}_i(\mathsf{near}^f_\alpha(\tilde{s}))} \Delta(f(s), \tilde{s}_i) = \sum_{i \in [|\tilde{s}|]} \sum_{s \in \mathsf{im}(g_i)} \Delta(f(s_{\leq i}), \tilde{s}_i) \tag{1}$$

$$|\mathsf{pre}(\mathsf{near}^f_\alpha(\tilde{s}))| = \sum_{i \in [|\tilde{s}|]} |\mathsf{pre}_i(\mathsf{near}^f_\alpha(\tilde{s}))| = \sum_{i \in [|\tilde{s}|]} |\mathsf{im}(g_i)| = \sum_{i \in [|\tilde{s}|]} \sum_{s \in \mathsf{im}(g_i)} 1. \tag{2}$$

To proceed, we note from the definition of g_i, that if $s \in \mathsf{im}(g_i)$ for some $i > 0$, then $s \in \mathsf{im}(g_{i'})$ for all $i \leq i' \leq |s|$. Using $i_*(s)$ to denote the smallest i such that $s \in \mathsf{im}(g_i)$ and defining $i_*(s) = |s| + 1$ if no such i exists, we get:

$$\begin{aligned}
\delta^f_{\mathsf{pre}}(\mathsf{near}^f_\alpha(\tilde{s}), \tilde{s}) &= \sum_{i \in [|\tilde{s}|]} \sum_{s \in \mathsf{near}^f_\alpha(\tilde{s})} \mathbb{1}\left(s \in \mathsf{im}(g_i)\right) \cdot \Delta(f(s_{\leq i}), \tilde{s}_i) && \text{(Equation 1)} \\
&= \sum_{s \in \mathsf{near}^f_\alpha(\tilde{s})} \sum_{i \in [|\tilde{s}|]} \mathbb{1}\left(s \in \mathsf{im}(g_i)\right) \cdot \Delta(f(s_{\leq i}), \tilde{s}_i)
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{s \in \mathsf{near}^f_\alpha(\tilde{s})} \sum_{i=i_*(s)}^{|\tilde{s}|} \Delta(f(s_{\le i}), \tilde{s}_i) \\
&\le \sum_{s \in \mathsf{near}^f_\alpha(\tilde{s})} r \cdot (|s| + 1 - i_*(s)) \cdot \delta_{\mathsf{suf}}\left(\overline{f}(s), \tilde{s}_{\le |s|}\right) \\
&\le \sum_{s \in \mathsf{near}^f_\alpha(\tilde{s})} r\alpha \cdot (|s| + 1 - i_*(s)) && (\text{As } s \in \mathsf{near}^f_\alpha(\tilde{s})) \\
&= r\alpha \cdot \sum_{s \in \mathsf{near}^f_\alpha(\tilde{s})} \sum_{i=i_*(s)}^{|\tilde{s}|} 1 \\
&= r\alpha \cdot |\mathsf{pre}(\mathsf{near}^f_\alpha(\tilde{s}))|. && (\text{Equation 2})
\end{aligned}
$$

□

We are now ready to define list tree codes.

Definition 4.4 (List-Decodable Tree Codes with Binary Alphabet)**.** *Let $r, L, \alpha > 0$ and $f : \Sigma^* \to \{0,1\}^r$ be a function. We say that f is an (r, L, α)-list tree code if for all $\tilde{s} \in (\{0,1\}^r)^*$, we have:*

$$|\mathsf{pre}(\mathsf{near}^f_\alpha(\tilde{s}))| < L \cdot |\tilde{s}|.$$

4.3 Boosted List Tree Codes

It turns out that the above notion of list tree codes will not be sufficient for our needs and we need to boost it to get stronger guarantees. We next define these boosted tree codes. Define the separation function $\mathsf{sep} : \{0,1,2,3,4\} \to \mathbb{R}$ as follows:

$$\mathsf{sep}(i) = \begin{cases} 0 & , i \in \{0,1\} \\ \frac{i}{4} & , i \in \{2,3\} \\ \frac{5}{4} & , i = 4 \end{cases}.$$

The function $\mathsf{sep}(i)$ captures the following intuition: Suppose that i bit strings are drawn at random, and consider the bit string formed by taking the coordinate-wise majority. Then, $\mathsf{sep}(i)$ is simply the expected sum of the fractional Hamming distances from the majority string to all the i original strings. We also extend the definition of $\mathsf{pre}(\cdot)$ to sets $S \subseteq \Sigma^* \times \mathbb{N}$ as follows :

$$\mathsf{pre}(S) = \bigcup_{i>0} \mathsf{pre}_i(S) \qquad \text{where} \qquad \mathsf{pre}_i(S) = \{s_{\le i} \mid (s,l) \in S, l < i \le |s|\}.$$

We are now ready to define boosted list tree codes and show that they exist.

Definition 4.5. *Let $r, L, \alpha > 0$ and $f : \Sigma^* \to \{0,1\}^r$ be a function. We say that f is an (r, L, α)-boosted list tree code if:*

- *f is an (r, L, α)-list tree code.*
- *For all $k > 0$ and all $S \subseteq \Sigma^{\leq k} \times \mathbb{N}$ such that $|S| \leq 4$ and $|\mathsf{pre}(S)| \geq k(1+\epsilon)$, we have:*

$$\sum_{i\in[k]} \min_{t\in\{0,1\}^r} \sum_{s\in\mathsf{pre}_i(S)} \Delta(f(s),t) \geq r(1-\epsilon)\cdot \sum_{i\in[k]} \mathsf{sep}(|\mathsf{pre}_i(S)|).$$

Theorem 4.6. *Let $\epsilon > 0$ be fixed. For all $r \geq \frac{100\cdot\log_2(|\Sigma|+1)}{\epsilon^3}$, $L \geq \frac{100}{\epsilon^2}$, there exists an $\left(r, L, \frac{1}{2}-\epsilon\right)$-boosted list tree code.*

Proof. Define $\alpha = \frac{1}{2} - \epsilon$. Let $r \geq \frac{100\cdot\log_2(|\Sigma|+1)}{\epsilon^3}$ and $L \geq \frac{100}{\epsilon^2}$ be fixed. Let $f : \Sigma^* \to \{0,1\}^r$ be a random function and consider the following events defined over the randomness in f

$$E_1 \equiv \exists \tilde{s} \in (\{0,1\}^r)^* : |\mathsf{pre}(\mathsf{near}^f_\alpha(\tilde{s}))| \geq L\cdot|\tilde{s}|.$$
$$E_2 \equiv \exists k > 0, S \subseteq \Sigma^{\leq k}\times\mathbb{N} : |S| \leq 4 \wedge |\mathsf{pre}(S)| \geq k(1+\epsilon)$$
$$\wedge \sum_{i\in[k]} \min_{t\in\{0,1\}^r} \sum_{s\in\mathsf{pre}_i(S)} \Delta(f(s),t) < r(1-\epsilon)\cdot\sum_{i\in[k]} \mathsf{sep}(|\mathsf{pre}_i(S)|).$$

To show the theorem, we simply show that

Claim 4.7. $\Pr(E_1) \leq 4\cdot 2^{-\epsilon^2 rL}$.

Claim 4.8. $\Pr(E_2) \leq 2\cdot 2^{-\frac{r\epsilon^3}{100}}$.

The theorem then follows as $\Pr(E_1)+\Pr(E_2) < 1$. We now show Claim 4.7 and Claim 4.8.

Proof of Claim 4.7. We have by the union bound:

$$\begin{aligned}
\Pr(E_1) &\leq \sum_{k>0} \sum_{\tilde{s}\in(\{0,1\}^r)^k} \Pr\left(|\mathsf{pre}(\mathsf{near}^f_\alpha(\tilde{s}))| \geq Lk\right)\\
&\leq \sum_{k>0} \sum_{\tilde{s}\in(\{0,1\}^r)^k} \Pr\left(\exists S\subseteq\Sigma^*, |\mathsf{pre}(S)| \geq Lk : \delta^f_{\mathsf{pre}}(S,\tilde{s}) \leq r\alpha\cdot|\mathsf{pre}(S)|\right) \quad \text{(Lemma 4.3)}\\
&\leq \sum_{k>0} \sum_{\tilde{s}\in(\{0,1\}^r)^k} \sum_{k'\geq Lk} \sum_{\substack{S\subseteq\Sigma^*\\|\mathsf{pre}(S)|=k'}} \Pr\left(\delta^f_{\mathsf{pre}}(S,\tilde{s}) \leq r\alpha k'\right)\\
&\leq \sum_{k>0} \sum_{\tilde{s}\in(\{0,1\}^r)^k} \sum_{k'\geq Lk} \sum_{\substack{S\subseteq\Sigma^*\\|\mathsf{pre}(S)|=k'}} \frac{1}{2^{rk'}} \sum_{z=0}^{r\alpha k'} \binom{rk'}{z}\\
&\leq \sum_{k>0} \sum_{\tilde{s}\in(\{0,1\}^r)^k} \sum_{k'\geq Lk} \sum_{\substack{S\subseteq\Sigma^*\\|\mathsf{pre}(S)|=k'}} 2^{rk'\cdot(H(\alpha)-1)},
\end{aligned}$$

using the well known identity $\sum_{i=0}^m \binom{n}{m} \leq 2^{nH(m/n)}$ when $0 \leq m \leq n/2 < n$ and $H(\cdot)$ is the binary entropy function, *i.e.*, $H(x) = -x\log_2(x) - (1-x)\log_2(1-x)$ for all $x \in (0,1)$. in order to continue, we note that

Claim 4.9. *For $k' > 0$, we have*

$$|\{S \subseteq \Sigma^* \mid |\mathsf{pre}(S)| = k'\}| \leq (|\Sigma|+1)^{2k'} \cdot 2^{k'} \leq (|\Sigma|+1)^{3k'}$$

Proof. Let $S \subseteq \mathcal{S}$ be such that $|\mathsf{pre}(S)| = k'$. To start, observe that we can view $\mathsf{pre}(S)$ as a tree with k' edges as follows: The root of this tree (depth 0) is the empty string ε, and the vertices at depth i are the elements of $\mathsf{pre}_i(S)$. There is an edge between a vertex at depth $i-1$ and a vertex at depth i if and only if the former is a prefix of the latter.

Next, we note that tree above determines a superset of S of size k'. Indeed, all the elements of S correspond to one of the $k'+1$ vertices in the tree (and no element corresponds to the root). Furthermore, the tree can be described by a string in $(\Sigma \cup \{\uparrow\})^{2k'}$ using a standard depth-first traversal (we assume without loss of generality that $\uparrow \notin \Sigma$). The lemma follows.

□

Using this bound and the fact that $1 - H(\alpha) \geq 2\epsilon^2$, we get:

$$\begin{aligned}
\Pr(E_1) &\leq \sum_{k>0} \sum_{\tilde{s} \in (\{0,1\}^r)^k} \sum_{k' \geq Lk} (|\Sigma|+1)^{3k'} \cdot 2^{-2\epsilon^2 r k'} \\
&\leq \sum_{k>0} \sum_{\tilde{s} \in (\{0,1\}^r)^k} \sum_{k' \geq Lk} 2^{-1.5 \cdot \epsilon^2 r k'} && \text{(Choice of } r\text{)} \\
&\leq 2 \cdot \sum_{k>0} \sum_{\tilde{s} \in (\{0,1\}^r)^k} 2^{-1.5 \cdot \epsilon^2 r L k} \\
&\leq 2 \cdot \sum_{k>0} 2^{-\epsilon^2 r L k} && \text{(Choice of } L\text{)} \\
&\leq 4 \cdot 2^{-\epsilon^2 r L}.
\end{aligned}$$

□

Proof of Claim 4.8. We have by the union bound:

$$\begin{aligned}
\Pr(E_2) &\leq \sum_{k>0} \sum_{\substack{S \subseteq \Sigma^{\leq k} \times \mathbb{N} \\ |S| \leq 4 \\ |\mathsf{pre}(S)| \geq k(1+\epsilon)}} \Pr\left(\sum_{i \in [k]} \min_{t \in \{0,1\}^r} \sum_{s \in \mathsf{pre}_i(S)} \Delta(f(s), t) < r(1-\epsilon) \cdot \sum_{i \in [k]} \mathsf{sep}(|\mathsf{pre}_i(S)|) \right) \\
&\leq \sum_{k>0} \sum_{\substack{S \subseteq \Sigma^{\leq k} \times \mathbb{N} \\ |S| \leq 4 \\ |\mathsf{pre}(S)| \geq k(1+\epsilon)}} \Pr\left(\sum_{i \in [k]} \sum_{i' \in [r]} \min_{b \in \{0,1\}} \sum_{s \in \mathsf{pre}_i(S)} \Delta(f_{i'}(s), b) < r(1-\epsilon) \cdot \sum_{i \in [k]} \mathsf{sep}(|\mathsf{pre}_i(S)|) \right).
\end{aligned}$$

We now upper bound each term in the summand above. Fix $k > 0$ and $S \subseteq \Sigma^{\leq k} \times \mathbb{N}$ be prefix free such that $|S| \leq 4$. For $i \in [k]$ and $i' \in [r]$, we define the random variable

$$X_{i,i'} = \min_{b \in \{0,1\}} \sum_{s \in \mathsf{pre}_i(S)} \Delta(f_{i'}(s), b).$$

We claim that

Claim 4.10. *For all $i \in [k]$, $i' \in [r]$, we have $\mathbb{E}[X_{i,i'}] = \mathsf{sep}(|\mathsf{pre}_i(S)|)$.*

Proof. We show this by a case analysis on $|\mathsf{pre}_i(S)|$. When $|\mathsf{pre}_i(S)| \in \{0, 1\}$, there is nothing to show as both sides are 0 deterministically. In all other cases, we use the fact that f is a random function to conclude that $f_{i'}(s)$ are chosen uniformly and independently chosen bits for all $s \in \mathsf{pre}_i(S)$. When $|\mathsf{pre}_i(S)| = 2$, we have

$$\mathbb{E}[X_{i,i'}] = \frac{1}{4} \cdot 0 + \frac{2}{4} \cdot 1 + \frac{1}{4} \cdot 0 = \frac{1}{2} = \mathsf{sep}(2).$$

When $|\mathsf{pre}_i(S)| = 3$, we have

$$\mathbb{E}[X_{i,i'}] = \frac{1}{8} \cdot 0 + \frac{3}{8} \cdot 1 + \frac{3}{8} \cdot 1 + \frac{1}{8} \cdot 0 = \frac{3}{4} = \mathsf{sep}(3).$$

Finally, when $|\mathsf{pre}_i(S)| = 4$, we have

$$\mathbb{E}[X_{i,i'}] = \frac{1}{16} \cdot 0 + \frac{4}{16} \cdot 1 + \frac{6}{16} \cdot 2 + \frac{4}{16} \cdot 1 + \frac{1}{16} \cdot 0 = \frac{5}{4} = \mathsf{sep}(4). \qquad \square$$

As $X_{i,i'}$ are independent random variables for all i, i', we have, using this bound and Lemma 3.1, that:

$$\begin{aligned}
\Pr(E_2) &\leq \sum_{k>0} \sum_{\substack{S \subseteq \Sigma^{\leq k} \times \mathbb{N} \\ |S| \leq 4 \\ |\mathsf{pre}(S)| \geq k(1+\epsilon)}} \Pr\left(\sum_{i \in [k]} \sum_{i' \in [r]} X_{i,i'} < (1-\epsilon) \cdot \sum_{i \in [k]} \sum_{i' \in [r]} \mathbb{E}[X_{i,i'}] \right) \\
&\leq \sum_{k>0} \sum_{\substack{S \subseteq \Sigma^{\leq k} \times \mathbb{N} \\ |S| \leq 4 \\ |\mathsf{pre}(S)| \geq k(1+\epsilon)}} \exp\left(-\frac{r\epsilon^2 \cdot \sum_{i \in [k]} \mathsf{sep}(|\mathsf{pre}_i(S)|)}{2} \right).
\end{aligned}$$

To continue, we use the fact that $|\mathsf{pre}(S)| \geq k(1+\epsilon) \implies \sum_{i \in [k]} \mathsf{sep}(|\mathsf{pre}_i(S)|) \geq \frac{\epsilon k}{4}$ to get:

$$\begin{aligned}
\Pr(E_2) &\leq \sum_{k>0} (|\Sigma| + 1)^{10k} \cdot \exp\left(-\frac{r\epsilon^3 k}{20} \right) \\
&\leq \sum_{k>0} 2^{-\frac{r\epsilon^3 k}{100}} && \text{(Choice of } r\text{)} \\
&\leq 2 \cdot 2^{-\frac{r\epsilon^3}{100}}. && \square
\end{aligned}$$

$\square$

5 Binary List-Decodable Interactive Codes With Small Lists

The goal of this section is to show:

Theorem 5.1. *Let $0 < \epsilon < \frac{1}{20}$ be a parameter and $\Pi = \{T, p, \mathcal{X}^C, \mathcal{Y}^C, f^C, \mathsf{out}^C\}_{C\in\{A,B\}}$ be a protocol in the binary two party communication model as in Subsection 3.1. There is a protocol $\Pi' = \{T', p', \mathcal{X}'^C, \mathcal{Y}'^C, f'^C, \mathsf{out}'^C\}_{C\in\{A,B\}}$ such that:*

1. *We have $T' = 10^{10} \cdot \frac{T}{\epsilon^{60}}$, $\mathcal{X}'^C = \mathcal{X}^C$ for all $C \in \{A, B\}$, and $\mathcal{Y}'^A = \mathcal{Y}'^B = 2^{\{0,1\}^T} \times \{d \mid d : \{0,1\}^T \to \mathbb{N}\}$.*
2. *For all $x^A \in \mathcal{X}^A$, $x^B \in \mathcal{X}^B$, and all adversaries Adv' for Π', if $(L^C, \mathsf{d}^C) = \mathsf{out}'^C_{\Pi',\mathsf{Adv}'}(x^A, x^B)$ for all $C \in \{A, B\}$, then*
   (a) *For all $C \in \{A, B\}$, we have $|L^C| \le 3$ and for all $s \neq s' \in L^C$, we have*
   $$\mathsf{d}^C(s) + \mathsf{d}^C(s') \ge \left(\frac{1}{4} - \frac{\epsilon}{2}\right) \cdot T'.$$
   (b) *If $\mathsf{corr}_{\Pi',\mathsf{Adv}'}(x^A, x^B) \le \left(\frac{5}{32} - \epsilon\right) \cdot T'$, then $L^A \cap L^B = \{\Pi(x^A, x^B)\}$ and for all $C \in \{A, B\}$, we have:*
   $$\mathsf{d}^C(\Pi(x^A, x^B)) \le \mathsf{corr}_{\Pi',\mathsf{Adv}'}(x^A, x^B) + \frac{\epsilon T'}{2}.$$

For the rest of the section, we fix a parameter $0 < \epsilon < \frac{1}{20}$ and a protocol $\Pi = \{T, p, \mathcal{X}^C, \mathcal{Y}^C, f^C, \mathsf{out}^C\}_{C\in\{A,B\}}$ in the binary two party communication model. At the cost of blowing up the number of rounds in Π by a factor of 2, we can assume that Π is an alternating binary protocol, *i.e.*, Alice transmits in the odd rounds of Π and Bob transmits in the even rounds of Π. Thus, henceforth, we assume Π has $2T$ rounds and $p_{2i-1} = A$ and $p_{2i} = B$ for all $i \in [T]$. Let Σ be an alphabet satisfying $\log_2(|\Sigma| + 1) = \frac{1}{\epsilon^{50}}$. We shall need the following definitions based on ϵ:

$$\begin{aligned} N &= \frac{32T}{\epsilon}, & r &= \frac{T'}{2N}, & \alpha &= \frac{1}{2} - \frac{\epsilon}{10}, \\ L &= \frac{10^4}{\epsilon^2}, & K &= \frac{100L}{\epsilon}, & \theta &= \frac{5}{32} - \epsilon. \end{aligned} \tag{3}$$

Observe that our choice of parameters in Equation 3 implies by Theorem 4.6 that an (r, L, α)-boosted list tree code over the alphabet Σ exists. We shall use TC to denote this tree code. Let Γ denote the set $\mathbb{N} \times \{0,1\} \times \{0,1\}^{\le 2}$. For $z = (a, b, c) \in \Gamma$, we define $z_1 = a$, $z_2 = b$, and $z_3 = c$. In our simulation, we also use the function $\mathsf{Enc} : \Gamma^* \to \Sigma^*$ guaranteed by the following fact (this follows from simple Huffman Coding):

Fact 5.2. *There exists a function $\mathsf{Enc} : \Gamma^* \to \Sigma^*$ satisfying:*

1. *For all* $Z \in \Gamma^*$*, we have that*

$$|\mathsf{Enc}(Z)| \leq 1 + \left\lfloor \frac{10}{\log_2(|\Sigma|)} \cdot \sum_{i \in [|Z|]} \log_2(Z_{i,1} + 1) \right\rfloor .$$

2. *For all* $Z_1 \neq Z_2 \in \Gamma^*$*, we have* $\mathsf{Enc}(Z_1) \not\preceq \mathsf{Enc}(Z_2)$*. In particular, this means that* $\mathsf{Enc}(Z) \neq \varepsilon$ *for any* $Z \in \Gamma^*$ *and that the function* Enc *is invertible. We use* Dec *to denote the function that, for all* $s \in \Sigma^*$*, either outputs the (unique)* $Z \in \Gamma^*$ *such that* $s = \mathsf{Enc}(Z)$*, or outputs 'not decodable', in case no such* Z *exists.*

We define the function $\mathsf{Enc}^* : (\Gamma^*)^* \to \Sigma^*$ to be such that, if $\mathcal{Z} = \left(\mathcal{Z}_1, \mathcal{Z}_2, \cdots, \mathcal{Z}_{|\mathcal{Z}|}\right) \in (\Gamma^*)^*$, we have

$$\mathsf{Enc}^*(\mathcal{Z}) = \mathsf{Enc}(\mathcal{Z}_1) \| \mathsf{Enc}(\mathcal{Z}_2) \| \cdots \| \mathsf{Enc}(\mathcal{Z}_{|\mathcal{Z}|}).$$

We define $\mathsf{Enc}^*(\varepsilon) = \varepsilon$ for convenience. Due to item 2 of Fact 5.2 above, we may define a function Dec^* such that, for $s \in \Sigma^*$, we have $\mathsf{Dec}^*(s) = \mathcal{Z}'$, where $\mathcal{Z}'$ is the longest such that $\mathsf{Enc}^*(\mathcal{Z}') \preceq s$. Note that $\mathsf{Dec}^*(s)$ is well defined as at least one such $\mathcal{Z}'$, namely $\mathcal{Z}' = \varepsilon$, the empty string, always exists, and $\mathsf{Dec}^*(\mathsf{Enc}^*(\mathcal{Z})) = \mathcal{Z}$ for all $\mathcal{Z} \in (\Gamma^*)^*$. Finally, we assume for $C \in \{A, B\}$ that $f^C(x^C, \psi) = 0$ for all $\psi \in \{0,1\}^*$ such that $|\psi| \geq 2T$. Accordingly, we will also assume $\Pi(x^A, x^B)_l = 0$ for all $l > 2T$.

We are now ready to present our simulation protocol Π' that proves Theorem 5.1. We only describe Alice's side of the protocol, *i.e.*, the functions f'^A and out'^A. Bob's side of the protocol is symmetric.

5.1 Proof of Theorem 5.1

We now present our proof of Theorem 5.1. We note that item 1 of Theorem 5.1 follows straightforwardly from the definition of the protocol Π' in Algorithm 2. It remains to show item 2. To this end, we fix inputs $x^A \in \mathcal{X}^A$, $x^B \in \mathcal{X}^B$, and an adversary Adv' for Π'. As the protocol Π' is deterministic, fixing x^A, x^B, and Adv' completely determines the execution of Π'. In particular, it fixes the value of all variables in Algorithm 2 at all points in its execution.

For a variable *var* in Algorithm 2, we shall use var^A to denote Alice's value of the variable *var* after Algorithm 2 has finished execution when the inputs are x^A, x^B and the adversary is Adv'. Observe that with this notation, we have $(L^C, \mathsf{d}^C) = \mathsf{out}'^C_{\Pi', \mathsf{Adv}'}(x^A, x^B)$ for $C \in \{A, B\}$, and item 2 of Theorem 5.1 follows from the following theorem.

Theorem 5.3. *For all* $C \in \{A, B\}$*, it holds that:*

1. $|L^C| \leq 3$ *and for all* $s \neq s' \in L^C$*, we have*

$$\mathsf{d}^C(s) + \mathsf{d}^C(s') \geq \left(\frac{1}{4} - \frac{\epsilon}{2}\right) \cdot T'.$$

Algorithm 2 Alice's side of the simulation protocol

Input: An input $x^A \in \mathcal{X}^A$.

Output: A list $L \subseteq \{0,1\}^{2T}$ and a function $\mathsf{d} : \{0,1\}^{2T} \to \mathbb{N}$.

$\forall s \in \Sigma^* : \mathsf{sent}(s) \leftarrow \perp$ and $\forall \psi \in \{0,1\}^{2T} : \mathsf{d}(\psi) \leftarrow \infty$.

$\sigma, \tau \leftarrow \varepsilon$.

Communication Phase:

for $i \in [N]$ **do**
  if $|\sigma| < i$ **then**
    if $i = 1$ **then**
      $\mathsf{sent}(\varepsilon) \leftarrow f^A(x^A, \varepsilon)$ and $E \leftarrow [(1, 0, f^A(x^A, \varepsilon))]$ and $\mathcal{E} \leftarrow \texttt{FIND}\,(E)$.
    else if $|\mathsf{near}^{\mathsf{TC}}_{\alpha,i-1}(\tau)| \leq K$ **then**
      $E \leftarrow \varepsilon$.
      for $s \in \mathsf{near}^{\mathsf{TC}}_{\alpha,i-1}(\tau)$ **do**
        Let s' be the largest prefix of s such that $\mathsf{sent}(s') \neq \perp$. As $\mathsf{sent}(\varepsilon) \neq \perp$, s' is well-defined. Define $B(b) = b \| f^A(x^A, \mathsf{sent}(s') \| b)$ for $b \in \{0,1\}$.
        $\tilde{\mathcal{E}} \leftarrow \texttt{FIND}\,(\mathsf{Dec}^*(s))$.
        if $\exists$ unique $b \in \{0,1\} : \exists \tilde{p} : \tilde{\mathcal{E}}[\tilde{p}] = \mathsf{sent}(s') \| b$ **then**
          $\mathsf{sent}(s) \leftarrow \mathsf{sent}(s') \| B(b)$.
          $p \leftarrow \max \{p' \in [|\mathcal{E}|] \mid \mathcal{E}_{p'} \in \{\mathsf{sent}(s'), \mathsf{sent}(s') \| B(0), \mathsf{sent}(s') \| B(1)\}\}$.
          $E \leftarrow E \| \left(|\mathcal{E}| + 1 - p, \mathbb{1}(\mathcal{E}_p \neq \mathsf{sent}(s')), B(b)\right)$.
          $\mathcal{E} \leftarrow \texttt{FIND}\,(\mathsf{Dec}^*(\sigma) \| E)$.
        end if
      end for
    end if
    $\sigma \leftarrow \sigma \| \mathsf{Enc}(E)$.
  end if
  Send $\mathsf{TC}(\sigma_{\leq i})$ to Bob.
  Receive message $m \in \{0,1\}^r$ from Bob. Set $\tau \leftarrow \tau \| m$.
end for

Output Phase:

$\Psi \leftarrow \left\{\psi \in \{0,1\}^{2T} \mid \forall i \in [T] : \psi_{2i-1} = f^A(x^A, \psi_{\leq 2(i-1)})\right\}$.

for $\psi \in \Psi$ **do**
  $S \leftarrow \{s \in \mathsf{near}^{\mathsf{TC}}_{\alpha}(\tau) \mid \exists \text{ unique } \psi \in \Psi : \exists p : \texttt{FIND}\,(\mathsf{Dec}^*(s))_p = \psi\}$.
  $\forall s \in S : I(s) \leftarrow \left\{i \in [|s|] \mid \exists p : \texttt{FIND}\,(\mathsf{Dec}^*(s_{\leq i}))_p = \psi\right\}$.
  $\mathsf{d}(\psi) \leftarrow \min_{s \in S} \min_{i \in I(s)} r\alpha \cdot (i + N - |s|) + \sum_{i'=i+1}^{|s|} \Delta\left(\mathsf{TC}(s_{\leq i'}), \tau_{i'}\right)$.
end for

Output $L = \left\{\psi \in \Psi \mid \mathsf{d}(\psi) < \left(\theta + \frac{\epsilon}{2}\right) T'\right\}$ and d.

Algorithm 3 The function $\texttt{FIND}(\mathcal{Z})$.

Input: $\mathcal{Z} = \left(\mathcal{Z}_1, \mathcal{Z}_2, \cdots, \mathcal{Z}_{|\mathcal{Z}|}\right) \in (\Gamma^*)^*$.
Output: $\mathcal{E} \in (\{0,1\}^*)^*$.
$Z \leftarrow \mathcal{Z}_1 \| \mathcal{Z}_2 \| \cdots \| \mathcal{Z}_{|\mathcal{Z}|}$, $\mathcal{E} \leftarrow \varepsilon$.
for $i \in [|Z|]$ **do**
  if $Z_{i,1} > i$ **then**
    return ϵ, the empty list.
  else if $Z_{i,1} = i$ **then**
    $\mathcal{E} \leftarrow \mathcal{E} \| Z_{i,3}$.
  else if $Z_{i,2} = 0$ **then**
    $\mathcal{E} \leftarrow \mathcal{E} \| \left(\mathcal{E}_{i-Z_{i,1}} \| Z_{i,3}\right)$.
  else
    $\mathcal{E} \leftarrow \mathcal{E} \| \left(\mathcal{E}_{i-Z_{i,1},\leq|\mathcal{E}_{i-Z_{i,1}}|-2} \| Z_{i,3}\right)$.
  end if
end for

2. *If* $\mathsf{corr}_{\Pi',\mathsf{Adv}'}(x^A, x^B) < \theta T'$, *then* $\Pi(x^A, x^B) \in L^C$ *and*

$$\mathsf{d}^C(\Pi(x^A, x^B)) \leq \mathsf{corr}_{\Pi',\mathsf{Adv}'}(x^A, x^B) + \frac{\epsilon T'}{2}.$$

Proof of item 2 of Theorem 5.1 assuming Theorem 5.3. All claims in item 2 of Theorem 5.1 follow directly except that

$$\mathsf{corr}_{\Pi',\mathsf{Adv}'}(x^A, x^B) < \theta T' \implies L^A \cap L^B \subseteq \{\Pi(x^A, x^B)\}.$$

Even this claim holds (in fact, without $\mathsf{corr}_{\Pi',\mathsf{Adv}'}(x^A, x^B) < \theta T'$ assumption) due to the following reasoning: As $L^A \subseteq \Psi^A$ and $L^B \subseteq \Psi^B$, we have $L^A \cap L^B \subseteq \Psi^A \cap \Psi^B$. Also, we have $\Psi^A \cap \Psi^B = \{\Pi(x^A, x^B)\}$ by definition of $\Pi(x^A, x^B)$ finishing the proof. □

Henceforth, we focus on showing Theorem 5.3. We only show Theorem 5.3 for $C = A$ as the proof for $C = B$ is analogous. As we only deal with the $C = A$, we omit A from our notation for variables for convenience, *i.e.*, *var* would denote var^A for all variables *var* in Algorithm 2. We prove item 1 of Theorem 5.3 in the rest of this subsection and devote the following subsections to the proof item 2 of Theorem 5.3. We note that the proof of item 1 is the only place where we use our boosted notion of list tree codes (Definition 4.5). The proof of item 2 only uses Definition 4.4 and would work for all list tree codes.

5.1.1 Proof of item 1 of Theorem 5.3

We show Lemma 5.5 and item 1 of Theorem 5.3 follows as a corollary. We start with the following simple observation.

Observation 5.4. *For all* $\mathcal{Z} \preceq \mathcal{Z}'$, *we have* $\texttt{FIND}(\mathcal{Z}) \preceq \texttt{FIND}(\mathcal{Z}')$.

Lemma 5.5. *If $L' \subseteq L$ such that $|L'| \leq 4$, we have*

$$\sum_{\psi \in L'} \mathsf{d}(\psi) \geq rN \cdot \left(1 - \frac{\epsilon}{2}\right) \cdot \mathsf{sep}(|L'|).$$

Proof. When $|L'| < 2$, there is nothing to show as $\mathsf{sep}(|L'|) = 0$. Suppose for the sake of contradiction that there exists $L' \subseteq L^C$, $2 \leq |L'| \leq 4$ such that

$$\sum_{\psi \in L'} \mathsf{d}(\psi) < rN \cdot \left(1 - \frac{\epsilon}{2}\right) \cdot \mathsf{sep}(|L'|). \tag{4}$$

This means that, for all $\psi \in L'$, the value of $\mathsf{d}(\psi)$ was set in Line 35 and there exist $s(\psi)$ and $i(\psi)$ such that $i(\psi) \leq |s(\psi)|$ and $\psi \in \Psi$ is unique such that $\exists p : \texttt{FIND}\left(\mathsf{Dec}^*(s(\psi)_{\leq i(\psi)})\right)_p = \psi$ (see Observation 5.4) and

$$\mathsf{d}(\psi) = r\alpha \cdot (i(\psi) + N - |s(\psi)|) + \sum_{i'=i(\psi)+1}^{|s(\psi)|} \Delta\left(\mathsf{TC}(s(\psi)_{\leq i'}), \tau_{i'}\right). \tag{5}$$

Next, we claim that

Claim 5.6. *For $\psi \neq \psi' \in L'$ and $i(\psi), i(\psi') \leq l \leq |s(\psi)|, |s(\psi')|$, we have $s(\psi)_{\leq l} \neq s(\psi')_{\leq l}$.*

Proof. Proof by contradiction. Suppose there exist $\psi \neq \psi' \in L'$ and $i(\psi), i(\psi') \leq l \leq |s(\psi)|, |s(\psi')|$ such that $s(\psi)_{\leq l} = s(\psi')_{\leq l}$. We assume that $i(\psi) \leq i(\psi')$ without loss of generality. Observe that $s(\psi)_{\leq l} = s(\psi')_{\leq l}$ implies that $s(\psi)_{\leq i(\psi')} = s(\psi')_{\leq i(\psi')}$ By Observation 5.4 and our choice of $i(\cdot)$, we have that

$$\exists p : \texttt{FIND}\left(\mathsf{Dec}^*(s(\psi)_{\leq i(\psi')})\right)_p = \psi \qquad \text{and} \qquad \exists p : \texttt{FIND}\left(\mathsf{Dec}^*(s(\psi')_{\leq i(\psi')})\right)_p = \psi',$$

contradicting the fact that $\psi' \in \Psi$ is unique such that $\exists p : \texttt{FIND}\left(\mathsf{Dec}^*(s(\psi')_{\leq i(\psi')})\right)_p = \psi'$. □

Define the set $S' = \{(s(\psi), i(\psi)) \mid \psi \in L'\}$. It follows from Claim 5.6 that $|S'| = |L'|$. Using the definition of $\mathsf{pre}(\cdot)$, we derive:

$$\begin{aligned}
|\mathsf{pre}(S')| &= \sum_{\psi \in L'} |s(\psi)| - i(\psi) && \text{(Claim 5.6)} \\
&\geq \sum_{\psi \in L'} N - \frac{1}{r\alpha} \cdot \mathsf{d}(\psi) && \text{(Equation 5)} \\
&> N \cdot |L'| - N \cdot \frac{\left(1 - \frac{\epsilon}{2}\right) \cdot \mathsf{sep}(|L'|)}{\alpha} && \text{(Equation 4)} \\
&\geq N\left(1 + \frac{\epsilon}{8}\right). && (2 \leq |L'| \leq 4)
\end{aligned}$$

This allow us to use Definition 4.5 to get:

$$\sum_{i' \in [N]} \sum_{s' \in \mathsf{pre}_{i'}(S')} \Delta(\mathsf{TC}(s'), \tau_{i'}) \geq r \cdot \left(1 - \frac{\epsilon}{10}\right) \cdot \sum_{i' \in [N]} \mathsf{sep}(|\mathsf{pre}_{i'}(S')|). \tag{6}$$

This allows us to derive a contradiction as follows. We have

$$\begin{aligned}
rN \cdot \left(1 - \frac{\epsilon}{2}\right) \cdot \mathsf{sep}(|S'|) &= rN \cdot \left(1 - \frac{\epsilon}{2}\right) \cdot \mathsf{sep}(|L'|) && (|S'| = |L'|) \\
&> \sum_{\psi \in L'} \mathsf{d}(\psi) && \text{(Equation 4)} \\
&\geq \sum_{\psi \in L'} r\alpha \cdot (i(\psi) + N - |s(\psi)|) + \sum_{\psi \in L'} \sum_{i'=i(\psi)+1}^{|s(\psi)|} \Delta\left(\mathsf{TC}(s(\psi)_{\leq i'}), \tau_{i'}\right) \\
& && \text{(Equation 5)} \\
&\geq r\alpha \cdot \sum_{i' \in [N]} |S'| - |\mathsf{pre}_{i'}(S')| + \sum_{i' \in [N]} \sum_{s' \in \mathsf{pre}_{i'}(S')} \Delta(\mathsf{TC}(s'), \tau_{i'}) \\
& && \text{(Claim 5.6 and } |L'| = |S'|) \\
&\geq r\alpha \cdot \sum_{i' \in [N]} |S'| - |\mathsf{pre}_{i'}(S')| + r \cdot \left(1 - \frac{\epsilon}{10}\right) \cdot \sum_{i' \in [N]} \mathsf{sep}(|\mathsf{pre}_{i'}(S')|) \\
& && \text{(Equation 6)} \\
&\geq r \cdot \left(1 - \frac{\epsilon}{2}\right) \cdot \left(\sum_{i' \in [N]} \frac{1}{2} \cdot (|S'| - |\mathsf{pre}_{i'}(S')|) + \mathsf{sep}(|\mathsf{pre}_{i'}(S')|) \right) \\
& && \text{(Equation 3)} \\
&\geq rN \cdot \left(1 - \frac{\epsilon}{2}\right) \cdot \mathsf{sep}(|S'|), && (2 \leq |L'| = |S'| \leq 4)
\end{aligned}$$

a contradiction.

□

Proof of item 1 of Theorem 5.3. Note that $|L| \leq 3$ as otherwise, there exists a subset $L' \subseteq L$ such that $|L'| = 4$. Applying Lemma 5.5 on L', we get

$$rN \cdot \left(1 - \frac{\epsilon}{2}\right) \cdot \frac{5}{4} \leq \sum_{\psi \in L'} \mathsf{d}(\psi) < 4\left(\theta + \frac{\epsilon}{2}\right) T',$$

contradiction due to Equation 3. Also, for $s \neq s' \in L$, by applying Lemma 5.5 on the set $\{s, s'\}$ and using Equation 3, we get

$$\mathsf{d}(s) + \mathsf{d}(s') \geq \left(\frac{1}{4} - \frac{\epsilon}{8}\right) \cdot T'.$$

□

5.2 Proof of item 2 of Theorem 5.3

We now turn our attention to proving item 2 of Theorem 5.3. We will need some additional notation for this part of the proof. Observe that Algorithm 2 loops over $i \in [N]$. For $i \in [N]$, we define $m^{A,(i)}$ to be the number of times Alice executes the loop in Line 15 in iteration i. For $i \in [N]$, $i' \in [m^{A,(i)}]$, and a variable var, we shall use $var^{A,(i,i')}$ to denote the value of the

variable var after i' iterations of Line 15 in iteration i of Alice's execution and $var^{A,(i,0)}$ to denote the value of var before Line 15 in iteration i of Alice's execution.

Similarly, we define $var^{A,(i)}$ to denote the value of the variable var after iteration i of Alice's execution and $var^{A,(0)}$ to denote the value of var at the beginning of Alice's execution. We define $var^{A} = var^{A,(N)}$ as in the foregoing section. We also define $m^{A,(1)} = 1$, $s^{A,(1,1)} = \varepsilon$, $var^{A,(1,0)} = var^{A,(0)}$, and $var^{A,(1,1)} = var^{A,(1)}$ for all variables var for convenience.

We define the set $\mathcal{T}_s^A$ to be the set of all pairs (i, i') such that $i \in [N]$, $i' \in [m^{A,(i)}]$, and Line 19 is executed by Alice in the iteration (i, i'). We shall assume for convenience that $(1, 1) \in \mathcal{T}_s^A$. We define analogous notation for Bob by replacing the superscript A with B everywhere. As in the previous section, we sometimes omit the superscript when it is A. Finally, throughout this proof, we make implicit use of the following observation.

Observation 5.7. *For all $i \in [N]$ and $i' \in [m^{(i)}]$, if $t = s^{(i,i')}$ or $|t| < i - 1$, we have $\mathsf{sent}^{(i,i')}(t) = \mathsf{sent}(t)$.*

5.2.1 Analyzing Algorithm 3

Observation 5.8 (Generalization of Observation 5.4)**.** *Let $i_1, i_2 \in [N]$, $i_1' \in [m^{(i_1)}]$, and $i_2' \in [m^{(i_2)}]$ be given. If $(i_1, i_1') \leq (i_2, i_2')$, it holds that $\mathcal{E}^{(i_1,i_1')} \preceq \mathcal{E}^{(i_2,i_2')}$.*

Lemma 5.9. *For all $(i, i') \in \mathcal{T}_s$, we have $\mathcal{E}^{(i,i')}_{|\mathcal{E}^{(i,i')}|} = \mathsf{sent}(s^{(i,i')})$. Furthermore, if $(i, i') \neq (1, 1)$, there exists $p' > 0$ such that $\mathcal{E}^{(i,i')}_{p'} = \mathsf{sent}(s^{(i,i')})_{\leq |\mathsf{sent}(s^{(i,i')})|-2}$.*

Proof. Proof by induction on i. The base case $(i, i') = (1, 1)$ is trivial. We show that the statement holds for $(i, i') \neq (1, 1)$ by assuming it holds for smaller values. We first show the following claim that implies the furthermore part due to Observation 5.8 and Line 19.

Claim 5.10. *There exists $p' > 0$ such that $\mathcal{E}^{(i,i'-1)}_{p'} = \mathsf{sent}(s'^{(i,i')})$.*

Proof. This is because, by definition, we have $\mathsf{sent}(s'^{(i,i')}) \neq \perp$, and therefore, by Line 12 and Line 19, we have $(i_1, i_1') < (i, i')$ such that $s'^{(i,i')} = s^{(i_1,i_1')}$. By the induction hypothesis and Observation 5.8, it follows that $\mathcal{E}^{(i,i'-1)}_{|\mathcal{E}^{(i_1,i_1')}|} = \mathsf{sent}(s'^{(i,i')})$, as required. □

To continue, we observe that

$$E^{(i,i')} = E^{(i,i'-1)} \| \left(|\mathcal{E}^{(i,i')}| - p^{(i,i')}, \mathbb{1}\left(\mathcal{E}^{(i,i'-1)}_{p^{(i,i')}} \neq \mathsf{sent}(s'^{(i,i')})\right), B^{(i,i')}(b^{(i,i')}) \right),$$

implying, due to Algorithm 3 and Claim 5.10 (that implies $p^{(i,i')} > 0$) that:

$$\begin{aligned}
\mathcal{E}^{(i,i')} &= \mathcal{E}^{(i,i'-1)} \| \left(\mathcal{E}^{(i,i'-1)}_{p^{(i,i')}} \left[1 : \left| \mathcal{E}^{(i,i'-1)}_{p^{(i,i')}} \right| - 2 \cdot \mathbb{1}\left(\mathcal{E}^{(i,i'-1)}_{p^{(i,i')}} \neq \mathsf{sent}(s'^{(i,i')})\right) \right] \| B^{(i,i')}(b^{(i,i')}) \right) \\
&= \mathcal{E}^{(i,i'-1)} \| \left(\mathsf{sent}(s'^{(i,i')}) \| B^{(i,i')}(b^{(i,i')}) \right) \\
&= \mathcal{E}^{(i,i'-1)} \| \mathsf{sent}(s^{(i,i')}),
\end{aligned}$$

and the lemma follows.

□

Corollary 5.11. *For all $p \in [|\mathcal{E}|]$ and all $j \in [|\mathcal{E}_p|]$ such that $|\mathcal{E}_p| - j$ is even, we have that*

$$\mathcal{E}_{p,j} = f^A\left(x^A, \mathcal{E}_{p,<j}\right).$$

Proof. We first show the claim for all $p \in [|\mathcal{E}|]$ and $j = |\mathcal{E}_p|$. Fix $p \in [|\mathcal{E}|]$. As $|\mathcal{E}|$ increases by at most 1 in any iteration (i, i'), we have that there exists (i, i') such that $p = |\mathcal{E}^{(i,i')}|$. By Observation 5.8 and Lemma 5.9, we get that

$$\mathcal{E}_p = \mathcal{E}_p^{(i,i')} = \mathcal{E}^{(i,i')}_{|\mathcal{E}^{(i,i')}|} = \mathsf{sent}(s^{(i,i')}).$$

The claim now follows from Line 12 and Line 19. To show the claim for other values of j, we argue that for all $p \in [|\mathcal{E}|]$ and all $j < |\mathcal{E}_p|$ such that $|\mathcal{E}_p| - j$ is even, we have $p' \in [|\mathcal{E}|]$ satisfying $\mathcal{E}_{p'} = \mathcal{E}_{p,\leq j}$ and the result follows. Suppose not and let (p, j) be the lexicographically smallest such that the statement is not true. By our choice of j, we have $p'' \in [|\mathcal{E}|]$ satisfying $\mathcal{E}_{p''} = \mathcal{E}_{p,\leq j+2}$.

As $|\mathcal{E}|$ increases by at most 1 in any iteration (i, i'), we have that there exists (i_1, i'_1) such that $p'' = |\mathcal{E}^{(i_1,i'_1)}|$. By Observation 5.8 and the furthermore part of Lemma 5.9, we get p' such that

$$\mathcal{E}_{p'} = \mathcal{E}_{p'}^{(i_1,i'_1)} = \mathcal{E}^{(i_1,i'_1)}_{p'',\leq \left|\mathcal{E}_{p''}^{(i_1,i'_1)}\right|-2} = \mathcal{E}_{p'',\leq |\mathcal{E}_{p''}|-2} = \mathcal{E}_{p,\leq j},$$

a contradiction. □

5.2.2 Some Helper Lemmas Concerning Algorithm 2

Lemma 5.12. *For all $i \in \{0\} \cup [N]$, we have $i \leq |\sigma^{(i)}|$.*

Proof. Proof by induction on i. The base case $i = 0$ is trivial. For $i > 0$, we either have $i - 1 < |\sigma^{(i-1)}|$ in which case the lemma follows because $|\sigma^{(i-1)}| \leq |\sigma^{(i)}|$, or we have $i - 1 = |\sigma^{(i-1)}| < i$ by the induction hypothesis. When this happens, then Line 26 is executed in iteration i and we get

$$i = i - 1 + 1 \leq |\sigma^{(i-1)}| + |\mathsf{Enc}(E^{(i)})| = |\sigma^{(i)}|,$$

where the inequality follows from item 2 of Fact 5.2. □

Lemma 5.13. *Let $t' \preceq t \in \Sigma^*$ be such that $\mathsf{sent}(t), \mathsf{sent}(t') \neq \perp$. We have*

1. *$\mathsf{sent}(t') \preceq \mathsf{sent}(t)$ with equality only if $t = t'$.*
2. *There exists p such that $\mathtt{FIND}(\mathsf{Dec}^*(t))_p = \mathsf{sent}(t)_{<|\mathsf{sent}(t)|}$.*
3. *If $t \preceq \sigma^B$, then $\mathsf{sent}(t) \preceq \Pi(x^A, x^B)$.*

Proof. Proof by induction on $|t|$. The statement clearly holds when $t = \varepsilon$. For $l > 0$, we show the claim assuming that $|t| = l$ given it holds when $|t| < l$. Without loss of generality, we can assume that $t' \neq t$. As $\mathsf{sent}(t) \neq \perp$, we have by Line 19 that there exists $(i, i') \in \mathcal{T}_s$ such that $t = s^{(i,i')}$. We have

$$\mathsf{sent}(t) = \mathsf{sent}(s'^{(i,i')}) \| B^{(i,i')}(b^{(i,i')}) = \mathsf{sent}(s'^{(i,i')}) \| b^{(i,i')} \| f^A(x^A, \mathsf{sent}(s'^{(i,i')}) \| b^{(i,i')}). \tag{7}$$

We now show each part in turn:

1. For the first part, we note that our assumption that $t \neq t'$ and our choice of $s'^{(i,i')}$ implies that $t' \preceq s'^{(i,i')}$ whence we can use our induction hypothesis and Equation 7 to get
$$\mathsf{sent}(t') \preceq \mathsf{sent}(s'^{(i,i')}) \prec \mathsf{sent}(t).$$

2. Straightforward by definition of $b^{(i,i')}$ in Line 18.

3. For this part, we observe that $t \preceq \sigma^B$ implies $s'^{(i,i')} \preceq \sigma^B$ by our choice of $s'^{(i,i')}$. It follows that $\mathsf{sent}(s'^{(i,i')}) \preceq \Pi(x^A, x^B)$. Now, by the induction hypothesis and Equation 7, it is sufficient to show that $b^{(i,i')} = f^B(x^B, \mathsf{sent}(s'^{(i,i')}))$ in order to finish the proof.

   Using Bob's version of Corollary 5.11, it is sufficient to show that there exists $p \in [|\mathcal{E}^B|]$ such that $\mathcal{E}^B_p = \mathsf{sent}(s'^{(i,i')}) \| b^{(i,i')}$. In fact, by our choice of $b^{(i,i')}$ in Line 18, it is sufficient to show that $\tilde{\mathcal{E}}^{(i,i')} \preceq \mathcal{E}^B$. This follows because, by definition of Dec^* and Observation 5.4, we have
$$\tilde{\mathcal{E}}^{(i,i')} = \mathtt{FIND}\left(\mathsf{Dec}^*(s^{(i,i')})\right) = \mathtt{FIND}\left(\mathsf{Dec}^*(t)\right) \preceq \mathtt{FIND}\left(\mathsf{Dec}^*\left(\sigma^B\right)\right) = \mathcal{E}^B. \qquad \square$$

Lemma 5.14. *It holds that*

$$\sum_{(i,i') \in \mathcal{T}_s} |\mathcal{E}^{(i,i')}| - p^{(i,i')} \leq 3NLK.$$

Proof. For $i \in [N]$ and $i' \in [m^{(i)}]$, define the set

$$\mathcal{S}_{s,\leq(i,i')} = \{s^{(i_1,i'_1)} \mid (i_1, i'_1) \in \mathcal{T}_s, (i_1, i'_1) \leq (i, i')\}.$$

We claim that:

Claim 5.15. *For all $(i, i') \neq (1, 1) \in \mathcal{T}_s$, we have*

$$|\mathcal{E}^{(i,i')}| - p^{(i,i')} \leq 2K \cdot \left(\left|\mathsf{pre}\left(\mathcal{S}_{s,\leq(i,i')}\right)\right| - \left|\mathsf{pre}\left(\mathcal{S}_{s,\leq(i,i'-1)}\right)\right|\right).$$

Proof. We stat by observing that, by definition of $\mathsf{pre}(\cdot)$, we have that

$$\left|\mathsf{pre}\left(\mathcal{S}_{s,\leq(i,i')}\right)\right| - \left|\mathsf{pre}\left(\mathcal{S}_{s,\leq(i,i'-1)}\right)\right| = |s^{(i,i')}| - k, \tag{8}$$

where k is the largest (possibly 0) such that there exists $s'' \in \mathcal{S}_{s,\leq(i,i'-1)}$ such that $s^{(i,i')}_{\leq k} = s''_{\leq k}$. As $(i,i') \neq (1,1)$, we have that $s'^{(i,i')}$ is well defined and $\mathsf{sent}(s'^{(i,i')}) \neq \perp$ implying by Line 12 and Line 19 that there exists $(i_1,i'_1) < (i,i')$ such that $s^{(i_1,i'_1)} = s'^{(i,i')}$ implying that $s'^{(i,i')} \in \mathcal{S}_{s,\leq(i,i'-1)}$. It follows that $|s'^{(i,i')}| \leq k$.

Next, we observe that either $|s'^{(i,i')}| = k$ or, by our choice of $s'^{(i,i')}$, there exists $(i_2,i'_2) \leq (i,i'-1)$ such that

$$k \leq |s^{(i_2,i'_2)}| \qquad \text{and} \qquad s'^{(i,i')} \prec s^{(i_2,i'_2)} \not\preceq s^{(i,i')}.$$

We let (i_2,i'_2) be the smallest such pair. By our choice of (i_2,i'_2), we get that $s'^{(i,i')} = s'^{(i_2,i'_2)}$.

Therefore, in either case, there exists $(i_*,i'_*) \leq (i,i'-1)$ such that $k \leq |s^{(i_*,i'_*)}|$ and $s'^{(i,i')} \in \{s'^{(i_*,i'_*)}, s^{(i_*,i'_*)}\}$ implying that

$$\mathsf{sent}\left(s^{(i_*,i'_*)}\right) \in \left\{\mathsf{sent}\left(s'^{(i,i')}\right), \mathsf{sent}\left(s'^{(i,i')}\right) \| B^{(i,i')}(0), \mathsf{sent}\left(s'^{(i,i')}\right) \| B^{(i,i')}(1)\right\}.$$

We derive:

$$\begin{aligned}
|\mathcal{E}^{(i,i')}| - p^{(i,i')} &\leq |\mathcal{E}^{(i,i')}| - |\mathcal{E}^{(i_*,i'_*)}| && \text{(Observation 5.8 and Lemma 5.9)}\\
&\leq K + K \cdot (i - i_*) && \text{(Line 13)}\\
&\leq K + K \cdot \left(|s^{(i,i')}| - |s^{(i_*,i'_*)}|\right)\\
&\leq K + K \cdot \left(|s^{(i,i')}| - k\right) && (k \leq |s^{(i_*,i'_*)}|)\\
&\leq K + K \cdot \left(\left|\mathsf{pre}\left(\mathcal{S}_{s,\leq(i,i')}\right)\right| - \left|\mathsf{pre}\left(\mathcal{S}_{s,\leq(i,i'-1)}\right)\right|\right) && \text{(Equation 8)}\\
&\leq 2K \cdot \left(\left|\mathsf{pre}\left(\mathcal{S}_{s,\leq(i,i')}\right)\right| - \left|\mathsf{pre}\left(\mathcal{S}_{s,\leq(i,i'-1)}\right)\right|\right). && \square
\end{aligned}$$

It follows that:

$$\begin{aligned}
\sum_{(i,i')\in\mathcal{T}_s} |\mathcal{E}^{(i,i')}| - p^{(i,i')} &\leq 1 + \sum_{(i,i')\neq(1,1)\in\mathcal{T}_s} 2K \cdot \left(\left|\mathsf{pre}\left(\mathcal{S}_{s,\leq(i,i')}\right)\right| - \left|\mathsf{pre}\left(\mathcal{S}_{s,\leq(i,i'-1)}\right)\right|\right)\\
&\leq 1 + 2K \cdot \left|\mathsf{pre}\left(\mathsf{near}^{\mathsf{TC}}_{\alpha}(\tau)\right)\right|\\
&\leq 3NLK. && \text{(Definition 4.4)}
\end{aligned}$$

$\square$

5.2.3 Analyzing Algorithm 2

In order to continue our analysis of Algorithm 2, we need to consider both Alice's and Bob's version of Algorithm 2. Define the sets:

$$\begin{aligned}
\mathcal{A} &= \{i \in [N-1] \mid \max\left(|\sigma^{A,(i-1)}|, |\sigma^{B,(i-1)}|\right) < i \wedge \max\left(|\sigma^{A,(i)}|, |\sigma^{B,(i)}|\right) < i+1\}.\\
\mathcal{B} &= \{i \in [N] \mid |\mathsf{near}^{\mathsf{TC}}_{\alpha,i}(\tau^A)| \leq K \wedge |\mathsf{near}^{\mathsf{TC}}_{\alpha,i}(\tau^B)| \leq K\}.\\
\mathcal{C} &= \{i \in [N] \mid \sigma^A_{\leq i} \in \mathsf{near}^{\mathsf{TC}}_{\alpha,i}(\tau^B) \wedge \sigma^B_{\leq i} \in \mathsf{near}^{\mathsf{TC}}_{\alpha,i}(\tau^A)\} \cup \{0\}.\\
\mathcal{C}^* &= \mathcal{A} \cap \mathcal{B} \cap \mathcal{C}.
\end{aligned}$$

We first show that all of these sets are large.

Lemma 5.16. $|\mathcal{A}| \geq N \cdot \left(1 - \frac{\epsilon}{5}\right)$.

Proof. For $C \in \{A, B\}$, define the set $\mathcal{A}'^C = \{i \in [N] \mid |\sigma^{C,(i-1)}| < i\}$. We show that $|\mathcal{A}'^C| \geq N \cdot \left(1 - \frac{\epsilon}{20}\right)$ for all $C \in \{A, B\}$ and the lemma follows. We restrict ourselves to $C = A$ as the case case $C = B$ is similar. We have:

$$
\begin{aligned}
N &\leq |\sigma| && \text{(Lemma 5.12)} \\
&\leq \sum_{i \in \mathcal{A}'^A} |\mathsf{Enc}(E^{(i)})| && \text{(Line 26)} \\
&\leq |\mathcal{A}'^A| + \sum_{i \in \mathcal{A}'^A} \left\lfloor \frac{10}{\log_2(|\Sigma|)} \cdot \sum_{j \in [|E^{(i)}|]} \log_2(E^{(i)}_{j,1} + 1) \right\rfloor . && \text{(Fact 5.2, item 1)}
\end{aligned}
$$

To continue, we observe from Equation 3 that, either $\log_2(|\Sigma|) > 10 \cdot \sum_{j \in [|E^{(i)}|]} \log_2(E^{(i)}_{j,1} + 1)$ or $\sum_{j \in [|E^{(i)}|]} E^{(i)}_{j,1} + 1 \geq \frac{1}{\epsilon^{31}} \implies \sum_{i' : (i,i') \in \mathcal{T}_s} |\mathcal{E}^{(i,i')}| - p^{(i,i')} > \frac{1}{\epsilon^{30}}$. Let $I \subseteq [N]$ be the set of all i for which the latter holds. Also, recall the identity $\frac{\log_2(x)}{\log_2(y)} \leq 1 + \frac{x}{y}$ for all $y \geq 5$ and $x > 0$[9]. We get:

$$
\begin{aligned}
N &\leq |\mathcal{A}'^A| + \sum_{i \in \mathcal{A}'^A \cap I} \frac{10}{\log_2(|\Sigma|)} \cdot \sum_{j \in [|E^{(i)}|]} \log_2(E^{(i)}_{j,1} + 1) \\
&\leq |\mathcal{A}'^A| + \sum_{i \in \mathcal{A}'^A \cap I} 10K + 10 \cdot \sum_{j \in [|E^{(i)}|]} \frac{E^{(i)}_{j,1} + 1}{|\Sigma|} \\
&\leq |\mathcal{A}'^A| + 10K \cdot |I| + 10 \cdot \sum_{i \in \mathcal{A}'^A} \sum_{i' : (i,i') \in \mathcal{T}_s} \frac{1 + |\mathcal{E}^{(i,i')}| - p^{(i,i')}}{|\Sigma|} \\
&\leq |\mathcal{A}'^A| + 10K \cdot \sum_{(i,i') \in \mathcal{T}_s} \epsilon^{30} \cdot \left(|\mathcal{E}^{(i,i')}| - p^{(i,i')}\right) + \frac{10NK}{|\Sigma|} + 10 \cdot \sum_{(i,i') \in \mathcal{T}_s} \frac{|\mathcal{E}^{(i,i')}| - p^{(i,i')}}{|\Sigma|} \\
& && \text{(Definition of } I \text{ and Markov)} \\
&\leq |\mathcal{A}'^A| + \epsilon^{30} \cdot 50NLK^2 + \frac{50NLK}{|\Sigma|} && \text{(Lemma 5.14)} \\
&\leq |\mathcal{A}'^A| + \frac{\epsilon N}{20}. && \text{(Equation 3)}
\end{aligned}
$$

□

Lemma 5.17. $|\mathcal{B}| \geq N \cdot \left(1 - \frac{\epsilon}{10}\right)$.

Proof. For $C \in \{A, B\}$, define the set $\mathcal{B}'^C = \{i \in [N] \mid |\mathsf{near}^{\mathsf{TC}}_{\alpha,i}(\tau^C)| \leq K\}$. We show that

[9] To see this indentity, note that if $y \geq 5, x$, then $\frac{\log_2(x)}{\log_2(y)} \leq 1 < 1 + \frac{x}{y}$ and if $x > y \geq 5$, then $\frac{\log_2(x)}{x} \leq \frac{\log_2(y)}{y} \implies \frac{\log_2(x)}{\log_2(y)} \leq \frac{x}{y} < 1 + \frac{x}{y}$.

$|\mathcal{B}'^C| \geq N \cdot \left(1 - \frac{\epsilon}{20}\right)$ for all $C \in \{A, B\}$ and the lemma follows. We restrict ourselves to $C = A$ as the case $C = B$ is similar.

Proof by contradiction. If $|\mathcal{B}'^A| < N(1-\epsilon)$, we get:

$$\frac{\epsilon NK}{20} < \sum_{i \in [N]} |\mathsf{near}^{\mathsf{TC}}_{\alpha,i}(\tau^A)| \leq |\mathsf{near}^{\mathsf{TC}}_{\alpha}(\tau^A)| \leq |\mathsf{pre}(\mathsf{near}^{\mathsf{TC}}_{\alpha}(\tau^A))| \leq NL \leq \frac{\epsilon NK}{100},$$

by Definition 4.4 and Equation 3. This is a clear contradiction. □

Lemma 5.18. *For all $0 \leq i_1 \leq i_2$, if $i_1 \in \mathcal{C}$, it holds that*

$$r\alpha \cdot |(i_1, i_2] \setminus \mathcal{C}| \leq \mathsf{corr}_{\Pi', \mathsf{Adv}', (2ri_1 : 2ri_2]}(x^A, x^B)$$

Proof. Proof by induction on $i_2 - i_1$. The base case $i_1 = i_2$ is trivial. For $l > 0$, we prove the claim for $i_1 < i_2 = i_1 + l$ assuming that it is true for all $0 \leq i_1 \leq i_2$ such that $i_2 - i_1 < l$. If $i_2 \in \mathcal{C}$, we have:

$$r\alpha \cdot |(i_1, i_2] \setminus \mathcal{C}| = r\alpha \cdot |(i_1, i_2) \setminus \mathcal{C}| \leq \mathsf{corr}_{\Pi', \mathsf{Adv}', (2ri_1 : 2ri_2)}(x^A, x^B) \leq \mathsf{corr}_{\Pi', \mathsf{Adv}', (2ri_1 : 2ri_2]}(x^A, x^B).$$

We now assume that $i_2 \notin \mathcal{C}$. By definition of $\mathcal{C}$, this means that either $\sigma^A_{\leq i_2} \in \mathsf{near}^{\mathsf{TC}}_{\alpha, i_2}(\tau^B)$ or $\sigma^B_{\leq i_2} \in \mathsf{near}^{\mathsf{TC}}_{\alpha, i_2}(\tau^A)$. Without loss of generality, we assume the former. Using the definition of near, we get that

$$\delta_{\mathsf{suf}}(\overline{\mathsf{TC}}(\sigma^A_{\leq i_2}), \tau^B_{\leq i_2}) \geq \alpha.$$

This implies, using Definition 3.2, that there is an $i_3 < i_2$ such that $\mathsf{corr}_{\Pi', \mathsf{Adv}', (2ri_3 : 2ri_2]}(x^A, x^B) \geq r\alpha \cdot (i_2 - i_3)$. We let i_3 denote the largest such value. We claim that $i_1 \leq i_3$. Suppose not. Then, we have

$$\begin{aligned}
\mathsf{corr}_{\Pi', \mathsf{Adv}', (2ri_3 : 2ri_2]}(x^A, x^B) &= \mathsf{corr}_{\Pi', \mathsf{Adv}', (2ri_3 : 2ri_1]}(x^A, x^B) + \mathsf{corr}_{\Pi', \mathsf{Adv}', (2ri_1 : 2ri_2]}(x^A, x^B) \\
&< \mathsf{corr}_{\Pi', \mathsf{Adv}', (2ri_3 : 2ri_1]}(x^A, x^B) + r\alpha \cdot (i_2 - i_1) \qquad \text{(Choice of } i_3\text{)} \\
&\leq \delta_{\mathsf{suf}}(\overline{\mathsf{TC}}(\sigma^A_{\leq i_1}), \tau^B_{\leq i_1}) \cdot r \cdot (i_1 - i_3) + r\alpha \cdot (i_2 - i_1) \\
&\leq r\alpha \cdot (i_1 - i_3) + r\alpha \cdot (i_2 - i_1) \qquad (i_1 \in \mathcal{C}) \\
&\leq r\alpha \cdot (i_2 - i_3),
\end{aligned}$$

a contradiction. Using the induction hypothesis, we get:

$$\begin{aligned}
r\alpha \cdot |(i_1, i_2] \setminus \mathcal{C}| &\leq r\alpha \cdot |(i_1, i_3] \setminus \mathcal{C}| + r\alpha \cdot (i_2 - i_3) \\
&\leq \mathsf{corr}_{\Pi', \mathsf{Adv}', (2ri_1, 2ri_3]}(x^A, x^B) + \mathsf{corr}_{\Pi', \mathsf{Adv}', (2ri_3 : 2ri_2]}(x^A, x^B) \\
&\leq \mathsf{corr}_{\Pi', \mathsf{Adv}', (2ri_1, 2ri_2]}(x^A, x^B).
\end{aligned}$$

□

Lemma 5.19. *If $\mathsf{corr}_{\Pi', \mathsf{Adv}'}(x^A, x^B) < \theta T'$, then $|\mathcal{C}^*| \geq N \cdot \left(\frac{3}{8} - 7\epsilon\right) > T$.*

Proof. We show this by upper bounding $|\overline{\mathcal{C}^*}|$. We have:

$$|\overline{\mathcal{C}^*}| \leq |\overline{\mathcal{A}}| + |\overline{\mathcal{B}}| + |\overline{\mathcal{C}}|$$

$$\leq 7\epsilon N + |\overline{\mathcal{C}}| \qquad \text{(Lemma 5.16, Lemma 5.17)}$$
$$\leq 7\epsilon N + \frac{1}{r\alpha} \cdot \mathsf{corr}_{\Pi',\mathsf{Adv}'}(x^A, x^B) \qquad \text{(Lemma 5.18)}$$
$$< 7\epsilon N + \frac{2\theta N}{\alpha} \leq 7\epsilon N + \frac{5}{8}N. \qquad \text{(Equation 3)}$$

This yields $|\mathcal{C}^*| > N \cdot \left(\frac{3}{8} - 7\epsilon\right) > T$ by Equation 3.

□

5.2.4 Finishing the Proof of item 2 of Theorem 5.3

Proof of item 2 of Theorem 5.3. We simply show that $\mathsf{d}(\Pi(x^A, x^B)) \leq \mathsf{corr}_{\Pi',\mathsf{Adv}'}(x^A, x^B) + \frac{\epsilon T'}{2}$ and lemma follows by our definition of L. To see why $\mathsf{d}(\Pi(x^A, x^B)) \leq \mathsf{corr}_{\Pi',\mathsf{Adv}'}(x^A, x^B) + \frac{\epsilon T'}{2}$, define $i_1 = \max(\mathcal{C}^*)$ to be the largest element in $\mathcal{C}^*$ and i_2 to be the unique element in $\mathcal{C}^*$ such that $|[i_2] \cap \mathcal{C}^*| = T$. Note that i_2 is well defined due Lemma 5.19 and $i_2 \leq i_1$.

As $i_1 \in \mathcal{C}^*$, we have that $\sigma^B_{\leq i_1} \in \mathsf{near}^{\mathsf{TC}}_{\alpha,i_1}(\tau^A)$. We claim that $\Pi(x^A, x^B) \in \Psi$ is unique such that $\exists p : \mathtt{FIND}\left(\mathsf{Dec}^*(\sigma^B_{\leq i_2})\right)_p = \Pi(x^A, x^B)$. We show this claim later but assuming it for now we get that

$$\begin{aligned}
\mathsf{d}(\Pi(x^A, x^B)) &\leq r\alpha \cdot (i_2 + N - i_1) + \sum_{i'=i_2+1}^{i_1} \Delta\left(\mathsf{TC}(\sigma^B_{\leq i'}), \tau^A_{i'}\right) \\
&\leq r\alpha \cdot (|(0 : i_2] \setminus \mathcal{C}^*| + T + |(i_1 : N] \setminus \mathcal{C}^*|) + \mathsf{corr}_{\Pi',\mathsf{Adv}',(2ri_2:2ri_1]}(x^A, x^B) \\
&\qquad \text{(Definition of } i_1, i_2, \mathsf{corr}) \\
&\leq \frac{\epsilon T'}{2} + r\alpha \cdot (|(0 : i_2] \setminus \mathcal{C}| + |(i_1 : N] \setminus \mathcal{C}|) + \mathsf{corr}_{\Pi',\mathsf{Adv}',(2ri_2:2ri_1]}(x^A, x^B) \\
&\qquad \text{(Equation 3, Lemma 5.16, Lemma 5.17)} \\
&\leq \frac{\epsilon T'}{2} + \mathsf{corr}_{\Pi',\mathsf{Adv}'}(x^A, x^B). \qquad \text{(Lemma 5.18)}
\end{aligned}$$

It remains to show the claim. We first show that if any $\psi \in \Psi$ is such that $\exists p : \mathtt{FIND}\left(\mathsf{Dec}^*(\sigma^B_{\leq i_2})\right)_p = \psi$, then $\psi = \Pi(x^A, x^B)$ and then show that indeed $\exists p : \mathtt{FIND}\left(\mathsf{Dec}^*(\sigma^B_{\leq i_2})\right)_p = \Pi(x^A, x^B)$. For the first part, observe by definition of Dec^* and Observation 5.4, we have that if $\exists p : \mathtt{FIND}\left(\mathsf{Dec}^*(\sigma^B_{\leq i_2})\right)_p = \psi$, then, $\exists p : \mathcal{E}^B_p = \psi$. Now, using Corollary 5.11, we get that $\psi \in \Psi^B$. However, if $\psi \in \Psi^A$ and $\psi \in \Psi^B$, then $\psi = \Pi(x^A, x^B)$ and we are done.

We now show that $\exists p : \mathtt{FIND}\left(\mathsf{Dec}^*(\sigma^B_{\leq i_2})\right)_p = \Pi(x^A, x^B)$ by showing $\exists j \leq i_2 : \mathsf{sent}(\sigma^B_{\leq j}) = \Pi(x^A, x^B)_{\leq 2T+1}$ and applying Lemma 5.13 and Observation 5.4. We show this inductively in the following lemma.

Lemma 5.20. *For all $i \in \{0\} \cup \mathcal{C}^*$, there exists $0 \leq j^A, j^B \leq i$ such that*

$$|\mathsf{sent}^A(\sigma^B_{\leq j^A})| = 2 \cdot |[i] \cap \mathcal{C}^*| + 1,$$

$$|\mathsf{sent}^B(\sigma^A_{\leq j^B})| = 2 \cdot |[i] \cap \mathcal{C}^*|.$$

Proof. Proof by induction on i. The base case $i = 0$ is trivial. We prove the claim for $i > 0$ assuming that it is true for values smaller than i. Let $i_1 < i$ be the largest such that $i_1 \in \{0\} \cup \mathcal{C}^*$. We apply the induction hypothesis on i_1 to get $0 \leq j_1^A, j_1^B \leq i_1$ such that

$$|\mathsf{sent}^A(\sigma^B_{\leq j_1^A})| = 2 \cdot |[i] \cap \mathcal{C}^*| - 1,$$
$$|\mathsf{sent}^B(\sigma^A_{\leq j_1^B})| = 2 \cdot |[i] \cap \mathcal{C}^*| - 2.$$

by our choice of i_1. By Lemma 5.13, we get:

$$\begin{aligned} \mathsf{sent}^A(\sigma^B_{\leq j_1^A}) &= \Pi(x^A, x^B)_{\leq 2\cdot|[i]\cap\mathcal{C}^*|-1}, \\ \mathsf{sent}^B(\sigma^A_{\leq j_1^B}) &= \Pi(x^A, x^B)_{\leq 2\cdot|[i]\cap\mathcal{C}^*|-2}. \end{aligned} \tag{9}$$

Let us now consider iteration i. As $i > 0$, we have $i \in \mathcal{C}^* = \mathcal{A} \cap \mathcal{B} \cap \mathcal{C}$. By definition of $\mathcal{B}$, we have $|\mathsf{near}^{\mathsf{TC}}_{\alpha,i}(\tau^A)| \leq K$ and $|\mathsf{near}^{\mathsf{TC}}_{\alpha,i}(\tau^B)| \leq K$, which together with the fact that $i \in \mathcal{A}$ imply that $m^{A,(i+1)}, m^{B,(i)} > 0$. Now, we use the definition of $\mathcal{C}$ to get $i'^A \in [m^{A,(i+1)}]$ and $i'^B \in [m^{B,(i)}]$ such that

$$s^{A,(i+1,i'^A)} = \sigma^B_{\leq i} \qquad \text{and} \qquad s^{B,(i,i'^B)} = \sigma^A_{\leq i}. \tag{10}$$

We show each of the claims in the lemma in turn:

- **Showing that $\exists 0 \leq j^B \leq i : |\mathsf{sent}^B(\sigma^A_{\leq j^B})| = 2 \cdot |[i] \cap \mathcal{C}^*|$:** We show this by contradiction. Assume that

  $$\nexists 0 \leq j^B \leq i : |\mathsf{sent}^B(\sigma^A_{\leq j^B})| = 2 \cdot |[i] \cap \mathcal{C}^*|. \tag{11}$$

  From Equation 9, we know that $\mathsf{sent}^A(\sigma^B_{\leq j_1^A}) = \Pi(x^A, x^B)_{\leq 2\cdot|[i]\cap\mathcal{C}^*|-1}$. Next, we invoke Lemma 5.9 to get that there exists p such that $\mathcal{E}_p^{A,(i_1+1)} = \Pi(x^A, x^B)_{\leq 2\cdot|[i]\cap\mathcal{C}^*|-1}$ or equivalently that $\mathtt{FIND}\left(\mathsf{Dec}^*(\sigma^{A,(i_1+1)})\right)_p = \Pi(x^A, x^B)_{\leq 2\cdot|[i]\cap\mathcal{C}^*|-1}$. As $i \in \mathcal{A}$, we have by Lemma 5.12, that

  $$\mathtt{FIND}\left(\mathsf{Dec}^*(\sigma^A_{\leq i})\right)_p = \Pi(x^A, x^B)_{\leq 2\cdot|[i]\cap\mathcal{C}^*|-1}. \tag{12}$$

  Let us now consider iteration (i, i'^B) for Bob. We know by Equation 10 that $s^{B,(i,i'^B)} = \sigma^A_{\leq i}$. Also, it follows from Equation 9, Equation 11, and Lemma 5.13, that $s'^{B,(i,i'^B)} = \sigma^A_{\leq j_1^B}$. Now, for Equation 11 to hold for $j^B = i$, the condition in Line 18 must evaluate to false in iteration (i, i'^B). However, this contradicts Equation 12 and Corollary 5.11.

- **Showing that $\exists 0 \leq j^A \leq i : |\mathsf{sent}^A(\sigma^B_{\leq j^A})| = 2 \cdot |[i] \cap \mathcal{C}^*| + 1$:**

  We show this by contradiction. Assume that

  $$\nexists 0 \leq j^A \leq i : |\mathsf{sent}^A(\sigma^B_{\leq j^A})| = 2 \cdot |[i] \cap \mathcal{C}^*| + 1. \tag{13}$$

From the previous part and Lemma 5.13, we know that there exists $0 \leq j^B \leq i$ such that $\mathsf{sent}^B(\sigma^A_{\leq j^B}) = \Pi(x^A, x^B)_{\leq 2\cdot|[i]\cap\mathcal{C}^*|}$. Next, we invoke Lemma 5.9 to get that there exists p such that $\mathcal{E}_p^{B,(i)} = \Pi(x^A, x^B)_{\leq 2\cdot|[i]\cap\mathcal{C}^*|}$ or equivalently that $\mathtt{FIND}\left(\mathsf{Dec}^*(\sigma^{B,(i)})\right)_p = \Pi(x^A, x^B)_{\leq 2\cdot|[i]\cap\mathcal{C}^*|}$. As $i \in \mathcal{A}$, we get by Lemma 5.12 that

$$\mathtt{FIND}\left(\mathsf{Dec}^*(\sigma^B_{\leq i})\right)_p = \Pi(x^A, x^B)_{\leq 2\cdot|[i]\cap\mathcal{C}^*|}. \tag{14}$$

Let us now consider iteration $(i+1, i'^A)$ for Alice. We know by Equation 10 that $s^{A,(i+1,i'^A)} = \sigma^B_{\leq i}$. Also, it follows from Equation 9, Equation 13, and Lemma 5.13, that $s'^{A,(i+1,i'^A)} = \sigma^B_{\leq j_1^A}$. Now, for Equation 13 to holds for $j^A = i$, the condition in Line 18 must evaluate to false in iteration $(i+1, i'^A)$. However, this contradicts Equation 14 and Corollary 5.11.

□

□

6 Our Simulation Procedure

We are now ready to state and prove our main result, Theorem 6.1, which is a formalization of Theorem 1.1.

Theorem 6.1. *Let $0 < \epsilon < \frac{1}{20}$ be a parameter and $\Pi = \{T, p, \mathcal{X}^C, \mathcal{Y}^C, f^C, \mathsf{out}^C\}_{C\in\{A,B\}}$ be a protocol in the binary two party communication model as in Subsection 3.1. There is a protocol $\hat{\Pi} = \{\hat{T}, \hat{p}, \hat{\mathcal{X}}^C, \hat{\mathcal{Y}}^C, \hat{f}^C, \widehat{\mathsf{out}}^C\}_{C\in\{A,B\}}$ such that:*

1. *We have $\hat{T} = \frac{39}{32}\cdot n_0(\epsilon/10)\cdot 10^{70}\cdot\frac{T}{\epsilon^{60}}$ where $n_0(\cdot)$ is as promised by Lemma 4.1. We also have $\hat{\mathcal{X}}^C = \mathcal{X}^C$ for all $C \in \{A, B\}$, and $\hat{\mathcal{Y}}^A = \hat{\mathcal{Y}}^B = \{0,1\}^T$.*
2. *For all $x^A \in \mathcal{X}^A$, $x^B \in \mathcal{X}^B$, and all adversaries $\hat{\mathsf{Adv}}$ for $\hat{\Pi}$, we have for all $C \in \{A, B\}$ that*

$$\mathsf{corr}_{\hat{\Pi},\hat{\mathsf{Adv}}}(x^A, x^B) \leq \left(\frac{5}{39} - \epsilon\right)\cdot\hat{T} \implies \widehat{\mathsf{out}}^C_{\hat{\Pi},\hat{\mathsf{Adv}}}(x^A, x^B) = \Pi(x^A, x^B).$$

The rest of this paper has our proof of Theorem 6.1. Fix a parameter $0 < \epsilon < \frac{1}{20}$ and a protocol $\Pi = \{T, p, \mathcal{X}^C, \mathcal{Y}^C, f^C, \mathsf{out}^C\}_{C\in\{A,B\}}$ in the binary two party communication model. At the cost of blowing up the number of rounds in Π by a factor of 2, we can assume that Π is an alternating binary protocol, *i.e.*, Alice transmits in the odd rounds of Π and Bob transmits in the even rounds of Π. Thus, henceforth, we assume Π has $2T$ rounds and $p_{2i-1} = A$ and $p_{2i} = B$ for all $i \in [T]$. We let Π_{List} denote the protocol promised by Theorem 5.1 with the parameters $\epsilon/10$ and Π and define $\hat{N} = \frac{\hat{T}}{39}$ for convenience. Observe that Π_{List} has $32\hat{N}$ rounds.

We shall use ECC to denote the functions $\mathsf{ECC}_{4\hat{N},\epsilon/10}$ promised by Lemma 4.1 and C_n will denote the function promised by Lemma 4.2. The protocol $\hat{\Pi}$ that proves Theorem 6.1 is described in Algorithm 4 (Alice's side) and Algorithm 5 (Bob's side).

Algorithm 4 Alice's side of the simulation protocol

Input: An input $x^A \in \mathcal{X}^A$.
Output: A transcript $\pi \in \{0,1\}^{2T}$.

Phase 1:

Run Π_{List} with input x^A and get output (L, d). Interpret L as a list by ordering the elements in some canonical way.

Phase 2:

Listen for the next $4\hat{N}$ rounds. Let $\sigma \in \{0,1\}^{4\hat{N}}$ be the symbols received.

Compute Output:

For $l \in L$, set $e(l) \leftarrow \min_{L' : |L'| \leq 3 \text{ and } L \cap L' = \{l\}} \Delta\left(\mathsf{ECC}(L'), \sigma\right)$ and let $E(l)$ be the minimizer.
$\pi \leftarrow \arg\min_{l \in L} \mathsf{d}(l) + e(l)$.
Set $\mathsf{p} \in [3]$ to be the position of π in $E(\pi)$.

Phase 3:

For the next $3\hat{N} - \frac{3}{2} \cdot e(\pi)$ rounds, transmit $\mathsf{C}_{\hat{N} - \frac{e(\pi)}{2}}(\mathsf{p})$.
For the remaining $\frac{3}{2} \cdot e(\pi)$ rounds, transmit $\mathsf{C}_{\frac{e(\pi)}{2}}(0)$.

Algorithm 5 Bob's side of the simulation protocol

Input: An input $x^B \in \mathcal{X}^B$.
Output: A transcript $\pi \in \{0,1\}^{2T}$.

Phase 1:

Run Π_{List} with input x^B and get output (L, d). Interpret L as a list by ordering the elements in some canonical way.

Phase 2:

For the next $4\hat{N}$ rounds, transmit $\mathsf{ECC}(L) \in \{0,1\}^{4\hat{N}}$, symbol by symbol.

Phase 3:

Listen in the remaining $3\hat{N}$ rounds. Let $\tau \in \{0,1\}^{3\hat{N}}$ be the symbol received.

Compute Output:

For $\tilde{\mathsf{p}} \in \{0,1,2,3\}$, set $e(\tilde{\mathsf{p}}) \leftarrow \Delta\left(\mathsf{C}_{\hat{N}}(\tilde{\mathsf{p}}), \tau\right)$.
$\mathsf{q} \leftarrow \arg\min_{j \in [3]} \Big(\mathsf{d}(L_j) + \min\Big(e(j), e(0) + 3\hat{N} \cdot \mathbb{1}\left(j \neq \arg\min_{j' \in [3]} \mathsf{d}(L_{j'})\right)\Big)\Big)$.
$\pi \leftarrow L_{\mathsf{q}}$.

6.1 Proof of Theorem 6.1

We now prove Theorem 6.1.

Proof of Theorem 6.1. Observe that item 1 of Theorem 6.1 follows straightforwardly from our definition of $\hat{\Pi}$ in Algorithm 4 and Algorithm 5.

For item 2 of Theorem 6.1, we fix inputs x^A and x^B for Alice and Bob respectively and an adversary $\hat{\mathsf{Adv}}$ for the protocol $\hat{\Pi}$. As our protocol is deterministic, fixing x^A, x^B, and $\hat{\mathsf{Adv}}$ fixes the values of all the variables in Algorithm 4 and Algorithm 5. In our analysis, we shall use var^A (respectively, var^B) to denote the value of variable var in Algorithm 4 (resp. Algorithm 5). We shall drop the superscript if the variable appears in only one of Algorithm 4 and Algorithm 5. Observe that, with this notation, we have for all $C \in \{A, B\}$ that $\widehat{\mathsf{out}}^C_{\hat{\Pi},\hat{\mathsf{Adv}}}(x^A, x^B) = \pi^C$ and in order to show item 2 of Theorem 6.1, we have to show for all $C \in \{A, B\}$ that

$$\mathsf{corr}_{\hat{\Pi},\hat{\mathsf{Adv}}}(x^A, x^B) \leq \left(\frac{5}{39} - \epsilon\right) \cdot \hat{T} \implies \pi^C = \Pi(x^A, x^B).$$

To start with, we use item 2 of Theorem 5.1 to get that $\mathsf{corr}_{\hat{\Pi},\hat{\mathsf{Adv}}}(x^A, x^B) \leq \left(\frac{5}{39} - \epsilon\right) \cdot \hat{T}$ implies $L^A \cap L^B = \{\Pi(x^A, x^B)\}$ and for all $C \in \{A, B\}$ and $s, s' \in L^C$, we have:

$$\mathsf{d}^C(\Pi(x^A, x^B)) \leq \mathsf{corr}_{\hat{\Pi},\hat{\mathsf{Adv}},[32\hat{N}]}(x^A, x^B) + 2\epsilon\hat{N}. \tag{15}$$

$$\mathsf{d}^C(s) + \mathsf{d}^C(s') \geq \left(\frac{1}{4} - \frac{\epsilon}{20}\right) \cdot 32\hat{N}. \tag{16}$$

We divide the rest of this proof into two parts and use the fact that $L^A \cap L^B = \{\Pi(x^A, x^B)\}$ implicitly throughout.

- **Showing $\boldsymbol{\pi^A = \Pi(x^A, x^B)}$:** To start, observe that $e^A(\Pi(x^A, x^B)) \leq \mathsf{corr}_{\Pi',\mathsf{Adv}',(32\hat{N},39\hat{N}]}(x^A, x^B)$. Now, if $\pi^A \neq \Pi(x^A, x^B)$, using the fact that $\pi^A \in L^A$ by definition, we get:

$$\begin{aligned}
&2 \cdot \left(\mathsf{d}^A(\Pi(x^A, x^B)) + e^A(\Pi(x^A, x^B))\right) \\
&\qquad \leq 2 \cdot \mathsf{corr}_{\hat{\Pi},\hat{\mathsf{Adv}}}(x^A, x^B) && \text{(Equation 15)} \\
&\qquad < (10 - 5\epsilon)\hat{N} \\
&\qquad \leq \mathsf{d}^A(\Pi(x^A, x^B)) + \mathsf{d}^A(\pi^A) + \Delta\left(\mathsf{ECC}(E(\Pi(x^A, x^B))), \mathsf{ECC}(E(\pi^A))\right) \\
&\qquad && \text{(Equation 16 and Lemma 4.1)} \\
&\qquad \leq \mathsf{d}^A(\Pi(x^A, x^B)) + \mathsf{d}^A(\pi^A) + e^A(\Pi(x^A, x^B)) + e^A(\pi^A), \\
&\qquad && \text{(Triangle inequality)}
\end{aligned}$$

  a contradiction to the choice of π^A.

- **Showing $\boldsymbol{\pi^B = \Pi(x^A, x^B)}$:**

  We further break this part of the proof into two cases:

– **When $E(\pi^A) = L^B$:** In this case, we get that $L^B_{\mathsf{p}} = \pi^A = \Pi(x^A, x^B)$. Now, if $\pi^B \neq \Pi(x^A, x^B)$ then $\mathsf{q} \neq \mathsf{p}$ and we get:

$$\begin{aligned}
2 \cdot \left(d^B(L^B_{\mathsf{p}}) + e^B(\mathsf{p})\right) &\leq 2 \cdot \left(\mathsf{corr}_{\hat{\Pi},\hat{\mathsf{Adv}},[32\hat{N}]}(x^A, x^B) + e^B(\mathsf{p})\right) \\
&\leq 2 \cdot \left(\mathsf{corr}_{\hat{\Pi},\hat{\mathsf{Adv}},[32\hat{N}]}(x^A, x^B) + \Delta\left(\mathsf{C}_{\hat{N}}(\mathsf{p}), \tau\right)\right) \\
&\qquad\qquad \text{(Definition of } e^B(\cdot)\text{)} \\
&\leq 2 \cdot \Big(\mathsf{corr}_{\hat{\Pi},\hat{\mathsf{Adv}},[32\hat{N}]}(x^A, x^B) \\
&\qquad + \mathsf{corr}_{\hat{\Pi},\hat{\mathsf{Adv}},(36\hat{N}:39\hat{N}]}(x^A, x^B) + e^A(\Pi(x^A, x^B))\Big) \\
&\qquad\qquad \text{(Line 55 and Line 56)} \\
&< (10 - 5\epsilon)\hat{N} \\
&\qquad\qquad (e^A(\Pi(x^A, x^B)) \leq \mathsf{corr}_{\hat{\Pi},\hat{\mathsf{Adv}},(32\hat{N}:36\hat{N}]}(x^A, x^B)) \\
&\leq \mathsf{d}^B(L^B_{\mathsf{p}}) + \mathsf{d}^B(L^B_{\mathsf{q}}) \\
&\qquad + \min\left(\Delta\left(\mathsf{C}_{\hat{N}}(\mathsf{p}), \mathsf{C}_{\hat{N}}(\mathsf{q})\right), \Delta\left(\mathsf{C}_{\hat{N}}(\mathsf{p}), \mathsf{C}_{\hat{N}}(0)\right)\right) \\
&\qquad\qquad \text{(Equation 16 and Lemma 4.2)} \\
&\leq \mathsf{d}^B(L^B_{\mathsf{p}}) + \mathsf{d}^B(L^B_{\mathsf{q}}) + e^A(\mathsf{p}) + \min\left(e^B(\mathsf{q}), e^B(0)\right), \\
&\qquad\qquad \text{(Triangle inequality)}
\end{aligned}$$

a contradiction to the choice of q.

– **When $E(\pi^A) \neq L^B$:** In this case, we first show that $\mathsf{corr}_{\hat{\Pi},\hat{\mathsf{Adv}},(32\hat{N}:36\hat{N}]}(x^A, x^B) \geq (1-\epsilon)\hat{N}$. We have:

$$\begin{aligned}
\mathsf{corr}_{\hat{\Pi},\hat{\mathsf{Adv}},(32\hat{N}:36\hat{N}]}(x^A, x^B) &\geq \Delta\left(\mathsf{ECC}(L^B), \sigma\right) \\
&\geq \max\left(\Delta\left(\mathsf{ECC}(E(\pi^A), \sigma\right), (2-\epsilon)\hat{N} - \Delta\left(\mathsf{ECC}(E(\pi^A), \sigma\right)\right) \\
&\qquad\qquad \text{(Definition of } E \text{ and Lemma 4.1)} \\
&\geq (1-\epsilon)\hat{N}.
\end{aligned}$$

Therefore, we have $\mathsf{corr}_{\hat{\Pi},\hat{\mathsf{Adv}},[32\hat{N}]}(x^A, x^B) < (4-\epsilon)\hat{N}$. This implies that $\Pi(x^A, x^B) = L^B_{\mathsf{q}^*}$ where $\mathsf{q}^* = \arg\min_{j' \in [3]} \mathsf{d}^B(L_{j'})$. If not, then, as $\Pi(x^A, x^B) \in L^B$, we have by Theorem 5.1,

$$\begin{aligned}
2 \cdot \mathsf{d}^B(\Pi(x^A, x^B)) &\leq 2 \cdot \mathsf{corr}_{\hat{\Pi},\hat{\mathsf{Adv}},[32\hat{N}]}(x^A, x^B) \\
&< (8 - 2\epsilon)\hat{N} \\
&\leq \mathsf{d}^B(\Pi(x^A, x^B)) + \mathsf{d}^B(L^B_{\mathsf{q}^*}), \qquad \text{(Equation 16)}
\end{aligned}$$

a contradiction to the choice of q^*. Now that we have $\Pi(x^A, x^B) = L^B_{\mathsf{q}^*}$, if $\pi^B \neq \Pi(x^A, x^B)$, then $\mathsf{q} \neq \mathsf{q}^*$ and we have

$$2 \cdot \left(\mathsf{d}^B(L^B_{\mathsf{q}^*}) + e^B(0)\right) \leq 2 \cdot \left(\mathsf{corr}_{\Pi',\mathsf{Adv}',[32\hat{N}]}(x^A, x^B) + e^B(0)\right)$$

$$
\begin{aligned}
&\leq 2 \cdot \left(\mathsf{corr}_{\hat{\Pi},\hat{\mathsf{Adv}},[32\hat{N}]}(x^A, x^B) + \Delta\left(\mathsf{C}_{\hat{N}}(0), \tau\right)\right) \\
&\qquad\qquad \text{(Definition of } e^B(\cdot)\text{)} \\
&\leq 2 \cdot \Big(\mathsf{corr}_{\hat{\Pi},\hat{\mathsf{Adv}},[32\hat{N}]}(x^A, x^B) + \mathsf{corr}_{\hat{\Pi},\hat{\mathsf{Adv}},(36\hat{N}:39\hat{N}]}(x^A, x^B) \\
&\qquad + 2\hat{N} - e^A(\Pi(x^A, x^B))\Big) \quad \text{(Line 55 and Line 56)} \\
&\leq 2 \cdot \Big(\mathsf{corr}_{\hat{\Pi},\hat{\mathsf{Adv}},[32\hat{N}]}(x^A, x^B) + \mathsf{corr}_{\hat{\Pi},\hat{\mathsf{Adv}},(36\hat{N}:39\hat{N}]}(x^A, x^B) \\
&\qquad + \Delta\left(\mathsf{ECC}(L^B), \sigma\right) + 2\epsilon\hat{N}\Big) \qquad (E(\pi^A) \neq L^B) \\
&< (10 - 5\epsilon)\hat{N} \\
&\qquad\qquad (e^A(\Pi(x^A, x^B)) \leq \mathsf{corr}_{\hat{\Pi},\hat{\mathsf{Adv}},(32\hat{N}:36\hat{N}]}(x^A, x^B)) \\
&\leq \mathsf{d}^B(L^B_{\mathsf{q}^*}) + d^B(L^B_{\mathsf{q}}) + \Delta\left(\mathsf{C}_{\hat{N}}(\mathsf{q}), \mathsf{C}_{\hat{N}}(0)\right) \\
&\qquad\qquad \text{(Equation 16 and Lemma 4.2)} \\
&\leq \mathsf{d}^B(L^B_{\mathsf{q}^*}) + d^B(L^B_{\mathsf{q}}) + e^B(0) + e^B(\mathsf{q}), \\
&\qquad\qquad \text{(Triangle inequality)}
\end{aligned}
$$

a contradiction to the choice of q.

□

References

[ABP18] Noga Alon, Boris Bukh, and Yury Polyanskiy. List-decodable zero-rate codes. *IEEE Transactions on Information Theory*, 65(3):1657–1667, 2018. 3

[AGS16] Shweta Agrawal, Ran Gelles, and Amit Sahai. Adaptive protocols for interactive communication. In *Information Theory (ISIT)*, pages 595–599. IEEE, 2016. 3

[BE17] Mark Braverman and Klim Efremenko. List and unique coding for interactive communication in the presence of adversarial noise. *SIAM Journal on Computing*, 46(1):388–428, 2017. 2, 3, 4, 6, 11

[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. 3

[BKN14] Zvika Brakerski, Yael Tauman Kalai, and Moni Naor. Fast interactive coding against adversarial noise. *Journal of the ACM (JACM)*, 61(6):35, 2014. 3

[Bli86] Vladimir Markovich Blinovskii. Bounds for codes in the case of finite-volume list decoding. *Problemy Peredachi Informatsii*, 22(1):11–25, 1986. 3, 11

[BR11] Mark Braverman and Anup Rao. Towards coding for maximum errors in interactive communication. In *Symposium on Theory of computing (STOC)*, pages 159–166. ACM, 2011. 1, 2, 3, 4, 5, 6, 9, 11

[Bra12] Mark Braverman. Towards deterministic tree code constructions. In *Innovations in Theoretical Computer Science (ITCS)*, pages 161–167. ACM, 2012. 3

[EGH16] Klim Efremenko, Ran Gelles, and Bernhard Haeupler. Maximal noise in interactive communication over erasure channels and channels with feedback. *IEEE Transactions on Information Theory*, 62(8):4575–4588, 2016. 2, 3

[EKS20a] Klim Efremenko, Gillat Kol, and Raghuvansh Saxena. Interactive error resilience beyond 2/7. In *Symposium on Theory of Computing (STOC)*. ACM, 2020. 3

[EKS20b] Klim Efremenko, Gillat Kol, and Raghuvansh Saxena. Optimal error resilience of adaptive message exchange. Manuscript, 2020. 3

[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. 3

[Gel17] Ran Gelles. Coding for interactive communication: A survey. *Foundations and Trends® in Theoretical Computer Science*, 13(1–2):1–157, 2017. 2, 3

[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, 11

[GHK+18] Ran Gelles, Bernhard Haeupler, Gillat Kol, Noga Ron-Zewi, and Avi Wigderson. Explicit capacity approaching coding for interactive communication. *IEEE Transactions on Information Theory*, 64(10):6546–6560, 2018. 3

[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. 3, 4

[GMS11] Ran Gelles, Ankur Moitra, and Amit Sahai. Efficient and explicit coding for interactive communication. In *Foundations of Computer Science (FOCS)*, pages 768–777. IEEE, 2011. 3

[GMS14] Ran Gelles, Ankur Moitra, and Amit Sahai. Efficient coding for interactive communication. *IEEE Transactions on Information Theory*, 60(3):1899–1913, 2014. 3

[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)*, pages 75:1–75:14, 2018. 3

[KR13] Gillat Kol and Ran Raz. Interactive channel capacity. In *Symposium on Theory of computing (STOC)*, pages 715–724, 2013. 3

[Sch92] Leonard J Schulman. Communication on noisy channels: A coding theorem for computation. In *Foundations of Computer Science (FOCS)*, pages 724–733. IEEE, 1992. 3

[Sch93] Leonard J Schulman. Deterministic coding for interactive communication. In *Symposium on Theory of computing (STOC)*, pages 747–756. ACM, 1993. 3

[Sch96] Leonard J. Schulman. Coding for interactive communication. *IEEE Transactions on Information Theory*, 42(6):1745–1756, 1996. 1, 3, 5

[Sha09] Ofer Shayevitz. On error correction with feedback under list decoding. In *IEEE International Symposium on Information Theory*, pages 1253–1257. IEEE, 2009. 4

[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. 3

A Binary List Decoding Protocols With $\frac{1}{4}$ Errors Needs Exponential List Size

We now sketch the proof of a claim made in Subsection 2.2 that a list decodable protocol with a binary alphabet for the message exchange task requires exponential sized lists if the fraction of errors is larger than $\frac{1}{4}$.

Proof sketch. Let $\epsilon > 0$ be fixed and let Π be a non-adaptive list decodable protocol for the message exchange task that is resilient to $\frac{1}{4} + \epsilon$ fraction of errors. Without loss of generality, let Alice be the party that speaks less in Π and note that this is well defined as Π is non-adaptive.

Fix Bob's input x^B arbitrarily. For every input x^A for Alice, define the set $S(x^A)$ of Bob's transcripts as follows: We say that a transcript $\pi \in S(x^A)$ iff there exists an adversary Adv for Π that corrupts at most $\frac{1}{4} + \epsilon$ fraction of messages and ensures that the transcript received by Bob is π when Alice and Bob have inputs x^A and x^B respectively.

Also define, for a transcript π received by Bob, the list $L(\pi)$ to be the list output by Bob when he receives this transcript and his input is x^B. Observe that, for all $\pi \in S(x^A)$, we must have $x^A \in L(\pi)$. We get that

$$\sum_{x^A} |S(x^A)| \leq \sum_{\pi} |L(\pi)| \leq |\Pi_B| \cdot \max_{\pi} |L(\pi)|,$$

where Π_B is the set of all transcript that Bob can potentially receive. Next, we observe that, for all x^A, a random transcript $\pi \in S(x^A)$ with probability at least $\frac{1}{2}$. If $\mathcal{X}^A$ denote the set of all Alice's inputs, we get that:

$$\frac{1}{2} \cdot |\Pi_B| \cdot |\mathcal{X}^A| \leq |\Pi_B| \cdot \max_{\pi} |L(\pi)| \implies \max_{\pi} |L(\pi)| \geq \frac{1}{2} \cdot |\mathcal{X}^A|.$$

The proof is done with the observation that $\max_\pi |L(\pi)|$ is the list size of the protocol.

□

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