Electronic Colloquium on Computational Complexity, Report No. 7 (2014)

List and Unique Coding for Interactive Communication in the Presence of Adversarial Noise

Mark Braverman*and Klim Efremenko†

Abstract

In this paper we extend the notion of list decoding to the setting of interactive communication and study its limits. In particular, we show that any protocol can be encoded, with a constant rate, into a list-decodable protocol which is resilient to a noise rate of up to $1/2 - \varepsilon$, and that this is tight.

Using our list-decodable construction, we study a more nuanced model of noise where the adversary can corrupt up-to α fraction of Alice's communication and up-to β fraction of Bob's communication. We use list-decoding in order to fully characterize the region $\mathcal{R}_U$ of pairs (α, β) for which unique decoding with constant rate is possible. The region $\mathcal{R}_U$ turns out to be quite unusual in its shape. In particular, it is bounded by a piecewise-differentiable curve with infinitely many pieces. We show that outside this region the rate must be exponential. This suggests that in some error regimes list-decoding is necessary for optimal unique decoding. We also consider the question what if only one party of the communication must to output the correct answer. We precisely characterize the region of all pairs (α, β) for which one-sided unique decoding is possible in a way that Alice will output the correct answer.

1 Introduction

We consider the problem of interactive computation in the presence of adversarial errors. In this setting Alice and Bob want to perform a computation over a channel utilizing an alphabet Σ which is affected by an adversarial noise of rate η. In other words, if the total number of symbols transmitted by Alice and Bob is N (which is known *a-priori* to all the participants), then the adversary is allowed to corrupt at most ηN symbols of the transmission. The goal is to provide a scheme that can simulate any communication protocol in an error-resilient way.

The non-interactive version of the problem is the well-studied problem of encoding a message $M \in \Sigma^N$ with an error-correcting code $C : \Sigma^N \to \Sigma_2^{N'}$ resilient to adversarial errors. To be resilient to errors of rate η we need the Hamming distance between each two codewords $C(M_1)$ and $C(M_2)$ to be sufficiently well-spaced, so that corrupt versions of these words can be recovered correctly. Specifically, we need $d_H(C(M_1), C(M_2)) > 2\eta N'$ for all $M_1, M_2 \in \Sigma^N$. A code C is said to be good if it has a constant rate: $N' = O(N)$ and $\log |\Sigma_2| = O(\log |\Sigma|)$; in other words, a good code stretches

*Department of Computer Science, Princeton University. Research supported in part by an Alfred P. Sloan Fellowship, an NSF CAREER award (CCF-1149888), NSF CCF-0832797, a Turing Centenary Fellowship, and a Packard Fellowship in Science and Engineering.

†Department of Computer Science, University of Chicago. Research supported by a Simons Fellowship in Theoretical Computer Science. Part of this work was done while KE was a member of the Institute for Advanced Studies and funded by NSF CCF-0832797 and DMS-0835373.

ISSN 1433-8092

the input only by a constant factor. The most interesting case studied is when $|\Sigma| = O(1)$. For any $\eta = 1/2 - \varepsilon < 1/2$, a simple probabilistic argument shows that there exist good codes against adversarial errors of rate η. There are several well-known constructions of good codes.

Once the error rate η exceeds $1/2$, there is no hope of recovering from a η-fraction of errors, since for any M_1, M_2 there is a message $\tilde{M}$ such that $d_H(C(M_1), \tilde{M}) \leq \eta N'$ and $d_H(C(M_2), \tilde{M}) \leq \eta N'$, which means that $\tilde{M}$ could be a $\eta N'$-corrupted encoding of either M_1 and M_2. Nonetheless, using *list-decoding*, it is possible to recover from error rates exceeding $1/2$. A code is said to be η-list-decodable with list of size L if for every word $\tilde{M} \in \Sigma_2^{N'}$, the number of codewords within relative distance η from that word is at most L. The notion of list-decoding dates back to works by Elias [Eli57] and Wozencraft [Woz58] in the 1950s. Once again, good list-decodable codes for any $\eta = 1 - \varepsilon < 1$ can be shown to exist probabilistically. Moreover, efficient constructions of list-decodable codes exist, and have numerous applications [Gur04].

In the interactive setting, it is not at all obvious that good error-correction is possible against adversarial substitution errors of *any* rate. Note that any attempt to apply standard error-correcting codes round-by-round are bound to fail, since all the adversary has to do to derail the execution of the protocol is to completely corrupt a single round. Therefore, a sub-constant error rate of $1/r$ suffices to foil an r-round protocol protected with a round-by-round code. In a striking work, Schulman [Sch96] showed that there exist good error-correcting codes against errors of rate $\eta < 1/240$. This work introduced the *tree code* primitive, variations on which have been used in follow-up works, including the present one. Informally, the tree code combines two desirable properties: (1) it is an "online" code: the i-th symbol of the encoding can be computed from the first i symbols of the original word, thus allowing the encoding of the conversation to be computed as it progresses; (2) its error correction properties are such that two messages are encoded to a two codewords which are far from each other. Where how far a way these two codewords depends on the first place where two messages are different.

Interest in interactive error correction has been renewed recently, with Braverman and Rao [BR11] showing that the error rate that can be tolerated can be improved to $\eta < 1/4$. The constructions of both [Sch96] and [BR11] are not computationally efficient. A series of recent works made significant progress towards making interactive error correction computationally efficient [GMS11, Bra12, BK12, FGOS12, BN13], in some cases by restricting the error parameter or augmenting or restricting the model. In particular, the work of Brakerski and Kalai [BK12] shows how to implement interactive error correction efficiently for $\eta < 1/16$ with a beautiful scheme which uses private (non-shared) randomness.

As with previous works, in this paper we will only consider *robust protocols*: protocols in which at all times, whose turn to speak is known to both Alice and Bob, regardless of the amount of errors introduced into the communication so far (even if this amount exceeds the adversary's budget). It is not hard to see [BR11] that robustness is equivalent to the property that the identity of the speaker at round i at any execution of the protocol depends only on i. Non-robust protocols have recently received some attention. In particular, very recently, [AGS13] and [GHS13] considered non-robust models and concluded that indeed non-adaptive protocols can withstand a higher error rate. It would be interesting to combine these results, particularly from [GHS13] with ours to calculate the tight error-rate region for the non-robust case.

In this work we investigate the limits of error rates which can be corrected in the interactive setting. Further, we develop the notion of *interactive list decoding*, which is the list-analogue of interactive error-correction. Its definition is quite straightforward: after the execution of a protocol,

each party will output a constant-size list of possible original conversations. If the fraction of errors has not exceeded η, each list will contain the intended conversation. We show that constant-rate interactive list decodable coding is possible for all error rates $\eta < 1/2$, which we show to be the best error tolerance one can hope for.

Moreover, interactive list decoding turns out to be the right tool for giving tight bounds on the error rates one can tolerate in the *unique decoding* setting. In the interactive setting it is natural to consider pairs (α, β) of error rates, with α representing the fraction of Alice's communication that may be corrupted and β representing the fraction of Bob's communication that may be corrupted. Using our list decoding results we are able to give a precise characterization of the region $\mathcal{R}_U$ of pairs (α, β) of error rates for which constant-rate unique decoding is possible. Previously, by the construction of Braverman and Rao [BR11] unique decoding has been known to be possible when $\alpha+\beta < 1/2$. At the same time, it is easy to see that unique decoding is impossible when $\alpha \geq 1/2$ or $\beta \geq 1/2$. The region $\mathcal{R}_U$ turns out to be quite unusual in its shape. In particular, it is bounded by a piecewise-differentiable curve with infinitely many pieces. When Alice and Bob are affected by an equal amount of error, the intersection of $\mathcal{R}_U$ with the line $\alpha = \beta$ is the region $\{(\alpha, \alpha) : \alpha < 1/3\}$. Thus we can handle error of up to $1/3 - \varepsilon$ affecting each of Alice and Bob. Previously, only a lower bound of $1/4 - \varepsilon$ and an upper bound of $1/2$ were known [BR11].

The notion of list-decodable interactive codes has been independently developed by Ghaffari, Haeupler, and Sudan [GHS13, GH13]. While the key definitions are, as expected, similar, there are some differences between the two lines of work. All the codes we consider in our work are constant rate, i.e. an n-symbol interactive protocol is always encoded into an $O(n)$-symbol protocol. The codes in [GHS13] convert n symbols into n^2 symbols, while the improved [GH13] converts n symbols into $n \cdot 2^{O(\log^* n \cdot \log\log^* n}$. Our analysis is more detailed in terms of the error region (we analyze the pairs of error-rates that we can tolerate, and not only their sum). On the other hand, our scheme is not computationally efficient, while the scheme of [GH13] is. More excitingly, it appears that by combining [GH13] with the present paper one obtains efficient, constant rate, schemes with optimal error dependence[1]. The question of finding the *optimal-rate* interactive error correcting schemes remains wide open.

Main results

List decoding. Our first set of results deals with list-decoding interactive protocols. We say that protocol can handle (α, β) adversarial noise if adversary can corrupt up-to α fraction of Alice's messages and up-to β fraction of Bob's messages. For formal definition see Section 3.

Theorem 1. *For each $\varepsilon > 0$ and for every protocol π there exists another protocol π', with $CC(\pi') = O_\varepsilon(CC(\pi))$ which is resilient (α, β) adversarial noise for all $\alpha + \beta < 1 - \varepsilon$. The protocol $\pi'(x, y)$ outputs a list of size $O_\varepsilon(1)$ of transcripts such that $\pi(x, y)$ is on the list.*

On the other hand, we show that for each pair (α, β) with $\alpha+\beta \geq 1$, list decoding is impossible.

Theorem 2. *For every α, β such that $\alpha + \beta \geq 1$. Let π be a protocol which is resilient to (α, β) adversarial noise and which solves list decoding problem of Pointer Jumping Problem of depth T with list of size $\exp(o(T))$. Then $CC(\pi) = \exp(\Omega(T))$.*

[1] Haeupler, personal communication.

Note that the special case of Theorem 2 where $(\alpha, \beta) = (1, 0)$ is trivial, since in this case no useful communication is transmitted by Alice.

We can give an even tighter result by considering a slightly generalized error notion in Theorem 1. Let us only consider standard protocols π' where Alice and Bob send one message at a time in an alternating fashion. We can partition such a π' into blocks of two symbols, the first one being sent by Alice and the second by Bob. We say that a block is corrupted if the transmission of either symbol in the block is corrupted. Let η be the fraction of blocks the adversary corrupts. Note that it is always the case that $\max(\alpha, \beta) \leq \eta \leq \alpha + \beta$. We show that we can handle η-symmetric noise(for exact definition of symmetric noise see Section 3) up to $1 - \varepsilon$, matching the one-way list decoding bounds.

Theorem 3. *For each $\eta < 1$ and for every protocol π there exists another protocol π', with $CC(\pi') = O_\eta(CC(\pi))$ which is resilient η-symmetric noise. The protocol $\pi'(x, y)$ outputs a list of size $O(\frac{1}{\eta})$ of transcripts such that $\pi(x, y)$ is in the list.*

Note that since $\eta \leq \alpha + \beta$, Theorem 3 implies Theorem 1.

Unique decoding. Next, we turn our attention to the problem of unique decoding. In the unique decoding setting, at the end of the execution of π' Alice and Bob need to be able to correctly recover the original protocol transcript π. In Section 12 we also consider the asymmetric case where only Alice needs to recover π uniquely.

We study the set $\mathcal{R}_U$ of pairs (α, β) for which unique decoding is possible. The set $\mathcal{R}_U$ is rather peculiar, and is defined as follows. Let

$$L_2(\alpha) := \frac{1}{2}\left(1 - \frac{1}{(1 + \{\frac{1}{\alpha}\}) \cdot 2^{\lfloor \frac{1}{\alpha} \rfloor - 1} - 1}\right),$$

where $\{x\} := x - \lfloor x \rfloor$ is the fractional part of x. Then the unique decoding region is defined by

$$\mathcal{R}_U := \left\{(\alpha, \beta) : (\alpha \leq \frac{1}{3} \text{ and } \beta < L_2(\alpha)) \text{ or } (\beta \leq \frac{1}{3} \text{ and } \alpha < L_2(\beta))\right\}. \quad (1)$$

It is plotted on Figure 1. Note that L_2 is continuous and piecewise-differentiable with infinitely many pieces. Also, $L_2(\frac{1}{3}) = \frac{1}{3}$, with $(\frac{1}{3}, \frac{1}{3})$ being the intersection point between the boundary $\partial\mathcal{R}_U$ and the line (α, α).

Theorem 4. *For each $(\alpha, \beta) \in \mathcal{R}_U$ and for every protocol π there exists another protocol π', with $CC(\pi') = O_{\alpha,\beta}(CC(\pi))$ which is resilient (α, β) adversarial noise. Such that $\pi'(x, y)$ outputs transcript of $\pi(x, y)$.*

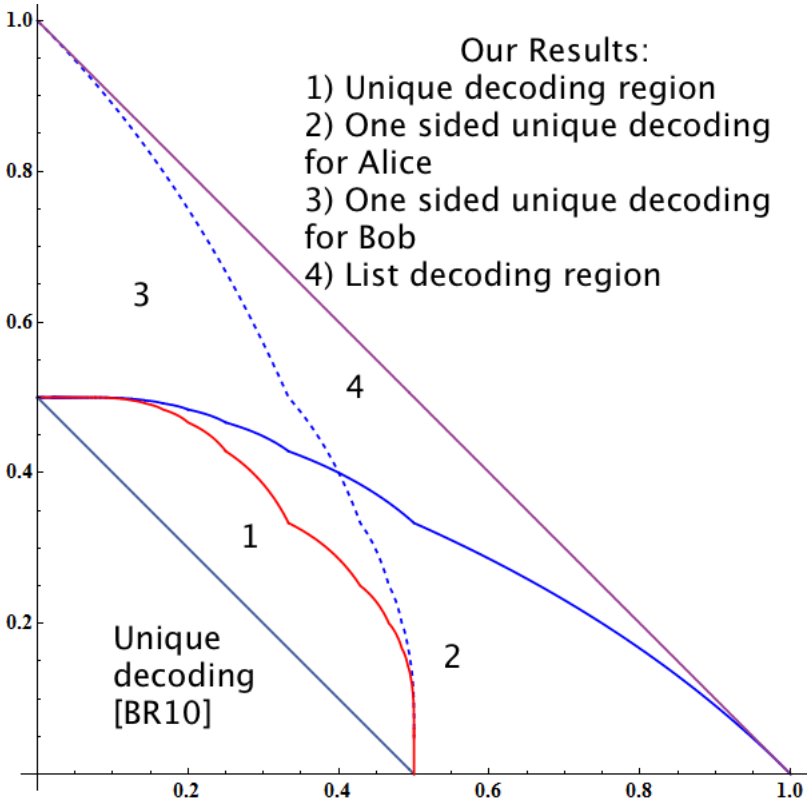
Theorem 4 is stronger than the main upper bound of [BR11], which only shows how to deal with (α, β) in the sub-region $\alpha + \beta < 1/2$. More importantly, Theorem 4 turns out to be essentially tight, as shown in the following theorem. Let $\overline{\mathcal{R}}_U$ be the closure of $\mathcal{R}_U$, i.e. the union of $\mathcal{R}_U$ and its boundary. The Pointer Jumping Problem can be viewed as the generic communication complexity problem and is formally defined in Section 2.

Theorem 5. *For every $(\alpha, \beta) \notin \overline{\mathcal{R}}_U$ and for every T the following holds. Let π' be a protocol resilient to (α, β) adversarial noise which solves the Pointer Jumping Problem of depth T. Then $CC(\pi') = 2^{\Omega_{\alpha,\beta}(T)}$.*

We note that in noiseless regime one can solve Pointer Jumping Problem of depth T with communication complexity T. Therefore outside of $\overline{\mathcal{R}}_U$ one cannot perform unique decoding with constant stretch in communication.

Remark 6. *The lower bound statement of Theorem 5 is the best we can hope for in the following sense: as long as $\alpha < 1/2$ and $\beta < 1/2$, Alice and Bob can achieve unique-decodable communication with exponential stretch. Alice can use a (non-interactive) good code to send Bob her messages on* all *potential protocol transcripts, and Bob can do the same. This guarantees correct execution, but causes an exponential stretch in communication.*

While Theorem 4 shows that unique decoding is possible in the interior of $\mathcal{R}_U$, and Theorem 5 shows that it is impossible in the interior of its complement, unlike the list decoding case we do not establish whether unique decoding is possible for error rates on the boundary $\partial\mathcal{R}_U$. We conjecture that similarly to the list-decoding case, the boundary belongs to the region where decoding is not possible.



Main techniques and discussion

We start with a discussion of the proof of the positive results on interactive list decoding. The overall strategy is similar to that of other recent works: "make progress as long as the error is not too high". From the viewpoint of, say, Bob, this means the following: at each step, Bob will try to decode from Alice's messages he received so far what her (noiseless) messages have been, and sends a response that makes progress on the overall protocol. Thus we make progress as long as Bob decodes Alice's messages correctly. It turns out that if Alice and Bob encode their messages using

tree codes, this strategy works as long as $\alpha + \beta < 1/2 - \varepsilon$ [BR11]: in other words, in sufficiently many rounds, Alice and Bob will correctly reconstruct each others' transmissions so far.

What goes wrong when $1/2 < \alpha + \beta < 1$? In this case, for example if $\alpha > 1/2$, as in the case of one-way communication, there is no hope for Bob to be able to ever unambiguously reconstruct Alice's message. As in the case of one-way list decoding, Bob can still hope to be able to recover a constant size list L of potential Alice's communications so far. To make progress, Bob will send $|L|$ responses to simultaneously make progress on all of these communications. Therefore, list size of size ℓ causes an ℓ-fold explosion in communication, and ℓ needs to be kept constant at (almost) all times. A substantial amount of technical work is required to keep the rate of the code constant in the case of a constant-size alphabet.

The analogue of tree codes that allows us to carry out the above plan is a new primitive we call a *list tree code*. The actual definition is somewhat technical, but informally, a list tree code is a prefix code such that for any received word w, for a significant number of rounds, there are at most ℓ proper codewords that are $(1-\varepsilon)$-close to w. With correctly selected parameters, a random prefix code is a list tree code except with an exponentially small probability. This contrasts with the case of ordinary tree codes, for which a random prefix code is unlikely to be a tree code. List tree codes have resemblance to *potent tree codes* of Gelles, Moitra and Sahai [GMS11], which can informally be viewed as "list tree codes" with list size $\ell = 1 + \varepsilon$.

Next we turn our attention to the unique decoding regime. It appears that understanding (and being able to carry out) interactive list decoding is needed to achieve optimal unique decoding. To illustrate this point, consider an attempt to break the $(\frac{1}{4}, \frac{1}{4})$-barrier from previous (non-list decoding) works. To be specific, let $(\alpha, \beta) = (\frac{1}{4}, \frac{1}{4})$. The adversary can corrupt half of Alice's messages in the first half of the protocol, and half of Bob's messages in the second half of the protocol. It is easy to see that in the first half of the execution, unique decoding by Bob is not possible, since there is $\frac{1}{2}$-error on Alice's messages. Therefore, if the parties wait until they can decode uniquely, they will not make any progress in the first half of the protocol. Similarly, they will also not make progress in the second half. On the other hand, with list decoding, in this scenario, we can achieve that by the end of the first part of the protocol Alice has decoded the transcript π, and Bob has at most two candidate transcripts π_1 and π_2. He can then use the second (non-corrupt) part of Alice's transmission to decide whether to output π_1 or π_2.

By just using the list-decodable interactive scheme, and having each party output the answer closest to the received transcript, we can already overcome the $\alpha + \beta < 1/2$ barrier, and get an error-correcting scheme that works for (α, β) in the region:

$$\mathcal{R}_2 := \{(\alpha, \beta) : \ \alpha + 2\beta < 1 \text{ and } 2\alpha + \beta < 1\}.$$

In particular, $\mathcal{R}_2$ covers all (α, α) for $\alpha < 1/3$.

In our main list-decodable scheme, Alice and Bob speak at the same rate throughout the protocol (i.e. by the time Alice communicates a p-fraction of her messages, Bob communicates a p-fraction of his messages). It turns out that for some values of (α, β) outside of $\mathcal{R}_2$, we can still achieve unique decoding by having Alice and Bob alter the relative rates at which they speak throughout the execution of the scheme. Note that our scheme is still robust, and therefore these rates will be predetermined by (α, β). For example, if we consider the point $(\frac{1}{4}, \frac{3}{7} - \varepsilon) \in \mathcal{R}_U \setminus \mathcal{R}_2$, the unique decodable protocol will look as follows: by the time Alice communicates $\frac{1}{4}$ of her messages, Bob will communicate approximately $\frac{1}{7}$ of his; after that, for the next $\frac{3}{4}$ of Alice's messages, Bob will send the remaining $\frac{6}{7}$ of his communication in a uniform fashion. The most striking feature of this

general regime is the complicated shape of the unique decoding region $\mathcal{R}_U$, which is the result of the complicated way in which altering relative transmission rates achieves decodability.

Finally, we discuss our matching lower bounds. Going back to the list decoding setting, consider the case when $\alpha + \beta \geq 1$. If Alice and Bob speak at the same rate, the adversary can erase the first α-fraction of Alice's communication, and the last β fraction of Bob's communication. This way there is no overlap between the part where Alice speaks and the part where Bob speaks. Therefore the encoded protocol is equivalent to a two-round protocol between Alice and Bob. There are communication problems which require more than two rounds to execute efficiently (even with a small success probability). By starting with such a problem, we see that a list decodable encoding that can withstand (α, β)-error must result in a significant communication blowup. The argument above fails if Alice and Bob speak at different rates throughout the protocol. For example, if most of Alice's communication is concentrated early in the execution and most of Bob's communication is concentrated late. Using a slightly more complicated argument, we can show that any protocol resilient to (α, β)-noise with $\alpha + \beta \geq 1$ can be simulated by a 3-round protocol, leading to a similar contradiction. The argument for the tightness of the region $\mathcal{R}_U$ is more involved but follows a similar methodology.

Acknowledgments

We thank Ran Gelles for for the many insightful comments on an earlier draft of this paper.

2 Communication protocols, pointer jumping and errors

For sake of completeness we put here section from [BR11].

Given inputs x, y from some domain, a deterministic protocol with alphabet Σ is a rooted tree where every internal vertex has $|\Sigma|$ children, each corresponding to an element of the alphabet. Every non-leaf vertex in the tree v is associated with a function $f_v(x)$ (or $f_v(y)$) that maps one of the inputs to an element of Σ. The outcome of the protocol on inputs x, y is the unique leaf that is reached by first evaluating the function associated with the root, then evaluating the function associated with the child obtained by the first evaluation, and so on. The depth of the tree, T is called the communication complexity of the protocol. Two parties who each have access to just one of the inputs can compute the outcome of the protocol by communicating at most T symbols to each other, in the obvious way.

In this paper, we only consider deterministic communication protocols, since our results easily extend to the case of randomized protocols by viewing a randomized protocol as a distribution over deterministic protocols.

Let $\mathcal{T}$ be a rooted s-ary tree of depth T. Let $\mathcal{X}$ denote the set of edges leaving vertices at even depth, and $\mathcal{Y}$ denote the set of edges leaving vertices at odd depth. Given any set A of edges in the tree, we say that A is *consistent* if A contains at most one edge leaving every vertex of the tree. We write $v(A)$ to denote the unique vertex of maximal depth that is reachable from the root using the edges of A and $e(A)$ will be the last edge in this path.

Let $X \subset \mathcal{X}$, $Y \subset \mathcal{Y}$ be consistent subsets. Then observe that $X \cup Y$ is also consistent. In the pointer jumping problem, the two parties are given such sets X, Y and the goal is compute $v(X \cup Y)$. Since every communication protocol with communication complexity t bits can be

reduced to solving pointer jumping on a tree of depth $2t$, it will suffice for us to find a protocol that can compute $v(X \cup Y)$ even if there are transmission errors.

In this paper, we define communication protocols that are resilient to transmission errors. In our protocols, every vertex at odd depth will be labeled by a function of the first party's input, and every vertex at even depth will be labeled by a function of the second party's input. Given such a protocol, each party runs it as above, using her transmissions and the messages she receives to determine her current position in the protocol tree. In general, each party will conclude the protocol at a different leaf. We are going to consider in this paper 3 problems: First is just pointer jumping problem where both Alice and Bob have to compute $v(X \cup Y)$. Second is one-way pointer jumping problem where at the end of the protocol only Alice has to know $v(X \cup Y)$ and the last one is list decodable pointer jumping problem is where at the end of the protocol both Alice and Bob has to output small list of nodes such that $v(X \cup Y)$ is in the list.

Notation 7. *We will denote by $PJP(T, s)$ the pointer jumping problem of depth T and alphabet size s.*

Theorem 8. *[[NW93]] For every $k < T$ there exists a function $f(x, y)$ which is solvable in T communication complexity, but every k round protocol $\pi(x, y)$ which computes $f(x, y)$ with probability at least $\frac{2}{3}$ has communication complexity at least $\Omega(\frac{\exp(\Omega(\frac{T}{k}))}{k^2})$.*

Proof. Set $T = O(\frac{\log n}{k})$ and use Theorems 2.3, 2.6 from [NW93]. □

Corollary 9. *For every ε, k, if protocol π solves one way $PJP(2T, 2)$ in $k-4$ rounds with probability ε then the communication complexity of π is at least $\Omega(\varepsilon \cdot \exp(\Omega_k(T)))$*

Proof. Let us assume that we have protocol π which solves one way $PJP(2T, 2)$ in $k - 4$ rounds with probability ε. Let us show that we can solve $PJP(2T, 2)$ in k rounds with probability $\frac{2}{3}$ and with communication complexity $O(\frac{CC(\pi)+T}{\varepsilon})$. Alice and Bob can execute protocol π in parallel $t = \frac{10}{\varepsilon}$ times. Let us assume that $v_{A1}, v_{A2} \dots v_{At}$ be a leafs that Alice got on these executions and $v_{B1}, v_{B2} \dots v_{Bt}$ be a leafs that Bob got on these executions. Then with probability at least 0.9 one of these leafs is the correct answer. If one of the v_i is the correct answer we can find it in 4 rounds and with $O(\frac{T}{\varepsilon})$ communication. It is easy to see that if v consistent with Alice's input and v_{Bi} consistent with Bob's input then v is the correct output. Then Alice and Bob will send all his leafs consistent with his input and out of these leafs they will see which one is the correct answer and tell it to an other party.

Since $PJP(2T, 2)$ is complete for communication complexity our protocol can compute function $f(x, y)$ in k rounds with probability $\frac{2}{3}$. Therefore from Theorem 8 it follows that $O(\frac{CC(\pi)+T}{\varepsilon}) \geq \Omega(\frac{\exp(\Omega(\frac{T}{k}))}{k^2})$. Therefore, $CC(\pi) \geq \Omega(\varepsilon \exp(\Omega_k(T)))$. □

3 Definition of Adversarial Error Protocol

Let us denote by $D(\Sigma)$ to be the set of probability distributions over Σ. In this paper we will consider the scenario when adversary can corrupt fraction of symbol i.e., each party sends symbol, but may receive a distribution over symbols. The amount of error if was send symabol a and received distribution b is: $Prob(a \neq b)$. For simplicity of notation We will identify Σ with probability distributions supported on single symbol. For $x, y \in D(\Sigma)^m$ we can define $d(x, y) = \sum_{i=1}^{m} Pr(x[i] \neq$

$y[i]$) we will also denote by $agr(x,y) = m - d(x,y)$. Note that if $x, y \in \Sigma^n$ then d is exactly hamming distance between x, y.

Definition 10. *We say that a protocol π' list decodes π with list of size L, if for every input x, y both parties of the protocol of $\pi'(x,y)$ output list of size at most L with $\pi(x,y)$ in the list.*

Definition 11. *We say that a protocol π one sided protocol if its correctness depends only on the output of one party.*

In this paper we will assume that all our protocols are oblivious i.e., who is speaking at round i depends only on i and not on randomness or input of the protocol x, y. Alphabet of our protocols will be Σ_{out}. We will say that communication complexity of the protocol is the maximal over all inputs of bits send by our protocol. We will also have alphabet Σ_{out} over which our protocol is executed. Alice(Bob) communication complexity $n_A(n_B)$ is the maximal number of symbols(from Σ_{out}) sent by Alice(Bob).

In this section we would like to discuss several variants of definitions of protocols resilient to adversarial error. We will assume in this paper that all our protocols are are deterministic.

Informally Communication Protocol Resilient to (α, β) noise is protocol which output correct answer when adversary can change α fraction of symbols sent by Alice and β fraction of symbols sent by Bob.

Definition 12 (Communication Protocols Resilient to (α, β) noise)**.** *Assume that we have deterministic protocol $\pi(x,y)$. For $w_A \in \Sigma_{out}^{n_B}$ let $\pi_A(x)(w_A)$ be a messages sent by Alice when she received w_A from Bob. We say that protocol computing $f(x,y)$ is resilient to (α, β) noise if for every input x, y and for every $w_A \in D(\Sigma_{out})^{n_B}, w_B \in D(\Sigma_{out})^{n_A}$ such that*

$$d(\pi_A(x)(w_A), w_B) \leq \alpha n_A,$$

i.e.,at most α fraction of bits sent by Alice is corrupted. And the same for Bob:

$$d(\pi_B(y)(w_B), w_A) \leq \beta n_B \ .$$

It holds that both Alice and Bob perform task T.

In case that we allow more than $\frac{1}{2}$ noise of total communication we have a problem since Eve can corrupt all the communication of one of the parties. Therefore we will limit Eve's power to prevent this scenario. We will divide our communication to blocks of two so that in each block both both Alice and Bob exchange messages. We will say that some block is corrupted if message of one of the parties is corrupted. That is in this case if Eve decides to corrupt the communication of Alice in some block she can corrupt all the communication of Bob in this block at the same cost. Let us first give formal definitions:

Definition 13 (Communication Protocols Resilinet to (α)-symmetric noise)**.** *In this definition we will consider only protocols where at each even round Alice speak and at odd rounds Bob speaks. The protocol runs for $2n$ rounds.*

We say that protocol performing task $f(x,y)$ is resilient to α- symmetric noise if for every input x, y and for every $w_A \in D(\Sigma_{out})^n, w_B \in D(\Sigma_{out})^n$, let us define $c_A = \pi_A(x)(w_A)$ be an output of Alice on input w_A and $c_B = \pi_B(y)(w_B)$ be an output of Bob.

$$\sum_{i=1}^{n} max(d(w_B[i], c_A[i]), d(w_A[i], c_B[i])) \leq \alpha n \ ,$$

Then Alice and Bob will correctly perform task $f(x, y)$.

Part I
List Decoding

4 List Tree Codes

Definition 14. *A prefix code $C : \Sigma_{in}^n \to \Sigma_{out}^n$ is a code such that $C(x)_i$ depends only on $x[1..i]$. By abuse of notation we will also write for $i < n, x \in \Sigma_{in}^i, C(x) \in \Sigma_{out}^i$.*

Definition 15. *The suffix distance between two strings $x, y \in D(\Sigma)^m$ is defined as*

$$\delta_s(x, y) = \max_{i=1..m} d(x[i..m], y[i..m])/(m - i + 1).$$

In other words suffix distance is the largest relative distance between corresponding suffixes of two words.

Now we will define analogue of the list in one-way communication.

Definition 16. *For every $w \in D(\Sigma_{out})^n$ the i-th level ε-list of w under C is given by*

$$List_i(w, C, \varepsilon) := \{x \in \Sigma_{in}^i : \ \delta_s(C(x), w[1..i]) \leq 1 - \varepsilon\}.$$

The ε-list of w under C is given by

$$List(w, C, \varepsilon) := \bigcup_{i=1}^{|w|} List_i(w, C, \varepsilon).$$

We also define

$$PrefixList(w, C, \varepsilon) := \{y \in \Sigma_{in}^* : \ y = x[1..k] \text{ for some } k \text{ and some } x \in List(w, C, \varepsilon)\}.$$

We will identify words $x \in \Sigma_{in}^*$ with nodes in a tree, where $x \in \Sigma_{in}^i$ will be a father of $x \circ c \in \Sigma^{i+1}$. The tree $PrefixList$ is the rooted subtree that spans all the nodes in $List$. In the Figure 1 we give an example of $PrefixList$. Note that in this example although $|\cup List_i| = 5$. The size of PrefixList is 9.

For any rooted subtree PL of full Σ_{in}-ary tree we will denote by $w(PL)$ the tree where we write $w[i]$ on all nodes at depth i and $C(PL)$ just a restriction of C to PL. We will always consider trees in this paper as one-way graphs directed from root downwards. If we have some tree PL and two different labelings of edges $w(PL), C(PL)$ than $d(w(PL), c(PL)) = \sum_{e \in PL} d(w(e), C(e))$.

The following lemma is a good excise on above definitions.

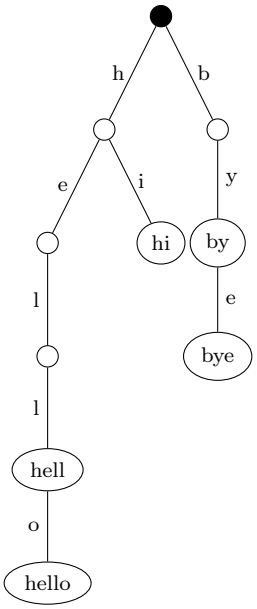


Figure 1: PrefixTree with $List_2 = \{"hi", "by"\}, List_3 = "bye", List_4 = "hell", List_5 = "hello"$

Lemma 17. *Let v be a leaf in the the $PrefixList(w, C, \varepsilon)$ then for every path P ending at v we have $d(C(P), w(P)) \leq (1-\varepsilon)|P|$*

Proof. By definition of $PrefixList$ all its leafs v are in $List(w, C, \varepsilon)$. Thus for some i, $v \in List_i(w, C, \varepsilon)$. By definition of $List_i$ we have $\delta_s(C(v), w) \leq 1 - \varepsilon$. If $|P| = k$ then also by definition $C(P) = C(v)[i-k-1, i]$. Thus from definition of δ_s the lemma follows. □

We will define the size of the tree to be number of its vertexes. Now we are ready to define the main object of this paper.

Definition 18. *An (ε, L)-*list tree code *$C : \Sigma_{in}^n \to \Sigma_{out}^n$ with average list size L and decoding distance $1-\varepsilon$ is a prefix code such that for all $w \in D(\Sigma_{out})^n$ let $PL(w) = PrefixList(w, C, \varepsilon)$ then*

1. *$|PL(w)| \leq L \cdot n$.*
2. *$agr(C(PL), w(PL)) \leq \varepsilon L n$*

As we will see from the next lemma first property is included in the second property. We stated here first property since we are going to use mainly this property of the code. The only reason why we require w to be string of distribution rather than just string of symbols is because this way we can perform soft decoding.

Lemma 19. *Let $PL \triangleq PrefixList(w, C, \varepsilon)$, then $d(C(PL), w(PL)) \leq (1-\varepsilon)|PL|$.*

Proof. We will prove this by induction on number of leafs. If PL has only one leaf v Then PL is a path from root to v and the claim follows from the Lemma 17. For an induction step let PL be a branch of v(i.e. path from w to v where w is a first predecessor of v which has more than one child.) Then $PL \backslash P$ has one leaf less than PL. Thus from induction we get that

$$d(C(PL \backslash P), w(PL \backslash P)) \leq (1-\varepsilon)(|PL| - |P|).$$

From Lemma 17 we know that $d(C(P), w(P)) \leq (1-\varepsilon)|P|$. Thus

$$d(C(PL), w(PL)) = d(C(PL\backslash P), w(PL\backslash P)) + d(C(P), w(P)) \leq (1-\varepsilon)|PL| \ .$$

□

Theorem 20. *For all $\varepsilon > 0$ Σ_{in}, let $|\Sigma_{out}| > (2\Sigma_{in})^{\frac{3}{\varepsilon^2}}$. Then a random prefix code $C : \Sigma_{in}^n \to \Sigma_{out}^n$ is a (ε, L) list-tree code with $L = \frac{1}{\varepsilon} + 1$ with probability at least $1 - 2^{-n}$.*

Before proving this we will need the following lemma.

Lemma 21. *Let PL be full d-ary tree then there exists at most $(d+1)^{2s}$ rooted subtrees of PL of size s.*

Proof. First we will write our tree as a path where each symbol says to what child to go and $d+1$ symbol will say to go up. Next note if we make DFS on subtree of size s then we will pass on every edge twice once when we go down and once when we go up. Thus we can write every subtree of size s by $2s-2$ symbols. □

Proof of Theorem 20. First let us note that from Lemma 19 it follows that $agr(C(PL), w(PL)) \geq \varepsilon|PL|$ thus second property of the list tree codes imply the first property.

Lemma 19 also imply that if PL is a PrefixList that violates the conditions of the theorem then $agr(C(PL), w(PL)) \geq \varepsilon \max\{|PL|, Ln\}$. Now we are going to prove that w.h.p. $agr(C(PL), w(PL)) < \varepsilon \max\{|PL|, Ln\}$ for every subtree PL and for every w. We can assume w.l.g. $|PL| \geq Ln$. Let us first fix some subtree PL of complete Σ_{in}-ary tree of size $s \geq Ln$ and $w \in \Sigma_{out}^n$ then the next claim together with Lemma 19 gives us bound on probability that the conditions of the list tree code broken for this specific w and PL.

Claim 22. *For every $w \in \Sigma_{out}^n$ and tree PL of size at most s it holds that*

$$\mathsf{P}\left[agr(C(PL), w(PL)) \geq \varepsilon s\right] \leq |\Sigma_{out}|^{-\varepsilon s}\binom{s}{\varepsilon s} \leq |\Sigma_{out}|^{-\varepsilon s}2^s,$$

where randomness is taken over the random choice of code C.

Proof. The first inequality follows from union bound where we take union over all possible locations where $C(PL), w(PL)$ have an agreement. The second inequality follows from the fact that

$$2^s = (1+1)^s \geq \binom{s}{\varepsilon s}.$$

□

Thus using union bound over all possible trees and words w we get that probability that C is not list tree code is bounded by

$$\sum_{s=Ln}^{\infty} |\Sigma_{out}|^{-\varepsilon s}2^s \#\{\text{rooted subtrees of size } s\}|\Sigma_{out}|^n$$

From Lemma 21 it follows that probability that C is not tree code is bounded by

$$\sum_{s=Ln}^{\infty} |\Sigma_{out}|^{-\varepsilon s} 2^s (|\Sigma_{in}|+1)^{2s} |\Sigma_{out}|^n$$

Since $|\Sigma_{out}| > (2|\Sigma_{in}|)^{\frac{3}{\varepsilon^2}}$ above quantity is bounded by:

$$(2|\Sigma_{in}|)^{\frac{3n}{\varepsilon^2}} \sum_{s=Ln}^{\infty} (2|\Sigma_{in}|)^{-\frac{3s}{\varepsilon}} 2^s (|\Sigma_{in}|+1)^{2s} \leq 2(2|\Sigma_{in}|)^{\frac{3n}{\varepsilon^2}} (2|\Sigma_{in}|)^{-\frac{3Ln}{\varepsilon}} 2^{Ln} (|\Sigma_{in}|+1)^{2Ln}$$

Recalling that $L = \frac{1+\varepsilon}{\varepsilon}$ we can bound above as:

$$2(2|\Sigma_{in}|)^{-\frac{3}{\varepsilon}n} 2^{\frac{1+\varepsilon}{\varepsilon}n} (|\Sigma_{in}|+1)^{\frac{2+2\varepsilon}{\varepsilon}n} \leq 2^{-n/\varepsilon} \leq 2^{-n} .$$

To see that the theorem holds for all $w \in D(\Sigma_{out})^n$ rather than $w \in \Sigma_{out}^n$ observe that for every subtree PL and for every code $C(PL)$ the closest codeword $w \in D(\Sigma_{out})^n$ to $C(PL)$ is w, where $w[i]$ is the most frequent symbol of $C(PL)$ at level i, thus it is in Σ_{out}^n. □

5 Communication transcript

During our protocol Alice and Bob are going to send edges from the tree $\mathcal{T}$ of the original protocol. In order to describe some specific edge from $\mathcal{T}$ it may take $O(T)$ bits. Also as in Braverman Rao [BR11] we will need one additional level of encoding that will compress our communication. First idea of the compression is that we are not sending a random edges, we never send any edge before we send his grandparent. Therefore we can describe an edge by a link to grandparent and a path from grandparent to an edge. This will solve us the problem for the case that we have alphabet polynomial in the length of our communication since in this case we do not care of the length of the links. In case of constant size alphabet the size of the link is important and therefore we sometimes instead of sending link to grandparent we will send link to cousin who may be much closer to the edge we are sending.

Each entry of our transcript will correspond to some edge in $\mathcal{T}$. Every entry $a_i = (r_i, b_i, s_i)$ of the transcript will consist of integer $r_i \in \mathbb{N}$ which will be a pointer to another edge that appeared r_i entries earlier in our transcript. $b_i \in \{0,1\}$ bit will indicate whether the reference edge is grandparent or cousin[2] and $s_i \in \Sigma_T^{\leq 2}$ which will indicate path from grandparent to the edge. We will assume that an integer k takes at most $2\log k + 2$ bits to encode.

Procedure of decoding $E(A)$: Now let us describe formally how we will decode our transcript for $i = 1, \ldots k$, we will do the following to decode e_i from $a_i = (r_i, b_i, s_i)$:

1. if $r_i = 0$ set e_i to be an edge at depth at most 2 specified by bits s_i.
2. if $r_i \geq i$ return error.
3. if $b_i = 0$ set $p_i = e_{i-r_i}$
4. if $b_i = 1$ set p_i to be grandparent of e_{i-r_i}
5. set e_i be the edge specified by starting at the child vertex of the edge p_i and then taking (at most) two steps in the tree using the bits s_i.

[2]In fact we will need cousins only for a small alphabet

Procedure Encoding $Add(A, e)$**:** Now assume that we want to add an edge e_i to our transcript $a_1, a_2, \dots a_{i-1}$. If e_i is at depth at most two we will set $r_i = 0$ and s_i to correspond to path from root to e_i. Else we will decode edges $e_1, \dots e_{i-1}$ from $a_1, \dots a_{i-1}$. Next we are going to find maximal index j such that e_j is a cousin or grandparent of e_i and then we are going to set $r_i = j - i$ and we will set $b_i = 0$ if e_j is grandparent of e_i and $b_i = 1$ if e_j is cousin of e_i. We will set s_i to correspond to path from grandparent of e_i to e_i. For the purposes of this procedure, an edge is its own cousin.

Thus for $A = (a_1, a_2, \dots a_k)$ we have procedure $E(A)$ which returns edges encoded by A and $Add(A, e)$ which adds edge e to transcript A.

We think of the transcript as a stream of bits and we will also have the following functions:

- Procedure $\mathbf{size}(A)$: which will return the size of the transcript.
- Procedure $\mathbf{End-round}(A, i)$: which will set the size of the transcript to be maximum between $\log \Sigma_{in}^i$ and $size(A)$ and padding if necessary transcript to this size by adding some 0(which will represent no edge).

6 Recovering from errors using a polynomial size alphabet

The goal of this section is to prove the following theorem:

Theorem 23. *For every $\varepsilon, c > 0$ there exists a protocol π is resilient to $1 - \varepsilon$-symmetric noise and which solves the problem of list decoding of $PJP(T, T^c)$ with list of size $O(\frac{1}{\varepsilon})$. Moreover the protocol π runs $O(\frac{T}{\varepsilon})$ rounds and in each round sends $O_{\varepsilon,c}(\log T)$ bits.*

Let $\varepsilon' = c_1\varepsilon$ for some small constant $0 < c_1 < 1$ to be defined later. Define also $L = \frac{1}{\varepsilon'} + 1$. During the protocol we are going to send $O(T)$ edges and encode them with the transcript defined in previous section. The links in the transcript will be of size at most $O(T)$ therefore every entry of transcript takes at most $O(\log T)$ bits. Let us set Σ_{in} to be large enough to hold $Ent = \frac{L}{\varepsilon'} = O(\frac{1}{\varepsilon^2})$ entries of the transcript. Thus $\log |\Sigma_{in}| = O_{\varepsilon,c}(\log T)$. Let $n = \frac{T}{\varepsilon'} = O(\frac{T}{\varepsilon})$ be the number of rounds of the protocol. By Theorem 20 there exist $C : \Sigma_{in}^n \to \Sigma_{out}^n$ which is $(L, \varepsilon') = (O(\frac{1}{\varepsilon}), O(\varepsilon))$ list tree code with $\log |\Sigma_{out}| = O_\varepsilon(\log |\Sigma_{in}|) = O_{\varepsilon,c}(\log T)$.

Remark 24. *Note that we have here three alphabets:*

First is the alphabet of the original protocol Σ_T of size T^c.

Second is Σ_{in} which corresponds to a non-encoded single message that we are going to send during each round.

The third one is Σ_{out} which corresponds to an alphabet which we are going to send over noisy channel.

Let C be encoding and for $w \in \Sigma_{out}^i$, $D(w)$ will return $List_i(w, C, \varepsilon)$. At every round i Alice will decode received codeword $w \in D(\Sigma_{out})^i$ and will get list of possible transcripts of Bob $B_1, B_2, \dots B_k$ (Bob will similarly decode Alice's message). For every such transcript Alice will calculate $E(B_i)$ edges that Bob sent and send next edge from $X \subset \mathcal{X}$ by first adding this edge to transcript and then encoding transcript with list-tree code.

The protocols of Alice and Bob will be symmetric so we will introduce only protocol for Alice. We will denote by w_A, w_B received codewords by Alice and Bob correspondently. Alice will maintain transcript A which will be initialized by empty set. Formally at block i Alice will,

Input: $w_A \in D(\Sigma_{out})^i$

1. Calculate $Decode(w_A) = List_i(w_A, C, \varepsilon) = \{B_1, B_2, \ldots, B_k\}$
2. For $j = 1, \ldots, \min\{k, Ent\}$ do
   (a) If $E(B_j) \cup X$ has unique path from root and let e be a last edge on this path then $Add(A, e)$
3. $End - round(A, i+1)$.
4. Send $C(A)[i+1]$ to Bob.

We will run the protocol n blocks. At the end Alice will construct $PrefixList(w_A, C, \varepsilon)$.

Definition 25. *For a node $v \in PrefixList(w_A, C, \varepsilon)$ we will call it* output node if $E(v) \cup X$ has unique path from root and this path ends at a leaf ℓ and none of predecessors of v has this property. The output of the output node is the leaf ℓ.

In other words, the output node v is the node at which corresponds to some possible output. Let $v, v_s \in PrefixList(w_A, C, \varepsilon)$ such that v_s is a successor of v Let v be node at depth i and v_s at depth j and let $s \in \Sigma_{out}^{j-i+1}$ to be string corresponding to path from v to v_s. Then $agr(v \to v_s, w) \triangleq agr(s, w[i, \ldots, j])$. For output Alice will do the following:

1. Construct $PrefixList(w_A, C, \varepsilon)$.
2. For every output node $v \in PrefixList(w_A, C, \varepsilon)$
3. If exists successor of v, $v_s \in PrefixList(w_A, C, \varepsilon)$ such that $agr(v \to v_s, w) > T$ then output v.

In other words Alice will output all answers which corresponds to communication and has at least T places of agreement with w after output node.

Lemma 26. *The protocol above solves the task of list decoding pointer jumping problem with list size $O(\frac{1}{\varepsilon})$*

The rest of the section will prove this lemma. Let us denote by $PL_A = PrefixList(w_A, C, \varepsilon)$ and by $PL_B = PrefixList(w_B, C, \varepsilon)$

Claim 27. *Output list size of the protocol is at most $O(\frac{1}{\varepsilon})$.*

Proof. By definition of list tree codes agreement between $C(PL_A)$ and w_A is less than $\varepsilon' Ln$. In order for some node v to be in the output list we require that there will be an agreement of at least $T = \varepsilon' n$, which are appears after we reached output node. Note that by definition output nodes can not be successors of each other. Thus each different node from the output list corresponds to a disjoint agreement of at least T between $C(PL_A)$ and w_A. Thus the size of the output list can not be larger than $\frac{\varepsilon' Ln}{T} = L = O(\frac{1}{\varepsilon})$. □

Now the rest of the section we will prove that if there is at most $1 - \varepsilon$ error then the correct answer is in the list.

Definition 28. *Let P be the unique path from the root in $X \cup Y$. We say that we advance in the block i if at this block Alice or Bob adds a new edge from P to his transcript.*

Definition 29. *We say that the block i is good if:*

1. *At block i both Alice and Bob decoded correctly each other's messages i.e., $B_{[1,\ldots i-1]} \in List_{i-1}(w_A, C, \varepsilon)$ and $A_{[1,\ldots i-1]} \in List_{i-1}(w_B, C, \varepsilon)$.*
2. *The list size of both Alice and Bob at block i was less than Ent.*

Claim 30. *At every good block we advance.*

Proof. By definition at the good block both Alice has transcript B of Bob and Bob has transcript A of Alice. Moreover, the list in this block is small, and thus Alice will add last edge from $E(B) \cup X$ and Bob will add $E(A) \cup Y$. Let e be a first edge from P which does not appear in $E(A) \cup E(B)$. Let us assume w.l.o.g. that $e \in X$. Then e is a last edge from $E(B) \cup X$. Thus we will advance in this block. □

Thus now we will need to prove that there are many good blocks. We first we will note that there are not too many blocks which do not satisfy the second condition and then we will prove that there are many blocks which do satisfy the first condition.

Claim 31. *There are at most $O(\varepsilon' n)$ blocks which do not satisfy the second condition of being good.*

Proof. Note that by definition of $PrefixList$, $\sum_{i=1}^{n} List_i(w_A, C, \varepsilon) \leq PrefixList(w_A, C, \varepsilon)$. Since at every bad block we have that $List_i(w_A, C, \varepsilon) \geq Ent = \frac{L}{\varepsilon'}$ (or same for Bob.) We have at most

$$\frac{(|PL_A| + |PL_B|)}{Ent} \leq \frac{2Ln}{Ent} = 2\varepsilon' n \leq O(\varepsilon' n)$$

such blocks. □

Lemma 32. *Let $c_A = C(A),\ c_B = C(B)$ be a messages that Alice (Bob) has sent and w_A, w_B is what they received. Let us assume that:*

$$\sum_{i=1}^{n} max(d(w_B[i], c_A[i]), d(w_A[i], c_B[i])) \leq (1-\varepsilon)n\ ,$$

then there are at least $(\varepsilon - \varepsilon')n$ blocks i at which first condition of being good is satisfied.

The lemma will follow for the following claim.

Claim 33. *Let $w \in D(\Sigma)^n, c \in \Sigma^n$ $d(w, c) < (1-\beta)n$. Define the set*

$$S_\alpha(n) \triangleq \{i \leq n : \delta_s(w[1, \ldots, i], c[1, \ldots, i]) \leq 1 - \alpha\}$$

Then $|S_\alpha(n)| \geq (\beta - \alpha)n$.

Proof of Lemma 32. Recall that by definition of $List_i$ we have that $B[1,\dots i]$ in the $List_i(w_A, C, \varepsilon)$ if

$$\delta_s(w_A[1,\dots i], c_B[1,\dots i]) \leq 1-\varepsilon' \ ,$$

and the same for Bob. For each i let us define $w[i], c[i]$ to be either $w_B[i], c_A[i]$ or $w_A[i], c_B[i]$ such that $d(w[i], c[i]) = max(d(w_B[i], c_A[i]), d(w_A[i], c_B[i]))$. Then we are given that $d(w, c) \leq (1-\varepsilon)n$. Then using Claim 33 we will get that there are at least $(\varepsilon - \varepsilon')n$ indexes i such that $\delta_s(w_{[1,\dots,i]}, c_{[1,\dots,i]}) \leq 1-\varepsilon'$. For every such i the first condition holds since by definition of w, c it holds that

$$\delta_s(w_A[1,\dots i], c_B[1,\dots i]) \leq \delta_s(w[1,\dots,i], c[1,\dots,i]) \leq 1-\varepsilon',$$

And the same for Bob, i.e.,

$$\delta_s(w_B[1,\dots i], c_A[1,\dots i]) \leq \delta_s(w[1,\dots,i], c[1,\dots,i]) \leq 1-\varepsilon'$$

□

Proof of Claim 33. We will prove the claim by induction. For $n = 1$ the claim trivially follows since in this case if $\beta > \alpha$, $d(w, c) < 1-\beta < 1-\alpha$.

If we know claim for all strings up-to $n-1$ then we separate in two cases if $n \in S_\alpha$ then We have that $|S_\alpha(n)| = |S_\alpha(n-1)| + 1$, by induction on $w_{[1,\dots,n-1]}, c_{[1,\dots,n-1]}$ we know that

$$d(w_{[1,\dots,n-1]}, c_{[1,\dots,n-1]}) \leq (1-\beta)n = (1 - \frac{\beta n - 1}{n-1})(n-1)$$

Thus $|S_\alpha(n-1)| \geq \beta n - 1 - \alpha(n-1) \geq (\beta-\alpha)n - 1$. Therefore $S_\alpha(n) \geq (\beta-\alpha)n$.

If $n \notin S_\alpha$ then by definition there exists an n' such that $d(c_{[n',\dots,n]}, w_{[n',\dots n]}) > (n-n'+1)(1-\alpha)$ thus

$$\begin{aligned} &d(w_{[1,\dots n'-1]}, c_{[1,n'-1]}) < (1-\beta)n - (n-n'+1)(1-\alpha) = \\ &= (n'-1) - (\alpha(n'-1) + (\beta-\alpha)n). \end{aligned}$$

By induction on $n'-1$ we get that

$$|S_\alpha(n)| \geq |S_\alpha(n'-1)| \geq \alpha(n'-1) + (\beta-\alpha)n - \alpha(n'-1) = (\beta-\alpha)n$$

□

Be Claim 31 and Lemma 32 we get:

Corollary 34. *If our protocol corrupted in at most $(1-\varepsilon)n$ blocks then there are at least $\varepsilon n - O(\varepsilon' n)$ good blocks.*

Now we are ready to prove that $v(X \cup Y)$ is in the list.

Claim 35. *$v(X \cup Y)$ is in the list.*

Proof. Let $P \subset PL_A$ be a path in PrefixList that corresponds to the correct path(i.e. it is actually the message that Bob has sent). Let e_1 be a length of P. Then in order to prove that the correct output is in the list we have to show there exists output node on P at some place e_0 and that agreement of P and w_A between e_0 and e_1 is at least T.

If $c_1 < \frac{1}{2}$ then $\varepsilon n - T - \varepsilon' n > 0$. Let i_0 be a place such that we have agreement $\varepsilon n - T - \varepsilon' n$ before i_0 and agreement $T + \varepsilon' n$ after i_0 i.e.,

$$\sum_{i=1}^{i_0} max(d(w_B[i], c_A[i]), d(w_A[i], c_B[i])) = i_0 - \varepsilon n + T + \varepsilon' n \ ,$$

There are at least $\varepsilon n - T - \varepsilon' n - O(\varepsilon' n) = \frac{T}{c_1} - O(T)$ good blocks before i_0. By definition of i_0 there is at least $T + \varepsilon' n$ agreement after i_0.

Thus by taking c_1 small enough we will get at least T good blocks before i_0. Since at every good block we advance we will get to output node after T good blocks in particular output node with correct answer is before i_0.

Let us show that there exists i_1 such that $\delta_s(w_A[1, \dots i_1], c_B[1, \dots i_1]) \leq 1 - \varepsilon'$ and such that agreement before i_1 is at least $\varepsilon n - \varepsilon' n$. If $\delta_s(w_A[1, \dots n], c_B[1, \dots n]) \leq 1 - \varepsilon'$ then take $i_1 = n$ and we are done. Else by definition of δ_s there exists an i such that

$$d(w_A[i, \dots n], c_B[i, \dots n]) > (1 - \varepsilon')(n - i + 1)$$

Let us take $i_1 = i_2 - 1$ where $i_2 \neq 1$ is a minimal value i satisfying equation above. We claim that $\delta_s(w_A[1, \dots i_1], c_B[1, \dots i_1]) \leq 1 - \varepsilon'$ else there exists $i_3 < i_2$ such that

$$d(w_A[i_3, \dots i_2 - 1], c_B[i_3, \dots i_2 - 1]) > (1 - \varepsilon')(i_2 - i_3)$$

Therefore

$$d(w_A[i_3, \dots n], c_B[i_3, \dots n]) =$$
$$d(w_A[i_3, \dots i_2 - 1], c_B[i_3, \dots i_2 - 1]) + d(w_A[i_2, \dots n], c_B[i_2, \dots n]) >$$
$$(1 - \varepsilon')(i_2 - i_3 + n - i_2 + 1) = (1 - \varepsilon')(n - i_3 + 1)$$

Contradiction to minimality of i_2. Note also that by definition of i_1 agreement from i_1 to n is at most $\varepsilon' n$. Thus agreement between i_0 which is after output node and i_1 point in PrefixList is at least T. In other words $agr(w_A[i_0, \dots i_1], c_B[i_0, \dots, i_1]) \geq T$ and as just proved $c_A[1, \dots i_1]$ is in $PrefixList(w_A, C, \varepsilon)$. Thus the correct codeword should be in the list. □

7 Constant Alphabet

The goal of this section is to prove the following theorem:

Theorem 36. *For every $\varepsilon, c > 0$ there exists a protocol π is resilient to $1 - \varepsilon$-symmetric noise and which solves the problem of list decoding of $PJP(T, 2)$ with list of size $O(\frac{1}{\varepsilon})$. Moreover the protocol π runs in $O(\frac{T}{\varepsilon})$ rounds and in each round sends $O_\varepsilon(1)$ bits.*

We start by observing that Theorem 3 is a corollary of this theorem.

Proof of Theorem 3. Since Pointer Jumping Problem is complete for communication complexity any protocol π with $CC(\pi) = T$ can be converted to Pointer Jumping Problem of depth $2T$. Then from Theorem 36 it follows that there exists a protocol π' which solves list decoding Pointer Jumping Problem and resilient to $\eta < 1$ noise with $CC(\pi') = O_\eta(T)$ and with list size $O(\frac{1}{1-\eta})$. □

As in previous section define $\varepsilon' = c_1\varepsilon$ for some small constant c_1 to be defined later. Define also $L = \frac{1}{\varepsilon'} + 1$.

In case of constant alphabet we will have Σ_{in} to be large enough to contain $Ent = \frac{L}{\varepsilon'}$ transcript entries with link of size $MLSize = \frac{L^2}{\varepsilon'^2} = O(\frac{1}{\varepsilon^4})$. Note that the size of Σ_{in} is a constant depending on ε. By Theorem 20 there exist $C : \Sigma_{in}^n \to \Sigma_{out}^n$ which is $(L, \varepsilon') = (O(\frac{1}{\varepsilon}), O(\varepsilon))$ list tree code with $\log|\Sigma_{out}| = O_\varepsilon(\log|\Sigma_{in}|) = O_\varepsilon(1)$.

Let $n = \frac{T}{\varepsilon'} = O(\frac{T}{\varepsilon})$ be the number of rounds of the protocol.

In case we will send the link of size larger than $MLSize$ we will say that we send a long link. In general it may happen that we will need to send a long link which will not fit in the space we dedicated for it. In this case we will send it anyway at cost of not sending other edges we wanted to send in this and maybe several next rounds (up to $\log n$ rounds). We are going to modify our protocol a little bit in order to be able to prove that the sum of all links in our protocol is at most the size of $PrefixList < Ln$. At last we going to conclude that there are at most $O(\varepsilon' n)$ blocks at which we are sending a long link and thus those links will not disturb our protocol too much.

Note that in list decoding regime adversary can corrupt up-to $1 - \varepsilon$ fraction of communication therefore we need to be very careful when we are sending a long links since most of the time we will respond to an adversary communication. We are going to do the following adjustment to our protocol. Note that in Section 6 every edge Alice (or Bob) sent was a response on the decoding of some transcript which on his side corresponds to some node in $PrefixList(w_A, C, \varepsilon)$. In this protocol we are going to remember for every edge in our transcript for which node in $PrefixList$ it corresponds and we will allow only to send edge e at node $v \in PrefixList$ only in case that grandparent of e was sent at predecessor of v. As we will see later this way we will prevent from Alice to send too many long links. As we will see the sum of all links send in this way will approximately the size of the $PrefixList$. It may happen that some node v has many successors which are far a way who are trying to send link to his grandfather at node v. In order that the sum of links will be not too large we need a cousin trick[3].

Now we let us write pseudo-code for Alice's communication on the i'th block. Alice will maintain her PrefixList tree PL_A and for some nodes of the tree she will associate edges from her set of edges X.

input: $w_A \in D(\Sigma_{out})^i$

1. Calculate $Decode(w_A) = List_i(w_A, C, \varepsilon) = \{B_1, B_2, \ldots, B_k\}$
2. Update tree PL_A from $B_1, \ldots B_k$.
3. For $j = 1 \ldots \min\{k, Ent\}$ do:
   (a) If $Size(A) > (i+1)\log\Sigma_{in}$ then Send C(A)[i+1] to Bob and continue to next round.
   (b) Set $v \in PL_A$ to be node corresponding to B_j.
   (c) Let $u \in PL_A$ be the last predecessor (i.e. closest to v) of v associated with some edge $e_{\text{grandparent}} \in \mathcal{T}$.
   (d) If $E(B_j)$ contains exactly one son of $e_{\text{grandparent}}$(let us call it e_{father}) then:
      i. Let $e_{\text{son}} \in X$ to be son of e_{father}.

[3]Recall that we send a link to the closest either grandfather or cousin

ii. $Add(A, e_{son})$.

iii. Associate e_{son} with v.

4. $End - round(A, i+1)$

5. Send $C(A)[i+1]$ to Bob and continue to next round.

Also as with case of large alphabet we will run the protocol $n = \frac{T}{\varepsilon'} = O(\frac{n}{\varepsilon})$ rounds. Output will be the same

1. Construct $PrefixList(w_A, C, \varepsilon)$.

2. For every output node $v \in PrefixList(w_A, C, \varepsilon)$

3. If exists successor of v, $v_s \in PrefixList(w_A, C, \varepsilon)$ such that $agr(v \to v_s, w) > T$ then output v.

8 Analysis

The goal of this section is to prove

Theorem 37. *The protocol from Section 7 is resilient $(1-\varepsilon)$-symmetric noise and it solves the task of list decoding Pointer Jumping Problem of depth T with list size $O(\frac{1}{\varepsilon})$. The communication complexity of this protocol is $O_\varepsilon(T)$.*

The analysis in this section is similar to the one we had for large alphabet. The bound on the list size is exactly the same. If some node v from PL_A or PL_B has an edge associated to it we will say that we speak at node v. We will have here several differences. First we say that we advance at block i if Alice or Bob spoke at node associated with a correct transcript. Let $P_A \subset PL_A$ and $P_B \subset PL_B$ be the paths from the root which corresponds to transcripts of Bob and Alice, respectively.

Remark 38. *Note that we have here three trees $\mathcal{T}, PL_A, PL_B$ as well three "correct" paths $P \subset \mathcal{T}, P_A \subset PL_A, P_B \subset PL_B$.*

Then we are saying that we advance at block i if $P_A[i]$ or $P_B[i]$ is associated with an edge in other words its mean that at round i either Alice or Bob got the correct transcript in his list and send an edge as a respond to this transcript. Let X_i be a set of edges associated with $P_A[1, \dots i]$ and Y_i be the set of edges associated with $P_B[1, \dots i]$. As we will see $X_i \cup Y_i$ is a rooted sub-path of P thus if we advanced T times then we will reach the end of P thus if we will advance T times the correct codeword will be in the list. The following lemma is essentially says that if we will look on the "correct paths" of Alice and Bob then we will see that at this path the correct communication.

Lemma 39. *The set $X_i \cup Y_i$ is a rooted sub-path of P.*

Proof. Informally the claim follows since Alice can send a new edge at node v only after she will receive a respond from the last edge she sent. Now let us try to write it more formally. Let us prove this by induction. At round 0 the statement trivially follows. If at round i nobody spoke at nodes $P_A[i], P_B[i]$ then $X_i = X_{i-1}, Y_i = Y_{i-1}$ and again it trivially follows. If at block i Alice spoke at $P_A[i]$(or the same proof for Bob $P_B[i]$), let $j < i$ be last time that Alice spoke at $P_A[j]$ let $e_j \in \mathcal{T}$

be the corresponding edge associated with this node. Then according to our protocol e_j will be set to be $e_{\text{grandparent}}$. Note that by definition of $P_A[i]$ it corresponds to the correct transcript. Thus $E(P_A[i]) \subset Y$ therefore $e_{\text{father}} \in Y$. By induction we assume that e_j is the last node of $X_j \cup Y_j$ and it is on the path P. Therefore e_{father} next node on P and e_{son} is the node after it. □

Note that if the following three conditions are satisfied, we advance at block i.

1. At block i both Alice and Bob decoded correctly each other's messages i.e., $B[1, \dots i-1] \in List_{i-1}(w_A, C, \varepsilon)$ and $A[1, \dots i-1] \in List_{i-1}(w_B, C, \varepsilon)$.
2. The list size at block i was less than Ent.
3. We did not send long link at block i and we did not send long link at previous blocks which we have not finished sending by block i.

From Lemma 32 it follows that there are at least $\varepsilon n - O(\varepsilon' n)$ blocks that satisfy the first condition. As in case with large alphabet we know that there are at most $O(\varepsilon' n)$ blocks which do not satisfy second condition. All that is left for us is to bound the number of blocks that do not satisfy the third condition.

The following lemma takes care of the third issue.

Lemma 40. *The number of blocks at which long links where being sent is at most $O(\varepsilon' n)$.*

We will dedicate the rest of the section to proving this lemma.

Every edge that Alice has sent is associated with some node from PL_A. For every node $v \in PL_A$ for which we associated an edge we let $link(v)$ be a offset in transcript of this edge.

Definition 41. *Let us assume that the i-th edge e_i in the transcript was sent at node v and that it was sent at round j. Suppose e_i in the transcript has a link to e_{i-t}. Let us assume that e_{i-t} was sent at round k; then we define length of the link by $j - k$. We denote it by $|link(v)|$.*

Note that the length of the link is not much different than the link itself.

Claim 42. $link(v) \leq Ent \cdot (|link(v)| + 1)$.

We denote by $PL_A(v)$ to be a subtree of PL_A rooted at v.

Claim 43. *Let $v \in PL_A$ be a node at which we have sent an edge then*

$$\sum_{u \in PL_A(v) \setminus \{v\}} |link(u)| \leq |PL_A(v)|.$$

Proof. We are going to prove the lemma by induction. For leafs it is trivially true. Denote by

$$S(v) = \sum_{u \in PL_A(v) \setminus \{v\}} |link(u)|.$$

Next assume that at node v we have sent an edge $e \in M$. Let us consider all nodes in $PL_A(v)$: $u_1, u_2, \dots u_k \in PL_A(v)$ at which we have sent grandchildren of e. First note that according to

our protocol on the nodes between v and u_i we did not sent any edges. Therefore using this and induction assumption we get

$$S(v) = \sum_{i=1}^{k} |link(u_i)| + \sum_{i=1}^{k} S(u_i) = \sum_{i=1}^{k} |link(u_i)| + \sum_{i=1}^{k} |PL_A(u_i)|.$$

Let us also assume u_i are sorted according the the order they appear in the transcript. Then u_1 will send a link to v (or maybe somebody later, and set that this is his grandparent), u_2 will send a link to u_1, $\ldots u_k$ will send the link to u_{k-1} (or maybe somebody later) and all these nodes have links to their cousins. Thus

$$\sum_{i=1}^{k} |link(u_i)| \leq dist_{PL_A}(v, u_k).$$

Therefore

$$S(v) \leq \sum_{i=1}^{k} |PL_A(u_i)| + dist_{PL_A}(v, u_k).$$

Note that since none of u_i's are predecessor of another u_j $PL_A(u_i)$ are disjoint subtrees of $PL_A(v)$. Also note that path from v to any u_i does not belong to any of these trees thus we get that

$$\sum_{i=1}^{k} |PL_A(u_i)| + dist_{PL_A}(v, u_k) \leq |PL_A(v)| \ .$$

And the claim follows. □

Now we are ready to prove lemma Lemma 40.

Proof of Lemma 40. Recall that we assume that every link l takes $2 \log l + 2$ space of storage. The number of blocks at which long link of size $l > MLSize$ is sent is at most $\frac{\log l+2}{\log MLSize} + 2 < \frac{l}{MLSize} + 3$. From Claims 42, 43 it follows that:

$$\sum_{u \in PL_A} link(u) \leq Ent \cdot |PL_A| \leq \frac{L^2}{\varepsilon'} n \ .$$

And the same for PL_B. Therefore the number of long links is bounded by: $LL = \frac{2L^2}{\varepsilon' MLSize} n = O(\varepsilon' n)$.

The number of blocks at which long links where sent is bounded by

$$\sum_{u \in PL_A} \frac{link(u)}{MLSize} + \sum_{u \in PL_B,} \frac{link(u)}{MLSize} + 3LL \leq O(\frac{L^2}{\varepsilon' MLSize} n) + O(\varepsilon' n) = O(\varepsilon' n)$$

□

We will use the following theorem for unique decoding:

Theorem 44. *If some integer $i_0 \leq n$ it holds that*

$$\sum_{i=1}^{i_0} min(agr(w_B[i], c_A[i]), agr(w_A[i], c_B[i])) \geq T + O(\varepsilon' n) = O(\varepsilon n),$$

Then there exists output node on both paths P_A, P_B[4] and this node is located before level i_0.

Proof. To prove this theorem it is enough to show that there are at least T rounds before time i_0 at which we advance. This follows from Lemma 32, Lemma 40 and Claim 31. □

We are going to use the following corollary of this theorem:

Corollary 45. *If output communication point on P_A or P_B is located after s then*

$$d(w_B[1, \dots s], c_A[1, \dots, s]) + d(w_A[1, \dots, s], c_B[1, \dots, s]) \geq s - O(\varepsilon n) \ .$$

Claim 46. *$v(X \cup Y)$ is in the list.*

Proof. The proof here is almost the same as in Claim 35. First find i_0 such that we have agreement $\varepsilon n - T - \varepsilon' n$ before i_0 and agreement $T + \varepsilon' n$ after i_0. From Theorem 44 it holds that by taking c_0 small enough we will reach output communication node before i_0. Next proceed the proof exactly as in Claim 35. □

Proof of Theorem 3. We are already know that the protocol from Section 7 solves correctly the task of list decoding pointer jumping problem. Since pointer jumping problem is complete for communication protocols we In order to prove the theorem we will execute the protocol above. Note that number of rounds is $O_\varepsilon(T)$. Also note that Σ_{in} is of constant size (depending on ε, but not on T), thus Σ_{out} by Theorem 20 is also of constant size. □

9 Optimality of Our Results for List Decoding

In this section we are going to show that for any α, β with $\alpha + \beta \geq 1$ there exists protocol such that any list decoding protocol resilient to (α, β) noise with linear communication complexity must have exponential list size.

We will show this by showing that if one can perform some task with (α, β) noise than one can perform the same noiseless task in 3 rounds.

Theorem 47. *Let $\alpha, \beta \in [0, 1]$ such that $\alpha + \beta \geq 1$. Let us assume that there exists a protocol performing task T which is resilient to (α, β) noise. Then there exists a 3-round noiseless protocol with the same communication complexity performing task T.*

Proof. Let us assume that during the protocol Alice sent n_A messages and Bob sent n_B messages. Let us also define $A(i), B(i)$ be number of messages sent by Alice and by Bob before round i. Consider $f(i) = \frac{A(i)}{n_A} - \frac{B(i)}{n_B}$, let i_{max} be a point at which this function is maximal. If $n_A - A(i_{max}) \leq \alpha n_A$ and $B(i_{max}) \leq \beta n_B$ then Eve can destroy communication of Bob before i_{max} and communication of Alice after i_{max} and thus we will get protocol equivalent to a two round protocol

[4]Recall that this are the "correct" paths

where Alice send all her information in rounds $1, \ldots, i_{max}$ and Bob sends all his information in rounds $i_{max}, \ldots, n$.

Now assume that $B(i_{max}) > \beta n_B$. Let i_0 be a point such that $B(i_0) = B(i_{max}) - \beta n_B$. In this case Eve will corrupt the communication of Alice in the interval $[0, \ldots i_0]$, the communication of Bob in the interval $[i_0, \ldots, i_{max}]$, and the communication of Alice in the interval $[i_{max}, \ldots, n_A + n_B]$. This way we will obtain protocol which is equivalent to a 3 round protocol. By definition of i_0 in the interval $[i_0, i_{max}]$ there are exactly β fraction of Bob's communication. By definition of i_{max} we have that

$$\frac{A(i_{max})}{n_A} - \frac{B(i_{max})}{n_B} \geq \frac{A(i_0)}{n_A} - \frac{B(i_0)}{n_B} .$$

Rearranging this inequality we will get

$$\frac{A(i_{max}) - A(i_0)}{n_A} \geq \frac{B(i_{max}) - B(i_0)}{n_B} = \beta$$

Thus Alice communication in the intervals everywhere except in $[i_0, i_{max}]$ is at most $1 - \beta \leq \alpha$.

Now let us assume that $n_A - A(i_{max}) > \alpha n_A$. Let i_1 be such that $A(i_1) - A(i_{max}) = \alpha n_A$. In this case we will destroy communication of Alice in the interval $[i_{max}, i_1]$ and the communication of Bob everywhere else. In this case from definition of i_{max} we will get

$$\alpha = \frac{A(i_1) - A(i_{max})}{n_A} \leq \frac{B(i_1) - B(i_{max})}{n_B}$$

Thus communication of Bob in the interval $[i_{max}, i_1]$ is at least α therefore his communication on other two intervals will be at most $1 - \alpha \leq \beta$. □

Now we are ready to prove Theorem 2.

Theorem 48 (Theorem 2)**.** *For every α, β such that $\alpha + \beta \geq 1$. Let π be a protocol which is resilient to (α, β) adversarial noise and which solves list decoding problem of Pointer Jumping Problem of depth T with list of size $L = exp(o(T))$. Then $CC(\pi) = \exp(\Omega(T))$.*

Proof. If protocol π is resilient to (α, β) noise with $\alpha + \beta \geq 1$ then by Theorem 47 we know that there exists a three round protocol which solves list decoding of Pointer Jumping Problem. Consider a protocol which at the end outputs a random codeword from the list. Then this protocol solves the Pointer Jumping Problem with probability at least $\frac{1}{L}$. From Corollary 9 it follows that this protocol must have $\exp(\Omega(T))$ communication complexity. □

Part II
Unique Decoding

In this part we are going to consider the following question: assume that Alice and Bob want to perform an interactive communication protocol π and assume that Eve can adversary corrupt up-to α fraction of messages sent by Alice and β fraction of coordinates sent by Bob. We want to find a region at which we still can perform interactive communication in the unique decoding regime. The answer to this question is not obvious and quite surprising.

10 Unique Decoding up-to $\alpha + 2\beta < 1$.

In this section we are going to show how to perform unique decoding up-to $\alpha + 2\beta < 1$ and $2\alpha + \beta < 1$. Although, as we will see from the next sections, this is not optimal. The algorithm described here gives essential ideas for the next sections.

Let us assume that $\beta < \frac{1}{2}(1-\alpha)$ we are going to show that in this case Alice will output the correct answer. Let $\varepsilon = 1 - 2\beta - \alpha$. Note that $\varepsilon > 0$. Let $\varepsilon' = c\varepsilon$ for some small constant c to be defined later. The protocol is very simple we will perform list decoding protocol from previous section which is resilient to $(1-\varepsilon')$ noise, but instead of outputting a list at the end we will output the closest answer i.e., we will find $c_{output} \in C$ such that

$$d(c_{output}, w_A) = \min\{d(c, w_A) : c \in C\} .$$

Here C is the set of all codewords. We will calculate $E(c_{output})$ and output $v(X \cup E(c_{output}))$.

Now let us show why this will be the correct answer. First we will need the following lemma about $(\varepsilon', \frac{1}{\varepsilon'} + 1)$ list decodable tree codes:

Lemma 49. *Let C be $(\varepsilon', \frac{1}{\varepsilon'} + 1)$ list decodable tree code . For every $x, y \in \Sigma_{in}^n$ let s be first index where $x[s] \neq y[s]$. Then $d(C(x), C(y)) \geq n - s - 2\varepsilon' n$.*

Proof. Let us take w to be $C(x)$ on the first $n - \varepsilon'$ locations and to be $C(y)$ at the last $\varepsilon' n$ locations. Then $\delta_s(w, C(y)) \leq 1 - \varepsilon'$ and also $\delta_s(w[1, \dots n - \varepsilon' n], C(x)[1, \dots, n - \varepsilon' n]) = 0 \leq 1 - \varepsilon'$. By definition of list tree code we know that agreement between w and PrefixList is at most $\varepsilon' L n = (1 + \varepsilon')n$. Note that agreement between w and $C(x)$ on the first $n - \varepsilon' n$ locations is $n - \varepsilon' n$. Note also that starting level s, $C(x)$ and $C(y)$ represent different branches in the PrefixList. Agreement between w and $C(y)$ starting level s is at least the agreement between $C(x)_{[i,\dots,n]}$ and $C(y)[s, \dots, n]$. Thus

$$n - \varepsilon' n + agr(C(y)[s, \dots, n], C(x)[s, \dots, n]) \leq \varepsilon' L n = n + \varepsilon' n$$

Thus $agr(C(y)[s, \dots, n], C(x)[s, \dots, n]) \leq 2\varepsilon' n$ □

Let us assume by contradiction that protocol outputs wrong answer. Let us assume that Alice outputs the wrong answer. Let $c_B \in \Sigma_{out}^n$ be a codeword which was sent by Bob. Let us assume that w_A was the codeword received by Alice. Assume assume that c_{output} is the closest codeword to w_A. There are two important points on c_B one is a "split" point s to be a first place where $c_{output}[s] \neq c_B[s]$ and an other is e_{end} output communication point (we will show soon that exists one on the "correct" path c_B). Observe that if output node is located before split point s then we output the correct answer. Thus by contradiction assumption $s < e_{end}$. The proof now follows

from Lemma 49 which will give lower bounds on s and from Corollary 45 which will give upper bounds on e_{end} and thus also on s.

Let us define $B_1 = d(c_A[1,\dots s], w_B[1,\dots s])$ and $B_2 = d(c_A[s+1,\dots n], w_B[s+1,\dots n])$ note that $B_1 + B_2 \leq \beta n$. From the Lemma 49 we know that $d(c_{output}, c_B) \geq n - s - O(\varepsilon' n)$; thus, since w_A is closer to c_{output} than to c_B we have that the number of errors in Bob's messages in last $n-s$ rounds was at least $\frac{n-s-O(\varepsilon' n)}{2}$. Thus we have that

$$B_2 \geq \frac{n - s - O(\varepsilon' n)}{2}$$

Rewriting thus we get

$$s \geq n - 2B_2 - O(\varepsilon' n) \tag{2}$$

On other hand from Corollary 45 it follows that

$$B_1 + \alpha n \geq d(w_B[1,\dots s], c_A[1,\dots,s]) + d(w_A[1,\dots,s], c_B[1,\dots,s]) \geq s - O(\varepsilon' n)\ .$$

Thus we got that

$$B_1 + \alpha n \geq n - 2B_2 - O(\varepsilon' n)$$

Rewrite this and we will get that

$$2\beta n + \alpha n \geq B_1 + 2B_2 + \alpha n \geq n - O(\varepsilon' n)\ .$$

Thus we have got that $2\beta+\alpha > 1-O(\varepsilon')$ thus by taking c small enough we will get that $2\beta+\alpha > 1-\varepsilon$ contradiction to the definition of ε.

11 Repetition Power of the Protocol

In order to construct a protocol which can handle optimal fraction of errors we need to define the repetition power of a protocol. In the case of one way communication repetition power k just means that we will repeat each symbol k times. In case of interactive protocol with n rounds. Let $a : [n] \to \mathbb{N}$; we will do the same: we will just send the symbol of the ith round $a(i)$ times. Unlike one-way communication here we need to explain how Alice (Bob) acts when at round i she receives $a(i-1)$ symbols. Since these are all these are soft symbols (i.e., elements from $D(\Sigma)$) Alice will just calculate the average of all received distributions. It will be more convenient for us to give a definition with $a(i), b(i)$ where $a(i)$ is the number of symbols which Alice send at her i'th round and $b(i)$ the number of symbols Bob sent at his i-th round.

Let π be a protocol with $2n$ alternating rounds of Alice and Bob over alphabet Σ_{out}. Let as assume that Alice talks at rounds $2i-1$ and Bob talks at rounds $2i$ for $i \in [n]$. At every round i of the protocol π Alice or Bob receives symbol from $D(\Sigma_{out})$ and outputs symbol in Σ_{out}.

Definition 50. *Let $a : [n] \to \mathbb{N}$, $b : [n] \to \mathbb{N}$ be any functions. The* repetition power $\pi^{a,b}$ *of the protocol π. Is the following protocol: In the protocol $\pi^{a,b}$ Alice will receive at round $2i-1$ $b(i-1)$ symbols $w_1, \dots w_{b(i-1)} \in D(\Sigma_{out})$ calculate average $w = \frac{1}{b(i-1)} \sum_{i=1}^{b(i-1)} w_i$. Run protocol π with input symbol w. Get output $c_A[i]$ and send $a(i)$ copies of $c_A[i]$ to Bob. Bob at round $2i$ will receive $a(i)$ symbols from $D(\Sigma_{out})$ run π on their average and sent $b(i)$ copies of output of the protocol π.*

First note the simple connection between communication costs of π and $\pi^{a,b}$

Lemma 51. *Let π be a protocol with $2n$ alternating rounds of Alice and Bob over alphabet Σ_{out}. Then communication cost of $\pi^{a,b}$ is $\sum_{i=1}^{n} a(i) + b(i)$.*

Proof. Follows from the definition. □

The following lemma makes a connection between amount of noise in protocol π and protocol $\pi^{a,b}$.

Lemma 52. *Let us assume that c'_A, c'_B what Alice and Bob sent in the protocol $\pi^{a,b}$. Let w'_A, w'_B is what they have received in protocol $\pi^{a,b}$. Let c_A, c_B, w_A, w_B be a corresponding values on which the protocol π was simulated the the following holds*

$$d(c'_A, w'_B) = \sum_{i=1}^{n} a(i) d(c_A[i], w_B[i]) \ ,$$

And the same for Bob i.e.,

$$d(c'_B, w'_A) = \sum_{i=1}^{n} b(i) d(c_B[i], w_A[i]) \ .$$

This lemma essentially shows that using repetition we can give different weight to the symbols of our protocol.

Proof. Let us consider c'_A, w'_B as matrix vectors with $c'_A[i,j]$ for $1 \leq i \leq n, 1 \leq j \leq a(i)$. Then $c'_A[i,j] = c_A[i]$ for every j. It also holds that $w_B[i] = \frac{1}{a(i)} \sum_{j=1}^{a(i)} w'_B[i,j]$(note here we are summing up probability distributions). Next note that for every $\sigma \in \Sigma$ and k distributions $\sigma_1, \ldots, \sigma_k$ it holds that $d(\sigma, \frac{1}{k} \sum \sigma_i) = \frac{1}{k} \sum d(\sigma, \sigma_i)$. Therefore for every i it holds that

$$\sum_{j=1}^{a(i)} d(c'_A[i,j], w'_B[i,j]) = \sum_{j=1}^{a(i)} d(c_A[i], w'_B[i,j]) = a(i) d(c_A[i], \frac{1}{a(i)} \sum_{j=1}^{a(i)} w'_B[i,j]) = a(i) d(c_A[i], w_B[i]) \ .$$

By summing over all i we will get the first equation of the theorem. The proof of the second equation is the same when Alice is replaced by Bob. □

12 Decoding of One Sided Protocols

In this section we will consider an easier task, we will require that only Alice will output the correct answer. Recall the analysis of the algorithm from the Section 10. Note that essentially it means that in order for Eve to corrupt output of Alice she needs to pick some $i \in [1, n]$ and corrupt at least α_1, β_1 fraction of communication of Alice and Bob before i, such that $\alpha_1 + \beta_1 > 1 - O(\varepsilon)$ and at $n - i$ last rounds Eve needs to corrupt $1/2 - O(\varepsilon)$ fraction of the communication sent by Bob. In the scenario when Alice and Bob speak at the same speed it always optimal for Eve to corrupt Alice's communication before i and half of the communication of Bob after i. Now let us change protocol such that at the beginning Alice speaks more and at the end Bob speaks more. In this case Eve will need spent more error to corrupt Alice's communication at the beginning and more error to corrupt Bob's communication at the end. Essentially we are going to carefully choose functions a, b and our protocol in this section will be $\pi^{a,b}$ where π is the list decodable protocol

from earlier in the paper. Let us consider the following example: let $\alpha = \frac{1}{3}$. Let us now divide our communication into three parts where in each part we will have $\frac{n}{3}$ of blocks of communication. Let us consider the following functions $a(i) = 1$ for every $i \in n$, $b(i) = 1$ for $i = 1 \dots \frac{n}{3}$, $b(i) = 2$ for $i = \frac{n}{3}, \dots \frac{2n}{3}$ and $b(i) = 4$ for $i = \frac{2n}{3}, \dots, n$. Then in the first part of the communication block of communication will be one Bob's and one Alice's message. In the second part in each block Bob will send twice his (the same) message and Alice will send one message; in the last part in each block Bob will send his message four times and Alice will send one message. So in total Alice will send n messages and Bob will send $7n$ messages.

In this scenario suppose we want to pick some i and corrupt almost all (up to $O(\varepsilon n)$) blocks before i and half of the messages of Bob in rounds after i. If we want to corrupt the minimum amount of communication of Bob, we will corrupt his messages in the first $i - \frac{n}{3}$ rounds then Alice's messages in next $\frac{n}{3}$ rounds and half of Bob's messages in the last $n - i$ rounds. It is easy to see that in this case we will corrupt $3n$ Bob's messages. Thus we can decode up-to $\beta < \frac{3}{7}$.

Now let us analyze such protocols for different functions $a(i)$. For every integrable function $f : [0, 1] \to \mathbb{R}^+$ with $\int_0^1 f(t)dt = 1$. The function $f(t)$ will denote the number of messages Bob sends at round tn per every Alice's message. Let us now define for every f:

$$L_1(\alpha, f) = \inf\{\int_0^s f(t)g(t)dt + \frac{1}{2}\int_s^1 f(t)dt : s \in [\alpha, 1], 0 \le g(t) \le 1, \int_0^s g(t)dt \ge s - \alpha\}$$

Remark 53. *We want to mention that once we have picked s, the functions g which minimizes an expression above is 1 on set $S, \mu(S) = s - \alpha$. Here S is the set of form $\{x < s : f(x) < t\} \cup A_t$, where $A_t \subset \{x : f(x) = t\}$ for some t.*

That is, s in the expression above is exactly where correct and non correct codewords are split and $g(t)$ corresponds to the fraction of communication of Bob corrupted at round tn for $t \le s$.

Now let us define non explicitly the function which will be the boundary till which we can perform one sided unique decoding. In the next subsection we will give an explicit formula to this function. Let us set

$$L_1(\alpha) = \sup_f(L_1(\alpha, f) : \int_0^1 f(t)dt = 1, f(t) \ge 0)$$

Thus $L_1(\alpha)$ is the maximal amount of error Bob can handle when Alice has α error and we want Alice to output the correct answer.

The following theorem says that we can to decode with one sided answer if $\beta < L_1(\alpha)$ and that if $\beta > L_1(\alpha)$ we cannot decode. We do not know what happens on the line $\beta = L_1(\alpha)$.

Theorem 54. *For every (α, β) such that $\beta < L_1(\alpha)$. There exists a protocol which is resilient to (α, β) noise and solves the one way pointer jumping problem. The protocol uses $O_{\alpha,\beta}(T)$ communication.*

Proof. Let us set f be a function such that $L_1(\alpha) = L_1(\alpha, f)$ (the next section shows such an f actually exists). Let $\varepsilon = O(L_1(\alpha) - \beta)$ be a small constant.

For any given function $f(t)$ and for any integer n let us define a function

$$b(i) \triangleq \lfloor \frac{n}{\varepsilon} \int_{i/n}^{(i+1)/n} f(t)dt \rfloor \ .$$

Let $a(i) = \frac{1}{\varepsilon}$ for every i. Let π be a list decoding protocol from previous section with code C which is $(\varepsilon, \frac{1}{\varepsilon}+1)$ list decodable and with n rounds. Our protocol will be $\pi^{a,b}$. Let us define c'_A, c'_B, w'_A, w'_B to be messages sent by Alice and Bob and messages received by them. Let c_A, c_B, w_A, w_B be a corresponding messages on which list decoding protocol π was simulated.

We will see in the next section that supremum of L_1 is achieved at the functions with $\sup f(x) < \infty, \inf f(x) > 0$. Let us assume that $\varepsilon < \inf f(x)$. Let $R = \sup f(x)$. Then every round Bob send at least one and at most (constant) R messages. Note also that from Lemma 51 it follows that in protocol $\pi^{a,b}$ Alice sends $\frac{n}{\varepsilon}$ messages and Bob sends $\frac{n}{\varepsilon} - t$ where $t < n$ messages. For the sake of simplicity let us assume that Bob sends $\frac{n}{\varepsilon}$ messages. (say by sending dummy messages at the end.)

Let c_{output} be the closest codeword to w_A, where here we give to symbol i weight $b(i)$ i.e., $c_{output} \in C$ we choose minimizes the expression

$$\sum_{i=1}^{n} b(i) d(c[i], w_A[i]) \ .$$

Let us assume that first place where c_B and c_{output} are different is $s \cdot n$. Let us assume by contradiction that Alice outputs a wrong answer then output node is located after $s \cdot n$.

Then by Lemma 49 we know that $d(c_B[s \cdot n, \ldots, n], c_{output}[s \cdot n, \ldots, n]) \geq (n - s \cdot n) - O(\varepsilon n)$. Therefore

$$\sum_{i=s \cdot n}^{n} b(i) d(c_{output}[i], c_B[i]) \geq \sum_{i=s \cdot n}^{n} b(i) - R \cdot O(\varepsilon n) = \sum_{i=s \cdot n}^{n} b(i) - O(\varepsilon n)$$

Therefore

$$\sum_{i=s \cdot n}^{n} b(i) d(w_A[i], c_B[i]) \geq \frac{1}{2} \sum_{i=s \cdot n}^{n} b(i) - O(\varepsilon n) \geq \frac{n}{\varepsilon} \frac{1}{2} \int_s^1 f(t) dt - O(\varepsilon n) \ .$$

On other hand from Corollary 45 it follows that

$$d(w_B[1, \ldots s \cdot n], c_A[1, \ldots, s \cdot n]) + d(w_A[1, \ldots, s \cdot n], c_B[1, \ldots, s \cdot n]) \geq s \cdot n - O(\varepsilon' n) \ . \tag{3}$$

Note that since $a(i) = \frac{1}{\varepsilon}$ is constant from Lemma 52 it follows that

$$d(w_B[1, \ldots, s \cdot n], c_A[1, \ldots, s \cdot n]) = \varepsilon d(w'_B[1, \ldots, \frac{s \cdot n}{\varepsilon}], c'_A[1, \ldots, \frac{s \cdot n}{\varepsilon}]) \ . \tag{4}$$

We are assuming that at most α fraction of Alice's communication was corrupted thus

$$d(w'_B[1, \ldots, \frac{s \cdot n}{\varepsilon}], c'_A[1, \ldots, \frac{s \cdot n}{\varepsilon}]) \leq \alpha \frac{n}{\varepsilon} \ . \tag{5}$$

Therefore from Equations (3), (4), (5) it follows that,

$$d(w_A[1, \ldots, s \cdot n], c_B[1, \ldots, s \cdot n]) \geq s \cdot n - \alpha n - O(\varepsilon n) \ . \tag{6}$$

Let $s' \cdot n = \sum_{i=1}^{s \cdot n} b(i)$ then from Lemma 52 it follows that

$$d(w'_A[1, \ldots, s' \cdot n], c'_B[1, \ldots, s' \cdot n]) = \sum_{i=1}^{s \cdot n} b(i) d(w_A[i], c_B[i]).$$

Let us define $g(t) = d(w_A[\lfloor tn \rfloor], c_B[\lfloor tn \rfloor])$. Then we have

$$d(w'_A[1,\ldots,s'\cdot n], c'_B[1,\ldots,s'\cdot n]) \geq \frac{n}{\varepsilon}(\int_0^s g(t)f(t)dt - O(\varepsilon)) \ .$$

Thus we have that

$$d(w'_A, c'_B) = \sum_{i=1}^{n} b(i)d(w_A[i], c_B[i]) \geq \frac{n}{\varepsilon}\left(\int_0^s g(t)f(t)dt + \frac{1}{2}\int_s^1 f(t)dt - O(\varepsilon)\right) \ .$$

Note that from Equation (6) it follows that

$$\int_0^s g(t) \geq s - \alpha - O(\varepsilon) \ .$$

Therefore since f is bounded it follows that:

$$d(w'_A, c'_B) \geq L_1(\alpha, f) - O(\varepsilon) \ .$$

By taking ε small enough we will get a contradiction.

□

We also claim that the opposite direction is true:

Theorem 55. *For every α, β with $\beta > L_1(\alpha)$, if a protocol π is resilient to (α, β)-noise, then there exists a constant $O(\frac{1}{\beta - L_1(\alpha)})$ rounds protocol with the same communication complexity in which Alice solves the same task.*

Proof. Let us consider the protocol π. Assume that during the protocol Alice sent n_A symbols and Bob sent n_B symbols. Let $\varepsilon > 0$ be a small constant to be chosen later. Let $T(1)$ be the time when Alice sent her εn_A's symbol, and $T(i)$ be the time when Alice sent her $i\varepsilon n_A$'s symbol. Let $n_1 = T(1) - \varepsilon n_A$ be number of messages Bob sent by time Alice send εn_A of her messages. Let $n_i = T(i) - in\varepsilon$ be number of messages Bob sent in the time Alice sent $i\varepsilon n_A$ of her communication. Let us set $f(x)$ to be equal $\frac{n_i - n_{i-1}}{\varepsilon n_B}$ in the interval $[(i-1)\varepsilon, i\varepsilon]$. Note that $\int_0^1 f(t)dt = 1$ in other words $f(x)$ is an approximation of the rate at which Bob speaks. Thus by definition of $L_1(\alpha)$ there exists $s \in [0,1]$ and $0 \leq g(t) \leq 1$ such that $\int_0^s g(t) \geq s - \alpha$ such that $\int_0^s f(t)g(t)dt + \frac{1}{2}\int_s^1 f(t)dt \leq L_1(\alpha) < \beta$.

Let $t = \lfloor \frac{s-\alpha}{\varepsilon} \rfloor$ Note that from Remark 53 it follows that we can assume w.l.o.g. that $g = 1_S$ where S is union of t of intervals of form $[(i-1)\varepsilon, i\varepsilon]$ plus a part of such an interval. Let us define

$$S = \cup_{k=1}^{t}[(i_k - 1)\varepsilon, i_k\varepsilon] \cup J \subset \cup_{k=1}^{t+1}[(i_k - 1)\varepsilon, i_k\varepsilon] \ ,$$

where $J \subset [(i_{t+1} - 1)\varepsilon, i_{t+1}\varepsilon]$. The minimum value is achieved when we take the intervals with minimal values of $f(x)$. Since $\int_0^1 f(t)dt = 1$ value of f on $[(i_{t+1} - 1)\varepsilon, i_{t+1}\varepsilon]$ is at most $\frac{1}{\alpha - \varepsilon}$. Let us define $S' = \cup_{k=1}^{t+1}[(i_k - 1)\varepsilon, i_k\varepsilon]$. Then for small enough ε it holds that

$$\int_0^s f(t)1_{S'}dt + \frac{1}{2}\int_s^1 f(t)dt \leq \int_0^s f(t)1_S dt + \frac{1}{2}\int_s^1 f(t)dt + O(\varepsilon) \leq L_1(\alpha) + O(\varepsilon) < \beta$$

Let us now assume that Eve has erased communication of Bob on the intervals $[T(i_k-1),\dots,T(i_k)]$ (from the representation of the set S above), and the communication of Alice in all the rest intervals before s. Note that since $\mu(S')\geq s-\alpha$ we have erased at most α fraction of the communication of Alice and at exactly $\int_0^s f(t)1_{S'}$ fraction of communication of Bob. Note that after time s Bob communicates $\int_s^1 f(t)$ communication thus after time s Eve can corrupt at least $\frac{1}{2}$ of communication of Bob, since $\beta-\int_0^s f(t)1_{S'}dt\geq\frac{1}{2}\int_s^1 f(t)dt$

We claim that by time s Alice knows the answer. Therefore we are done since then we have $\frac{2}{\varepsilon}$ round protocol in which Alice solves the task. Assume by contradiction that there exists two inputs to Bob which are consistent with what Alice has saw by time s i.e., there exists two values y_1,y_2 such that for every non corrupted round j it holds that:

$$\pi_B(x,y_1)(w_B[1,\dots,s\cdot n])[j]=\pi_B(x,y_2)(w_B[1,\dots,s\cdot n])[j]$$

Where w_B is the corrupted codeword sent by Alice. Since Eve can corrupt $\frac{1}{2}$ of the communication after round s, Eve will construct a soft codeword which with probability $\frac{1}{2}$ takes value sent by $\pi_B(x,y_1)$ and with probability $\frac{1}{2}$ takes value sent by $\pi_B(x,y_2)$. Therefore we got codeword w_A which Alice could receive for both Bob's inputs y_1,y_2. Therefore, since we assume that Alice output on x,y_1 is different from output on input x,y_2, on one of these inputs Alice outputs the wrong answer.

□

13 Calculating the functions L_1

In this section for every α we want to calculate $\sup_f L_1(\alpha,f)$. At almost no more cost we can calculate more general function which we will need in the next section. For $\alpha<\gamma\leq 1$ let us define

$$L_1(\alpha,\gamma,C)=\{\sup L_1(\alpha,f):\int_0^\gamma f(t)dt\geq C,\int_0^1 f(t)dt=1\}.$$

Note that $L_1(\alpha)=L_1(\alpha,1,1)$.

Theorem 56. *Let us assume that function f satisfies the following properties: Let $F(s)=\int_0^s f(t)dt$.*

1. *$f\geq 0$, $F(1)=1$.*
2. *$F(\gamma)=C$*
3. *Denote by $l=\lfloor\frac{\gamma}{\alpha}\rfloor$. The function f is constant on the interval $[(l-1)\alpha,l\alpha]$.*
4. *The function $T(s)=\frac{1}{2}(1-F(s))+F(s-\alpha)$ is constant in the interval $[\alpha,\gamma]$ and $L_1(\alpha,f)=T(\alpha)$.*

Then $L_1(\alpha,\beta,C)=L_1(\alpha,f)$.

Proof. Let us now assume by contradiction that we have a function h with $L_1(\alpha,h)\geq L_1(\alpha,f)$ and $\int_0^\gamma h(t)dt\geq C$. Define $H(s)=\int_0^s h(t)dt$. Then for every $s\in[\alpha,\gamma]$ let us take $g=1_{[0,s-\alpha]}$ in the definition of the L_1. Then we will get that

$$\frac{1}{2}(1-H(s))+H(s-\alpha)\geq L_1(\alpha,f)=\frac{1}{2}(1-F(s))+F(s-\alpha)=T(\alpha).\tag{7}$$

Claim 57. *For every $i = 1, \ldots, l$ it holds that $H(i\alpha) - H((i-1)\alpha) \leq F(i\alpha) - F((i-1)\alpha)$ in particular $H(i\alpha) \leq F(i\alpha)$.*

Proof. Let us prove this by induction on i. For $i = 1$ this follows from Equation (7) for $s = \alpha$.

For i we will get it from Equation (7) for $s = i\alpha$ and induction assumption on $i - 1$.

$$\frac{1}{2}(1 - H(i\alpha)) + H((i-1)\alpha) \geq \frac{1}{2}(1 - F(i\alpha)) + F((i-1)\alpha)$$

Rewriting the above inequality we get:

$$H(i\alpha) - H((i-1)\alpha) \leq F(i\alpha) - F((i-1)\alpha) - (F((i-1)\alpha) - H((i-1)\alpha)) \ .$$

Since by induction we assume that $F((i-1)\alpha) - H((i-1)\alpha) \geq 0$ the claim follows. □

On other hand by using Equation (7) at $s = \gamma$ we get that from second property of the theorem it follows that:

$$T(\alpha) = \frac{1}{2}(1 - F(\gamma)) + F(\gamma - \alpha) = \frac{1 - C}{2} + F(\gamma - \alpha)$$

Using third property of the theorem we will get that $F(\gamma - \alpha) = F((l-1)\alpha) + \{\frac{\gamma}{\alpha}\}(F(l\alpha) - F((l-1)\alpha))$ thus

$$T(\alpha) = \frac{1 - C}{2} + F((l-1)\alpha) + \left\{\frac{\gamma}{\alpha}\right\}(F(l\alpha) - F((l-1)\alpha)) \ . \tag{8}$$

Now let us take in the definition of L_1 value of $s = \gamma$ and $g(x)$ to be 1 on $[0, (l-1)\alpha]$ and $\left\{\frac{\gamma}{\alpha}\right\}$ on $[(l-1), \alpha, l\alpha]$. Note that $\int_0^\gamma g = \alpha(l-1) + \alpha\left\{\frac{\gamma}{\alpha}\right\} = (l + \left\{\frac{\gamma}{\alpha}\right\})\alpha - \alpha = \gamma - \alpha$. Using this g we will get an equation

$$\frac{1}{2}(1 - H(\gamma)) + H((l-1)\alpha) + \left\{\frac{\gamma}{\alpha}\right\}(H(l\alpha) - H((l-1)\alpha)) \geq T(\alpha)$$

Since $H(\gamma) \geq C$ it follows that

$$H((l-1)\alpha) + \left\{\frac{\gamma}{\alpha}\right\}(H(l\alpha) - H((l-1)\alpha)) \geq T(\alpha) - \frac{1 - C}{2} \ .$$

From Equation 8 it follows that

$$H((l-1)\alpha) + \left\{\frac{\gamma}{\alpha}\right\}(H(l\alpha) - H((l-1)\alpha)) \geq F((l-1)\alpha) + \left\{\frac{\gamma}{\alpha}\right\}(F(l\alpha) - F((l-1)\alpha)) \ .$$

Therefore from Claim 57 and above equation it follows that an equality should hold. Therefore:

$$\begin{aligned} L_1(\alpha, g) \leq \ & \tfrac{1}{2}(1 - H(\gamma)) + H((l-1)\alpha) + \left\{\tfrac{\gamma}{\alpha}\right\}(H(l\alpha) - H((l-1)\alpha)) = \\ & = \tfrac{1}{2}(1 - H(\gamma)) - \tfrac{1-C}{2} + T(\alpha) \leq T(\alpha) = L_1(\alpha, f) \ . \end{aligned}$$

Thus $L_1(\alpha, f) = L_1(\alpha, h)$.

□

Now in order to calculate $L_1(\alpha)$ all we need is to construct the functions that admits all four conditions of Theorem 56 with $C = \gamma = 1$.

Let us define functions f_α as follows: $\tilde{f}_\alpha(x) = 2^i$ in the interval $(i\alpha, \min\{(i+1)\alpha, 1\}]$. Define $N_\alpha = \int_0^1 \tilde{f}_\alpha(t)dt$. Define $f_\alpha(x) = \frac{\tilde{f}_\alpha}{N_\alpha}$. Note that f_α has a property that $f_\alpha(x - \alpha) = \frac{1}{2}f(x)$ for $x \in [\alpha, 1]$. Let us define $F_\alpha(s) = \int_0^s f_\alpha(t)dt$. Properties 1-3 follow immediately. Property 4 follows from the following lemma:

Claim 58. *The function:*

$$T(s) = \frac{1}{2}(1 - F_\alpha(s)) + F_\alpha(s - \alpha)$$

is constant.

Proof. Let us differentiate $T(s)$.

$$T'(s) = f_\alpha(s - \alpha) - \frac{1}{2} f_\alpha(s)$$

But this is zero since $f_\alpha(s - \alpha) = 2f(s)$. □

Thus we have that for every s it holds that $C_\alpha = L_1(\alpha, f_\alpha) = \frac{1}{2}(1 - F_\alpha(s)) + F_\alpha(s - \alpha)$ for every s. We can now easily calculate C_α.

Lemma 59.

$$C_\alpha = L_1(\alpha, f_\alpha) = \frac{1}{2}\left(1 - \frac{1}{(1 + \{\frac{1}{\alpha}\})2^{\lfloor \frac{1}{\alpha} \rfloor} - 1}\right).$$

Proof. We can pick $s = \alpha$. In this case $T(s) = \frac{1}{2}(1 - F_\alpha(\alpha)) = \frac{1}{2}\left(1 - \frac{1}{(1+\{\frac{1}{\alpha}\})2^{\lfloor \frac{1}{\alpha} \rfloor} - 1}\right)$. □

Therefore we have proved the following lemma.

Lemma 60. *For $\alpha = \frac{1}{k}$ for integer k we have $L_1(\alpha) = \frac{1}{2}(1 - \frac{1}{2^k - 1})$. For all other α it is $\frac{1}{2}\left(1 - \frac{1}{(1+\{\frac{1}{\alpha}\})2^{\lfloor \frac{1}{\alpha} \rfloor} - 1}\right)$.*

14 Two sided communication

In this section we would like to answer the question in what is the maximal amount of error we can handle in case we want that both Alice and Bob will return the correct answer. As in previous section we can define the function $f(t)$ of the rate at which Bob speaks with respect to Alice's rate. Let us define

$$L_2(\alpha, f) = \inf\left\{\int_0^s g(t)f(t)dt : 0 \leq g(t) \leq 1, \int_0^s g(t)dt \geq \frac{s - 2\alpha + 1}{2}, s \in [1 - 2\alpha, 1]\right\}.$$

Let us define

$$L_2(\alpha) = \sup_f\{min(L_1(\alpha, f), L_2(\alpha, f)) : \int_0^1 f(t)dt = 1, f(t) \geq 0\}\ .$$

The following theorems have almost the same proof as Theorems 54 and 55.

Theorem 61. *For every (α, β) such that $\beta < L_2(\alpha)$. There exists a protocol π which is resilient to (α, β) noise and which solves pointer jumping problem. The protocol π uses $O_{\alpha,\beta}(T)$ communication.*

Proof. Let us set f to be a function such that $\beta < L_1(\alpha, f)$ and $\beta < L_2(\alpha, f)$. Let $\varepsilon = O(L_2(\alpha) - \beta)$ be a small constant. Define the function

$$b(i) \triangleq \left\lfloor \frac{n}{\varepsilon} \int_{i/n}^{(i+1)/n} f(t)dt \right\rfloor .$$

Let $a(i) = \frac{1}{\varepsilon}$ for every i. Let π be the list decoding protocol from previous section with code C which is $(\varepsilon, \frac{1}{\varepsilon}+1)$ list decodable and with n rounds. Our protocol will be $\pi^{a,b}$. Let us define c'_A, c'_B, w'_A, w'_B to be messages sent by Alice and Bob and messages received by them. Let c_A, c_B, w_A, w_B be a corresponding messages on which the list decoding protocol π was simulated. Alice output the codeword $c \in C$ which minimizes an expression

$$\sum_{i=1}^{n} b(i)d(c[i], w_A[i]) .$$

Bob outputs $c \in C$ which minimizes an expression

$$\sum_{i=1}^{n} a(i)d(c[i], w_B[i]) .$$

Since $a(i)$ is constant this is the same as to just output closest codeword to w_B. The output of Alice is correct by Theorem 55. So now we need to prove that output of Bob is correct. Let c_{output} the output of the Bob. Let $s \cdot n$ be a first place where c_{output} is different from c_A; then we know from Lemma 49 that

$$d(c_{output}[s, \ldots, n], w_B[s, \ldots, n]) \geq \frac{n - s \cdot n}{2} - O(\varepsilon n) .$$

Let $s' = \sum_{i=1}^{s \cdot n} b(i)$ then from Lemma 52 it follows that

$$L_2(\alpha, f) > \beta \geq d(w'_A[1, \ldots s'], c'_B[1, \ldots, s']) = \sum_{i=1}^{s \cdot n} b(i)d(w_A[i], c_B[i]).$$

Let us define $g(t) = d(w_A[\lfloor tn \rfloor], c_B[\lfloor tn \rfloor])$. By taking ε small enough we have

$$\int_0^s g(t)f(t)dt < L_2(\alpha, f) .$$

Thus by definition of L_2 we get that

$$\int_0^s g(t)dt < \frac{s - 2\alpha + 1}{2} .$$

Note that the following holds

$$n \int_0^s g(t)dt = d(w_A[1, \ldots, s \cdot n], c_B[1, \ldots, s \cdot n]) .$$

From Corollary 45 it follows that

$$d(w_B[1, \ldots s \cdot n], c_A[1, \ldots, s \cdot n]) + d(w_A[1, \ldots, s \cdot n], c_B[1, \ldots, s \cdot n]) \geq s \cdot n - O(\varepsilon n) .$$

Thus we have that for small enough ε

$$d(w_B[1,\dots s\cdot n], c_A[1,\dots,s\cdot n]) > \frac{s\cdot n + 2\alpha n - n}{2}\ .$$

Thus total error we have is

$$d(w_B[1,\dots n], c_A[1,\dots,n]) > \frac{n - s\cdot n}{2} + \frac{s\cdot n + 2\alpha n - n}{2} > \alpha n.$$

Contradiction to the assumption that there is at most α fraction of the communication is corrupted □

Theorem 62. *For every α, β with $\beta > L_2(\alpha)$, if protocol π resilient to (α, β) noise. Then there exist constant $c(\alpha, \beta)$ rounds protocol with the same communication complexity in which Alice or Bob solves the task.*

Proof. Let $\alpha' < \alpha$ be a number such that $\beta > L_2(\alpha')$. Let us take $\varepsilon = O(\min\{\alpha - \alpha', \beta - L_2(\alpha)\})$.

Let us consider protocol π. Let as assume that during the protocol Alice sent n_A symbols and Bob sent n_B symbols. Let $T(1)$ be the time when Alice sent her εn_A's symbol, and $T(i)$ be the time when Alice sent her $i\varepsilon n_A$'s symbol. Let $n_1 = T(1) - \varepsilon n_A$ be the number of messages Bob sent by the time Alice sent εn_A of her messages. Let $n_i = T(i) - i n_A \varepsilon$ be the number of messages Bob sent in the time when Alice sent $i\varepsilon n_A$ of her communication. Let us set $f(x)$ to be equal $\frac{n_i - n_{i-1}}{\varepsilon n_B}$ in the interval $[(i-1)\varepsilon, i\varepsilon]$. Note that $\int_0^1 f(t)dt = 1$. In other words $f(x)$ is an approximation of the rate at which Bob speaks.

By definition of $L_2(\alpha')$ we have that $\beta > L_1(\alpha', f)$ or $\beta > L_2(\alpha', f)$. If $\beta > L_1(\alpha', f)$ then from Theorem 55 we will get that there exists a constant round protocol in which Alice solves π. Thus let us assume that $\beta > L_2(\alpha', f)$.

Thus by definition of $L_2(\alpha')$ there exists $s \in [0,1]$ and $0 \le g(t) \le 1$ such that $\int_0^s g(t)dt \ge \frac{s - 2\alpha' + 1}{2}$ such that $\int_0^s f(t)g(t)dt < \beta$.

Let $t = \lfloor \frac{s - 2\alpha' + 1}{2\varepsilon} \rfloor$ We can assume w.l.o.g. that $g = 1_{S'}$ where S' is union of t of intervals of form $[(i-1)\varepsilon, i\varepsilon]$ plus part of such interval. Let us define

$$S = \cup_{k=1}^{t} [(i_k - 1)\varepsilon, i_k\varepsilon]\ .$$

Then it holds that

$$\int_0^s f(t) 1_S dt < \beta$$

Let us now assume that Eve has erased communication of Bob on the intervals $[T(i_k - 1), T(i_k)]$ (from the definition of S) and the communication of Alice in all the rest intervals before s. Note that since $\mu(S) \ge \frac{s - 2\alpha' + 1}{2} - \varepsilon$ we have erased at most $s - (\frac{s - 2\alpha' + 1}{2} - \varepsilon) \le \frac{s + 2\alpha - 1}{2}$ fraction of the communication of Alice before s and at most β fraction of communication of Bob. Note that after time s, Alice communicates $1 - s$ fraction of her communication. Thus after time s Eve can corrupt at least $\alpha - \frac{s + 2\alpha - 1}{2} = \frac{1 - s}{2}$. In other words Eve can corrupt $\frac{1}{2}$ of communication of Alice, after time s.

We claim that by time s Bob knows the answer. Therefore we are done since then we now have a $\frac{2}{\varepsilon}$ round protocol in which Bob solves the task. The proof of this fact is the same as in Theorem 55. □

15 Calculating the Function $L_2(\alpha)$

Assume that $\alpha \leq \frac{1}{3}$. Here we are going to use the following function. $\tilde{f}_\alpha(x) = 2^i$ in the interval $(i\alpha, \min\{(i+1)\alpha, 1-2\alpha\}]$ and in the interval $[1-2\alpha, 1]$ it will be $(1+\{\frac{1}{\alpha}\})2^{\lfloor\frac{1}{\alpha}\rfloor-2}$. Then we will define f_α to be normalization of $\tilde{f}_\alpha$. Let us also define $F_\alpha(s) = \int_0^s f_\alpha(t)dt$. The hardest part of this subsection is the following lemma

Lemma 63.

$$L_2(\alpha, f_\alpha) = L_1(\alpha, f_\alpha) = \frac{1}{2}\left(1 - \frac{1}{(1+\{\frac{1}{\alpha}\})2^{\lfloor\frac{1}{\alpha}\rfloor-1} - 1}\right) .$$

In particular we will get that $L_1(1/k) = L_2(1/k) = \frac{1}{2}(1 - \frac{1}{2^{k-1}-1})$.

Proof. First let us calculate $L_1(\alpha, f_\alpha)$. To do so we need to calculate for every s the value $\min\{\int_0^s f(t)g(t)dt + \frac{1}{2}(1 - F_\alpha(s)) : \int_0^s g(t)dt \geq s - \alpha\}$. For $s \leq 1 - \alpha(1 + \{\frac{1}{\alpha}\})\}$ the function which minimizes this expression is $g = 1_{[0,s-\alpha]}$. Thus for $s < 1 - 2\alpha$ this expression is equal to $\frac{1}{2}(1 - F_\alpha(s)) + F(s - \alpha)$ Consider the function:

$$T(s) = \frac{1}{2}(1 - F_\alpha(s)) + F_\alpha(s - \alpha) .$$

Note that $T'(s) = f(s-\alpha) - \frac{1}{2}f_\alpha(s) \geq 0$. Thus $T(s)$ is non-decreasing. We have that $T(\alpha) \leq T(s)$ for every $s \geq \alpha$. Thus for $s \leq 1 - \alpha(1 + \{\frac{1}{\alpha}\})$ we can get at least $T(\alpha)$. For $s > 1 - \alpha(1 + \{\frac{1}{\alpha}\})$ the optimal g is 1 on interval $[0, \ldots 1 - \alpha(2 + \{\frac{1}{\alpha}\})]$ and 1 on interval $[1 - \alpha(1 + \{\frac{1}{\alpha}\}), s]$ Therefore value of L_1 for $s > 1 - \alpha(1 + \{\frac{1}{\alpha}\})$ is

$$T(s) = F_\alpha(1 - \alpha(2 + \{\frac{1}{\alpha}\})) + F_\alpha(s) - F_\alpha(1 - \alpha(1 + \{\frac{1}{\alpha}\})) + \frac{1 - F(s)}{2} .$$

Again $T'(s) = \frac{1}{2}f(s) \geq 0$. Thus minimum is achieved at $s = 1 - \alpha(1 + \{\frac{1}{\alpha}\})$. Direct calculation shows that $T(\alpha) < T(1 - \alpha(1 + \{\frac{1}{\alpha}\}))$. Therefore

$$L_1(\alpha, f_\alpha) = \frac{1}{2}(1 - F_\alpha(\alpha)) = \frac{1}{2}\left(1 - \frac{1}{(1+\{\frac{1}{\alpha}\})2^{\lfloor\frac{1}{\alpha}\rfloor-1} - 1}\right) .$$

To calculate $L_2(\alpha, f_\alpha)$ note that for $s \geq 1 - 2\alpha$ an optimal g is 1 on $[0, 1 - \alpha(2 + \{\frac{1}{\alpha}\})]$ and 1 on $[\frac{s-2\alpha(1+\{\frac{1}{\alpha}\})+1}{2}, s]$. Thus

$$T_2(s) = F_\alpha(1 - \alpha(2 + \{\frac{1}{\alpha}\})) + F_\alpha(s) - F_\alpha(\frac{s - 2\alpha(1 + \{\frac{1}{\alpha}\}) + 1}{2}) .$$

Again differentiate T_2 we get that $T_2'(s) = f(s) - \frac{1}{2}f(\frac{s-2\alpha(1+\{\frac{1}{\alpha}\})+1}{2})$. It is easy to see that $T'(s) > 0$ for $s > 1 - 2\alpha$. Thus minimal is achieved when $s = 1 - 2\alpha$ Thus

$$L_2(\alpha, f_\alpha) = F_\alpha(1 - 2\alpha) = \frac{1}{2}(1 - F_\alpha(\alpha)) = \frac{1}{2}\left(1 - \frac{1}{(1+\{\frac{1}{\alpha}\})2^{\lfloor\frac{1}{\alpha}\rfloor-1} - 1}\right) .$$

□

Theorem 64.

$$L_2(\alpha) = L_1(\alpha, f_\alpha) = L_2(\alpha, f_\alpha) \ .$$

Proof. From Lemma 63 we know that $L_1(\alpha, f_\alpha) = L_2(\alpha, f_\alpha)$ let us denote this value by C. Let us assume that we have function g which is better than f_α i.e., $\int_0^1 g(t)dt = 1$ and

$$\min\{L_1(\alpha, g), L_2(\alpha, g)\} \geq \min\{L_1(\alpha, f_\alpha), L_2(\alpha, f_\alpha)\} = L_1(\alpha, f_\alpha) = L_2(\alpha, f_\alpha) = C$$

Denote by $G(s) = \int_0^s g(t)dt$. Then by taking $s = 1-2\alpha$ in definition of L_2 we have that $G(1-2\alpha) \geq L_2(\alpha, g) \geq F_\alpha(1-2\alpha) = C$. Thus we have that $L_1(\alpha, f) \leq L_1(\alpha, g) \leq L_1(\alpha, 1-2\alpha, C)$. Now to complete the proof we need to show that $L_1(\alpha, f_\alpha) = L_1(\alpha, 1-2\alpha, C)$. This follows from Theorem 56. The conditions 1-3 follows immediately. To see the condition 4 note that for $s \in [\alpha, \beta]$ it holds that $f_\alpha(s) = 2f_\alpha(s-\alpha)$. □

16 Proof of Theorems 4 and 5

Since from Theorems 61, 62 it follows that L_2 is the boundary of (α, β) we can handle L_2 must be symmetric in α, β i.e., if $L_2(\alpha) = \beta$ then $L_2(\beta) = \alpha$. Also note that L_2 is decreasing function and that $L_2(\frac{1}{3}) = \frac{1}{3}$. Therefore if $\beta = L_2(\alpha)$ then $\alpha \leq \frac{1}{3}$ or $\beta \leq \frac{1}{3}$. Thus the region $\mathcal{R}_u$ can be defined by

$$\mathcal{R}_U = \{(\alpha, \beta) : \beta < L_2(\alpha)\} \ .$$

Theorem 65. *For each $(\alpha, \beta) \in \mathcal{R}_U$ and for every protocol π there exists another protocol π', with $CC(\pi') = O_{\alpha,\beta}(CC(\pi))$ which is resilient (α, β) adversarial noise. Such that $\pi'(x, y)$ outputs transcript of $\pi(x, y)$.*

Proof. This follows from the Theorem 61 and the fact that Pointer Jumping Problem is complete for communication complexity. □

Theorem 66. *For every $(\alpha, \beta) \notin \overline{\mathcal{R}}_U$ and for every T the following holds. Let π' be a protocol resilient to (α, β) adversarial noise which solves the Pointer Jumping Problem of depth T. Then $CC(\pi') = 2^{\Omega_{\alpha,\beta}(T)}$.*

Proof. From Theorem 62 it follows that there exists constant $O_{\alpha,\beta}(1)$ round protocol π'' with the same communication complexity which solves one way pointer jumping problem. Then from Corollary 9 it follows that communication complexity of π'' is $2^{\Omega_{\alpha,\beta}(T)}$. □

17 Conclusions

In this paper we completely answered the question of characterizing the maximal error rate we can handle in adversarial interactive communication with a constant-rate encoding. We have defined list decoding of interactive communication and showed that this primitive is also important for unique decoding of interactive communication. Our encodings are constant rate, i.e. only incur constant blow-up in communication. However we have not tried to optimize this constant, leaving such optimization as an open problem. Below we highlight this and additional outstanding open problems:

Open Problem 1. *The big problem is to understand what is the best rate is possible for interactive communication. We want to note that this question is widely open even in the random noise scenario. Recently there was some progress on this question in random noise scenario with noise rate approaching* 0 *by Kol and Raz [KR13].*

Open Problem 2. *In this paper we made an encoding over a channel with a large constant-size alphabet. Without much effort one can modify all our protocols to work over a binary channel with a loss of a factor two in the error rates one can handle. However it is not clear at all that these error rates are best possible. Establishing the region of* (α, β) *for which unique decoding is possible over binary alphabet is an open problem.*

Open Problem 3. *Construct explicit list tree codes with computationally efficient encoding and decoding. It is plausible that construction list-decodable tree codes will be an easier problem than just a regular tree codes since, similarly to potent tree codes from [GMS11], a random prefix code is a list decodable tree code with high probability, while it is not a tree code with high probability.*

References

[AGS13] Shweta Agrawal, Ran Gelles, and Amit Sahai. Adaptive protocols for interactive communication. *arXiv preprint arXiv:1312.4182*, 2013.

[BK12] Zvika Brakerski and Yael Tauman Kalai. Efficient interactive coding against adversarial noise. In *Foundations of Computer Science (FOCS), 2012 IEEE 53rd Annual Symposium on*, pages 160–166. IEEE, 2012.

[BN13] Zvika Brakerski and Moni Naor. Fast algorithms for interactive coding. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 20, page 14, 2013.

[BR11] Mark Braverman and Anup Rao. Towards coding for maximum errors in interactive communication. In *Proceedings of the 43rd annual ACM symposium on Theory of computing*, pages 159–166. ACM, 2011.

[Bra12] Mark Braverman. Towards deterministic tree code constructions. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, pages 161–167. ACM, 2012.

[Eli57] Peter Elias. List decoding for noisy channels. 1957.

[FGOS12] Mattew Franklin, Ran Gelles, Rafail Ostrovsky, and Leonard J Schulman. Optimal coding for streaming authentication and interactive communication. In *Electronic Colloquium on Computational Complexity (ECCC)*, volume 19, page 104, 2012.

[GH13] Mohsen Ghaffari and Bernhard Haeupler. Optimal error rates for interactive coding ii: Efficiency and list decoding. *arXiv preprint arXiv:1312.1763*, 2013.

[GHS13] Mohsen Ghaffari, Bernhard Haeupler, and Madhu Sudan. Optimal error rates for interactive coding i: Adaptivity and other settings. *arXiv preprint arXiv:1312.1764*, 2013.

[GMS11] Ran Gelles, Ankur Moitra, and Amit Sahai. Efficient and explicit coding for interactive communication. In Rafail Ostrovsky, editor, *FOCS*, pages 768–777. IEEE, 2011.

[Gur04] Venkatesan Guruswami. *List decoding of error-correcting codes*. Springer, 2004.

[KR13] Gillat Kol and Ran Raz. Interactive channel capacity. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *STOC*, pages 715–724. ACM, 2013.

[NW93] Noam Nisan and Avi Wigderson. Rounds in communication complexity revisited. *SIAM J. Comput.*, 22(1):211–219, February 1993.

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

[Woz58] John M Wozencraft. List decoding. *Quarterly Progress Report*, 48:90–95, 1958.

ECCC ISSN 1433-8092

http://eccc.hpi-web.de